ABSTRACT

Title of Dissertation: ROBUST SPEAKER RECOGNITION BASED ON LATENT VARIABLE MODELS

Daniel Garcia-Romero, Doctor of Philosophy, 2012

Directed by: Professor Carol Y. Espy-Wilson
Department of Electrical and Computer Engineering

Automatic speaker recognition in uncontrolled environments is a very challenging task due to channel distortions, additive noise and reverberation. To address these issues, this thesis studies probabilistic latent variable models of short-term spectral information that leverage large amounts of data to achieve robustness in challenging conditions.

Current speaker recognition systems represent an entire speech utterance as a single point in a high-dimensional space. This representation is known as "supervector". This thesis starts by analyzing the properties of this representation. A novel visualization procedure of supervectors is presented by which qualitative insight about the information being captured is obtained. We then propose the use of an overcomplete dictionary to explicitly decompose a supervector into a speaker-specific component and an undesired variability component. An algorithm to learn the dictionary from a large collection of data is discussed and analyzed. A subset of the entries of the dictionary is learned to represent speaker-specific information and

another subset to represent distortions. After encoding the supervector as a linear combination of the dictionary entries, the undesired variability is removed by discarding the contribution of the distortion components. This paradigm is closely related to the previously proposed paradigm of Joint Factor Analysis modeling of supervectors. We establish a connection between the two approaches and show how our proposed method provides improvements in terms of computation and recognition accuracy.

An alternative way to handle undesired variability in supervector representations is to first project them into a lower dimensional space and then to model them in the reduced subspace. This low-dimensional projection is known as “i-vector”. Unfortunately, i-vectors exhibit non-Gaussian behavior, and direct statistical modeling requires the use of heavy-tailed distributions for optimal performance. These approaches lack closed-form solutions, and therefore are hard to analyze. Moreover, they do not scale well to large datasets. Instead of directly modeling i-vectors, we propose to first apply a non-linear transformation and then use a linear-Gaussian model. We present two alternative transformations and show experimentally that the transformed i-vectors can be optimally modeled by a simple linear-Gaussian model (factor analysis). We evaluate our method on a benchmark dataset with a large amount of channel variability and show that the results compare favorably against the competitors. Also, our approach has closed-form solutions and scales gracefully to large datasets.

Finally, a multi-classifier architecture trained on a multicondition fashion is proposed to address the problem of speaker recognition in the presence of additive

noise. A large number of experiments are conducted to analyze the proposed architecture and to obtain guidelines for optimal performance in noisy environments. Overall, it is shown that multicondition training of multi-classifier architectures not only produces great robustness in the anticipated conditions, but also generalizes well to unseen conditions.

ROBUST SPEAKER RECOGNITION BASED ON LATENT VARIABLE MODELS

by

Daniel Garcia-Romero

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2012

Advisory Committee:
Professor Carol Y. Espy-Wilson, Chair/Advisor
Professor K. J. Ray Liu
Professor Min Wu
Professor William J. Idsardi
Professor Ramani Duraiswami

To Audrey

Acknowledgements

First, I thank my advisor Professor Carol Y. Espy-Wilson for her guidance during my PhD. She has always been very supportive and has given me a lot of freedom to explore different ideas. When challenges were presented, she always provided words of encouragement and a positive attitude. From her I have learned the importance of pursuing the things that you are passionate about.

Next, I thank Professors Ray Liu, Min Wu, William Idsardi, and Ramani Duraiswami for serving as committee members for this dissertation. Outside of this dissertation, Professor Ray Liu has taught me a lot —inside and outside of the class. I was very fortunate to be able to take all his signal processing classes. He has always given me good advice and treated me as a friend. For that, I am very grateful. Also, I thank Professor Min Wu for her encouragement and willingness to discuss ideas.

I also thank the faculty of the Electrical Engineering Department from whom I have taken great courses and learned very good fundamentals. In particular, I thank Professor P. S. Krishnaprasad for allowing me to attend his group meetings and to pursue independent studies with him.

During my thesis I have had the opportunity to work closely with Dr. Xinhui Zhou. I admire his dedication and intellectual curiosity. I have greatly benefited from our conversations and endless questions. I am very thankful for our friendship.

I also thank all the present and past members of the Speech Communication Lab for their friendship and intellectually stimulating discussions.

In the past two years I have had the privilege to collaborate with Professor Hynek Hermanski. I have greatly benefited from his historical perspective of speech processing and his creativity.

Finally, I thank my wife Audrey for her love, care, support, and patience during my PhD. She is an invaluable source of moral support and joy in my life.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Automatic speaker recognition is concerned with designing algorithms that infer the identity of people by their voices. This is a very challenging task since the speech signals are highly variable. The sources of variability can be classified in two types: intrinsic and extrinsic. When interested in making inferences about identity, intrinsic sources of variability include: the linguistic message, language, vocal effort, speaking-style, emotional and health state. Extrinsic sources are the channel distortions introduced by acquisition devices (e.g., telephones), and environmental distortions like additive noise and room reverberation.

In order to design systems that are able to cope with such sources of variability in a wide number of domains, at least three key questions need to be addressed: i) how to train statistical models that leverage large amounts of data and are efficiently adapted to scenarios with limited amounts of data; ii) how to capture and represent diverse speaker-specific information that provides complementary robustness to different sources of variability; iii) how to adaptively select the optimal available representation for the condition at hand.

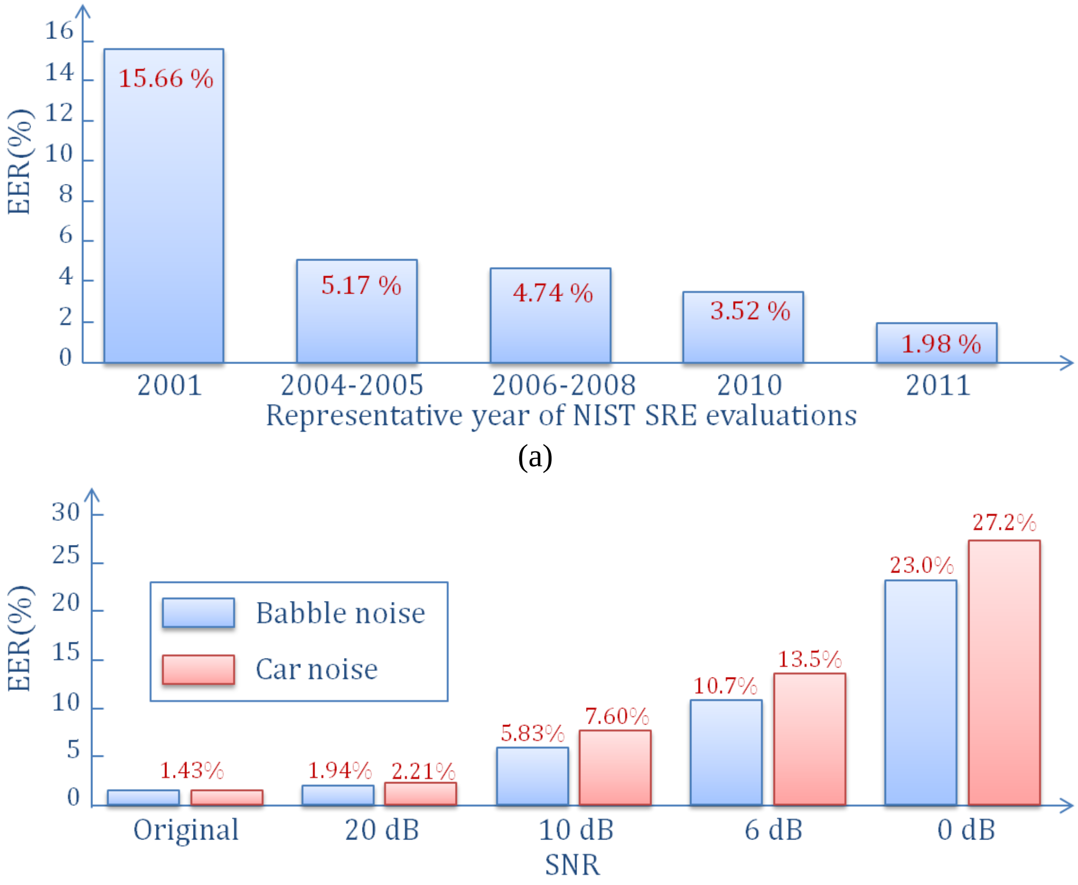
To partially address the first question—and mostly due to the emphasis placed by the NIST speaker recognition evaluations [1]—the main focus of the speaker recognition community in the past decade has been on coping with channel mismatch between speech samples. In particular, recent advances in speaker recognition are not necessarily due to new or better understanding of speaker characteristics that are informative or interpretable by humans; rather, they are the result of improvements in machine learning techniques that leverage large amounts of data.

Following this trend, in this thesis we focus on the first and third questions mentioned above. Specifically, we advance the state-of-the-art in speaker recognition systems based on probabilistic latent variable models of short-term spectral information that leverage large amounts of data. By doing so, we are able to obtain significant robustness to channel mismatch as well as additive noise.

Before continuing with a more detailed exposition of the organization of this thesis, the next section motivates this work by way of an example.

1.1. Motivation

Since 1996, the National Institute of Standards and Technology (NIST) has organized yearly evaluations of automatic speaker recognition systems [1]. This has provided a benchmark by which the technological improvements can be objectively assessed. The top panel of Figure 1.1 shows how state-of-the-art speaker verification systems—representative of the years indicated in the horizontal axis—would perform on the latest NIST evaluation data of 2010 (data recorded by both landlines and cell phones) [1]. The results are presented in terms of Equal Error Rate (EER) which corresponds to the value in which the probability of miss detection equals the probability of false acceptance. Notice that, according to this dataset, an 8-fold improvement has occurred within 10 years (from around 16% EER of a system from 2001 to the 2% of a system from 2011).



(a)

(b)

Figure 1.1: (a) Performance of the representative state-of-the-art technologies of the years in the horizontal axis on telephone data from the latest NIST 2010 evaluation[1]. (b) Performance degradation in terms of Equal Error Rate (EER) of a state-of-the-art speaker verification system as a function of SNR of the test data for babble and car noises.

[1] The numbers of Figure 1.1 (a) were provided by Brno University of Technology.

The numbers in the top panel provide a context for the results shown in the lower part of Figure 1.1. In particular, the lower panel shows how the performance of a state-of-the-art system (representative of 2011) decreases as function of the signal to noise ratio (SNR) of the test data (for babble and car noises). A system that produces a 2% EER in a 20dB scenario performs at a rate of around 14% for a 6 dB SNR. In other words, a 6 dB SNR produces a performance degradation equivalent to the improvements obtained over 10 years of research. This drastic decrease in performance illustrates the need for robust mechanisms and motivates the work of this thesis.

1.2. Dissertation Outline

The goal of this thesis is to improve the robustness of automatic speaker recognition systems so that they can be deployed in challenging scenarios in which channel distortions, additive noise, and reverberation are present. Specifically, we aim at advancing the state-of-the-art in speaker recognition systems—based on probabilistic generative models of short-term spectral information—that leverage large amounts of data.

The field of automatic speaker recognition is approximately 50 years old; with some of the earliest work dating back to the 1960s. A large amount of research has been conducted since then and great technological advances have been accomplished. For this reason, Chapter 2 presents a brief summary of the basic concepts in the field to provide a context for the work presented in this thesis.

A common theme among current speaker recognition systems based on short-time spectral information is the representation of a speech utterance as a single point in a high-dimensional space. This representation is denoted as “supervector” and all the systems studied in this thesis make us of it. Chapter 3 is dedicated to gaining a better understanding about the nature of this representation. A novel visualization procedure of supervectors is presented by which qualitative insight about the information being captured can be obtained. Based on this visualization approach, the Switchboard-I database (SWB-I) is used to establish a relationship between a data-driven partition of the acoustic space and a knowledge based partition in terms of broad phonetic classes.

The supervector formalism presented in Chapter 3 provides a mechanism to obtain a fixed-length representation of a variable length object. However, the direct use of this representation in a speaker recognition system is not optimal; since supervectors not only capture speaker-specific information but also contain a large amount of undesired variability entangled with the desired information. Hence, there is a need for a mechanism to disentangle the speaker-specific information and the undesired variability captured in the supervector representations. This is the objective of the work presented in Chapters 4, 5, and 6. The three chapters make use of probabilistic generative models with latent variables.

The use of speaker recognition systems based on supervector representations modeled by Joint Factor Analysis (JFA) advanced the state-of-the-art significantly from 2004 until 2008. The main goal of Chapter 4 is to provide a connection between the JFA paradigm and the use of signal coding in overcomplete dictionaries learned from data. Establishing this connection allowed for cross-pollination between fields and resulted in two algorithmic improvements over the baseline JFA system. One improvement came in the form of improved computation, whereas the other came in terms of improved recognition accuracy.

A significant breakthrough occurred around 2010 by using a Factor Analysis model of supervectors as an unsupervised dimensionality reduction technique [2], [3]. The computed factors were denoted as "i-vectors" and explicit modeling of speaker-specific and inter-session variability was performed in this lower-dimensional space. However, i-vectors were shown to exhibit non-Gaussian behavior and complex non-Gaussian generative models were needed for optimal performance [4]. As an alternative, Chapter 5 proposes the use of two different non-linear transformations of i-vectors to reduce their non-Gaussian behavior. After applying either one of these transformations, i-vectors can be successfully modeled by a simple linear-Gaussian model. The proposed transformations are shown to be extremely effective and produce the same or even better performance as the more complex alternatives (Heavy-tailed models based on Student's *t* distributions) while maintaining the simplicity and high scalability of the linear-Gaussian models. Results are presented on data from the latest NIST 2010 speaker recognition evaluation. The performance obtained for conditions with a high degree of channel variability is state-of-the-art.

Also working with i-vectors, Chapter 6 explores noise robustness. A novel multi-classifier architecture trained on a multicondition fashion is proposed to address the problem of speaker recognition in the presence of additive noise. A large number of experiments are conducted to analyze the proposed architecture, and to obtain guidelines for optimal performance in noisy environments. Overall, it is shown that multicondition training of multi-classifier architectures not only produces great robustness in the anticipated conditions, but also generalizes well to unseen conditions during training. The latest NIST 2010 evaluation data is used to validate these results.

Finally, Chapter 7 summarizes the contributions of this thesis and discusses future perspectives.

Chapter 2

Speaker Recognition: A Review

The early research on speaker recognition was almost entirely limited to human listening; and it was mostly motivated by the desire to produce natural sounding speech from speech codecs [5]. Although the synthetic speech generated by the vocoders was quite intelligible, it was only partially successful in carrying the speaker-specific information necessary to easily identify the speakers. This problem motivated some initial research about the factors that carry speaker-specific information in the speech signal [6].

In the midst of these studies, in the early 1960s, the influential (and highly controversial[2]) work of Lawrence Kersta about visual spectrographic voice identification was published [7]. The results of this work, the availability of digital computers, and the curiosity to see if machines could duplicate human performance,

[2] The use of the term "voiceprint", in an attempt to equate spectrograms to the characteristic patterns of human fingerprints, created false expectations about the reliability of visual inspection of spectrograms. Numerous criticisms have been presented with respect to the term "voiceprint" since it ignores the behavioral nature of the speech signals (see [94], [89] and references therein for more details).

motivated one of the first attempts at automatic speaker recognition by Pruzansky [8]. In this pioneering work, a long-term average spectrum feature vector was computed using a filterbank. Then, a similarity score was obtained by a simple Euclidean distance. Improvements upon this early work came in the form of: modified representations of spectral information [9]; alternative sources of speaker information (prosody) [10], better modeling of the temporal dynamics [11], and improved statistical modeling [12].

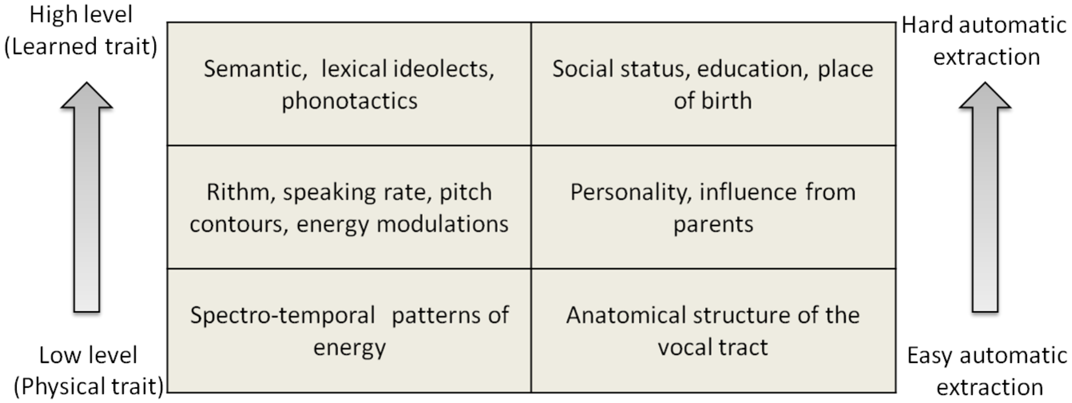
According to the historical review of Furui [13], the first fully automated large-scale (hundreds of speakers) speaker verification system with a high operational performance was developed by Texas Instrument. Since then, the field of automatic speaker recognition has attracted a lot of attention and significant progress has been made both in the way the speaker-information is captured as well as the statistical modeling techniques. A large number of reviews/tutorials have been published over the years. Two of the most recent ones are [14] and [13]. Also, less recent, but still quite instructive, are the classical reviews of Campbell [15] and Atal [5].

In the following, we present a succinct exposition of some basic concepts necessary to contextualize the work presented in this thesis (referring the reader to the abovementioned reviews for details). First we describe important sources of speaker-specific information in the speech signals. Then we provide some technical definitions and applications of speaker recognition systems. This is followed by an overview of the basic constituent elements of a generic speaker recognition system. Moreover, the classical paradigm of speaker recognition based on Gaussian Mixture

Models is reviewed. Finally, we survey some of the most common techniques used to add robustness to speaker recognition systems.

2.1. Speaker Specific Information in the Speech Signal

The speech signal is produced by the interaction of three mechanisms: the lungs, the vocal folds in the larynx, and the articulators. The lungs produce the airflow that is modulated by the vibration of the vocal folds in the larynx. The resulting acoustic signal is further transformed by the complex orchestration of the articulators—configurable elements of the voice production mechanism such as the tongue, jaw, soft-palate and lips. Changes in the way the vocal fold vibrate (including no vibration), and the vocal tract shape resulting from the configuration of the articulators are reflected on the acoustical properties of the signal. Many outstanding reviews exist about speech physiology (for example, [16]). Here we will focus on those aspects that are particularly relevant to the identity of the speaker.



Semantic, lexical ideolects, phonotactics	Social status, education, place of birth
Rithm, speaking rate, pitch contours, energy modulations	Personality, influence from parents
Spectro-temporal patterns of energy	Anatomical structure of the vocal tract

Figure 2.1: Hierarchy of speaker-specific information and associated determinant factors.

The speech signal conveys information about the physical, psychological and social characteristics of the speaker [17]. This information is present at different levels. Figure 2.1 shows a possible hierarchical classification of these sources of information as well as some associated factors.

Human listeners use these sources of information in a natural way to discriminate among speakers [18]. The idiosyncratic combination of these sources (e.g., low pitch, peculiar timbre, unique laughter, word choice, etc) facilitates an accurate identification. It is the unique relationship between these features that characterizes an individual's voice. Also, for human listeners, there is a big different in the way identification is carried out depending on the familiarity of the listener with the speaker (e.g., parents, spouse, children, etc) [18]. However, this distinction is not currently applicable to automatic speaker recognition systems. Nonetheless, the way automatic speaker recognition is carried out is consistent with the theory presented, in Chapter 6 of [18], about how humans discriminate between unfamiliar voices.

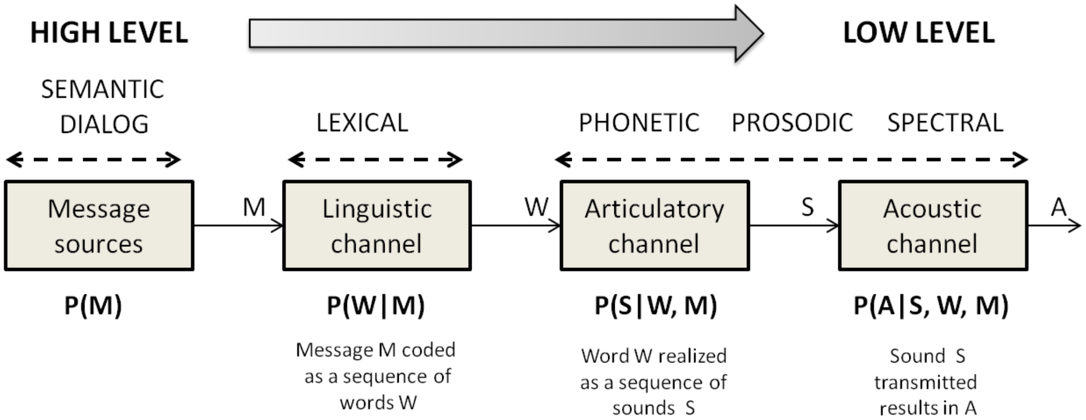


Figure 2.2: Information-theoretic model of speech production. (Adapted from [19]).

The underlying factors conditioning each of these sources in Figure 2.1 are very diverse. From a hierarchical perspective, at the lowest level, the physical characteristics of the individual, as well as the anatomical characteristics of the vocal tract, are reflected on the spectro-temporal composition of the signal. At the highest level, the habits and customs learned over a long period of time are the primal factors in the selection of words and semantic structures to convey a message.

Nowadays, one of the most successful frameworks for speech recognition is based on the formulation of the speech production chain in terms of an information-theoretic model [20]. This perspective provides a very useful conceptual framework that has also permeated to the area of automatic speaker recognition [21]. From this framework, Figure 2.2 shows the constituent stages of the speech production chain along with a hierarchy of the related levels of speaker-specific information. There are two main types of processes involved in this chain. On the one hand, there are psychological processes related to the higher levels of information. On the other hand, the lower levels of the hierarchy are associated with physiological processes. The high-complexity and elevated degree of abstraction that characterizes the psychological processes provides a partial explanation about the difficulty involved in the automatic extraction of the associated sources of speaker-specific information.

Analyzing Figure 2.2 in detail we can observe that the starting from an intended message M, the speaker selects a sequence of words W (modeled by the linguistic channel). At this level of abstraction, there are potential sources of speaker-specific information such as the particular tendencies to convey meaning as well as the conversational patterns of an individual [22]. Moreover, with respect to the linguistic channel, the particular word selection to convey a given message is also a potential source of information. Therefore, at the lexical level, the patterns of word usage of an individual speaker project its identity on the signal [23].

Following the linguistic channel, the articulatory channel transforms a discrete sequence of words into a continuous speech signal S in accordance with a set of phonological rules [24]. This stage is very rich in speaker-specific information [19]. The distinctive characteristics introduced at this stage belong to the levels of phonetic, prosodic and spectral information. The sounds produced in this stage are the results of physiological activities involving the interaction of the nervous system and the muscles. The orchestrated movements of the articulators transform the airflow to generate the acoustic signal S that passes through the acoustic channel to produce the measured speech signal A. This acoustic channel models both the physiological characteristics of the speaker as well as the extrinsic sources of variability such as the transmission channel and environmental noise.

Representative examples of the practical application of high level sources of information in recognition systems are the use of: conversational patterns [22]; lexical ideolects [23]; phonotactics [25]; and prosodic information [26].

Despite the availability of high level sources of speaker recognition, the vast majority of current automatic recognition systems relay mostly (if not uniquely) on low level information represented in terms of short-term spectro-temporal patterns of energy allocation. This is mostly due to the fact that the performance of systems based on spectral information is (at least) an order of magnitude better than the most competitive systems based on higher level information (see [27] for example). Also, in order to obtain a reliable model of the speaker based on higher levels of information, the amount of necessary speech is much larger than in the case of spectral information [21]. Nonetheless, the diversity of representation brought by the use of multiple sources of information is an effective way to obtain robustness to environmental noise and channel distortions [27]. In this thesis we focus on low level information and achieve robustness by improving the statistical models and the representation of the spectral information.

2.2. Automatic Speaker Recognition: Definitions and Applications

The term speaker recognition is normally used in a generic way in the speaker recognition community. It refers to any mode of operation that involves inferring the identity of a speaker. Within this generic term we can further differentiate between two particular tasks:

- **Speaker identification**: This mode of operation is concerned with associated an unknown with one particular speaker within a predefined set of speakers. Depending on the nature of the set it can be subdivided between *open-set* and *closed-set* identification. In the *open-set* situation it is possible that the observed speech sample might not belong to any of the predefined set of speakers. On the contrary, *closed-set* identification assumes that the observed sample belongs to one of the speakers in the set. Notice that *open-set* identification is more involved since it is necessary to establish a mechanism to determine if the test sample really belongs to any of the available speakers.

- **Speaker verification**: This mode of operation corresponds to a two-class (binary) classification problem in which we are interested in the question of whether a collection of utterances belong to the same speaker or not. Traditionally, a subset of utterances is collected in an initial enrollment stage and a statistical model of the speaker is built based on that data. Then the test utterance is compared against the model to produce a verification score. If the score is larger than a threshold (defined based on the application at hand) then the collection of utterances used for train and the test utterance are considered to come from the same speaker.

Another important difference between speaker recognition systems is based on the characteristics of the spoken text. In particular we can differentiate between the following:

- **Text-dependent**: In this scenario, the same speech content is required in all the utterances in order to produce a similarity score. Typical examples of this mode of operation are the use of a user PIN number or password. Alternatively, instead of requiring a fix utterance, a text-prompted strategy can be used in which the user is asked for a collection of words or short phrases from a predefined collection. Also, given two speech samples of unconstrained text content, an automatic speech recognition system can be used to find multiple occurrences of the same “token” and then perform text-dependent recognition based on them. This strategy assumes that there is enough speech such that the probability of having multiple occurrences is high.

- **Text-independent**: This modality does not impose any constraints in the linguistic content of the speech samples involved in the verification process. It is therefore less restrictive and also presents more challenges due to the lack of control over the content.

The particular choice of verification/identification and text-dependent/-independent will mostly depend on the particular application of the speaker recognition system. A possible grouping of applications follows:

- **Authentication**: This is the typical application for which a password would be use. Instead, a speaker verification system can be used to obtain access to a physical facility or login into any internet site.
- **Content indexing**: In this context the speaker recognition system is used to automatically index a multimedia collection (i.e., broadcast news, audio book archives, movies, etc) to facilitate searching and accessing content.
- **Forensic application**: In this context the similarity between speech samples is used as evidence for investigative purposes or in a court of law. The improved performance of the recognition systems is attracting more attention to this kind of applications [28].

In the next section we introduce the typical structure of a speaker recognition system.

2.3. Structure of Speaker Recognition Systems

The problem of speaker recognition, like the majority of problems in pattern recognition, can be divided into two parts: feature extraction and similarity computation. The feature extraction part is also denoted as "front-end" and the similarity computation as "back-end".

The ultimate goal of the front-end is to generate a representation from the speech signal that emphasizes the speaker-specific information while removing any undesired variability. This can be stated more formally in the following list of desiderata [29]:

- Efficient representation of speaker-specific information (i.e., small within-speaker variability and large between-speaker variability)
- Easy to compute
- Stable over time
- Occur naturally and frequently in speech
- Not be susceptible to mimicry
- Robust to environmental distortions

Usually, a speech utterance is converted into a sequence of feature vectors by densely sampling the signal in regular temporal intervals. In the case of low level spectral information the speech signal is analyzed using a short-time running window of approximately 20 to 40 ms that is shifted over time in 10 ms increments. The short-time segment of speech is normally denotes as "speech frame" and correspond to pseudo-stationary segments of speech. Among the most typical parameterizations of the information contained on a speech frame we find:

- **Linear Prediction Cepstral Coefficients (LPCC)** [30]: Based on a Linear Predictive Coding (LPC) [31] analysis of the speech frame, the set of prediction coefficients (typically 10 or 12) is transformed into a set of cepstral coefficients. The LPC analysis is based on an all-pole model of the speech signal that provides an efficient parametric representation of the spectral envelope.
- **Perceptual Linear Prediction (PLP)** [32]: Based on LPC analysis of a speech frame with several psychophysically based spectral transforms inspired from models of human perception. The transformations provide a small degree of robustness.
- **Mel-Frequency Cepstral Coefficients (MFCC)** [33]: Based on Fourier analysis of the speech frame and followed by a reduction of the frequency resolution by means of spectral integration using a collection of triangular filers spaced according to a mel-frequency scale. The output of the filters is mapped into the logarithmic domain and then projected onto a Discrete Cosine Transform (DCT) basis to reduce the correlation between the coefficients.

Each of the features described above can be finely tuned for the application at hand by optimizing the configuration of the building blocks of the feature extraction process. As an illustration, Figure 2.3 shows the typical signal processing chain used to compute MFCCs for speaker recognition.

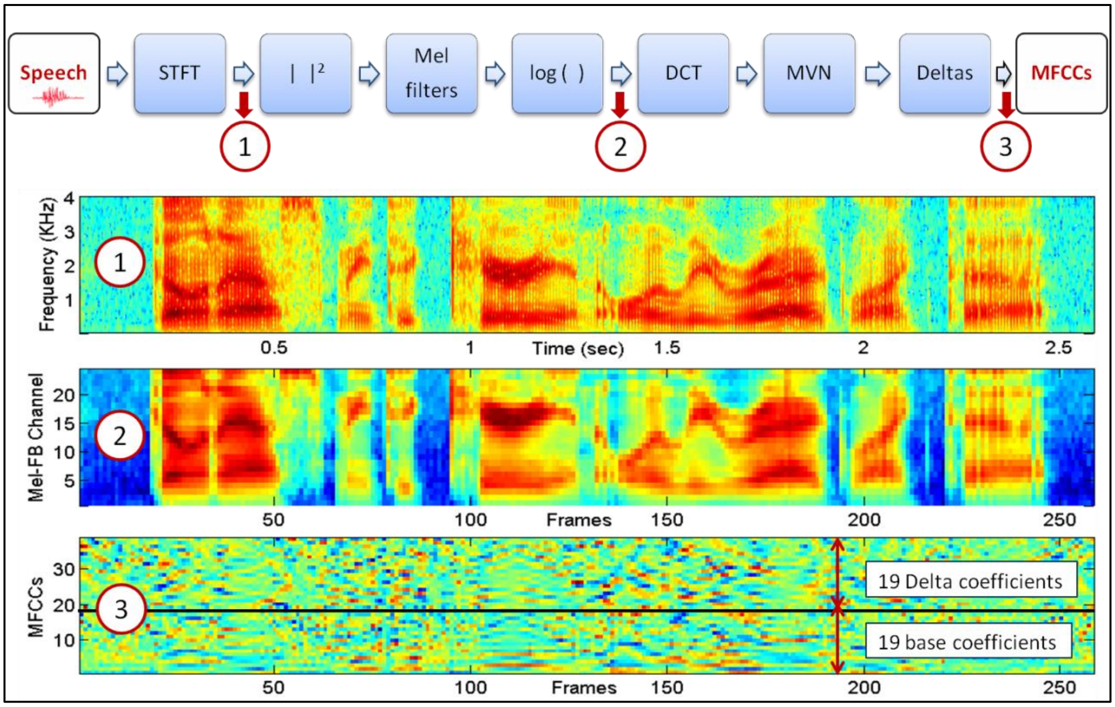


Figure 2.3: Signal processing chain of a conventional configuration of MFCCs for speaker recognition along with visual representations at three different points.

The first step involves computing a spectrogram based on the Short-Time Fourier Transform (STFT) that is applied over 20 ms windows with a 10 ms temporal increment. A lower resolution version of the spectrogram is obtained by averaging the spectral components of adjacent frequencies of the spectral slices (i.e., FFT coefficients of a speech frame) of the spectrogram. This spectral integration results in a dimensionality reduction and is performed according to a mel-frequency spacing of a collection of triangular filters [33]. For example, in the case of an 8 KHz sampling rate (4 KHz of speech bandwidth) the number of FFT coefficients is 128 and the number of mel-filters is typically 24.

Moreover, the output of the mel-filters is transformed into the logarithmic domain and projected into an orthogonal DCT basis. In practice, the first coefficient of the DCT (which corresponds to the geometric average of energy in dBs) is either discarded or sometimes replaced by the normalized log-energy of the speech frame. Also, only a subset of the remaining higher-order coefficients is preserved.

For speaker recognition applications it is customary to keep a larger number of DCT coefficients than for speech recognition (i.e., 19 coefficients as opposed to 13). By keeping a larger number of coefficients the details of the spectral envelope are represented with more accuracy.

Finally, temporally-steady spectral distortions are removed from the coefficients by applying normalizing transformations (see Section 2.5 for typical options), and a larger temporal context is obtained by computing first (delta) and second order (delta-delta) differences with the adjacent frames (normally a span of 2 frames from the left and right). In this way, an initial vector with 19 base DCT coefficients plus log-energy would result in a vector of 60 MFCCs by appending the delta and double-delta components to the base coefficients. Therefore, the final result corresponds to a temporal sequence of 60 dimensional MFCCs computed every 10 ms from temporal spans of around 100 ms.

Once a mechanism to extract information from a speech utterance is in place, the back-end is responsible for computing a similarity score between different utterances. There are two phases in the use of the back-end system: training and evaluation.

During the training phase, data from a particular speaker is used to build a model. It is also possible to require a large collection of utterances (development data) from a background population of speakers in order to build the speaker model (an example of this is given in the next section). Once a model is available, the back-end can operate in the evaluation mode and produce a similarity score between a speaker's model and a sample test utterance. Depending on the strategy used to construct the model, the score will have a probabilistic interpretation or it will simply quantify the similarity or distance between two speech samples. In both cases a higher number indicates a higher similarity.

A possible partition of back-end types in terms of the training paradigm is between *non-probabilistic* and *probabilistic* models. The non-probabilistic models use the training data to build a *discriminative function* that directly maps the input data into a similarity score (or class label in case of hard-decisions). A typical example of this approach that has been very successful in the speaker recognition community is the use of Support Vector Machines [34], [35]. In the case of probabilistic models, a further differentiation can be made between *generative* or *discriminative* [36] approaches. The main distinction between these two subclasses is that generative models attempt to model the class-conditional distributions, whereas the discriminative models target the posterior distribution of the classes directly. Notice that the class-conditional can be used along with the prior distributions to obtain the posterior probabilities using Bayes' rule. However, directly attempting to model the posterior distributions typically results in a smaller number of parameters [36] which may produce better estimates for a given fixed dataset.

All the speaker recognition systems presented in this thesis belong to the class of probabilistic generative models. Also, all of them evolved from the classic paradigm introduced by Reynolds et al. [37] based on adapted Gaussian Mixtures Models. For this reason, we review this paradigm in the next section.

2.4. Adapted Gaussian Mixture Models

The state-of-the-art systems discussed in this thesis evolved from the classic paradigm of Maximum a Posteriori (MAP) adapted Gaussian Mixture Model (GMM) introduced by Reynolds et al. [37]. As illustrated in Figure 2.4, this scheme can be seen as a likelihood ratio (LR) detector between a GMM model of a given speaker, and an average background GMM model, the so called Universal Background Model (UBM).

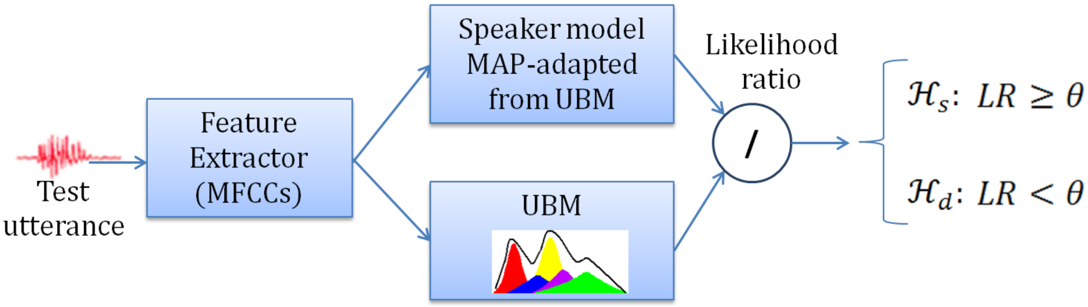


Figure 2.4: Speaker verification system based on likelihood ration between MAP-adapted speaker model and Universal Background Model GMM.

The UBM model is trained from a large collection of data using a Maximum Likelihood (ML) objective by the Expectation-Maximization (EM) algorithm [38]. It serves two purposes. The first one is to provide a model of a “generic” average speaker that will be used to compute a likelihood ratio. The second one is to provide a prior distribution to perform Maximum a Posteriori training of the speaker model [39]. Specifically, the parameters of the UBM are used to define the hyper-parameters of the conjugate prior distributions used for each of the Gaussians in the GMM of the speaker. Although it is possible to adapt all the parameters of the UBM (i.e., weights, means and covariance matrices) it is customary to only adapt the means of the Gaussian. This strategy (only adapting the means of the GMM from the UBM) has been proven empirically optimal by many researchers in the field for applications in which the amount of data available to train a speaker model is in the order of minutes [37]. Conceptually, this implies that the speaker-specific information contained in the training utterance is only encoded in the mean parameters of the speaker's GMM. That is, the particular ways in which a given speaker differs from a generic average speaker represented by the UBM are completely captured in the differences between the means of the UBM and the mean-only MAP-adapted speaker GMM.

Figure 2.5 illustrates this principle. The left picture depicts the configuration of a 3-mixture UBM that has already been trained in a two-dimensional feature space. On the right picture, the green crosses represent the feature vectors of the speaker's training utterance (e.g., MFCCs). Then, the GMM of the speaker (solid ellipsoids) is obtained by Bayesian adaptation of the means of the UBM [37]. Notice that only the means of the two Gaussians that are close to the observed data (responsible for the data) are adapted while the third one remains the same. Hence, for regions of the feature space in which no data is observed during training, the speaker model backs-off to the prior knowledge captured by the UBM (average generic speaker).

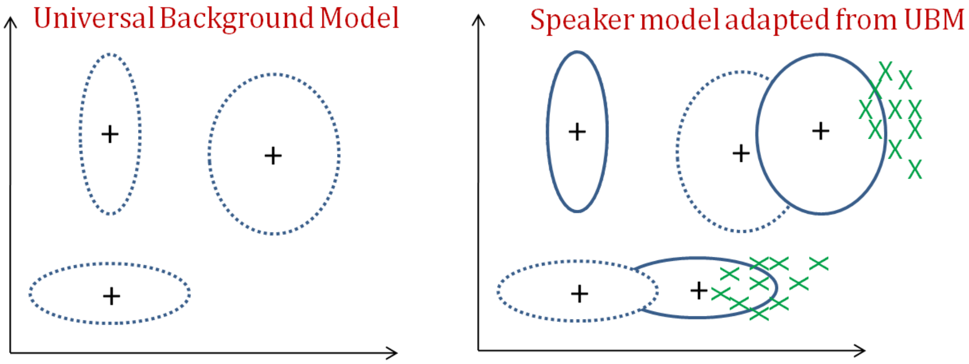


Figure 2.5: MAP adaptation of the means of the UBM based on observed data from speaker. Note that only the means of the mixtures responsible for the data are moved.

2.5. Undesired Variability and Compensation Approaches

As described in Section 2.1 the speech signal is the result of a complex process that involves respiratory, laryngeal, and vocal tract movements. This gives speakers a lot of degrees of freedom to alter their voices along dimensions such as: loudness, pitch, articulation rate, voice quality, etc. Moreover, the properties of a particular speech utterance vary along these dimensions as a function of a large collection of factors: phonetic content, language, speaking-style, environment, emotional state, health, etc. In this way, it is possible that a speaker never produces an utterance in the exact same way twice. Differences within a single speaker across occasions and utterances are called *intraspeaker* or *intrinsic* variability.

Besides the *intrinsic* variability, there are other factors of *extrinsic* variability such as the channel distortions introduced by acquisition devices (e.g., telephones), and the environmental distortions resulting from additive noise and room reverberation. The combination of both intrinsic and extrinsic variability is collectively referred to as *intersession* variability.

The success of a speaker recognition system relies on its ability to determine whether the nature and extent of the observed differences between two speech samples is better explained by the intersession variability (in which case the two utterances would belong to the same speaker) or by the *interspeaker* variability that arises from the speaker-specific information in the speech samples. To facilitate this judgment and improve the performance in a wide variety of application domains, the speaker recognition systems need mechanisms that suppress or attenuate the intersession variability.

One way to characterize these techniques is based on the domain in which they are applied: feature domain or model domain. Since most of the work in this thesis is based on improvements over model domain techniques, or transformations of latent variables from probabilistic generative models, we defer their exposition to Chapters 4, 5, and 6.

The following is a necessarily incomplete but representative list of the most widely used techniques for speaker recognition in the feature domain:

- **Cepstral mean normalization (CMN)** [30]: This technique is aimed at reducing the effects of convolutive noise from the channel. It is based on the principle that a convolutive distortion in the time domain is transformed into a constant offset into the cepstral domain. Therefore, by removing the mean of each cepstral coefficient the effects of the channel (assuming is not time-varying) are ameliorated.
- **Relative Spectral filtering (RASTA)** [40]: Based on knowledge about the dominant components of the modulation spectrum of the speech signal, the RASTA filter is designed as a band pass filter to eliminate the very slow changing components (convolutive noise) as well as the rapidly changing components (additive noise).
- **Feature Warping** [41]: This technique is aimed at reducing the effects of additive and convolutive noise by applying a nonlinear transformation that transforms the empirical distribution of each cepstral coefficient to a Gaussian distribution (Gaussianization). It is

normally applied using a running window of around 3 seconds of duration.

- **Feature Mapping** [42]: This is a data-driven technique that uses a collection of UBMs trained on data from a discrete set of distortions (i.e., cell phone speech, reverberant speech) to learn an inverse mapping of the distorted cepstral coefficients. This technique, along with its model domain counterpart [43], can be regarded as discrete versions of the state-of-the-art approaches based on Factor Analysis.

All the techniques mentioned above can also be combined with the model domain techniques that will be described in Chapters 4, 5, and 6.

2.6. Chapter Summary

In this chapter we presented a compact exposition of the basic concepts necessary to contextualize the work presented in this thesis. First we described the process by which speech signals are generated and the important sources of speaker-specific information they carry. Then we provided technical definitions about different speaker recognition modalities such as verification and identification, as well as the notions of text-dependent and text-independent Also, we listed the most typical applications in which they are used. This was followed by an overview of the basic constituent elements of a generic speaker recognition system. Moreover, the classical paradigm of speaker recognition based on Gaussian Mixture Models was summarized. Finally, we surveyed some of the most common techniques used to add robustness to speaker recognition systems that work in the feature domain.

Chapter 3

Supervector Representations

3.1. Introduction

A common theme among current speaker recognition systems based on short-time spectral information is the representation of a speech utterance as a single point in a high-dimensional space. This representation is denoted as “supervector” (SV) and all the systems studied in this thesis make us of it.

In this chapter we first provide some background knowledge and review the process used to map a sequence of feature vectors into a supervector. We then present a novel procedure for the visualization of supervectors by which qualitative insight about the information being captured can be obtained. Based on this visualization approach, the Switchboard-I database (SWB-I) is used to study the relationship between a data-driven partition of the acoustic space and a knowledge based partition in terms of broad phonetic classes.

3.2. Background

Obtaining fixed-size representations of variable-length objects is a pervasive technique among many pattern recognition applications [44], [45], [46]. The widespread use of these techniques stems from the fact that mapping variable-length objects into the same vector space facilitates the use of standard pattern recognition techniques. For example, we might be interested in classifying emails as spam/not-spam, and most likely, each email will have a different number of words. In this context, one of the best known examples of these techniques is the use of "bag-of-words" representations to describe documents [44]. This approach maps a document (considered as an unordered collection of words) into a fixed-length vector whose size equals the cardinality of a predefined vocabulary, and whose entries corresponds to the number of times each word appears in the document. Note that documents with different number of words are mapped into the same fixed-size space. This allows direct comparison between objects whose initial representation was of different size.

The same concept has also been applied to domains where the notion of "word" is not immediately apparent. For example, visual object categorization based on images of different sizes (i.e., different number of pixels) [45]. These approaches construct "visual words" by describing an image as a collection of patches (e.g., 5x5 pixel blocks) and performing some form of clustering to obtain a discrete set of codewords (i.e., cluster centroids). The predefined visual vocabulary (dictionary) is typically learned from a large collection of images representative of the task at hand. Once the vocabulary is set, the patches of a given image are clustered into the visual words and the image is represented as a histogram of the counts of each visual word.

In this way, we can highlight four important stages of these methodologies with examples from visual object categorization:

- **Patch formation:** The first step in this stage is the definition of a sampling grid from which the patches will be extracted. Two typical approaches are the use of uniform densely-sampled grids [47], or sparsely-sampled grids based on regions of interest (keypoints) [45]. Also the size of the patch is an important design variable.
- **Feature representation:** This stage transforms the patch content into a feature vector. A desired property of these feature vectors is robustness to typical sources of variability. SIFT descriptors [48] are commonly used for this reason in the vision community.
- **Dictionary construction:** This stage uses the feature vectors from a large collection of training data to obtain a discrete set of codewords that will be use to represent new images. Typically, the K-means algorithm is used to cluster the feature vectors of the training data into K codewords that will define the dictionary [49].
- **Object representation:** Once the dictionary is defined, an object (e.g., image) is represented as a fixed-size vector of codeword counts.

It is important to remark that the ordering of the data beyond the patch size is completely ignored by this representation (spatial structure for images or temporal structure for speech or text).

In the following section we describe how this general technique has been particularized (in the field of speaker recognition) to represent speech utterances.

3.3. From Sequences of MFCCs to Supervectors

As described in Section 2.3, the short-time spectral information in a speech signal is normally represented as a sequence of MFCCs. In this way, the notions of *patch formation* and *feature representations* described in the previous section are encapsulated in the way MFCCs are computed. In particular, the most typical setups in speaker recognition use 20 ms Hamming windows with 10 ms increments to compute the STFT. Hence, a MFCC feature vector comprising Delta and Double-Delta coefficients (with a span of two frames each) will contain $T = 9$ frames, which corresponds to a patch of 100 ms of speech. Note that this patch size is in the time scale of phonetic units in English [50].

Alternatively, considering the 2-D spectro-temporal representation of Mel-filterbank energies with C channels (typically $C = 24$) as the initial representation, a 2-D spectro-temporal segment of dimensions $(C \times T) = (24 \times 9)$ corresponds to the notion of patch. Moreover, the information contained in this spectro-temporal patch is compressed into a feature vector of MFCCs (normally 39 to 60 coefficients); thus, obtaining a compact representation for subsequent processing. The top left part of Figure 3.1 illustrates this process.

Once the notions of patch formation and feature representation are established, the next step is to define a dictionary. Unlike in the case of visual object categorization mentioned above, the strategy followed to compute speech supervectors is not based on hard-clustering thru K-means; instead, a soft-clustering of the acoustic space spanned by the MFCCs is performed using a GMM-UBM. Therefore, the "acoustic words" of the dictionary correspond to the means of each

Gaussian mixture. In practice, the typical number of mixtures of the GMM-UBM used to construct the dictionary is either 1024 or 2048.

The use of a GMM-UBM to perform a soft partition of the acoustic space is a natural choice in the context of speaker recognition; mostly because the classic recognition architecture is based on a GMM-UBM (see Section 2.4). A large collection of training data (typically 10 or 20 hours of data from around a thousand speakers) representative of the task at hand is used to train the GMM-UBM in a ML fashion. Normally, a few iterations of the EM algorithm (10 to 15 iterations) are enough to obtain a successful GMM-UBM.

Once a GMM-UBM is trained, $\lambda_{UBM} = (\{w_k\}, \{m_k\}, \{\Sigma_k\})$, a speech utterance parameterized in terms of sequences of MFCCs, $\mathcal{O} = \{o_t\}_{t=1}^{T}$ with $o_t \in \mathbb{R}^F$, is mapped into two supervectors (Figure 3.1 illustrates this process). The first supervector is denoted as the supervector of counts, and is constructed by appending together the soft-counts of the GMM. More formally, given the GMM-UBM λ_{UBM} and a feature vector o_t, the responsibility of mixture k for the observation frame o_t, at time t, is given by:

$$\gamma_{tk} = \frac{w_k \, \mathcal{N}(\mathrm{o}_t; m_k, \Sigma_k)}{\sum_{j=1}^{K} w_j \;\, \mathcal{N}\left(\mathrm{o}_t; m_j, \Sigma_j\right)}\,. \tag{3.1}$$

Moreover, the soft-count for mixture k is obtained by summing the responsibilities over all frames:

$$N_k = \sum_{t=1}^{T} \gamma_{tk}. \tag{3.2}$$

Then, the supervector of counts is formed as $N = [N_1 \; N_2 \; ... \; N_K]^T$.

The second supervector is denoted as the supervector of means, and for each mixture component is computed as the weighted average of the observed data; with the weights corresponding to the responsibilities of the mixture for each frame:

$$\mu_k = \frac{1}{N_k} \sum_{t=1}^{T} \gamma_{tk}\, o_t. \tag{3.3}$$

Then the supervector is obtained by appending the means for each mixture component as: $\mu = [\mu_1^T\ \mu_2^T \dots \mu_K^T]^T$.

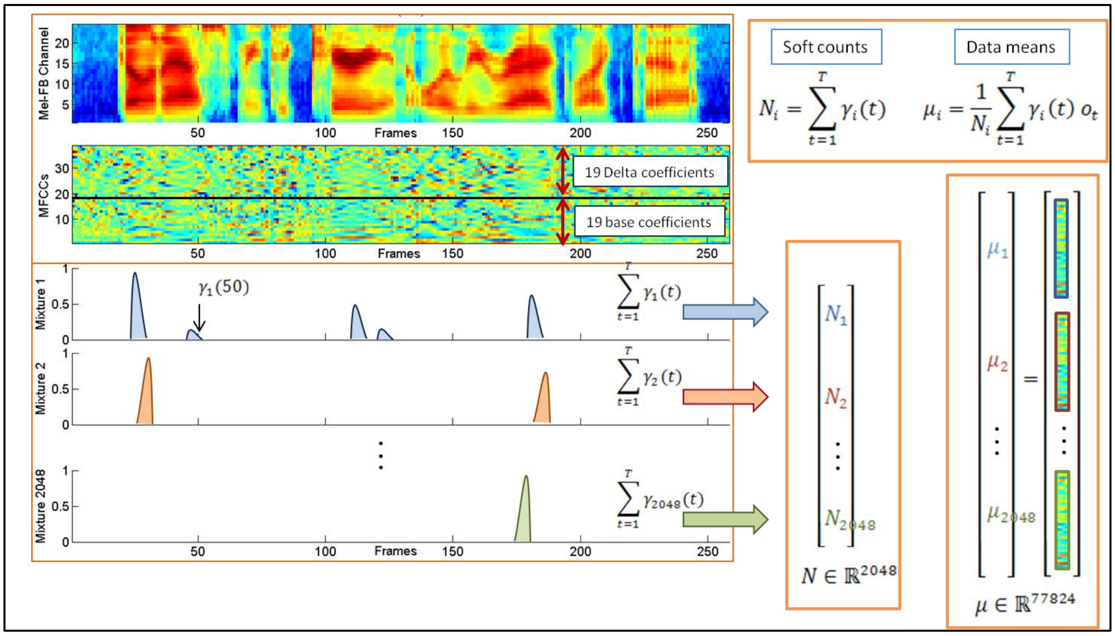

Mel-FB Channel
20
15
10
5
50
100
Frames
150
200
250
MFCCs
30
20
10
19 Delta coefficients
19 base coefficients
Mixture 1
1
0.5
0
$\gamma_1(50)$
$\sum_{t=1}^{T} \gamma_1(t)$
Mixture 2
$\sum_{t=1}^{T} \gamma_2(t)$
Mixture 2048
$\sum_{t=1}^{T} \gamma_{2048}(t)$
Soft counts
Data means
$N_i = \sum_{t=1}^{T} \gamma_i(t)$
$\mu_i = \frac{1}{N_i} \sum_{t=1}^{T} \gamma_i(t)\, o_t$
N_1
N_2
N_{2048}
$N \in \mathbb{R}^{2048}$
μ_1
μ_2
μ_{2048}
=
$\mu \in \mathbb{R}^{77824}$

Figure 3.1: Computation of supervector of counts and means from the temporal sequence of mixture responsibilities for each MFCC vector.

Figure 3.2 provides an alternative view of the process followed to compute both supervectors (assuming that the acoustic space is two-dimensional). Notice that instead of just creating a supervector of means μ, a supervector of offsets θ is created by centering μ around the supervector of GMM-UBM means $m = [m_1^T\ m_2^T \dots m_K^T]^T$. In this way, the information encoded in the supervector of offsets highlights how a particular speaker differs from an “average” speaker (represented by the GMM-

UBM) in the realization of the particular sounds that are being modeled by the corresponding GMM mixture. In particular, one can think of the GMM-UBM as an unsupervised data-driven mechanism to define regions of short-term patterns of spectral allocation of energy that occur very frequently. Then, considering the mean of each GMM component as an average “canonical” realization of the patterns represented by a region, the supervector of offsets encodes the characteristic way a particular speaker realizes those patterns.

Moreover, the supervector of counts represents the relative frequency with which a speaker produces those patterns. Hence, the counts will be highly dependent on the linguistic content (i.e., influenced by the statistical distribution of occurrence of the different sound of a language). However, they also encode the reliability of the corresponding components of the offset supervector; since the more often we observe a similar repetition of the same pattern, the more we can believe that it is a reliable descriptor of how a speaker realizes a patter over multiple instantiations.

In order to gain a better understanding of the information being captured by the supervector of offsets, it is important to answer the following question: Is there any relationship between a data-driven partition of the acoustic space and a knowledge-based partition? Answering this question will help understand the nature of the partition of the acoustic space, and therefore, the characteristics of the speaker-specific information represented in a supervector of offsets.

In the rest of this chapter we address this question in two different ways. First, we propose a novel technique for the visualization of supervectors of means. This visual representation provides qualitative insights into the information being captured.

Second, we conduct a quantitative analysis between the correspondence of an unsupervised data-driven partition of the acoustic space of MFCCs and a knowledge-based partition in terms of broad phonetic classes.

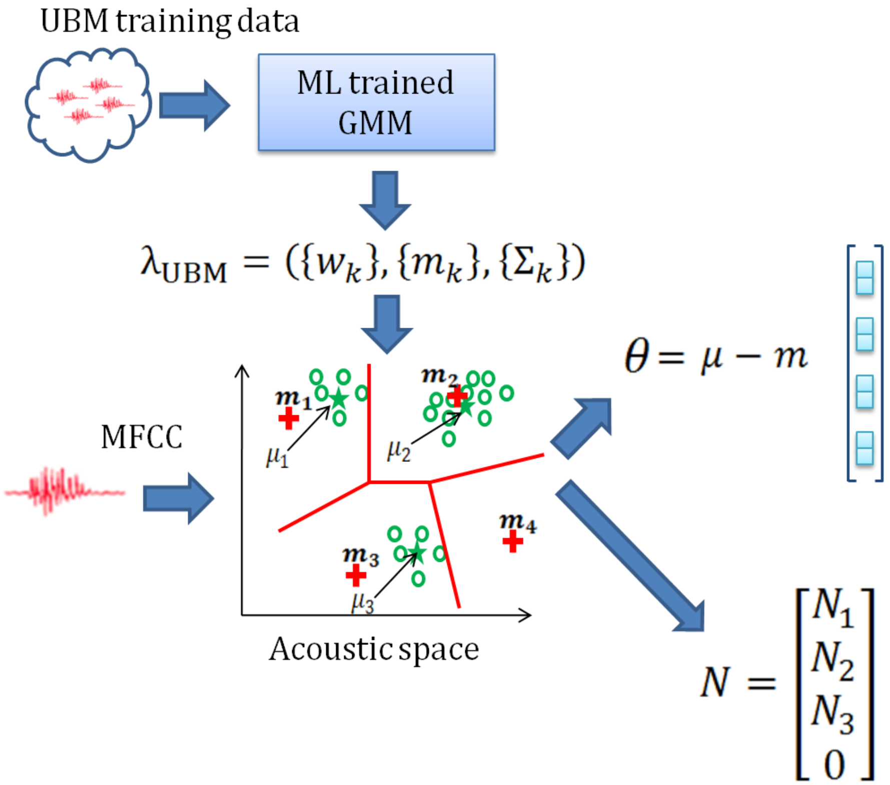

UBM training data
ML trained GMM
$\lambda_{\text{UBM}} = (\{w_k\}, \{m_k\}, \{\Sigma_k\})$
MFCC
m_1
m_2
m_3
m_4
μ_1
μ_2
μ_3
Acoustic space
$\theta = \mu - m$
$N = \begin{bmatrix} N_1 \\ N_2 \\ N_3 \\ 0 \end{bmatrix}$

Figure 3.2: Computation of supervectors of counts and data means using a GMM-UBM to partition the acoustic space of MFCCs.

3.4. Experimental Setup

In this section we present the details about the dataset used for our analysis as well as the configuration to obtain a GMM-UBM and the supervectors.

3.4.1. Switchboard-I Database

The Switchboard-I database is comprised of conversational speech between two speakers recorded over landline telephone channels with a sampling rate of 8

KHz [51]. The average duration of each conversation is 5 minutes (approx. 2.5 min per speaker) and each conversation side is recorded in a different file. The total number of speakers in the database is 520 with a balance in gender and recorded into 4856 speech files. The telephone handsets were either electret or carbon bottom with an approximate proportion of 70% and 30% respectively. The availability of manual phonetic transcriptions [21] along with a fairly limited amount of channel/handset variability makes this database a good candidate for the experiments in this chapter.

3.4.2. UBM Training

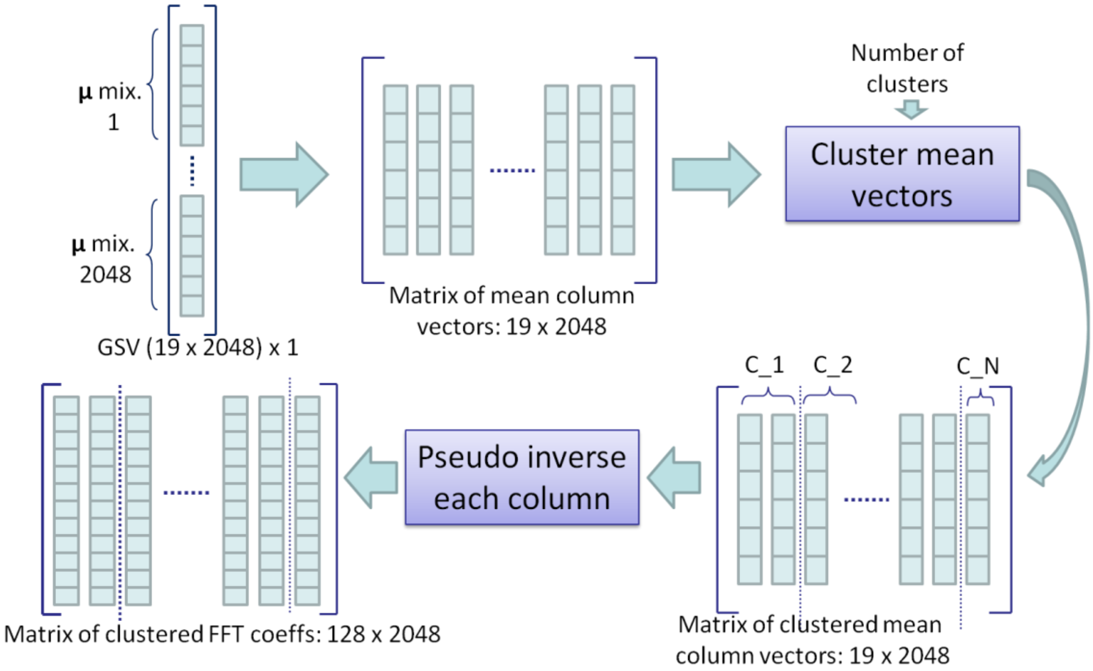
Each file in the database was parameterized into a sequence of 19-dimensional MFCC vectors using a 20ms Hamming window with a 10ms shift. The MFCC vectors were computed using a simulated triangular filterbank on the FFT spectrum. Prior to projecting the Mel-frequency band (MFB) energies into a DCT basis, bandlimiting was performed by discarding the filterbank outputs outside of the frequency range 300Hz-3138Hz. Finally, after projecting the MFB energies into a DCT basis and discarding C0, the 19-MFCC vectors were processed with RASTA filtering to reduce linear channel bias effects. No delta features were computed since we wanted to focus our analysis on static information only.

Two 2048-mixtures UBMs were trained based on a partition of SWB-I into two sets, P1 and P2, of 260 speakers each with a balance in gender and handset type. The UBM trained on P1 was used to obtain supervectors for the files in P2 and vice versa. The resulting dimension of the supervectors was $2048 \times 19 = 38{,}912$.

3.5. Visualization of Mean Supervectors

The speech technology community has greatly benefited from the ability to visualize spectro-temporal representations of the speech signal. A trained eye can gain a lot of qualitative insight by a simple inspection of a spectrogram. Unfortunately, what has proven very useful for information displaying (i.e., temporal sequences of FFT coefficients) is not optimal for other task unless further post-processing is applied. In the particular case of speaker recognition, examples of such post-processing include high-frequency resolution decrease, projection into orthogonal basis and dimensionality reduction. These standard signal processing techniques have tremendously improved the performance of the recognition systems. However, once the information has been processed in this way, it is extremely hard to make sense of what is really happening. One way to cope with this issue is to obtain a useful representation for the application at hand (i.e., speaker recognition) and then try to transform such representation to a domain in which qualitative knowledge can be obtained.

In this way, Figure 3.3 shows a diagram in which a SV of data means is transformed back into a matrix of clustered sets of FFT coefficients. The transformation process starts by reshaping the SV into a matrix with each mixture mean as a column. Subsequently, a number of clusters is specified and the mean vectors are grouped together by a K-means algorithm. As a result, the mean vectors corresponding to the Gaussian mixtures that are close together (i.e., in the Euclidean sense) in the acoustic space are clustered together. Up to this point, no meaningful transformation has being accomplished. The key of the process lies in the next step that we have denoted as “pseudo-inversion”.



μ mix. 1
μ mix. 2048
GSV (19 x 2048) x 1
Matrix of mean column vectors: 19 x 2048
Number of clusters
Cluster mean vectors
C_1
C_2
C_N
Matrix of clustered mean column vectors: 19 x 2048
Pseudo inverse each column
Matrix of clustered FFT coeffs: 128 x 2048

Figure 3.3: Visualization of SV of data means.

Figure 3.5 depicts the steps followed in the pseudo-inversion. It is clear that it attempts to inverse the orthogonalization of the DCT basis as well as the effect of the simulated triangular filterbank. However, since we dropped the C0 coefficient in the computation of the 19 MFCCs, the result of the DCT inversion will be a vector of 20 MFB normalized energies. Moreover, the triangular filterbank processing is not an invertible mapping since it is many-to-one. It is for this reason that the prefix “pseudo” is attached to this inversion process. Hence, the pseudo-inversion of this process is performed by constructing a matrix whose columns are the weights of each one of the triangular filters (i.e., dimensions 128 x 20). Finally, it is important to note that since the spectrum was bandlimited during the feature extraction process, the resulting FFT coefficients only span the frequency range 300Hz-3138Hz.

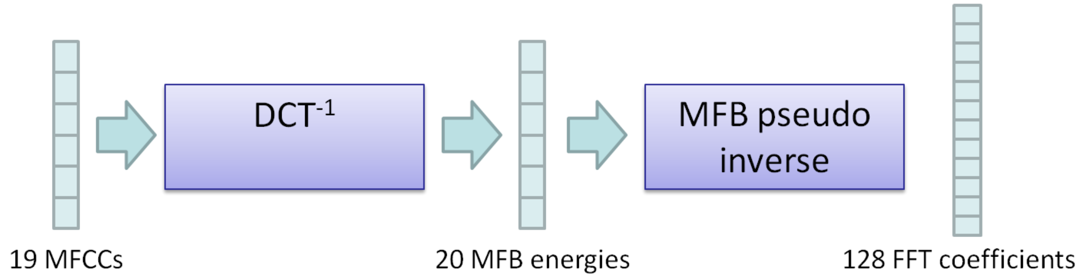
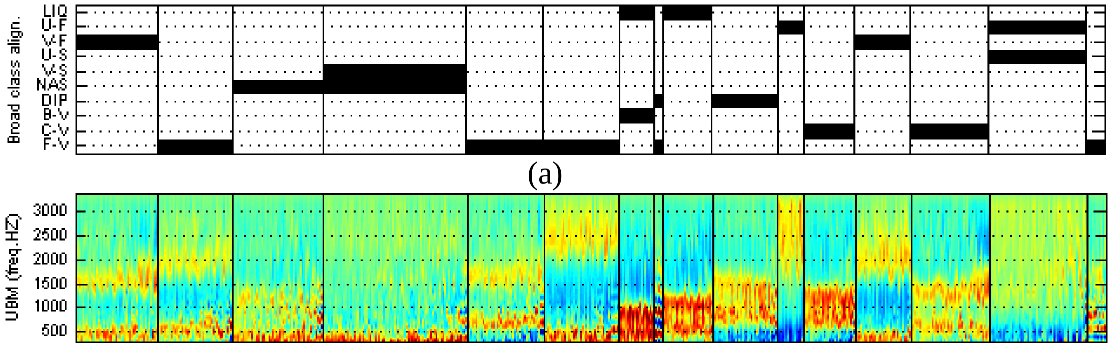


Figure 3.4: Pseudo-inversion of MFCC coefficients back to vectors of 128 FFT coefficients.

Panel (b) of Figure 3.5 shows the result of processing the SV of the UBM means of the partition P1 of SWB-I. In the following section we present the insights that this representation provides with respect to the information being captured in a supervector representation.



(a)

(b)

Figure 3.5: (a) Broad-phonetic class alignment with the data-driven partition of the acoustic space (see text for description of the codes used for the phonetic classes). (b) Visualization of the mean SV of the UBM for partition P1 of SWB-I.

3.6. Relation between Data-Driven Partitions and Broad Phonetic Classes

A mean supervector can be understood as a summary of the short-term average patterns of spectral allocation of energy of a particular speaker. Moreover, the linguistic content of the speech signal imposes some constraints on these patterns (e.g., relative position of formants). In this way, it seems natural that the elements (i.e., mean vectors) of the supervector will exhibit some kind of clustering. To check this, a K-means procedure was used to partition the elements of the two UBMs supervectors into a set of classes. The Euclidean distance between the mean vectors (i.e., 19 MFCC vectors) was used for the clustering and the number of classes was set to 16. We followed the visualization methodology of Figure 3.3 to display the GSV of the UBM for P1. Panel (b) of Figure 3.5 shows the result. The same behavior was observed for the UBM of P2.

It is important to note that the clustering was done prior to the pseudo-inversion stage (in the MFCC space) and therefore no imprecision was introduced in the process. An inspection of the UBM supervector of Figure 3.5 reveals that the mean vectors that get grouped together share in common their most predominant regions of spectral allocation of energy (i.e., formants). This raises the following question: Is there any relationship between a data-driven partition of the acoustic space and a knowledge-based partition (such as the broad phonetic classes)?

In order to answer this question the following experiment was conducted in each of the SWB-I partitions independently. First, for each file, the manual phonetic transcriptions of SWB-I [51] were used to align each feature vector with a broad phonetic class. The following set of phonetic classes was used: liquids (LIQ), nasals (NAS), voiced/unvoiced fricatives (V/U-F), voiced/unvoiced stops (V/U-S), diphthongs (DIP), and back/center/front vowels (B/C/F-V). Then a probabilistic alignment of each feature vector with their corresponding UBM was performed. Only the top-1 scoring mixture was used for each feature vector. During this process, we kept track of the number of times each one of the 2048 UBM’s mixtures was used to score a frame with a particular phonetic class label. As a result, we obtained a probabilistic alignment of each UBM mixture with the aforementioned set of broad phonetic classes. As an example, if a given mixture was used 80% of the time to score frames in nasal regions, the process would assign a 0.8 probability mass to that mixture with respect to nasals.

Two important observations were made. First, every mixture had a non-zero probability mass for each broad phonetic class. Second and most important, the probability mass was not uniformly distributed and was highly concentrated on one or two phonetic classes for each mixture. Moreover, in order to establish a connection between the data-driven clusters and the broad phonetic classes we averaged the probabilistic assignments among all the mixtures in the same data-driven cluster. The top panel of Figure 3.5 shows the result of thresholding this averaged probabilistic alignment to keep approximately 90% of the probability mass. Each data-driven cluster gets aligned with at most 2 broad phonetic classes. After a close analysis of the resulting pairings between data-driven clusters and phonetic classes, it can be observed that there is a good matching between the formant regions of the clusters and the canonical formant regions of the phonetic classes (see [50] for examples of these). Although Figure 3.5 only depicts the results for the partition P1 of SWB-I, the same observations were made by analyzing the results of P2. This supports the generality of the results.

Based on the experiments presented in this section we can claim that not only supervectors capture short-time average patterns of spectral allocation of energy, but also a phonetic meaning can be attached to partitions of the supervector.

3.7. Chapter Summary

In this chapter we first argued that obtaining fixed-size representations of variable-length objects is a pervasive technique among many pattern recognition applications. We then provided examples of how this idea has been used in pattern recognition applications based on test documents and images. These examples were used to abstract four fundamental stages common to these methodologies.

After that, we explored how this technique was manifested in the speaker recognition community. In particular, we identified how the four fundamental stages were particularized to represent speech utterances. That is, how to map a sequence of MFCCs into a supervector of counts and a supervector means. We described how a GMM-UBM was used to perform an unsupervised data-driven partition of the acoustic space of MFFCs. Then, we provided intuition about the information being captured by a supervector representation. This intuition was formalized by using a novel procedure for the visualization of supervectors by which qualitative insight about the information being captured was obtained. Based on this visualization approach, the Switchboard-I database (SWB-I) was used to study the relationship between a data-driven partition of the acoustic space and a knowledge based partition in terms of broad phonetic classes. The results of the analysis indicated that different subsets of supervector entries can be identified with a particular phonetic context with high probability. In light of that, a supervector of means can be understood as a summary of the short-term average patterns of spectral allocation of energy of a particular speaker in different phonetic contexts.

Chapter 4

Overcomplete Dictionaries for Speaker Recognition

4.1. Introduction

The supervector formalism presented in the previous chapter provides a mechanism to obtain a fixed-length representation of a variable length object. That is, an entire speech recording is mapped into a fixed-length supervector in a very high-dimensional space (of the order of 100,000 dimensions). Moreover, as detailed in Section 3.6, subsets of entries of a supervector can be associated with linguistically meaningful units such as broad phonetic classes. However, the direct use of this representation in a speaker recognition system is not optimal; since supervectors not only capture speaker-specific information but also contain a large amount of undesired variability entangled with the desired information (see Section 2.5). Hence, there is a need for a mechanism to disentangle the speaker-specific information and the undesired variability captured in the supervector representations.

Inspired by the speaker adaptation work based on eigenvoices [52], and its early application to the field of speaker recognition [53], the work in [54] proposed the use of Joint Factor Analysis (JFA) to explicitly model the speaker-specific and the undesired variability present in GMM mean supervectors. Specifically, JFA assumes that most of the speaker-specific information lies in a subspace of much lower dimensionality than the ambiance space, and that a supervector can be decomposed into a speaker-specific component and an undesired variability one. Based on this paradigm, removing undesired components from speech representation becomes much easier since they are explicitly modeled.

The theory of the JFA paradigm presents an elegant probabilistic perspective around a generative linear-Gaussian model on GMM mean supervectors. Hence, a consistent application of the product and sum rules of probability suffice to obtain a speaker recognition system based on likelihood ratios. However, the high-dimensional nature of the supervector space proved to be challenging and a lot of algorithmic approximations to the theory were explored (between the period of 2004 and 2008) in order to make the paradigm useful in realistic scenarios (see [55] for a review).

The main goal of this chapter is to provide a non-probabilistic view of the underlying processes followed in JFA. In particular, we explore the connection between the JFA paradigm and the use of signal coding in overcomplete dictionaries learned from data. By establishing this connection we are able to provide two algorithmic improvements over the baseline JFA system. One improvement comes in the form of improved computation whereas the other comes in terms of improved recognition accuracy. The remainder of this chapter is organized as follows. Section 4.2 provides an overview of the baseline JFA system. Section 4.3 establishes a novel connection between overcomplete dictionaries and JFA. Moreover, two algorithmic improvements are proposed. Section 4.4 provides details about the experimental setup used to test the proposed ideas. Section 4.5 presents the experimental results. Finally, Section 4.6 summarizes the chapter.

4.2. Overview of Joint Factor Analysis

Since the introduction of JFA in [54] a great number of modifications have been proposed [55]. In order to remove any ambiguity about our particular choice of JFA variant, this section presents an overview of the three fundamental steps involved in the construction of a speaker recognition system: model training, hyperparameter estimation and score computation.

4.2.1. Paradigm

The Joint Factor Analysis paradigm [56] assumes that a sequence of T I.I.D. observed vectors, $\mathcal{O} = \{\mathrm{o}_t\}_{t=1}^{T}$ with $\mathrm{o}_t \in \mathbb{R}^F$, comes from a two-stage generative model. The first stage corresponds to a K-component Gaussian Mixture Model (GMM), $\lambda = (\{w_k\}, \{\theta_k\}, \{\Sigma_k\})$, that is responsible for generating each observed vector o_t :

$$p_\lambda(\mathrm{o}_t|\{\theta_k\}) = \sum_{k=1}^{K} w_k \frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma_k|^{\frac{1}{2}}} \exp\left\{\frac{1}{2}(o_t - \theta_k)^T \Sigma_k^{-1} (o_t - \theta_k)\right\} \tag{4.1}$$

with $w_k \in \mathbb{R}, \theta_k \in \mathbb{R}^F$and $\Sigma_k \in \mathbb{R}^{F\times F}$ for $k = 1, \dots, K$

The weights and covariance matrices of the GMM are considered fixed and known a priori. The means are assumed to be random vectors generated by the second stage of the generative model. In particular, a mean supervector $\theta = [\theta_1^T, \dots, \theta_K^T]^T \in \mathbb{R}^{FK}$ is constructed by appending together the means of each mixture component and is assumed to obey an affine linear model of the form

$$\theta = m + \mathrm{U}x + \mathrm{V}y + \mathrm{D}z \tag{4.2}$$

where the vector $m \in \mathbb{R}^{FK}$ is a fixed offset, the matrices $\mathrm{V} \in \mathbb{R}^{FK\times P_v}$ and $\mathrm{U} \in \mathbb{R}^{FK\times P_u}$ correspond to factor loadings and the diagonal matrix $\mathrm{D} \in \mathbb{R}^{FK\times FK}$ is a scaling matrix. Moreover, the vectors $y \in \mathbb{R}^{P_u}$ and $x \in \mathbb{R}^{P_v}$ are considered as the common-factors and $z \in \mathbb{R}^{FK}$ as the residual-factors. All three vectors, x, y and z are assumed independent of each other and distributed according to a standard Normal distribution of appropriate dimension. Consequently, equation (4.2) implies that the prior distribution of the supervector θ is Gaussian with mean and covariance given by

$$\mathbb{E}[\theta] = m \;\text{ and }\; \mathrm{Cov}[\theta] = \mathrm{UU}^T + \mathrm{VV}^T + \mathrm{DD}^T. \tag{4.3}$$

The rationale behind equation (4.2) is that, aside from the offset m, the mean supervector is the superposition of three fundamental components with rather distinctive meanings. The component that lives in the $\mathrm{span}(\mathrm{U})$ is used to denote the undesired variability contained in the observed vectors (e.g., convolutive or additive noise). Additionally, the $\mathrm{span}(\mathrm{V})$ is where the basic constituting elements that capture the essence of the observed data live (speaker-specific information). Finally, the diagonal matrix D spans the entire ambiance space and provides a mechanism to account for the residual variability not captured by the other two components.

In equation (4.1), the weights $\{w_k\}$ and covariance matrices $\{\Sigma_k\}$ of the GMM λ are assumed to be fixed and known a priori. In practice, they are obtained from a previously trained GMM λ_{UBM} called Universal Background Model (UBM). This UBM must be trained using a large collection of data that is representative of the task at hand. Maximum Likelihood estimation is the most common approach [37].

4.2.2. JFA Model Training

Now that all the elements involved in the JFA model have been defined, we are in position to formulate the inference problem (i.e., model training). That is, given a sequence of observed vectors $\mathcal{O} = \{o_t\}_{t=1}^{T}$, we want to estimate the free parameters of the generating GMM that maximize the posterior distribution—which in this case are only the component means $\{\theta_k\}$. We will also assume that the hyperparameters $\{m, \mathrm{V}, \mathrm{U}, \mathrm{D}\}$ of the second stage of the generative process are also known (i.e., they have been obtained previously from a development data set via the ML approach described in next section). Thus, our optimization problem takes the form

$$\max_{\theta}\ p(\theta|\mathcal{O}) = \max_{\theta}\ p_{\lambda}(\mathcal{O}|\theta)\, p_{\theta}(\theta). \qquad (4.4)$$

In order to keep the formulas as clean as possible, we will refer to the entire collection of loading matrices by $\Phi = [\mathrm{U\ V\ D}] \in \mathbb{R}^{FK\times P}$and all the factors will be collected in $\beta = [x^T y^T z^T]^T \in \mathbb{R}^P$. Using this compact form, the mean supervector can also be expressed as

$$\theta = m + \Phi\beta. \qquad (4.5)$$

Moreover, based on the prior distributions of the factors x, y and z as well as their independence, the vector β is distributed according to a standard Gaussian distribution. That is

$$p_\beta(\beta) = \mathcal{N}(\beta; 0, \mathrm{I}). \tag{4.6}$$

Making use of equations (4.5) and (4.6) and substituting back into (4.4) an equivalent minimization problem can be obtained in terms of β:

$$\min_\beta \{-\log p_\lambda(\mathcal{O} | \theta = m + \Phi\beta) - \log p_\beta(\beta)\}. \tag{4.7}$$

Once the optimal β_{MAP} is obtained, we can compute the optimal mean supervector θ_{MAP} as:

$$\theta_{MAP} = m + \Phi\beta_{MAP}. \tag{4.8}$$

As usual, the analytical solution of this problem is not tractable and we use the EM algorithm to obtain a local optimizer. In the E-step we compute the occupations of the mixture component k for the observed vector o_t as

$$\gamma_{tk} = \frac{w_k\, \mathcal{N}\left(\mathrm{o}_t; \hat{\theta}_k, \Sigma_k\right)}{\sum_{j=1}^{K} w_j\ \mathcal{N}\left(\mathrm{o}_t; \hat{\theta}_j, \Sigma_j\right)}, \tag{4.9}$$

where $\hat{\theta} = \left[\hat{\theta}_1^T, \dots, \hat{\theta}_K^T\right]^T \in \mathbb{R}^{FK}$is initialized with m. Then, in the M-step we use the occupations $\{\gamma_{tk}\}$ to compute the complete-data log likelihood, that along with the prior for β, allow us to obtain the easier to optimize surrogate objective

$$\widetilde{\Psi}(\beta) = \frac{1}{2} \sum_{k=1}^{K} \sum_{t=1}^{T} \gamma_{tk}\ (o_t - m_k - \Phi_k\beta)^T \Sigma_k^{-1} (o_t - m_k - \Phi_k\beta) + \frac{1}{2}\ \beta^T\beta, \tag{4.10}$$

where m_k is the F-dimensional sub-vector of m indexed by the mixture component k. In order to obtain a complete vector-form expression for (4.10) without the summations, the following definitions are useful:

$$\gamma_k = \sum_{t=1}^{T} \gamma_{tk}\,,\ \ \Gamma_k = \gamma_k \mathrm{I} \in \mathbb{R}^{F\times F} \text{ and} \tag{4.11}$$

$$\Gamma = \mathrm{diag}(\Gamma_k) \in\ \mathbb{R}^{FK\times FK}.$$

The scalar γ_k represents how much of the observed data is accounted for by mixture k. The diagonal matrix $\Gamma_k$is an intermediate construct that replicates the scalar γ_k throughout F diagonal entries and the diagonal matrix Γ—constructed using the $\mathrm{diag}(\cdot)$ operator—contains the K matrices $\Gamma_k$in its diagonal entries. Additionally, the following objects are also useful:

$$\mu_k = \frac{1}{\gamma_k}\sum_{t=1}^{T} \gamma_{tk}\ o_t\,,\ \ \mu = [\mu_1^T, \dots, \mu_{\mathrm{K}}^T]^T \in \mathbb{R}^{FK} \text{ and} \tag{4.12}$$

with μ_k representing the weighted average of the observed data that is accounted for by the k^{th} mixture component. Taking equation (4.10), summing over the index t, and using μ_k from (4.12) we obtain the equivalent objective[3]

$$\Psi(\beta) = \frac{1}{2}\sum_{k=1}^{K} \gamma_k\ (\mu_k - m_k - \Phi_k\beta)^T \Sigma_k^{-1}(\mu_k - m_k - \Phi_k\beta) + \frac{1}{2}\ \beta^T\beta. \tag{4.13}$$

Finally, the summation over k can be taken care of—in an implicit way—by using the supervector notation:

$$\Psi(\beta) = \frac{1}{2}(\eta - \Phi\beta)^T \Gamma\Sigma^{-1}(\eta - \Phi\beta) + \frac{1}{2}\ \beta^T\beta, \tag{4.14}$$

where the diagonal matrix $\Sigma = \mathrm{diag}(\Sigma_k) \in \mathbb{R}^{FK\times FK}$. Moreover, letting $\mathrm{W} = \Gamma\Sigma^{-1}$, we can obtain the alternative expression:

[3] Note that equations (4.10) and (4.13) are not equal but just equivalent for the optimization process with respect to β. This stems from the fact that the covariance matrices $\{\Sigma_\mathrm{k}\}$ are taken from the UBM and kept fixed.

$$\Psi(\beta) = \frac{1}{2}\left\|\mathrm{W}^{\frac{1}{2}}(\eta - \Phi\beta)\right\|_2^2 + \frac{1}{2}\ \|\beta\|_2^2. \tag{4.15}$$

Noting that by construction W is diagonal positive semi-definite (or positive definite if all Gaussians are responsible for some data), it is easy to see that $\Psi(\beta)$ is strongly convex. Hence, computing the gradient and setting it to zero provides a necessary and sufficient condition for a unique global minimizer. Performing this operation we obtain a closed-form solution to problem (4.7):

$$\beta_{MAP} = (\mathrm{I} + \Phi^T \mathrm{W} \Phi)^{-1} \Phi^T \mathrm{W}\, \eta \tag{4.16}$$

4.2.3. Hyperparameter Estimation

Since the JFA paradigm is only as good as its hypermeters[4], the estimation of the set $\{m, \mathrm{V}, \mathrm{U}, \mathrm{D}\}$ has received a lot of attention. In particular, some of the variables being explored are: amount and type of data, number of dimensions of the subspaces, joint or independent estimation, generalization capabilities based on utterance duration and recording environments [57]. The most widespread criterion for the estimation process is the maximization of the likelihood function over a development data set [58]. The EM algorithm is used to maximize the likelihood. The offset supervector m comes from the UBM model. Independent estimation of the matrices U, V and D reduces the computational complexity greatly and provides state-of-the-art results [57]. Hence, that is the setup considered throughout this chapter. In particular, given an initial guess Φ_0—which depending on the matrix being updated is identified with $\mathrm{U}_0, \mathrm{V}_0$ or D_0—the E-step, for each data file r, produces the posterior means

[4] Note that we are not including the covariance matrices $\{\boldsymbol{\Sigma}_k\}$ as part of the hyperparameters to emphasize the fact that we keep them fixed once computed in the UBM training process.

$\nu_r = \beta_r^{MAP}$ and correlation matrices $\mathbb{E}[\beta_r \beta_r^T] = (\mathrm{I} + \Phi_0^T \mathrm{W}_r \Phi_0)^{-1} + \nu_r \nu_r^T$. The M-step results in the update equation [58]:

$$\Phi_{\text{new}}^{(k)} = \left(\sum_{r=1}^{R} \gamma_{rk} \ \eta_r^{(k)} \beta_r \right) \left(\sum_{r=1}^{R} \gamma_{rk} \ \mathbb{E}[\beta_r \beta_r^T] \right)^{-1}, \tag{4.17}$$

where the super-index k indicates the F-dimensional subset of rows corresponding to the mixture k and the index r runs through the elements of the training data set. Thus, if JFA comprises a GMM with Kcomponents and $\Phi_0 \in \mathbb{R}^{FK \times P}$, the updated Φ_{new} requires the solution of K independent systems of P equations with F right-hand side elements.

4.2.4. JFA Scoring

Once the hyperparameters and model training procedures are available, the only remaining component for a complete speaker recognition system is a similarity measure between models and test utterances. In [59] a comparison of scoring techniques ranging from a fully Bayesian approach to simple MAP point estimates was presented. The results indicated that—given enough data—a linear approximation of the log-likelihood results in a much faster computation of similarities without any significant loss in performance. Adapting their formulation to our notation, the speaker model is represented by $\hat{\eta} = \Phi \beta_{MAP} - \mathrm{U} x_{MAP}^{model}$ and the test utterance is summarized by its normalized, centered and session compensated first order sufficient statistics $\eta_{test} - \mathrm{U} x_{MAP}^{test}$. Recalling that $\mathrm{W}_{test} = \Gamma_{test} \Sigma^{-1}$, the final score is nothing more than the frame-normalized inner product

$$score = \frac{1}{T}\,\hat{\eta}^T\,\mathrm{W}_{test}(\eta_{test} - \mathrm{U}x_{MAP}^{test}) \qquad (4.18)$$

defined by the diagonal and positive definite matrix[5] W_{test}.

4.3. JFA and Signal Coding in Overcomplete Dictionaries

In this section we present a reinterpretation of JFA as a signal approximation methodology—based on Ridge regression—using an overcomplete dictionary Φ learned from data. With a simple change in perspective we will be able to abstract some of the unimportant details of JFA and bring to the foreground its essential principles. Moreover, establishing a connection between JFA and signal coding (SC) opens the doors for cross-pollination between fields (see [60] for a review of current trends in data-driven overcomplete dictionaries).

4.3.1. Signal Coding (SC)

We propose to deemphasize the two-stage generative model and focus on the EM part of the inference process. That is, to think of the E-step as a process that given a speech signal $\mathcal{O} = \{o_t\}_{t=1}^{T}$ with $o_t \in \mathbb{R}^F$ and a K-mixture UBM $\lambda_{UBM} = (\{w_k\},\{m_k\},\{\Sigma_k\})$ produces a fixed-length target vector $\eta \in \mathbb{R}^{FK}$ (see equation (4.12)) as well as a weighting diagonal matrix W. Then, the M-step can be reinterpreted as a signal coding process—of the target vector η—based on a weighted regularized linear regression approach. By looking at equation (4.15), we see that the objective function is comprised of two terms. The first one is a conventional

[5] Note that in the finite precision case where not all Gaussians may be responsible for at least one observation, the matrix $\mathbf{W}_{test}$ is in fact positive semi-definite. In that case, equation (4.18) is still correct if we define the inner product in the subspace where the diagonal entries of $\mathbf{W}_{test}$ are strictly positive.

weighted least squares loss; whereas the second is a penalty on the energy of the regression coefficients (i.e., ridge regularization term). These two terms represent a trade-off between the goodness-of-fit and the energy used to fit the target. The goal is to approximate the target vector η—as well as possible—with a linear combination of the columns of Φ while considering that there is a quadratic cost incurred by the amount of usage of each column. The diagonal weighting matrix W provides a mechanism to emphasize/deemphasize the relative importance of the coefficients of η in the approximation process. Fortunately, there is a unique closed-form solution to this problem and it was given in (4.16). Therefore, when using a JFA paradigm based on point estimates, the model training process is equivalent to a signal approximation. In this case, the signal being approximated happens to be the offsets—with respect to the UBM supervector m—of the normalized first order statistics η, contextualized by the soft-partition of the acoustic space induced by the UBM.

4.3.2. Dictionary Learning

Following the jargon particular to the sparse coding community, we will refer to the matrix Φ as a dictionary whose columns are denoted as atoms. For JFA, the dictionary is comprised of $\Phi_{JFA} = [\mathrm{UVD}]$ and is considered overcomplete since there are more columns than rows. This notation also applies to the eigenchannel configuration $\Phi_{ECH} = [\mathrm{UD}]$ as well as the relevance MAP formulation $\Phi_{rMAP} = \mathrm{D}_{rMAP}$ (although in this last case the dictionary is not overcomplete). The atoms of the dictionary should represent the basic constituent elements of the signals being coded as well as their typical distortions. In order for this to be the case, the best alternative is to learn these atomic representations from actual data similar to the one being coded. Thus, the process of learning a dictionary from data is equivalent to the estimation of hyperparameters in JFA. Specifically, given a training data set $\mathfrak{D} = \{\mathcal{O}_r\}_{r=1}^{R}$ with R utterances—after applying the E-step described in the (SC) section—the information in each utterance $\mathcal{O}_r$ is represented by the pair (η_r, W_r). Hence, the dictionary training problem is expressed as:

$$\min_{\Phi,\{\beta_r\}} \sum_{r=1}^{R} \left\| \mathrm{W}_r^{\frac{1}{2}}(\eta_r - \Phi\beta_r) \right\|_2^2 + \|\beta_r\|_2^2 . \qquad (4.19)$$

Note that unlike equation (4.15), the objective in (4.19) also involves the dictionary as an optimization variable. Hence, even though when considered as a function of either $\{\beta_r\}$ or Φ the objective is convex, it is not the case for the joint optimization in (4.19). This situation arises quite frequently and the use of alternating optimization [61] is one of the most conventional ways to address it.

Block Coordinate Descent (BCD) Minimization

A particular configuration of alternating optimization known as block coordinate minimization (i.e., non-linear Gauss-Seidel) is well suited for the case at hand [61]. Specifically, we consider a two step process. In one step, the block of variables Φ is fixed and the objective is minimized with respect to $\{\beta_r\}$. In the other step, the dictionary is updated while keeping the coefficients obtained in the previous step $\{\beta_r\}^{opt}$ fixed. Cycling between these two steps is repeated until convergence or sufficient decrease of the objective is obtained. Because the joint objective in (4.19) is non-convex this method only finds a local minimum and different initial values of the dictionary Φ_0 lead to different solutions. Note that the first step is exactly the SC stage described in the previous section. The second step is denoted as dictionary update (DU) and is addressed next.

Dictionary Update

Keeping the regression coefficients fixed for all utterances, the minimization of the objective in (4.19) with respect to the dictionary reduces to

$$\min_{\Phi} \sum_{r=1}^{R} \left\| \mathrm{W}_r^{\frac{1}{2}} (\eta_r - \Phi\beta_r) \right\|_2^2 . \tag{4.20}$$

A simple application of the definition of convexity reveals that for any given utterance (η_r, W_r) the corresponding term inside the summation is convex. Subsequently, the positive sum of R convex functions remains convex. Note that unlike in the SC step, in general the DU objective in (4.20) is not strongly convex and therefore there is no guarantee for a unique minimizer. However, any local minimizer is also global. Again, due to convexity, computing the gradient and setting it to zero provides a necessary and sufficient condition for a local/global minimizer. Using the identity $\mathrm{a}^T\mathrm{b} = \mathrm{tr}\{\mathrm{b}\mathrm{a}^T\}$ the problem in (4.20) is equivalent to

$$\min_{\Phi} \sum_{r=1}^{R} \mathrm{tr}\{\mathrm{W}_r \Phi \beta_r \beta_r^T \Phi^T\} - 2\ \mathrm{tr}\{\mathrm{W}_r \eta_r \beta_r^T \Phi^T\}. \tag{4.21}$$

Setting the gradient with respect to the dictionary Φ to zero results in

$$\sum_{r=1}^{R} \mathrm{W}_r \Phi \beta_r \beta_r^T = \sum_{r=1}^{R} \mathrm{W}_r \eta_r \beta_r^T. \tag{4.22}$$

Which, when restricted to the F rows corresponding to the k^{th} mixture simplifies to

$$\Phi_{new}^{(k)}\left(\sum_{r=1}^{R}\gamma_{rk}\beta_r\beta_r^T\right)=\sum_{r=1}^{R}\gamma_{rk}\eta_r^{(k)}\beta_r^T. \tag{4.23}$$

Comparing this result with the one obtained in (4.17) we can see that they are the same if we set the posterior covariance matrices $cov(\beta_r\beta_r^T)$ to zero. This is consistent with our formulation since we are ignoring the underlying probabilistic assumptions of the JFA model and treating the problem as a simple signal approximation.

Dictionary Learning Algorithm

An important algorithmic opportunity arises from this new perspective. In particular, we are going to exploit the computational advantage derived from the fact that no explicit matrix inversions are necessary. That is, we no longer need to compute $(\mathrm{I}+\Phi^T\mathrm{W}_r\Phi)^{-1}$ explicitly for each utterance to perform the dictionary update. This observation affects the DU step slightly but the most important gain comes from the SC step of the dictionary learning process. That is, much faster and numerically stable methods like Gauss-Seidel [62] or Cholesky factorizations can be used in the SC step[6] since no explicit matrix inversions are needed. Regarding the DU step, denoting the sum of R rank-one matrices corresponding to the k^{th} mixture by

$$\sum_{r=1}^{R}\gamma_{rk}\beta_r\beta_r^T=\mathrm{A}_R^{(k)}\in\mathbb{R}^{P\times P}, \tag{4.24}$$

and assuming that R is large enough so that $\mathrm{A}_R^{(k)}$is invertible, the updated Φ_{new} requires the solution of K independent systems of P equations with F right-hand side

[6] These techniques are also suitable for the JFA model, but if used instead of an explicit inversion, the task of computing the posterior covariance matrices still remains.

elements. A hybrid update formula between (4.23) and (4.17) can be obtained by setting

$$\mathrm{A}_R^{(k)} = \gamma_k \left(\mathrm{I} + \Phi_o^T \mathrm{W}_{avg}^{(k)} \Phi_o\right)^{-1} + \sum_{r=1}^{R} \gamma_{rk}\beta_r\beta_r^T \tag{4.25}$$

where Φ_o comes from the previous iteration of the DU (or from a simple PCA initialization for the first iteration). Also, $\mathrm{W}_{avg}^{(k)} = \Gamma_{avg}^{(k)}\Sigma^{-1}$ with $\Gamma_{avg}^{(k)} = \frac{1}{R}\sum_{r=1}^{R}\Gamma_r^{(k)}$ from the training set and $\gamma_k = \sum_{r=1}^{R}\gamma_{rk}$. In this way, instead of completely neglecting the covariance matrices $cov(\beta_r\beta_r^T)$ of the JFA model, we approximate all of them with a common one obtained by averaging the occupancy matrices $\Gamma_r^{(k)}$ over the entire training set. Also, using (4.25) removes any uncertainty about $\mathrm{A}_R^{(k)}$ being invertible.

```
Dictionary learning algorithm
Input: $\{(\eta_r, \mathrm{W}_r)\}_{r=1}^{R}$ and $\Phi_o$
Initialize: $\mathrm{W}_{avg} = \frac{1}{R}\sum_{r=1}^{R}\mathrm{W}_r$, $\Phi_{new} = \Phi_{old} = \Phi_o$
Until convergence:
  SC: Solve for each $\beta_r$ in (4.16) using Gauss-Seidel or Cholesky with $\Phi_{new}$
  Dictionary update (DU):
    For each mixture $k = 1{:}K$
      $\mathrm{A}_o = \gamma_k\left(\Phi_{old}^T \mathrm{W}_{avg} \Phi_{old} + \mathrm{I}\right)^{-1}$ or $\mathrm{A}_o = 0$ and $\mathrm{C}_o = 0$
      For each utterance $r = 1{:}R$
        $\mathrm{A}_r = \mathrm{A}_{r-1} + \gamma_{rk}\,\beta_r\beta_r^T$
        $\mathrm{C}_r = \mathrm{C}_{r-1} + \gamma_{rk}\eta_r^{(k)}\beta_r^T$
      End for each utterance
      Solve $\Phi_{new}^{(k)}\mathrm{A}_R = \mathrm{C}_R$ using Gauss-Seidel or Cholesky
    End for each mixture
    $\Phi_{new} = [\Phi_{new}^1; \dots; \Phi_{new}^K]$
  End Dictionary Update
```

```
16: End until convergence
```

Table 4.1: Dictionary learning algorithm based on alternating minimization with two steps.

Table 4.1 summarizes the proposed algorithm for the dictionary learning process. Note that throughout the theoretical presentation in Sections 4.2 and 4.3 we have used the dictionary Φ as a wild-card notation to refer to multiple combinations of the loading matrices U, V and D. Hence, the dictionary learning algorithm in Table 4.1 should be applied in a way consistent with the configuration at hand. As it was the case for the hyperparameter estimation procedure in Section 4.2, the formulation presented in this section is only applicable for the decoupled/independent estimation of U, V and D. Therefore, the way to present the data to the dictionary learning algorithm should be consistent with this approach. An experimental analysis regarding the influence of the choice of $\mathrm{A}_R^{(k)}$ in the resulting dictionary Φ is presented in Section 4.5.1. Moreover, the influence in speaker recognition performance is also analyzed.

Scoring

Given two utterances A and B defined by $(\eta_{\mathrm{A}}, \mathrm{W}_A)$ and (η_B, W_B)—after coding them with the dictionary $\Phi_{JFA} = [\mathrm{UVD}]$—we obtain two approximations $\hat{\eta}_A = \Phi_{JFA}\beta_A$ and $\hat{\eta}_B = \Phi_{JFA}\beta_B$. Since some of the atoms in the dictionary are explicitly representing undesired distortions (i.e., columns of U), setting to zero the corresponding entries in β_A and β_B yields a compensated approximation of the signals $\hat{\eta}_{A|c}$ and $\hat{\eta}_{B|c}$. Once these compensated signal approximations are obtained, a similarity measure can be defined by means of the frame normalized inner product

$$score = \frac{1}{T}\, \langle \hat{\eta}_{A|c}, \hat{\eta}_{B|c} \rangle_{\mathrm{W}_{\#}} = \frac{1}{T}\, \hat{\eta}_{A|c}^{T} \mathrm{W}_{\#} \hat{\eta}_{B|c} \tag{4.26}$$

where $W_{\#}$ can be any symmetric positive definite matrix. Immediate candidates are $\mathrm{W}_{A}, \mathrm{W}_{B}$ and W_{UBM}. Note that from the perspective of signal coding, the concepts of model and test segment are blurred since both utterances are represented in the same way. However, if we identify $\hat{\eta}_{A|c}$ as a model and set $\mathrm{W}_{\#} = \mathrm{W}_{B}$ the only difference between (4.26) and the linear approximation of the log-likelihood ratio in (4.18) is the way in which the test segment is encoded. Specifically, the test segment is represented by simply removing its encoding with respect to the atoms in U from η_{B}. A comparison of both approaches is presented in the next section. Finally, another interesting idea that will be explored in the experiments is the effect of normalizing the scores (i.e., using the cosine of the angle between the compensated approximations as the similarity measure).

$$norm\_score = \frac{\langle \hat{\eta}_{A|c}, \hat{\eta}_{B|c} \rangle_{\mathrm{W}_{\#}}}{\langle \hat{\eta}_{A|c}, \hat{\eta}_{A|c} \rangle_{\mathrm{W}_{\#}}^{1/2} \langle \hat{\eta}_{B|c}, \hat{\eta}_{B|c} \rangle_{\mathrm{W}_{\#}}^{1/2}} \tag{4.27}$$

This normalization technique has already produced successful results when used as a kernel for SVMs on the speaker factor space spanned by the columns of V [63]. Moreover, an extension of that work into a new subspace—denoted as total variability space—has validated the excellent discriminative power of this similarity measure [64]. However, to the best of our knowledge, no use of this normalization has been directly studied in the mean supervector space.

4.4. Experimental Setup

4.4.1. Switchboard-I Database (SWB-I)

The Switchboard-I database is comprised of conversational speech between two speakers recorded over landline telephone channels with a sampling rate of 8 KHz. The average duration of each conversation is 5 minutes (approx. 2.5 min per speaker) and each conversation side is recorded in a different file. The total number of speakers in the database is 520 with a balance in gender and recorded into 4856 speech files. The telephone handsets were either electret or carbon button with an approximate proportion of 70% and 30% respectively.

4.4.2. Configuration of Recognition System

Each file in the database was parameterized into a sequence of 19-dimensional MFCC vectors using a 20ms Hamming window with a 10ms shift. The MFCC vectors were computed using a simulated triangular filterbank on the FFT spectrum. Prior to projecting the Mel-frequency band (MFB) energies into a DCT basis, bandlimiting was performed by discarding the filterbank outputs outside of the frequency range 300Hz-3138Hz. Finally, after projecting the MFB energies into a DCT basis and discarding C0, the 19-MFCC vectors were augmented with delta features resulting in $F = 39$ coefficients per frame.

SWB-I was partitioned into two sets, *P1* and *P2*, of 260 speakers each with a balance in gender and handset type. A 2048-mixture gender-independent UBM with diagonal covariance matrices was trained on *P2*. The data in *P2* was also used for hyperparameter/dictionary learning. In particular, we used an eigenchannel setup $\Phi_{ECH} = [\mathrm{UD}]$ with $KF = 77824$, $U \in \mathbb{R}^{KF \times P}$ and the standard relevance-MAP diagonal matrix D was fixed to $\mathrm{D}^2 = \Sigma/\tau$ with $\tau = 16$. This configuration is general enough to validate our theoretical developments while avoiding unnecessary complexity in illustrating the underlying principles of the proposed techniques.

4.5. Experiments

In order to evaluate the theoretical exposition of the previous section we present three different sets of experiments. The first one is concerned with the effects of different DU steps in the learned dictionaries as well as the effect in speaker recognition accuracy. The second set of experiments is designed to evaluate the influence of different signal coding strategies along with various types of inner products for scoring. Finally, the third batch of experiments analyzes the influence of the normalization of the scores according to (4.27) in a verification task and compares our proposed similarity measure with the linear approximation of the log-likelihood introduced in [59].

4.5.1. Analysis of Dictionary Learning Procedure

Equation (4.17) from the JFA model as well as equations (4.24) and (4.25) from the SC model provide three different DU mechanisms. We will refer to the updates in (4.17), (4.24) and (4.25) as Full, Zero and Average updates respectively. This notation stems from the fact that (4.17) takes a full account of the posterior covariance matrices; (4.24) can be understood as setting them to zero; and (4.25) considers a common and averaged covariance matrix for all utterances in the dictionary training set. The computational advantages of (4.24) and (4.25) over (4.17) were briefly discussed in Section 4.3.2. However, the effects of this computational saving in the learned dictionaries are not evident and thus require some experimental analysis. We would like to know how the dynamics of the sequence of dictionaries generated by multiple iterations of the dictionary learning algorithm in Table 4.1 are affected. To study this, we apply the dictionary learning algorithm with the full, average and zero updates to obtain a sequence of eigenchannel subspaces $\mathrm{U}_F(i), \mathrm{U}_A(i)$ and $\mathrm{U}_z(i)$ with $i = 0, \dots, 10$. The 2411 utterances coming from the 260 speakers of $P2$ where used for each iteration. To quantify the similarity between two subspaces, we used a metric between the subspaces spanned by the columns of the matrices $\mathrm{A} \in \mathbb{R}^{FK \times P_A}$ and $\mathrm{B} \in \mathbb{R}^{KF \times P_B}$ known as the projection distance [65]

$$pdist(\mathrm{A}, \mathrm{B}) = \|\mathrm{A}\mathrm{A}^T - \mathrm{B}\mathrm{B}^T\|_F^2. \tag{4.28}$$

Since the projection distance is at most the $\min(P_A, P_B)$, we normalized (4.28) to produce results between $[0,1]$.

Figure 4.1 shows the projection distance of the average and zero updates with respect to the full update. The curves with the triangle markers refer to the distance between the full and average updates. The curves with the asterisk indicate the distance between the full and the zero update. Moreover, the color codes refer to the dimension of the subspaces computed (i.e., blue=128, green=64 and red=32 dimensions). A simple look at the y-axis shows that the normalized projection distances are very low for all configurations (since the maximum possible value is 1). Furthermore, the larger the dimensionality of the subspaces the larger the projection distance. As expected, the distance of the subspaces produced by the average update is smaller than those produced by the zero update. These results confirm that the three DU techniques produce very similar dictionaries.

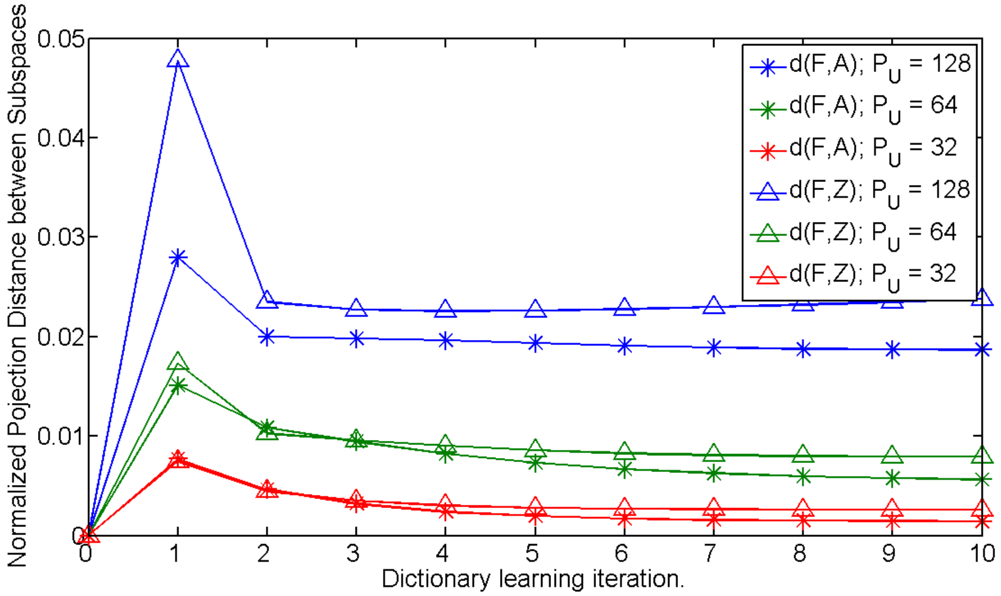


Figure 4.1: Normalized projection distance between subspaces of dimensions 128, 64 and 32 learned using different DU formulas.

Even though the distance between subspaces might not be too large, the effects in the recognition accuracy may not behave in the same way. To check this, a closed-set identification experiment was used. We coded each of the 2408 utterances from partition *P1* using the dictionaries obtained after the 6th iteration. The normalized score in equation (4.27) was used with the inner product defined by the weights and covariance matrices of the UBM. We obtained 33866 identification trials based on the 2408 utterances. The details about how we constructed these trials are provided in next section. Table 4.2 shows that the effect in identification accuracy is negligible. Hence, we can claim that for a scenario where enough utterances are available for dictionary training, the average and zero update rules provide computational advantages without any significant loss in performance.

P_U	Full $A_R^{(k)}$	Avg. $A_R^{(k)}$	Zero $A_R^{(k)}$
128	95.0%	94.9%	94.9%
64	94.5%	94.5%	94.5%
32	93.3%	93.3%	93.3%

Table 4.2: Closed-set identification accuracy for dictionaries learned with three DU formulas (full, average and zero). Three dimensios of the eigenchannel space U are presented.

4.5.2. Closed-set Speaker Identification

This section explores the influence of different signal coding strategies along with various types of inner products in the context of speaker identification. We intentionally selected an identification setup in order to remove the influence of a verification threshold from the analysis. We obtained 33866 identification trials based on the 2408 utterances from 260 speakers in *P1*. The protocol followed was as follows: for each speaker we picked one of its utterances and encode it to represent a model, and then, another utterance form that same speaker was selected as the test segment; the remaining utterances from the rest of the speakers were used as models. This procedure was repeated exhaustively for all the utterances of each speaker and for all the speakers. The dimensionality of the eigenchannel space was explored and 128 dimensions produced the best results. Also, the average update rule was used in the learning process.

Figure 4.2 shows the influence of three different inner products in our SC formulation with normalized scoring (middle left panel). The three inner products are defined by the matrices $W_I = I$, W_{ubm} and W_{test}. The last two have already been discussed and the first one indicates the standard inner product. For comparison, we also analyze the influence of these inner products in other techniques such as: ML model training (top left), relevance MAP (top right), and the standard eigenchannel configuration with linear scoring (middle right). A general trend is observed regardless of the modeling technique used; the use of the standard inner product performs much worse in all cases. This makes sense since not all the information is evenly distributed across the acoustic space. Therefore, penalizing by the amount of data (i.e., small value of the first order statistics) as well as the variability within the soft regions associated with each Gaussian (i.e., covariance of the UBM) is very effective. This concept is not new and has been exploited in the formulation of the KL-kernel (i.e., inner product defined by W_{ubm}) in [35]. The results obtained with W_{ubm} and W_{test} change depending on the modeling strategy followed. For our SC approach, the use of both inner products produces comparable results. However, for the standard eigenchannel model with linear scoring, W_{test} produces significantly better results (and in the same range as the SC approach). The sensitivity with respect to the inner product is understandable since the linear scoring is an approximation of the log-likelihood ratio and by changing the inner product the approximation is less accurate.

After the first two iterations not much difference is obtained in the identification performance. This extremely fast convergence might be explained by the fact that the dictionary training data and the test set are very similar. Also, identification results based on only the factors (bottom right) and the information in the eigenchannel subspace (bottom left) are included for diagnostic purposes. In particular we can observe that the eigenchannel subspace also contains speaker information since an accuracy of almost 70% is obtained. The factors x and z behave as expected.

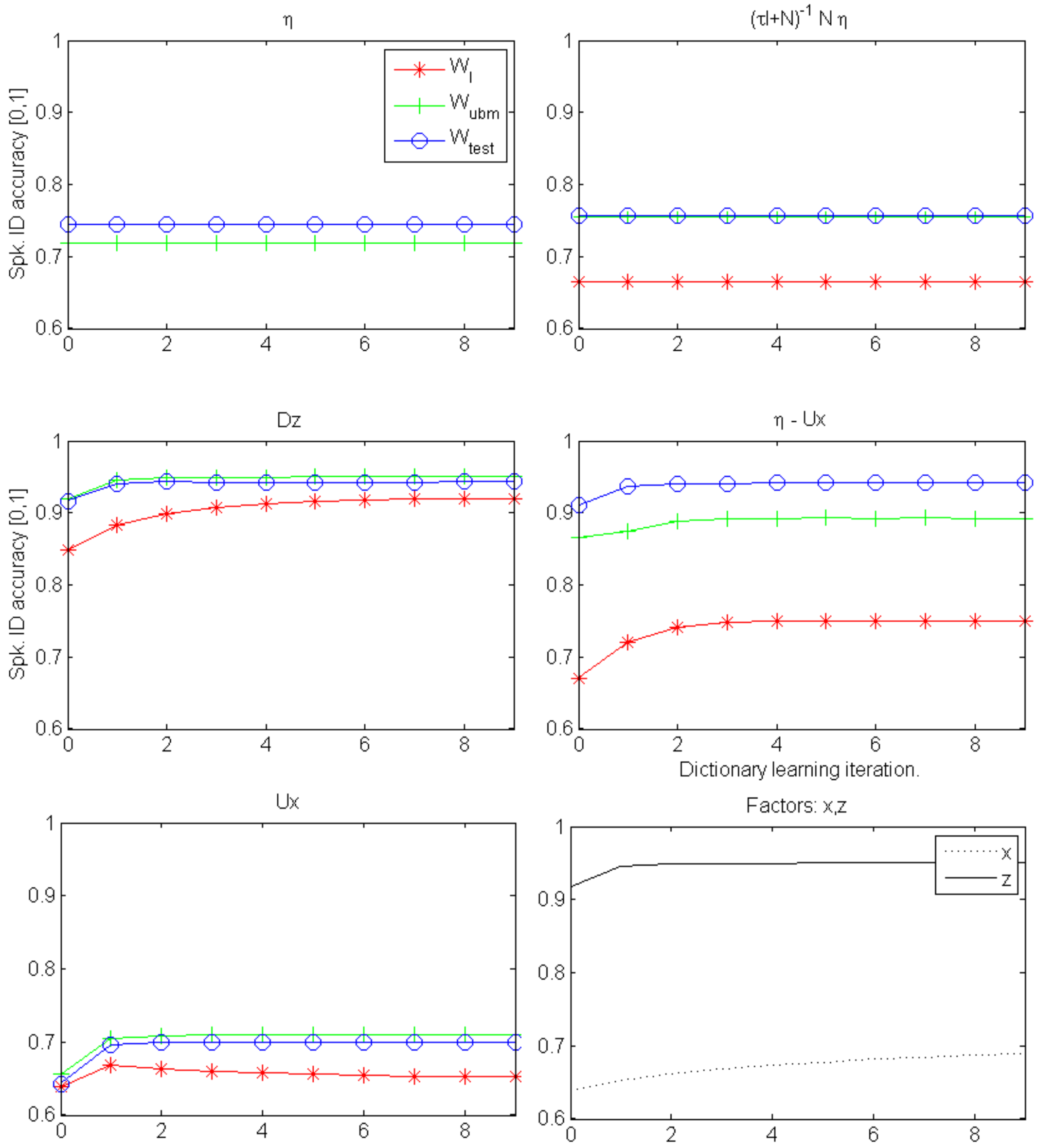


Figure 4.2: Closed-set identification results for six different modeling approaches (see main text for description) along with three different scoring techniques based on the inner products defined by the symmetric positive definite matrices W_I, W_{ubm}, W_{test}.

Finally, the performance of the normalized and un-normalized scoring techniques was assessed. No significant difference was observed for neither the SC approach nor the standard eigenchannel formulation. This makes sense since for identification purposes what matters is the relative positioning of scores and not their scaling. In the next section we explore this issue in the context of speaker verification where the scaling of the scores is critical.

4.5.3. Speaker Verification

Based on the 2408 utterances from the 260 speakers in *P1* a verification experiment was designed. Specifically, a leave- one-out strategy was used. That is, each file was used as a model and the remaining 2407 utterances were used as test segments. This protocol produced a great number of trials (33,866 target and 5,764,598 non-target). However, since our proposed scoring as well as the linear scoring methods are simple inner products between high dimensional vectors, the entire set of trials was computed in a less than 5 minutes. The main purpose of this setup was to assess the influence of the score normalization proposed in (4.27).

Figure 4.3 shows the verification results. Three observations are in place. First, using the cosine of the angle between the vectors results in more than a 25% relative improvement in EER for both linear scoring in (4.17) and the proposed un-normalized inner product of (4.26). Second, the effects of the normalization are slightly better for our approach. Finally, while the performance of the un-normalized scores is better for the linear scoring, the normalized SC scores produce slightly better performance under normalization.

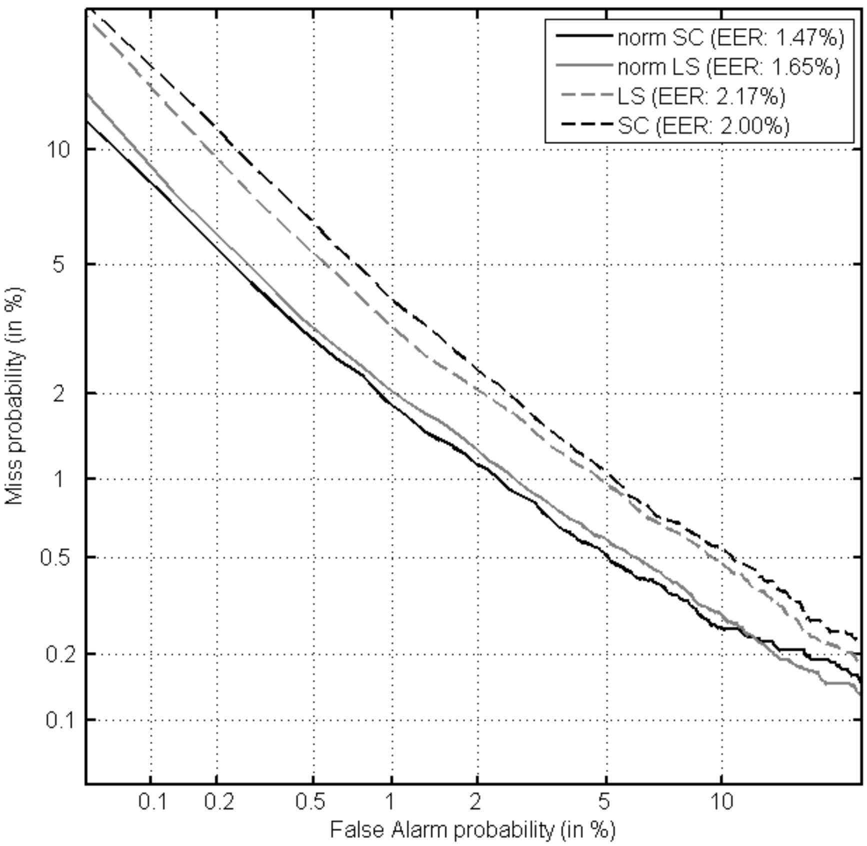


Figure 4.3: Verification results for different scoring methods.

4.6. Chapter Summary

In this chapter, we have established a connection between the Joint Factor Analysis paradigm for speaker recognition and signal coding using an overcomplete dictionary learned from data. The probabilistic concepts of model training, hyperparameter estimation and likelihood ratio computation were equated to the non-probabilistic notions of signal coding, dictionary learning, and similarity computation respectively. Two novel ideas were proposed that resulted in algorithmic improvements. The first idea provided computational improvements by allowing a faster estimation of the JFA model hyperparameter. The second idea provided an alternative scoring technique with performance improvements. Specifically, the proposed technique for hyperparameter estimation was able to avoid the need for explicit matrix inversions in the M-step of the ML estimation. This allowed the use of faster techniques such as Gauss-Seidel or Cholesky factorizations for the computation of the posterior means of the factors x, y and z during the E-step. Regarding the scoring, different similarity measures based on inner products—defined by symmetric positive definite matrices derived from data—were studied. A simple normalization technique of these inner products was shown to improve the verification performance of our recognition system using a dictionary comprised of eigenchannels and a fixed relevance-MAP matrix D. Based on this experimental setup, slightly better results than those produced by the state-of-the-art linear scoring approach were reported. The experimental validation of these two novel techniques was presented using closed-set identification and speaker verification experiments over the Switchboard database.

Chapter 5

Speaker Recognition Based on I-vector Representations

5.1. Introduction

The use of speaker recognition systems based on supervector representations modeled by Joint Factor Analysis (described in Chapter 4) advanced the state-of-the-art significantly from 2004 until 2008. However, the fact that using the projection onto the inter-session subspace (i.e., channel factors) to perform speaker recognition produced results better than chance motivated the introduction of Factor Analysis (FA) of supervectors as an unsupervised dimensionality reduction technique [2], [3]. The computed factors were denoted as "i-vectors" and explicit modeling of speaker-specific and inter-session variability was performed in this lower-dimensional space (i-vector space). The initial formulation to compensate for undesired variability made use of a combination of Linear Discriminant Analysis (LDA) along with the use of

Within Class covariance Normalization (WCCN) [3]. Moreover, a verification score was computed by means of a cosine similarity between the compensated i-vectors.

While this approach improved the performance with respect to JFA, more principled approaches based on a probabilistic generative model of i-vectors were suggested in [66] and [4]. These models adapted the Probabilistic Linear Discriminant Analysis (PLDA) model introduced in [67] to the task of speaker recognition[7]. A common theme among these probabilistic approaches is that they ignore the process by which i-vectors were extracted (i.e., FA model) and instead pretend they were generated by a prescribed generative model. The distinguishing factor between these approaches is the set of assumptions embedded in the model. The two most commonly used assumptions are that: i) the speaker and channel components are statistically independent; and ii) they are Gaussian distributed. The main advantage of these assumptions is that the speaker detection likelihood ratios can be obtained in closed-form. The work in [66] is an example of a Gaussian-PLDA (G-PLDA) model.

Alternatively, the Heavy-Tailed PLDA model (HT-PLDA) presented in [4] replaced the Gaussian priors by a Student's *t* distribution. Two main motivations were behind this approach. First, to allow for larger deviations from the mean (e.g., severe channel distortions). Second, to increase the robustness to outliers in the ML estimation of the model parameters. Since no closed-form solution of the speaker detection likelihood ratio is obtained when using the heavy-tailed priors, variational Bayes was used to approximate it [36]. The results presented in [4] showed superior performance of the HT-PLDA model over the Gaussian prior based alternative;

[7] PLDA was proposed in the context of face recognition and corresponds to a particularization of the JFA model to the case of a single Gaussian.

hence, providing strong empirical evidence towards non-Gaussian behavior of speaker and channel effects.

In this chapter we pursue an alternative approach to deal with the non-Gaussian behavior of the i-vectors. That is, we keep the Gaussian assumptions in the model, but perform a non-linear transformation of the i-vectors to reduce the non-Gaussian behavior (i.e., i-vector Gaussianization). The goal is to obtain a system that matches the high performance of the more complex HT-PLDA model while maintaining the simplicity and high scalability of the G-PLDA model. The rest of this chapter is organized as follows. Section 5.2 presents a formal mathematical description of how i-vectors are computed as well as the two variants of the PLDA model. Section 5.3 describes the novel non-linear transformations proposed in this thesis. Section 5.4 analyzes the behavior of the proposed transformations and demonstrates their excellent performance on the telephone portion of NIST SRE 2010. Finally, Section 5.5 summarizes the contributions of this chapter.

5.2. Background Knowledge

5.2.1. I-vector Extractor

An i-vector extractor [3] is a system that maps a sequence of feature vectors $\mathcal{O} = \{o_t\}_{t=1}^{T}$ with $o_t \in \mathbb{R}^F$, obtained from a speech utterance, to a fixed-size vector $\eta \in \mathbb{R}^D$. An L-component GMM, $\lambda_{UBM} = \left(\{w_{UBM}^{l}\}, \{m_{UBM}^{l}\}, \{\Sigma_{UBM}^{l}\}\right)$, denoted as Universal Background Model (UBM) is used to collect zero- and first-order Baum-Welch sufficient statistics from the utterance: $\{N_l\}$ and $\{\theta_l\}$ for $l = 1 \dots L$, where

$N_l = \sum_t N_{lt}$; $\theta_l = (\sum_t N_{lt}\, o_t)/N_l$; and N_{lt} is the soft occupation count of mixture l for frame o_t. Subsequently, a supervector $\theta = [\theta_1^T, \dots, \theta_L^T]^T \in \mathbb{R}^{FL}$ is constructed by appending together the first-order statistics for each mixture component and is assumed to obey an affine linear model (i.e., factor analysis (FA) model) of the form:

$$\theta = m_{UBM} + \mathrm{T}x + \epsilon_{UBM} \tag{5.1}$$

where the supervector $m_{UBM} \in \mathbb{R}^{FL}$ is formed by appending the means of the UBM; the columns of the low-rank matrix $\mathrm{T} \in \mathbb{R}^{FL\times D}$ span the subspace where most of the speaker-specific information lives (along with channel variability); x is a standard-normally distributed latent variable; and ϵ_{UBM} is a Gaussian noise term with zero mean and precision matrix $\mathrm{W} \in \mathbb{R}^{FL\times FL}$. The diagonal matrix $\mathrm{W} \in \mathbb{R}^{FL\times FL}$ (or block-diagonal in case the UBM comprises full-covariance matrices) is constructed by multiplying each of the L inverse covariance matrices of the UBM by the corresponding zero-order statistic N_l and placing the resulting matrices as part of the block-diagonal entries of W. For each speech utterance, an i-vector η is obtained as the MAP point estimate of x :

$$\eta = x_{MAP} = (\mathrm{I} + \mathrm{T}^T\mathrm{W}\mathrm{T})^{-1}\mathrm{T}^T\mathrm{W}(\theta - m_{UBM}). \tag{5.2}$$

The matrix T is learned from a large collection of representative data by ML estimation [3].

The i-vector extraction process is unsupervised with respect to the identity of the speaker in each utterance. Therefore, it can be regarded as a simple unsupervised data-driven dimensionality reduction technique that maps supervectors into i-vectors. In practice, the typical dimensionality of i-vector spaces ranges from 400 to 800 dimensions [3], [68]. For a supervector space derived from 60-dimensional feature vectors and a 2048 mixture UBM, the compression ratio is approximately 300:1. Despite this remarkable dimensionality reduction, the data-driven nature of the process allows i-vector representations to capture a great amount of the speaker-specific information contained in the speech utterances.

5.2.2. Gaussian Probabilistic Linear Discriminant Analysis (G-PLDA)

A common theme among probabilistic approaches in i-vector space is that they ignore the process by which i-vectors were extracted (i.e., MAP point estimates of latent variables in a FA model) and instead consider them as observations from a prescribed *probabilistic generative model*. In this section we focus on the Probabilistic Linear Discriminant Analysis (PLDA) model proposed in [67] since its adaptation to the field of speaker recognition represents the state-of-the-art [69], [68], [66]. In the following we present the details about the PLDA model as well as how to use it to compute verification scores.

i) *Modeling*: PLDA regards an i-vector as the sum of a speaker-specific component and an undesired variability component (often referred to as the channel component or inter-session variability). Moreover, the speaker and channel components are assumed statistically independent and Gaussian distributed. In particular, assuming R utterances for speaker i, and denoting the corresponding collection of i-vectors as $\{\eta_{i,r}\}$, with $r = 1, \dots, R_i$, the PLDA paradigm models an observed i-vector $\eta_{i,r}$ as:

$$\eta_{i,r} \;= m + \Phi\beta_i + \Gamma\alpha_{i,r} + \;\epsilon_{i,r}. \tag{5.3}$$

The speaker-specific part $s_i \;= m + \Phi\beta_i$ describes the between-speaker variability and does not depend on the particular utterance. The channel component $c_{i,r} = \Gamma\alpha_{i,r} + \;\epsilon_{i,r}$ is utterance dependent and describes the within-speaker variability. Specifically, m is a global offset common to all speakers; the columns of Φ provide a basis for the speaker-specific subspace (eigenvoices); β is a latent identity vector having a standard normal distribution; the columns of Γ provide a basis for the channel subspace (eigenchannels); α is a latent vector having a standard normal distribution; and ϵ is a residual term assumed to be Gaussian with zero mean and diagonal covariance Σ. Note that the distributions of both α and ϵ are the same for all speakers and utterances. Also, all latent variables are assumed statistically independent. Since the i-vectors considered in this work are of sufficiently low dimensionality (i.e., 400 for our experiments), we follow the modification used in [68] and assume that Σ is a full-covariance matrix and remove the eigenchannels from (5.3). Thus, the modified PLDA model used in this work follows:

$$\eta_{i,r} \;= m + \Phi\beta_i + \;\epsilon_{i,r}. \tag{5.4}$$

Equation (5.4) can also be considered as a generalization of a FA model where the noise term is full-covariance instead of diagonal. The ML point estimates of the model parameters $\{m, \Phi, \Sigma\}$ are obtained from the development data using an EM algorithm as in [67].

ii) *Verification score*: A verification trial comprises an identity claim and a test utterance (i.e., a person claims to be speaker i and provides a speech utterance). In the multicondition training setup of this work, a model for speaker i is represented by a collection of K i-vectors $\{\eta_{i,r}\}$, with $r = 1, \dots, K$. Also, the test segment is mapped into an i-vector denoted as η_T. In the PLDA framework, a verification score is computed as the log-likelihood ratio between two alternative hypothesis: the same-speaker hypothesis $\mathcal{H}_s$, and the different-speaker hypothesis $\mathcal{H}_d$. Under the same-speaker hypothesis, the collection of i-vectors from the claimed identity and the test segment are assumed to follow the generative model:

$$\begin{bmatrix} \eta_{i,1} \\ \vdots \\ \eta_{i,K} \\ \eta_T \end{bmatrix} = \begin{bmatrix} m \\ \vdots \\ m \\ m \end{bmatrix} + \begin{bmatrix} \Phi \\ \vdots \\ \Phi \\ \Phi \end{bmatrix} \beta_i + \begin{bmatrix} \epsilon_{i,1} \\ \vdots \\ \epsilon_{i,K} \\ \epsilon_T \end{bmatrix} \tag{5.5}$$

or

$$\eta' \ = \ m' + \Phi' \beta_i \ + \ \epsilon'$$

On the other hand, for the different-speaker hypothesis, the i-vectors are assumed to follow:

$$\begin{bmatrix} \eta_{i,1} \\ \vdots \\ \eta_{i,K} \\ \eta_T \end{bmatrix} = \begin{bmatrix} m \\ \vdots \\ m \\ m \end{bmatrix} + \begin{bmatrix} \Phi & 0 \\ \vdots & \vdots \\ \Phi & 0 \\ 0 & \Phi \end{bmatrix} \begin{bmatrix} \beta_i \\ \beta_j \end{bmatrix} + \begin{bmatrix} \epsilon_{i,1} \\ \vdots \\ \epsilon_{i,K} \\ \epsilon_T \end{bmatrix}, \tag{5.6}$$

or

$$\eta' \ = \ m' \ + \ \Phi'' \beta'' \quad + \ \epsilon''$$

where $i \neq j$. Hence, in (5.5) all the i-vectors are generated using the same latent identity variable β_i, whereas in (5.6) two different identity variables are involved. Based on the Gaussian assumptions about the noise and latent identity variables, and the independence of the speaker-specific component and channel component, it follows that the verification score is the ratio of two Gaussian distributions:

$$
\begin{aligned}
score &= \log p\left(\{\eta_{i,r}\},\eta_T \middle| \mathcal{H}_s\right) \; - \log\, p\left(\{\eta_{i,r}\},\eta_T \middle| \mathcal{H}_d\right) \\
&= \; \log \mathcal{N}(\eta';m',\Phi'\Phi'^{\mathrm{T}}+\Sigma') - \log \mathcal{N}(\eta';m',\Phi''\Phi''^{\mathrm{T}}+\Sigma'),
\end{aligned}
\tag{5.7}
$$

where Σ' is a block diagonal matrix with $K+1$ copies of the noise covariance matrix Σ in the diagonal. Hence, the log-likelihood ratio involves two Gaussian distributions with the same means but different covariance matrices. The verification score can be computed very efficiently by making use of the matrix inversion lemma—avoiding the explicit computation and storage of the high-dimensional covariance matrices.

It is important to note that the verification score is not based on point estimates of the latent identity variables; but on the Bayesian principle of marginalization over latent variables. This paradigm acknowledges the uncertainty of the inference process. Hence, the question we are asking during verification is whether the observed i-vectors were generated from the same identity variable or not; regardless of what the actual identity was. Therefore, a high verification score means that it is more likely that the observed i-vectors were generated by the same speaker (i.e., a single latent identity variable) than from two different speakers (i.e., two distinct latent identity variables).

When training a classifier based on a PLDA model with a multicondition data set, we will refer to it as Pooled-PLDA. This will help to differentiate it from a single-condition PLDA model, and to emphasize that we are pooling the data together to obtain a common set of parameters for all the conditions or subset of conditions used to train the system. We will make use of this terminology in Chapter 6.

5.2.3. Heavy-Tailed Probabilistic Linear Discriminant Analysis (HT-PLDA)

The HT-PLDA model was first introduced in [4]. While the general formulation includes a channel subspace, the HT-PLDA model used in our experiments will be based on (5.4) but with priors on β and ϵ_r following multivariate Student's *t* distributions rather than Gaussian. Precisely, β is assumed to have zero mean, identity scale matrix and n_β degrees of freedom. Also, ϵ_r is assumed to have zero mean, full scale matrix Σ and n_{ϵ_r} degrees of freedom. It is important to note the number of degrees of freedom parameter controls the behavior of the tails of the distribution. That is, the smaller the number of degrees of freedom the heavier the tails. On the other hand, as the number of degrees of freedom increases the Student's *t* distribution converges to a Gaussian distribution [36]. In this way, one can think of the G-PLDA model as a particularization of the HT-PLDA model where the number of degrees of freedom grows to infinity.

For the HT-PLDA model, the log-likelihood ratio in (5.7) does not have a closed form solution. In [4], a variational lower bound was used as a proxy for each of the marginal likelihoods (i.e., evidence) involved in the log-likelihood ratio. In this way, each verification score requires much more computation than in the case of G-PLDA.

5.3. I-vector Transformations

The results presented in [4,69] showed superior performance of the HT-PLDA model over G-PLDA for the telephone conditions of NIST SRE 2010. This provides strong empirical evidence of non-Gaussian behavior of speaker and channel effects in i-vector representations. However, due to the simplicity and computational efficiency of G-PLDA we are interested in keeping the Gaussian assumptions in the model and performing a transformation of the i-vectors to reduce the non-Gaussian behavior. A successful i-vector transformation should close the gap in performance between HT-PLDA and G-PLDA.

5.3.1. Radial Gaussianization (RG)

Besides the Gaussian prior assumption, the statistical independence between speaker and channel factors is also questionable. As noted in [4], the success of cosine scoring [2] suggests that there is a principal axis of channel variation that is dependent on the speaker identity. Thus, if we drop the independence assumption and keep the multivariate Student's t distribution assumption for the prior on the latent variables, the generative i-vector model can be expressed as:

$$\eta \ = m + \Omega z, \tag{5.8}$$

where the latent variable z now represents both the speaker and channel factors and follows a Student's t distribution. In this way, η is nothing more than an affine transformation of a multivariate Student's t distribution which belongs to the family of Elliptically Symmetric Densities (ESD) [70]. Thus, a transformation of the i-vectors—that renders the Gaussian and statistical independence assumptions appropriate—needs to be able to transform members of the ESD family into a Gaussian distribution. As pointed out in [70], linear transforms have no effect on the dependencies beyond second order for ESD. Thus, if z follows a multivariate

Student's t distribution, we need to resort to non-linear transformations to accomplish our goal. Fortunately, an effective technique denoted as Radial Gaussianization (RG) was proposed in [70]. This technique follows a two step process. First, the ESD is transformed into a Spherically Symmetric Density (SSD) by removing the mean and applying a linear whitening transformation learned from data samples. Second, a non-linear histogram warping of the length distribution of the centered and whitened variable η_{wht} is performed (this second step stems from the fact that the length of vectors drawn from a standard Gaussian distribution follows a Chi distribution with degrees of freedom equal to the dimension of the vector). In particular, the length transformation function is given by

$$g(\|\eta_{wht}\|) \ = F_{\mathrm{X}}^{-1} F_r(\|\eta_{wht}\|). \tag{5.9}$$

This is nothing more than the function composition of the inverse cumulative Chi distribution with the cumulative distribution of the length random variable $r = \|\eta_{wht}\|$. In practice, F_r needs to be estimated from data.

5.3.2. Length Normalization (LN)

The need to estimate the cumulative distribution of the length random variable $r = \|\eta_{wht}\|$ in the RG process requires the use of a held-out set of data. To avoid this constrain, we propose to simplify the second step of the RG process and simply scale the length (i.e., norm) of each centered and whitened i-vector η_{wht} to unit length. This approximation is very reasonable in spaces of high dimensionality (such as i-vector spaces with typical dimensionality in the range of 400 to 800), since in high dimensions most of the probability mass of a standard Gaussian distribution is concentrated on a thin shell around a hyper-sphere (see [36], pp. 36-37 for more details). Moreover, the thickness of the shell decreases with the dimensionality of the space. Hence, projecting the data onto a hyper-sphere (whose radius depends on the dimensionality of the space) results in an approximation of the second step of RG that becomes more exact as the dimensionality of the space increases. Also, using (5.7) it is easy to show that the radius of the hyper-sphere is unimportant since it only translates into an offset in the verification score. Therefore, we will use a unit radius hyper-sphere for convenience. In particular, for an i-vector space of dimension 400, the distribution of the lengths of data drawn from a standard Gaussian follows a Chi distribution with 400 degrees of freedom. Figure 5.1 shows this distribution (black curve) and we can observe that the radius of the hyper-sphere is approximately equal to 20 (i.e., the mode of the distribution) and most of the probability mass is concentrated around it.

To summarize, the length normalization transformation involves two steps:

Step 1: Centering and whitening

- Compute mean and sample covariance $\left(\hat{m}, \hat{\mathrm{S}}\right)$ from development data
- Obtain whitening transformation $\mathrm{A} = \mathrm{D}^{-\frac{1}{2}}\mathrm{U}^{\mathrm{T}}$ with $\hat{\mathrm{S}} = \mathrm{UDU}^{\mathrm{T}}$
- Center and whiten i-vector: $\eta_{wht} = \mathrm{A}(\eta - \hat{m})$

Step 2: Scaling

- Project onto unit sphere: $\eta_{LN} = \frac{\eta_{wht}}{\|\eta_{wht}\|}$

Finally, during verification, the transformation learned from the development data is applied to both development and evaluation data.

5.4. Experiments

In this section we present an experimental validation of the benefits of i-vector transformation in speaker verification performance. The following section provides details about the experimental setup used throughout all the experiments.

5.4.1. Setup

The NIST SRE 2010 data from the extended-core telephone-telephone condition (i.e., condition 5) was used. Throughout the experiments we refer to this set as the evaluation data. Verification performance is reported in terms of Equal Error Rate (EER) as well as the Detection Cost Function (DCF) defined by ($C_{MISS} = 1$, $C_{FA} = 1$ and $P_{tar} = 0.001$).

For all our experiments, we have used the i-vectors provided by Brno University of Technology (BUT) [69]. They are extracted using a 20ms short-time Gaussianized MFCCs plus delta and double-delta. A full-covariance gender-independent UBM with 2048 mixtures was trained from NIST SRE 04 and 05 telephone data. A gender-dependent i-vector extractor was trained from telephone data from: NIST SRE 04, 05, 06, Switchboard and Fisher. The dimension of the i-vectors is 400. Both the G-PLDA and HT-PLDA model parameters were estimated from the same data used in the i-vector extractor (excluding data from Fisher database since it was found in [69] to deteriorate the verification performance). We refer to this set as development data. The number of eigenvoices in G-PLDA was set to 120. Also, in order to reduce the computational cost of the HT-PLDA system, a LDA dimensionality reduction to 120 dimensions was used prior to any other processing of the i-vectors. The number of eigenvoices was also set to 120 as in the G-PLDA case. No score normalization is used in the reported results since it did not help improve the performance.

5.4.2. I-vector Length Analysis

As mentioned before, the length of vectors drawn from a standard Gaussian distribution follows a Chi distribution with number of degrees of freedom (DOF) equal to the dimension of the vector (i.e., 400). In principle, an i-vector extractor is supposed to generate i-vectors that behave in this way (especially if a Minimum Divergence [56] step is used).

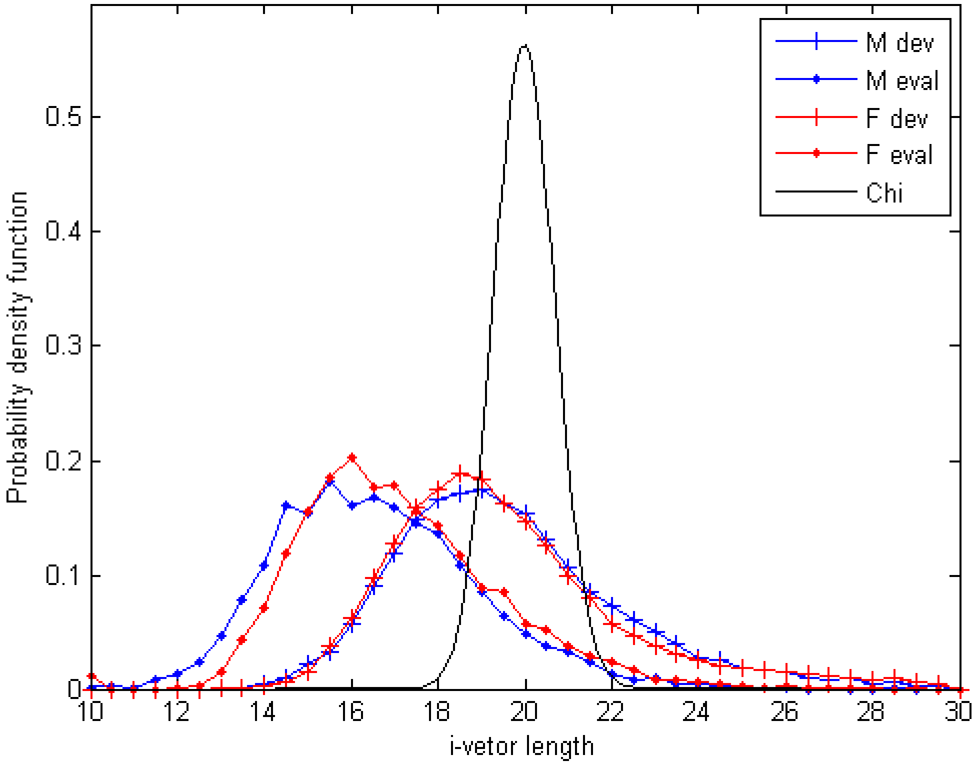


Figure 5.1: Histograms of the i-vector length distribution for development and evaluation data separated by gender: male (M) and female (F). Also the probability density function of a Chi distribution with 400 degrees of freedom is depicted.

Figure 5.1 depicts the probability density function of a Chi distribution with 400 DOF. Also, histograms of the i-vector length distribution for development and evaluation data (separated by gender) are presented. Three important observations are in order. First, neither the development data not the evaluation data match the Chi distribution. Second, the behavior for both genders is similar. Third, and most important, there is a significant mismatch between the length distributions of the development and evaluation i-vectors. This is not surprising since the i-vector extractor is trained on the development set and therefore fits this data set better than the evaluation set. Hence, when considering both development and evaluation data together, the bimodal distribution of the lengths indicates non-Gaussian behavior in the i-vectors. Although not surprising, this behavior is undesirable—especially if we are interested in using a simple G-PLDA model—since the mismatch can be considered as a strong source of heavy-tailed behavior. To further investigate the effects of this mismatch, we used the HT-PLDA system and checked how the ML estimates of the degrees of freedom parameters n_β and n_ϵ behaved as we transformed both the development and evaluation i-vectors by RG and length (L) normalization. The results are summarized in Table 5.1. We can observe that the behavior between male and female speakers is consistent. More interestingly, both RG and LN transformations increase the value of n_β and decrease the value of n_ϵ when compared to the original i-vectors. This indicates that the transformations make the HT-PLDA more like a partially-HT model where the eigenvoices have lighter tails (i.e., more Gaussian) and the residual shows a stronger heavy-tailed behavior. Also, the LN normalization seems to induce a more extreme behavior.

Transformation type	Eigenvoices (n_β)		Residual (n_ϵ)	
	Male	Female	Male	Female
dev_eval	11.09	12.39	17.10	17.42
RG-dev_RG-eval	25.35	27.30	13.24	14.81
LN-dev_LN-eval	48.07	54.71	9.21	10.42

Table 5.1: Value of the degrees of freedom parameters for i-vector transformations in the HT-PLDA system.

5.4.3. Speaker Verification Results

Table 5.2 summarizes the results for multiple combinations of transformations of development and evaluation sets. For the G-PLDA system both the RG and LN transformation provide an impressive improvement over the unprocessed i-vectors both in EER and in DCF. Also, the simpler length normalization achieves equivalent performance to the RG technique with the advantage of not requiring a held-out set to estimate the empirical cumulative distribution of the lengths.

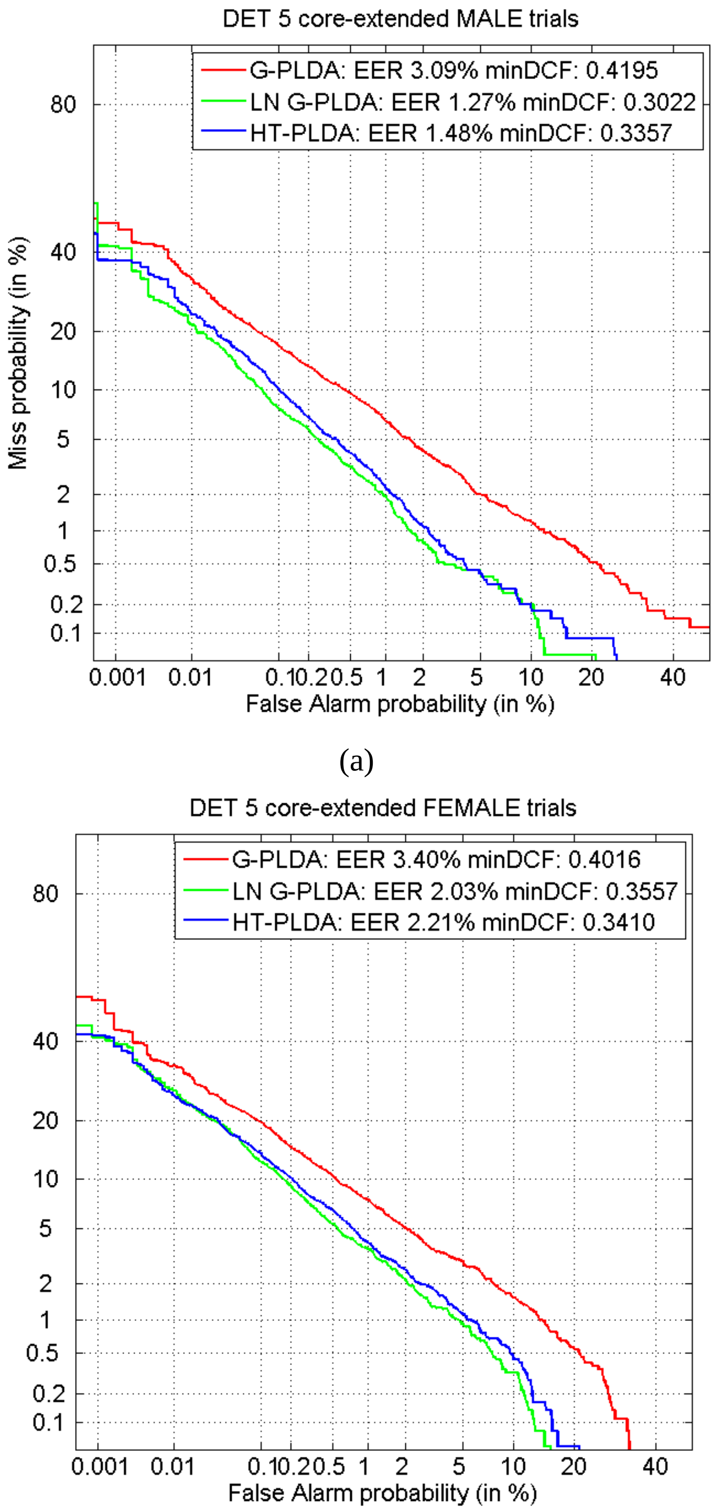
System codes	Male scores		Female scores	
	EER(%)	minDCF	EER(%)	minDCF
U_U G-PLDA	3.08	0.4193	3.41	0.4008
U_RG G-PLDA	1.44	0.3032	2.15	0.3503
U_LN G-PLDA	1.29	0.3084	1.97	0.3511
LN_LN G-PLDA	**1.27**	**0.3019**	2.02	0.3562
RG_RG G-PLDA	1.37	0.3066	2.16	0.3393
U_U HT-PLDA	1.48	0.3357	2.21	0.3410
LN_LN HT-PLDA	1.28	0.3036	**1.95**	**0.3297**
RG_RG HT-PLDA	**1.27**	0.3143	**1.95**	0.3339

Table 5.2: Verification results for extended-core condition 5 of NIST SRE 2010. G-PLDA and HT-PLDA systems are evaluated with various combinations of i-vector transformations. Both systems use 120 eigenvoices and full-covariance residual. The top 5 rows correspond to G-PLDA system and the lower 3 to HT-PLDA system. The system codes correspond to: dev_eval system. For example, the first row indicates that both the dev and eval data were not transformed and the system was G-PLDA.

Another interesting observation is that as long as the evaluation data is transformed, keeping the development i-vectors in their original form does not affect the performance much (rows 2 and 3). Thus, the key step is the transformation of the evaluation data i-vectors. In the case of length normalization, this can be explained by taking a look at the scoring equation (5.7) and noting that a global scaling of the length of all the evaluation i-vectors only produces a global scaling of the scores (i.e., it does not alter the relative position of the scores). Hence, once the length normalization has been applied, instead of unit-length we can select the target length to match the mode of the development data distribution. In this way, we have greatly eliminated the mismatch and the results should reflect that. Thus, the choice of unit length is an arbitrary one and we can think that effectively the length normalization is mapping the length of all the evaluation data to the mode of the development data length histogram.

Regarding the HT-PLDA system, first we can note that the performance gap between G-PLDA and HT-PLDA is greatly reduced (if not completely eliminated). Also, although HT-PLDA is able to successfully cope with the development and evaluation mismatch induced by the i-vector extraction procedure, a small improvement is observed after transforming the i-vectors.

Figure 5.2 shows the DET curves for the G-PLDA, LN G-PLDA and HT-PLDA systems separated by gender. It can be observed that the improvements brought by the LN transformation are present at all operating points. In terms of EER, a 58% relative improvement is obtained for the male trials and 40% for the female trials by using LN G-PLDA.



(a)

(b)

Figure 5.2: (a) Verification results for male trials of the extended-core condition 5 of NIST SRE 2010. (b) Results for female trials. The DET curves correspond to the G-PLDA, Length normalized LN G-PLDA and HT-PLDA systems.

5.5. Chapter Summary

In this chapter we have presented a method to boost the performance of probabilistic generative models that work with i-vector representations. First we reviewed the mathematical formulation of the i-vector representation. Then the Gaussian PLDA and the Heavy-Tailed PLDA models were introduced. It was noted that the better performance of HT-PLDA provided strong evidence about the non-Gaussian behavior of the i-vectors. However, the success of HT-PLDA came at the expense of larger complexity and slower computation of verification scores. The goal of this chapter was to obtain a system that matched the high performance of the more complex HT-PLDA model while maintaining the simplicity and high scalability of the G-PLDA model. That is, to keep the Gaussian assumptions in the model and perform a non-linear transformation of the i-vectors to reduce the non-Gaussian behavior (i.e., i-vector Gaussianization). Two transformations were suggested, namely: radial Gaussianization and length normalization. Moreover, length normalization was formulated as an approximation of the previously proposed radial Gaussianization. This approximation becomes more exact as the dimensionality of the space is increased. Also, unlike in radial Gaussianization, length normalization was able to avoid the use for a held-out set.

Experimental validation on the telephone portion of the NIST SRE 2010 evaluation showed that by performing a simple length normalization of the i-vectors, the performance of a G-PLDA system was able to match that of a more complicated HT-PLDA one. Also, the mismatched induced by the i-vector extraction mechanism was identified as a major source of non-Gaussian behavior. Overall, a 58% relative improvement in terms of EER was obtained for the male trials and 40% for the female trials (with respect to the baseline G-PLDA system) by using length-normalized i-vectors with the G-PLDA model.

Chapter 6

Noise-Robust Speaker Recognition

In this chapter we explore noise robustness using a model compensation approach based on direct multicondition training in i-vector space. Section 6.1 identifies two main causes of performance degradation of current speaker recognition systems in the presence of additive noise. Section 6.2 reviews the most common approaches used to ameliorate the effects of additive noise in system performance. Section 6.3 presents the system architectures proposed in this chapter. Section 6.4 provides details about the experimental setup as well as the process followed to generate multicondition training data. Section 6.5 presents an extensive set of experiments that characterize the behavior of the proposed architecture and show its robustness in terms of speaker verification performance. Finally, Section 6.6 summarizes the chapter.

6.1. Introduction

Automatic speaker recognition is concerned with designing algorithms that infer the identity of people by their voices. This is a very challenging task since the speech signals are highly variable. The sources of variability can be either intrinsic or extrinsic. When interested in making inferences about identity, intrinsic sources of variability include: the linguistic message, language, dialect, accent, vocal effort, speaking-style, emotional and health state. Extrinsic sources are the channel distortions introduced by acquisition devices (e.g., telephones), and environmental distortions like additive noise and reverberation.

In the past decade, the main focus of the speaker recognition community has been on ameliorating the effects of extrinsic variations. Recent advances in speaker recognition are not necessarily due to new or a better understanding of speaker-specific characteristics and how extrinsic sources distort or mask them; rather, they are the result of improvements in machine learning techniques that leverage large amounts of data that are representative of the application domain [56], [71], [42], [3], [68], [4],[72], [73]. In this chapter we continue along this path, and focus on speaker recognition in the presence of background noise, since it is one of the "grand challenges" for the ubiquitous use of speech technologies [74], [75].

Two main factors make speaker recognition in the presence of background noise a challenging task:

i) Loss of speaker-specific information by noise masking: This results in increased uncertainty about the speaker identity. That is, even if there is no mismatch between the model and test data, the loss of information due to noise masking makes the problem harder.

ii) Complex changes in the statistical properties of the noisy signal: This induces a mismatch between the models obtained from "clean" speech and the noisy observations [76], [77]. This mismatch is hard to characterize, since, in most real scenarios, the noise is time-varying and unpredictable. Moreover, even if the noise is well characterized, the non-linear interaction between the speech signal and noise in the feature space (i.e., cepstral domain) makes the task of noise compensation difficult [76].

In the next section we review the main approaches that have been suggested in the literature to obtain robustness to additive noise.

6.2. Related Work

The issue of noise robustness has received a lot of attention in the speech community and many methods have been proposed in the literature. They can be grouped in two classes: feature compensation and model compensation [78].

The goal of *feature compensation* is to reduce the mismatch between the test segments and the speaker models by attenuating the noise. It is computationally cheaper than model compensation and independent of the recognizer. Also, it is normally performed either in the spectral domain (e.g., spectral subtraction [79]) or directly in the feature domain (e.g., RASTA [80], and MMSE [81], [82]). In its most basic form, the main drawback of feature compensation is that it produces point estimates of the clean features without providing any information about the reliability of the estimates (i.e., ignoring uncertainty). This drawback can be eliminated by producing full posterior distributions in the compensation domain and then propagating them to the recognition domain. This framework is referred to as *uncertainty propagation* and deals with the issue of how to propagate probability distributions (or statistical moments) through nonlinearities [83].

Another alternative, and the most commonly used to account for both uncertainty and model mismatch, is to use *model compensation* techniques. These techniques deal with model mismatch and uncertainty by modifying the parameters of the modeling distributions. They come in four flavors: i) direct multicondition training; ii) general purpose data-driven adaptation; iii) mismatch-function techniques; iv) missing-feature theory.

The direct training approach simply creates a single model [84] or a collection of models [85] using a multicondition dataset. In general, single model multicondition training results in improvements for both seen and unseen training conditions [86]. However, this normally comes at the expense of a small reduction in performance for the clean condition [87]. To partially overcome this issue, multiple models can be trained based on partitions of the multi-condition dataset. However, partitioning the data too much may result in loss of generalization. Typical ways to partition the data involve noise types and SNR levels [85], [88]. At recognition time, either the best matching model is selected [85], or a combination of all their outputs [89].

In their basic form, general purpose adaptation approaches assume a prior "clean" model and adapt its parameters to different conditions by imposing some general functional form of the transformation (i.e., affine). Specifically, they do not make any explicit assumptions about the interaction between the noise and the clean signal and simply focus on adapting the parameters of the "clean" model based on adaptation data. Classic examples of these techniques are: Maximum Likelihood Linear Regression (MLLR) [90] and Maximum a Posteriori (MAP) adaptation [39]. More sophisticated training strategies, like Noise Adaptive Training (NAT) [91], can be used in situations where assuming a clean model is not realistic.

Alternatively, techniques based on *mismatch-functions* explicitly model the interaction between the noise and speech features to produce a parametric model in the feature domain. Unfortunately, the interactions described by the mismatch-functions do not have a closed-form solution in the model domain and different types of approximations have been proposed. Two good representatives of these approximations are Data-driven PMC (DPMC) [92] and Vector Taylor Series (VTS) [76], [77]. In contrast with general purpose adaptation techniques, they are more complex, but require less adaptation data. In fact, most of them only require estimates of the noise moments (i.e., mean and variance).

Finally, the family of *missing-feature* approaches [93], [94] assumes that a reliable estimation of the properties of the noise is not realistic and simply ignores the severely degraded feature components (i.e., those that are not well accounted for by the model).

The majority of the techniques mentioned so far were initially developed in the context of speech recognition. However, they are directly applicable to the area of speaker recognition. In particular, to handle extrinsic variability, the mainstream of speaker recognition techniques has mostly followed either the route of general purpose data-driven adaptation or direct training. Even though most of the techniques have been presented in the context of channel variability, most of them are general purpose techniques[8]. Within the context of generative probabilistic formulations, the classic technique of relevance MAP adaptation of Gaussian Mixture Models (GMMs) [37] was the precursor of a series of developments that resulted in the current state-of-the-art based on i-vector representations [3]. An extension to the basic GMM-UBM framework, denoted as feature mapping [95] was followed by the use of eigenchannels [96], [97]. This framework was extended into the Joint Factor Analysis (JFA) paradigm [56], that finally evolved into the current state-of-the-art based on i-vector representations[9] [3].

The i-vector formulation provides an elegant way to obtain a low dimensional fixed-length representation of an entire speech utterance. The low-dimensional nature of the i-vector space facilitates the use of large amounts of data to remove/attenuate the effects of adverse conditions. It also has opened the door for new ways to decompose a speech signal into a speaker-specific component and an undesired variability component.

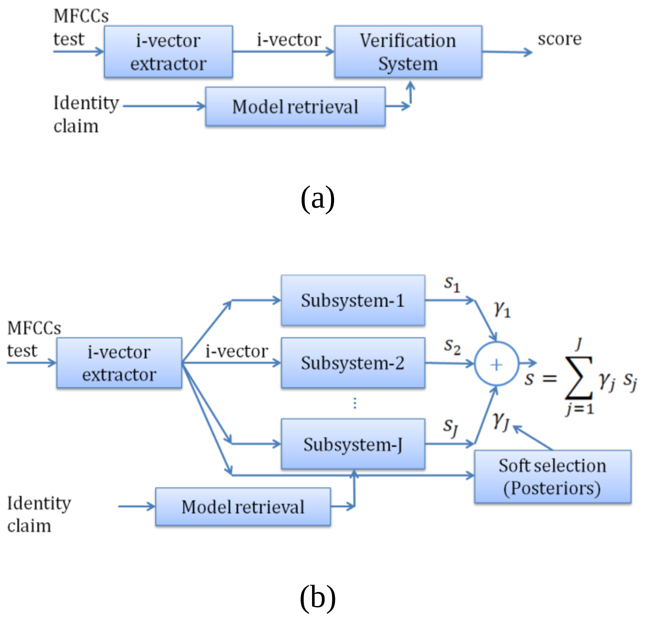
6.3. Speaker Recognition System

All the recognition systems studied in this chapter are based on an i-vector front-end followed by a back-end comprising one or more Gaussian probabilistic generative models. Moreover, we assume access to a multicondition *development set* comprising S speakers with multiple utterances per speaker observed under K conditions (e.g., different noise types and SNRs). The development set was used to

[8] In fact, that is the reason why it is quite common to see the use of the word “channel” variability to refer to anything that it is not related to speaker-specific information.

[9] Note that the list of key contributions from GMM-UBM to i-vectors is not an exhaustive one and many other ideas were involved in the process.

train the i-vector front-end as well as the back-ends. Also, in order to assess the system performance, an *evaluation set* with data from speakers not present in the development set was used. The evaluation set comprises model segments and test segments. A speaker model will be constructed by using a collection of model segments from a single speaker that have been observed under the K conditions represented in the development set. The test segments will be used to produce verification trials. Some of the test segments will belong to one of the K anticipated conditions in the development set while others will not. This will allow us to assess the recognition performance in anticipated as well as in unseen conditions.



(a)

(b)

Figure 6.1: (a) Single-classifier architecture. (b) Multi-classifier architecture with J subsystems. The final score is a linear combination of the individual scores.

In the following we present an overview of the four speaker recognition architectures studied in this chapter. This is followed by a detailed description of a modified version of the PLDA model introduced in Section 5.2.2 denoted as TIED-PLDA and a score combination block.

6.3.1. System Architectures

Figure 6.1 depicts two speaker recognition systems: one based on a single classifier (panel a), and another one based on multiple classifiers (panel b). Combining the single- and multiple-classifier setups with the option to train them in a single-condition or multicondition fashion results in the four architectures studied in this work.

For the single-classifier architectures, the verification score is the log-likelihood ratio outputted by the classifier. Since we are only considering probabilistic generative models, if the classifier parameters are trained using multicondition data, the optimal set of parameters will achieve a compromise that results in a good average representation for all conditions. While this might help to improve the generalization capability of the system, it might also be an unrealistic model when the conditions under which the system will operate vary widely (i.e., SNRs from 40 to 0 dB). For this reason we also consider architectures with multiple classifiers where each subsystem is only trained on a subset of conditions. This approach increases the number of parameters of the system and therefore reduces the need to compromise between conditions. In particular, for a multi-classifier setup with J classifiers, the final verification score is a convex mixture of the individual scores of each subsystem (details about how to obtain the mixture coefficients are given in Section 6.3.3). Note that the total number of classifiers J and the number of observation conditions K need not be the same. In fact, the mapping between conditions and classifiers is a key aspect of the system design and it is analyzed in Section 6.5.6.

For all the architectures, the front-end i-vector extractor (see Section 5.2.1) is trained only with clean data (i.e., the original development dataset before it is augmented with synthesized corrupted versions). The back-ends make use of the entire multicondition data set, except for the baseline system that is trained on the original data only.

The four system architectures analyzed in this work are:

- *Single classifier and Single condition*: This setup represents the baseline system and is used to characterize the behavior of a state-of-the-art system when no explicit compensation for noise is included. All performance improvements due to multicondition training are compared to this baseline system.

- *Single classifier and Multicondition*: In this setup, all the available multicondition development data is pooled together and a unique set of classifier parameters is learned. This configuration provides the largest ratio of data points to model parameters.

- *Multiple classifiers and Single condition*: In this setup, each subsystem is trained on data from only one of the K available conditions (i.e., the number of subsystems J is equal to the number of conditions K). The final score is a convex mixture of the subsystem scores.

- *Multiple classifiers and Multicondition*: This is the most general architecture studied in this work. There are two key aspects of this setup that control the number of free parameters of the architecture. The first one is the mapping between conditions and subsystems. Note that we are not assuming a partition of a set of K conditions into J disjoint subsets. Hence, data from the same condition can be reused to train multiple subsystems. The second one is the selection of the multicondition training scheme (i.e., Pooled- or Tied-PLDA approaches described in Sections 5.2.2 and 6.3.2 respectively). When using this architecture, special care is needed to prevent overfitting.

In the following we provide a detailed presentation of the two building blocks that are specific of the abovementioned architectures.

6.3.2. Tied-Probabilistic Linear Discriminant Analysis

When i-vectors are observed under a wide range of conditions (e.g., high and low SNRs), the PLDA model introduced in Section 5.2.2 might be too restrictive. Mostly because it assumes that given a latent identity variable all observations are generated using the same set of parameters $\{m, \Phi, \Sigma\}$. A generalization of the PLDA model, denoted as Tied-PLDA [67], is obtained by clustering the observed i-vectors into K conditions and allowing the parameters to be different depending on the observation condition. In the following we describe how to model i-vectors and how to compute scores using Tied-PLDA.

i) Modeling: Consider a total of K observation conditions and one utterance per condition from speaker i. A Tied-PLDA model for the observed i-vectors follows:

$$\begin{bmatrix} \eta_{i,1} \\ \eta_{i,2} \\ \vdots \\ \eta_{i,K} \end{bmatrix} = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_K \end{bmatrix} + \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_K \end{bmatrix} \beta_i + \begin{bmatrix} \epsilon_{i,1} \\ \epsilon_{i,2} \\ \vdots \\ \epsilon_{i,K} \end{bmatrix}. \tag{6.1}$$

Note that the latent identity variable β_i is the same across all conditions (i.e., it ties the hyperparameters). Also, the number of parameters of a Tied-PLDA system with K

conditions $\{m_k, \Phi_k, \Sigma_k\}_{k=1}^{K}$ is K times larger than the number of parameters in the basic PLDA system. Hence, the modeling power of Tied-PLDA is much larger and caution is needed not to over fit the model. As in the PLDA case, ML point estimates of the model parameters are obtained from the development set using the EM algorithm [67]. A number of hybrid models can be created by only allowing a subset of the parameters to be condition-dependent. For example, one might consider a system in which only the noise covariance matrices depend on the observation conditions $\{m, \Phi, \Sigma_k\}_{k=1}^{K}$. These hybrid models provide a mechanism to control the number of parameters.

ii) Verification score: The procedure to compute verification scores under the Tied-PLDA formulation is essentially the same as in the PLDA case with two minor distinctions. First, equations (5.5), (5.6) and (5.7) need to be modified to account for the condition-dependent nature of the parameters. Second, the test segment i-vector needs to be assigned to one of the K conditions. Since we only use Tied-PLDA in the multi-classifier setup, we perform this assignment by defining one subsystem per condition (i.e., $J = K$) and assuming that the test segment was generated by the condition represented by the classifier. For example, if we consider $K = 3$ conditions, and subsystem $j \in \{1,2,3\}$ is trained on data from conditions 1 and 2, with the goal of being an "expert" for condition 2, then, under the same-speaker hypothesis $\mathcal{H}_s$ the observed i-vectors would follow:

$$\begin{bmatrix}\eta_{i,1}\\ \eta_{i,2}\\ \eta_T\end{bmatrix} = \begin{bmatrix}m_1^j\\ m_2^j\\ m_2^j\end{bmatrix} + \begin{bmatrix}\Phi_1^j\\ \Phi_2^j\\ \Phi_2^j\end{bmatrix}\beta_i + \begin{bmatrix}\epsilon_{i,1}^j\\ \epsilon_{i,2}^j\\ \epsilon_{T,2}^j\end{bmatrix}. \qquad (6.2)$$

The model for the different-speaker hypothesis $\mathcal{H}_d$ follows immediately by adding a second latent identity variable in the same way as it was done in (5.6). Hence the subsystem j that represents (is an expert in) condition 2, and was trained with data from conditions 1 and 2, will be characterized by the Tied-PLDA parameters: $(\{m_1^j, \Phi_1^j, \Sigma_1^j\}, \{m_2^j, \Phi_2^j, \Sigma_2^j\})$.

6.3.3. Multi-classifier Combination

In the multi-classifier architectures, the final verification score is obtained as a convex mixture of a collection of J scores $\{s_j\}$ according to the weights $\{\gamma_j\}$ (see Figure 6.1 (b)). Specifically, weight γ_j corresponds to the posterior probability of system j having generated the test i-vector η_T, regardless of the claimed identity. Denoting α_j as the prior probability of observing data from subsystem j, and letting the likelihood of the test i-vector for subsystem j be $p(\eta_T|j)$, a direct application of Bayes' theorem results in:

$$\gamma_j \;=\; \alpha_j\, p(\eta_T|j) \;/\; \sum_{j'=1}^{J} \alpha_l\, p(\eta_T|j')\,. \tag{6.3}$$

Ideally we would like the vector of posterior probabilities $\gamma = [\gamma_1, \dots, \gamma_J]^T$ to be sparse and have most of the probability mass concentrated around the systems trained on data similar to the conditions of the test segment at hand. In this way, the systems that better match the test condition will have the largest contribution to the final score. Therefore, we can consider each subsystem as an "expert" on a subset of conditions and the final score as a weighted combination of their opinions.

6.4. Experimental Setup

6.4.1. Original Dataset and Configuration Setup

All the experiments were conducted on the male part of condition 2 of the extended NIST SRE 2010 evaluation [1] (i.e., interview data). Throughout the experiments we refer to this set as *evaluation data*. We selected condition 2 because it provides the largest amount of trials among the 9 available conditions. However, since there is no reason to suspect that the effects of multicondition training will be dependent on the gender of the speaker, we focused on the male trials to keep the amount of data more manageable. This subset comprises 1,108 models and 3,328 test segments from which 6,932 target trials and 1,215,586 non-target trials were obtained. Verification performance is reported in terms of Equal Error Rate (EER).

We used 400 dimensional i-vectors in all experiments. They were computed using a gender-dependent i-vector extractor trained on data from NIST SRE 04, 05, 06, 08, Switchboard and Fisher. The necessary Baum-Welch sufficient statistics are collected using a diagonal-covariance gender-dependent UBM with 2048 mixtures trained on the same data. Note that neither the UBM nor the i-vector extractor were exposed to the noisy conditions, since they were trained only on this “original” data.

We constructed a *development set* by selecting a subset of the data used to train the i-vector extractor. In particular, we only kept the data from speakers with more than 3 speech utterances. This resulted in 907 male speakers with a total of 10,695 files. The PLDA model parameters of all the aforementioned architectures were trained using this dataset (or the augmented multicondition version described in the next section).

All speech files were parameterized using 38 LFCCs [98] (i.e., 19 base coefficients without c0 plus deltas) obtained every 10 ms from a 20 ms Hamming window. Only the information in the frequency band of 300-3400 Hz was used. Global mean and variance normalization was applied to the entire utterance.

For each file in the original development and evaluation datasets, voice activity detection (VAD) was performed using a combination of the ASR transcripts provided by NIST [1] and an energy-based VAD system. Both channels (i.e., interviewer and interviewee) were used to remove the interviewer's speech from the interviewee channel.

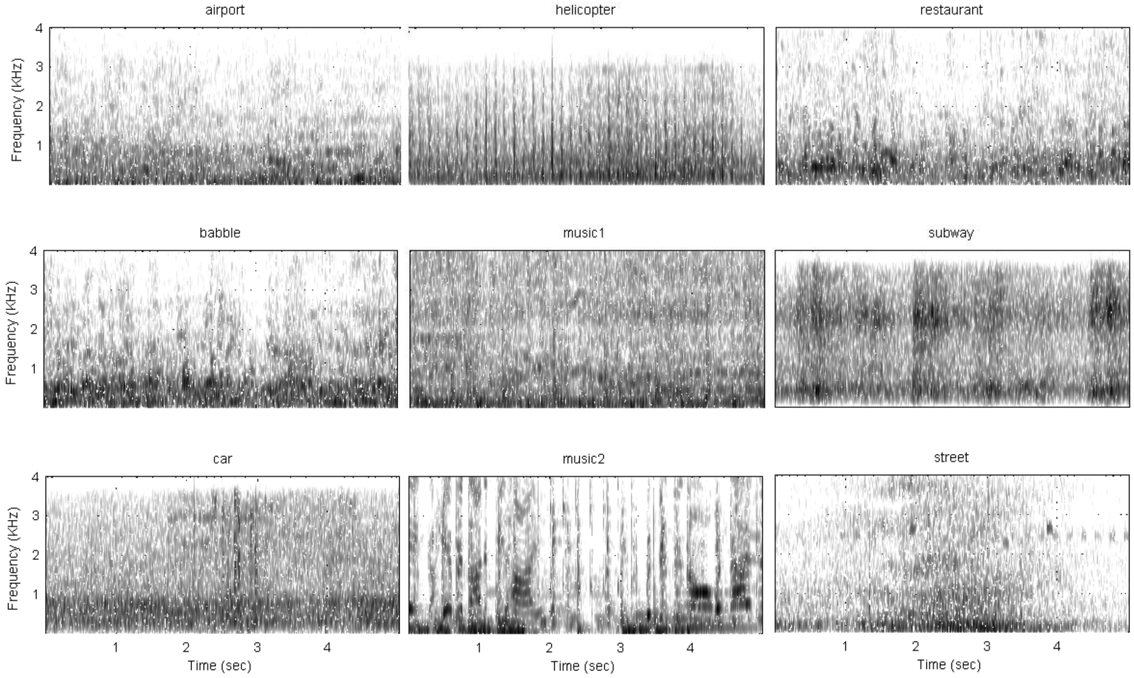


Figure 6.2: Spectrograms of 5 second portions of the noises (except white noise) used to generate the multicondition development and evaluation dataset.

6.4.2. Multicondition Data Generation

In order to create a set for *multicondition* training and testing, 70 noisy copies of each file from the evaluation and development sets were created by electronically adding 10 different types of noises: white, babble, car, helicopter, airport, subway, street, restaurant, music (music-A), and music (music-B) at 7 SNR levels: 30, 20, 15, 10, 6, 3, and 0 dB. Figure 6.2 shows spectrograms of a 5 second portion of each of the noises (except for the white noise). I-vectors were computed for each file in the multicondition development and evaluation sets (i.e., original plus 70 corrupted versions). Since there are 4,436 files in the original evaluation set and 10,695 in the development set, the multicondition dataset comprises 314,956 (i.e., 71 x 4,436) evaluation and 759,345 development i-vectors.

The resulting SNR of the noisy files was defined by the energy levels of speech and the added noise in the speech regions (determined by VAD) and the silence regions were excluded for the SNR calculation. Also, the same VAD computed from the original files was used for the noisy versions. This VAD strategy is suboptimal since it lets too much noise into the system for low SNRs. However, the focus of this work is on the relative trends among the explored architectures, therefore, this is not a big concern. In future work we will explore the effects of better VAD strategies in the absolute values of the speaker verification performance.

During the experiments, we defined two groups based on the noise types: P1 = {original, babble, car, helicopter, white}, and P2 = {original, airport, subway, street, restaurant, music-A, music-B}. Note that except for the original data, the two subsets are disjoint. We also defined Pall as the union of P1 and P2. The purpose of these partitions is to evaluate the multicondition training strategies under both anticipated and unanticipated conditions. When the systems are trained on data from P1 and tested on data from P2 (or vice versa) we refer to it as the unseen condition. On the contrary, when the train and test data belongs to the same partition, we refer to it as then anticipated condition.

6.5. Experiments

In the following we present all the experiments conducted in this chapter. We start with an analysis of the effects of i-vector length normalization. This is followed by a characterization of the baseline system when no explicit noise compensation is applied. After this, we start exploring various aspects of multicondition training. In particular, we first analyze the effects of SNR granularity, and then address the issue of how to achieve good performance in a broad range of conditions. We continue with a detailed comparison of the different multi-classifier training strategies and an assessment of the optimal number of classifiers. After that, we examine the behavior of the score combination module as well as the effects of multicondition training on the parameters of the PLDA model. Finally, we conclude with an evaluation of the score calibration of the systems.

6.5.1. Length Normalization

In [68] it was shown that the current strategy (e.g., [3]) used to extract i-vectors induces a severe mismatch between the length of the development and evaluation i-vectors. This was identified as a major source of non-Gaussian behavior. A nonlinear transformation of the i-vectors denoted as *length normalization* was proposed to reduce this mismatch and allow for effective Gaussian modeling. Here we further extend those observations by looking at the distribution of i-vector lengths as a function of the SNR.

Figure 6.3 shows the results of fitting Gaussians (i.e., one Gaussian per SNR) to the length distributions of i-vectors from utterances corrupted by babble noise at different SNRs. As a general trend we can observe that lower SNRs result in larger shrinkage. This is also true for other noises as well as for the i-vectors in the development set. Since both the UBM and i-vector extractor are trained on clean data only, the noisy conditions are not represented as well in the i-vector subspace and produce lower energy (shorter) i-vectors. Thus, when considering a collection of data with a wide range of SNRs, severe length mismatch is not only observed between development and evaluation datasets but also within them.

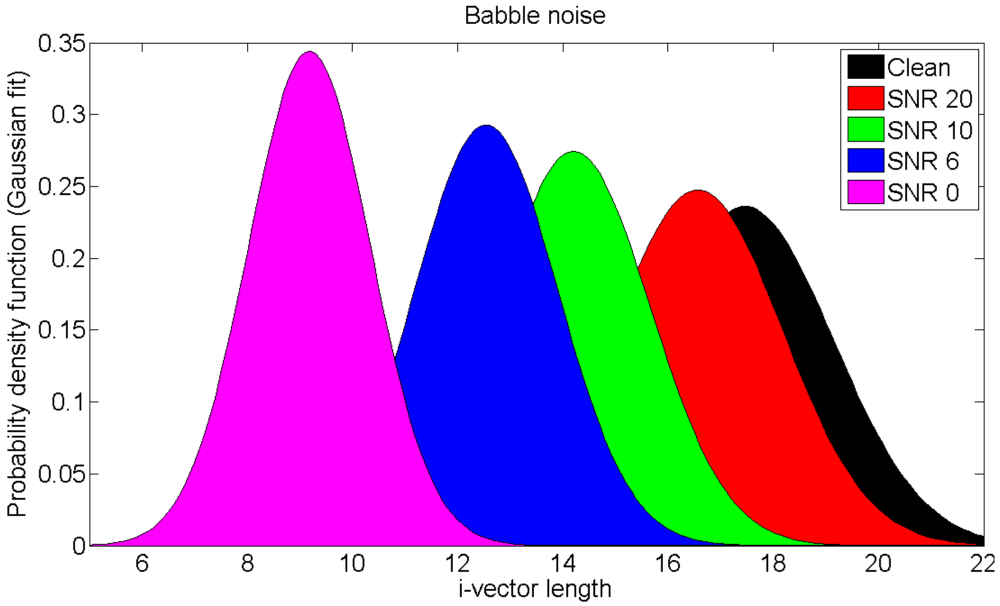


Figure 6.3: Gaussian fits to the distributions of i-vector lengths of the evaluation data for different SNRs in babble noise.

Noise type	SNR (dB)	Single classifier architecture			
		Baseline PLDA		Multicondition trained Pooled-PLDA	
		Length normalization		Length normalization	
		NO	YES	NO	YES
Original	-	**1.35**	**0.97 (28%)**	**1.88**	**1.22 (35%)**
White	20	4.25	2.86 (33%)	2.69	1.95 (28%)
	10	19.02	14.31 (25%)	10.49	8.3 (21%)
	6	27.77	22.66 (18%)	18.42	15.23 (17%)
	0	39.94	36.22 (9%)	32.9	28.46 (13%)
White Average		**22.75**	**19.01 (16%)**	**16.13**	**13.49 (16%)**

Table 6.1: Verification performance in EER along with relative improvement (in parenthesis) due to length normalization. The baseline system was trained only on the original data. The multicondition pooled-PLDA system was trained on: original, babble, car, and helicopter data. Results are presented for test segments in white noise, hence representing an unseen condition.

In order to assess the effects of this behavior in recognition performance, Table 6.1 shows the verification results of two systems: one with length normalization and another without it. Both systems are based on single-classifier architectures. The baseline system is trained on the original development data, whereas the Pooled-PLDA system is trained on: original, babble, car, and helicopter data. Results are presented for test segments corrupted by white noise; hence, representing an unseen condition for both systems. Based on the relative improvements shown in parenthesis, we can observe that both systems greatly benefit from the use of length-normalized i-vectors. Also, on average, the same relative improvement is observed regardless of multicondition or single-condition training. Moreover, the effectiveness of length normalization decreases as the SNR increases. This suggests that, at lower SNRs, the degradation in recognition performance is mostly dominated by the loss of speaker information and the effects of using the wrong modeling assumptions (i.e., assuming that the data can be well represented with a Gaussian model) are less prominent. Hence, even though the within development/evaluation i-vector length mismatch may seem as an opportunity for length normalization to offer even larger relative improvements, the dominating artifact at low SNRs is the loss of information and length normalization cannot compensate for that. However, the overall performance improvements of length normalization are quite impressive, and therefore, all the experiments in this work are conducted using length-normalized i-vectors.

6.5.2. Baseline System

The baseline system is only trained on the original data. It is used to characterize the behavior of a state-of-the-art system when no explicit compensation for noise is included. Figure 6.4 and Table 6.2 show the performance of the baseline system across all noises from both partitions: P1 and P2. We can observe a fast degradation in performance for all noises. For example, on average, the EER increases approximately by 3, 6 and 23 times when the SNR falls to 15dB, 10dB and 0dB respectively. Interestingly, the degradation rate for white and subway noises is much faster. This can be attributed to two factors.

First, the SNR is an average statistic over the entire file and only provides a partial description of the noise characteristics. Hence, care must be taken when making comparisons across noise types that have the same average SNR. In particular, the variance of the noise energy across time plays an important role in i-vector representations and it is not reflected in the average SNR. Since i-vectors are based on averages of clusters of speech frames over time, if the variance of the noise energy is very high, it means that a small portion of the frames might be severely corrupted while the rest remain less affected. This uneven corruption of the information may facilitate the extraction of speaker-specific information from the less corrupted regions. On the contrary, a noise with very low energy variance results in almost all frames being corrupted similarly, and therefore, makes the extraction of the speaker-specific information harder; hence resulting in reduced performance. These claims are consistent with the behavior observed in the spectrograms of the different noises (5 second portions shown in Figure 6.2).

The second factor is related to the spectral distribution of energy. Since we are using a linearly-spaced filterbank to obtain the LFCCs, the higher resolution in the high-frequency range makes the system more sensitive to noise corruptions with high energy in the high frequencies (i.e., white noise and subway noise have a much flatter spectrum than the other noises which results in higher energy at high frequencies). However, when considering the overall picture, the higher sensitivity to these two types of noises might be a price worth paying, especially if considering utterances from females speakers, due to the consistently better performance of LFCCs over MFCCs in the original utterances of condition 2 of SRE10 [98].

Test noise	SNR (dB)	Single-classifier architecture							Multi-classifier architecture		
		Base	Pooled-PLDA with all SNRs			Pooled-PLDA with subset SNRs			2 class Pooled-PLDA		
		Train-orig.	Train-Pall	Train-P1	Train-P2	Train-Pall	Train-P1	Train-P2	Train-Pall	Train-P1	Train-P2
Orig.	-	0.99	1.29	1.27	1.34	1.21	1.19	1.17	0.88	0.90	0.92
Avg. across P1 noises	30	1.04	1.33	1.34	1.36	1.24	1.24	1.18	0.91	0.91	0.91
	20	1.77	1.52	1.53	1.52	1.43	1.41	1.38	1.20	1.18	1.27
	15	3.59	1.85	1.85	2.20	1.75	1.73	2.01	1.62	1.63	2.00
	10	7.53	2.85	2.73	4.09	2.79	2.57	3.85	2.75	2.63	3.93
	6	13.22	4.87	4.61	7.66	4.84	4.39	7.29	4.79	4.53	7.53
	3	19.39	7.80	7.42	12.10	7.78	7.15	11.66	7.78	7.34	11.94
	0	26.59	12.37	11.64	18.30	12.33	11.32	17.86	12.30	11.53	18.15
	Avg. 30-15	2.13 (0%)	1.57 (27%)	1.57 (27%)	1.69 (21%)	1.47 (31%)	1.46 (31%)	1.52 (29%)	1.24 (42%)	1.24 (42%)	1.39 (35%)
	Avg. 10-0	16.68 (0%)	6.98 (58%)	6.60 (60%)	10.54 (37%)	6.94 (58%)	6.36 (62%)	10.17 (39%)	6.91 (59%)	6.51 (61%)	10.39 (38%)
Avg. Across P2 noises	30	1.05	1.33	1.34	1.37	1.24	1.23	1.18	0.91	0.92	0.94
	20	1.63	1.47	1.56	1.50	1.38	1.44	1.32	1.11	1.18	1.15
	15	2.98	1.71	2.06	1.73	1.64	1.92	1.58	1.50	1.75	1.55
	10	6.11	2.36	3.58	2.37	2.30	3.40	2.21	2.27	3.34	2.27
	6	10.70	3.70	6.53	3.60	3.66	6.26	3.46	3.64	6.28	3.56
	3	15.60	5.73	10.30	5.49	5.70	9.96	5.37	5.68	10.08	5.47
	0	21.74	9.29	15.69	8.85	9.21	15.21	8.74	9.20	15.42	8.80
	Avg. 30-15	1.89 (0%)	1.50 (20%)	1.65 (12%)	1.53 (19%)	1.42 (25%)	1.53 (19%)	1.36 (28%)	1.18 (38%)	1.28 (32%)	1.21 (36%)
	Avg. 10-0	13.53 (0%)	5.27 (61%)	9.02 (33%)	5.08 (62%)	5.22 (61%)	8.71 (36%)	4.94 (63%)	5.20 (62%)	8.78 (35%)	5.02 (63%)

Table 6.2: Verification performance in EER along with relative improvement over the Baseline system (in parenthesis). The systems are either based on a Single- or Multi-classifier architecture. All systems (except baseline) are trained using multicondition data from the three partitions (Pall, P1, P2). The verification results are also separated in partitions to facilitate evaluation of anticipated and unseen conditions. There are two variants of the Single-classifier architecture that use different subsets of SNRs. The Multi-classifier system uses 2 classifiers trained in Pooled mode (see Section 6.5.6 for details about the 2 classifiers).

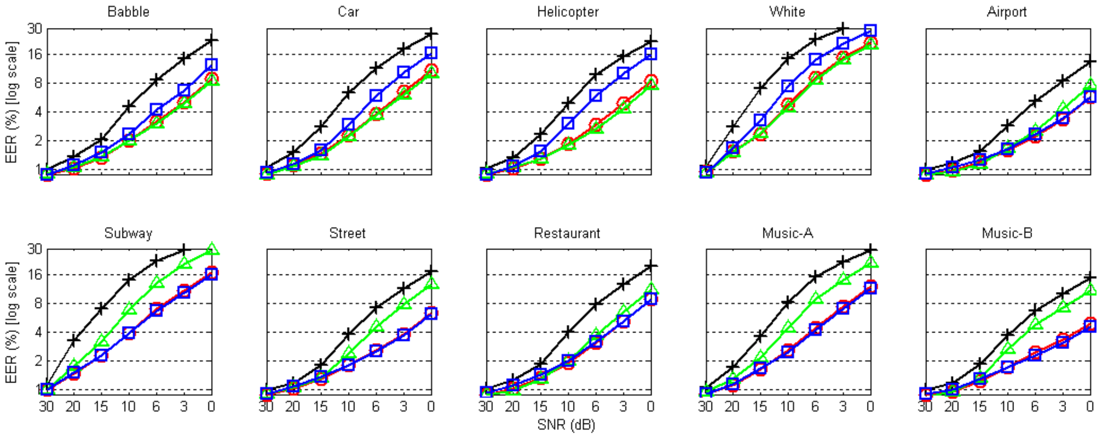


Figure 6.4: Comparison of the baseline system with a 2-classifier Pooled-PLDA system trained on different partitions (Pall, P1, and P2) of the development data. The performance is reported in EER for all noises and SNRs in the evaluation set.

6.5.3. SNR Granularity

In this section we explore the effects of the SNR granularity on the verification performance. That is, we are interested in knowing what SNR increments should be considered when generating a multicondition dataset. We focus our experiments on the single-classifier Pooled-PLDA architecture, since there is no reason to believe that the choice of architecture would have a significant effect on the question at hand.

During the experiments we trained two Pooled-PLDA systems using different combinations of SNRs. In one system we used all the SNRs available in our development dataset and in the other we removed the samples corresponding to SNRs of 15 dB and 3 dB. Also, the experiments were done taking into account the noise partitions where we computed results for three training variants. The first variant used noises from both partitions (Pall), and therefore, all test conditions were seen during training. The second one only used the noise data from partition P1, and the third one from P2. In this way we can observe the effects of SNR granularity in both anticipated and unseen conditions.

Columns 4 thru 9 of Table 6.2 show the results for the six configurations (2 systems with 3 training variants). Before analyzing the influence of the SNR granularity, it is important to point out that both systems significantly outperform the baseline system. Also, as expected, the improvement in anticipated conditions is better than in unseen conditions. However, the performance in unseen conditions is still quite impressive. For example, if we consider the performance of the system trained on both partitions (Pall) as the "oracle" performance, then, the results for unseen conditions (i.e., trained on P1 and tested on P2 and vice versa) get very close to the oracle for the range of SNRs between 30 to 15 dB. For lower SNRs, the gap between the actual performance and the oracle is larger. However, the achieved performance always exceeds 50% of its full potential when considering the oracle as an upper bound.

Going back to the SNR granularity, if we compare the results in columns 4 and 7, we can see that the trend of performance with SNR is not affected by the fact that the second system was trained only on a subset of all the available SNRs. That is, the results for SNRs of 15 dB and 3 dB follow the same trend in both systems. Note that the same is true when the systems are trained only on noises from P1 or P2 and tested on both anticipated and unseen conditions. This suggests that sampling SNRs in increments smaller than 6 dB is unlikely to improve performance. Interestingly, this low sensitivity to SNR granularity was also pointed out for speech recognition in

[85]. Moreover, the verification performance obtained with the sub-sampled system for higher SNRs (i.e., average between 30 and 15 dBs) is slightly better than that of the system using all available SNRs. This can be explained by the fact that the average SNR across all the training data for the sub-sampled system is 13.2 dB as opposed to 12 dB for the full set. Hence, this bias towards higher SNRs produces a small improvement on the test conditions with higher SNR. This observation is further explored in the next section.

6.5.4. Balancing Development Conditions

An important question in the multicondition training setup is: What is the appropriate way to balance the training data to achieve good performance across a broad range of testing conditions? The success of multicondition training lies in its ability to expose the unreliable components of a representation. This allows the model to focus its representational power on the robust (invariant) components. However, for a fixed model complexity (i.e., number of system parameters), the larger the range of conditions in which we expect the system to be competent, the less outstanding it will be for any particular one. This concept is succinctly captured in the well known figure of speech: “Jack of all trades, master on none”. To explore how this idea is manifested in our particular setup, we trained three systems using different subsets of SNRs. The three SNR ranges were denoted as: ALL = (original, 30, 20, 10, 6, 0); HI-SNR = (original, 30, 20, 10); and LOW-SNR = (original, 10, 6, 0). The systems were based on a single-classifier Pooled-PLDA architecture and used all the available noise types. Also, we omitted the 15 dB and 3dB sets based on the results reported in the previous section.

Figure 6.5 shows the verification performance averaged over all noise types at different SNR levels for the three systems. The system trained on all the SNRs produces good results for the entire range, but it is outperformed in the higher SNR range (original – 15 dB) by the system trained on the HI-SNR subset, and in the lower SNR range (3 – 0 dB) by the systems trained on the LOW-SNR subset. However, the performance of the biased systems (either towards HI-SNR or LOW-SNR) is much worse than that of the system trained on all the data for the SNR ranges not included in their training sets. Hence, for single-classifier architectures, a good performance in a wide range of operating conditions comes at the expense of performance degradation in any particular one. This suggests that a multi-classifier setup might be a good way around this tradeoff. For example, based on the results in Figure 6.5, if we were to use the system biased towards high SNRs to produce scores for verification trials comprising high SNR test segments, and the system biased towards low SNRs for test segments with low SNRs, we would obtain a system that covers the same range as the ALL data single-classifier system but with improved performance. However, this strategy assumes the ability to automatically select the system that should produce the verification score. Moreover, it is not immediately apparent how many classifiers to use as well as which training methodology to select. These questions are explored in the following sections starting with the analysis of the training strategy.

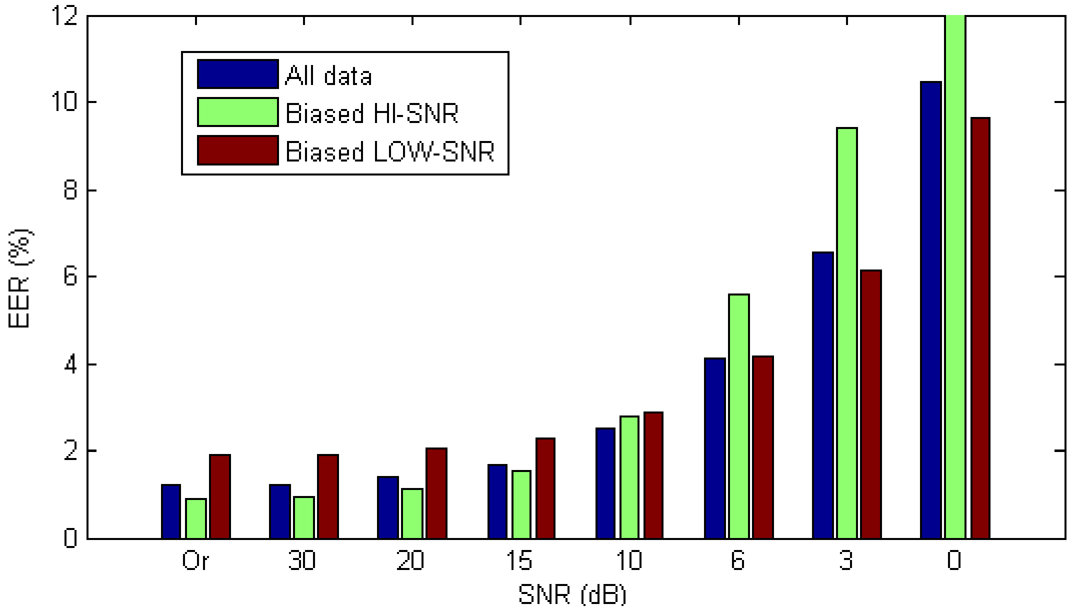


Figure 6.5: Verification performance averaged over all noise types at different SNR levels for three systems trained on different subsets of SNRs: all (or,30,20,10,6,0), biased-hi (or,30,20,10), biased-low (or,10,6,0).

6.5.5. Multi-classifier Training Strategies

The use of multi-classifier techniques has the potential to produce verification systems that cover a broad range of conditions without sacrificing performance in each one of them. In this section we compare the verification performance of three multi-classifier architectures. Two of them are based on multicondition training (i.e., Pooled- or Tied-PLDA) and the other one is based on single condition training and it is denoted as Individual-PLDA training.

The Individual-PLDA training architecture assumes that the number of classifiers is equal to the number of development conditions (i.e., $J = K$). Hence, for the experiments in this section we use the same configuration for the two other architectures to facilitate comparison. Note that under this configuration the ratio of data points to model parameters is the smallest and care must be taken not to over fit the data. For this reason, we performed the experiments taking into account the two partitions: P1 and P2. In this way, the generalization capabilities of the three training approaches can be assessed for unseen conditions. In particular, the number of classifiers for the systems trained on P1 and P2 are 21 and 31 respectively (since we are not including the subsets of 15 dB and 3 dB in the training sets).

Additionally, the multicondition training of the Tied- and Pooled-PLDA systems was done by using the original data along with the data of the corresponding condition of the classifier. For example, the classifier trained for babble noise at 20 dB used the development data corresponding to that condition plus the original data set. This strategy was found to be successful in the preliminary work reported in [73]. However, the lack of an extensive set of unseen conditions in [73] did not allow for a very conclusive assessment of the generalization ability of each training scheme. Here we have expanded those experiments using a larger set of unseen conditions.

Relative improvement over baseline (%)		Test			
		Average 30-15 dB		Average 10-0 dB	
Train	System	P1	P2	P1	P2
P1	Pooled	**40**	25	**66**	27
	Tied	**42**	21	**68**	17
	Individual	**17**	2	**49**	0
P2	Pooled	26	**28**	33	**67**
	Tied	26	**33**	27	**69**
	Individual	15	**14**	16	**50**

Table 6.3: Relative improvement of EER over the baseline system for three multi-classifier architectures (Pooled, Tied, Individual). Results are averaged over two ranges of SNRs (30-15 and 10-0) for the four combinations of training/testing of the two partitions. See text for details about the multi-classifier setup.

Table 6.3 shows the performance for the three training strategies in terms of relative EER improvement over the baseline system. The results are averaged for all the noises in each partition over two ranges of SNRs: 30 to 15 dB and 10 to 0 dB. The numbers in bold correspond to the anticipated conditions and the rest to unseen conditions. It can be observed that the three approaches outperform the baseline system in both conditions. However, the Pooled- and Tied-PLDA systems significantly outperform the Individual-PLDA approach. Specifically, for the anticipated conditions and SNR range of 30-15 dB, the performance of the Individual-PLDA is less than half of the performance of the other two systems. For lower SNRs the gap is smaller but still significant. Also, in the unseen conditions, the improvement of the Individual-PLDA model trained on P1 is negligible. These observations indicate that, under the multi-classifier PLDA framework, the key to successfully leverage the multicondition data is to train the parameters of each classifier using data from multiple conditions. This can be explained by the fact that when the training set comprises speech utterances corrupted by different distortions, the reliability/unreliability of the components of the representation becomes more apparent. This in turn facilitates the identification of the speaker-specific component and the intersession variability.

Comparing the Tied- and Pooled-PLDA systems we can observe that Tied-PLDA slightly outperforms Pooled-PLDA for the anticipated conditions. However, the results for unseen conditions follow the opposite trend. This indicates that Tied-PLDA is overfitting the data since it has twice as many parameters as the Pooled-PLDA system. The fact that doubling the number of parameters of the model only leads to slight improvements in anticipated conditions—at the expense of a noticeable loss in generalization capability—leads us to select Pooled-PLDA as a more reliable alternative. The rest of the experiments will therefore be focused on this alternative. In the next subsection we explore different strategies to map conditions to classifiers.

6.5.6. How Many Classifiers?

Up to this point we have observed two behaviors that provide guidelines regarding the question: How to map development conditions to classifiers? On the one hand, based on the results in Section 6.5.4, we observed that multi-classifier techniques have the potential to produce verification systems that cover a broad range of conditions without sacrificing performance. On the other hand, the results in Section 6.5.5 showed that care must be taken to avoid overfitting the data, and therefore, controlling the ratio of data points to number of model parameters is a key design variable. These two opposing principles suggest that a successful strategy can be obtained by striking a balance between: spreading the data too thinly, and averaging too much.

To explore this trade-off, we trained four Pooled-PLDA systems from the development data in P1 (omitting the 15 dB and 3 dB subsets). The four different mappings between conditions and classifiers were: i) pooling all data together into a single classifier; ii) two classifiers, one based on (original, 30 dB, 20 dB, 10 dB) and the other (original, 10 dB, 6 dB, 0 dB); iii) six classifiers, one for each SNR, where we pool all the data from the same SNR along with the original set; and iv) twenty one classifiers with data from one noise type and SNR per classifier, along with the original set. Figure 6.6 shows the verification performance of each architecture averaged across noise types. The top panel corresponds to anticipated conditions and the bottom to unseen ones. Even though we are only displaying results for the system trained on P1, the same behavior is true for the system trained on P2. We can notice that the system with 21 classifiers outperforms the rest in anticipated conditions (especially in the low-SNR region); but performs the worst in unseen conditions at the lower SNRs. This is a clear indication of the overfitting risk that we have previously discussed. On the contrary, the systems based on one and two classifiers have the best generalization capability, and provide a competitive performance for the whole range of SNRs in the anticipated conditions. Also, the behavior of the 6-classifier system indicates that the corresponding mapping between conditions and classifiers is not as good as the other alternatives. The observed results suggest that, unless the system is deployed in a very controlled scenario, where the operating conditions are always very close to those included in the development set, the single-classifier or the two-classifier systems are the best alternatives. A more detailed comparison between these two can be established by looking at the last six columns of Table 6.2. We can notice that for the low-SNR range the two systems provide nearly identical results. However, the two-classifier system outperforms the single-classifier architecture in the original dataset as well as the high-SNR range (30 dB–15 dB). This is true for both anticipated and unseen conditions. Noting that the single-classifier architecture works worse than the baseline system in the original and 30 dB datasets helps us to appreciate the benefits of the two-classifier architecture. In particular, the use of two classifiers, one targeted towards high-SNR and another towards low-SNR results in a system that does not compromise performance in high-SNR in order to produce good results in low-SNR levels. In the next section we analyze the behavior of the score combination mechanism (i.e., Figure 6.1.b) for this architecture.

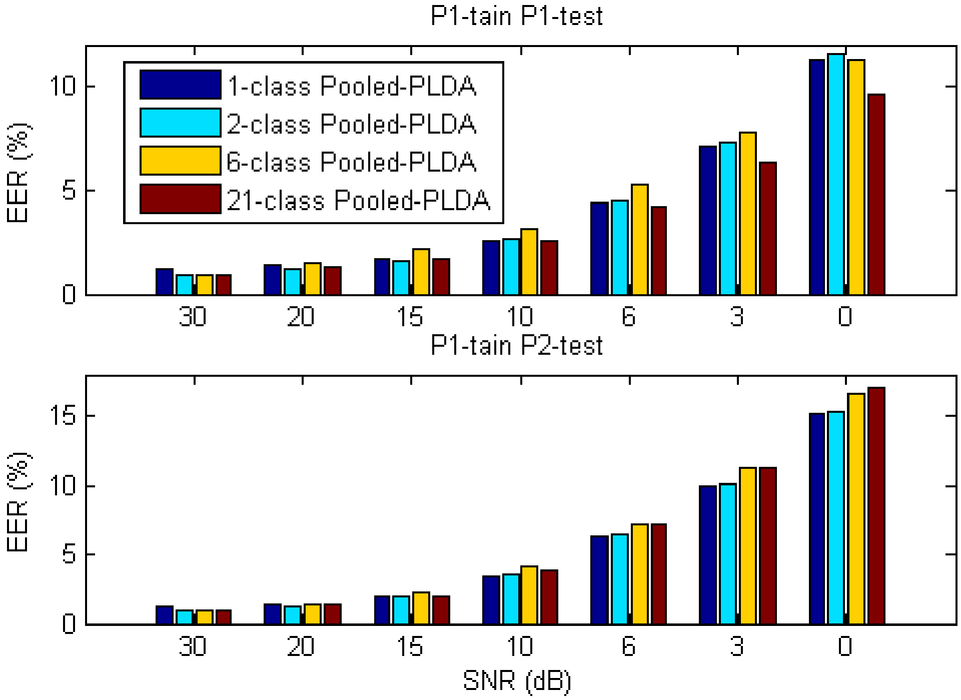


Figure 6.6: Comparison of multi-classifier architectures based on the number of classifiers. The performance is averaged across noise types and presented for each SNR. The top panel shows results for systems trained and tested on P1. The bottom panel shows results for systems trained on P1 and tested on P2 (unseen conditions).

6.5.7. Analysis Mixing Coefficients

As described in Section 6.3.3, the final score for a verification trial is obtained by combining the scores of each classifier based on the posterior probability of the subsystem given the trial at hand. Therefore, the success of this approach relies heavily on the quality of these mixing coefficients.

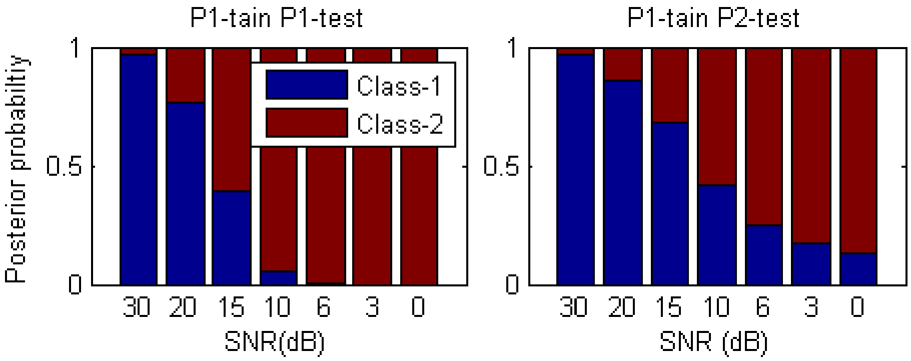


Figure 6.7: Posterior probability of a 2-cassifier Pooled-PLDA system for anticipated conditions (left) and unseen conditions (right). The probability corresponds to the portion of the bar encoded with the color of the classifier (as indicated in the legend).

Figure 6.7 shows the mixing coefficients (i.e., subsystem posterior probability) for the two-classifier Pooled-PLDA system averaged across all the verification trials of each SNR. Recall that the sub-system biased towards high-SNRs was trained on (original, 30 dB, 20 dB, 10 dB), whereas the subsystem biased towards low-SNRs was trained on (original, 10 dB, 6 dB, 0 dB). The left plot corresponds to the performance for the anticipated conditions and the right plot to unseen conditions. Note that we are only displaying the results for the system trained on P1 but the same behavior is true when training with P2. Ideally, the mixing coefficient for the high-SNR sub-system should be one for trials with SNRs above 10 dB, and zero for SNRs below 10 dB. Compared to the ideal case, the results for the anticipated conditions show a bias towards the low-SNR subsystem. However, the high verification performance of the system indicates that this bias is not detrimental. In fact, similar experiments in the preliminary work reported in [73] showed that no improvement in performance was obtained by replacing the actual mixing coefficients with the ideal (oracle) ones. Moreover, the mixing coefficients for the unseen conditions are quite close to the ideal case. This shows that the system is competent in performing a gross classification of high- and low-SNR trials for noise conditions not observed during training.

6.5.8. Effects of Multicondition PLDA Parameters

In this section we study the effects of multicondition training on the speaker-specific subspace learned by a single-classifier Pooled PLDA system. In particular, we learn four speaker-specific subspaces using different partitions of the development set (original, Pall, P1, and P2) and then analyze the variability captured by the subspaces as well as the distance between them. All the subspaces are of dimension 200.

Table 6.4 shows the results of these two analyses. The quantities in parenthesis indicate the percentage of variability (i.e., $tr(\Phi\Phi^{\mathrm{T}})/tr(\Phi\Phi^{\mathrm{T}}+\Sigma)$) captured by each of the four speaker-specific subspaces. Also, the similarity between subspaces is quantified by means of the projection distance [99], which has been normalized to take values in the interval (0,1). A projection distance equal to 1 indicates that the subspaces are orthogonal. As expected, the speaker-specific subspace learned using the original set captures the largest percentage of variability. When the dataset is augmented with noisy observations, the learning algorithm correctly assigns more energy to the intersession variability component. Regarding the projection distance, the subspace learned on the original data is the farthest from the other three. Also, the distance is almost the same with respect to all of them. Additionally, the distance between the subspaces trained on P1 and P2 is almost half of their distance with respect to the subspace trained on the original data. This explains the better generalization capability of the systems trained in a multicondition fashion, and validates the idea that multicondition training exposes properties of the signal that facilitate discarding fragile components; even if the noises observed in the deployment conditions are not anticipated in the development set.

Projection distance	Pall (23.1%)	Original (32.8%)	P1 (22.8%)	P2 (23.9%)
Pall	0	0.45	0.20	0.16
Original	*	0	0.46	0.44
P1	*	*	0	0.27
P2	*	*	*	0

Table 6.4: Normalized projection distance [99] between the speaker-specific subspaces of a single-classifier Pooled-PLDA system trained on different partitions of the development data. In parenthesis is the percentage of variability captured by the speaker subspace. All the subspaces have 200 dimensions.

6.5.9. Calibration of Scores

In all the previous sections of this chapter we have reported system performance in terms of EER separated by noise type and SNR level. This was done to facilitate a detailed comparisons in terms of discriminative power alone; regardless of the potential score misalignments across noise types and SNR levels. However, at deployment time, we want our system to produce well-calibrated scores across all conditions. Hence, we dedicate this section to analyze the effects of multicondition training on score calibration[10].

In particular, we first compute an experimental upper bound in performance and then compare it with the actual performance of our system. The gap between the "oracle" and actual performances serves as an indicator of the calibration quality of the system. That is, the smaller the gap, the better the system calibration. We use

[10] Note that we are using the term calibration rather loosely to refer to score alignment across conditions.

Detection Error Tradeoff (DET) plots to show the results across all operating points. The experimental upper bound is obtained by applying a condition-dependent oracle calibration to the scores before pooling them together. A collection of noise type and SNR dependent affine calibrations was trained using the logistic regression function included in the Bosaris toolkit [100]. We refer to them as oracle calibrations because they were trained directly on the actual scores of the system instead of a held-out development set. Moreover, they were applied to the scores making use of the noise type and SNR labels. On the contrary, the actual performance of the system is obtained by simply pooling together all the scores.

Figure 6.8 shows the DET curves for the baseline system and a single-classifier Pooled-PLDA system trained on P1. Both systems are tested on P1 and P2. The oracle results are indicated in the figure's legend. A few observations are in place. First, the multicondition Pooled-PLDA system greatly outperforms the baseline system in all operating points (not just for the EER point reported in previous sections). Second, both the Pooled-PLDA and baseline systems provide better calibration in the low false-alarm region (which is the region of interest for most forensic applications) than in the high false-alarm region. Third, the effects of score misalignment seem to be proportional to the system performance and therefore the absolute value of the gap is smaller for the multicondition Pooled-PLDA system. Finally, the overall behavior of the scores indicates that both the target and non-target distributions are multimodal. This suggests that a calibration strategy that uses quality measures [100],[101] to uncover different modes could be helpful to reduce the performance gap for the high false alarm region. However, this is out of the scope of this thesis and it is not explored here.

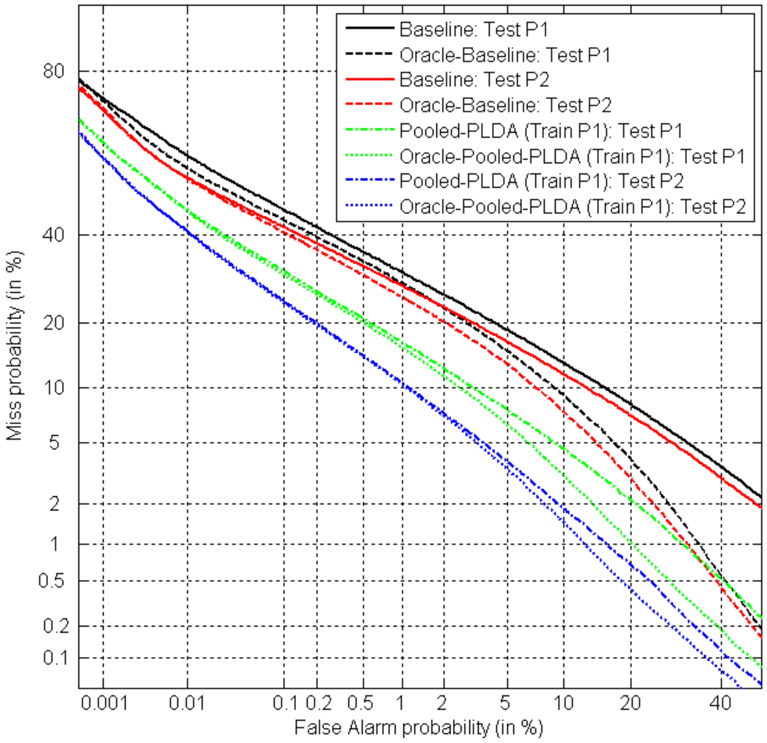


Figure 6.8: DET curves obtained by pooling together the scores produced by three systems for all conditions (noise type and SNR). The dashed lines correspond to scores preprocessed by an oracle calibration prior to pooling them to compute the DET curve. (See Section 6.5.9 for details).

6.6. Chapter summary

This chapter investigated the use of multi-classifier architectures trained on a multicondition fashion to address the problem of speaker recognition in the presence of additive noise. We used i-vector representations of the speech utterances and proposed four architectures based on PLDA models of i-vectors. A detailed description of the three building blocks of the systems (i.e., i-vector extractor, PLDA models, and score combination) was presented. Multicondition development and evaluation sets were created by adding 10 different types of noise at 7 SNRs to a subset of data from the NIST 2010 speaker recognition evaluation. Two different partitions of the data were defined to allow cross-validation and also to characterize the behavior of the systems in conditions seen during training and also unseen. Based on these data sets, a large number of experiments were conducted to compare the proposed architectures. The results of these experiments suggest a number of important guidelines to obtain optimal performance in noisy environments.

First, length normalization produces great performance improvements for multicondition and single-condition training schemes. Also, any of the multicondition approaches greatly outperforms the baseline system in both anticipated and unseen conditions. A SNR granularity beyond 6 dB is unlikely to improve performance. While a single-classifier Pooled-PLDA system is able to produce good results for a broad range of conditions, better results can be obtained with a multi-classifier architecture. In particular, after analyzing different mappings between conditions and systems, and three training strategies (pooled, tied, and individual), a 2-classifier Pooled-PLDA architecture (one targeted towards high SNR and the other towards low) was able to cover a broad range of conditions without sacrificing performance in each one of them. Moreover, the analysis of the score combination module showed that the estimated mixing coefficients were close to the ideal case; even for the unseen conditions. Finally, an analysis of the score calibration indicated that the score alignment is quite good for the low false-alarm region; however, that is not the case for the high false-alarm region. We intend to address this issue in future work by using quality-based score calibration.

Overall, we can conclude that, besides being a highly scalable solution, multicondition training of multi-classifier architectures in i-vector space, not only produces great robustness in the anticipated conditions (up to 60% average relative improvement in EER over the baseline at low SNRs), but also generalizes well to unseen conditions (up to 30% average relative improvement in EER for the noises considered in this work).

Chapter 7

Conclusions and Future Perspectives

7.1. Conclusions

In this dissertation we have advance the state-of-the-art in automatic speaker recognition based on probabilistic latent variable models of short-term spectral information that leverage large amounts of data. By doing so, we have expanded the applicability of automatic speaker recognition systems towards challenging scenarios with severe channel mismatch and environmental distortions.

After reviewing the basic principles of automatic speaker recognition systems in Chapter 2, Chapter 3 was dedicated to gaining a better understanding of the information being captured by the widely used supervector representation of speech utterances. In particular, we proposed a novel procedure for the visualization of supervectors by which qualitative insight about the information being captured was obtained. Based on this visualization approach, the Switchboard-I database (SWB-I) was used to study the relationship between a data-driven partition of the acoustic space and a knowledge based partition in terms of broad phonetic classes. The results of the analysis indicated that different subsets of supervector entries can be identified with a particular phonetic context with high probability. In light of that, a supervector can be understood as a summary of the short-term average patterns of spectral allocation of energy of a particular speaker in different phonetic contexts.

In Chapter 4, we established a connection between the Joint Factor Analysis model of speaker supervectors and signal coding using an overcomplete dictionary learned from data. Two novel ideas were proposed that resulted in algorithmic improvements. The first idea provided computational improvements by allowing a faster estimation of the JFA model hyperparameter. The second idea provided an alternative scoring technique with performance improvements.

An alternative way to handle undesired variability in supervector representations is to first project them into a lower dimensional space and then to model them in the reduced subspace. This low-dimensional projection is known as i-vector. In Chapter 5, we presented a method to boost the performance of probabilistic generative models that work with i-vector representations. First we reviewed the mathematical formulation of the i-vector representation. Then, the Gaussian PLDA and the Heavy-Tailed PLDA models were introduced. It was noted that the better performance of HT-PLDA provided strong evidence about the non-Gaussian behavior of the i-vectors. However, the success of HT-PLDA came at the expense of larger complexity and slower computation of verification scores. In light of this, we proposed to transform the i-vectors so that a linear-Gaussian model could be used. Two transformations were suggested, namely: radial Gaussianization and length normalization. Experimental validation on the telephone portion of the NIST SRE 2010 evaluation showed that by performing a simple length normalization of the i-vectors, the performance of a G-PLDA system was able to match that of a more complicated HT-PLDA one. Also, the mismatched induced by the i-vector extraction mechanism was identified as a major source of non-Gaussian behavior. Overall, using length-normalized i-vectors with the G-PLDA model was able to produced state-of-the-art performance on a challenging dataset comprising a large amount of channel variability.

Finally, in Chapter 6 we investigated the use of multi-classifier architectures trained on a multicondition fashion to address the problem of speaker recognition in the presence of additive noise. We used i-vector representations of the speech utterances and proposed four architectures based on PLDA models of i-vectors. A detailed description of the three building blocks of the systems (i.e., i-vector extractor, PLDA models, and score combination) was presented. A large number of experiments were conducted to compare the proposed architectures. The results suggested a number of important guidelines for optimal performance in noisy environments. First, length normalization produced great performance improvements for multicondition and single-condition training schemes. Also, any of the multicondition approaches greatly outperforms the baseline system in both anticipated and unseen conditions. A SNR granularity beyond 6 dB is unlikely to improve performance. While a single-classifier Pooled-PLDA system is able to produce good results for a broad range of conditions, better results can be obtained with a multi-classifier architecture. Moreover, the analysis of the score combination module showed that the estimated mixing coefficients were close to the ideal case; even for the unseen conditions. Overall, it was observed that, besides being a highly scalable solution, multicondition training of multi-classifier architectures in i-vector space, not only produced great robustness in the anticipated conditions, but also generalized well to unseen conditions. This made the proposed architecture an excellent candidate for application scenarios with additive noise.

7.2. Future Perspectives

As mentioned before, recent advances in speaker recognition are not necessarily due to new or better understanding of speaker characteristics that are informative or interpretable by humans; rather, they are the result of improvements in machine learning techniques that leverage large amounts of data. While this is perfectly valid for a large array of applications, it falls short in the case of forensic speaker recognition where interpretability of results by humans is of great importance. To address this issue, a long-term goal of my research is to modify the current recognition systems to make them informative to humans. In this way, the systems will not only provide accurate answers to the question of whether two speech samples are from the same speaker or not, but will also make more apparent what exactly it is that makes two particular voices similar or different. These systems will be useful to empirically validate our current linguistic theories as well as provide new insights about how speaker identity is conveyed in the properties of the speech signal.

Also, the overwhelming majority of the research in robust speaker recognition (including the work in this thesis) has mostly focused on ameliorating the effects of extrinsic variations (channel mismatch, noise and reverberation). Moreover, this has been done in a highly compartmentalized way by addressing the problem one variable at a time. While this is the correct way to start addressing the problem, there is a need for joint models of the phenomena; as real scenarios usually comprise complex interactions between a large collection of sources of variability—both intrinsic and extrinsic. To go in this direction, a large collection of data that represents these complex interactions is needed.

Finally, since speaker-specific information is conveyed at multiple levels in the speech signal (prosodic, phonetic, lexical, semantical, etc), tapping into these alternative sources might be the key to obtain even more robust systems.

Bibliography

[1] 2010 NIST Speaker Recognition Evaluation. [Online]. http://www.itl.nist.gov/iad/mig//tests/sre/2010/NIST_SRE10_evalplan.r6.pdf

[2] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Support Vector Machines versus Fast Scoring in the Low-Dimensional Total Variability Space for Speaker Verification," in *Interspeech 2009*, Brighton, UK, 2009.

[3] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-End Factor Analysis For Speaker Verification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, no. 4, pp. 788 - 798, May 2010.

[4] P. Kenny, "Bayesian Speaker Verification with Heavy-Tailed Priors," in *Odyssey 2010 - The Speaker and Language Recognition Workshop*, Brno, Czech Republic, 2010.

[5] B. S. Atal, "Automatic Recognition of Speakers from their Voices," *Proceedings of the IEEE*, vol. 64, no. 4, pp. 460-475, April 1976.

[6] I. Pollack, J. M. Pickett, and W. H. Sumby, "On the Identification of Speakers by Voice," *Journal of the Acoustical Society of America*, vol. 26, no. 3, pp. 403-406.

[7] L. G. Kersta, "Voiceprint Identification," *Nature*, vol. 196, p. 1253–1257, 1962.

[8] S. Pruzansky, "Pattern-Matching Procedure for Automatic Talker Recognition," *Journal of the Acoustical Society of America (JASA)*, vol. 35, no. 3, pp. 354-358, 1963.

[9] B. S. Atal, "Effectiveness of Linear Prediction Characteristics of the Speech Wave for Automatic Speaker Identification and Verification," *Journal of the Acoustical Society of America (JASA)*, vol. 55, no. 6, pp. 1304-1312, 1974.

[10] B. S. Atal, "Automatic Speaker Recognition Based on Pitch Contours," *Journal of the Acoustical Society of America (JASA)*, vol. 52, no. 6, pp. 1687-1697, 1972.

[11] G. R. Doddington, "A Method of Speaker Verification," *Journal of the Acoustical society of America (JASA)*, vol. 49, p. 139, January 1971.

[12] S. Pruzansky and M. V. Mathews, "Talker-Recognition Procedure Based on Analysis of Variance," *Journal of the Acoustical Society of America*, vol. 36, no. 11, pp. 2041-2047, 1964.

[13] S. Furui, "40 Years of Progress in Automatic Speaker Recognition,"

Proceedings of the Third International Conference on Advances in Biometrics, pp. 1050-1059, 2009.

[14] T. Kinnunen and H. Li, "An Overview of Text-independent Speaker Recognition: From Features to Supervectors," *Speech Comunication*, vol. 52, pp. 12-40, 2010.

[15] J. P. Campbell, "Speaker Recognition: A Tutorial," *Proceedings of the IEEE*, vol. 85, no. 9, p. 1437–1462, 1997.

[16] K. N. Stevens, *Acoustic Phonetics*.: The MIT Press, 2000.

[17] J. Laver, *The Phonetic Description of Voice Quality*.: Cambridge University Press, 1980.

[18] J. Kreiman and D. Sidtis, *Foundations of Voice Studies: An Interdisciplinary Approach to Voice Production and Perception*.: Wiley-Blackwell, 2011.

[19] C. H. Lee and Q. Huo, "On Adaptive Decision Rules and Decision Parameter Adaptation for Automatic Speech Recognition," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1241-1269, 2000.

[20] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A Maximum Likelihood Approach to Continuous Speech Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 5, no. 2, pp. 179-190, March 1983.

[21] D. A. Reynolds, W. Andrews, J. Campbell, J. Navratil, B. Peskin, A. Adami, Q. Jin, and D. Klusacek, "The SuperSID Project: Exploiting High-Level Information for High-Accuracy Speaker Recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Hong Kong, 2003, p. 784–787.

[22] B. Peskin, J. Navratil, J. Abramson, D. Jones, D. Klusacek, D. Reynolds, and B. Xiang, "Using Prosodic and Conversational Features for High-Performance Speaker Recognition: Report from JHU WS'02," in *International Conference on Acoustics, Speech, and Language Processing (ICASSP)*, Hong Kong, 2003, pp. 792-795.

[23] G. R. Doddington, "Speaker Recognition based on Idiolectal Differences between Speakers," in *Proceedings of EUROSPEECH-2001*, Aalborg, Denmark, 2001, pp. 2521-2524.

[24] D. F. Finch and H. O. Lira, *A Course in English Phonetics for Spanish Speakers*.: Heinemann, 1982.

[25] J. Navratil, Q. Jin, W. Andrews, and J. Campbell, "Phonetic Speaker Recognition using Maximum-Likelihood Binary-Decision Tree Modeling," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Hong Kong, 2003, pp. 796-799.

[26] M. Kockmann, L. Ferrer, L. Burget, E. Shriberg, and J. Cernocky, "Recent Progress in Prosodic Speaker Verification," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, 2011, pp. 4556-4559.

[27] Y. Lei, L. Burget, L. Ferrer, M. Graciarena, and N. Scheffer, "Towards Noise Robust Speaker Recognition Using Probabilistic Linear Discriminant Analysis," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Kyoto, Japan, 2012, pp. 4253-4256.

[28] J.P. Campbell, W. Shen, W. Campbell, R. Schwartz, J.F. Bonastre, and D. Matrouf, "Forensic Speaker Recognition," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 95-103, March 2009.

[29] J. J. Wolf, "Efficient Acoustic Parameters for Speaker Recognition," *Journal of the Acoustical Society of America (JASA)*, vol. 51, no. 6B, pp. 2044-2056, 1972.

[30] S. Furui, "Cepstral Analysis Technique for Automatic Speaker Verification," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, no. 2, pp. 254-272, April 1981.

[31] J. I. Makhoul and J. J. Wolf, "Linear Prediction and the Spectral Analysis of Speech," *IEEE Transactions on Electroacoustics*, vol. 21, no. 3, pp. 140-148, June 1973.

[32] H. Hermansky, B. A. Hanson, and H. Wakita, "Perceptually based Linear Predictive Analysis of Speech," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Tampa, Florida, 1985, pp. 509-512.

[33] S.B. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, p. 357–366, 1980.

[34] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Wiley, Ed., 1998.

[35] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support Vector Machines Using GMM Supervectors for Speaker Verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308-311, May 2006.

[36] C. M. Bishop, *Pattern Recognition and Machine Learning* , 1st ed.: Springer, 2006.

[37] 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.

[38] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical*

Society, vol. B39, no. 1, pp. 1-38, 1977.

[39] J.L. Gauvain and C. H. Lee, "Maximum a Posteriori Estimation for Multivariate Gaussian Mixture observations of Markov Chains," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 2, pp. 291-298, April 1994.

[40] H. Hermansky and N. Morgan, "RASTA Processing of Speech," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 4, pp. 578-589, 1994.

[41] J. Pelecanos and S. Sridharan, "Feature Warping for Robust Speaker Verification," in *Proceeding of Odyssey*, Crete, Greece, 2001, pp. 213-218.

[42] Robust Speaker Recognition Over Varying Channels. [Online]. http://www.clsp.jhu.edu/workshops/ws08/groups/rsrovc/

[43] R. Teunen, B. Shahshahani, and L. Heck, "A Model-Based Transformational Approach to Robust Speaker Recognition," in *Proceedings of ICSLP*, Beijing, China, 2000, pp. 495-498.

[44] D. D. Lewis, "Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval," in *Proceedings of ECML98 10th European Conference on Machine Learning*, Claire and Rouveirol, CélineEditors Nédellec, Ed.: Springer Verlag, Heidelberg, DE, 1998, pp. 4-15.

[45] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual Categorization with Bags of Keypoints," in *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004, pp. 1-22.

[46] D. Garcia-Romero and C. Y. Espy-Wilson, "Intersession Variability in Speaker Recognition: A behind the Scene Analysis," in *Proceedings of Interspeech*, Brisbane, Australia, 2008, pp. 1413-1416.

[47] T. Leung and J. Malik, "Representing and Recognizing the Visual Appearance of Materials using Three-dimensional Textons," *International Journal of Computer Vision*, vol. 43, no. 1, pp. 29-44, June 2001.

[48] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, November 2004.

[49] F. Jurie and B. Triggs, "Creating Efficient Codebooks for Visual Recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2005, pp. 604-610.

[50] J. P. Olive, A. Greenwood , and J. Coleman, *Acoustics of American English Speech: A Dynamic Approach.*: Springer, 1993.

[51] J. Godfrey, E. Holliman, and J. McDaniel, "SWITCHBOARD: Telephone Speech Corpus for Research and Development," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, San Francisco, USA, 1992,

pp. 517-520.

[52] R. Kuhn, J-C. Junqua, P. Nguyen, and N.Niedzielski, "Rapid Speaker Adaptation in Eigenvoice Space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 695-706, November 2000.

[53] O. Thyes, R. Kuhn, P. Nguyen, and J-C. Junqua, "Speaker Identification and Verification using Eigenvoices," in *Proceedings of ICSLP*, Beijing, China, 2000.

[54] P. Kenny and P. Dumouchel, "Experiments in Speaker Verification using Factor Analysis Likelihood Ratios," in *Proceedings of Odyssey04 - Speaker and Language Recognition Workshop*, Toledo, Spain, 2004.

[55] L. Burget, P. Matejka, H. Valiantsina, and J. Honza, "Investigation into variants of Joint Factor Analysis for Speaker Recognition," in *Interspeech 2009*, Brighton, 2009, pp. 1263-1266.

[56] P. Kenny, "Joint Factor Analysis of Speaker and Session Variability : Theory and Algorithms," CRIM, Montreal, (Report) CRIM-06/08-13, 2005.

[57] R. Vogt, B. Baker, and S. Sridharan, "Factor analysis subspace estimation for speaker verification with short utterances," in *Interspeech*, Brisbane, 2008.

[58] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A Study of Inter-Speaker Variability in Speaker Verification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 5, pp. 980-988, July 2008.

[59] O. Glembek, L. Burget, N. Dehak, N. Brummer, and P. Kenny, "Comparison of scoring methods used in speaker recognition with Joint Factor Analysis," in *ICASSP*, 2009, pp. 4057-4060.

[60] R. Rubinstein, A.M. Bruckstein, and M. Elad, "Dictionaries for Sparse Representation Modeling," *Proceedings of the IEEE*, 2010 (to appear).

[61] D. P. Bertsekas, *Nonlinear Programming*, 2nd ed.: Athena Scientific, 1999.

[62] R. Vogt and S. Sridharan, "Explicit Modelling of Session Variability for Speaker Verification," *Computer Speech and Language*, vol. 22, no. 1, pp. 17-38, 2008.

[63] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet,, "Support Vector Machines and Joint Factor Analysis for Speaker Verification," in *ICASSP*, 2009, pp. 4237 - 4240.

[64] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet,, "Support Vector Machines versus Fast Scoring in the Low-Dimensional Total Variability Space for Speaker Verification," in *Interspeech*, Brighton, 2009, pp. 1559-1562.

[65] A. Edelman, T. Arias, and S. Smith, "The Geometry Of Algorithms With

Orthogonality Constraints ," *SIAM J. Matrix Anal. Appl.*, 1999.

[66] N. Brummer and E. De Villiers, "The Speaker Partitioning Problem," in *Odyssey 2010 - The Speaker and Language Recognition Workshop*, Brno, Czech Republic, 2010.

[67] S. J. D. Prince, "Probabilistic Linear Discriminant Analysis for Inferences About Identity," in *IEEE 11th International Conference on Computer Vision (ICCV)*, Rio de Janeiro, 2007 , pp. 1 - 8.

[68] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of I-vector Length Normalization in Speaker Recognition Systems," in *Proceedings of Interspeech*, Florence, Italy, August 2011, pp. 249-252.

[69] P. Matejka, O. Glembek, F. Castaldo, M.J. Alam, P. Kenny, L. Burget, and J. Cernocky, "Full-Covariance UBM and Heavy-Tailed PLDA in I-Vector Speaker Verification," in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, 2011.

[70] S. Lyu and E. P. Simoncelli, "Nonlinear Extraction of Independent Components of Natural Images using Radial Gaussianization," *Neural Computation*, vol. 21, no. 6, June 2009.

[71] L. Burget, P. Matejka, P. Schwarz, O. Glembek, and J. Cernocký, "Analysis of Feature Extraction and Channel Compensation in a GMM Speaker Recognition System," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 7, pp. 1979-1986, September 2007.

[72] J. Villalba and N. Brummer, "Towards Fully Bayesian Speaker Recognition: Integrating Out the Between-Speaker Covariance," in *Proceedings of Interspeech*, Florence, Italy, August 2011.

[73] D. Garcia-Romero and C. Y. Espy-Wilson, "Multicondition Training of Gaussian PLDA Models in I-vector Space for Noise and Reverberation Robust Speaker Recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Kyoto, Japan, 2012, pp. 4257-4260.

[74] J. Baker, L. Deng, J. Glass, S. Khudanpur, C.H. Lee, N. Morgan, and D. O'Shaugnessy, "Developments and Directions in Speech Recognition and Understanding, Part 1," *IEEE Signal Processing Magazine*, vol. 26, no. 3, p. 75–80, May 2009.

[75] J. Baker, L. Deng, J. Glass, S. Khudanpur, C.H. Lee, N. Morgan, and D. O'Shaugnessy, "Developments and Directions in Speech Recognition and Understanding, Part 2," *IEEE Signal Processing Magazine*, vol. 26, no. 4, pp. 78-85, July 2009.

[76] P. J. Moreno, B. Raj, and R. M. Stern, "A Vector Taylor Series Approach for Environment-independent Speech Recognition," in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*

(ICASSP), Atlanta, GA, USA, 1996, pp. 733-736.

[77] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, "HMM Adaptation Using Vector Taylor Series for Noisy Speech Recognition," in *Proceedings of Interspeech*, Beijing, China, 2000, p. 869–872.

[78] D. Garcia-Romero, X. Zhou, and Carol Y. Espy-Wilson, "Noise-Robust Speaker Recognition in I-vector Space," *IEEE Transactions on Audio, Speech, and Language Processing*, 2012 (in review).

[79] S. Boll, "Suppression of Acoustic Noise in Speech Using Spectral Subtraction," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 27, no. 2, pp. 113-120, April 1979.

[80] H. Hermansky and N. Morgan, "RASTA Processing of Speech," *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 4, pp. 587-589, October 1994.

[81] D. Yu, L. Deng, J. Droppo, J. Wu, Y. Gong, and A. Acero, "A Minimum Mean-Square-Error Noise Reduction Algorithm on Mel-Frequency Cepstra for Robust Speech Recognition," in *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Las Vegas, Nevada, USA, 2008, pp. 4041-4044.

[82] ETSI: ETSI standard document, "Speech Processing, Transmission and Quality Aspects (STQ); Distributed Speech Recognition; Front-end feature Extraction Algorithm; Compression Algorithms, ETSI ES 202 050 v1.1.5", January 2007.

[83] R.F. Astudillo, D. Kolossa, and R. Orglmeister, "Propagation of Statistical Information Through Non-linear Feature Extractions for Robust Speech Recognition," in *Proceedings AIP Conference: MaxEnt*, 2007, pp. 245-252.

[84] R. Lippmann, E. Martin, and D. Paul, "Multi-style Training for Robust Isolated-word Speech Recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Dallas, TX, USA, 1987, pp. 705-708.

[85] H. Xu, P. Dalsgaard, Z. H. Tan, and B. Lindberg, "Noise Condition-dependent Training Based on Noise Classification and SNR Estimation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 243-2443, November 2007.

[86] D. Pearce and H.G. Hirsch, "The Aurora Experimental Framework for the Performance Evaluation of Speech Recognition Systems under Noisy Conditions," in *Proceedings of ISCA ITRW ASR*, Beijing, China, 2000, pp. 29-32.

[87] L. Deng, A. Acero, M. Plumpe, and X. Huang, "Large-vocabulary Speech Recognition Under Adverse Acoustic Environments," in *Proceedings of Interspeech*, Beijing, China, 2000, pp. 806-809.

[88] M. Akbacak and J. Hansen, "Environmental Sniffing: Noise Knowledge Estimation for Robust Speech Systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 2, pp. 465-477, February 2007.

[89] J. Fiscus, "A Post-processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER)," in *Automatic Speech Recognition and Understanding (ASRU)*, Santa Barbara, CA , USA, 1997, pp. 347-354.

[90] C. J. Leggetter and P. C. Woodland, "Maximum Likelihood Linear Regression for Speaker Adaptation of Continuous Density Hidden Markov Mdels," *Computer Speech and Language*, vol. 9, no. 2, pp. 171-185, April 1995.

[91] O. Kalinli, M. L. Seltzer, and Acero A., "Noise Adaptive Training Using a Vector Taylor Series Approach for Robust Automatic Speech Recognition," in *Proceedings International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Taipei, Taiwan, 2009, pp. 3825-3828.

[92] M.J.F. Gales and S.J. Young, "Robust Continuous Speech Recognition using Parallel Model Combination," *IEEE Transactions on Speech and Audio Processing*, vol. 4, no. 5, pp. 352-359, September 1996.

[93] B. Raj and R.M. Stern, "Missing-feature Approaches in Speech Recognition," *IEEE Signal Processing Magazine*, vol. 22, no. 5, pp. 101-116, September 2005.

[94] J. Ming, T. J. Hazen, J. R. Glass, and D. A. Reynolds, "Robust Speaker Recognition in Noisy Conditions," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 5, pp. 1711-1723, July 2007.

[95] D. Reynolds, "Channel Robust Speaker Verifcation via Feature Mapping," in *Proceedings of International Conference on Acoustic Speech and Signal Processing (ICASSP)*, Hong Kong, 2003, pp. 53-56.

[96] P. Kenny and P. Dumouchel, "Disentangling Speaker and Channel Effects in Speaker Verification," in *Proceedings of ICASSP*, Montreal, Canada, 2004, pp. 47-40.

[97] R. Vogt, B. Baker, and S. Sridharan, "Modeling Session Variability in Text-independent Speaker Verication," in *Proceedings Eurospeech*, Lisbon, Portugal, 2005, pp. 3117-3120.

[98] X. Zhou, D. Garcia-Romero, R. Duraiswami, C. Espy-Wilson, and and S. Shamma, "Linear versus Mel Frequency Cepstral Coefficients for Speaker Recognition," in *Automatic Speech Recognition and Understanding Workshop (ASRU)*, Hawaii, 2011, pp. 559-564.

[99] A. Edelman, T. Arias, and S. Smith, "The Geometry Of Algorithms With Orthogonality Constraints," *SIAM Journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303-353, April 1999.

[100] N. Brummer and E. de Villiers, "The BOSARIS Toolkit: Theory, Algorithms and Code for Surviving the New DCF," in *NIST SRE11 Workshop*, Atlanta, November 2011.

[101] D. Garcia-Romero, J. Fierrez-Aguilar, J. Gonzalez-Rodriguez, and J. Ortega-Garcia, "Using Quality Measures for Multilevel Speaker Recognition," *Computer Speech & Language*, vol. 20, no. 2-3, pp. 192-209, April-July 2006.

[102] J. L. Flanagan, "Curious Science," *IEEE Signal Processing Magazine*, vol. 26, no. 3, pp. 10-36, May 2009.