# ONE-VECTOR REPRESENTATIONS OF STOCHASTIC SIGNALS FOR PATTERN RECOGNITION

BY

HAO TANG

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2010

Urbana, Illinois

# ABSTRACT

When building a pattern recognition system, we primarily deal with stochastic signals such as speech, image, video, and so forth. Often, a stochastic signal is ideally of a one-vector form so that it appears as a single data point in a possibly high-dimensional representational space, as the majority of pattern recognition algorithms by design handle stochastic signals having a one-vector representation. More importantly, a one-vector representation naturally allows for optimal distance metric learning from the data, which generally accounts for significant performance increases in many pattern recognition tasks. This is motivated and demonstrated by our work on semi-supervised speaker clustering, where a speech utterance is represented by a Gaussian mixture model (GMM) mean supervector formed based on the component means of a GMM that is adapted from a universal background model (UBM) which encodes our prior knowledge of speakers in general. Combined with a novel distance metric learning technique that we propose, namely linear spherical discriminant analysis, which performs discriminant analysis in the cosine space, the GMM mean supervector representation of utterances leads to the state-of-the-art speaker clustering performance. Noting that the main criticism of the GMM mean supervector representation is that it assumes independent and identically distributed feature vectors, which is far from true in practice, we propose a novel one-vector representation of stochastic signals based on adapted ergodic hidden Markov models (HMMs) and a novel one-vector representation of stochastic signals based on adapted left-to-right HMMs. In these one-vector representations, a single vector is constructed based on a transformation of the parameters of an HMM that is adapted from a UBM by various controllable degrees, where the transformation is mathematically derived based on an upper bound approximation of the Kullback-Leibler divergence rate between two adapted HMMs. These one-vector representations possess a set of very attractive properties and are rather generic in nature, so they can be used with various types of stochastic signals (e.g. speech, image, video, etc.) and applied to a broad range of pattern recognition tasks (e.g. classification, regression, etc.). In addition, we propose a general framework for one-vector representations of stochastic signals for pattern recognition, of which the proposed one-vector representations based on adapted ergodic HMMs and adapted left-to-right HMMs respectively are two special cases. The general framework can serve as a unified and principled guide for constructing "the best" one-vector representations of stochastic signals of various types and for various pattern recognition tasks. Based on different types of underlying statistical models carefully and cleverly chosen to best fit the nature of the stochastic signals, "the best" one-vector representations of the stochastic signals may be constructed by a possibly nonlinear transformation of the parameters of the underlying statistical models which are learned from the stochastic signals, where the transformation may be mathematically derived from a properly chosen distance measure between two statistical models that has an elegant root in the Kullback-Leibler theory.

Since most work in this dissertation is based on HMMs, we contribute to this fascinating tool via proposing a new maximum likelihood learning algorithm for HMMs, which we refer to as the boosting Baum-Welch algorithm. In the proposed boosting Baum-Welch algorithm, we formulate the HMM learning problem as an incremental optimization procedure which performs a sequential gradient descent search on a loss functional for a good fit in an inner product function space. The boosting Baum-Welch algorithm can serve as an alternative maximum likelihood learning algorithm for HMMs to the traditional Baum-Welch or expectation-maximization (EM) algorithm, and a preferred method for use in situations where there is insufficient training data available. Compared to the traditional Baum-Welch or EM algorithm, the boosting Baum-Welch algorithm is less susceptible to the over-fitting problem (known as a general property of maximum likelihood estimation techniques) in that the boosting Baum-Welch algorithm has a tendency to produce a "large margin" effect.

*To my wife Yanna, for her love and support*

# ACKNOWLEDGMENTS

I owe my deepest gratitude to my doctoral adviser, Prof. Thomas Huang, for his unceasing personal supervision, support, encouragement and care, without which this dissertation would not have been possible. I am heartily thankful to my co-adviser, Prof. Mark Hasegawa-Johnson, for his continued guidance and support over the years. I am grateful to Prof. Stephen Levinson and Prof. Yanfeng Ouyang for spending their precious time serving on my doctoral committee and offering their invaluable comments and suggestions on my work. Very special thanks go to Dr. Stephen Chu from the IBM T.J. Watson Research Center for his close collaboration on the work of semi-supervised speaker clustering, and to Dr. Mangu Lidia and Dr. Michael Picheny for their kind support and encouragement during my visit at IBM in the summer of 2008. I owe my family, in particular my wife Ms. Yanna Wu, huge appreciation for their love, understanding and support on the long journey to this day. Lastly, I offer my regards and blessings to all of those who supported me in any respect during the completion of this dissertation.

# TABLE OF CONTENTS

# CHAPTER 1

# INTRODUCTION

The ever-expanding field of pattern recognition is primarily concerned with problems such as the automatic discovery of regularities in the data and optimal decision making or action taking (e.g., classification, regression, etc.) with the use of the discovered regularities [1]. When we design a pattern recognition system, we are mainly dealing with a particular kind of signal that we call random or stochastic and that is assumed to be generated by certain underlying sources governed by random or stochastic processes [2]. A speech utterance, an audio segment, an image, a video clip, and a piece of multimedia data, etc., are all concrete examples of stochastic signals, when they serve as the input to a pattern recognition system. Usually, when performing a pattern recognition task, we do not work directly with the raw input signals. Rather, as a preprocessing step, we compute one or more feature vectors from a raw input signal through a process known as feature extraction [3]. The feature extraction process is in general heavily domain and task dependent and is most often smartly and purposely designed by domain knowledge experts with care. In a word, the goal of feature extraction is to seek an efficient and effective representation of an input stochastic signal which can make the subsequent pattern recognition tasks easier.

The number of feature vectors that we extract from a stochastic signal can range from one to several hundreds or thousands (or even much more), depending on the stochastic signal and the specific method used for the feature extraction process. Without loss of generality, for an arbitrary stochastic signal $s$, we may assume that it can be efficiently and effectively represented by an ordered set of $n$ feature vectors

$$F(s) = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n\} \tag{1.1}$$

where $F(s)$ denotes the feature vector set of $s$ and each element of the feature vector set $\mathbf{v}_k$ is a $d$-dimensional feature vector. In this way, the raw stochastic signal $s$ may be safely discarded after feature extraction, and we need only to work with the feature vector set $F(s)$ in lieu of the raw stochastic signal $s$ in our pattern recognition tasks. Such a feature vector set representation of a stochastic signal is a general one for pattern recognition.

However, often, a stochastic signal is ideally of a single-vector or one-vector form in a possibly high-dimensional representational space. That is, given a stochastic signal, we desire to represent it by a single vector so that it appears as a single data point in a possibly high-dimensional space. The reasons for such a desire are two-fold. First, the majority of the existing pattern recognition algorithms to date, such as the k-nearest-neighbor [4], artificial neural network [5], support vector machine [6], AdaBoost [7], etc., by design handle stochastic signals having a single-vector or one-vector representation. In this sense, the benefit of a one-vector representation of stochastic signals is tremendous. We can directly apply those excellent "off-the-shelf" pattern recognition algorithms developed in the past to our specific pattern recognition tasks in hand if we can always construct a one-vector representation of the stochastic signals to be processed by these tasks. Second, many pattern recognition algorithms rely on a distance metric that provides a measure of how dissimilar two given stochastic signals are in a certain meaningful sense. Such a distance metric normally plays a vitally important role for the performance of the pattern recognition algorithms. Although there currently exist a number of distance metrics that we may use, such as the Euclidean distance, cosine distance, Mahalanobis distance [8], and so on, a distance metric that is considered optimal in some meaningful sense for a specific task and data set is usually the one that is directly learned from the data for that particular task. Undoubtedly, a one-vector representation of stochastic signals naturally allows us to learn an optimal distance metric directly from the data according to a certain criterion set for our particular pattern recognition task, and the practice of optimal distance metric learning from the data generally accounts for significant performance increases in a wide range of pattern recognition tasks.

Conceptually, a one-vector representation $\mathbf{s}$ may be constructed for a stochastic signal $s$ from its feature vector set $F(s)$

$$\mathcal{T}: \; F(s) \to \mathbf{s} \tag{1.2}$$

and this dissertation is mainly concerned with the study of the transformation process $\mathcal{T}$ for pattern recognition purposes.

Although the term "one-vector representation" is for the first time proposed by this dissertation, similar concepts and ideas have been vaguely present in the literature so far. There have existed several methods which may be used to construct a one-vector representation of stochastic signals. Among them, the most basic method is the so-called "holistic" method [9], where all the feature vectors in the feature vector set are stacked orderly to form a single high-dimensional vector $\mathbf{s} = [\mathbf{v}_1^T, \mathbf{v}_2^T, \cdots, \mathbf{v}_n^T]^T$. This holistic method has been widely used as a one-vector representation of grayscale or color images in some image-based recognition tasks such as face recognition [10], where the pixel intensities or color values (or other features extracted from the pixel locations) of an image are stacked column-wise to form a single high-dimensional vector. While the holistic method is very simple and straightforward, it suffers from several serious limitations. First, in order to generate representational vectors of the same length for all stochastic signals to be processed, the number of feature vectors extracted from different signals must be the same. This normally requires that the signals be of the same size or be scaled to the same size prior to feature extraction. Second, due to the way in which the representational vectors are formed, the feature vectors extracted from different signals are to be one-to-one corresponding. This implies that the fairly exact alignment of the signals must be required in order for the representational vectors to be useful. Third, for reasons which are obvious, a one-vector representation constructed by this method is not robust to partial occlusions or corruption in the signals. Another method is the so-called "bag of words" method [11], where a single vector is formed out of the histogram of the feature vectors in the feature vector set. This bag of words method has been very popular in many text and image classification tasks [12, 13]. While demonstrated to be a fairly good one-vector representation of stochastic signals for several application domains, this method also has a few constraints and weaknesses. First, in order to compute the histogram of the feature vectors in the feature vector set, the feature vectors must be first quantized into a set of discrete code words. In this case, loss of information will occur, and if such loss of information is important or even critical to the pattern recognition task, the results will be adversely affected. Second, the histogram is a rather coarse model for describing probability distributions.

Thus, it cannot provide an accurate estimate of the probability distribution of the feature vectors. Third, the histogram can easily break down in high-dimensional spaces. This phenomenon is commonly known as the “curse of dimensionality” [3, 1], which can prevent the method from being utilized to solve large-scale problems.

In fact, the holistic and the bag of words methods can represent the two extreme ends of the range of possible methods for constructing a one-vector representation of stochastic signals. The holistic method is the most rigid and non-probabilistic method, whereas the bag of words method is the most flexible method that is based on a special kind of non-parametric probabilistic model – the histogram. Somewhere between these two extremes lies the Gaussian mixture model (GMM) mean supervector representation of stochastic signals [14]. In the GMM mean supervector representation, a Gaussian mixture model (GMM) [15], which is a continuous parametric probabilistic model, widely used due to its excellent properties, is first learned to describe the probability distribution of the feature vectors in the feature vector set, and the mean vectors of all the Gaussian components of the learned GMM are then concatenated to form a single high-dimensional vector usually referred to as a GMM mean supervector. The GMM mean supervector representation was originally proposed by researchers in the field of speaker recognition (a.k.a. speaker identification and verification) [14], and later became popular in several areas of speech and language processing [16]. Recently, the GMM mean supervector representation was introduced to the image processing and computer vision community and has begun to receive increased attention from researchers in this field [17]. The main criticism of the GMM mean supervector representation is, however, that it assumes that the feature vectors in the feature vector set are independent and identically distributed (i.i.d.), due to the fact that a GMM is used to model the marginal probability distribution of the feature vectors rather than to model the joint probability distribution of the feature vectors. This is certainly a weakness that can lead to serious problems in real-world applications, as in practice, for most kinds of stochastic signals, the strong assumption that the feature vectors extracted from a stochastic signal are i.i.d. is far from valid.

In this dissertation, we mainly study one-vector representations of stochastic signals for pattern recognition. Notably, we propose a novel one-vector representation of stochastic signals based on adapted ergodic hidden Markov models (EHMMs) [18, 19] and a novel one-vector representation of stochastic signals based on adapted left-to-right hidden Markov models (LRHMMs). These proposed one-vector representations of stochastic signals are aimed at overcoming the limitations, constraints, and weaknesses of the aforementioned methods. Specifically, the major contributions of this dissertation are highlighted as follows.

1. We propose the conceptually new idea of, and novel strategies for, semi-supervised speaker clustering [20, 21, 22, 23], where semi-supervision here refers to the use of our prior knowledge of speakers in general to assist the unsupervised speaker clustering process. By means of an independent training data set, we encode the prior knowledge at the various stages of the speaker clustering pipeline via (1) learning a speaker-discriminative acoustic feature transformation, (2) learning a universal speaker prior model, and (3) learning a discriminative speaker subspace, or equivalently, a speaker-discriminative distance metric. We discover the directional scattering property of the GMM mean supervector representation of utterances in the high-dimensional space, and advocate the use of the cosine distance metric instead of the Euclidean distance metric for speaker clustering in the GMM mean supervector space. We propose to perform discriminant analysis based on the cosine distance metric, which leads to a novel distance metric learning algorithm – linear spherical discriminant analysis (LSDA). We show that the proposed LSDA formulation can be systematically solved within the elegant "graph embedding" general dimensionality reduction framework. Our extensive speaker clustering experiments on the GALE database clearly indicate that (1) our speaker clustering methods based on the GMM mean supervector representation and vector-based distance metrics outperform traditional speaker clustering methods based on the "bag of acoustic features" representation and likelihood-based distance metrics, (2) our advocated use of the cosine distance metric yields consistent increases in the speaker clustering performance as compared to the commonly used Euclidean distance metric, (3) our semi-supervised speaker clustering concept and strategies significantly improve the speaker clustering performance over the baselines, and (4) our proposed LSDA algorithm further leads to the state-of-the-art

speaker clustering performance. One may note that this contribution seems to stand apart from the rest of the dissertation. However, in addition to being of great value by itself, this contribution not only serves as an important motivation for the study of one-vector representations of stochastic signals for pattern recognition in this dissertation, but also helps to illustrate the essential concepts and many benefits of one-vector representations of stochastic signals (such as optimal distance metric learning from the data).

2. We propose a new maximum likelihood learning algorithm for hidden Markov models (HMMs), which we refer to as the boosting Baum-Welch algorithm [24, 25]. In the proposed boosting Baum-Welch algorithm, we formulate the HMM learning problem as an incremental optimization procedure which performs a sequential gradient descent search on a loss functional for a good fit in an inner product function space. Such a sequential optimization procedure may be used to provide a theoretical interpretation for the boosting algorithm from a very different perspective. Hence the name of the boosting Baum-Welch algorithm. The boosting Baum-Welch algorithm can serve as an alternative maximum likelihood learning algorithm for HMMs to the traditional Baum-Welch or expectation-maximization (EM) algorithm, and a preferred method for use in situations where there is insufficient training data available. Compared to the traditional Baum-Welch or EM algorithm, the boosting Baum-Welch algorithm is less susceptible to the over-fitting problem (known as a general property of maximum likelihood estimation techniques) in that the boosting Baum-Welch algorithm has a tendency to produce a "large margin" effect. Since HMMs form the basis of the one-vector representations of stochastic signals proposed in this dissertation, this contribution is relevant and important.

3. We propose a novel one-vector representation of stochastic signals based on adapted ergodic hidden Markov models (EHMMs) [18, 19] and a novel one-vector representation of stochastic signals based on adapted left-to-right hidden Markov models (LRHMMs). These one-vector representations of stochastic signals possess very attractive properties. First, the representation summarizes the probability distribution of the

feature vectors in the feature vector set compactly and accurately and allows the statistical dependence among the feature vectors to be modeled with a systematic underlying structure of first-order Markov chain. Second, the representation performs unsupervised segmentation of the stochastic signals implicitly to reveal the local structures of the signals and to allow for localized, segment-wise comparison of the signals. Third, the representation is in a one-vector form ready for either supervised, semi-supervised or unsupervised distance metric learning from the data to further reenforce its discriminatory power for classification. In addition to the above advantages, the representation is rather generic in nature and may be used with various types of stochastic signals (e.g. image, video, speech, etc.) and applied to a broad range of pattern recognition tasks (e.g. classification, regression, etc.). It does not require the signals to be of the same size, nor does it require the alignment of the signals. In addition, it is supposed to be robust to partial occlusions or corruption in the signals.

4. We propose a general framework for one-vector representations of stochastic signals for pattern recognition, of which the proposed one-vector representation based on adapted ergodic HMMs and one-vector representation based on adapted left-to-right HMMs are two special cases. The general framework claims that, based on different types of underlying statistical models carefully and cleverly chosen to best fit the nature of the stochastic signals, "the best" one-vector representations of the stochastic signals may be constructed by a nonlinear transformation of the parameters of the underlying statistical models which are learned from the stochastic signals, where the nonlinear transformation may be mathematically derived from a properly chosen distance measure between two statistical models that has an elegant root in the Kullback-Leibler (KL) theory. The general framework can serve as a unified and principled guide for constructing "the best" one-vector representations of stochastic signals of various types and for various pattern recognition tasks.

The remainder of this dissertation is organized as follows. In Chapter 2, we address the problem of speaker clustering, which is the specific task of assigning every speech utterance in an audio stream to its speaker. We propose the conceptually new idea of, and offer a complete treatment of, semi-supervised speaker clustering [20, 21, 22, 23], where semi-supervision here refers to the use of our prior knowledge of speakers in general to assist the unsupervised speaker clustering process. By means of an independent training data set, we encode the prior knowledge at the various stages of the speaker clustering pipeline via (1) learning a speaker-discriminative acoustic feature transformation, (2) learning a universal speaker prior model, and (3) learning a discriminative speaker subspace, or equivalently, a speaker-discriminative distance metric. We discover the directional scattering property of the Gaussian mixture model (GMM) mean supervector representation of utterances in the high-dimensional space, and advocate the use of the cosine distance metric instead of the Euclidean distance metric for speaker clustering in the GMM mean supervector space. We propose to perform discriminant analysis based on the cosine distance metric, which leads to a novel distance metric learning algorithm – linear spherical discriminant analysis (LSDA). We show that the proposed LSDA formulation can be systematically solved within the elegant "graph embedding" general dimensionality reduction framework. Our speaker clustering experiments on the GALE database clearly indicate that (1) our speaker clustering methods based on the GMM mean supervector representation and vector-based distance metrics outperform traditional speaker clustering methods based on the bag of acoustic features representation and likelihood-based distance metrics, (2) our advocated use of the cosine distance metric yields consistent increases in the speaker clustering performance as compared to the commonly used Euclidean distance metric, (3) our semi-supervised speaker clustering concept and strategies significantly improve the speaker clustering performance over the baselines, and (4) our proposed LSDA algorithm further leads to the state-of-the-art speaker clustering performance. This chapter, in addition to being of great value by itself, serves as an important motivation for the study of one-vector representations of stochastic signals for pattern recognition in this dissertation, and more importantly, helps to illustrate the essential concepts and many benefits of one-vector representations of stochastic signals (such as optimal distance metric learning from the data).

In Chapter 3, we provide a detailed review of the basics of HMMs [26, 27, 28]. These include the basic concepts and definitions of HMMs as well as the solutions to three fundamental problems of HMMs. Under the context of HMMs, we also introduce the concept and strategy of universal background modeling, and present several popular model adaptation techniques for HMMs. Since most of the techniques developed in this dissertation are for or based on HMMs, a fundamental review of this fascinating tool is necessary and inevitable.

In Chapter 4, we propose a new maximum likelihood learning algorithm for HMMs, which we refer to as the boosting Baum-Welch algorithm [24, 25]. In the proposed boosting Baum-Welch algorithm, the problem of HMM learning is formulated as a sequential optimization problem on a loss functional in an inner product function space instead of an iterative optimization problem on a log likelihood objective function in a model parameter space. Such a sequential optimization procedure in a function space may be used to provide a theoretical interpretation for the boosting algorithm from a very different perspective. Hence the name of boosting Baum-Welch algorithm. The boosting Baum-Welch algorithm can serve as an alternative maximum likelihood learning algorithm for HMMs to the traditional Baum-Welch or EM algorithm, and a preferred method for use in situations where there is insufficient training data available. The reason that the boosting Baum-Welch algorithm – being itself a maximum likelihood estimation technique – is less susceptible to the over-fitting problem (known as a general property of maximum likelihood estimation techniques) than the traditional Baum-Welch or EM algorithm is that by design the boosting Baum-Welch algorithm tends to produce a "large margin" effect. Our experiments show that the boosting Baum-Welch algorithm can lead to a significant performance increase in an HMM-based speech emotion classification task in the case of small-size training data sets.

In Chapter 5, we propose a novel one-vector representation of stochastic signals based on adapted ergodic HMMs [18, 19]. This one-vector representation is generic in nature and may be used with various types of stochastic signals (e.g. image, video, speech, etc.) and applied to a broad range of pattern recognition tasks (e.g. classification, regression, etc.). More importantly, by combining the one-vector representation with optimal distance metric learning (e.g. linear discriminant analysis) directly from the data, the performance of a pattern recognition system may be significantly improved. Our experiments on an image-based recognition task, namely gender recognition based on facial images, clearly demonstrate the effectiveness of the proposed one-vector representation of stochastic signals for potential use in many pattern recognition systems. To further demonstrate that the proposed one-vector representation of stochastic signals based on adapted ergodic HMMs can be an effective one-vector representation of images, we apply it to a practical application in computer vision, namely the challenging problem of automatic facial expression recognition from non-frontal view facial images [19, 29, 30]. Our experiments of recognizing six universal facial expressions over extensive multiview facial images with seven pan angles and five tilt angles (i.e. a total of 35 views), which are synthesized from the BU-3DFE facial expression database, show promising results that outperform the state of the art recently reported.

In Chapter 6, we propose a novel one-vector representation of stochastic signals based on adapted left-to-right HMMs. This one-vector representation of stochastic signals is complimentary to the one-vector representation of stochastic signals proposed in Chapter 5, and turns out to be a potentially more appropriate one-vector representation for stochastic signals such as speech and video which are sequential in nature or which have a clear temporal dimension of which the dynamics needs to be captured.

In Chapter 7, we propose a general framework for one-vector representations of stochastic signals for pattern recognition, which can serve as a unified and principled guide for constructing “the best” one-vector representations of stochastic signals of various types and for various pattern recognition tasks. The general framework claims that, based on different types of underlying statistical models carefully and cleverly chosen to best fit the nature of the stochastic signals, “the best” one-vector representations of the stochastic signals may be constructed by a nonlinear transformation of the parameters of the underlying statistical models which are learned from the stochastic signals, where the nonlinear transformation may be mathematically derived from a properly chosen distance measure between two statistical models that has an elegant root in the Kullback-Leibler (KL) theory. The one-vector representations of stochastic signals proposed in Chapters 5 and 6 are considered as two special cases of this general framework, with the underlying statistical models being chosen as adapted ergodic HMMs and adapted left-to-right HMMs, respectively.

Finally, Chapter 8 concludes the dissertation, with a brief discussion of the future research directions.

# CHAPTER 2

# SEMI-SUPERVISED SPEAKER CLUSTERING

In this chapter, we address the problem of speaker clustering [31, 32, 33, 34, 35, 36]. Speaker clustering aims to assign every speech utterance in an audio stream to its respective speaker, and is an essential part of a task known as speaker diarization [37, 38, 39, 40, 41, 42]. Also referred to as speaker segmentation and clustering, or "who spoken when," speaker diarization is the process of partitioning an input audio stream into temporal regions of speech signal energy contributed from the same speakers. A typical speaker diarization system consists of three stages. The first is the speech detection stage, where we find the portions of speech in the audio stream. The second is the segmentation stage, where we find the locations in the audio stream likely to be change points between speakers. At this stage, we often over-segment the audio stream, resulting in only one single speaker in each segment. The last is the clustering stage, where we associate the segments from the same speakers together. Figure 2.1 illustrates the process of speaker diarization. In this chapter, we mainly focus on the clustering stage, not only because the clustering stage is the most important part of speaker diarization, but also because most techniques developed for the clustering stage can be readily applied to the segmentation stage (for example, with the help of a sliding window of fixed or variable length).

Unlike speaker recognition (i.e. identification and verification), where we have training data for the speakers and thus recognition can be done supervised, speaker clustering is usually performed in a completely unsupervised manner. The output of speaker clustering is the internal labels of speakers (e.g. spk1, spk2, etc.) rather than their real identities (e.g. Tom, Mary, etc.). An interesting question is: Can we do semi-supervised speaker clustering? That is, can we make use of all available information that can be helpful to speaker clustering?

Our answer to this question is affirmative. It is worth noting that a few

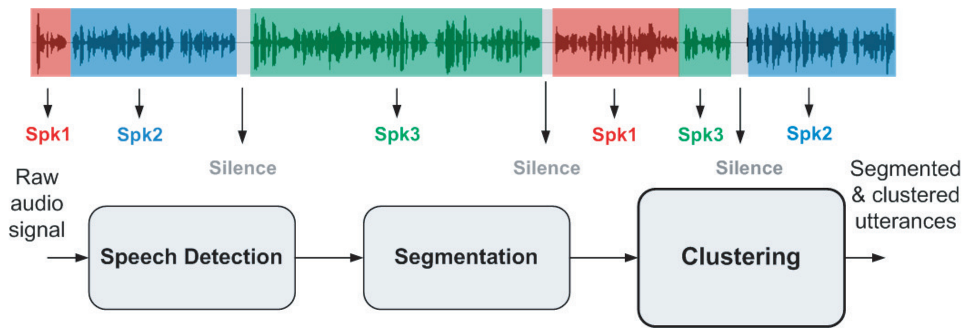

Figure 2.1: The process of speaker diarization. A typical speaker diarization system consists of a speech detection stage, a segmentation stage, and a clustering stage.

researchers in the field of speaker diarization have already tried to incorporate some prior knowledge into their methods and indeed gained noticeable improvements on the performance. For example, the use of a universal background model (UBM) for adapted Gaussian mixture model (GMM) based clustering was attempted in [37] and [38], and the GMM mean supervector as the utterance representation was recently adopted in [41] and [42]. However, none of the above work addresses every facet of the problem. In this chapter, we offer a complete treatment of the conceptually new idea of semi-supervised speaker clustering, where semi-supervision refers to the use of our prior knowledge of speakers in general to assist the unsupervised speaker clustering process. By means of an independent training data set, we acquire the prior knowledge by (1) learning a speaker-discriminative acoustic feature transformation, (2) learning a universal speaker prior model (i.e. a UBM) which is then adapted to the individual utterances to form the GMM mean supervector representation possessing the very nice directional scattering property, and (3) learning a discriminative speaker subspace, or equivalently, a speaker-discriminative distance metric.

## 2.1 The Speaker Clustering Pipeline

Figure 2.2 is a general speaker clustering pipeline. Basically, there are four critical elements in any speaker clustering algorithm and it is these elements that make a difference. We incorporate our prior knowledge of speakers into the various stages of this pipeline through an independent training data

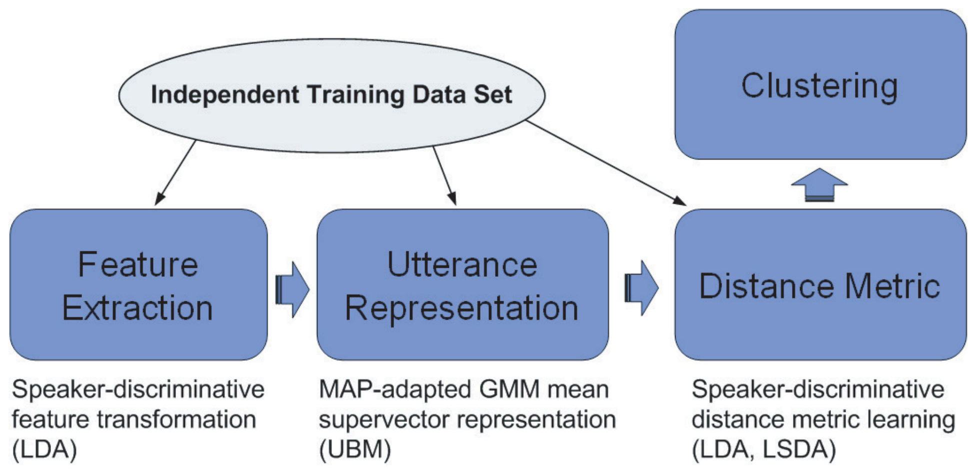
Independent Training Data Set
Clustering
Feature Extraction
Utterance Representation
Distance Metric
Speaker-discriminative feature transformation (LDA)
MAP-adapted GMM mean supervector representation (UBM)
Speaker-discriminative distance metric learning (LDA, LSDA)

Figure 2.2: The general speaker clustering pipeline. There are four essential elements in any speaker clustering algorithm and it is these elements that do make a difference.

set. First, at the feature extraction stage, we learn a speaker-discriminative acoustic feature transformation based on linear discriminant analysis (LDA) [3]. Second, at the utterance representation stage, we adopt the *maximum a posteriori* (MAP) adapted GMM mean supervector representation [14] based on a UBM [43], which can be considered as a universal speaker prior model. Third, at the distance metric stage, we learn a speaker-discriminative distance metric through a novel algorithm – linear spherical discriminant analysis (LSDA). Note that at the clustering stage, conventional clustering techniques such as k-means [44] and hierarchical clustering [45] can be naturally employed.

## 2.2 Feature Extraction

### 2.2.1 Acoustic Features

The first stage of the speaker clustering pipeline is feature extraction. Feature extraction is the process of identifying the most important cues from the measured data while removing unimportant ones for a specific task or purpose based on domain knowledge. For speaker clustering, the most widely used features are the short-time spectrum envelop based acoustic features such as the mel-frequency cepstral coefficients (MFCC) [46] and perceptual linear predictive (PLP) coefficients [47]. Although MFCC and PLP were not originally designed for representing the information relevant to distinguishing among different speakers — in fact, their primary use is in speech recognition — they work reasonably well for speaker clustering in practice. The higher-order MFCCs (e.g. 13-19) are known to correspond to the source characteristics in the source-filter model of speech production [48], and thus convey speaker information. In order to account for the temporal dynamics of the spectrum changes, the basic MFCC or PLP features are usually augmented by their first-order derivatives (i.e. the delta coefficients) and second-order derivatives (i.e. the acceleration coefficients). Higher-order derivatives may be used too, although that is rarely seen. These derivatives incorporate the time-evolving properties of the speech signal and are expected to increase the robustness of the acoustic features.

### 2.2.2 Speaker-Discriminative Acoustic Feature Transformation

The use of first and second order derivatives of the basic acoustic features introduces the characterization of the temporal dynamics of the spectrum changes. However, such characterization is completely unsupervised, and thus lacks the potential to discriminate between speakers. Using an independent training data set, we can simultaneously characterize the temporal dynamics of the spectrum changes and maximize the discriminative power of the augmented acoustic features based on a discriminative learning framework. Specifically, we first compute 13 PLP features for every speech frame with cepstral mean subtraction (CMS) and cepstral variance normalization (CVN) to compensate for the inter-section and inter-channel variability [49]. Then, instead of augmenting the basic PLP features by their first and second order derivatives, we augment them by the basic PLP features of the neighboring frames spanning a window centered on the current frame. More precisely, the PLP features of the current frame, those of the $K_L$ (e.g. 4) frames to the left and those of the $K_R$ (e.g. 4) frames to the right are concatenated to form a high-dimensional feature vector, referred to as the context-expanded feature vector. In the context-expanded feature vector space, we learn a speaker-discriminative acoustic feature transformation by

LDA based on the known speaker labels of the independent training data set. The context-expanded feature vectors can then be projected onto a low-dimensional (e.g. 40) speaker-discriminative feature subspace, which is expected to provide optimal speaker separability. Note that this is one of the strategies by which we implement our semi-supervised speaker clustering concept.

In the experiment section, we specifically compare the proposed LDA transformed acoustic features with the acoustic features traditionally augmented with the first and second order derivatives and show that the LDA transformed acoustic features outperform the traditional acoustic features on speaker clustering under the same clustering conditions. This validates that the proposed speaker-discriminative acoustic feature transformation strategy can provide a better frontend to speaker clustering as compared to traditional ones.

## 2.3 Utterance Representation

### 2.3.1 Bag of Acoustic Features Representation

The second stage of the speaker clustering pipeline is utterance representation. Utterance representation, as its name suggests, is the way to compactly encode the acoustic features of an utterance. In the literature of speaker clustering, the mainstream utterance representation is the so-called "bag of acoustic features" representation where the acoustic feature vectors are described by a statistical model such as a Gaussian or GMM. The rationale behind this representation is that in speaker clustering the linguistic content of the speech signal is considered to be irrelevant and normally disregarded. Thus, temporal independence between inter-frame acoustic features is assumed.

Most often, due to its unimodal nature, a single Gaussian is far from being sufficient to model the probability distribution of the acoustic features of an utterance, and a GMM is preferred. The theoretical property that a GMM can approximate any continuous probability density function (PDF) arbitrarily closely given a sufficient number of Gaussian components makes the GMM a popular choice for parametric PDF estimators.

The acoustic features of an utterance are modeled by an $m$-component GMM, defined as a weighted sum of $m$ component Gaussian densities

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{m} w_i N(\mathbf{x}|\mu_i, \Sigma_i) \tag{2.1}$$

where $\mathbf{x}$ is a $d$-dimensional random vector, $w_i$ is the $i^{th}$ mixture weight, and $N(\mathbf{x}|\mu_i, \Sigma_i)$ is a multivariate Gaussian PDF

$$N(\mathbf{x}|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu_i)^T \Sigma_i^{-1}(\mathbf{x}-\mu_i)} \tag{2.2}$$

with mean vector $\mu_i$ and covariance matrix $\Sigma_i$. Weight $w_i$ can be interpreted as the *a priori* probability that an observation of $\mathbf{x}$ comes from the source governed by the $i^{th}$ Gaussian distribution. Thus it satisfies the properties $0 \leq w_i \leq 1$ and $\sum_{i=1}^{m} w_i = 1$. A GMM is completely specified by its parameters $\lambda = \{w_i, \mu_i, \Sigma_i\}_{i=1}^{m}$ and the estimation of the PDF reduces to finding the proper values of $\lambda$.

A central problem of the GMM is how to estimate the model parameters $\lambda$. This problem can be practically solved by maximum likelihood estimation (MLE) techniques such as the expectation-maximization (EM) algorithm [50]. However, a fundamental drawback of MLE is that it suffers from insufficient training data. The number of free parameters of a GMM, $p$, depends on the feature dimension $d$ and the number of Gaussian components $m$. More precisely, $p = md^2/2+3md/2+m-1$, which grows linearly in $m$ but quadratically in $d$. In order to alleviate this "curse of dimensionality" [1], diagonal covariance matrices are often used in the Gaussian components. In this case, $p = 2md + m - 1$, which grows linearly in both $m$ and $d$.

### 2.3.2 GMM Mean Supervector Representation

A relatively new utterance representation that has emerged in the speaker recognition area is the GMM mean supervector representation, which is obtained by concatenating the mean vectors of the Gaussian components of a GMM trained on the acoustic features of a particular utterance [51].

UBM and MAP Adaptation

When an utterance is short, the number of acoustic feature vectors available for training a GMM is small. To avoid over-fitting (an intrinsic problem related to MLE), we can first train a single GMM on an independent training data set, leading to a well-trained GMM known as the UBM in the speaker recognition literature. Since the amount of data used to train the UBM is normally large, and the data is fairly evenly distributed across many speakers, the UBM is believed to provide a good representation of speakers in general. Therefore, it can be considered as a universal speaker prior model in which we encode the common characteristics of different speakers. Given a specific utterance, we can then derive a target GMM by adapting the UBM to the acoustic features of the utterance. This is done by MAP adaptation [52].

MAP adaptation starts with a prior model (i.e. the UBM) and iteratively performs EM estimation. In the E step, the posterior probability of a training vector falling into every Gaussian component is computed

$$p(i|\mathbf{x}_t) = \frac{w_{0i} N(\mathbf{x}_t|\mu_{0i}, \Sigma_{0i})}{\sum_{j=1}^{m} w_{0j} N(\mathbf{x}_t|\mu_{0j}, \Sigma_{0j})}, \quad i = 1, 2, \cdots, m \tag{2.3}$$

Note that Equation (2.3) is the probability that we re-assign the training vector $\mathbf{x}_t$ to the $i^{th}$ Gaussian component of the UBM $\lambda_0 = \{w_{0i}, \mu_{0i}, \Sigma_{0i}\}_{i=1}^{m}$. Based on these posterior probabilities, we compute the sufficient statistics of the training data

$$n_i = \sum_{t=1}^{T} p(i|\mathbf{x}_t) \tag{2.4}$$

$$E_i = \frac{1}{n_i} \sum_{t=1}^{T} p(i|\mathbf{x}_t)\mathbf{x}_t \tag{2.5}$$

$$E_i^2 = \frac{1}{n_i} \sum_{t=1}^{T} p(i|\mathbf{x}_t)\mathbf{x}_t^2 \tag{2.6}$$

In the M step, the sufficient statistics of the training data are combined with the prior model sufficient statistics by interpolation. The new model parameters are obtained as follows:

$$w_i' = [\alpha_i n_i / T + (1 - \alpha_i) w_i]\delta \tag{2.7}$$

$$\mu_i' = \beta_i E_i + (1 - \beta_i)\mu_i \tag{2.8}$$

$$\sigma_i'^2 = \gamma_i E_i^2 + (1 - \gamma_i)(\sigma_i^2 + \mu_i^2) - \mu_i'^2 \tag{2.9}$$

where $\delta$ is a scaling factor computed over all new mixture weights to ensure that they sum to unity. The interpolation coefficients in Equations (2.7)-(2.9) are data dependent and automatically determined for every Gaussian component using the empirical formula $\nu_i = n_i/(n_i + r^\nu)$ where $\nu \in \{\alpha, \beta, \gamma\}$ and $r^\nu$ is a fixed relevance factor for $\nu$. This empirical formula offers a smart mechanism to control the balance between the new and old sufficient statistics. Figure 2.3 demonstrates the basic idea of MAP adaptation for a GMM.

### GMM Mean Supervectors

The GMM mean supervector representation of an utterance is obtained by first MAP adapting the UBM to the acoustic features of the utterance and then concatenating the component mean vectors of the target GMM to form a long column vector. Figure 2.4 gives a block diagram that shows how a GMM mean supervector is generated.

Typically, only the component means are adapted. Once a target GMM is obtained for an utterance, its component means are stacked to form a GMM mean supervector

$$\mathbf{s} = [\mu_1{}^T \quad \mu_2{}^T \quad \dots \quad \mu_m{}^T]^T \tag{2.10}$$

It is numerically beneficial to subtract the mean supervector of the UBM

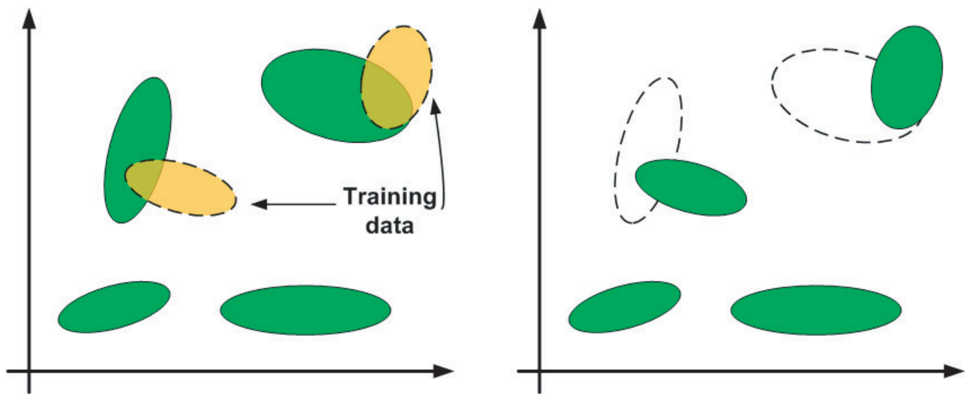

Figure 2.3: The basic idea of MAP adaptation. MAP adaptation starts with a prior model and iteratively performs EM estimation.

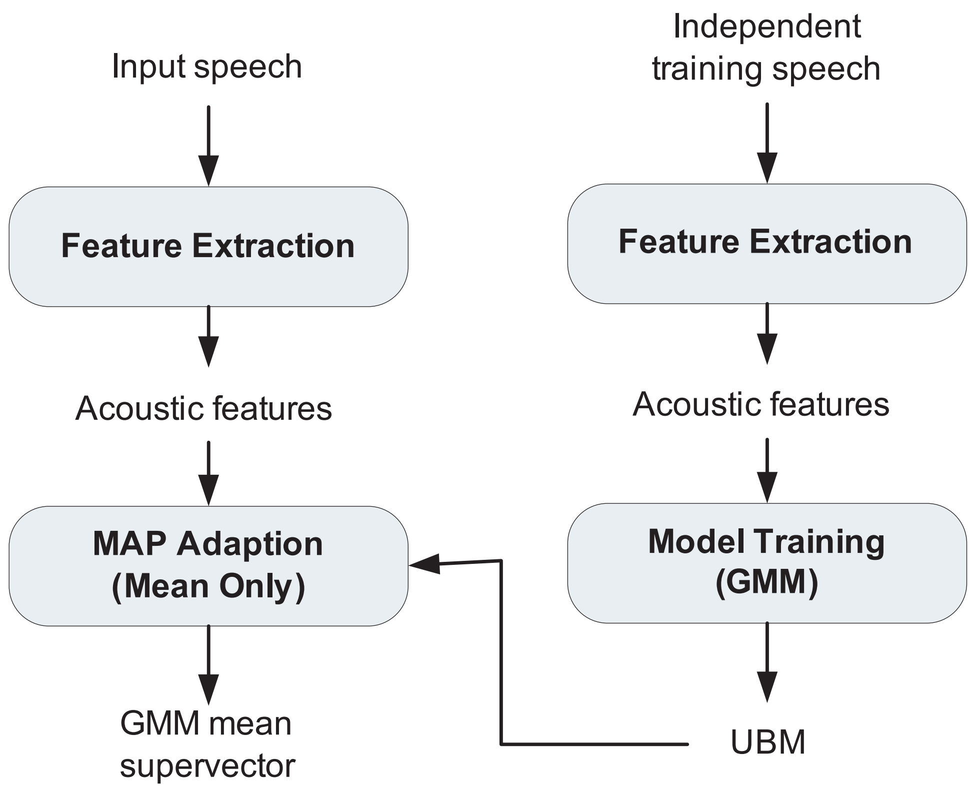

Figure 2.4: The generation of a GMM mean supervector. A GMM mean supervector is obtained by MAP adapting only the component means of a UBM.

from a GMM mean supervector, namely,

$$\mathbf{s}' = \mathbf{s} - \mathbf{s}_0 \tag{2.11}$$

where $\mathbf{s}_0$ is the mean supervector of the UBM. Without causing any ambiguity, we call $\mathbf{s}'$ (instead of $\mathbf{s}$) a GMM mean supervector. A complete set of GMM mean supervectors forms a high-dimensional space called the GMM mean supervector space.

## Property of GMM Mean Supervectors

The GMM mean supervector is an effective utterance representation that has been applied to speaker recognition. However, it has come to our attention that the use of the GMM mean supervector representation for speaker clustering is still rare. The GMM mean supervector representation allows us to represent an utterance as a single data point in a high-dimensional space,

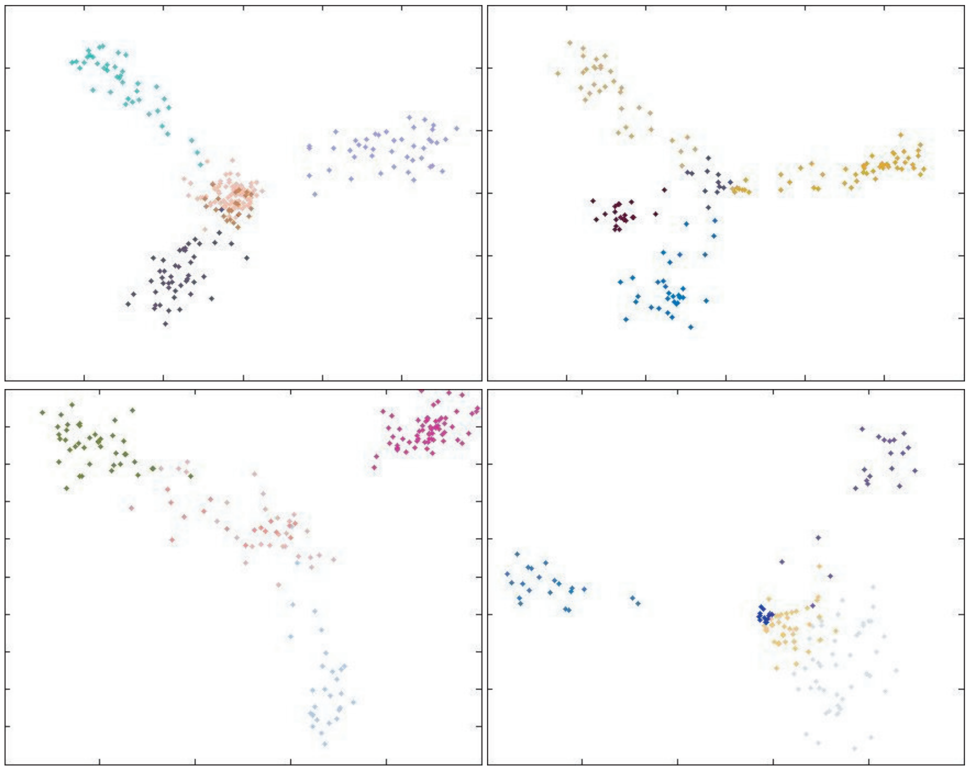

Figure 2.5: The property of GMM mean supervectors. The data points show strong directional scattering patterns.

where conventional clustering techniques such as k-means and the hierarchical clustering can be naturally applied.

Figure 2.5 visualizes the GMM mean supervectors of many utterances from five different speakers using 2D scatter plots of their two principal components. In each plot, the different speakers are shown in different colors. For each speaker, there are about 150 utterances, denoted by small dots. As one can see, the data points belonging to the same speaker tend to cluster together. Thus, the Euclidean distance metric is a reasonable choice for speaker clustering in the GMM mean supervector space. However, one can also observe that the data points show very strong directional scattering patterns. The directions of the data points seem to be more informative and indicative than their magnitudes. This observation motivated us to favor the cosine distance metric over the Euclidean distance metric for speaker clustering in the GMM mean supervector space.

A reasonable explanation as to why the GMM mean supervectors show strong directional scattering patterns is that when we perform mean-only

MAP adaptation, only a subset of the UBM component means is adjusted, and the particular subset that is adjusted seems to be rather speaker-dependent. Hence, the speaker-specific information is encoded in those component means which are adapted. Therefore, the utterances from the same speaker tend to yield a cluster of GMM mean supervectors that scatter in a particular direction in the GMM mean supervector space.

As presented later in the experiment section, our experiment results on all speaker clustering tasks clearly demonstrate that the cosine distance metric consistently outperforms the Euclidean distance metric when using the GMM mean supervector as the utterance representation. This strongly supports our discovery of the directional scattering property of the GMM mean supervectors and forms the foundation of our original motivation to perform discriminant analysis in the cosine distance metric space.

## 2.4 Distance Metric

The third stage of the speaker clustering pipeline is distance metric. The distance metrics that can be used for speaker clustering are closely related to the particular choice of utterance representations. Two popular categories of distance metrics, namely likelihood-based distance metrics and vector-based distance metrics, are widely used for the two corresponding utterance representations, respectively.

### 2.4.1 Likelihood-Based Distance Metrics

For the bag of acoustic features utterance representation, the distance metric should represent some measure of the distance between two statistical models. A famous likelihood-based distance metric extensively used for speaker clustering is the Bayesian information criterion (BIC) [53]. For a given utterance, the BIC value indicates how well a model fits the utterance and is given by

$$BIC(M_i) = \log L(X_i|M_i) - \frac{\lambda}{2} k_i \log(n_i) \qquad (2.12)$$

where $L(X_i|M_i)$ is the likelihood of the acoustic features $X_i$ given the model $M_i$, $\lambda$ is a design parameter, $k_i$ is the number of free parameters in $M_i$, and $n_i$

is the number of feature vectors in $X_i$. The distance between two utterances $X_i$ and $X_j$ is given by the $\Delta BIC$ value. If we assume $M_i$ and $M_j$ are both Gaussian, then $\Delta BIC$ is given by

$$\Delta BIC(X_i, X_j) = n \log \Sigma - n_i \log \Sigma_i - n_j \log \Sigma_j - \lambda P \tag{2.13}$$

where $n = n_i + n_j$, $\Sigma_i$ and $\Sigma_j$ are the covariance matrices of $X_i$ and $X_j$, respectively, $\Sigma$ is the covariance matrix of the aggregate of $X_i$ and $X_j$, and $P$ is a penalty term given by

$$P = \frac{1}{2}(d + \frac{1}{2}d(d+1)) \log n \tag{2.14}$$

with $d$ being the dimension of the acoustic feature vectors.

Other likelihood-based distance metrics include the generalized likelihood ratio (GLR), Gish distance, Kullback-Leibler divergence (KLD), divergence shape distance (DSD), Gaussian divergence (GD), cross BIC (XBIC), cross log likelihood ratio (XLLR), and so forth [54]. All these metrics have been proposed for the bag of acoustic features utterance representation.

### 2.4.2 Vector-Based Distance Metrics

For the GMM mean supervector utterance representation, since an utterance can be represented as a single data point in a high-dimensional vector space, the most often used distance metric is the Euclidean distance metric

$$d(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T (\mathbf{x} - \mathbf{y}) \tag{2.15}$$

As we discussed earlier, in the GMM mean supervector space, the data points belonging to the same speaker tend to cluster together. Thus, the Euclidean distance metric is a reasonable choice for speaker clustering in the GMM mean supervector space. However, we observe that the data points show very strong directional scattering patterns. The directions of the data points seem to be more informative and indicative than their magnitudes. This observation motivated us to advocate the use of the cosine distance metric instead of the Euclidean distance metric for speaker clustering in the GMM mean supervector space. The cosine distance metric is a measure of the angle between two vectors in the space and is irrelevant to the norms of the vectors. It is defined as

$$d(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x}^T\mathbf{y}}{\sqrt{\mathbf{x}^T\mathbf{x}}\sqrt{\mathbf{y}^T\mathbf{y}}} \tag{2.16}$$

Our experiments show that the cosine distance metric consistently outperforms the Euclidean distance metric for speaker clustering in the GMM mean supervector space.

### 2.4.3 Distance Metric Learning vs. Linear Subspace Learning

Although the Euclidean and cosine distance metrics can be directly used, they are optimal only if the data points are uniformly distributed in the entire space. In a high-dimensional space, most often the data points lie in or near a low-dimensional manifold, or preferably a linear subspace, of the original space. In this case, it is extremely advantageous if we can learn an optimal distance metric for the data.

We define a generalized Euclidean distance metric between two data points $\mathbf{x}$ and $\mathbf{y}$ as

$$d(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T A(\mathbf{x} - \mathbf{y}) \tag{2.17}$$

where the positive definite matrix $A$ is intended to compensate for the non-uniform data distribution. If $A$ coincides with the covariance matrix of the data, this generalized Euclidean distance metric reduces to the Mahalanobis distance [8]. Equation (2.17) can be re-written as

$$\begin{aligned} d(\mathbf{x}, \mathbf{y}) &= (A^{\frac{1}{2}}\mathbf{x} - A^{\frac{1}{2}}\mathbf{y})^T(A^{\frac{1}{2}}\mathbf{x} - A^{\frac{1}{2}}\mathbf{y}) \\ &= (W\mathbf{x} - W\mathbf{y})^T(W\mathbf{x} - W\mathbf{y}) \end{aligned} \tag{2.18}$$

That is, the generalized Euclidean distance metric between $\mathbf{x}$ and $\mathbf{y}$ can be re-organized as the Euclidean distance metric between two linearly transformed data points $W\mathbf{x}$ and $W\mathbf{y}$ where $W = A^{\frac{1}{2}}$.

Similarly, we define a generalized cosine distance metric between $\mathbf{x}$ and $\mathbf{y}$

as

$$\begin{aligned} d(\mathbf{x},\mathbf{y}) &= 1-\frac{\mathbf{x}^T A\mathbf{y}}{\sqrt{\mathbf{x}^T A\mathbf{x}}\sqrt{\mathbf{y}^T A\mathbf{y}}} \\ &= 1-\frac{(A^{1/2}\mathbf{x})^T(A^{1/2}\mathbf{y})}{\sqrt{(A^{1/2}\mathbf{x})^T(A^{1/2}\mathbf{x})}\sqrt{(A^{1/2}\mathbf{y})^T(A^{1/2}\mathbf{y})}} \\ &= 1-\frac{(W\mathbf{x})^T(W\mathbf{y})}{\sqrt{(W\mathbf{x})^T(W\mathbf{x})}\sqrt{(W\mathbf{y})^T(W\mathbf{y})}} \end{aligned} \tag{2.19}$$

Likewise, the generalized cosine distance metric between $\mathbf{x}$ and $\mathbf{y}$ is the cosine distance metric between two linearly transformed data points $W\mathbf{x}$ and $W\mathbf{y}$ where $W = A^{\frac{1}{2}}$. In this sense, it is clear that learning an optimal distance metric is equivalent to learning an optimal linear subspace of the original high-dimensional space. There exist various linear subspace learning methods that can fit into this context.

### 2.4.4 Distance Metric Learning in Euclidean Space

Linear subspace learning can be classified into two distinct categories: unsupervised learning and supervised learning. For unsupervised linear subspace learning, principal component analysis (PCA) [3] may be the most early developed and prevailing technique and, when applied to speech or speaker recognition, is known as the eigenvoice approach [55]. Other more recent unsupervised learning techniques include the locality preserving projection (LPP) [56], neighborhood preserving embedding (NPE) [57], etc. All these techniques may be applied to speaker clustering. However, we are most interested in supervised learning since the goal of speaker clustering is related to classification. It is natural that we prefer a learning technique that is discriminative rather than generative. The most famous technique for supervised linear subspace learning is Fisher's LDA. LDA has been applied to speaker clustering, and the resulting technique is termed the fishervoice approach [22]. The term "fishervoice" is analogous to "fisherface" in the face recognition literature, where the fisherface approach refers to the face recognition method based on LDA while the eigenface approach refers to the face recognition method based on PCA.

### 2.4.5 Distance Metric Learning in Cosine Space

Most existing linear subspace learning techniques (e.g. PCA and LDA) are implicitly based on the Euclidean distance metric. As we mentioned earlier, due to the directional scattering property of the GMM mean supervectors, we favor the cosine distance metric over the Euclidean distance metric for speaker clustering in the GMM mean supervector space. Therefore, we propose to perform discriminant analysis in the cosine distance metric space.

#### Linear Spherical Discriminant Analysis

We coined the phrase "spherical discriminant analysis" (SDA) to denote discriminant analysis in the cosine distance metric space. We define a projection from a $d$-dimensional linear space to an $h$-dimensional hypersphere where $h < d$

$$\mathbf{y} = \frac{W^T\mathbf{x}}{\|W^T\mathbf{x}\|} \tag{2.20}$$

We note that such a projection is nonlinear. However, under two mild conditions, this projection can be linearized. One condition is that the objective function for learning the projection only involves the cosine distance metric. The other condition is that only the cosine distance metric is used in the projected space. In this case, the norm of the projected vector $\mathbf{y}$ has an impact on neither the objective function nor distance computation in the projected space. Thus, the denominator term of Equation (2.20) can be safely dropped, leading to a linear projection $\mathbf{y} = W^T\mathbf{x}$, which is called "linear spherical discriminant analysis" (LSDA). Figure 2.6 illustrates the basic ideas of SDA and LSDA.

Formally speaking, the goal of LSDA is to seek a linear projection $W$ such that the average within-class cosine similarity of the projected data is maximized while the average between-class cosine similarity of the projected data is minimized. Assuming that there are $c$ classes, the average within-class cosine similarity is defined to be the average of the class-dependent average cosine similarities between the projected data vectors. It can be written in terms of the unknown projection matrix $W$ and original data points $\mathbf{x}$

$$S_W = \frac{1}{c}\sum_{i=1}^{c} S_i \tag{2.21}$$

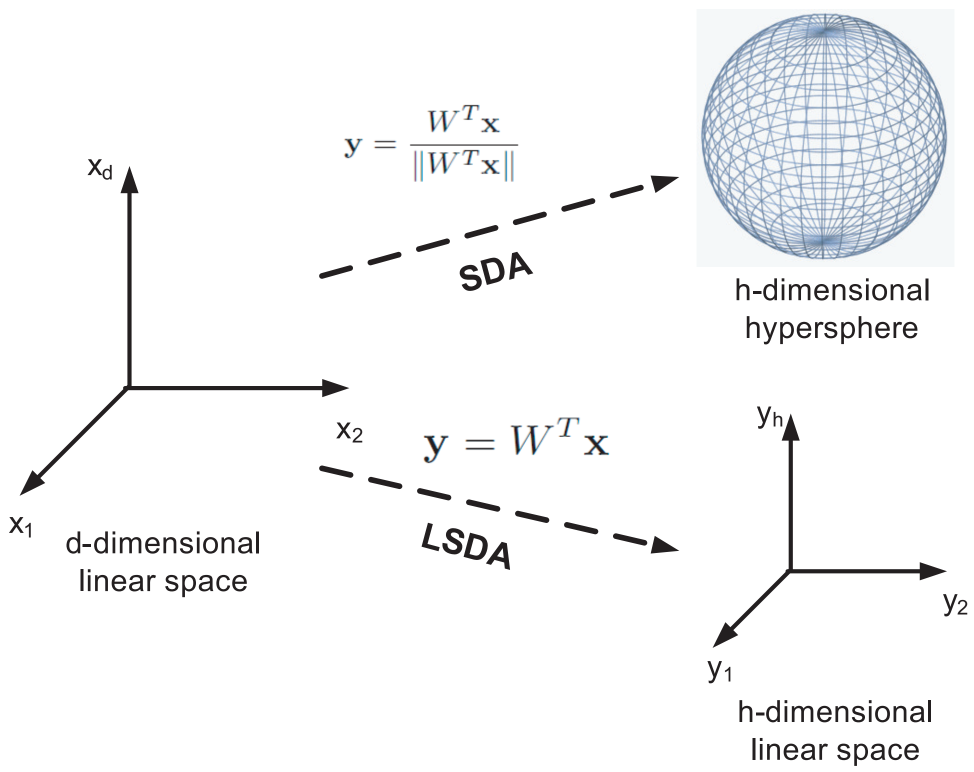
$\mathbf{y} = \frac{W^T\mathbf{x}}{\|W^T\mathbf{x}\|}$
SDA
h-dimensional hypersphere
$x_d$
$x_2$
$x_1$
d-dimensional linear space
$\mathbf{y} = W^T\mathbf{x}$
LSDA
$y_h$
$y_2$
$y_1$
h-dimensional linear space

Figure 2.6: The schematic illustration of SDA and LSDA. Under two mild conditions, the nonlinear projection can be linearized and thus SDA reduces to LSDA.

$$\begin{aligned} S_i &= \frac{1}{|D_i||D_i|} \sum_{\mathbf{x}_j,\mathbf{x}_k \in D_i} \frac{\mathbf{y}_j^T \mathbf{y}_k}{\sqrt{\mathbf{y}_j^T \mathbf{y}_j}\sqrt{\mathbf{y}_k^T \mathbf{y}_k}} \\ &= \frac{1}{|D_i||D_i|} \sum_{\mathbf{x}_j,\mathbf{x}_k \in D_i} \frac{\mathbf{x}_j^T W W^T \mathbf{x}_k}{\sqrt{\mathbf{x}_j^T W W^T \mathbf{x}_j}\sqrt{\mathbf{x}_k^T W W^T \mathbf{x}_k}} \end{aligned} \tag{2.22}$$

where $|D_i|$ denotes the number of data points in the $i^{th}$ class. Similarly, the average between-class cosine similarity is defined to be the average of the average cosine similarities between any two pairs of classes. It can be likewise written in terms of $W$ and $\mathbf{x}$

$$S_B = \frac{1}{c(c-1)} \sum_{m=1}^{c} \sum_{n=1}^{c} S_{mn} \quad (m \neq n) \tag{2.23}$$

$$
\begin{aligned}
S_{mn} &= \frac{1}{|D_m||D_n|} \sum_{\substack{\mathbf{x}_j \in D_m \\ \mathbf{x}_k \in D_n}} \frac{\mathbf{y}_j^T \mathbf{y}_k}{\sqrt{\mathbf{y}_j^T \mathbf{y}_j}\sqrt{\mathbf{y}_k^T \mathbf{y}_k}} \\
&= \frac{1}{|D_m||D_n|} \sum_{\substack{\mathbf{x}_j \in D_m \\ \mathbf{x}_k \in D_n}} \frac{\mathbf{x}_j^T W W^T \mathbf{x}_k}{\sqrt{\mathbf{x}_j^T W W^T \mathbf{x}_j}\sqrt{\mathbf{x}_k^T W W^T \mathbf{x}_k}}
\end{aligned} \tag{2.24}
$$

where $|D_m|$ and $|D_n|$ denote the number of data points in the $m^{th}$ and $n^{th}$ classes, respectively.

The LSDA criterion is to maximize $S_W$ while minimizing $S_B$, which can be written in the trace difference form

$$
W = arg \max_W (S_W - S_B) \tag{2.25}
$$

Note that there are various forms of the criterion that may be adopted. We choose the trace difference form, which is similar to the work of Ma et al. [58]. However, we systematically solve our LSDA formulation in an elegant general dimensionality reduction framework known as graph embedding [59, 60].

## Graph Embedding Solution to LSDA

Graph embedding is a general framework for dimensionality reduction. In graph embedding, an undirected weighted graph, $G = \{X, S\}$, with vertex set $X$ and similarity matrix $S$, is used to characterize certain statistical or geometrical properties of a data set. A vertex $\mathbf{x}_i$ in $X$ represents a data point in the high-dimensional space. An entry $s_{ij}$ in $S$, denoted as the weight of the edge connecting $\mathbf{x}_i$ and $\mathbf{x}_j$, represents the similarity between the two corresponding data points. The purpose of graph embedding is to represent each vertex of the graph as a low dimensional vector that preserves the similarities in $S$.

For a specific dimensionality reduction algorithm, there may exist two graphs. One is the intrinsic graph $\{X, S^{(i)}\}$, which characterizes the data properties that the algorithm aims to preserve. The other is the penalty graph $\{X, S^{(p)}\}$, which characterizes the data properties that the algorithm aims to avoid. These two graphs share the same vertex set but have different similarity matrices. The graph similarity preserving criterion is given by

$$W = arg\min_{W} \sum_{i \neq j} \|f(\mathbf{x}_i, W) - f(\mathbf{x}_j, W)\|^2 (s_{ij}^{(i)} - s_{ij}^{(p)}) \tag{2.26}$$

where $f(\mathbf{x}, W)$ is a general projection with parameters $W$. One can easily see that minimizing the above objective function ensures that if the data points $\mathbf{x}_i$ and $\mathbf{x}_j$ are close in the sense of the similarities in $S^{(i)}$ and $S^{(p)}$, then their projections in the low-dimensional space are close, too.

While the projection function $f(\mathbf{x}, W)$ can be of any form, a widely used one is the linear projection $f(\mathbf{x}, W) = W^T\mathbf{x}$. The criterion for the linear specialization of graph embedding becomes

$$W = arg\min_{W} \sum_{i \neq j} \|W^T\mathbf{x}_i - W^T\mathbf{x}_j\|^2 (s_{ij}^{(i)} - s_{ij}^{(p)}) \tag{2.27}$$

It has been shown in [61] that minimizing Equation (2.27) is equivalent to maximizing

$$\mathbf{w} = arg\max_{\mathbf{w}} \frac{\mathbf{w}^T XSX^T \mathbf{w}}{\mathbf{w}^T XDX^T \mathbf{w}} \tag{2.28}$$

where $S = S^{(i)} - S^{(p)}$, $D$ is a diagonal matrix with the diagonal elements being the column sums of $S$, namely $D_{ii} = \sum_j s_{ji}$, and the $\mathbf{w}$'s are the columns of $W$, which correspond to the eigenvectors of the generalized eigenvalue problem

$$XSX^T\mathbf{w} = \lambda XDX^T\mathbf{w} \tag{2.29}$$

With different choices of $S$, this linear graph embedding framework can lead to different linear dimensionality reduction algorithms such as LDA, LPP, NPE, etc.

However, our LSDA formulation cannot be treated in the linear graph embedding framework. Recall that SDA reduces to LSDA under two mild conditions, one of which is that only the cosine distance metric is used in the objective function for learning the projection. In the objective function in Equation (2.27), the $L_2$ norm term, which is the Euclidean distance metric, violates this condition. Thus, we need to choose a spherical projection of the form of Equation (2.20), which leads to the following criterion:

$$W = arg \min_W \sum_{i \neq j} \left\| \frac{W^T \mathbf{x}_i}{\|W^T \mathbf{x}_i\|} - \frac{W^T \mathbf{x}_j}{\|W^T \mathbf{x}_j\|} \right\|^2 (s_{ij}^{(i)} - s_{ij}^{(p)}) \tag{2.30}$$

In the above revised objective function, since the projected vectors are of unit norm, the Euclidean distance metric and the cosine distance metric are equivalent.

Although there is no closed-form solution to the optimization problem of Equation (2.30), as shown in [60], this problem can be solved using a steepest descent algorithm, with the gradient derived as

$$G = 2 \sum_{i \neq j} \left\{ \frac{f_{ij} W^T \mathbf{x}_i \mathbf{x}_i^T}{f_i^3 f_j} + \frac{f_{ij} W^T \mathbf{x}_j \mathbf{x}_j^T}{f_j^3 f_i} - \frac{W^T (\mathbf{x}_i \mathbf{x}_j^T + \mathbf{x}_j \mathbf{x}_i^T)}{f_i f_j} \right\} (s_{ij}^{(i)} - s_{ij}^{(p)}) \tag{2.31}$$

where $f_i = \sqrt{\mathbf{x}_i^T W W^T \mathbf{x}_i}$, $f_j = \sqrt{\mathbf{x}_j^T W W^T \mathbf{x}_j}$, and $f_{ij} = \mathbf{x}_i^T W W^T \mathbf{x}_j$. If we expand the $L_2$ norm term of Equation (2.30), by some simple manipulations we obtain

$$\left\| \frac{W^T \mathbf{x}_i}{\|W^T \mathbf{x}_i\|} - \frac{W^T \mathbf{x}_j}{\|W^T \mathbf{x}_j\|} \right\|^2 = 2 \left( 1 - \frac{\mathbf{x_i}^T W W^T \mathbf{x_j}}{\|W^T \mathbf{x_i}\| \, \|W^T \mathbf{x_j}\|} \right) \tag{2.32}$$

Thus, the criterion in Equation (2.30) is equivalent to the following criterion:

$$W = arg \max_W \sum_{i \neq j} \frac{\mathbf{x_i}^T W W^T \mathbf{x_j}}{\sqrt{\mathbf{x}_i^T W W^T \mathbf{x}_i} \sqrt{\mathbf{x}_j^T W W^T \mathbf{x}_j}} (s_{ij}^{(i)} - s_{ij}^{(p)}) \tag{2.33}$$

By comparing Equation (2.33) to Equations (2.21)–(2.25), we conclude that the graph embedding criterion of Equation (2.30) is the equivalent to the LSDA criterion of Equation (2.25) if the entries of the similarity matrices $S^{(i)}$ and $S^{(p)}$ are set to proper values, as follows:

$$\begin{aligned} s_{jk}^{(i)} &\leftarrow \frac{1}{c|D_i||D_i|} \quad \text{if } \mathbf{x}_j, \mathbf{x}_k \in D_i, \quad i = 1, ..., c \\ s_{jk}^{(p)} &\leftarrow \frac{1}{c(c-1)|D_m||D_n|} \quad \text{if } \mathbf{x}_j \in D_m, \mathbf{x}_k \in D_n \\ & \quad m, n = 1, ..., c, m \neq n \end{aligned} \tag{2.34}$$

That is, by assigning appropriate values to the weights of the intrinsic and penalty graphs, our LSDA formulation can be systematically solved within the elegant graph embedding general dimensionality reduction framework.

## 2.5 Clustering

The last stage of the speaker clustering pipeline is clustering. We are interested in conventional clustering techniques which can be applied to the GMM mean supervector space. We mainly focus on two traditional classes of algorithms. One is "flat" clustering – clustering by partitioning the data space. The other is hierarchical clustering, where we try to construct not a partition but a nested hierarchy of partitions. In most real-world applications, one of these two classes of algorithms is employed.

The representative flat clustering algorithm is k-means, whose objective is to partition the data space in such a way that the total intra-cluster variance is minimized. It iterates between a cluster assigning step and a mean updating step until convergence. Spherical k-means [62] is an extension of k-means that is based on the cosine distance metric.

The representative hierarchical clustering algorithm is agglomerative clustering [3] whose objective is to obtain a complete hierarchy of clusters in the form of a dendrogram. The algorithm adopts a bottom-up strategy. First, it starts with each data point being a cluster. Then, it checks which clusters are the closest and merges them into a new cluster. As the algorithm proceeds, it always merges the two closest clusters until there is only one single cluster left.

A remarkable question related to agglomerative clustering is how to determine which clusters are the closest. There exist several methods that measure the distance between two clusters, for instance, the single linkage, complete linkage, average linkage, "ward" linkage, and so on [45, 63]. We empirically discover that the "ward" linkage yields the best performance for speaker clustering. The "ward" linkage is a function that specifies the distance between two clusters $X$ and $Y$ by the increase in the error sum of squares (ESS) after

Table 2.1: Experiment settings.

| | **Test set** | **Indep. training set** |
|---|---|---|
| #speaker | 630 | 498 |
| #utterance | 19024 | 18327 |
| #utt/spk (ave.) | $30 \sim 40$ | $30 \sim 40$ |
| utt duration (ave.) | $3 \sim 4$s | $3 \sim 4$s |

merging of $X$ and $Y$

$$\begin{aligned} \bar{\mathbf{x}} &= \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i, \quad ESS(X) = \sum_{i=1}^{N}|\mathbf{x}_i - \bar{\mathbf{x}}|^2 \\ d(X,Y) &= ESS(XY) - [ESS(X) + ESS(Y)] \end{aligned} \tag{2.35}$$

## 2.6 Experiments

### 2.6.1 Experiment Setup and Protocol

We conduct extensive speaker clustering experiments on the GALE Mandarin database [64]. The GALE database contains about 1900 hours of broadcast news speech data collected from various Mandarin TV programs at various times. The waveforms were sampled at 16 kHz and quantized at 16 bits per sample, and were automatically over-segmented into short utterances using the BIC criterion, with each utterance being as pure as possible, namely, each utterance being from a single speaker. A random sample of the results was further verified by human listeners.

Our experiments are based on a test set of 630 speakers and 19024 utterances extracted from the GALE database. In order to implement our semi-supervised speaker clustering strategies at the various stages of the speaker clustering pipeline, we employ an independent training set, which was also extracted from the GALE database. Note that the test set and the independent training set were chosen in such a way that speakers in the independent training set do not exist in the test set. Table 2.1 lists the detailed experiment settings.

To guarantee a statistically significant performance comparison, we carry out our experiments as follows.

1. A case is an experiment associated with a specific number of test speakers, namely 2, 5, 10, 20, 50, and 100, respectively.
2. For each case, this number of speakers are drawn randomly from the test set, and all the utterances from the selected speakers are used in the experiment.
3. For each case, 10 independent trials are run, each of which involves a random draw of the test speakers.
4. For each case, the average of the clustering results over the 10 independent trials is reported.

### 2.6.2 Performance Evaluation Metrics

We report our experiment results based on two performance evaluation metrics, namely the clustering accuracy and the normalized mutual information (NMI) [65]. These two metrics are standard for evaluating (general) data clustering results [66]. The clustering accuracy is given by

$$r = \frac{1}{N}\sum_{i=1}^{N}[c_i = l_i] \tag{2.36}$$

where $N$ denotes the number of test utterances, $c_i$ is the cluster label of the $i^{th}$ utterance returned by the algorithm, $l_i$ is the true cluster label, and $[v]$ is an indication function which returns 1 if $v$ is true and 0 otherwise.

The NMI is another popular, information-theoretically interpreted metric given by

$$r = \frac{I(C, L)}{[H(C) + H(L)]/2} \tag{2.37}$$

where $I(C, L)$ is the mutual information

$$I(C, L) = \sum_i \sum_j \frac{|c_i \bigcap l_j|}{N} \log \frac{N|c_i \bigcap l_j|}{|c_i||l_j|} \tag{2.38}$$

and $H(C)$ and $H(L)$ are the entropy

$$H(C) = -\sum_i \frac{|c_i|}{N} \log \frac{|c_i|}{N}, \quad H(L) = -\sum_i \frac{|l_i|}{N} \log \frac{|l_i|}{N} \tag{2.39}$$

In the above formulas, $|c_i|$, $|l_j|$ and $|c_i \bigcap l_j|$ are the number of utterances from speaker $c_i$, $l_j$, and $c_i \bigcap l_j$, respectively.

The above two metrics are used for utterance-based evaluations. We extend them to frame-based evaluations by simply replacing the number of utterances in the above formulas with the corresponding number of frames. This allows us to investigate how the duration of an utterance affects the clustering performance.

### 2.6.3 Experiment Results

We present our experiment results in Tables 2.2-2.5. Specifically, we conduct speaker clustering (1) in the GMM mean supervector space with the Euclidean and cosine distance metrics, (2) in the PCA subspace with the Euclidean distance metric (i.e. the eigenvoice approach), (3) in the LPP subspace with the Euclidean distance metric, (4) in the NPE subspace with the Euclidean distance metric, (5) in the LDA subspace with the Euclidean distance metric (i.e. the fishervoice approach), (6) in the LSDA subspace with the cosine distance metric. In each experiment, we utilize both k-means (or spherical k-means) and agglomerative clustering. In order to compare our methods to the traditional bag of acoustic features methods, we employ the "Gaussian+BIC" method as the baseline. The experiment results are presented in four forms – utterance-based clustering accuracies (Table 2.2), utterance-based NMIs (Table 2.3), frame-based clustering accuracies (Table 2.4), and frame-based NMIs (Table 2.5). In the tables, Orig stands for the original GMM mean supervector space, *k* for k-means, and *h* for hierarchical clustering.

Additionally, we compare the proposed LDA transformed acoustic features with the acoustic features traditionally augmented with the first and second order derivatives. Specifically, 13 basic PLP features augmented by their first and second order derivatives form a 39-dimensional traditional acoustic feature vector. Table 2.6 gives a comparison of the results (clustering accuracies) of both kinds of acoustic features on speaker clustering under the same clustering conditions. In this table, "traditional" stands for traditional acoustic features, and "LDA" for the proposed LDA transformed acoustic features. These experiment results show that the LDA transformed acoustic features consistently outperform the traditional acoustic features by 1%-3%, which validates that the proposed speaker-discriminative acoustic feature transformation strategy can provide a better frontend to speaker clustering as compared to traditional ones. In the table, EU stands for Euclidean, COS for cosine, Trad for the traditional acoustic features, LDA for LDA transformed acoustic features, *k* for k-means, and *h* for hierarchical clustering.

Table 2.2: Performance comparison of speaker clustering based on utterance-based clustering accuracies.

| $\times$100% | | 2 spk $\sim$ 60 utt | | 5 spk $\sim$ 150 utt | | 10 spk $\sim$ 300 utt | | 20 spk $\sim$ 600 utt | | 50 spk $\sim$ 1500 utt | | 100 spk $\sim$ 3000 utt | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *k* | *h* | *k* | *h* | *k* | *h* | *k* | *h* | *k* | *h* | *k* | *h* |
| **EU** | Orig | 94.7 | 96.0 | 81.6 | 85.0 | 77.3 | 82.6 | 70.5 | 78.1 | 58.4 | 69.4 | 47.2 | 57.7 |
| | PCA | 96.6 | 96.2 | 84.8 | 85.5 | 81.3 | 82.9 | 78.5 | 79.3 | 69.7 | 69.9 | 59.4 | 58.5 |
| | LPP | 98.3 | 98.1 | 92.5 | 92.8 | 87.9 | 88.6 | 85.6 | 84.7 | 77.4 | 77.0 | 70.1 | 69.8 |
| | NPE | 97.4 | 97.0 | 84.3 | 85.0 | 83.2 | 83.9 | 78.7 | 79.3 | 70.4 | 69.8 | 58.1 | 58.3 |
| | LDA | 98.3 | 98.4 | 94.1 | 94.0 | 89.9 | 90.8 | 87.2 | 86.6 | 79.5 | 79.6 | 73.1 | 72.3 |
| **COS** | Orig | 99.0 | 99.1 | 88.3 | 90.7 | 84.1 | 86.5 | 80.6 | 82.2 | 74.7 | 77.7 | 66.4 | 69.3 |
| | **LSDA** | **99.2** | **99.1** | **97.8** | **98.0** | **95.0** | **94.7** | **90.3** | **90.0** | **84.3** | **85.9** | **77.9** | **79.4** |
| Baseline | | 82.5 | 83.8 | 71.6 | 72.0 | 58.3 | 60.5 | 53.1 | 52.7 | 43.2 | 44.1 | 35.0 | 37.4 |

Table 2.3: Performance comparison of speaker clustering based on utterance-based NMIs.

| $\times$100% | | 2 spk $\sim$ 60 utt | | 5 spk $\sim$ 150 utt | | 10 spk $\sim$ 300 utt | | 20 spk $\sim$ 600 utt | | 50 spk $\sim$ 1500 utt | | 100 spk $\sim$ 3000 utt | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *k* | *h* | *k* | *h* | *k* | *h* | *k* | *h* | *k* | *h* | *k* | *h* |
| **EU** | Orig | 91.9 | 93.7 | 79.0 | 82.7 | 74.4 | 80.3 | 67.7 | 75.3 | 55.9 | 66.4 | 44.3 | 55.3 |
| | PCA | 93.9 | 93.5 | 82.7 | 83.0 | 78.9 | 79.9 | 76.2 | 76.7 | 66.9 | 67.6 | 56.9 | 56.3 |
| | LPP | 96.1 | 95.4 | 89.9 | 89.9 | 85.1 | 86.2 | 83.1 | 82.0 | 74.8 | 74.0 | 67.2 | 67.3 |
| | NPE | 95.3 | 94.3 | 81.6 | 82.3 | 81.1 | 81.3 | 76.0 | 77.1 | 67.8 | 67.0 | 55.7 | 55.9 |
| | LDA | 96.2 | 95.8 | 92.0 | 91.5 | 87.5 | 88.2 | 85.0 | 84.0 | 77.4 | 77.6 | 71.1 | 70.2 |
| **COS** | Orig | 96.6 | 96.8 | 86.0 | 88.0 | 81.4 | 83.6 | 78.2 | 79.2 | 72.1 | 75.2 | 63.8 | 67.1 |
| | **LSDA** | **97.2** | **97.0** | **95.3** | **95.3** | **92.2** | **92.6** | **87.3** | **87.9** | **82.3** | **83.2** | **75.3** | **77.4** |
| Baseline | | 79.9 | 81.5 | 69.3 | 69.5 | 56.2 | 58.4 | 50.9 | 50.4 | 41.0 | 41.5 | 32.2 | 35.2 |

Table 2.4: Performance comparison of speaker clustering based on frame-based clustering accuracies.

| ×100% | | 2 spk ∼ 60 utt | | 5 spk ∼ 150 utt | | 10 spk ∼ 300 utt | | 20 spk ∼ 600 utt | | 50 spk ∼ 1500 utt | | 100 spk ∼ 3000 utt | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ |
| **EU** | Orig | 95.9 | 97.2 | 83.4 | 87.7 | 80.6 | 85.8 | 74.4 | 82.5 | 63.3 | 74.7 | 53.5 | 64.4 |
| | PCA | 97.4 | 97.7 | 86.2 | 86.6 | 84.4 | 85.7 | 81.9 | 83.1 | 74.5 | 74.8 | 66.1 | 67.2 |
| | LPP | 99.1 | 99.0 | 94.7 | 94.8 | 90.5 | 90.6 | 88.7 | 88.9 | 82.6 | 82.3 | 76.7 | 76.9 |
| | NPE | 98.8 | 98.5 | 87.1 | 87.7 | 86.1 | 86.7 | 82.5 | 83.6 | 74.9 | 74.7 | 64.5 | 64.7 |
| | LDA | 99.2 | 99.4 | 96.2 | 96.0 | 92.5 | 92.6 | 91.0 | 90.8 | 84.4 | 84.3 | 80.2 | 81.7 |
| **COS** | Orig | 99.4 | 99.5 | 89.9 | 92.5 | 87.3 | 88.1 | 85.1 | 86.5 | 79.3 | 81.2 | 72.3 | 74.9 |
| | **LSDA** | **99.7** | **99.5** | **98.9** | **99.2** | **97.1** | **96.8** | **92.5** | **92.1** | **87.0** | **88.2** | **82.6** | **83.2** |
| Baseline | | 84.7 | 85.1 | 72.9 | 73.4 | 61.6 | 63.2 | 57.7 | 56.9 | 48.6 | 49.4 | 42.7 | 44.1 |

Table 2.5: Performance comparison of speaker clustering based on frame-based NMIs.

| ×100% | | 2 spk ∼ 60 utt | | 5 spk ∼ 150 utt | | 10 spk ∼ 300 utt | | 20 spk ∼ 600 utt | | 50 spk ∼ 1500 utt | | 100 spk ∼ 3000 utt | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ |
| **EU** | Orig | 93.4 | 95.0 | 81.2 | 84.9 | 78.4 | 80.1 | 71.5 | 80.2 | 60.7 | 72.4 | 51.0 | 61.6 |
| | PCA | 95.0 | 95.6 | 83.5 | 84.5 | 81.9 | 82.7 | 78.9 | 80.1 | 72.2 | 72.1 | 63.4 | 65.1 |
| | LPP | 96.8 | 96.8 | 92.4 | 92.3 | 87.6 | 88.2 | 86.2 | 86.9 | 80.1 | 80.1 | 74.0 | 74.6 |
| | NPE | 96.2 | 95.7 | 84.7 | 85.0 | 83.5 | 84.6 | 80.2 | 80.8 | 72.0 | 72.5 | 61.8 | 62.0 |
| | LDA | 97.0 | 96.5 | 93.6 | 93.2 | 89.5 | 90.4 | 88.7 | 88.0 | 81.6 | 81.7 | 77.2 | 79.2 |
| **COS** | Orig | 97.1 | 97.0 | 87.8 | 89.9 | 84.7 | 85.8 | 82.2 | 84.4 | 76.8 | 78.7 | 69.5 | 72.6 |
| | **LSDA** | **97.4** | **97.3** | **96.1** | **96.5** | **94.9** | **94.5** | **89.9** | **89.4** | **84.5** | **86.2** | **79.8** | **81.0** |
| Baseline | | 82.5 | 82.8 | 70.2 | 71.3 | 59.4 | 60.7 | 55.4 | 54.1 | 46.3 | 47.0 | 40.2 | 41.7 |

### 2.6.4 Discussion

Our experiment results show that our speaker clustering methods based on the GMM mean supervector representation and vector-based distance metrics significantly outperform traditional speaker clustering methods based on the bag of acoustic features representation and likelihood-based distance metric such as the BIC. It is worth mentioning that in the Gaussian+BIC method, if we utilize agglomerative clustering, the computational load can become prohibitive as the number of speakers increases. This is because at each iteration, along with a new cluster being formed by merging the two closest ones, a new statistical model representing the new cluster has to be

Table 2.6: Performance comparison of the proposed LDA transformed acoustic features with the traditional acoustic features based on clustering accuracies.

| ×100% | | 2 spk ∼ 60 utt | | 5 spk ∼ 150 utt | | 10 spk ∼ 300 utt | | 20 spk ∼ 600 utt | | 50 spk ∼ 1500 utt | | 100 spk ∼ 3000 utt | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ | $k$ | $h$ |
| **EU** | Trad | 93.5 | 94.7 | 80.1 | 83.7 | 76.2 | 81.3 | 69.0 | 76.6 | 57.5 | 68.4 | 46.0 | 56.6 |
| | LDA | **94.7** | **96.0** | **81.6** | **85.0** | **77.3** | **82.6** | **70.5** | **78.1** | **58.4** | **69.4** | **47.2** | **57.7** |
| **COS** | Trad | 97.4 | 97.7 | 87.3 | 89.4 | 82.9 | 85.4 | 79.6 | 81.2 | 73.5 | 76.4 | 64.9 | 67.8 |
| | LDA | **99.0** | **99.1** | **88.3** | **90.7** | **84.1** | **86.5** | **80.6** | **82.2** | **74.7** | **77.7** | **66.4** | **69.3** |

re-trained, and the distance between the new model and any other model updated. On the contrary, agglomerative clustering can be done very efficiently in the GMM mean supervector space by using the "ward" linkage.

In the GMM mean supervector space, although speaker clustering based on the Euclidean distance metric achieves reasonably good results, the cosine distance metric consistently outperforms the Euclidean distance metric, thanks to the directional scattering property of the GMM mean supervectors.

Due to the difficulty of handling high-dimensional data, and in order to alleviate the "curse of dimensionality," linear subspace learning methods are used to derive various subspaces in which the final speaker clustering is performed. From the experiment results, we see that the unsupervised methods (i.e. PCA or the eigenface approach, LPP, NPE) more or less improve the performance, but significant improvements of the performance are achieved by supervised methods (i.e. LDA or the fishervoice approach, LSDA). Notably, our proposed LSDA algorithm leads to the state-of-the-art speaker clustering performance.

Also noted is that the frame-based performance is better than the utterance-based performance. The rationale behind this is that longer utterances tend to be correctly classified more than shorter utterances, which is reasonable because longer utterances can provide more speaker-discriminative information than shorter ones.

In every experiment, we employ two clustering algorithms, namely k-means and agglomerative clustering. Although there are pros and cons in each algorithm, we observe that in the same subspace, the speaker clustering performance of the two algorithms is comparable. However, k-means is sensitive to initialization, which means the result of a single run is not deterministic. Thus, we need to restart the k-means algorithm many times (e.g. 50) with a different initialization at each time, and record the best result. Therefore, k-means is normally much slower than agglomerative clustering. On the other hand, agglomerative clustering with the "ward" linkage method runs very fast. For the case of 100 speakers (about 3000 utterances), it takes our Matlab program less than one minute to complete the job on a Linux machine with a mainstream configuration.

Finally, the proposed discriminative acoustic feature transformation by itself can yield increased speaker clustering performance as compared to traditional acoustic features used for speaker clustering or diarization, as demonstrated by a separate experiment.

An issue that is not addressed in this chapter is the determination of the number of speakers. Automatically finding the number of clusters in a dataset in a completely unsupervised manner is still an open research problem. Many speaker diarization systems deal with this problem through hierarchical clustering using a BIC-based stopping criterion [39]. A similar method could have been used to determine the number of speakers automatically in this chapter. However, the exact number of speakers is hardly found by this simple method. In general, clustering results may vary dramatically for different numbers of speakers determined. In order to eliminate the influence of the number of speakers and single out the extent to which the proposed semi-supervised strategies may improve the speaker clustering performance, we assume that the number of test speakers is known a priori, and defer the investigation of this issue to our future work.

## 2.7 Summary

In this chapter, we propose the conceptually new idea of, and offer a complete treatment of, semi-supervised speaker clustering. By means of an independent training data set, our strategies are to encode the prior knowledge of speakers in general at the various stages of the speaker clustering pipeline via (1) learning a speaker-discriminative acoustic feature transformation, (2) learning a universal speaker prior model, and (3) learning a discriminative speaker subspace, or equivalently, a speaker-discriminative distance metric.

We discover the directional scattering property of the GMM mean supervector representation of utterances and advocate the use of the cosine distance metric instead of the Euclidean distance metric. We propose to perform discriminant analysis based on the cosine distance metric, leading to a novel distance metric learning algorithm – LSDA. We show that the proposed LSDA formulation can be systematically solved within the elegant graph embedding framework. Our speaker clustering experiments indicate that (1) our speaker clustering methods based on the GMM mean supervector representation and vector-based distance metrics outperform traditional methods based on the bag of acoustic features representation and likelihood-based distance metrics, (2) our advocated use of the cosine distance metric yields consistent increases in the speaker clustering performance as compared to the commonly used Euclidean distance metric, thanks to the directional scattering property of the GMM mean supervectors discovered, (3) our semi-supervised speaker clustering concept and strategies significantly improve the speaker clustering performance over the baselines, and (4) our proposed LSDA algorithm further leads to the state-of-the-art speaker clustering performance.

This chapter specifically treats the problem of speaker clustering, which is of great value by itself. However, it is seemingly stand-alone from the rest of the dissertation. We emphasize that this chapter serves as an important motivation of the study of one-vector representations of stochastic signals for pattern recognition, which is the main theme of the subsequent chapters in the dissertation. The use of the GMM mean supervector representation of utterances for speaker clustering in this chapter is very successful. The GMM mean supervector representation is a good example of one-vector representations of stochastic signals, whose essential concepts and many benefits are well illustrated within the application context of speaker clustering in this chapter.

# CHAPTER 3

# HIDDEN MARKOV MODELS

This chapter provides a somehow detailed review of the basics of hidden Markov models (HMMs) [26, 27, 28]. Since most of the techniques developed in this dissertation are for or based on HMMs, a fundamental review of this fascinating tool is necessary and inevitable. We begin with the basic concepts and definitions of HMMs, and proceed to the solutions to three fundamental problems of HMMs. We then introduce the concept and strategy of universal background modeling in the context of HMMs, and close this chapter with a description of two popular model adaptation techniques for HMMs, namely the maximum likelihood linear regression (MLLR) and maximum a posteriori (MAP) adaptation techniques [67].

## 3.1 Basic Concepts and Definitions

A hidden Markov model (HMM) is a doubly stochastic process which consists of an underlying, hidden, discrete random process possessing the Markov property (namely a Markov chain having a finite number of states) and an observed, discrete, continuous, or mixed discrete-continuous random process generated by a set of probabilistic functions of the underlying Markov chain, one of which is associated with each state of the Markov chain. At a discrete time instant, the HMM is assumed to be at a certain state, and an observation is generated by the probabilistic function associated with that particular state. The underlying Markov chain changes its state at every time instant according to some state transition probability distribution. In an HMM, it is only the observations that are seen by an observer, who does not have any direct knowledge of the underlying state sequence that generated these observations. HMMs have been proven to be flexible and natural probabilistic models for describing sequential data such as speech signals (and other time series), sequences of the DNA, sequences of characters in English sentences, and so forth. They are generative models which we can take advantage of to learn about and characterize the properties of many data sources without having (or having to create) the actual data sources. In the real world, HMMs have been successfully applied to a number of practical applications including automatic speech recognition [46, 68], automatic speech understanding [28], natural language processing [69], on-line handwriting recognition [70], etc., just to name a few.

From the state-space perspective, by definition an HMM consists of the following elements:

1. The number of discrete states in the model $N$
2. A set of discrete states $S = \{S_1, S_2, \cdots, S_N\}$
3. The number of discrete observation symbols $M$
4. A set of discrete observation symbols $V = \{v_1, v_2, \cdots, v_M\}$
5. The state transition probability distribution $A = [a_{ij}]_{N\times N}$

$$a_{ij} = P(q_{t+1} = S_j | q_t = S_i), \quad i = 1, 2, \cdots, N, j = 1, 2, \cdots, N \tag{3.1}$$

6. The state-dependent observation probability distribution $B = [b_j(k)]_{N\times M}$

$$b_j(k) = P(v_k \text{ at } t | q_t = S_j), \quad j = 1, 2, \cdots, N, k = 1, 2, \cdots, M \tag{3.2}$$

7. The initial state probability distribution $\Pi = [\pi_i]_{1\times N}$

$$\pi_i = P(q_1 = S_i), \quad i = 1, 2, \cdots, N \tag{3.3}$$

Note that the above definitions are given for a special kind of HMM, that is, those HMMs whose observations are drawn from a discrete finite alphabet according to the discrete observation probability distributions associated with the states of the underlying Markov chain. This kind of HMM is referred to as a discrete-observation HMM. There is another kind of HMM, namely the continuous-observation HMM, whose observations are continuous quantities drawn from a set of continuous state-dependent observation probability distributions. The state-dependent observation probability distributions can be either univariate or multivariate. The extension of the above definitions for continuous-observation HMMs is natural. In the case of continuous-observation HMMs, a state-dependent observation probability distribution $b_j(k)$ is specified by a continuous univariate or multivariate probability density function (PDF) [71], normally a finite mixture model [72], where $M$ in this case denotes the number of mixture components in the finite mixture model.

From the above definitions, an HMM is completely determined by its parameters $\lambda = \{A, B, \Pi\}$. Note that in the definitions of an HMM, the specification of the initial state probability distribution $\Pi$ is not mandatory. Instead of starting from the time instant $t = 1$ with a specified initial state probability distribution $\Pi$, we can start from the time instant $t = 0$ at a deterministic state (e.g. state 1). In this way, the initial state probability distribution $\Pi$ is absorbed into the state transition probability distribution $A$. The benefit of doing so is that it reduces the number of parameters of an HMM to be estimated and can potentially increase the robustness of HMM training.

## 3.2 Three Fundamental Problems of HMMs

Historically, there are three fundamental problems associated with HMMs [27]: scoring (or evaluation), structure identification, and learning (or training). These three fundamental problems of HMMs are critically important and must be solved before we can apply HMMs to practical applications in the real world. Here, we offer concise definitions of these three problems:

1. **Scoring**: Given an observation sequence $O = o_1 o_2 \cdots o_T$ and an HMM $\lambda$, the scoring problem aims to compute the probability of the observation sequence given the model $P(O|\lambda)$.

2. **Structure identification**: Given an observation sequence $O = o_1 o_2 \cdots o_T$ and an HMM $\lambda$, the structure identification problem aims to determine a state sequence $Q = q_1 q_2 \cdots q_T$ which is optimal in some meaningful sense.

3. **Leaning**: Given an observation sequence $O = o_1 o_2 \cdots o_T$, the learning problem aims to adjust the model parameters $\lambda$ to satisfy a certain

given criterion. For example, if the criterion is to maximize $P(O|\lambda)$, we end up with maximum likelihood learning for HMMs.

In the following subsections, we will discuss in detail how these three problems are tackled.

### 3.2.1 Problem 1: Scoring

The goal of scoring is to compute $P(O|\lambda)$. If we assume that $O$ is generated from a fixed state sequence $Q = q_1 q_2 \cdots q_T$, then

$$P(O|Q, \lambda) = b_{q_1}(o_1) b_{q_2}(o_2) \cdots b_{q_T}(o_T) \tag{3.4}$$

The probability of this state sequence, $Q$, is given by

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} \cdots a_{q_{T-1} q_T} \tag{3.5}$$

From the product rule of probability, we have

$$P(O, Q|\lambda) = P(O|Q, \lambda) P(Q|\lambda) \tag{3.6}$$

Marginalizing over all possible state sequences $Q$, we obtain

$$P(O|\lambda) = \sum_{Q} P(O, Q|\lambda) \tag{3.7}$$

At a first sight, we might think that this is the end of the story. However, this “brute-force” approach is computationally intractable. There is an efficient algorithm, called the forward-backward algorithm (or $\alpha$-$\beta$ algorithm), that can solve the same problem dramatically faster.

Let us define a “forward” variable, $\alpha_t(i)$, as the probability of the partial observation sequence up to time $t$ and state $S_i$ being occupied at time $t$

$$\alpha_t(i) = P(o_1 o_2 \cdots o_t, q_t = S_i|\lambda) \tag{3.8}$$

It is obvious that at time $t = 1$

$$\alpha_1(i) = P(o_1, q_1 = S_i|\lambda) = \pi_i b_i(o_1), \quad i = 1, 2, \cdots, N \tag{3.9}$$

For $t = 2, 3, \cdots, T$, by induction, we have

$$\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(o_t), \quad j = 1, 2, \cdots, N \tag{3.10}$$

We can then compute $P(O|\lambda)$ from the forward variable by

$$P(O|\lambda) = \sum_{i=1}^{N} P(o_1 o_2 \cdots o_T, q_T = S_i|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{3.11}$$

Compared to the "brute force" approach, which requires approximately $2TN^T$ arithmetic operations, the forward algorithm only requires $N^2T$ arithmetic operations, which implies a tremendous reduction in computational complexity.

Although the forward algorithm alone suffices for solving the scoring problem of HMMs, the backward algorithm is equally important. The reason for this will become obvious when we discuss the learning problem of HMMs.

Let us define a "backward" variable, $\beta_t(i)$, as the probability of the partial observation sequence from time $t+1$ to $T$ given that the state $S_i$ is occupied at time $t$

$$\beta_t(i) = P(o_{t+1} o_{t+2} \cdots o_T | q_t = S_i, \lambda) \tag{3.12}$$

We let

$$\beta_T(i) = 1, \quad i = 1, 2, \cdots, N \tag{3.13}$$

For $t = T-1, T-2, \cdots, 1$, by induction, we have

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \quad i = 1, 2, \cdots, N \tag{3.14}$$

As in the forward algorithm, the backward algorithm requires $N^2T$ arithmetic operations. Likewise, we can compute $P(O|\lambda)$ from the backward variable by

$$P(O|\lambda) = \sum_{i=1}^{N} P(o_1 o_2 \cdots o_T | q_1 = S_i, \lambda) P(q_1 = S_i|\lambda) = \sum_{i=1}^{N} \beta_1(i) \pi_i \tag{3.15}$$

In general, $P(O|\lambda)$ can be computed using both forward and backward variables. In the following derivation, without causing any ambiguity, we drop the notion of the model parameters $\lambda$ to avoid cluttering the formulas.

$$\begin{aligned} P(O|\lambda) &= \sum_{i=1}^{N}\sum_{j=1}^{N} P(o_1 o_2 \cdots o_T, q_t = S_i, q_{t+1} = S_j) \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \\ &= \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) \quad \text{for arbitrary } t : 1 \leq t \leq T \end{aligned} \tag{3.16}$$

It is worth mentioning that when we compute $P(O|\lambda)$ using the forward-backward algorithm, we marginalize over all the possible state sequences that could have generated the observation sequence $O$. This is distinct from a suboptimal alternative to computing $P$, which is to find the maximum $P$ among all the possible state sequences that could have produced $O$, namely

$$P = \max_{Q} P(O, Q|\lambda) \tag{3.17}$$

However, most often, $P \approx P(O|\lambda)$, because the majority of $P(O|\lambda)$ is attributed to the $P$ that is produced by the maximum likelihood state sequence. In the next subsection, we will discuss how we compute this maximum likelihood state sequence $Q$ as well as the associated probability $P$ which we can then use to approximate $P(O|\lambda)$.

### 3.2.2 Problem 2: Structure Identification

The goal of structure identification is to compute the "best" state sequence $Q$ of an HMM $\lambda$ that could have produced the observation sequence $O$ in some optimal sense. Usually, we are interested in finding the single best state sequence that maximizes $P(O, Q|\lambda)$, referred to as the maximum likelihood state sequence. The probability $P(O, Q^*|\lambda)$ associated with the maximum likelihood state sequence $Q^*$ is called the maximum likelihood observation probability. In practice, we may at times favor this maximum likelihood observation probability over the total observation probability computed by the forward-backward algorithm. This is not only because the maximum likelihood observation probability requires less computation than the total observation probability, but also because the maximum likelihood observation probability can better facilitate embedded training for HMMs (required by large vocabulary continuous speech recognition) than the total observation probability [67]. The solution to the structure identification problem is known as the Viterbi algorithm – a dynamic programming algorithm that finds the maximum likelihood or Viterbi path in a time-state trellis [73].

Let us define a variable $\phi_t(i)$ as the maximum probability of the partial observation sequence up to time $t$ along any path that ends at state $i$ at time $t$

$$\phi_t(i) = \max_{q_1 q_2 \cdots q_{t-1}} P(o_1 o_2 \cdots o_t, q_1 q_2 \cdots q_{t-1}, q_t = i) \tag{3.18}$$

We need to introduce an accompanying variable, $\psi_t(i)$, to keep track of the index of the state that transitions to state $i$ at time $t$ to facilitate a subsequent back-tracking step.

The Viterbi algorithm proceeds as follows:

1. Initialization

$$\phi_1(i) = \pi_i b_i(o_1), \quad i = 1, 2, \cdots, N \tag{3.19}$$

$$\psi_1(i) = 0, \quad i = 1, 2, \cdots, N \tag{3.20}$$

2. Reclusion

$$\begin{aligned}\phi_t(j) &= \max_{1 \le i \le N} [\phi_{t-1}(i) a_{ij}]\, b_j(o_t) \\ &\quad j = 1, 2, \cdots, N, t = 2, 3, \cdots, T\end{aligned} \tag{3.21}$$

$$\begin{aligned}\psi_t(j) &= arg \max_{1 \le i \le N} [\phi_{t-1}(i) a_{ij}] \\ &\quad j = 1, 2, \cdots, N, t = 2, 3, \cdots, T\end{aligned} \tag{3.22}$$

3. Termination

$$P = \max_{1 \le i \le N} \phi_T(i) \tag{3.23}$$

$$q_T = arg \max_{1 \le i \le N} \phi_T(i) \tag{3.24}$$

4. Back-tracking

$$q_t = \psi_{t+1}(q_{t+1}), \quad t = T-1, T-2, \cdots, 1 \tag{3.25}$$

As one can see, the Viterbi algorithm efficiently computes the maximum likelihood observation probability $P$ and the maximum likelihood state sequence $Q = \{q_1 q_2 \cdots q_T\}$ with a computational complexity of $N^2T$ arithmetic operations.

An alternative but equivalent way of implementing the Viterbi algorithm is to work in the log domain. Let

$$\begin{aligned} \tilde{\pi}_i &= \log \pi & (3.26) \\ \tilde{b}_j(o_t) &= \log b_j(o_t) & (3.27) \\ \tilde{a}_{ij} &= \log a_{ij} & (3.28) \end{aligned}$$

The Viterbi algorithm then proceeds as follows:

1. Initialization

$$\begin{aligned} \phi_1(i) &= \tilde{\pi}_i + \tilde{b}_i(o_1), \quad i = 1, 2, \cdots, N & (3.29) \\ \psi_1(i) &= 0, \quad i = 1, 2, \cdots, N & (3.30) \end{aligned}$$

2. Reclusion

$$\begin{aligned} \phi_t(j) &= \max_{1 \le i \le N} [\phi_{t-1}(i) + \tilde{a}_{ij}] + \tilde{b}_j(o_t) & (3.31) \\ & \quad j = 1, 2, \cdots, N, t = 2, 3, \cdots, T & \\ \psi_t(j) &= arg \max_{1 \le i \le N} [\phi_{t-1}(i) + \tilde{a}_{ij}] & (3.32) \\ & \quad j = 1, 2, \cdots, N, t = 2, 3, \cdots, T & \end{aligned}$$

3. Termination

$$\begin{aligned} P &= \max_{1 \le i \le N} \phi_T(i) & (3.33) \\ q_T &= arg \max_{1 \le i \le N} \phi_T(i) & (3.34) \end{aligned}$$

4. Back-tracking

$$q_t = \psi_{t+1}(q_{t+1}), \quad t = T-1, T-2, \cdots, 1 \qquad (3.35)$$

Working in the log domain avoids numerical problems such as "underflows." As we will see in the next subsection, numerical problems occur during the HMM learning process, which has to be resolved by a scaling mechanism. By working in the log domain, the Viterbi algorithm avoids the numerical problems and thus does not need such a scaling mechanism.

### 3.2.3 Problem 3: Learning

We finally arrive at the hardest problem of the three – the learning problem. The learning problem aims to estimate the parameters of an HMM, $\lambda$, given an observation sequence, $O$, such that a certain criterion is satisfied. Although there are a variety of criteria that can be employed, we typically perform maximum likelihood estimation of the model parameters. The famous classic algorithm that performs maximum likelihood estimation of the parameters of an HMM is known as the Baum-Welch algorithm [74]. As we will discuss later in this subsection, the Baum-Welch algorithm is in fact a particular form of the more general expectation-maximization (EM) algorithm [50] widely recognized in the machine learning literature.

Just like the EM algorithm, the Baum-Welch algorithm is an iterative re-estimation procedure. The basic idea is that, given the current parameter estimate at the $l^{th}$ iteration, $\lambda(l)$, we compute the expected values of certain model events and, based on these computed expected values, obtain a refined estimate of the model parameters, $\lambda(l+1)$, at the $(l+1)^{th}$ iteration. This procedure continues iteratively, each step producing a refined estimate of the model parameters over the previous estimate, until convergence is achieved. The Baum-Welch algorithm is guaranteed to converge to a local maximum of the log likelihood function $\log P(O|\lambda)$. However, since the log likelihood function is highly non-linear and non-convex, there is no guarantee that the algorithm converges to the global maximum. Therefore, the result of parameter estimation by the Baum-Welch algorithm is not deterministic, which means, different initializations can possibly lead to different final parameter estimates, each of which corresponds to a different local maximum of the log likelihood function. In practice, the Baum-Welch algorithm works reasonably well given the proper initialization of the model parameters. It may be worthwhile to note that in most situations it is not necessary, nor useful, for the algorithm to converge to the global maximum of the log likelihood function, because the global maximum is sometimes related to a severe problem known as over-fitting [3, 1] in the pattern recognition literature and it is for sure that we want to prevent this from happening. For example, in the case where there is little training data available, if the model is flexible enough (i.e. having a large number of free parameters), we will most likely find the global maximum of the log likelihood function. In this case, however, we should definitely avoid the global maximum because by finding the global maximum we are actually over-fitting the model to the data. As the data is noisy (that is why we want to describe it using a stochastic model such as an HMM rather than a deterministic model), we are likely to have fitted the model to the noise present in the data rather than to have captured the underlying regularities of the data itself. We should be aware that over-fitting is an inherent problem of maximum likelihood estimation techniques that aries naturally when there is little training data available, as maximum likelihood estimation techniques are aimed at finding the model that can "best" explain the data without assuming any form of prior knowledge about the regularities of the data. When there is insufficient training data available, maximum likelihood estimation techniques will most likely find "too best" a model, in the sense that the model explains everything present in the data, including the noise, which is highly undesired.

Now let us return to the discussion of the Baum-Welch algorithm for learning the HMM parameters. We define a variable, $\gamma_t(i,j)$, to be the probability of being in state $S_i$ at time $t$ and in state $S_j$ at time $t+1$ given the observation sequence $O$, and define a variable, $\gamma_t(i)$, to be the probability of being in state $S_i$ at time $t$ given $O$, namely

$$\begin{aligned} \gamma_t(i,j) &= P(q_t = S_i, q_{t+1} = S_j | O, \lambda) & (3.36) \\ & \quad i,j = 1,2,\cdots,N, t = 1,2,\cdots,T \\ \gamma_t(i) &= P(q_t = S_i | O, \lambda) & (3.37) \\ & \quad i = 1,2,\cdots,N, t = 1,2,\cdots,T \end{aligned}$$

Obviously,

$$\gamma_t(i) = \sum_{j=1}^{N} \gamma_t(i,j) \tag{3.38}$$

We further let

$$\gamma_{ij} = \sum_{t=1}^{T-1} \gamma_t(i,j) \tag{3.39}$$

$$\gamma_{i} = \sum_{t=1}^{T-1} \gamma_t(i) \tag{3.40}$$

Both $\gamma_{ij}$ and $\gamma_i$ have specific physical meanings that are related to a model event counting mechanism. That is, $\gamma_{ij}$ refers to the expected number of transitions from state $S_i$ to state $S_j$ given the HMM $\lambda$ and observation sequence $O$, and $\gamma_i$ refers to the expected number of transitions from state $S_i$ (regardless of whichever states the transitions are to), conditioned on $\lambda$ and $O$. Through this model event counting mechanism, the Baum-Welch re-estimation formulas may be intuitively obtained from the following:

$$\begin{aligned} \hat{\pi}_i &= \text{Expected number of times in state } S_i \text{ at time } t = 1 \\ &= \gamma_1(i) \end{aligned} \tag{3.41}$$

$$\begin{aligned} \hat{a}_{ij} &= \frac{\text{Expected number of transitions from state } S_i \text{ to } S_j}{\text{Expected number of transitions from state } S_i} \\ &= \frac{\gamma_{ij}}{\gamma_i} \end{aligned} \tag{3.42}$$

$$\begin{aligned} \hat{b}_j(v_k) &= \frac{\text{Expected number of times in state } S_j \text{ while emitting } v_k}{\text{Expected number of times in state } S_j} \\ &= \frac{\sum_{t:o_t=v_k} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(i)} \end{aligned} \tag{3.43}$$

Formally, the Baum-Welch algorithm for learning the parameters of an HMM is stated as follows:

1. Initialization: The initial parameters of the model, $\lambda_0$, must be provided, either randomly or a by smarter initialization scheme.

2. Re-estimation: The iterative Baum-Welch re-estimation formulas are

derived as follows:

$$\begin{aligned}
\hat{\pi}_i &= P(q_1 = S_i|O,\lambda) \\
&= \frac{P(O, q_1 = S_i|\lambda)}{P(O|\lambda)} \\
&= \frac{\alpha_1(i)\beta_1(i)}{\sum_{i=1}^{N} \alpha_1(i)\beta_1(i)} \qquad (3.44)
\end{aligned}$$

$$\begin{aligned}
\hat{a}_{ij} &= \frac{\sum_{t=1}^{T-1} P(q_t = S_i, q_{t+1} = S_j|O,\lambda)}{\sum_{t=1}^{T-1} P(q_t = S_i|O,\lambda)} \\
&= \frac{\sum_{t=1}^{T-1} P(O, q_t = S_i, q_{t+1} = S_j|\lambda)}{\sum_{t=1}^{T-1} P(O, q_t = S_i|\lambda)} \\
&= \frac{\sum_{t=1}^{T-1} \alpha_t(i)a_{ij}b_j(o_{t+1})\beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i)\beta_t(i)} \qquad (3.45)
\end{aligned}$$

$$\begin{aligned}
\hat{b}_j(v_k) &= \frac{\sum_{t:o_t=v_k} P(q_t = S_j|O,\lambda)}{\sum_{t=1}^{T} P(q_t = S_j|O,\lambda)} \\
&= \frac{\sum_{t:o_t=v_k} P(O, q_t = S_j|\lambda)}{\sum_{t=1}^{T} P(O, q_t = S_j|\lambda)} \\
&= \frac{\sum_{t:o_t=v_k} \alpha_t(j)\beta_t(j)}{\sum_{t=1}^{T} \alpha_t(j)\beta_t(j)} \qquad (3.46)
\end{aligned}$$

3. Convergence test: The Baum-Welch re-estimation formulas are applied iteratively, each step increasing the value of the log likelihood function by a certain amount. It is intuitive to employ such a convergence criterion: if the value of the log likelihood function no longer increases, or the value increase is smaller than a threshold, then we claim that convergence is achieved. This could be problematic, however. Since the log likelihood function is highly non-linear and non-convex, it is very likely that at some iterations, the value of the log likelihood function remains about the same yet a local maximum is not reached. We call such places in the domain of the log likelihood function the saddle points. Preferably, we expect that the saddle points should be bypassed. Therefore, a better convergence criterion is to test whether the new estimated parameters are close enough to the previous estimated parameters. That is, if the Euclidean distance between the parameter vector at iteration $l$ and the parameter vector at iteration $l-1$ is

smaller than a threshold, then we can safely say that convergence is achieved.

Note that the Baum-Welch re-estimation formulas can also be directly derived from the EM framework, which will not be covered in detail here. In addition, in the above derivations, we mainly focus on learning the parameters of a discrete-observation HMM with the state-dependent observation probability distributions described by (table-form) probability mass functions (PMFs) [75]. In a continuous-observation HMM, the state-dependent observation probability distributions are described by (function-form) PDFs. While a continuous-observation HMM may be equipped with various forms of state-dependent observation PDFs, the Gaussian PDFs and Gaussian mixture PDFs [15] are the two most well-known forms in practice. For a continuous-observation HMM, the Baum-Welch re-estimation formulas should need to be modified accordingly.

In a Gaussian mixture continuous-observation HMM, a state-dependent observation probability distribution is given by a Gaussian mixture PDF of the form

$$\begin{aligned} b_j(o_t) &= \sum_{k=1}^{M} c_{jk}\mathcal{N}(o_t|\mu_{jk},\Sigma_{jk}) \\ & \quad 1 \le j \le N, 1 \le k \le M \end{aligned} \tag{3.47}$$

where $M$ is the number of Gaussian components, $c_{jk}$ is the $k^{th}$ mixture weight, and $N(o_t|\mu_{jk},\Sigma_{jk})$ is a multivariate Gaussian PDF with mean vector $\mu_{jk}$ and covariance matrix $\Sigma_{jk}$

$$\mathcal{N}(o_t|\mu_{jk},\Sigma_{jk}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_{jk}|^{\frac{1}{2}}} e^{-\frac{1}{2}(o_t-\mu_{jk})^T\Sigma_{jk}^{-1}(o_t-\mu_{jk})} \tag{3.48}$$

The Baum-Welch re-estimation formulas for the Gaussian mixture state-dependent observation probability distributions are given as follows:

$$\hat{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{3.49}$$

$$\hat{\mu}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) o_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{3.50}$$

$$\hat{\Sigma}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)(o_t - \hat{\mu}_{jk})(o_t - \hat{\mu}_{jk})^T}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{3.51}$$

where $1 \le i, j \le N, 1 \le k \le M$, and

$$\gamma_t(j,k) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \frac{c_{jk}\mathcal{N}(o_t|\mu_{jk}, \Sigma_{jk})}{\sum_{m=1}^{M} c_{jm}\mathcal{N}(o_t|\mu_{jm}, \Sigma_{jm})} \tag{3.52}$$

Note that for Gaussian mixture continuous-observation HMMs, the Baum-Welch re-estimation formulas for the initial state probability distribution and state transition probability distribution remain the same as in case of discrete-observation HMMs.

In the solutions to the three fundamental problems of HMMs presented above, there are several practical implementation issues that must be addressed in order to write a working computer program for the solutions. These issues include scaling, multiple observation sequences, initial parameter estimates, missing data, choice of model size and type, etc. For a detailed prescription of these issues, please refer to the widely read and now classic papers [26, 27].

One last note in this section is on state duration modeling in HMMs. Perhaps the major weakness of conventional HMMs is that they lack the capability of modeling state durations in the desirable way, as the inherent distribution of state durations in an HMM is implicitly exponential. For most real-world stochastic signals, the exponential state duration distribution is inappropriate. Therefore, values have been added to the theory of HMMs by the inclusion of explicit state duration distributions of some analytic forms [76, 77].

## 3.3 Universal Background Modeling

The Baum-Welch algorithm presented above is a maximum likelihood learning technique for learning the model parameters of an HMM given a set of training observation sequences. Given a sufficient amount of training data, the Baum-Welch algorithm can be used to learn high-quality HMMs. Together with other techniques such as scoring and decoding, the HMMs learned by the Baum-Welch algorithm can produce high-performance results in a variety of real-world applications such as large vocabulary automatic speech recognition. However, in situations where there is insufficient training data available for learning an HMM, the Baum-Welch algorithm is likely to lead to a poorly estimated model. This is known as the over-fitting problem, which is a general property of maximum likelihood estimation techniques. To overcome this difficulty, we introduce universal background modeling. Universal background modeling was originally proposed for biometric verification systems which use a universal background model (UBM) to represent the general and person-independent feature characteristics to be compared against a model of person-specific feature characteristics when making an accept or reject decision. For example, in a speaker verification system, the UBM is a speaker-independent Gaussian mixture model (GMM) trained with speech samples that come from a large set of speakers, which is used when training the speaker-specific model by acting as the prior model in MAP parameter estimation [78]. Under the context of HMMs, the UBM is obtained as follows. We first pool the separate training data for learning the individual HMMs together to form an aggregated training data set. This aggregated training data set is normally large enough that we can assume that it is likely to capture sufficient variations in the population of the data. We then use the Baum-Welch algorithm to learn a single global HMM based on the aggregated training data set. We call this single global HMM learned on the aggregated training data set the UBM, as this single global HMM is supposed to represent the probability distribution of the observations drawn from the population of the data. The UBM is in general a well-trained and robust model, from which we can derive particular individual HMMs specific to small amounts of training data using one of the available model adaptation techniques for HMMs, as described in the next section. An HMM that is adapted from the well-trained UBM with a small amount of training data is proven to be far more robust than an HMM that is learned directly with the same small amount of training data using the Baum-Welch algorithm.

A deeper understanding of universal background modeling may be acquired by connecting it to statistical classification problems. In the Bayesian or minimum error classification rule [3, 1], an optimal decision boundary is formed by the posterior probability distributions of two classes, which may be computed from the class-dependent likelihood probability distributions of the two classes, respectively. There may be a lot of fine structures in the class-dependent likelihood probability distribution of either class, and normally we require a lot of training data to learn these fine structures. However, as far as the classification problem is concerned, only the regions of the class-dependent likelihood probability distributions near the decision boundary are important. The fine structures of the class-dependent likelihood probability distributions which are away from the decision boundary are of no use. Therefore, it is a waste of the precious training data to try to learn these fine structures all over, and more disastrously, the fine structures (both near and away from the decision boundary) will never be properly learned if the available training data is insufficient. The introduction of universal background modeling allows us to learn the fine structures irrelevant to the classification problem using a large aggregated training data set, and focus on the regions near the decision boundary by learning the fine structures within these regions using small amounts of training data.

The concept of universal background modeling is related to the more elegant Bayesian learning theory [1]. In Bayesian learning, a prior probability distribution is imposed on the model parameters, which will be adjusted as more and more evidence is present. The UBM may be considered as a prior model corresponding to the prior probability distribution of the model parameters. Bayesian learning is a powerful learning paradigm which has many advantages over maximum likelihood learning. However, it is computationally very expensive. Universal background modeling serves as a good trade-off point between full Bayesian learning and maximum likelihood learning.

## 3.4 Adaptation Techniques for HMMs

Once we have a well-trained UBM that has been learned from an aggregated training data set, we can derive the individual HMMs from small amounts of training data using one of the model adaptation techniques for HMMs. In the literature, there are two popular model adaptation techniques for HMMs, namely the maximum likelihood linear regression (MLLR) and maximum a posteriori (MAP) adaptation techniques [67].

### 3.4.1 MLLR

The maximum likelihood linear regression or MLLR adaptation technique computes a set of transformations that will reduce the mismatch between the UBM and a small amount of available training or adaptation data. More specifically, MLLR estimates a set of linear transformations for the mean and variance parameters of a Gaussian mixture HMM. The effect of these transformations is to alter the mean and variance parameters of the HMM so that it is more likely to generate the adaptation data.

The linear transformation for the mean parameters is given by

$$\hat{\mu} = W\xi \tag{3.53}$$

where $W$ is the $d \times (d+1)$ transformation matrix to be estimated, with $d$ being the dimension of the observation vectors, and $\xi$ is the $(d+1) \times 1$ extended mean vector $\xi = [1 \quad \mu^T]^T$. We may decompose $W$ into $W = [b \quad A]$. In this case, $A$ represents a $d \times d$ transformation matrix and $b$ represents a bias vector.

The linear transformation for the variance parameters is given by

$$\hat{\Sigma} = B^T H B \tag{3.54}$$

where $H$ is a $d \times d$ transformation matrix to be estimated, and $B$ is the inverse of the Choleski factor of $\Sigma^{-1}$, namely

$$B = C^{-1}, \quad \Sigma^{-1} = CC^T \tag{3.55}$$

An alternative and more efficient form of the linear transformation for the variance parameters is given by

$$\hat{\Sigma} = H \Sigma H^T \tag{3.56}$$

where $H$ is the $d \times d$ transformation matrix to be estimated. This form of the linear transformation for the variance parameters allows it to be efficiently implemented as a transformation of the means and observation vectors, as follows:

$$\mathcal{N}(o|\mu, H\Sigma H) = \frac{1}{|H|}\mathcal{N}(H^{-1}o|H^{-1}\mu, \Sigma) = |A|\mathcal{N}(Ao|A\mu, \Sigma) \tag{3.57}$$

where $A = H^{-1}$.

In MLLR, the transform matrices in the above equations are obtained by solving an optimization problem using the EM algorithm with a standard auxiliary function. For the details of how to derive the formulas for MLLR using the EM algorithm, please refer to [67].

### 3.4.2 MAP

The maximum a posteriori or MAP adaptation technique, sometimes referred to as the Bayesian adaptation technique for reasons mentioned earlier, involves the use of the prior knowledge about the distribution of the model parameters. The rationale behind MAP is that if we know what the model parameters are likely to be, then we have a better chance to learn a decent model from a small amount of training data. For the MAP adaptation purpose, the prior knowledge about the distribution of the model parameters is encoded in the parameters of the UBM.

In the Baum-Welch algorithm described above, when we re-estimate the parameters of the Gaussian mixture state observation PDFs, $\hat{c}_{jk}, \hat{\mu}_{jk}, \hat{\Sigma}_{jk}, 1 \leq j \leq N, 1 \leq k \leq M$, instead of starting with randomly initialized parameters, we start with a UBM with parameters $\bar{c}_{jk}, \bar{\mu}_{jk}, \bar{\Sigma}_{jk}, 1 \leq j \leq N, 1 \leq k \leq M$. In the following, we will drop the index ranges to avoid cluttering the equations by assuming that $1 \leq j \leq N, 1 \leq k \leq M, 1 \leq t \leq T$. For each observation vector $o_t$, we compute the posterior probability of $o_t$ being generated by state $S_j$ and Gaussian component $k$ of the UBM

$$\gamma_t(j,k) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \frac{\bar{c}_{jk}N(o_t|\bar{\mu}_{jk},\bar{\Sigma}_{jk})}{\sum_{m=1}^{M}\bar{c}_{jm}N(o_t|\bar{\mu}_{jm},\bar{\Sigma}_{jm})} \tag{3.58}$$

Based on these posterior probabilities, we compute the data sufficient statistics

$$n(j,k) = \sum_{t=1}^{T}\gamma_t(j,k) \tag{3.59}$$

$$\mu(j,k) = \frac{1}{n(j,k)}\sum_{t=1}^{T}\gamma_t(j,k)o_t \tag{3.60}$$

$$S(j,k) = \frac{1}{n(j,k)}\sum_{t=1}^{T}\gamma_t(j,k)o_t o_t^T \tag{3.61}$$

where $n(j,k)$ can be interpreted as the (fractional) number of observation vectors for which the state $S_j$ and Gaussian component $k$ of the UBM are responsible. Notice that the model sufficient statistics given by the UBM are

$$\tilde{n}(j,k) = T\bar{c}_{jk} \tag{3.62}$$

$$\tilde{\mu}(j,k) = \bar{\mu}_{jk} \tag{3.63}$$

$$\tilde{S}(j,k) = \bar{\Sigma}_{jk} + \bar{\mu}_{jk}\bar{\mu}_{jk}^T \tag{3.64}$$

The MAP adaptation technique generates a new set of sufficient statistics by interpolating the data and model sufficient statistics, namely

$$\hat{n}(j,k) = \rho^{(1)}n(j,k) + (1-\rho^{(1)})\tilde{n}(j,k) \tag{3.65}$$

$$\hat{\mu}(j,k) = \rho^{(2)}\mu(j,k) + (1-\rho^{(2)})\tilde{\mu}(j,k) \tag{3.66}$$

$$\hat{S}(j,k) = \rho^{(3)}S(j,k) + (1-\rho^{(3)})\tilde{S}(j,k) \tag{3.67}$$

where the interpolation coefficients, $\rho^{(1)}, \rho^{(2)}, \rho^{(3)}$, are smartly adaptive to the amount of available training data according to the following empirical formula:

$$\rho^{(l)} = \frac{n(j,k)}{n(j,k)+r^{(l)}}, \quad l = 1,2,3 \tag{3.68}$$

with $r^{(l)}$ being a tunable constant specified by the user.

The new set of sufficient statistics is now used for re-estimating the model parameters

$$\hat{c}_{jk} = \hat{n}(j,k)/T \tag{3.69}$$

$$\hat{\mu}_{jk} = \hat{\mu}(j,k) \tag{3.70}$$

$$\hat{\Sigma}_{jk} = \hat{S}(j,k) - \hat{\mu}_{jk}\hat{\mu}_{jk}^T \tag{3.71}$$

In the MAP adaptation algorithm, the re-estimation formulas for the Gaussian mixture state observation PDFs are replaced by the above MAP adaptation formulas. Note that in the second iteration of the algorithm, the newly adapted Gaussian mixture state observation PDFs will replace the UBM in the re-estimation formulas. From then on, the estimated Gaussian mixture observation PDFs at a current iteration will serve as a prior model for the next iteration. The MAP adaptation of the state initial probability distribution and state transition probability distribution may be likewise derived [67].

It is worth mentioning that in Equation (3.68), when the number of observation vectors for which state $S_j$ and Gaussian component $k$ are responsible is small, i.e., $n(j,k) \approx 0$, $\rho^{(l)}$ approaches $1/r^{(l)}$. In this case, the model sufficient statistics will dominate the new sufficient statistics, and hence the new model parameters will remain close to those of the prior model (e.g. the UBM). When the number of observation vectors for which state $S_j$ and Gaussian component $k$ are responsible is large, i.e., $n(j,k) \to \infty$, $\rho^{(l)}$ approaches 1. In this case, the model sufficient statistics will vanish from the interpolation formulas, and the new sufficient statistics consist of only the data sufficient statistics. This is exactly the case of the original Baum-Welch algorithm.

One drawback of the MAP adaptation technique as compared to the MLLR adaptation technique is that MAP requires more adaptation data than MLLR in order to yield a good estimate. When the amount of adaptation data is extremely small, MLLR outperforms MAP. However, as the amount of adaptation data increases, MAP begins to perform better than MLLR. Therefore, if we can make good use of the complementary advantages and disadvantages of MAP and MLLR, both adaptation techniques may be combined in a smart way to further improve the performance of the adaptation for HMMs. One practical way of combining the two adaptation techniques is to first use MLLR to adapt the UBM with the available adaptation data, and the model generated by MLLR is then used as the prior model for MAP adaptation with the same adaptation data. In this way, the Gaussian components of the state-dependent observation probability distributions that have a low occupation likelihood in the adaptation data (which hence would not change much using MAP alone) have been adapted by MLLR [67].

# CHAPTER 4

# BOOSTING LEARNING FOR HMMS

As reviewed in Chapter 3, the traditional Baum-Welch or EM algorithm for learning the model parameters of an HMM is a maximum likelihood estimation technique which suffers from a serious problem known as the over-fitting problem [3, 1], especially when there is insufficient training data available. Over-fitting is regarded as a general and inherent property of maximum likelihood estimation techniques, where a solution is obtained by optimizing an objective function in a model parameter space. In this chapter, we propose a new maximum likelihood learning algorithm for HMMs, which we call the boosting Baum-Welch algorithm for reasons that will become obvious later in the chapter. In the proposed boosting Baum-Welch algorithm, we formulate the HMM learning problem as an incremental optimization procedure which performs a sequential gradient descent search on a loss functional for a good fit in an inner product function space. The boosting Baum-Welch algorithm can serve as an alternative maximum likelihood learning algorithm for HMMs to the traditional Baum-Welch or EM algorithm, and a preferred method for use in situations where there is insufficient training data available. The reason that the boosting Baum-Welch algorithm – being itself a maximum likelihood estimation technique – is less susceptible to the over-fitting problem than the traditional Baum-Welch or EM algorithm is that the boosting Baum-Welch algorithm tends to produce a "large margin" effect. Our experiments show that the boosting Baum-Welch algorithm can lead to a significant performance increase in an HMM-based speech emotion classification task in the case of the small-size training data sets.

## 4.1 Boosting Probability Density Estimation

In recent years, ensemble methods such as boosting [79] have gained considerable interest in the pattern recognition and machine learning community. Boosting is essentially a "voting" method which incrementally builds a linear combination of a set of "weak" learners to generate a single "strong" predictive model. The method sequentially fits weak learners to weighted versions of the training samples, where the weights at the current step are determined according to the performance of the model built by the previous step to give more emphasis to those training samples which are difficult to predict. The success of boosting is attributed to the "boosting" effect that the linear combination of weak learners achieves as compared to the performance of the individual weak learners, thanks to a nice property of boosting that was discovered several years after the method was invented, namely the separating or margin maximizing property [80].

Given a data set $\{z_i\}_{i=1}^n$, a space of weak learners $\mathcal{H}$, and a loss function $L$, the objective of boosting is to sequentially find a set of weak learners $h_1, h_2, \cdots, h_T \in \mathcal{H}$ and a set of corresponding constants $\alpha_1, \alpha_2, \cdots, \alpha_T \in \mathcal{R}$ which minimize a total loss on the data set $\sum_{i=1}^n L\left(z_i, \sum_{t=1}^T \alpha_t h_t(z_i)\right)$. For example, in the famous AdaBoost algorithm [7], which is a boosting algorithm originally proposed for the binary classification task, $z_i = (x_i, y_i)$, $y_i \in \{-1, +1\}$, and $L = L\left(y_i, \sum_{t=1}^T \alpha_t h_t(x_i)\right)$.

The theoretical study of Mason et al. [81] shows that AdaBoost can be described as a gradient descent search algorithm, where the weights at each step of the algorithm correspond to the gradient of the following exponential loss function at the "current" fit:

$$L = \sum_{i=1}^{n} \exp\left(-y_i \sum_{t=1}^{T} \alpha_t h_t(x_i)\right) \tag{4.1}$$

Note that the loss function in Equation (4.1) implies the tendency of AdaBoost to produce "large margin" classifiers.

This view of boosting as a gradient descent search for a good fit in a function space has motivated a general boosting framework for the unsupervised learning problem of probability density estimation [82]. Let $\mathcal{F} = \text{lin}\{\mathcal{H}\}$ be a function space of all linear combinations of weak learners in $\mathcal{H}$, and $L : \mathcal{F} \to \mathcal{R}$ a loss functional on $\mathcal{F}$. The goal of a boosting algorithm is to find a function $F \in \mathcal{F}$ such that $L(F)$ is minimized. This goal may be achieved iteratively via a gradient descent procedure, described as follows.

Assume that at step $t$ of a boosting algorithm, the model built so far is

$$F_{t-1}(z) = \sum_{j<t} \alpha_j h_j(z) \tag{4.2}$$

We now seek a weak learner from $\mathcal{H}$ and add it to the current model with a small positive coefficient $\epsilon$

$$F_t(z) = F_{t-1}(z) + \epsilon h(z) \tag{4.3}$$

Naturally, we want the loss functional on $F_t(z)$ to decrease. Viewed in function space terms, we look for the "direction" in the function space such that $L(F_t)$ decreases the most rapidly. Without any constraints, the desired direction would be simply the negative gradient of the loss functional on the current model

$$h = -\nabla L(F_{t-1}) \tag{4.4}$$

However, the choice of an $h$ is restricted to be from the space $\mathcal{H}$. Therefore, we instead search for an $h$ at step $t$ to maximize the inner product between $h$ and the negative gradient of the loss functional on the current model

$$h_t = \arg\max_{h \in \mathcal{H}} \langle -\nabla L(F_{t-1}), h \rangle \tag{4.5}$$

The validity of Equation (4.5) may be seen from a first-order Taylor approximation of the loss functional (if $\epsilon$ is small enough)

$$L(F_{t-1} + \epsilon h) \approx L(F_{t-1}) + \epsilon \langle \nabla L(F_{t-1}), h \rangle \tag{4.6}$$

Equation (4.6) indicates that in order to decrease the loss functional on the augmented model, we need to minimize the term $\langle \nabla L(F_{t-1}), h \rangle$. This leads to Equation (4.5).

For the problem of probability density estimation, we adopt a maximum likelihood criterion. Let the loss functional be the negative log likelihood of the data set

$$L(F_t) = -\sum_{i=1}^{n} \log\left(F_t(z_i)\right) \tag{4.7}$$

We have

$$\langle -\nabla L(F_{t-1}), h \rangle = \sum_{i=1}^{n} \frac{1}{F_{t-1}(z_i)} h(z_i) \tag{4.8}$$

Thus, for the problem of probability density estimation, a weak learner is chosen at time $t$ according to the following criterion:

$$h_t = \arg\max_{h \in \mathcal{H}} \sum_{i=1}^{n} \frac{1}{F_{t-1}(z_i)} h(z_i) \tag{4.9}$$

One might have noticed that the model augmentation scheme given by Equation (4.3) does not guarantee a valid probability density function (PDF). To ensure the validity of a PDF at all steps, we use the following model augmentation scheme instead:

$$F_t(z) = (1 - \alpha) F_{t-1}(z) + \alpha h(z), \; 0 \leq \alpha \leq 1 \tag{4.10}$$

It can be shown that the new model augmentation scheme in Equation (4.10) does not change the result in Equation (4.9). If we expand $L(F_t)$ around $F_{t-1}$, we have

$$\begin{aligned} L(F_t) &= L\left((1-\alpha)F_{t-1} + \alpha h\right) \\ &= L\left(F_{t-1} + \alpha(h - F_{t-1})\right) \\ &\approx L(F_{t-1}) + \alpha \left\langle \nabla L(F_{t-1}), h - F_{t-1} \right\rangle \end{aligned} \tag{4.11}$$

In Equation (4.11), the term $\langle \nabla L(F_{t-1}), h - F_{t-1} \rangle$ must be negative in order to ensure the decrease of the loss functional. Whenever this term is equal to or greater then zero, we say that the algorithm converge to a critical point of the loss functional. For probability density estimation, we have

$$\langle \nabla L(F_{t-1}), h - F_{t-1} \rangle = n - \sum_{i=1}^{n} \frac{h(z_i)}{F_{t-1}} \tag{4.12}$$

In summary, the general boosting framework for probability density estimation is given in Algorithm (4.1).

**Algorithm 4.1** A general boosting framework for probability density estimation.

```
Set $F_0(z)$ to uniform on the domain of $z$
for $t = 1$ to $T$ do
    $(a)$ Set $w_i = \frac{1}{F_{t-1}(z_i)}$
    $(b)$ Find $h_t \in \mathcal{H}$ to maximize $\sum_i w_i h_t(z_i)$
    $(c)$ Convergence test: if $\sum_i w_i h_t(z_i) \leq n$ then break
    $(d)$ Find $\alpha_t = \arg\min_{0 \leq \alpha \leq 1} - \sum_i \log\left((1-\alpha)F_{t-1}(z_i) + \alpha h_t(z_i)\right)$
    $(e)$ Set $F_t = (1-\alpha_t)F_{t-1}(z_i) + \alpha_t h_t(z_i)$
end for
Output the final model $F_T$
```

## 4.2 Boosting Learning for GMMs

Gaussian mixture models (GMMs) are popular parametric models widely used for probability density estimation. A GMM is a finite sum of weighted Gaussian distributions, which has a nice property that it is capable of approximating arbitrary continuous PDFs arbitrarily closely provided that a sufficiently large number of Gaussian components are present in the finite sum. In the section, we focus on the maximum likelihood learning of GMMs. We show how we derive a novel boosting learning algorithm for GMMs, following the methodology of the general boosting framework for probability density estimation, which was introduced in the previous section. The resulting boosting learning algorithm for GMMs is thus referred to as the boosting GMM algorithm.

For the learning of GMMs, the space of weak learners is constrained to be all Gaussian mixture models having $M$ mixture components

$$\mathcal{H} = \{\text{All GMMs having } M \text{ mixture components}\} \tag{4.13}$$

We start with an initial estimate, which is a GMM

$$F_0(x) = \sum_{k=1}^{M} c_{0k} \mathcal{N}(x|\mu_{0k}, \Sigma_{0k}) \tag{4.14}$$

and iteratively add to this estimate a small component at step $t$

$$F_t(x) = (1-\alpha_t)F_{t-1}(x) + \alpha_t h_t(x) \tag{4.15}$$

where $h_t(x)$ is also a GMM

$$h_t(x) = \sum_{k=1}^{M} c_{tk} \mathcal{N}(x|\mu_{tk}, \Sigma_{tk}) \tag{4.16}$$

It can be easily proved that $F_t(x)$ is a valid PDF and a GMM at all steps of the boosting algorithm.

Let the loss functional on $F_t$ be

$$L(F_t) = -\sum_i \log\left(F_t(x_i)\right) \tag{4.17}$$

The first-order Taylor expansion of $L(F_t)$ around $F_{t-1}$ reads

$$\begin{aligned} L(F_t) &= L\left((1-\alpha_t)F_{t-1} + \alpha_t h_t\right) \\ &= L\left(F_{t-1} + \alpha_t(h_t - F_{t-1})\right) \\ &\approx L(F_{t-1}) - \alpha_t \sum_i \frac{h_t(x_i) - F_{t-1}(x_i)}{F_{t-1}(x_i)} \\ &= L(F_{t-1}) - \alpha_t \sum_i \frac{h_t(x_i)}{F_{t-1}(x_i)} + n\alpha_t \end{aligned} \tag{4.18}$$

Thus, in order to decrease $L(F_t)$, we form an optimization problem

$$\begin{aligned} h_t^* &= \arg\max_{h_t} \sum_i \frac{h_t(x_i)}{F_{t-1}(x_i)} \\ &= \arg\max_{h_t} \sum_i w_t(x_i) h_t(x_i) \end{aligned} \tag{4.19}$$

where $w_t(x_i)$ may be considered as the weight on the training sample $x_i$ at step $t$, which is defined as

$$w_t(x_i) = \frac{1}{F_{t-1}(x_i)} \tag{4.20}$$

The optimization problem in Equation (4.19) may be solved iteratively via the EM algorithm. The re-estimation formulas for the parameters of $h_t$

**Algorithm 4.2** The boosting GMM algorithm.

```
Input: $X = \{x_1, x_2, \cdots, x_n\}$, $T$
Initialize: $w(x_i) = \frac{1}{n}$, $F_0 = 0$
for $t = 1$ to $T$ do
    (a) Estimate a GMM $h_t$ according to Equations (4.21), (4.22), and (4.23)
    (b) Find $\alpha_t = \arg\max_{0 \le \alpha \le 1} \sum_i \log\left((1-\alpha)F_{t-1}(x_i) + \alpha h_t(x_i)\right)$
    (c) If $\sum_i \log\left((1-\alpha_t)F_{t-1}(x_i) + \alpha_t h_t(x_i)\right) \le \sum_i \log\left(F_{t-1}(x_i)\right)$ then break
    (d) Set $F_t = (1-\alpha_t)F_{t-1} + \alpha_t h_t$
    (e) Update the weights $w(x_i) = \frac{1}{F_t(x_i)}$
end for
Output the final model $F_T$
```

(reminder: $h_t$ is a GMM) are given by

$$\hat{c}_k = \sum_i w_t(x_i)\gamma_k(x_i) \tag{4.21}$$

$$\hat{\mu}_k = \frac{\sum_i w_t(x_i)\gamma_k(x_i)x_i}{\sum_i w_t(x_i)\gamma_k(x_i)} \tag{4.22}$$

$$\hat{\Sigma}_k = \frac{\sum_i w_t(x_i)\gamma_k(x_i)(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T}{\sum_i w_t(x_i)\gamma_k(x_i)} \tag{4.23}$$

where $\gamma_k(x_i)$ is the posterior probability of the $k^{th}$ Gaussian component of the GMM given the training sample $x_i$, namely $\gamma_k(x_i) = P(k|x_i)$, and is computed as follows:

$$\gamma_k(x_i) = \frac{c_k \mathcal{N}(x_i|\mu_k, \Sigma_k)}{\sum_j c_j \mathcal{N}(x_i|\mu_j, \Sigma_j)} \tag{4.24}$$

where $\gamma_k(x_i)$ may be intuitively interpreted as the "responsibility" of the $k^{th}$ Gaussian component of the GMM for the training sample $x_i$. The resulting boosting GMM algorithm is summarized in Algorithm (4.2).

Equations (4.21), (4.22), and (4.23) suggest that a GMM $h_t$ may also be learned from a weighted version of the training samples in the standard EM manner, where the weights on the training samples rely on the (reciprocal) probabilities of the training samples given by the model built at the previous step. This meets our intuition of boosting that strong emphasis is put on those training samples which have low probabilities given by the previous model. Thus, an alternative boosting GMM algorithm is described in Algo-

**Algorithm 4.3** The alternative boosting GMM algorithm.

```
Input: $X = \{x_1, x_2, \cdots, x_n\}$, $r$, $T$
Initialize: $w(x_i) = \frac{1}{n}$, $F_0 = 0$
**for** $t = 1$ to $T$ **do**
    (a) Sample $X_t$ from $X$ according to $w$ and $r$
    (b) Estimate a GMM $h_t$ from $X_t$
    (c) Find $\alpha_t = \arg\max_{0\leq\alpha\leq 1} \sum_i \log\left((1-\alpha)F_{t-1}(x_i) + \alpha h_t(x_i)\right)$
    (d) If $\sum_i \log\left((1-\alpha_t)F_{t-1}(x_i) + \alpha_t h_t(x_i)\right) \leq \sum_i \log\left(F_{t-1}(x_i)\right)$ then break
    (e) Set $F_t = (1-\alpha_t)F_{t-1} + \alpha_t h_t$
    (f) Update the weights $w(x_i) = \frac{1}{F_t(x_i)}$
**end for**
Output the final model $F_T$
```

rithm (4.3), which is similar to the ones in [83, 25], where at each boosting step, we randomly re-sample the training data set according to the weights on the training samples and train a GMM on the re-sampled training data set with the standard EM algorithm.

In the alternative boosting GMM algorithm presented in Algorithm (4.3), a remaining question is how we re-sample the training data set. In general, we can sort the training samples in the training data set by their weights in descending order and keep only a fraction $r$ (e.g. $r = 0.3$) of the training samples from the beginning of the sorted list. Another heuristic is to keep only the first $l$ training samples in the sorted list where $l$ is a quantity motivated by the entropy of the weight distribution

$$l = \text{ROUND}(\exp\{-\sum_i w_t(x_i)\log w_t(x_i)\}) \tag{4.25}$$

## 4.3 Boosting Learning for HMMs

Hidden Markov models (HMMs) are another kind of powerful parametric model for probability density estimation. Unlike GMMs and many other parametric probability density models, which are used to model the marginal probability distribution of independent and identically distributed (i.i.d.) observations, HMMs are used to model the joint probability distribution of an entire sequence of observations. Therefore, the general boosting framework for probability density estimation is not directly applicable to the learning of

HMMs. In this section, we show how we derive a boosting learning algorithm for HMMs, following the methodology of the general boosting framework for probability density estimation. This leads to a novel maximum likelihood learning algorithm for HMMs that we refer to as the boosting Baum-Welch algorithm.

If we were to directly apply the general boosting framework for probability density estimation to the learning of HMMs, we would end up with a so-called "finite mixture of HMMs" model. While the finite mixture of HMMs model can potentially be an extension of the current HMM framework, we will not adopt such an approach here, as it would require that we completely re-design the training and decoding algorithms. Noting that in an HMM, the initial state probability distribution and state transition probability distribution are relatively easy to learn robustly, we will focus on the much more difficult problem of learning the Gaussian mixture state observation PDFs.

To apply the boosting framework to the learning of the Gaussian mixture state observation PDFs of an HMM, an ad-hoc solution is to first perform Viterbi segmentation of the observation sequence and then apply the boosting GMM algorithm to learning the state observation PDFs independently based on the observations aligned to the individual states of the HMM. After that, the initial state probability distribution and state transition probability distribution are updated in the same way as in the original Baum-Welch algorithm. These steps are repeated until convergence is achieved.

Such a "hard" partition of the observation sequence into the individual states of the HMM using the Viterbi algorithm is somehow heuristic. In the following, we show that a boosting Baum-Welch algorithm can be achieved in a systematic manner by performing a "soft" partition of the observation sequence.

At a certain step of a boosting learning algorithm for HMMs, if the only thing that we want to update is the state observation PDF at state $j$, we can compute the negative gradient of the loss functional on the HMM likelihood function with respect to the the state observation PDF at state $j$

$$\begin{aligned} -\nabla L(P)|_{b_j} &= -\frac{\partial - \log P(O|\lambda)}{\partial b_j} \\ &= \frac{1}{P(O|\lambda)} \sum_{q_1 \cdots q_\tau = j, q_{\tau+1} \cdots q_T} \pi_{q_1} a_{q_1 q_2} a_{q_{\tau-1} j} a_{j q_{\tau+1}} \cdots a_{q_{T-1} q_T} \\ & \prod_{t=1}^{T} b_{q_t}(o_t))/b_j(o_\tau) \end{aligned} \tag{4.26}$$

where $P(O|\lambda)$ is the HMM likelihood function given by

$$P(O|\lambda) = \sum_{q_1 q_2 \cdots q_T} \left[ \pi_{q_1} b_{q_1}(o_1) \prod_{t=2}^{T} a_{q_{t-1} q_t} b_{q_t}(o_t) \right] \tag{4.27}$$

By some manipulations, we have

$$\begin{aligned} -\nabla L(p)|_{b_j} &= \frac{p(O, q_t = j|\lambda)}{P(O|\lambda) b_j(o_t)} \\ &= \frac{P(q_t = j|O, \lambda)}{b_j(o_t)} \\ &= \frac{\gamma_t(j)}{b_j(o_t)} \end{aligned} \tag{4.28}$$

where $\gamma_t(j) = P(q_t = j|O, \lambda)$ and may be computed as

$$\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{4.29}$$

In Equation (4.29), $\gamma_t(j)$ may be interpreted as the "responsibility" of state $j$ for the observation $o_t$. Thus, at this step of a boosting learning algorithm for HMMs, we find

$$\begin{aligned} h_j &= \arg\max_h \left\langle -\nabla L(p)|_{b_j}, h \right\rangle \\ &= \arg\max_h \sum_{t=1}^{T} \frac{\gamma_t(j)}{b_j(o_t)} h(o_t) \end{aligned} \tag{4.30}$$

If we define the weight on the training sample $o_t$ as

$$w_j(o_t) = \frac{\gamma_t(j)}{b_j(o_t)} \tag{4.31}$$

**Algorithm 4.4** The boosting Baum-Welch algorithm.

```
Input: $O = \{o_1, o_2, \cdots, o_T\}$, $L$
Initialize: Flat start on $\pi_i$, $a_{ij}$, $b_j(o_t)$
for $l = 1$ to $L$ do
    (a) Compute $w_j(o_t)$ according to Equations (4.29) and (4.31)
    (b) Estimate a GMM $h_j$ according to Equations (4.33), (4.34), (4.35)
    (c) Find $\alpha_j = \arg\max_{0\le\alpha\le1} \sum_{t=1}^{T} \log((1-\alpha)b_j(o_t) + \alpha h_j(o_t))$
    (d) Set $b_j = (1-\alpha_j)b_j + \alpha_j h_j$
    (e) Update $\pi_i$, $a_{ij}$
end for
Output: $\pi_i$, $a_{ij}$, $b_j(o_t)$
```

the optimization problem in Equation (4.30) becomes

$$h_j = \arg\max_h \sum_{t=1}^{T} w_j(o_t)h(o_t) \tag{4.32}$$

which is identical to the optimization problem for the boosting GMM algorithm in Equation (4.19). The re-estimation formulas for the Gaussian mixture state observation PDF at state $j$ are given as follows:

$$\hat{c}_{jk} = \sum_{t=1}^{T} w_j(o_t)\gamma_{jk}(o_t) \tag{4.33}$$

$$\hat{\mu}_{jk} = \frac{\sum_{t=1}^{T} w_j(o_t)\gamma_{jk}(o_t)o_t}{\sum_{t=1}^{T} w_j(o_t)\gamma_{jk}(o_t)} \tag{4.34}$$

$$\hat{\Sigma}_{jk} = \frac{\sum_{t=1}^{T} w_j(o_t)\gamma_{jk}(o_t)(o_t-\hat{\mu}_{jk})(o_t-\hat{\mu}_{jk})^T}{\sum_{t=1}^{T} w_j(o_t)\gamma_{jk}(o_t)} \tag{4.35}$$

where

$$\gamma_{jk}(o_t) = \frac{c_{jk}\mathcal{N}(o_t|\mu_{jk},\Sigma_{jk})}{\sum_m c_{jm}\mathcal{N}(o_t|\mu_{jm},\Sigma_{jm})} \tag{4.36}$$

After the state observation PDFs are updated, the initial state probability distribution and state transition probability distribution are updated in the same way as in the original Baum-Welch algorithm. The boosting Baum-Welch algorithm is summarized in Algorithm (4.4).

## 4.4 Experiments

### 4.4.1 Bayesian Classification

In order to demonstrate the effectiveness of the boosting GMM algorithm and boosting Baum-Welch algorithm for learning GMMs and HMMs respectively, we conduct GMM-based and HMM-based emotion classification experiments over an emotional speech database that we collected for several emotional speech analysis and synthesis purposes. In the experiments, we adopt a Bayesian or minimum error rate classifier for both classification tasks. Without loss of generality, we assume that a speech utterance is effectively represented by a sequence of observation vectors $O = o_1 o_2 \cdots o_T$. The joint probability distribution of $O$ is a described by a generative model which could have generated $O$. Let $P(O|c)$ denote the generative model for the $c^{th}$ emotion category. A Bayesian classifier or the minimum-error-rate classification rule [3, 1] is given by

$$\begin{aligned} c^* &= \arg\max_c P(c|O) \\ &= \arg\max_c P(O|c)P(c) \end{aligned} \tag{4.37}$$

where $P(c)$ is the prior probability distribution of the $c^{th}$ emotion category.

For the GMM-based classification experiments, $P(O|c)$ is modeled by a GMM, trained on the observation sequences belonging to the $c^{th}$ emotion category. Likewise, for the HMM-based classification experiments, $P(O|c)$ is modeled by an HMM, trained on the observation sequences belonging to the $c^{th}$ emotion category.

### 4.4.2 Speech Emotion Recognition

Speech emotion recognition is a relatively new direction in the areas of speech signal processing and pattern recognition [84, 85, 86, 87, 88]. Unlike speech recognition, which aims to extract the linguistic content from a speech signal while considering the emotions carried in the signal as irrelevant noise, speech emotion recognition aims to extract the non-lexical, paralinguistic information from the speech signal regardless of its verbal content. Like speech recognition, speech emotion recognition has turned out to be an important research topic and has many useful applications in our daily lives.

Just like speech recognition and many other pattern recognition problems, the problem of speech emotion recognition is often tackled by generative model based pattern recognition methods such as Bayesian classification with Gaussian mixture models (GMMs) [89] and hidden Markov models (HMMs) [28] as well as through classification of low-level acoustic features such as mel-frequency cesptral coefficients (MFCCs) [46] or perceptual linear prediction (PLP) coefficients [47].

### 4.4.3 Emotional Speech Database

We have collected an emotional speech database for several analysis and synthesis tasks. Our script consists of 720 semantically-neutral English sentences which were chosen to maximize the phonetic coverage [90]. A student actress whose mother language is American English was hired to speak each of these sentences, as naturally as possible, in the neutral, happy, sad, and angry manners. The speech waveforms were recorded in a studio environment at 44.1 kHz using a MOTU 8pre firewire audio interface and a Studio Projects B1 condenser microphone, and were downsampled to 16 kHz prior to further processing. The average length of the utterances in the database is about 3 to 4 seconds, depending on the emotion category. Thus, each of the four emotion categories in the database contains 720 utterances, that is speech data about 36 to 48 minutes long.

### 4.4.4 Experiment Results

#### Boosting GMM

We perform speech emotion recognition experiments on the emotional speech database described above using Bayesian classification with GMMs. For each experiment, we randomly selected from the database a training set consisting of 10 utterances per emotion and a test set consisting of 90 utterances per emotion. Therefore, the training set consisted of 40 utterances in total and the test set 360 utterances in total. Note that there were no overlapping utterances in the training and test sets. Instead of conducting a complete

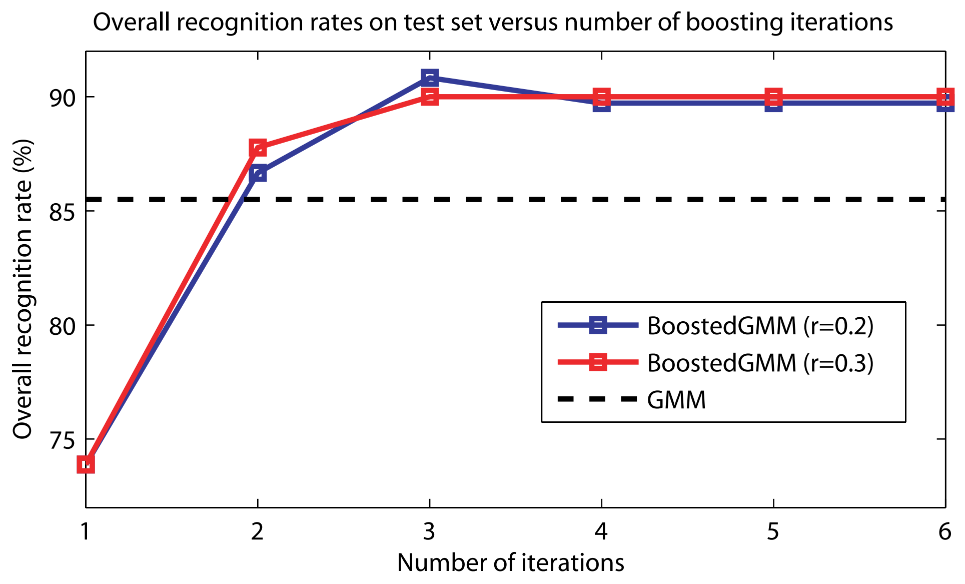

Figure 4.1: Comparison of the overall emotion recognition rates of the boosting GMM algorithm and the EM algorithm.

cross-validation process, which would be very time-consuming, we ran 10 such experiments independently, each of which involved a random selection of the training set and test set from the database, and the emotion recognition rates of these 10 experiments were averaged. We believed that in this way such average would represent a well generalized emotion recognition rate over the database. An experiment was carried out as follows. For each speech frame in an utterance, we extracted a set of acoustic features including 12 MFCCs, the log energy, and the pitch value ($f_0$) using a 25 ms hamming window at a 10 ms frame rate. These basic parameters were augmented with their first derivative to form for each frame a 28-dimensional feature vector. Based on the training set, the class-conditional feature vector distributions were modeled by GMMs and estimated using both the boosting GMM algorithm and the standard EM algorithm, and the same Bayesian classifier was applied with two sets of estimated class-conditional GMMs.

Figure 4.1 shows the average overall emotion recognition rates (i.e., the average recognition rates across all 4 emotions of 10 independent experiments) of the boosting GMM algorithm on the test sets with two sampling fractions $r$ versus the number of boosting iterations used during training, and the figure compares these recognition rates with the average overall emotion recognition rate of the EM algorithm on the same test sets. The number of Gaussian mixtures in the GMMs and the number of EM iterations required for convergence were automatically determined by the F-J algorithm [91]. Our experiment results demonstrate that the emotion recognition rates can be effectively and significantly "boosted" by the boosting GMM algorithm, which is a natural, expected result and clear indication that the class-conditional probability distributions can be more accurately estimated by the boosting GMM algorithm than by the EM algorithm. It is worth mentioning that the boosting GMM algorithm converges very fast - only a few iterations (less than 5) would be sufficient to lead to a stable result. This relaxes the possible concern that the boosting GMM might require a lot more training time than the EM algorithm.

### Boosting Baum-Welch

We perform speech emotion recognition experiments based on the same emotional speech database using Bayesian classification with HMMs. For each experiment, we randomly selected from the database a training set consisting of 180 utterances per emotion and a test set consisting of 540 utterances per emotion. Therefore, the training set consists of $180 \times 4 = 720$ utterances in total and the test set $540 \times 4 = 2160$ utterances in total. Note that there are no overlapping utterances in the training and test sets. We ran 10 experiments independently, each of which involved a random selection of the training set and test set from the database, and the emotion recognition rates of these 10 experiments were averaged. We believe that in this way such average would represent a well generalized emotion recognition rate over the entire database. An experiment was carried out as follows. For each speech frame in an utterance, we extracted a set of basic acoustic features including 19 MFCCs, log energy, and pitch ($f_0$) using a 25 ms hamming window at a 10 ms frame rate. These basic acoustic features were augmented with their first and second derivatives to form a 63-dimensional feature vector.

For each emotion category, we trained an HMM using the Baum-Welch algorithm and the boosting Baum-Welch algorithm, respectively. The HMM was designed to have a left-to-right topology with 10 states, and each state is modeled by a Gaussian mixture state observation PDF. Maximum likelihood classification of the emotion was performed for every utterance in the test set.

Table 4.1: HMM-based speech emotion recognition experiment results.

| **Algorithm** | **Average Recognition Rate** |
|---|---|
| Baum-Welch | 87.8% |
| Boosting Baum-Welch | 91.2% |

Table 4.2: Average confusion matrices for HMM-based speech emotion recognition experiment results. The numbers are percentages. A/B stands for A: the Baum-Welch algorithm and B: the boosting Baum-Welch algorithm.

| | Neutral | Happiness | Sadness | Anger |
|---|---|---|---|---|
| Neutral | **94.4/96.9** | 6.0/3.1 | 0/0 | 0/0 |
| Happiness | 0/0 | **71.3/79.4** | 0/0 | 28.7/20.6 |
| Sadness | 1.9/0.9 | 0/0 | **93.2/95.5** | 4.9/3.6 |
| Anger | 0/0 | 9.4/7.0 | 0/0 | **90.6/93.0** |

We compare the average recognition results of the two HMM learning algorithms, namely the Baum-Welch algorithm and the boosting Baum-Welch algorithms, in Table 4.1. The experiment results clearly show that the proposed boosting Baum-Welch algorithm indeed improves the learning accuracy of the Gaussian mixture state observation PDFs of the emotion category dependent HMMs, leading to better emotion recognition performance than the original Baum-Welch algorithm. Further details of the experiment results may be revealed by the average confusion matrices shown in Table 4.2.

## 4.5 Summary

In this chapter, we propose a new maximum likelihood learning algorithm for HMMs, which we refer to as the boosting Baum-Welch algorithm. In the proposed boosting Baum-Welch algorithm, the problem HMM learning is formulated as a sequential optimization problem on a loss functional in an inner product function space instead of an iterative optimization problem on a log likelihood objective function in a model parameter space. Such a sequential optimization procedure in a function space may be used to provide a theoretical interpretation for the boosting algorithm from a very different perspective. Hence the name of boosting Baum-Welch algorithm. The boosting Baum-Welch algorithm can serve as an alternative maximum likelihood learning algorithm for HMMs to the traditional Baum-Welch or EM algorithm, and a preferred method for use in situations where there is insufficient training data available. The reason that the boosting Baum-Welch algorithm – being itself a maximum likelihood estimation technique – is less susceptible to the over-fitting problem than the traditional Baum-Welch or EM algorithm is that by design the boosting Baum-Welch algorithm tends to produce a “large margin” effect. Our experiments show that the boosting Baum-Welch algorithm can lead to a significant performance increase in an HMM-based speech emotion classification task in the case of small-size training data sets.

# CHAPTER 5

# ONE-VECTOR REPRESENTATION OF STOCHASTIC SIGNALS BASED ON ADAPTED ERGODIC HMMS

In this chapter, we propose a novel one-vector representation of stochastic signals for pattern recognition based on adapted ergodic HMMs. First, a single global HMM known as a universal background model (UBM) is learned from all stochastic signals in a training data set. Then, the UBM is adapted to the individual stochastic signals in the training data set using one of the HMM adaptation techniques described in Chapter 3 to produce the signal-specific HMMs. Finally, based on these signal-specific HMMs, we derive a one-vector representation of the stochastic signals by means of an upper bound approximation of the Kullback-Leibler divergence rate (KLDR) [92, 93] between two HMMs. Our experiments on an image-based recognition task, namely gender recognition from facial images, clearly demonstrate the effectiveness of the proposed one-vector representation of stochastic signals for potential use in many pattern recognition systems. To further demonstrate that the proposed one-vector representation of stochastic signals based on adapted ergodic HMMs can be an effective one-vector representation of images, we apply it to a practical application in computer vision, namely the challenging problem of automatic facial expression recognition from non-frontal view facial images. Our experiments of recognizing six universal facial expressions over extensive multiview facial images with seven pan angles and five tilt angles (i.e. a total of 35 views), which are synthesized from the BU-3DFE facial expression database [94], show promising results that outperform the state of the art recently reported.

The significant contribution of the proposed one-vector representation is that it possesses the following advantageous properties:

- The representation summarizes the probability distribution of the feature vectors in the feature vector set compactly and accurately and allows the statistical dependence among the feature vectors to be modeled with a systematic underlying structure of first-order Markov chain

[95].

- The representation performs unsupervised segmentation of the stochastic signals implicitly to reveal the local structures of the signals and to allow localized, segment-wise comparison of the signals.
- The representation is in a one-vector form ready for either supervised, semi-supervised, or unsupervised distance metric learning from the data to further reenforce its discriminatory power for classification.

In addition to the above properties, the proposed one-vector representation is rather generic in nature and may be used with various types of stochastic signals (e.g. image, video, speech, etc.) and applied to a broad range of pattern recognition tasks (e.g. classification, regression, etc.). It does not require the signals to be of the same size, nor does it require alignment of the signals. It is supposed to be robust to partial occlusions or corruption in the signals.

## 5.1 UBM and HMM Adaptation

As described in Chapter 3, HMMs are powerful statistical tools for modeling sequential data. An HMM is a doubly stochastic process consisting of an underlying, hidden, discrete random process which possesses the Markov property (namely a Markov chain having a finite number of states) and an observed, discrete, continuous, or mixed discrete-continuous random process which is a probabilistic function of the underlying Markov chain. The likelihood function of an HMM is given by

$$P(O|\lambda) = \sum_{q_1 q_2 \cdots q_T} \left[ \pi_{q_1} b_{q_1}(\mathbf{o}_1) \prod_{t=2}^{T} a_{q_{t-1}q_t} b_{q_t}(\mathbf{o}_t) \right] \tag{5.1}$$

Here, an HMM is completely determined by its parameters $\lambda = \{A, B, \Pi\}$, where $A$ is the state transition probability distribution matrix whose entries, $a_{ij} = p(q_t = S_j | q_{t-1} = S_i)$, $1 \leq i, j \leq N$, specify the probabilities of transition from state $S_i$ to state $S_j$ at time $t$; $B$ is the state-dependent observation probability distribution matrix whose entries, $b_{jk} = p(\mathbf{o_t} = \mathbf{v_k} | q_t = S_j)$, $1 \leq j \leq N, 1 \leq k \leq M$, specify the probabilities of emitting an observation symbol $\mathbf{v_k}$ given that the model is in state $S_j$ at time $t$; and $\Pi$ is the initial state probability distribution matrix whose entries, $\pi_i = p(q_1 = S_i)$, $1 \leq i \leq N$, specify the probabilities of the model being initially in state $S_i$. For the case of continuous observations, the entries of $B$ are given by continuous probability density functions, namely $b_j(\mathbf{o_t}) = p(\mathbf{o_t}|q_t = S_j)$, $1 \leq j \leq N$. One important class of continuous probability density function widely used for the state-dependent observation probability distributions of continuous-observation HMMs is the Gaussian mixture probability density function of the form

$$\begin{aligned} b_j(\mathbf{o}_t) &= \sum_{k=1}^{M} c_{jk} N(\mathbf{o}_t|\mu_{jk}, \Sigma_{jk}) \\ & \quad 1 \leq j \leq N, 1 \leq k \leq M \end{aligned} \tag{5.2}$$

where $M$ is the number of Gaussian components, $c_{jk}$ is the $k^{th}$ mixture weight, and $N(o_t|\mu_{jk}, \Sigma_{jk})$ is a multivariate Gaussian probability density function with mean vector $\mu_{jk}$ and covariance matrix $\Sigma_{jk}$.

It is very important to note that an HMM does not necessarily have to be used to model time series. Rather, it is capable of modeling a variety of stochastic signals. Different typologies of the HMM may be designed for different types of stochastic signals. For instance, a left-to-right HMM may be used for time series such as speech signals which have a clear temporal dimension, and an ergodic HMM may be used for data which is non-sequential in nature (e.g. images) by presenting the feature vectors in sequence. A single-state HMM reduces to a Gaussian mixture model (GMM), which may be used to model i.i.d. signals.

As described in Chapter 3, an HMM may be trained directly for each stochastic signal $O = \mathbf{o}_1, \mathbf{o}_2, \cdots, \mathbf{o}_T$ using the classical Bawm-Welch or expectation maximization (EM) algorithm [27]. However, since both Bawm-Welch and EM belong to maximum likelihood estimation techniques, training an HMM in this way will not be robust due to data scarcity. The development of Bayesian learning for HMMs provides a better solution. First, a single global HMM known as a universal background model (UBM) is learned (by the Baum-Welch or EM algorithm) from all stochastic signals in a training data set (or from all stochastic signals in a separate "universal" training set). Then, the UBM is adapted to the individual stochastic signals to produce the signal-specific HMMs via the linear transformation adaptation technique (i.e. maximum likelihood linear regression, or MLLR) [96] or maximum a posteriori (MAP) adaptation technique [52]. An HMM obtained for each stochastic signal in this manner is considered to be a robust statistical model for the signal in the sense that it describes the joint probability distribution of the feature vectors extracted from the signal accurately.

## 5.2 One-Vector Representation Formation

In the previous section, a stochastic signal is encoded by an adapted HMM. We now turn to derive a one-vector representation of the stochastic signals out of the adapted HMMs. Our objective is that the Euclidean distance between any two representational vectors is equivalent to the "distance" between the two corresponding HMMs in a statistical sense. A popular distance measure between two statistical models, $f(\mathbf{x})$ and $g(\mathbf{x})$, is given by the Kullback-Leibler divergence (KLD) [92]

$$D(f\|g) = \int_{\mathbf{x}} f(\mathbf{x}) \log \frac{f(\mathbf{x})}{g(\mathbf{x})} d\mathbf{x} \tag{5.3}$$

Given two N-state HMMs with M-component Gaussian mixture state observation densities, $\lambda_1 = \{A_1, B_1, \Pi_1\}$ and $\lambda_2 = \{A_2, B_2, \Pi_2\}$, where $A_p = \{\mathbf{a}_i^{[p]}\}_{i=1}^N = \{a_{ij}^{[p]}\}_{i,j=1}^N$, $B_p = \{b_i^{[p]}\}_{i=1}^N = \{\mathbf{c}_i^{[p]} = \{c_{ik}^{[p]}\}_{k=1}^M, \{\mu_{ik}^{[p]}, \Sigma_{ik}^{[p]}\}_{k=1}^M\}_{i=1}^N$, and $\Pi_p = \{\pi_i^{[p]}\}_{i=1}^N$, with $p = 1, 2$ denoting the model index, a natural extension of the KLD is the KLD rate (KLDR) [92, 93]

$$R(\lambda_1\|\lambda_2) = \lim_{T->\infty} \frac{1}{T} D(\lambda_1\|\lambda_2) \tag{5.4}$$

For two HMMs, the KLDR does not have a closed-form expression. However, Do [97] shows that there is a simple closed-form expression for a fairly tight upper bound of the KLDR between two ergodic HMMs

$$R(\lambda_1\|\lambda_2) \leq \sum_{i=1}^N \pi_i^{[1]} \left[ D(\mathbf{a}_i^{[1]}\|\mathbf{a}_i^{[2]}) + D(b_i^{[1]}\|b_i^{[2]}) \right] \tag{5.5}$$

Similarly, there is no closed-form expression for the KLD between two Gaussian mixture densities. However, there exists an upper bound of the KLD

$$
\begin{aligned}
D(b_i^{[1]} \| b_i^{[2]}) \leq D(\mathbf{c}_i^{[1]} \| \mathbf{c}_i^{[2]}) + \\
\sum_{k=1}^{M} c_{ik}^{[1]} D\left(N(\cdot|\mu_{ik}^{[1]}, \Sigma_{ik}^{[1]}) \| N(\cdot|\mu_{ik}^{[2]}, \Sigma_{ik}^{[2]})\right)
\end{aligned}
\tag{5.6}
$$

For two d-dimensional Gaussians, we can obtain a closed-form solution to the KLD as follows:

$$
\begin{aligned}
D\left(N(\cdot|\mu_{ik}^{[1]}, \Sigma_{ik}^{[1]}) \| N(\cdot|\mu_{ik}^{[2]}, \Sigma_{ik}^{[2]})\right) = \\
\frac{1}{2} \log \frac{\det \Sigma_{ik}^{[2]}}{\det \Sigma_{ik}^{[1]}} + \frac{1}{2} \mathrm{trace}({\Sigma_{ik}^{[2]}}^{-1} \Sigma_{ik}^{[1]}) + \\
\frac{1}{2} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T {\Sigma_{ik}^{[2]}}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) - \frac{d}{2}
\end{aligned}
\tag{5.7}
$$

Combining Equations (5.5), (5.6), and (5.7), we have

$$
\begin{aligned}
R(\lambda_1 \| \lambda_2) \leq \sum_{i=1}^{N} \pi_i^{[1]} \Bigg[ \sum_{l=1}^{N} \mathbf{a}_i^{[1]} \log \frac{\mathbf{a}_i^{[1]}}{\mathbf{a}_i^{[2]}} + \sum_{l=1}^{N} \mathbf{c}_i^{[1]} \log \frac{\mathbf{c}_i^{[1]}}{\mathbf{c}_i^{[2]}} + \\
\sum_{k=1}^{M} c_{ik}^{[1]} \Bigg( \frac{1}{2} \log \frac{\det \Sigma_{ik}^{[2]}}{\det \Sigma_{ik}^{[1]}} + \frac{1}{2} \mathrm{trace}({\Sigma_{ik}^{[2]}}^{-1} \Sigma_{ik}^{[1]}) + \\
\frac{1}{2} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T {\Sigma_{ik}^{[2]}}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) - \frac{d}{2} \Bigg) \Bigg]
\end{aligned}
\tag{5.8}
$$

Normally, $\lambda_1$ and $\lambda_2$ are both obtained by adapting a UBM $\lambda = \{A, B, \Pi\}$ to a small amount of adaptation data using one of the adaptation techniques described in the previous section and Chapter 3. Since the available adaptation data is scarce (i.e. just a single stochastic signal), from a pattern recognition perspective, it is advantageous to merely adapt a limited number of model parameters of the UBM in order to avoid over-fitting. In the following, we will discuss four cases.

### 5.2.1 Case 1: Adaptation of Means Only

Suppose during the model adaptation process only the Gaussian component means of $\lambda_1$ and $\lambda_2$ are adapted from the UBM $\lambda$. That is, $\Pi_1 = \Pi_2 = \Pi$, $A_1 = A_2 = A$, and $\{\{c_{ik}^{[1]}\}_{k=1}^{M}, \{\Sigma_{ik}^{[1]}\}_{k=1}^{M}\}_{i=1}^{N} = \{\{c_{ik}^{[2]}\}_{k=1}^{M}, \{\Sigma_{ik}^{[2]}\}_{k=1}^{M}\}_{i=1}^{N} =$

$\{\{c_{ik}\}_{k=1}^{M}, \{\Sigma_{ik}\}_{k=1}^{M}\}_{i=1}^{N}$. In this case, by some linear algebra, Equation (5.8) becomes

$$\begin{aligned} R(\lambda_1\|\lambda_2) &\leq \sum_{i=1}^{N} \pi_i \\ &\left[\sum_{k=1}^{M} c_{ik} \left(\frac{1}{2}(\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T \Sigma_{ik}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})\right)\right] \\ &= \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \left\| \sqrt{\pi_i c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[1]} - \sqrt{\pi_i c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[2]} \right\|^2 \end{aligned} \tag{5.9}$$

where $\|\cdot\|$ denotes the $L_2$ norm or Euclidean distance.

From Equation (5.9), if we approximate $R(\lambda_1\|\lambda_2)$ by its upper bound and form the augmented vectors

$$\mathbf{s}_p = \left[\sqrt{\pi_i c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[p]}\right]_{i=1 k=1}^{N \quad M} \tag{5.10}$$

where $p = 1, 2$, the squared Euclidean distance between $\mathbf{s}_1$ and $\mathbf{s}_2$ is equivalent to the KLDR between the two corresponding adapted HMMs $\lambda_1$ and $\lambda_2$ (up to a constant scale $\frac{1}{2}$).

### 5.2.2 Case 2: Adaptation of Means and Variances

Suppose during the model adaptation process the Gaussian component means and variances of $\lambda_1$ and $\lambda_2$ are both adapted from the UBM $\lambda$. That is, $\Pi_1 = \Pi_2 = \Pi$, $A_1 = A_2 = A$, and $\{\mathbf{c}_i^{[1]}\}_{i=1}^{N} = \{\mathbf{c}_i^{[2]}\}_{i=1}^{N} = \{\mathbf{c}_i\}_{i=1}^{N}$. We further assume that all covariance matrices of the Gaussian mixture densities are diagonal. (Note: It is a general property that a Gaussian mixture density with non-diagonal covariance matrices may be approximated arbitrarily closely by a Gaussian mixture density with diagonal covariance matrices given a sufficient number of Gaussian components.) That is, $\Sigma_{ik} = \text{diag}(\sigma_{ik}^2)$. In this case, unlike in Case 1, we adopt a symmetric version of the KLDR

$$R_s(\lambda_1\|\lambda_2) = \frac{1}{2}\left[R(\lambda_1\|\lambda_2) + R(\lambda_2\|\lambda_1)\right] \tag{5.11}$$

Substituting Equation (5.5) into Equation (5.11), we have

$$R_s(\lambda_1 \| \lambda_2) \leq \sum_{i=1}^{N} \pi_i D_s(b_i^{[1]} \| b_i^{[2]}) \tag{5.12}$$

where

$$D_s(b_i^{[1]} \| b_i^{[2]}) = \frac{1}{2} \left[ D(b_i^{[1]} \| b_i^{[2]}) + D(b_i^{[2]} \| b_i^{[1]}) \right] \tag{5.13}$$

is the symmetric version of the KLD between two Gaussian mixture densities $b_i^{[1]}$ and $b_i^{[2]}$. Here, the terms involving $\mathbf{a}_i^{[1]}$ and $\mathbf{a}_i^{[2]}$ all vanish by symmetry, and $\pi_i^{[1]}$ and $\pi_i^{[2]}$ are both replaced by $\pi_i$ given the assumed conditions. Campbell [16] shows that the symmetric KLD between two Gaussians can be approximated by

$$\begin{aligned}
& D_s\left( N(\cdot|\mu_{ik}^{[1]}, \Sigma_{ik}^{[1]}) \| N(\cdot|\mu_{ik}^{[2]}, \Sigma_{ik}^{[2]}) \right) \\
& \approx \frac{1}{4} \text{trace} \left( (\Sigma_{ik}^{[1]} - \Sigma_{ik}^{[2]}) \Sigma_{ik}^{-2} (\Sigma_{ik}^{[1]} - \Sigma_{ik}^{[2]}) \right) + \\
& \frac{1}{4} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T ({\Sigma_{ik}^{[1]}}^{-1} + {\Sigma_{ik}^{[2]}}^{-1}) (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \\
& = \frac{1}{2} ({\sigma_{ik}^2}^{[1]} - {\sigma_{ik}^2}^{[2]})^T \frac{1}{2} \Sigma_{ik}^{-2} ({\sigma_{ik}^2}^{[1]} - {\sigma_{ik}^2}^{[2]}) + \\
& \frac{1}{2} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T (\frac{1}{2} {\Sigma_{ik}^{[1]}}^{-1} + \frac{1}{2} {\Sigma_{ik}^{[2]}}^{-1}) (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})
\end{aligned} \tag{5.14}$$

Substituting Equation (5.14) into Equation (5.12), we have

$$\begin{aligned}
R_s(\lambda_1 \| \lambda_2) & \leq \frac{1}{2} \sum_{i=1}^{N} \pi_i \sum_{k=1}^{M} c_{ik} ({\sigma_{ik}^2}^{[1]} - {\sigma_{ik}^2}^{[2]})^T \frac{1}{2} \Sigma_{ik}^{-2} ({\sigma_{ik}^2}^{[1]} - {\sigma_{ik}^2}^{[2]}) + \\
& \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \pi_i c_{ik} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T (\frac{1}{2} {\Sigma_{ik}^{[1]}}^{-1} + \frac{1}{2} {\Sigma_{ik}^{[2]}}^{-1}) (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \\
& \approx \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \pi_i c_{ik} ({\sigma_{ik}^2}^{[1]} - {\sigma_{ik}^2}^{[2]})^T \frac{1}{2} \Sigma_{ik}^{-2} ({\sigma_{ik}^2}^{[1]} - {\sigma_{ik}^2}^{[2]}) + \\
& \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \pi_i c_{ik} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T \Sigma_{ik}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})
\end{aligned} \tag{5.15}$$

Here, the terms involving $\mathbf{c}_1$ and $\mathbf{c}_2$ vanish by symmetry, $c_{ik}^{[1]}$ and $c_{ik}^{[2]}$ are both replaced by $c_{ik}$, and we have made a further approximation which replaces the average of the two adapted covariance matrices by the corresponding covariance matrix of the UBM. Through some linear algebra, we can re-write Equation (5.15) as

$$R_s(\lambda_1\|\lambda_2) \leq \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\left\|\sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\,[1]} - \sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\,[2]}\right\|^2$$
$$+\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\left\|\sqrt{\pi_i c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[1]} - \sqrt{\pi_i c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[2]}\right\|^2 \qquad (5.16)$$

From Equation (5.16), if we approximate $R_s(\lambda_1\|\lambda_2)$ by its upper bound and form the augmented vectors

$$\mathbf{s}_p = \left[\sqrt{\pi_i c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[p]}; \sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\,[p]}\right]_{i=1k=1}^{N\quad M} \qquad (5.17)$$

where $p = 1, 2$, the squared Euclidean distance between $\mathbf{s}_1$ and $\mathbf{s}_2$ is equivalent to the symmetric KLDR between $\lambda_1$ and $\lambda_2$ (up to a constant scale $\frac{1}{2}$).

### 5.2.3 Case 3: Adaptation of Means, Variances, and Mixture Weights

Suppose during the model adaptation process the Gaussian component means and variances as well as the mixture weights of $\lambda_1$ and $\lambda_2$ are all adapted from the UBM $\lambda$. In this case, Equation (5.11) becomes

$$R_s(\lambda_1\|\lambda_2) \leq \sum_{i=1}^{N}\pi_i\left[\sum_{k=1}^{M}(c_{ik}^{[1]} - c_{ik}^{[2]})\log\frac{c_{ik}^{[1]}}{c_{ik}^{[2]}} + D_s(b_i^{[1]}\|b_i^{[2]})\right] \qquad (5.18)$$

Next, we have

$$\begin{aligned}
&\sum_{k=1}^{M}(c_{ik}^{[1]} - c_{ik}^{[2]})\log\frac{c_{ik}^{[1]}}{c_{ik}^{[2]}} \\
&\quad= \sum_{k=1}^{M}(c_{ik}^{[1]} - c_{ik}^{[2]})(\log\frac{c_{ik}^{[1]}}{c_{ik}} - \log\frac{c_{ik}^{[2]}}{c_{ik}}) \\
&\quad\approx \sum_{k=1}^{M}(c_{ik}^{[1]} - c_{ik}^{[2]})(\frac{c_{ik}^{[1]}}{c_{ik}} - \frac{c_{ik}^{[2]}}{c_{ik}}) \\
&\quad= \sum_{k=1}^{M}\frac{(c_{ik}^{[1]} - c_{ik}^{[2]})^2}{c_{ik}} \qquad (5.19)
\end{aligned}$$

Here, we have used the lemma $\log x \approx x - 1$ when $x \to 1$. Thus, we have

$$R_s(\lambda_1 \| \lambda_2) \le \sum_{i=1}^{N}\sum_{k=1}^{M} \frac{\pi_i (c_{ik}^{[1]} - c_{ik}^{[2]})^2}{c_{ik}} +$$
$$\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M} \left\| \sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [1]} - \sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [2]} \right\|^2 +$$
$$\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M} \left\| \sqrt{\pi_i c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[1]} - \sqrt{\pi_i c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[2]} \right\|^2 \tag{5.20}$$

From Equation (5.20), if we approximate $R_s(\lambda_1 \| \lambda_2)$ by its upper bound and form the augmented vectors

$$\mathbf{s}_p = \left[ \sqrt{\pi_i / c_{ik}} c_{ik}^{[p]}; \sqrt{\pi_i c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[p]}; \sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [p]} \right]_{i=1 k=1}^{N \quad M} \tag{5.21}$$

where $p = 1, 2$, the squared Euclidean distance between $\mathbf{s}_1$ and $\mathbf{s}_2$ is equivalent to the symmetric KLDR between $\lambda_1$ and $\lambda_2$ (up to a constant scale $\frac{1}{2}$).

### 5.2.4 Case 4: Full Adaptation

Suppose during the model adaptation process all parameters of $\lambda_1$ and $\lambda_2$ are adapted from the UBM $\lambda$. Without loss of generality, we assume that $\Pi_1 = \Pi_2 = \Pi$, as in an HMM, the initial state probability distribution may be absorbed in the state transition probability distribution and become irrelevant. In this case, Equation (5.11) becomes

$$R_s(\lambda_1 \| \lambda_2) \le \sum_{i=1}^{N} \pi_i \left[ \sum_{l=1}^{N} (a_{il}^{[1]} - a_{il}^{[2]}) \log \frac{a_{il}^{[1]}}{a_{il}^{[2]}} + \right.$$
$$\left. \sum_{k=1}^{M} (c_{ik}^{[1]} - c_{ik}^{[2]}) \log \frac{c_{ik}^{[1]}}{c_{ik}^{[2]}} + D_s(b_i^{[1]} \| b_i^{[2]}) \right] \tag{5.22}$$

Likewise, we have

$$\begin{aligned}
&\sum_{l=1}^{N}(a_{il}^{[1]} - a_{il}^{[2]}) \log \frac{a_{il}^{[1]}}{a_{il}^{[2]}} \\
&\quad = \sum_{l=1}^{N}(a_{il}^{[1]} - a_{il}^{[2]})(\log \frac{a_{il}^{[1]}}{a_{il}} - \log \frac{a_{il}^{[2]}}{a_{il}}) \\
&\quad \approx \sum_{l=1}^{N}(a_{il}^{[1]} - a_{il}^{[2]})(\frac{a_{il}^{[1]}}{a_{il}} - \frac{a_{il}^{[2]}}{a_{il}}) \\
&\quad = \sum_{l=1}^{N} \frac{(a_{il}^{[1]} - a_{il}^{[2]})^2}{a_{il}} \qquad (5.23)
\end{aligned}$$

Here, we have again used the lemma $\log x \approx x - 1$ when $x \to 1$. Together with the similar expression obtained for the term $\sum_{k=1}^{M}(c_{ik}^{[1]} - c_{ik}^{[2]}) \log \frac{c_{ik}^{[1]}}{c_{ik}^{[2]}}$ in Case 3, we have

$$\begin{aligned}
R_s(\lambda_1 \| \lambda_2) \le & \sum_{i=1}^{N}\sum_{l=1}^{N} \frac{\pi_i (a_{il}^{[1]} - a_{il}^{[2]})^2}{a_{il}} + \sum_{i=1}^{N}\sum_{k=1}^{M} \frac{\pi_i (c_{ik}^{[1]} - c_{ik}^{[2]})^2}{c_{ik}} + \\
& \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M} \left\| \sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [1]} - \sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [2]} \right\|^2 + \\
& \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M} \left\| \sqrt{\pi_i c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[1]} - \sqrt{\pi_i c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[2]} \right\|^2 \qquad (5.24)
\end{aligned}$$

From Equation (5.24), if we approximate $R_s(\lambda_1 \| \lambda_2)$ by its upper bound and form the augmented vectors

$$\begin{aligned}
\mathbf{s}_p = \Bigg[ & \left[\sqrt{\pi_i / a_{il}} a_{il}^{[p]}\right]_{l=1}^{N} ; \left[\sqrt{\pi_i / c_{ik}} c_{ik}^{[p]}\right]_{k=1}^{M} ; \\
& \left[\sqrt{\pi_i c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[p]}\right]_{k=1}^{M} ; \left[\sqrt{\pi_i c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [p]}\right]_{k=1}^{M} \Bigg]_{i=1}^{N} \qquad (5.25)
\end{aligned}$$

where $p = 1, 2$, the squared Euclidean distance between $\mathbf{s}_1$ and $\mathbf{s}_2$ is equivalent to the symmetric KLDR between $\lambda_1$ and $\lambda_2$ (up to a constant scale $\frac{1}{2}$).

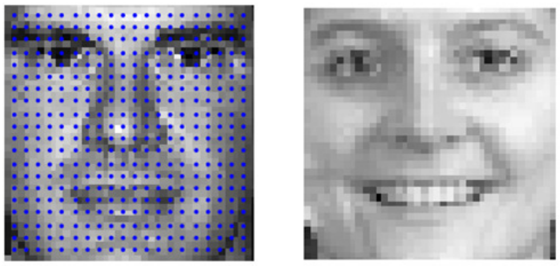

Figure 5.1: Male and female faces from the FERET database. SIFT feature vectors are extracted from a dense grid of pixels indicated by the blue dots.

## 5.3 Experiments

To show the effectiveness of the proposed one-vector representation of stochastic signals based on adapted ergodic HMMs, we carry out experiments on an image-based recognition task, namely gender recognition from facial images. Our data set consists of 4109 facial images from the grayscale FERET database [98]. Figure 5.1 shows a male face (left) and a female face (right) from the database. To increase the difficulty of the recognition task, the resolution of the faces is first reduced to $40 \times 40$ pixels. We then use the first 50% of the images for training and the remaining 50% for test. The faces in the images need not be aligned, and we simply use the outcome of a face detector [99]. For each image, we extract a set of 128-d SIFT [100] feature vectors from a dense grid of pixels, as illustrated in Figure 5.1 (left). These SIFT feature vectors form a sequence of observations for the image, which is used to learn a representational vector based on the techniques corresponding to Cases 1, 2, 3, and 4 in the one-vector representation formation process described in the previous section. The underlying MAP-adapted HMMs are ergodic, having 3 states, each modeled by Gaussian mixture densities. PCA is first used to reduce the dimensionality of the representational vectors to 500, and then LDA is used to learn an optimal distance metric from the training set. Finally, a simple, fast nearest-centroid classifier is used to perform gender classification. Table 5.1 displays the experiment results for the four cases and different numbers of Gaussian components in the Gaussian mixture state observation densities. Although we have not explored all possible design choices, the experiment results are very promising for the task. For comparison, the performance based on the "holistic" one-vector representation that the same classifier can achieve is 88.18%. Also note that our

Table 5.1: Experiment results in terms of gender recognition accuracy.

| #Gaussians | 8 | 16 | 32 |
|---|---|---|---|
| Case 1 | 89.98% | 92.02% | 93.14% |
| Case 2 | 91.39% | 92.07% | 92.07% |
| Case 3 | 91.33% | 92.07% | 92.63% |
| Case 4 | 91.24% | 92.07% | 92.99% |

proposed vector representation is a nonlinear transform of the HMM parameters, which has a root in the elegant K-L theory [92]. Clearly, the proposed vector representation of stochastic signals is effective for potential use in a variety of pattern recognition applications.

## 5.4 Application: Non-Frontal View Facial Expression Recognition

In this section, to further demonstrate that the proposed one-vector representation of stochastic signals based on adapted ergodic HMMs can be an effective one-vector representation of images, we apply it as a means of image representation to the challenging task of non-frontal view facial expression recognition and show that our method yields state-of-the-art performance on this difficult task.

Automatic facial expression recognition has been a popular research topic in the areas of multimedia, computer vision, and human-computer intelligent interaction [101]. One obvious reason for this is that machine recognition of facial expressions can be potentially applied to a variety of application scenarios in various facets of the society, for example, natural human-computer interaction interfaces, behavioral science study, smart advertising, movies and games, etc. Another reason is that automatic recognition of facial expressions may play a key role in many other tasks such as face/age/gender recognition (i.e. hard and soft biometrics) in the presence of facial expressions. For a very good general survey on this topic, refer to [102, 103, 101].

In the past few decades, research on facial expression recognition has mainly been focused on a particular type of facial image, namely that in which the facial pose is constrained to be frontal or near-frontal. While facial expression recognition from frontal or near-frontal facial images is of itself important, the heavily constrained facial pose greatly limits its practical utility. Nevertheless, the problem of facial expression recognition from non-frontal views has rarely been addressed in the literature. The reasons for this situation are many. Compared to facial expression recognition from the frontal or near-frontal view, facial expression recognition from non-frontal views is far more challenging due to the vast intra-class variations introduced by the different facial poses. More importantly, the research community has lacked a multiview facial expression database, partly due to the many difficulties in constructing one. Without such a database, research on non-frontal view facial expression recognition has been seriously impeded.

Fortunately, the recent development of a 3D facial expression database by Yin et al. at Binghamton University, known as the BU-3DFE database [94], offers an alternative opportunity. Based on the BU-3DFE database, a few researchers have begun to explore this fascinating area of non-frontal view facial expression recognition. They synthesized multiview facial images from the BU-3DFE database by rotating the 3D facial expression models in the database to the desired poses and projecting them onto a 2D image plane. Using the synthesized multiview facial images, Hu et al. [104] investigated the problem of facial expression recognition from non-frontal views with five pan angles: $0^o$, $30^o$, $45^o$, $60^o$ and $90^o$. They combined the "geometric features," defined by the location of 83 facial feature points, and various classifiers such as nearest neighbor and the support vector machine, to recognize six universal facial expressions. Zheng et al. [30] studied the same problem with the same five pan angles. Instead of using the geometric features, they employed the "texture features," defined as the scale-invariant feature transform (SIFT) [100] feature vectors extracted from the sparse location of the 83 facial feature points. They proposed a novel method for feature selection based on minimization of an upper bound of the Bayes error and reduced the dimensionality of the SIFT feature vectors. The reduced-dimensional feature vectors were then classified with the k-nearest-neighbor (KNN) classifier. However, there are two common pitfalls in both works. One is that they only investigate the non-frontal views of five coarsely-quantized pan angles, which is apparently far from being sufficient for realistic applications. The other is that their methods rely on the localization of the 83 facial feature points which were manually picked in their work. Automatic localization of facial feature points of itself is still an open research issue, especially for non-frontal view facial images. Therefore, these pitfalls have made the practical applicability of their methods very limited. In order to overcome these difficulties, Zheng et al. [29] extended their work in [30] in three aspects. First, they investigated 35 non-frontal views (combinations of seven pan angles and five tilt angles) instead of five. Second, they used dense SIFT features instead of sparse SIFT features, thus avoiding the difficulty of localizing the facial feature points. Third, they extended their feature selection method based on minimization of an upper bound of the Bayes error, assuming the distribution of the SIFT features is a mixture of Gaussians instead of a Gaussian. Their work has led to the state-of-the-art performance for non-frontal view facial expression based on the BU-3DFE database. To the best of our knowledge, the authors of these above-mentioned works are pioneers in this particular area of non-frontal view facial expression recognition.

In this section, we apply the proposed one-vector representation of stochastic signals based on adapted ergodic HMMs as a one-vector representation of facial images to tackling the challenging problem of non-frontal view facial expression recognition. Figure 5.2 gives a schematic overview of our approach. First, the SIFT feature vectors are extracted from a dense grid of every facial images. Next, an EHMM is trained over all facial images in the training set and is referred to as the universal background model (UBM). The UBM is then maximum a posteriori (MAP) [52] adapted to each facial image in the training and test sets to produce the image-specific EHMMs. Based on these image-specific EHMMs, a supervector representation of the facial images is obtained by means of our proposed one-vector representation of stochastic signals (Case 2) described earlier in this chapter. Finally, facial expression recognition is performed in the linear discriminant subspace of the EHMM supervectors (defined by a linear projection $W$) using the k-nearest-neighbor (KNN) classification algorithm. We conduct five-fold cross-validation experiments of recognizing six universal facial expressions over extensive multiview facial images with seven pan angles ($-45^o \sim +45^o$) and five tilt angles ($-30^o \sim +30^o$) (i.e. a total of 35 views), which are synthesized from the BU-3DFE facial expression database. Our experiment results are shown to be very promising, better than the state of the art recently reported.

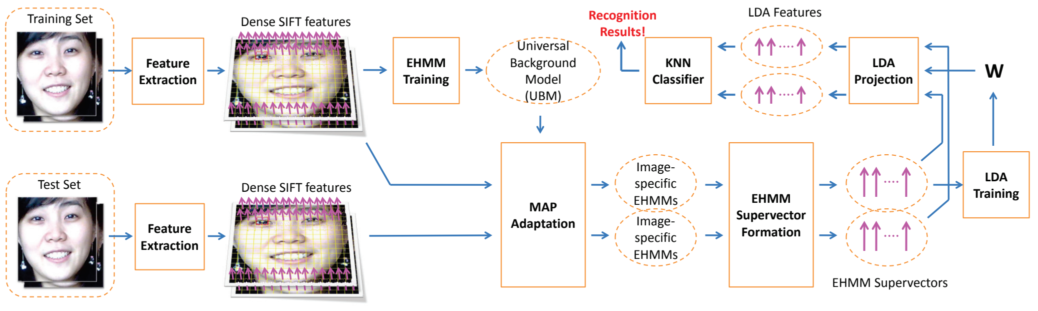

Figure 5.2: The schematic overview diagram of the proposed approach.

### 5.4.1 Database and Feature Extraction

The BU-3DFE database is a 3D facial expression database developed by Yin et al. at Binghamton University. It was designed to sample 3D facial behaviors with different prototypical emotional states. There are a total of 100 subjects in the database, 56 female and 44 male. The subjects are well distributed across different ethnic or racial ancestries, including White, Black, East-Asian, Middle-East Asian, Hispanic Latino, and others. During the recording session, each subject performed six universal facial expressions, namely anger (AN), disgust (DI), fear (FE), happiness (HA), sadness (SA), and surprise (SU), and the 3D geometric and texture models of the subject were captured. For a detailed description of the database, please refer to [94].

In order to synthesize multiview facial images from the BU-3DFE database, we first rotate every 3D facial expression model in the database by a certain pan angle and tilt angle. The ranges of pan and tilt angles of interest to our study are $\{-45^o, -30^o, -15^o, 0^o, 15^o, 30^o, 45^o\}$ and $\{-30^o, -15^o, 0^o, 15^o, 30^o\}$, respectively. We believe that the combination of these seven pan angles and five tilt angles is able to provide a sufficient level of quantization of the continuous non-frontal views in realistic environments. Then, we render the rotated 3D facial expression models into 2D images using OpenGL [105] with appropriate lighting simulation. Figure 5.3 illustrates a sample of the multiview facial images synthesized from the BU-3DFE database in the form of a cube.

For each facial image, we extract a set of dense SIFT features. Specifically, we place a dense grid on a facial image, and extract a 128-dimensional SIFT

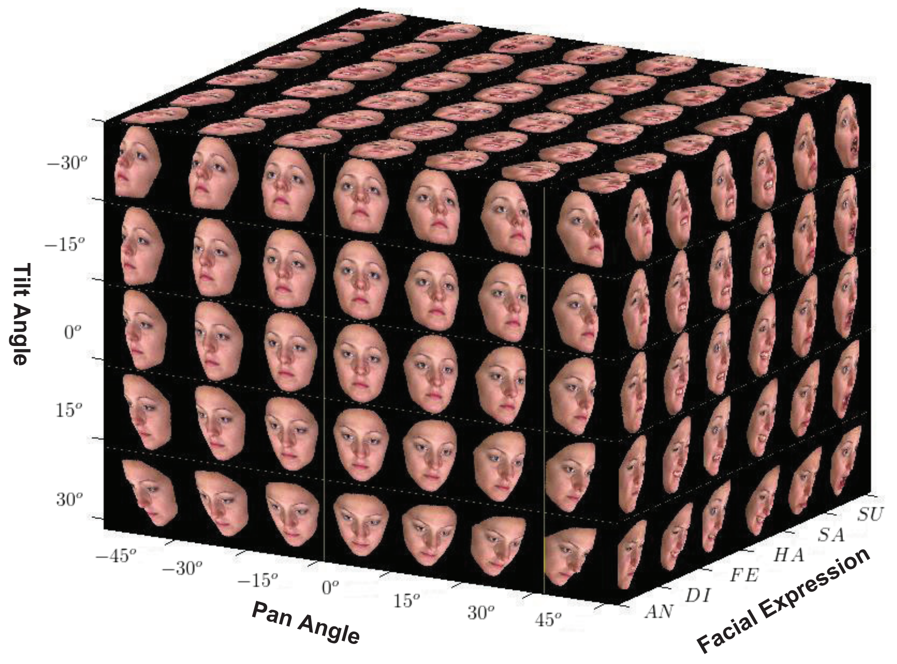

Figure 5.3: A sample of the multiview facial images synthesized from the BU-3DFE database in the form of a cube.

descriptor at each node of the grid with a fixed scale and orientation. The SIFT descriptor is formed from the histogram of intensity gradients within a neighborhood window of the grid node and is a distinctive feature to represent the texture variation in this local region. In this way, a facial image is encoded by a "bag" of SIFT feature vectors, as shown in Figure 5.4(a). Particular to the synthesized multiview facial images in this chapter, we perform a further step. Among the extracted SIFT feature vectors, we abandon those with extremely small magnitudes, which correspond to the SIFT feature vectors extracted from the black background in the images as well as those extracted from the low-contrast portion on the face, as shown in Figure 5.4(b).

For each facial image, the extracted SIFT feature vectors are sorted in the order of the grid node location $(x, y)$, with the $x$ coordinate of the location being the fastest changing variable. Note that the ordering here is in fact not important due to the EHMM modeling. Thus, an observation sequence, $O = \{\mathbf{o}_1, \mathbf{o}_2, \cdots, \mathbf{o}_T\}$, is formed for each facial image, where $\mathbf{o}_t$, $t = 1, 2, \cdots, T$ are the individual SIFT feature vectors (a.k.a. observations) and $T$ is the total number of SIFT feature vectors for the facial image.

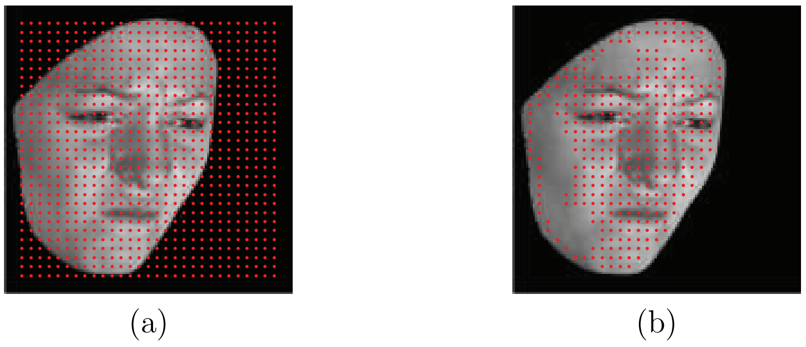

Figure 5.4: (a) The SIFT feature vectors are extracted from the grid nodes, shown as red dots. (b) We abandon the SIFT feature vectors with extremely small magnitudes, which correspond to the SIFT feature vectors extracted from the black background in the images as well as those extracted from the low-contrast portion on the face.

### 5.4.2 Experimental Results

To demonstrate the effectiveness of the proposed approach for non-frontal view facial expression recognition, we conduct experiments over extensive multiview facial images that we synthesize from the BU-3DFE database as described above. The projected facial images have an original resolution of $512 \times 512$ pixels. To speed up the feature extraction process, we downsample the facial images to $128 \times 128$ pixels and convert them into grayscale images. The 128-D SIFT feature vectors are then extracted from a dense grid with $4 \times 4$ pixel spacing on every grayscale facial image with a fixed scale (12 pixels) and orientation ($0^o$). The 100 subjects in the database are partitioned into five groups each of which consists of 20 subjects. We consider all six universal facial expressions, namely, anger (AN), disgust (DI), fear (FE), happiness (HA), sadness (SA), and surprise (SU). Thus, each group consists of $20(\text{subjects}) \times 7(\text{pan angles}) \times 5(\text{tilt angles}) \times 6(\text{facial expressions}) = 4200$ facial images. For classification purposes, we adopt a “universal” approach in which the classifier is trained with facial images of all views and tested with facial images of any view. Such a “universal” approach is very useful as it does not require the facial pose of a test facial image to be known a priori or estimated. In order to obtain confident results, we perform five-fold cross validations on the database. At each validation, four out of the five groups are used for training and the remaining group is used for test. The average overall facial expression recognition rate, as well as the average recognition

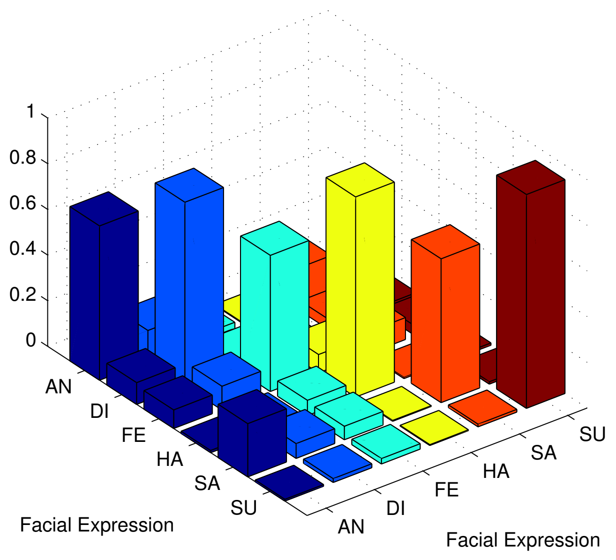

Figure 5.5: The average confusion matrix of six universal facial expressions: anger (AN), disgust (DI), fear (FE), happiness (HA), sadness (SA), and surprise (SU).

rates for different views and the average recognition rates for different facial expressions, are reported.

At each validation, all the facial images in the training set are used to train the UBM (3 states, 64-component Gaussian mixture emission densities), which is then MAP adapted to every facial image in both training and test sets to produce the image-specific EHMMs. The image-specific EHMMs are then used to construct the corresponding EHMM supervectors, corresponding to Case 2 in the one-vector representation formation process. The EHMM supervectors in the training set are then used to learn a linear discriminant analysis (LDA) [1] subspace, into which all the original EHMM supervectors in both training and test sets are projected. In the LDA subspace, the KNN classifier ($k = 20$) is employed to perform facial expression recognition.

The experiment results are shown in Table 5.2. The rightmost column represents the average recognition error rates for different views (a total of 35 views), the bottom row represents the average recognition error rates for different facial expressions (a total of six universal facial expressions), and the bottom-right corner cell represents the average overall recognition error rate. Figure 5.5 shows the average confusion matrix of the six universal facial expressions.

Table 5.2: Experiment results in terms of recognition error rates. The leftmost column indicates the different views (pan and tilt angles $x, y$ in degrees), and the top row indicates the different facial expressions.

| % | AN | DI | FE | HA | SA | SU | Ave. |
|---|---|---|---|---|---|---|---|
| $-45, -30$ | 21.0 | 34.0 | 48.0 | 10.0 | 45.0 | 5.0 | 27.2 |
| $-45, -15$ | 23.0 | 21.0 | 41.0 | 7.0 | 47.0 | 10.0 | 24.8 |
| $-45, +0$ | 34.0 | 19.0 | 35.0 | 17.0 | 32.0 | 7.0 | 24.0 |
| $-45, +15$ | 34.0 | 12.0 | 37.0 | 19.0 | 40.0 | 4.0 | 24.3 |
| $-45, +30$ | 40.0 | 18.0 | 30.0 | 14.0 | 41.0 | 9.0 | 25.3 |
| $-30, -30$ | 25.0 | 20.0 | 33.0 | 14.0 | 31.0 | 8.0 | 21.8 |
| $-30, -15$ | 30.0 | 24.0 | 54.0 | 22.0 | 37.0 | 8.0 | 29.2 |
| $-30, +0$ | 26.0 | 20.0 | 53.0 | 11.0 | 38.0 | 7.0 | 25.8 |
| $-30, +15$ | 20.0 | 19.0 | 41.0 | 7.0 | 43.0 | 6.0 | 22.7 |
| $-30, +30$ | 37.0 | 12.0 | 35.0 | 7.0 | 37.0 | 5.0 | 22.2 |
| $-15, -30$ | 37.0 | 13.0 | 43.0 | 11.0 | 44.0 | 4.0 | 25.3 |
| $-15, -15$ | 28.0 | 19.0 | 35.0 | 13.0 | 42.0 | 6.0 | 23.8 |
| $-15, +0$ | 25.0 | 18.0 | 37.0 | 14.0 | 38.0 | 8.0 | 23.3 |
| $-15, +15$ | 33.0 | 22.0 | 52.0 | 19.0 | 41.0 | 5.0 | 28.7 |
| $-15, +30$ | 28.0 | 29.0 | 38.0 | 12.0 | 33.0 | 6.0 | 24.3 |
| $+0, -30$ | 26.0 | 21.0 | 42.0 | 11.0 | 35.0 | 6.0 | 23.5 |
| $+0, -15$ | 36.0 | 17.0 | 46.0 | 11.0 | 32.0 | 6.0 | 24.7 |
| $+0, +0$ | 39.0 | 18.0 | 38.0 | 18.0 | 34.0 | 3.0 | 25.0 |
| $+0, +15$ | 27.0 | 15.0 | 32.0 | 10.0 | 42.0 | 5.0 | 21.8 |
| $+0, +30$ | 30.0 | 18.0 | 38.0 | 10.0 | 40.0 | 6.0 | 23.7 |
| $+15, -30$ | 35.0 | 23.0 | 45.0 | 12.0 | 33.0 | 6.0 | 25.7 |
| $+15, -15$ | 37.0 | 22.0 | 49.0 | 13.0 | 26.0 | 6.0 | 25.5 |
| $+15, +0$ | 37.0 | 22.0 | 49.0 | 11.0 | 30.0 | 9.0 | 26.3 |
| $+15, +15$ | 43.0 | 14.0 | 41.0 | 13.0 | 22.0 | 7.0 | 23.3 |
| $+15, +30$ | 37.0 | 13.0 | 31.0 | 8.0 | 41.0 | 5.0 | 22.5 |
| $+30, -30$ | 37.0 | 12.0 | 30.0 | 10.0 | 39.0 | 6.0 | 22.3 |
| $+30, -15$ | 29.0 | 16.0 | 39.0 | 12.0 | 35.0 | 4.0 | 22.5 |
| $+30, +0$ | 27.0 | 24.0 | 46.0 | 14.0 | 42.0 | 6.0 | 26.5 |
| $+30, +15$ | 32.0 | 31.0 | 43.0 | 15.0 | 28.0 | 6.0 | 25.8 |
| $+30, +30$ | 47.0 | 17.0 | 45.0 | 7.0 | 25.0 | 6.0 | 24.5 |
| $+45, -30$ | 45.0 | 14.0 | 28.0 | 6.0 | 32.0 | 5.0 | **21.7** |
| $+45, -15$ | 36.0 | 23.0 | 34.0 | 18.0 | 41.0 | 9.0 | 26.8 |
| $+45, +0$ | 42.0 | 21.0 | 38.0 | 11.0 | 41.0 | 9.0 | 27.0 |
| $+45, +15$ | 28.0 | 13.0 | 44.0 | 10.0 | 42.0 | 5.0 | 23.7 |
| $+45, +30$ | 24.0 | 33.0 | 44.0 | 15.0 | 46.0 | 9.0 | 28.5 |
| Ave. | 32.4 | 19.6 | 40.4 | 12.3 | 37.0 | **6.3** | **24.7** |

Table 5.3: Comparison of the recognition error rates of our approach and previous approaches.

| Hu et al. 2008 | Zheng et al. 2009 | Zheng et al. 2010 | Our approach |
|---|---|---|---|
| Average: 33.5%<br>Best view: 28.5% ($45^o$)<br>Best expr: 20.5% (SU) | 24.7%<br>20.5% ($60^o$)<br>9.3% (SU) | 31.8%<br>25.2% ($0^o, 0^o$)<br>12.6% (SU) | **24.7%**<br>**21.7%** ($45^o, -30^o$)<br>**6.3%** (SU) |
| 5 pan-angle views; Manual facial feature localization (ffl) | Same as Hu et al. 2008 | 35 total views; No ffl | Same as Zheng et al. 2009 |

Our experiment results are promising compared to the state of the art recently reported. In the work of Hu et al. [104], the best average overall facial expression recognition error rate was reported to be 33.5% with the help of a support vector machine (SVM) classifier. In the work of Zheng et al. [30], they reported a reduced overall facial expression recognition error rate of 21.7% on the same database under the same experimental settings. However, note that the experimental settings of both of their works only involve non-frontal views of five pan angles (as compared to 35 combinations of seven pan angles and five tilt angles in the experimental settings of our work). Also note that both works completely rely on the location of 83 facial feature points in all facial images, which have to be labeled manually. The impractical manual labeling of facial feature point locations seriously limits the applicability of their approaches. In contrast, our proposed approach is fully automatic, requiring neither facial alignment nor facial feature point localization. In our experiments, we achieve an average overall facial expression recognition error rate of 24.7%, which is significantly lower than that of Hu et al.'s work and comparable to that of Zheng et al.'s work, but under far more challenging and useful experimental settings. In addition, our method significantly outperforms the method in Zheng et al.'s new work [29] under the same experiment settings and leads to state-of-the-art performance for the task of non-frontal view facial expression recognition on the BU-3DFE database. We bring to the reader's attention that our experiment results in this chapter should be considered preliminary, as we have not had time to explore all possible design choices and to find the optimal parameters for the experiments which could possibly lead to even better experiment results. A detailed comparison of the results of our approach and those of the previous approaches on the same task is provided in Table 5.3.

We note that our approach yields better performance for disgust, happiness, and surprise than for anger, fear, and sadness. Especially, the performance for surprise is noticeably the best among all (6.3% average recognition error rate). The performances across different views are, however, comparable to one another. This observation strongly supports that our approach is robust to the varying views of the facial images.

## 5.5 Summary

In this chapter, we propose a novel one-vector representation of stochastic signals based on adapted ergodic HMMs. This one-vector representation is generic in nature and may be used with various types of stochastic signals (e.g. image, video, speech, etc.) and applied to a broad range of pattern recognition tasks (e.g. classification, regression, etc.). More importantly, by combining the one-vector representation with optimal distance metric learning (e.g. linear discriminant analysis) directly from the data, the performance of a pattern recognition system may be significantly improved. Our experiments on an image-based recognition task, namely gender recognition based on facial images, clearly demonstrate the effectiveness of the proposed one-vector representation of stochastic signals for potential use in many pattern recognition systems. To further demonstrate that the proposed one-vector representation of stochastic signals based on adapted ergodic HMMs can be an effective one-vector representation of images, we apply it to a practical application in computer vision, namely the challenging problem of automatic facial expression recognition from non-frontal view facial images. Our experiments of recognizing six universal facial expressions over extensive multiview facial images with seven pan angles and five tilt angles (i.e. a total of 35 views), which are synthesized from the BU-3DFE facial expression database, show promising results that outperform the state of the art recently reported.

# CHAPTER 6

# ONE-VECTOR REPRESENTATION OF STOCHASTIC SIGNALS BASED ON ADAPTED LEFT-TO-RIGHT HMMS

In Chapter 5, we propose a one-vector representation of stochastic signals for pattern recognition based on adapted ergodic HMMs. We experimentally show that it can be an effective one-vector representation of images. Due to their fully connected state diagrams of the underlying Markov chains, ergodic HMMs are capable of characterizing stochastic signals which are non-sequential in nature, such as images in the two-dimensional spatial space. However, for certain types of stochastic signals, such as speech, audio, and video, which are sequential in nature or which have a clear temporal dimension, left-to-right HMMs might turn out to be better models. A left-to-right HMM is an HMM with a special left-to-right topology, having the following constraints imposed on the initial state probability distribution and state transition probability distribution:

$$\pi_i = \begin{cases} 1 & i = 1 \\ 0 & i > 1 \end{cases} \tag{6.1}$$

$$a_{ij} = 0, i > j \tag{6.2}$$

$$a_{ij} = 0, j > i + \Delta \, (\Delta = 1 \text{ or } 2 \text{ typically}) \tag{6.3}$$

That is, in a left-to-right HMM, the underlying hidden state sequence always starts with the state $S_1$. All states of the HMM are emitting states, each of which generates an observation whenever it is entered, except that the last state of the HMM is a non-emitting state, which never generates observations. Figure 6.1 shows the underlying state diagram of a commonly seen left-to-right HMM ($\Delta = 1$). Such an HMM with a left-to-right topology and a final non-emitting absorbing state is transient, meaning that it will generate a finite number of observations almost surely. A fortunate note on left-to-right HMMs is that the above constraints on the initial state probability distribution and state transition probability distribution in a left-to-right

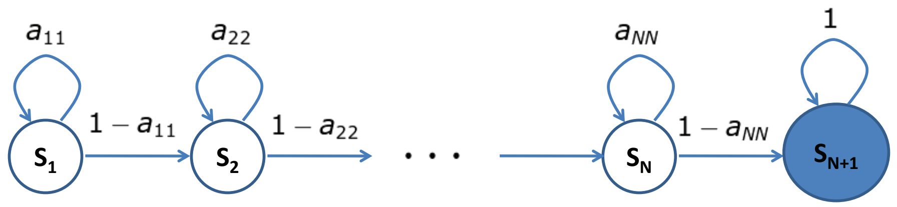

Figure 6.1: The underlying state diagram of a transient HMM with a left-to-right topology and a final non-emitting absorbing state.

HMM do not affect the formulas in the HMM learning and adaptation algorithms presented in Chapters 3 and 4.

In this chapter, we propose a novel one-vector representation of stochastic signals for pattern recognition based on adapted left-to-right HMMs. This one-vector representation of stochastic signals is complimentary to the one-vector representation of stochastic signals proposed in Chapter 5, and turns out to be a potentially more appropriate one-vector representation for stochastic signals such as speech and video which are sequential in nature or which have a clear temporal dimension of which the dynamics need to be captured.

## 6.1 One-Vector Representation Formation

Suppose two left-to-right transient HMMs, $\lambda_1 = \{A_1, B_1, \boldsymbol{\pi}_1\}$ and $\lambda_2 = \{A_2, B_2, \boldsymbol{\pi}_2\}$, are both adapted from a UBM, $\lambda = \{A, B, \boldsymbol{\pi}\}$, respectively, using one of the model adaptation techniques for HMMs presented in Chapter 3, where for $p = 1, 2$ (the model index), $\boldsymbol{\pi}_p = [1, 0, \cdots, 0]^T$, $B_p = \{b_i^{[p]}\}_{i=1}^N = \{\mathbf{c}_i^{[p]} = \{c_{ik}^{[p]}\}_{k=1}^M, \{\boldsymbol{\mu}_{ik}^{[p]}, \Sigma_{ik}^{[p]}\}_{k=1}^M\}_{i=1}^N$, and

$$A_p = \{\mathbf{a}_i^{[p]}\}_{i=1}^{N+1} = \begin{pmatrix} a_{11}^{[p]} & 1-a_{11}^{[p]} & 0 & \cdots & 0 & 0 \\ 0 & a_{22}^{[p]} & 1-a_{22}^{[p]} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1-a_{N-1,N-1}^{[p]} & 0 \\ 0 & 0 & 0 & \cdots & a_{NN}^{[p]} & 1-a_{NN}^{[p]} \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \tag{6.4}$$

Here, as shown in Figure 6.1, $N$ denotes the number of emitting states of the HMM, and $M$ denotes the number of Gaussian components in the Gaussian mixture state observation PDFs associated with the emitting states of the HMM. Let the family of all states of the HMM be denoted by $\mathcal{X} = \mathcal{X}_o \cup \mathcal{X}_u$, where $\mathcal{X}_o$ is the family of all emitting states, $\mathcal{X}_o = \{S_1, S_2, \cdots, S_N\}$, and $\mathcal{X}_u$ the family of all non-emitting states, $\mathcal{X}_u = \{S_{N+1}\}$. Let $\hat{A}_p$, $\hat{B}_p$ and $\hat{\boldsymbol{\pi}}_p$ denote the model parameters of the HMM which correspond to the family of emitting states $\mathcal{X}_o$, where, resultantly, $\hat{\boldsymbol{\pi}}_p = [1, 0, \cdots, 0]^T$, $\hat{B}_p = B_p$, and

$$\hat{A}_p = \{\mathbf{a}_i^{[p]}\}_{i=1}^N = \begin{pmatrix} a_{11}^{[p]} & 1 - a_{11}^{[p]} & 0 & \cdots & 0 \\ 0 & a_{22}^{[p]} & 1 - a_{22}^{[p]} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 - a_{N-1,N-1}^{[p]} \\ 0 & 0 & 0 & \cdots & a_{NN}^{[p]} \end{pmatrix} \tag{6.5}$$

It is shown in the recent work of Silva and Narayanan [106] that the KLDR between two left-to-right transient HMMs, $\lambda_1$ and $\lambda_2$, has a computationally efficient closed-form upper bound

$$R(\lambda_1 \| \lambda_2) \leq D(\boldsymbol{\pi}_1 \| \boldsymbol{\pi}_2) + \hat{\boldsymbol{\pi}}_1^T (I - \hat{A}_1)^{-1} \left[ \hat{d}_{KLD}^A + \hat{d}_{KLD}^B \right] \tag{6.6}$$

where $\hat{d}_{KLD}^A$ and $\hat{d}_{KLD}^B$ are two $N$-vectors defined as

$$\hat{d}_{KLD}^A = \left[ D(\mathbf{a}_1^{[1]} \| \mathbf{a}_1^{[2]}), D(\mathbf{a}_2^{[1]} \| \mathbf{a}_2^{[2]}), \cdots, D(\mathbf{a}_N^{[1]} \| \mathbf{a}_N^{[2]}) \right]^T \tag{6.7}$$

$$\hat{d}_{KLD}^B = \left[ D(b_1^{[1]} \| b_1^{[2]}), D(b_2^{[1]} \| b_2^{[2]}), \cdots, D(b_N^{[1]} \| b_N^{[2]}) \right]^T \tag{6.8}$$

In order to compute the inverse of $(I - \hat{A}_1)$, we use the result from [107], which offers an elegant and concise formula for the inverse of an $n \times n$ tridiagonal matrix $M_T$

$$M_T = \begin{pmatrix} a_1 & b_1 & & & \\ c_1 & a_2 & b_2 & & \\ & c_2 & \ddots & \ddots & \\ & & \ddots & \ddots & b_{n-1} \\ & & & c_{n-1} & a_n \end{pmatrix} \tag{6.9}$$

The $(i,j)$ element of the inverse of $M_T$ is given by

$$(M_T^{-1})_{ij} = \begin{cases} (-1)^{i+j} b_i \cdots b_{j-1} \theta_{i-1} \phi_{j+1}/\theta_n & \text{if } i \leq j \\ (-1)^{i+j} c_j \cdots c_{i-1} \theta_{j-1} \phi_{i+1}/\theta_n & \text{if } i > j \end{cases} \tag{6.10}$$

where

$$\theta_i = a_i \theta_{i-1} - b_{i-1} c_{i-1} \theta_{i-2}, \text{ for } i = 2, \cdots, n \tag{6.11}$$

with initial conditions $\theta_0 = 1$ and $\theta_1 = a_1$, and

$$\phi_i = a_i \phi_{i+1} - b_i c_i \phi_{i+2}, \text{ for } i = n-1, \cdots, 1 \tag{6.12}$$

with initial conditions $\phi_{n+1} = 1$ and $\phi_n = a_n$. The above formula can be specialized for an upper bidiagonal matrix $M_B$

$$M_B = \begin{pmatrix} a_1 & b_1 & & & \\ & a_2 & b_2 & & \\ & & \ddots & \ddots & \\ & & & \ddots & b_{n-1} \\ & & & & a_n \end{pmatrix} \tag{6.13}$$

which turns out to be an even simpler expression

$$(M_B^{-1})_{ij} = \begin{cases} \frac{1}{a_i} & \text{if } i = j \\ (-1)^{i+j} \frac{b_i \cdots b_{j-1}}{a_i \cdots a_j} & \text{if } i < j \\ 0 & \text{if } i > j \end{cases} \tag{6.14}$$

Since $\hat{A}_1$ is an upper bidiagonal matrix, $I - \hat{A}_1$ is also an upper bidiagonal matrix

$$I - \hat{A}_1 = \begin{pmatrix} 1 - a_{11}^{[1]} & -(1 - a_{11}^{[1]}) & 0 & \cdots & 0 \\ 0 & 1 - a_{22}^{[1]} & -(1 - a_{22}^{[1]}) & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -(1 - a_{N-1,N-1}^{[1]}) \\ 0 & 0 & 0 & \cdots & 1 - a_{NN}^{[1]} \end{pmatrix} \tag{6.15}$$

From Equations (6.14) and (6.15), we can easily identify the $(i,j)$ element of the inverse of $(I - A_1)$

$$\left((I - \hat{A}_1)^{-1}\right)_{ij} = \begin{cases} \frac{1-a_{ii}^{[1]}}{1-a_{jj}^{[1]}} & i \le j \\ 0 & i > j \end{cases} \tag{6.16}$$

The $N \times N$ matrix $(I - \hat{A}_1)^{-1}$ turns out to be upper triangular. We have

$$\hat{\boldsymbol{\pi}}_1^T (I - \hat{A}_1)^{-1} = \left[ \frac{1-a_{11}^{[1]}}{1-a_{11}^{[1]}}, \frac{1-a_{11}^{[1]}}{1-a_{22}^{[1]}}, \cdots, \frac{1-a_{11}^{[1]}}{1-a_{NN}^{[1]}} \right] \tag{6.17}$$

Thus, the KLDR upper bound in Equation (6.6) becomes

$$R(\lambda_1 \| \lambda_2) \le \sum_{i=1}^{N} \frac{1-a_{11}^{[1]}}{1-a_{ii}^{[1]}} \left( D(\mathbf{a}_i^{[1]} \| \mathbf{a}_i^{[2]}) + D(b_i^{[1]} \| b_i^{[2]}) \right) \tag{6.18}$$

As in the case of ergodic HMMs, here we adopt the symmetric version of the KLDR for left-to-right HMMs

$$R_s(\lambda_1 \| \lambda_2) = \frac{1}{2} \left( R(\lambda_1 \| \lambda_2) + R(\lambda_2 \| \lambda_1) \right) \tag{6.19}$$

which, straightforwardly, has an upper bound

$$\begin{aligned} R_s(\lambda_1 \| \lambda_2) \le\ & \frac{1}{2} \sum_{i=1}^{N} \frac{1-a_{11}^{[1]}}{1-a_{ii}^{[1]}} \left( D(\mathbf{a}_i^{[1]} \| \mathbf{a}_i^{[2]}) + D(b_i^{[1]} \| b_i^{[2]}) \right) \\ & + \frac{1}{2} \sum_{i=1}^{N} \frac{1-a_{11}^{[2]}}{1-a_{ii}^{[2]}} \left( D(\mathbf{a}_i^{[2]} \| \mathbf{a}_i^{[1]}) + D(b_i^{[2]} \| b_i^{[1]}) \right) \end{aligned} \tag{6.20}$$

When the parameters of the adapted HMMs $\lambda_1$ and $\lambda_2$ are adapted from the parameters of the UBM $\lambda$, if we impose a constraint that the state transition probability distribution $A$ be not adapted, that is, $A_1 = A_2 = A$, the upper bound of the symmetric KLDR in Equation (6.20) would become

$$R_s(\lambda_1 \| \lambda_2) \le \sum_{i=1}^{N} \frac{1-a_{11}}{1-a_{ii}} D_s(b_i^{[1]} \| b_i^{[2]}) \tag{6.21}$$

where

$$D_s(b_i^{[1]} \| b_i^{[2]}) = \frac{1}{2} \left( D(b_i^{[1]} \| b_i^{[2]}) + D(b_i^{[2]} \| b_i^{[1]}) \right) \tag{6.22}$$

is the symmetric KLD between two Gaussian mixture state observation PDFs $b_i^{[1]}$ and $b_i^{[2]}$.

If the state transition probability distribution $A$ is adapted, that is, $A_1 \neq A$ and $A_1 \neq A$, the R.H.S. of Equation (6.20) would become

$$\text{R.H.S.}(6.20) \approx \sum_{i=1}^{N} \frac{1-a_{11}}{1-a_{ii}} \left( D_s(\mathbf{a}_i^{[1]} \| \mathbf{a}_i^{[2]}) + D_s(b_i^{[1]} \| b_i^{[2]}) \right) \tag{6.23}$$

where

$$D_s(\mathbf{a}_i^{[1]} \| \mathbf{a}_i^{[2]}) = \frac{1}{2} \left( D(\mathbf{a}_i^{[1]} \| \mathbf{a}_i^{[2]}) + D(\mathbf{a}_i^{[2]} \| \mathbf{a}_i^{[1]}) \right) \tag{6.24}$$

is the symmetric KLD between two discrete state transition PMFs $\mathbf{a}_i^{[1]}$ and $\mathbf{a}_i^{[2]}$ (i.e., the $i^{th}$ rows of the stochastic matrices $A_1$ and $A_2$, respectively).

As in the case of ergodic HMMs, in the following, we will discuss four cases, corresponding to the different levels of adaptation from the UBM.

### 6.1.1 Case 1: Adaptation of Means Only

Suppose during the model adaptation process only the component mean vectors of the Gaussian mixture state observation PDFs of the HMMs $\lambda_1$ and $\lambda_2$ are adapted from the UBM $\lambda$. That is, $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = \boldsymbol{\pi}$, $A_1 = A_2 = A$, and $\{\{c_{ik}^{[1]}\}_{k=1}^M, \{\Sigma_{ik}^{[1]}\}_{k=1}^M\}_{i=1}^N = \{\{c_{ik}^{[2]}\}_{k=1}^M, \{\Sigma_{ik}^{[2]}\}_{k=1}^M\}_{i=1}^N = \{\{c_{ik}\}_{k=1}^M, \{\Sigma_{ik}\}_{k=1}^M\}_{i=1}^N$. In this case, it is easy to verify that

$$D_s(b_i^{[1]} \| b_i^{[2]}) \leq \sum_{k=1}^{M} c_{ik} \frac{1}{2} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T \Sigma_{ik}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \tag{6.25}$$

The upper bound of the symmetric KLDR in Equation (6.21) becomes

$$\begin{aligned} &R_s(\lambda_1 \| \lambda_2) \\ &\quad \leq \sum_{i=1}^{N} \frac{1-a_{11}}{1-a_{ii}} \sum_{k=1}^{M} c_{ik} \frac{1}{2} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T \Sigma_{ik}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \\ &\quad = \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \left\| \sqrt{\frac{1-a_{11}}{1-a_{ii}} c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[1]} - \sqrt{\frac{1-a_{11}}{1-a_{ii}} c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[2]} \right\|^2 \end{aligned} \tag{6.26}$$

Equation (6.26) indicates that if we approximate $R_s(\lambda_1 \| \lambda_2)$ by its upper bound in Equation (6.26) and form the augmented vectors

$$\mathbf{s}_p = \left[ \sqrt{\frac{1 - a_{11}}{1 - a_{ii}}} c_{ik} \Sigma_{ik}^{-1} \mu_{ik}^{[p]} \right]_{i=1k=1}^{N \quad M} \tag{6.27}$$

where $p = 1, 2$, the squared Euclidean distance between $\mathbf{s}_1$ and $\mathbf{s}_2$ is equivalent to the symmetric KLDR between the two corresponding adapted left-to-right HMMs $\lambda_1$ and $\lambda_2$ (up to a constant scale $\frac{1}{2}$).

### 6.1.2 Case 2: Adaptation of Means and Variances

Suppose during the model adaptation process both the component mean vectors and covariance matrices of the Gaussian mixture state observation PDFs of the HMMs $\lambda_1$ and $\lambda_2$ are adapted from the UBM $\lambda$. That is, $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = \boldsymbol{\pi}$, $A_1 = A_2 = A$, and $\{\{c_{ik}^{[1]}\}_{k=1}^M\}_{i=1}^N = \{\{c_{ik}^{[2]}\}_{k=1}^M\}_{i=1}^N = \{\{c_{ik}\}_{k=1}^M\}_{i=1}^N$. As in the case of ergodic HMMs, we further assume that all covariance matrices of the Gaussian mixture state observation PDFs are diagonal. In this case,

$$\begin{aligned} D_s(b_i^{[1]} \| b_i^{[2]}) &\leq \sum_{k=1}^M c_{ik} \frac{1}{2} (\sigma_{ik}^{2\ [1]} - \sigma_{ik}^{2\ [2]})^T \frac{1}{2} \Sigma_{ik}^{-2} (\sigma_{ik}^{2\ [1]} - \sigma_{ik}^{2\ [2]}) \\ &\quad + \sum_{k=1}^M c_{ik} \frac{1}{2} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T (\frac{1}{2} \Sigma_{ik}^{[1]^{-1}} + \frac{1}{2} \Sigma_{ik}^{[2]^{-1}}) (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \\ &\approx \frac{1}{2} \sum_{k=1}^M c_{ik} (\sigma_{ik}^{2\ [1]} - \sigma_{ik}^{2\ [2]})^T \frac{1}{2} \Sigma_{ik}^{-2} (\sigma_{ik}^{2\ [1]} - \sigma_{ik}^{2\ [2]}) \\ &\quad + \frac{1}{2} \sum_{k=1}^M c_{ik} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T \Sigma_{ik}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \end{aligned} \tag{6.28}$$

The upper bound of the symmetric KLDR in Equation (6.21) becomes

$$\begin{aligned}
R_s(\lambda_1 \| \lambda_2) &\leq \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\frac{1-a_{11}}{1-a_{ii}}c_{ik}(\sigma_{ik}^{2\ [1]} - \sigma_{ik}^{2\ [2]})^T\frac{1}{2}\Sigma_{ik}^{-2}(\sigma_{ik}^{2\ [1]} - \sigma_{ik}^{2\ [2]}) + \\
&\quad \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\frac{1-a_{11}}{1-a_{ii}}c_{ik}(\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T\Sigma_{ik}^{-1}(\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \\
&= \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\left\|\sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [1]} - \sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [2]}\right\|^2 \\
&\quad + \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\left\|\sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[1]} - \sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[2]}\right\|^2 \quad (6.29)
\end{aligned}$$

Equation (6.29) indicates that if we approximate $R_s(\lambda_1 \| \lambda_2)$ by its upper bound in Equation (6.29) and form the augmented vectors

$$\mathbf{s}_p = \left[\sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[p]}; \sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}/2}\Sigma_{ik}^{-1}\sigma_{ik}^{2\ [p]}\right]_{i=1k=1}^{N\quad M} \qquad (6.30)$$

where $p = 1, 2$, the squared Euclidean distance between $\mathbf{s}_1$ and $\mathbf{s}_2$ is equivalent to the symmetric KLDR between $\lambda_1$ and $\lambda_2$ (up to a constant scale $\frac{1}{2}$).

### 6.1.3 Case 3: Adaptation of Means, Variances and Mixture Weights

Suppose during the model adaptation process the component mean vectors, covariance matrices, and mixture weights of the Gaussian mixture state observation PDFs of the HMMs $\lambda_1$ and $\lambda_2$ are all adapted from the UBM $\lambda$.

That is, $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = \boldsymbol{\pi}$, and $A_1 = A_2 = A$. In this case,

$$\begin{aligned}
D_s(b_i^{[1]} \| b_i^{[2]}) &\leq D_s(\mathbf{c}_i^{[1]} \| \mathbf{c}_i^{[2]}) \\
&\quad + \sum_{k=1}^{M} c_{ik} \frac{1}{2} (\sigma_{ik}^{2\,[1]} - \sigma_{ik}^{2\,[2]})^T \frac{1}{2} \Sigma_{ik}^{-2} (\sigma_{ik}^{2\,[1]} - \sigma_{ik}^{2\,[2]}) \\
&\quad + \sum_{k=1}^{M} c_{ik} \frac{1}{2} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T (\frac{1}{2} \Sigma_{ik}^{[1]^{-1}} + \frac{1}{2} \Sigma_{ik}^{[2]^{-1}}) (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \\
&\approx \frac{1}{2} \sum_{k=1}^{M} (c_{ik}^{[1]} - c_{ik}^{[2]}) \log \frac{c_{ik}^{[1]}}{c_{ik}^{[2]}} \\
&\quad + \frac{1}{2} \sum_{k=1}^{M} c_{ik} (\sigma_{ik}^{2\,[1]} - \sigma_{ik}^{2\,[2]})^T \frac{1}{2} \Sigma_{ik}^{-2} (\sigma_{ik}^{2\,[1]} - \sigma_{ik}^{2\,[2]}) \\
&\quad + \frac{1}{2} \sum_{k=1}^{M} c_{ik} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T \Sigma_{ik}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \qquad (6.31)
\end{aligned}$$

where from Chapter 5, we know

$$\sum_{k=1}^{M} (c_{ik}^{[1]} - c_{ik}^{[2]}) \log \frac{c_{ik}^{[1]}}{c_{ik}^{[2]}} \approx \sum_{k=1}^{M} \frac{(c_{ik}^{[1]} - c_{ik}^{[2]})^2}{c_{ik}} \qquad (6.32)$$

The upper bound of the symmetric KLDR in Equation (6.21) becomes

$$\begin{aligned}
R_s(\lambda_1 \| \lambda_2) &\leq \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \frac{1 - a_{11}}{1 - a_{ii}} \frac{(c_{ik}^{[1]} - c_{ik}^{[2]})^2}{c_{ik}} \\
&\quad + \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \frac{1 - a_{11}}{1 - a_{ii}} c_{ik} (\sigma_{ik}^{2\,[1]} - \sigma_{ik}^{2\,[2]})^T \frac{1}{2} \Sigma_{ik}^{-2} (\sigma_{ik}^{2\,[1]} - \sigma_{ik}^{2\,[2]}) \\
&\quad + \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \frac{1 - a_{11}}{1 - a_{ii}} c_{ik} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]})^T \Sigma_{ik}^{-1} (\mu_{ik}^{[1]} - \mu_{ik}^{[2]}) \\
&\approx \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \left( \sqrt{\frac{1 - a_{11}}{(1 - a_{ii}) c_{ik}}} c_{ik}^{[1]} - \sqrt{\frac{1 - a_{11}}{(1 - a_{ii}) c_{ik}}} c_{ik}^{[2]} \right)^2 \\
&\quad + \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \left\| \sqrt{\frac{1 - a_{11}}{1 - a_{ii}} c_{ik}/2} \Sigma_{ik}^{-1} \sigma_{ik}^{2\,[1]} - \sqrt{\frac{1 - a_{11}}{1 - a_{ii}} c_{ik}/2} \Sigma_{ik}^{-1} \sigma_{ik}^{2\,[2]} \right\|^2 \\
&\quad + \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{M} \left\| \sqrt{\frac{1 - a_{11}}{1 - a_{ii}} c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[1]} - \sqrt{\frac{1 - a_{11}}{1 - a_{ii}} c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[2]} \right\|^2 \qquad (6.33)
\end{aligned}$$

Equation (6.33) indicates that if we approximate $R_s(\lambda_1\|\lambda_2)$ by its upper bound in Equation (6.33) and form the augmented vectors

$$\mathbf{s}_p = \left[\sqrt{\frac{1-a_{11}}{(1-a_{ii})c_{ik}}}c_{ik}^{[p]}; \sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[p]}; \sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}/2\Sigma_{ik}^{-1}}\sigma_{ik}^{2\ [p]}\right]_{i=1k=1}^{N\quad M} \tag{6.34}$$

where $p = 1, 2$, the squared Euclidean distance between $\mathbf{s}_1$ and $\mathbf{s}_2$ is equivalent to the symmetric KLDR between $\lambda_1$ and $\lambda_2$ (up to a constant scale $\frac{1}{2}$).

### 6.1.4 Case 4: Full Adaptation

Suppose during the model adaptation process all parameters of the HMMs $\lambda_1$ and $\lambda_2$ are adapted from the UBM $\lambda$ except the initial state probability distributions, which are identical for left-to-right HMMs by definition. That is, $\boldsymbol{\pi}_1 = \boldsymbol{\pi}_2 = \boldsymbol{\pi}$. In this case, the upper bound of the symmetric KLDR in Equation (6.23) becomes

$$\begin{aligned}
R_s(\lambda_1\|\lambda_2) \leq\ & \frac{1}{2}\sum_{i=1}^{N}\sum_{l=1}^{N}\frac{1-a_{11}}{1-a_{ii}}\frac{(a_{il}^{[1]}-a_{il}^{[2]})^2}{a_{il}} \\
&+\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\frac{1-a_{11}}{1-a_{ii}}\frac{(c_{ik}^{[1]}-c_{ik}^{[2]})^2}{c_{ik}} \\
&+\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\frac{1-a_{11}}{1-a_{ii}}c_{ik}(\sigma_{ik}^{2\ [1]}-\sigma_{ik}^{2\ [2]})^T\frac{1}{2}\Sigma_{ik}^{-2}(\sigma_{ik}^{2\ [1]}-\sigma_{ik}^{2\ [2]}) \\
&+\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\frac{1-a_{11}}{1-a_{ii}}c_{ik}(\mu_{ik}^{[1]}-\mu_{ik}^{[2]})^T\Sigma_{ik}^{-1}(\mu_{ik}^{[1]}-\mu_{ik}^{[2]}) \\
\approx\ & \frac{1}{2}\sum_{i=1}^{N}\sum_{l=1}^{N}\left(\sqrt{\frac{1-a_{11}}{(1-a_{ii})a_{il}}}a_{il}^{[1]}-\sqrt{\frac{1-a_{11}}{(1-a_{ii})a_{il}}}a_{il}^{[2]}\right)^2 \\
&+\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\left(\sqrt{\frac{1-a_{11}}{(1-a_{ii})c_{ik}}}c_{ik}^{[1]}-\sqrt{\frac{1-a_{11}}{(1-a_{ii})c_{ik}}}c_{ik}^{[2]}\right)^2 \\
&+\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\left\|\sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}/2\Sigma_{ik}^{-1}}\sigma_{ik}^{2\ [1]}-\sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}/2\Sigma_{ik}^{-1}}\sigma_{ik}^{2\ [2]}\right\|^2 \\
&+\frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{M}\left\|\sqrt{\frac{1-a_{11}}{(1-a_{ii})}c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[1]}-\sqrt{\frac{1-a_{11}}{1-a_{ii}}c_{ik}\Sigma_{ik}^{-1}}\mu_{ik}^{[2]}\right\|^2
\end{aligned} \tag{6.35}$$

Equation (6.35) indicates that if we approximate $R_s(\lambda_1\|\lambda_2)$ by its upper bound in Equation (6.35) and form the augmented vectors

$$\mathbf{s}_p = \left[ \left[ \sqrt{\frac{1-a_{11}}{(1-a_{ii})a_{il}}} a_{il}^{[p]} \right]_{l=1}^{N} ; \left[ \sqrt{\frac{1-a_{11}}{(1-a_{ii})c_{ik}}} c_{ik}^{[p]} \right]_{k=1}^{M} ; \right.$$
$$\left. \left[ \sqrt{\frac{1-a_{11}}{1-a_{ii}} c_{ik} \Sigma_{ik}^{-1}} \mu_{ik}^{[p]} \right]_{k=1}^{M} ; \left[ \sqrt{\frac{1-a_{11}}{1-a_{ii}} c_{ik}/2 \Sigma_{ik}^{-1}} \sigma_{ik}^{2\ [p]} \right]_{k=1}^{M} ; \right]_{i=1}^{N} \quad (6.36)$$

where $p = 1, 2$, the squared Euclidean distance between $\mathbf{s}_1$ and $\mathbf{s}_2$ is equivalent to the symmetric KLDR between $\lambda_1$ and $\lambda_2$ (up to a constant scale $\frac{1}{2}$).

## 6.2 Summary

In this chapter, we propose a novel one-vector representation of stochastic signals for pattern recognition based on adapted left-to-right HMMs. This one-vector representation of stochastic signals is complimentary to the one-vector representation of stochastic signals proposed in Chapter 5, and turns out to be a potentially more appropriate one-vector representation for stochastic signals such as speech and video, which are sequential in nature or which have a clear temporal dimension of which the dynamics need to be captured.

# CHAPTER 7

# A GENERAL FRAMEWORK FOR ONE-VECTOR REPRESENTATIONS OF STOCHASTIC SIGNALS

In Chapter 5, we propose a novel one-vector representation of stochastic signals based on adapted ergodic HMMs. Similarly, in Chapter 6, we propose a novel one-vector representation of stochastic signals based on adapted left-to-right HMMs. The one-vector representations proposed in Chapter 5 and Chapter 6 are based on two particular types of powerful statistical models, namely ergodic HMMs and left-to-right HMMs, respectively. One might have noticed that in the two proposed one-vector representations, except for the different choices of the type of the underlying statistical models, the formation processes of the one-vector representations are highly similar, and in fact, almost identical. First, an ergodic (or left-to-right) HMM is learned (through the adaptation of a UBM) to represent the joint probability distribution of the feature vectors extracted from an input stochastic signal. Then, the model parameters of the learned HMM are nonlinearly transformed to produce the one-vector representation of the input stochastic signal based on the criterion that the Euclidean distance between any two representational vectors is (approximately) equivalent to the Kullback-Leibler divergence rate between the two corresponding HMMs. Such a two-stage one-vector representation formation process can be extended and generalized to form a general framework for one-vector representations of stochastic signals.

In this chapter, we propose a general framework for one-vector representations of stochastic signals for pattern recognition. The proposed general framework is schematically illustrated in Figure 7.1. As shown, the general framework consists of two successive stages, namely statistical model learning and nonlinear transform. In the first stage of the framework, namely statistical model learning, a proper statistical model of some type is learned based on an input stochastic signal. The statistical model should be carefully and cleverly chosen so that it best characterizes the joint probability distribution of the feature vectors extracted from the input stochastic signal. In the

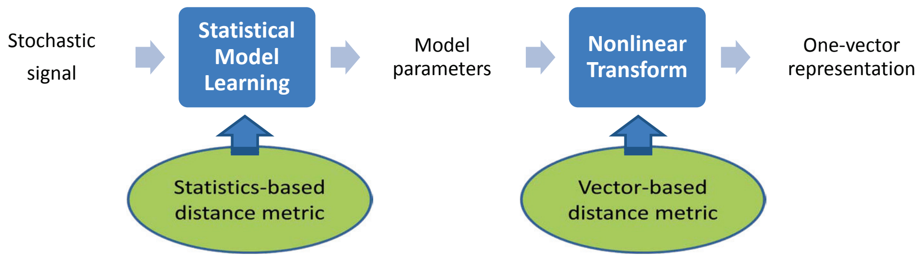
Stochastic signal
Statistical Model Learning
Model parameters
Nonlinear Transform
One-vector representation
Statistics-based distance metric
Vector-based distance metric

Figure 7.1: A general framework for one-vector representations of stochastic signals for pattern recognition.

second stage of the framework, namely nonlinear transform, the parameters of the statistical model learned previously in the first stage are nonlinearly transformed into a one-vector representation, where the nonlinear transform may be sought mathematically based on the criterion that a task-motivating vector-based distance metric (e.g. the Euclidean distance) between any two representational vectors is (approximately) equivalent to a certain statistical model based distance measure (e.g. the Kullback-Leibler divergence) between the two corresponding underlying statistical models. Such a general framework for one-vector representations of stochastic signals can serve as a unified and principled guide for constructing "the best" one-vector representations of stochastic signals of various types and for various pattern recognition tasks. The general framework may also be demonstrated by Equations (7.1) and (7.2), where, in Equation (7.1), the joint probability distribution of the feature vectors in the feature vector set of a stochastic signal $s$ is encoded by the parameters of an underlying statistical model $\lambda$, and in Equation (7.2), the parameters of the underlying statistical model $\lambda$ are nonlinearly transformed to construct a one-vector representation which compactly summarizes the joint probability distribution of the feature vectors.

$$F(s) = \{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n\} \rightarrow \lambda \tag{7.1}$$

$$\lambda \rightarrow \mathbf{s} = \mathcal{T}(\lambda) \tag{7.2}$$

Although in general HMMs are very powerful statistical models which may be appropriate for use to well characterize the joint probability distributions of the feature vectors for a wide range of stochastic signals such as speech, audio, images, video, and so on, they may not be the best statistical models for all kinds of stochastic signals in the physical world. For example, the sequential nature of the conventional one-dimensional HMMs would make them seemingly not quite ideal for use to model stochastic signals which are obviously non-sequential in nature, such as two-dimensional images. In Chapter 5, we claim that ergodic HMMs are capable of modeling stochastic signals which are non-sequential in nature, and experimentally show that the one-vector representation based on adapted ergodic HMMs can be an effective one-vector representation of images. The evidence lies in the fact that ergodic HMMs can implicitly perform unsupervised segmentation of the stochastic signals and hopefully recover the structures of the signals that were destroyed when we intentionally feed the feature vectors to the HMMs in a sequential manner, and in the fact that some important information regarding the structures of the stochastic signals is encoded in the statistical dependence among the feature vectors which is captured and described by the ergodic HMMs. However, there are certainly other types of statistical models which might turn out to be better than ergodic HMMs to do the same job. For example, pseudo 2D HMMs [108, 109], embedded 2D HMMs [110, 111], embedded Bayesian networks [112], true or full 2DHMMs [113, 114, 115, 116] as well as hidden Markov random fields (HMRFs) [117] are considered by a number of researchers in the image processing and computer vision community as potentially better statistical models for representing two-dimensional images than the conventional one-dimensional sequential HMMs. Therefore, it is highly advisable and desirable that, when we construct a one-vector representation of stochastic signals following the approach guided by the proposed general framework, the type of the underlying statistical models in the first stage must be carefully and cleverly chosen according to the nature of the stochastic signals. Once the type of the underlying statistical models is chosen, there might be other issues such as those structure- or signal-related parameters that need to be determined. For example, if HMMs are chosen to be the underlying statistical models, according to the nature of the stochastic signals, the topology of the underlying HMMs can be either ergodic or left-to-right, and the state observation probability distributions of the HMMs can be represented by either continuous PDFs or discrete PMFs. Figure 7.2 hierarchically displays an incomplete list of various underlying statistical models which may be used for constructing one-vector representations of stochastic signals under the guidance of the general framework.

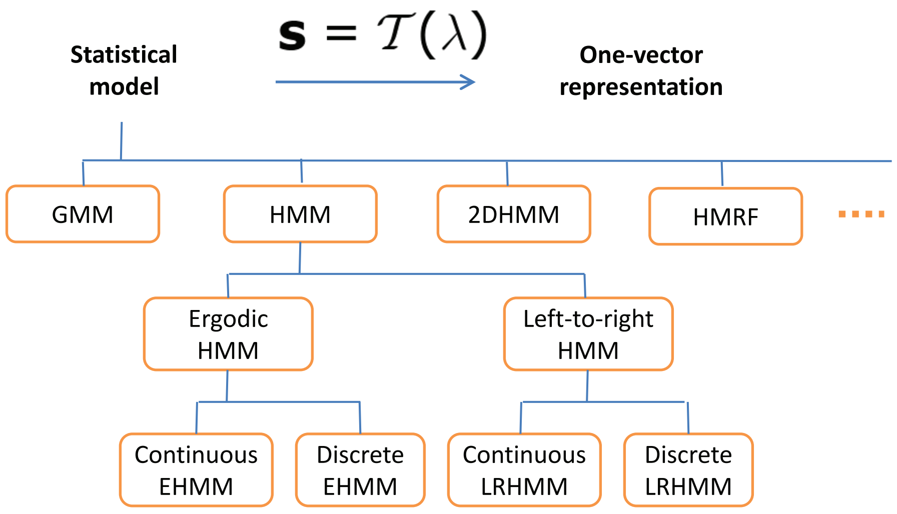

Figure 7.2: An incomplete list of various underlying statistical models which may be used for constructing one-vector representations of stochastic signals under the guidance of the general framework.

The proposed general framework for one-vector representations of stochastic signals can serve as a unified and principled guide for constructing "the best" one-vector representations of stochastic signals of various types and for various pattern recognition tasks. Based on different types of underlying statistical models carefully and cleverly chosen to best fit the nature of the stochastic signals, "the best" one-vector representations of the stochastic signals may be constructed by a nonlinear transform of the parameters of the underlying statistical models learned from the stochastic signals, where the nonlinear transform may be mathematically derived from a properly chosen distance measure between two underlying statistical models that has an elegant root in the Kullback-Leibler (KL) theory [118, 92]. In the following, we close this chapter by summarizing the design choices that are involved in constructing "the best" one-vector representations of stochastic signals following the guidance of the proposed general framework for one-vector representations of stochastic signals for pattern recognition.

- *The choice of underlying statistical model.* In the general framework, the type of the underlying statistical model should be carefully and cleverly chosen so that it best characterizes the joint probability distributions of the feature vectors extracted from the stochastic signals.

The topology of the underlying statistical model should be determined based on the nature of the stochastic signals, conforming to the physical or imaginary generative process that would have generated the observations of the stochastic signals. The type of the observation probability distributions (continuous or discrete) should be determined together by the type of the feature vectors and the scales of the problem and data set.

- *The choice of learning and adaptation strategies.* In the general framework, the learning and adaptation strategies for the underlying statistical models should be decided according to the specific statistical models and the specific the pattern recognition tasks. That is, different learning and adaptation techniques should be adopted to learn the parameters of the underlying statistical models for different purposes. In our HMM-based one-vector representations, we have employed maximum likelihood estimation techniques to learn the model parameters of the underlying ergodic or left-to-right HMMs. However, how to choose the ideal learning and adaptation techniques can be closely related to the specific pattern recognition tasks that will subsequently make use of the one-vector representation. For example, if the pattern recognition task is solely to discriminatively classify the stochastic signals into different disjoint groups, then a discriminative learning technique [119] used during the formation process of the one-vector representation may lead to a performance increase in the final classification results.

- *The choice of distance metrics.* In the general framework, two kinds of distance metrics must be determined before we can proceed to derive the formation formulas for the one-vector representation. One is the vector-based distance metric that we use to measure the distance between two resulting vectors in our representational space. The other is the statistical model based distance measure that we use to measure the dissimilarity between two probability distributions. In our proposed one-vector representations based on adapted ergodic HMMs and adapted left-to-right HMMs respectively, we have used the Euclidean distance metric for the vector-based distance metric between two representational vectors and the K-L divergence rate for the statistical model based distance measure between two ergodic or left-to-right

HMMs. However, there are other choices of vector-based distance metrics, which may be appropriate for a different purpose. For example, the cosine distance metric should be chosen if the goal is to form a one-vector representation in the cosine space rather than in the Euclidean space. Likewise, there are other choices of statistical model based distance measures, too. For example, the statistical decision theory motivated cross log likelihood ratio (CLLR) [54] may sometimes be a better choice than the information theory motivated K-L divergence (KLD) for measuring the dissimilarity between two probability distributions.

# CHAPTER 8

# CONCLUSIONS

In this dissertation, we mainly study one-vector representations of stochastic signals for pattern recognition. Notably, we propose a novel one-vector representation of stochastic signals based on adapted ergodic hidden Markov models (EHMMs) and a novel one-vector representation of stochastic signals based on adapted left-to-right hidden Markov models (LRHMMs). These proposed one-vector representations of stochastic signals are aimed at overcoming the limitations, constraints, and weaknesses of the existing methods for constructing one-vector representations of stochastic signals, namely the holistic method, the bag of words method, and the GMM mean supervector representation. Specifically, the major contributions of this dissertation are highlighted as follows.

1. We propose the conceptually new idea of, and novel strategies for, semi-supervised speaker clustering, where semi-supervision here refers to the use of our prior knowledge of speakers in general to assist the unsupervised speaker clustering process. By means of an independent training data set, we encode the prior knowledge at the various stages of the speaker clustering pipeline via (1) learning a speaker-discriminative acoustic feature transformation, (2) learning a universal speaker prior model, and (3) learning a discriminative speaker subspace, or equivalently, a speaker-discriminative distance metric. We discover the directional scattering property of the GMM mean supervector representation of utterances in the high-dimensional space, and advocate the use of the cosine distance metric instead of the Euclidean distance metric for speaker clustering in the GMM mean supervector space. We propose to perform discriminant analysis based on the cosine distance metric, which leads to a novel distance metric learning algorithm – linear spherical discriminant analysis (LSDA). We show that the proposed LSDA formulation can be systematically solved within the elegant "graph em-

bedding" general dimensionality reduction framework. Our speaker clustering experiments on the GALE database clearly indicate that (1) our speaker clustering methods based on the GMM mean supervector representation and vector-based distance metrics outperform traditional speaker clustering methods based on the bag of acoustic features representation and likelihood-based distance metrics, (2) our advocated use of the cosine distance metric yields consistent increases in the speaker clustering performance as compared to the commonly used Euclidean distance metric, (3) our semi-supervised speaker clustering concept and strategies significantly improve the speaker clustering performance over the baselines, and (4) our proposed LSDA algorithm further leads to the state-of-the-art speaker clustering performance. Note that this contribution is seemingly stand-alone from the rest of the dissertation. However, in addition to being of great value by itself, this contribution not only serves as an important motivation for the study of one-vector representations of stochastic signals for pattern recognition in this dissertation, but also helps to illustrate the essential concepts and many benefits of one-vector representations of stochastic signals (such as optimal distance metric learning from the data).

2. We propose a new maximum likelihood learning algorithm for hidden Markov models (HMMs), which we refer to as the boosting Baum-Welch algorithm. In the proposed boosting Baum-Welch algorithm, we formulate the HMM learning problem as an incremental optimization procedure which performs a sequential gradient descent search on a loss functional for a good fit in an inner product function space. Such a sequential optimization procedure may be used to provide a theoretical interpretation for the boosting algorithm from a very different perspective. Hence the name of the boosting Baum-Welch algorithm. The boosting Baum-Welch algorithm can serve as an alternative maximum likelihood learning algorithm for HMMs to the traditional Baum-Welch or expectation-maximization (EM) algorithm, and a preferred method for use in situations where there is insufficient training data available. Compared to the traditional Baum-Welch or EM algorithm, the boosting Baum-Welch algorithm is less susceptible to the over-fitting problem (known as a general property of maximum likelihood estimation tech-

niques) in that the boosting Baum-Welch algorithm has a tendency to produce a "large margin" effect. Since HMMs form the basis of the one-vector representations of stochastic signals proposed in this dissertation, this contribution is relevant and important.

3. We propose a novel one-vector representation of stochastic signals based on adapted ergodic hidden Markov models (EHMMs) and a novel one-vector representation of stochastic signals based on adapted left-to-right hidden Markov models (LRHMMs). These one-vector representations of stochastic signals possess very attractive properties. First, the representation summarizes the probability distribution of the feature vectors in the feature vector set compactly and accurately and allows the statistical dependence among the feature vectors to be modeled with a systematic underlying structure of first-order Markov chain. Second, the representation performs unsupervised segmentation of the stochastic signals implicitly to reveal the local structures of the signals and to allow for localized, segment-wise comparison of the signals. Third, the representation is in a one-vector form ready for either supervised, semi-supervised or unsupervised distance metric learning from the data to further reenforce its discriminatory power for classification. In addition to the above advantages, the representation is rather generic in nature and may be used with various types of stochastic signals (e.g. image, video, speech, etc.) and applied to a broad range of pattern recognition tasks (e.g. classification, regression, etc.). It does not require the signals to be of the same size, nor does it require the alignment of the signals. In addition, it is supposed to be robust to partial occlusions or corruption in the signals.

4. We propose a general framework for one-vector representations of stochastic signals for pattern recognition, of which the proposed one-vector representation based on adapted ergodic HMMs and one-vector representation based on adapted left-to-right HMMs are two special cases. The general framework claims that, based on different types of underlying statistical models carefully and cleverly chosen to best fit the nature of the stochastic signals, "the best" one-vector representations of the stochastic signals may be constructed by a nonlinear transformation of the parameters of the underlying statistical models which are

learned from the stochastic signals, where the nonlinear transformation may be mathematically derived from a properly chosen distance measure between two statistical models that has an elegant root in the Kullback-Leibler (KL) theory. The general framework can serve as a unified and principled guide for constructing "the best" one-vector representations of stochastic signals of various types and for various pattern recognition tasks.

Future research following in the path of this dissertation may be focused on the following aspects:

1. Although the boosting Baum-Welch algorithm for HMM learning proposed in Chapter 4 is based on the maximum likelihood criterion, it is possible to extend it with discriminative training criteria, the incorporation of which has been known in the speech recognition community to boost the speech recognition performance significantly.

2. It is desirable to derive closed-form one-vector representations of stochastic signals based on several concrete types of underlying statistical models, for instance, 2DHMMs, HMRFs, etc., and based on different strategies for training and adapting the underlying statistical models, as well as to investigate the most appropriate stochastic signals and the most appropriate application domains that the derived one-vector representations can fit into.

# REFERENCES

[1] C. Bishop, *Pattern Recognition and Machine Learning.* Secaucus, NJ: Springer-Verlag, 2006.

[2] A. Papoulis, S. Pillai, and S. Unnikrishna, *Probability, Random Variables, and Stochastic Processes.* Columbus, OH: McGraw-Hill, 2002.

[3] R. Duda, P. Hart, and D. Stork, *Pattern Classification.* Hoboken, NJ: Wiley-Interscience, 2001.

[4] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* San Diego, CA: Academic Press, 1990.

[5] C. Bishop, *Neural Networks for Pattern Recognition.* Oxford, UK: Oxford University Press, 1995.

[6] V. Vapnik, *The Nature of Statistical Learning Theory.* New York, NY: Springer-Verlag, 2000.

[7] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *Computational Learning Theory*, P. Vitanyi, Ed. Berlin, Germany: Springer-Verlag, 1995, pp. 23–37.

[8] P. Mahalanobis, "On the generalized distance in statistics," in *Proceedings of the National Institute of Science*, vol. 2, India, 1936, pp. 49–55.

[9] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Maui, Hawaii, 1991, pp. 586–591.

[10] S. Li and A. Jain, *Handbook of Face Recognition.* Secaucus, NJ: Springer-Verlag, 2005.

[11] D. Lewis, "Naive (Bayes) at forty: The independence assumption in information retrieval," in *10th European Conference on Machine Learning*, Chemnitz, Germany, 1998, pp. 4–15.

[12] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[13] L. Fei-Fei, R. Fergus, and A. Torralba, "Recognizing and learning object categories," short course presented at IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, 2007. [Online]. Available: http://people.csail.mit.edu/torralba/shortCourseRLOC/index.html

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

[15] D. Reynolds, "A Gaussian mixture modeling approach to text-independent speaker identification," Ph.D. dissertation, Georgia Institute of Technology, Atlanta, GA, 1992.

[16] W. Campbell, "A covariance kernel for SVM language recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, NV, 2008, pp. 4141–4144.

[17] S. Yan, X. Zhou, M. Liu, M. Hasegawa-Johnson, and T. Huang, "Regression from patch-kernel," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Anchorage, Alaska, 2008, pp. 1–8.

[18] H. Tang, M. Hasegawa-Johnson, and T. Huang, "A novel vector representation of stochastic signals based on adapted ergodic HMMs," *IEEE Signal Processing Letters*, vol. 17, no. 8, pp. 715–718, 2010.

[19] H. Tang, M. Hasegawa-Johnson, and T. Huang, "Non-frontal view facial expression recognition based on ergodic hidden Markov model supervectors," in *IEEE International Conference on Multimedia & Expo*, Singapore, 2010, pp. 1202–1207.

[20] H. Tang, S. Chu, and T. Huang, "Spherical discriminant analysis in semi-supervised speaker clustering," in *The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies*, Boulder, CO, May 2009, pp. 57–60.

[21] S. Chu, H. Tang, and T. Huang, "Locality preserving speaker clustering," in *IEEE International Conference on Multimedia & Expo*, New York, NY, 2009, pp. 494–497.

[22] S. Chu, H. Tang, and T. Huang, "Fishervoice and semi-supervised speaker clustering," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, Taiwan, 2009, pp. 4089–4092.

[23] H. Tang, S. Chu, and T. Huang, "Generative model-based speaker clustering via mixture of von Mises-Fisher distributions," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, Taiwan, 2009, pp. 4101–4104.

[24] H. Tang, M. Hasegawa-Johnson, and T. Huang, "Toward robust learning of the Gaussian mixture state emission densities for hidden Markov models," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Dallas, TX, 2010, pp. 5242–5245.

[25] H. Tang, S. Chu, M. Hasegawa-Johnson, and T. Huang, "Emotion recognition from speech via boosted Gaussian mixture models," in *IEEE International Conference on Multimedia & Expo*, New York, NY, 2009, pp. 294–297.

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

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

[28] S. Levinson, *Mathematical Models for Speech Technology.* Hoboken, NJ: Wiley-Interscience, 2005.

[29] W. Zheng, H. Tang, Z. Lin, and T. Huang, "Emotion recognition from arbitrary view facial images," in *11th European Conference on Computer Vision*, Crete, Greece, 2010, pp. 490–503.

[30] W. Zheng, H. Tang, Z. Lin, and T. Huang, "A novel approach to expression recognition from non-frontal face images," in *12th IEEE International Conference on Computer Vision*, Kyoto, Japan, 2009, pp. 1901–1908.

[31] H. Jin, F. Kubala, and R. Schwartz, "Automatic speaker clustering," in *Proceedings of the DARPA Speech Recognition Workshop*, Chantilly, VA, 1997, pp. 108–111.

[32] S. Chen and P. Gopalakrishnan, "Clustering via the Bayesian information criterion with applications in speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Seattle, WA, May 1998, pp. 645–648.

[33] D. Reynolds, E. Singer, B. Carlson, G. O'Leary, J. McLaughlin, and M. Zissman, "Blind clustering of speech utterances based on speaker

and language characteristics," in *5th International Conference on Spoken Language Processing*, Sydney, Australia, 1998, pp. 3193–3196.

[34] A. Solomonoff, A. Mielke, M. Schmidt, and H. Gish, "Clustering speakers by their voices," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Orlando, FL, May 2002, pp. 757–760.

[35] R. Faltlhauser and G. Ruske, "Robust speaker clustering in eigenspace," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, Trento, Italy, 2001, pp. 57–60.

[36] W. Tsai, S. Cheng, and H. Wang, "Speaker clustering of speech utterances using a voice characteristic reference space," in *8th International Conference on Spoken Language Processing*, Jeju Island, Korea, 2004, pp. 2937–2940.

[37] M. Ben, M. Betser, F. Bimbot, and G. Gravier, "Speaker diarization using bottom-up clustering based on a parameter-derived distance between adapted GMMs," in *8th International Conference on Spoken Language Processing*, Jeju Island, Korea, 2004, pp. 1125–1128.

[38] C. Barras, X. Zhu, S. Meignier, and J. Gauvain, "Multistage speaker diarization of broadcast news," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1505–1512, 2006.

[39] S. Tranter and D. Reynolds, "An overview of automatic speaker diarization systems," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1557–1565, 2006.

[40] C. Wooters and M. Huijbregts, "The ICSI RT07s speaker diarization system," in *Multimodal Technologies for Perception of Humans*, Berlin, Germany, 2008, pp. 509–519.

[41] P. Kenny, "Bayesian analysis of speaker diarization with eigenvoice priors," CRIM, Montreal, Canada, Tech. Rep., May 2008.

[42] D. Reynolds, P. Kenny, and F. Castaldo, "A study of new approaches to speaker diarization," in *Interspeech*, Brighton, UK, 2009, pp. 1047–1050.

[43] D. Reynolds, T. Quatieri, and R. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1, pp. 19–41, 2000.

[44] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, CA, 1967, pp. 281–297.

[45] A. Jain and R. Dubes, *Algorithms for Clustering Data.* Englewood Cliffs, NJ: Prentice Hall, 1988.

[46] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition.* Englewood Cliffs, NJ: Prentice Hall, 1993.

[47] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[48] G. Fant, R. Jakobson, and C. van Schooneveld, *Acoustic Theory of Speech Production.* Berlin, Germany: Mouton de Gruyter, 1960.

[49] P. Jain and H. Hermansky, "Improved mean and variance normalization for robust speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Salt Lake City, Utah, May 2001, pp. 4015–4018.

[50] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[51] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 3, pp. 345–354, 2005.

[52] J. Gauvain and C. 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, 2004.

[53] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[54] X. Anguera, "Robust speaker diarization for meetings," Ph.D. dissertation, Universitat Politecnica de Catalunya, Barcelona, Spain, 2006.

[55] R. Kuhn, P. Nguyen, J. Junqua, L. Goldwasser, N. Niedzielski, S. Fincke, K. Field, and M. Contolini, "Eigenvoices for speaker adaptation," in *5th International Conference on Spoken Language Processing*, Sydney, Australia, 1998, pp. 1771–1774.

[56] X. He and P. Niyogi, "Locality preserving projections," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2003, pp. 1–8.

[57] X. He, D. Cai, S. Yan, and H. Zhang, "Neighborhood preserving embedding," in *10th IEEE International Conference on Computer Vision*, Beijing, China, 2005, pp. 1208–1213.

[58] Y. Ma, S. Lao, E. Takikawa, and M. Kawade, "Discriminant analysis in correlation similarity measure space," in *24th International Conference on Machine Learning*, Corvallis, OR, 2007, pp. 577–584.

[59] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, 2007.

[60] Y. Fu, S. Yan, and T. Huang, "Correlation metric for generalized feature extraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 12, pp. 2229–2235, 2008.

[61] D. Cai, X. He, Y. Hu, J. Han, and T. Huang, "Learning a spatially smooth subspace for face recognition," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Minneapolis, MN, 2007, pp. 1–7.

[62] I. Dhillon and D. Modha, "Concept decompositions for large sparse text data using clustering," *Machine Learning*, vol. 42, no. 1, pp. 143–175, 2001.

[63] A. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.

[64] S. Chu, H. Kuo, L. Mangu, Y. Liu, Y. Qin, Q. Shi, S. Zhang, and H. Aronowitz, "Recent advances in the IBM GALE mandarin transcription system," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Las Vegas, NV, 2008, pp. 4329–4332.

[65] P. Viola and W. Wells III, "Alignment by maximization of mutual information," *International Journal of Computer Vision*, vol. 24, no. 2, pp. 137–154, 1997.

[66] J. Moore, E. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, and B. Mobasher, "Web page categorization and feature selection using association rule and principal component clustering," Department of Computer Science, University of Minnesota, Minneapolis, MN, Tech. Rep. 9405380, 2001.

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

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

[69] C. Manning and H. Schutze, *Foundations of Statistical Natural Language Processing.* Cambridge, MA: The MIT Press, 1999.

[70] R. Nag, K. Wong, and F. Fallside, "Script recognition using hidden Markov models," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Tokyo, Japan, 1986, pp. 2071–2074.

[71] M. Evans, N. Hastings, and B. Peacock, "Statistical distributions," *Measurement Science and Technology*, vol. 12, p. 117, 2001.

[72] B. Everitt, *Finite Mixture Distributions.* Hoboken, NJ: Wiley-Interscience, 1981.

[73] G. Forney Jr., "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[74] L. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[75] N. Johnson, A. Kemp, and S. Kotz, *Univariate Discrete Distributions.* Hoboken, NJ: Wiley-Interscience, 2005.

[76] M. Russell and R. Moore, "Explicit modelling of state occupancy in hidden Markov models for automatic speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Tampa, FL, 1985, pp. 5–8.

[77] S. Levinson, "Continuously variable duration hidden Markov models for automatic speech recognition," *Computer Speech & Language*, vol. 1, no. 1, pp. 29–45, 1986.

[78] D. A. Reynolds, "Universal background models," in *Encyclopedia of Biometric Recognition*, S. Li, Ed. Secaucus, NJ: Springer-Verlag, 2008, pp. 31–36.

[79] Y. Freund, "Boosting a weak learning algorithm by majority," *Information and Computation*, vol. 121, no. 2, pp. 256–285, 1995.

[80] R. Schapire, Y. Freund, P. Bartlett, and W. Lee, "Boosting the margin: A new explanation for the effectiveness of voting methods," *The Annals of Statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.

[81] L. Mason, J. Baxter, P. Bartlett, and M. Frean, "Boosting algorithms as gradient descent in function space," in *Advances in Neural Information Processing Systems*, Denver, CO, 1999, pp. 512–518.

[82] S. Rosset and E. Segal, "Boosting density estimation," in *Advances in Neural Information Processing Systems*, Vancouver, Canada, 2003, pp. 657–664.

[83] F. Wang, C. Zhang, and N. Lu, "Boosting GMM and its two applications," *Multiple Classifier Systems*, vol. 3541, no. 1, pp. 12–21, 2005.

[84] D. Seppi, A. Batliner, B. Schuller, S. Steidl, T. Vogt, J. Wagner, L. Devillers, L. Vidrascu, N. Amir, and V. Aharonson, "Patterns, prototypes, performance: Classifying emotional user states," in *Interspeech*, Brisbane, Australia, 2008, pp. 601–604.

[85] D. Ververidis and C. Kotropoulos, "Emotional speech recognition: Resources, features, and methods," *Speech Communication*, vol. 48, no. 9, pp. 1162–1181, 2006.

[86] D. Neiberg, K. Elenius, and K. Laskowski, "Emotion recognition in spontaneous speech using GMMs," in *9th International Conference on Spoken Language Processing*, Pittsburgh, PA, 2006, pp. 809–812.

[87] C. Lee, S. Yildirim, M. Bulut, A. Kazemzadeh, C. Busso, Z. Deng, S. Lee, and S. Narayanan, "Emotion recognition based on phoneme classes," in *8th International Conference on Spoken Language Processing*, Jeju, Korea, 2004, pp. 889–892.

[88] T. Nwe, S. Foo, and L. De Silva, "Speech emotion recognition using hidden Markov models," *Speech Communication*, vol. 41, no. 4, pp. 603–623, 2003.

[89] D. A. Reynolds, "Gaussian mixture models," in *Encyclopedia of Biometric Recognition*, S. Li, Ed. Secaucus, NJ: Springer-Verlag, 2008, pp. 117–121.

[90] "Harvard sentences. From the appendix of: IEEE Subcommittee on Subjective Measurements IEEE Recommended Practices for Speech Quality Measurements," *IEEE Transactions on Audio and Electroacoustics*, vol. 17, pp. 227–246, 1969.

[91] M. Figueiredo and A. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 381–396, 2002.

[92] T. Cover, J. Thomas, and J. Wiley, *Elements of Information Theory.* Hoboken, NJ: Wiley-Interscience, 1991.

[93] Z. Rached, F. Alajaji, and L. Campbell, "The Kullback-Leibler divergence rate between Markov sources," *IEEE Transactions on Information Theory*, vol. 50, no. 5, pp. 917–921, 2004.

[94] L. Yin, X. Wei, Y. Sun, J. Wang, and M. Rosato, "A 3D facial expression database for facial behavior research," in *7th International Conference on Automatic Face and Gesture Recognition*, Southampton, UK, 2006, pp. 211–216.

[95] J. Norris, *Markov Chains.* Cambridge, UK: Cambridge University Press, 1997.

[96] C. Leggetter and P. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.

[97] M. Do, "Fast approximation of Kullback-Leibler distance for dependence trees and hidden Markov models," *IEEE Signal Processing Letters*, vol. 10, no. 4, pp. 115–118, 2003.

[98] P. Phillips, H. Wechsler, J. Huang, and P. Rauss, "The FERET database and evaluation procedure for face-recognition algorithms," *Image and Vision Computing*, vol. 16, no. 5, pp. 295–306, 1998.

[99] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Kauai, HI, 2001, pp. 511–518.

[100] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[101] Y. Tian, T. Kanade, and J. Cohn, "Facial expression analysis," in *Handbook of Face Recognition*, S. Li and A. Jain, Eds. Secaucus, NJ: Springer-Verlag, 2005, pp. 247–275.

[102] B. Fasel and J. Luettin, "Automatic facial expression analysis: A survey," *Pattern Recognition*, vol. 36, no. 1, pp. 259–275, 2003.

[103] M. Pantic and L. Rothkrantz, "Automatic analysis of facial expressions: The state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1424–1445, 2002.

[104] Y. Hu, Z. Zeng, L. Yin, X. Wei, J. Tu, and T. Huang, "A study of non-frontal-view facial expressions recognition," in *19th International Conference on Pattern Recognition*, Tampa, FL, 2008, pp. 1–4.

[105] R. Wright and B. Lipchak, *OpenGL Superbible.* Indianapolis, IN: Sams, 2004.

[106] J. Silva and S. Narayanan, "Upper bound Kullback-Leibler divergence for transient hidden Markov models," *IEEE Transactions on Signal Processing*, vol. 56, no. 9, pp. 4176–4188, 2008.

[107] R. Usmani, "Inversion of a tridiagonal Jacobi matrix," *Linear Algebra and its Applications*, vol. 212, pp. 413–414, 1994.

[108] F. Samaria, "Face recognition using hidden Markov models," Ph.D. dissertation, University of Cambridge, Cambridge, UK, 1994.

[109] F. Samaria and S. Young, "HMM-based architecture for face identification," *Image and Vision Computing*, vol. 12, no. 8, pp. 537–543, 1994.

[110] A. Nefian and M. Hayes III, "An embedded HMM-based approach for face detection and recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Phoenix, AZ, 1999, pp. 3553–3556.

[111] A. Nefian and M. Hayes III, "Maximum likelihood training of the embedded HMM for face detection and recognition," in *IEEE International Conference on Image Processing*, Vancouver, Canada, 2000, pp. 33–36.

[112] A. Nefian, "Embedded Bayesian networks for face recognition," in *IEEE International Conference on Multimedia & Expo*, Lusanne, Switzerland, 2002, pp. 133–136.

[113] H. Park and S. Lee, "A truly 2-D hidden Markov model for off-line handwritten character recognition," *Pattern Recognition*, vol. 31, no. 12, pp. 1849–1864, 1998.

[114] J. Li, A. Najmi, and R. Gray, "Image classification by a two-dimensional hidden Markov model," *IEEE Transactions on Signal Processing*, vol. 48, no. 2, pp. 517–533, 2002.

[115] H. Othman and T. Aboulnasr, "A simplified second-order HMM with application to face recognition," in *IEEE International Symposium on Circuits and Systems*, Sydney, Australia, May 2001, pp. 161–164.

[116] H. Othman and T. Aboulnasr, "A separable low complexity 2D HMM with application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1229–1238, 2003.

[117] Y. Zhang, M. Brady, and S. Smith, "Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm," *IEEE Transactions on Medical Imaging*, vol. 20, no. 1, pp. 45–57, 2002.

[118] S. Kullback and R. Leibler, “On information and sufficiency,” *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[119] H. Jiang, “Discriminative training of HMMs for automatic speech recognition: A survey,” *Computer Speech & Language*, vol. 24, no. 4, pp. 589–608, 2010.