Model-Based and Data-Driven Techniques for Environment-Robust Automatic Speech Recognition

주변 환경에 강인한 음성인식을 위한 모델 및 데이터기반 기법

Ph.D. Dissertation

Shin Jae Kang

School of Electrical Engineering and Computer Science

Seoul National University

Abstract

In this thesis, we propose model-based and data-driven techniques for environment-robust automatic speech recognition. The model-based technique is the feature enhancement method in the reverberant noisy environment to improve the performance of Gaussian mixture model-hidden Markov model (HMM) system. It is based on the interacting multiple model (IMM), which was originally developed in single-channel scenario. We extend the single-channel IMM algorithm such that it can handle the multi-channel inputs under the Bayesian framework. The multi-channel IMM algorithm is capable of tracking time-varying room impulse responses and background noises by updating the relevant parameters in an on-line manner. In order to reduce the computation as the number of microphones increases, a computationally efficient algorithm is also devised. In various simulated and real environmental conditions, the performance gain of the proposed method has been confirmed.

The data-driven techniques are based on deep neural network (DNN)-HMM hybrid system. In order to enhance the performance of DNN-HMM system in the adverse environments, we propose three techniques. Firstly, we propose a novel supervised pre-training technique for DNN-HMM system to achieve robust speech recognition in adverse environments. In the proposed approach, our aim is to initialize the DNN parameters such that they yield abstract features robust to acoustic environment variations. In order to achieve this, we first derive the abstract features from an early fine-tuned DNN model which is trained based on a clean speech database. By using the derived abstract features as the target values, the standard error back-propagation algorithm with the stochastic gradient descent method is performed to estimate the initial parameters of the DNN. The performance of the proposed algorithm was evaluated on Aurora-4 DB and better results were observed compared to a number of conventional pre-training methods.

Secondly, a new DNN-based robust speech recognition approaches taking advantage of noise estimates are proposed. A novel part of the proposed approaches is that the time-varying noise estimates are applied to the DNN as additional inputs. For this, we extract the noise estimates in a frame-by-frame manner from the IMM algorithm which has been known to show good performance in tracking slowly-varying background noise. The performance of the proposed approaches is evaluated on Aurora-4 DB and better performance is observed compared to the conventional DNN-based robust speech recognition algorithms.

Finally, a new approach to DNN-based robust speech recognition using soft target labels is proposed. The soft target labeling means that each target value of the DNN output is not restricted to 0 or 1 but takes non negative values in (0,1) and their sum equals 1. In this study, the soft target labels are obtained from the forward-backward algorithm well-known in HMM training. The proposed method makes the DNN training be more robust in noisy and unseen conditions. The performance of the proposed approach was evaluated on Aurora-4 DB and various mismatched noise test conditions, and found better compared to the conventional hard target labeling method.

Furthermore, in the data-driven approaches, an integrated technique using above three algorithms and model-based technique is described. In matched and mismatched noise conditions, the performance results are discussed. In matched noise conditions, the initialization method for the DNN was effective to enhance the recognition performance. In mismatched noise conditions, the combination of using the noise estimates as an DNN input and soft target labels showed the best recognition results in all the tested combinations of the proposed techniques.

Keywords: Robust speech recognition, multi-channel, interacting multiple model (IMM), dereverberation, pre-training, denoising, background noise estimation, deep neural network (DNN), DNN-based regression, back-propagation, soft target labels, Viterbi alignment, forward-backward algorithm.

Student number: 2010-30974

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, automatic speech recognition (ASR) is playing an important role in a huge number of applications such as smart phones, smart TV, car navigations and wearable devices. The performance of ASR has been improved by using hidden Markov model (HMM) to deal with the temporal variability of speech and modeling its emission probability with Gaussian mixture model (GMM). As an alternative approach, artificial neural networks (ANNs) was used to predict HMM states. At that time, however, neither the hardware nor the learning algorithms were adequate for training neural networks with many hidden layers on large amounts of data, and the performance benefits of using neural networks with a single hidden layer were not sufficiently large to seriously challenge GMMs. Over the last few years, advances in both machine learning algorithms and computer hardware have led to more efficient methods for training deep neural networks (DNNs) that contain many layers of nonlinear hidden units and a very large output layer. The DNN-HMM hybrid system to estimate the emission probabilites using DNN instead of ANN has outperformed the performance of the GMM-HMM system. However, the input signals for ASR systems are often degraded by acoustic reverberation, background noise and other interferences, which naturally lead to the performance deterioration of ASR systems in adverse environments. It should be considered to design the ASR system.

In order to alleviate this performance degradation in adverse environments, a variety of techniques using either a single microphone or multiple microphones have been developed e.g., speech enhancement, feature compensation and model adaptation algorithms [1]- [2]. Though separate algorithms perform differently, their ultimate goal is nothing but to reduce the mismatch between the degraded input signal and the trained recognition model parameters.

In most cases the target speech and noise or other interference sources reside in different spatial locations. Multiple microphone arrays are useful to extract the desired signal especially when each sound source is separated spatially. During the past several decades multi-channel based beamforming techniques such as the generalized sidelobe canceller (GSC) [3], [4] have been proposed to attenuate the coherent interfering sources and acoustic reverberation. However, the performance is sensitive to the estimation of transfer function (TF) or time of arrival (TOA) between the microphone array and the fixed or moving speaker target. So, the estimation of TF or TOA still remains a challenging task. Multi-channel based criteria are also directly applied in the feature domain for robust speech recognition [5], which however focus on additive background noise conditions only.

A natural way to deal with reverberation and background noise in the adverse environments is to use multi-condition training, which trains the acoustic model with not only the clean speech data but also all available reverberant noisy speech data. However, there are two major problems with multi-condition training. The first is that during training it is hard to enumerate all reverberant noise types, reverberation times and SNRs encountered in test environments. The second is that the model trained with multi-condition training has a very broad distribution because it needs to model all the environments. Given the unsatisfactory behavior of multi-condition training, it is necessary to work on technologies that directly deal with the noise and reverberation.

In this thesis, model-based and data-driven techniques for environment-robust automatic speech recognition are proposed. The proposed methods can be applied to both GMM-HMM and DNN-HMM system. In Chapter 4, we propose a novel approach to feature enhancement in multi-channel scenario. Our approach is based on the interacting multiple model (IMM), which was originally developed in single-channel scenario. We extend the single-channel IMM algorithm such that it can handle the multi-channel inputs under the Bayesian framework. The multi-channel IMM algorithm is capable of tracking time-varying room impulse responses and background noises by updating the relevant parameters in an on-line manner. In order to reduce the computation as the number of microphones increases, a computationally efficient algorithm is also devised. In various simulated and real environmental conditions, the performance gain of the proposed method has been confirmed.

In Chapter 5, we propose a novel supervised pre-training technique for DNN-HMM systems to achieve robust speech recognition in adverse environments. In the proposed approach, our aim is to initialize the DNN parameters such that they yield abstract features robust to acoustic environment variations. In order to achieve this, we first derive the abstract features from an early fine-tuned DNN model which is trained based on a clean speech database. By using the derived abstract features as the target values, the standard error back-propagation algorithm with the stochastic gradient descent method is performed to estimate the initial parameters of the DNN. The performance of the proposed algorithm was evaluated on Aurora-4 DB and better results were observed compared to a number of conventional pre-training methods.

In Chapter 6, a new DNN-based robust speech recognition approaches taking advantage of noise estimates are proposed. A novel part of the proposed approaches is that the time-varying noise estimates are applied to the DNN as additional inputs. For this, we extract the noise estimates in a frame-by-frame manner from the IMM algorithm which has been known to show good performance in tracking slowly-varying background noise. The performance of the proposed approaches is evaluated on Aurora-4 DB and better performance is observed compared to the conventional DNN-based robust speech recognition algorithms.

In Chapter 7, a novel approach to DNN-based robust speech recognition using soft target labels is proposed. The soft target labeling means that each target value of the DNN output is not restricted to 0 or 1 but takes non negative values in (0,1) and their sum equals 1. In this study, the soft target labels are obtained from the forward-backward algorithm well-known in hidden Markov model training. The proposed method makes the DNN training be more robust in noisy and unseen conditions. The performance of the proposed approach was evaluated on Aurora-4 DB and various noise mismatched test conditions, and found better compared to the conventional hard target labeling method. Furthermore, an integrated technique using the algorithms proposed in Chapter 5 and Chapter 6 is also described. In noise matched and mismatched conditions, the performance results are discussed.

The rest of the thesis is organized as follows: The next chapter introduces the experimental environments and database used in this thesis. In Chapter 3, the conventional robust speech recognition techniques are described. In Chapter 4, a multi-channel IMM-based feature enhancement algorithm is proposed. In Chapter 5, a supervised denoising pre-training for robust ASR in the DNN-HMM system is proposed. In Chapter 6, DNN-based frameworks for robust speech recognition using noise estimates are introduced. Finally, a DNN-based robust speech recognition using soft target labels is proposed in Chapter 7. The conclusions are drawn in Chapter 8.

Chapter 2

Experimental Environments and Database

We describe the experimental environments assumed in this thesis. In order to evaluate the performance of the proposed techniques, database is explained. For the multi-channel reverberant noisy conditions, we made the simulation data according to the target speaker positions using TI digits corpus. Aurora-4 DB was used for the DNN-based techniques, which is widely used in the robust speech recognition area.

2.1 ASR in Hands-Free Scenario and Feature Extraction

We consider a typical hands-free scenario for ASR as shown in Fig. 2.1. The target speaker is located at a certain distance from a far-field microphone in an enclosed room, which results in acoustic reverberation. Let $\bar{y}[t]$ be the corrupted speech captured at a microphone with $t \in \{0, 1, \cdots\}$ denoting time index. It consists

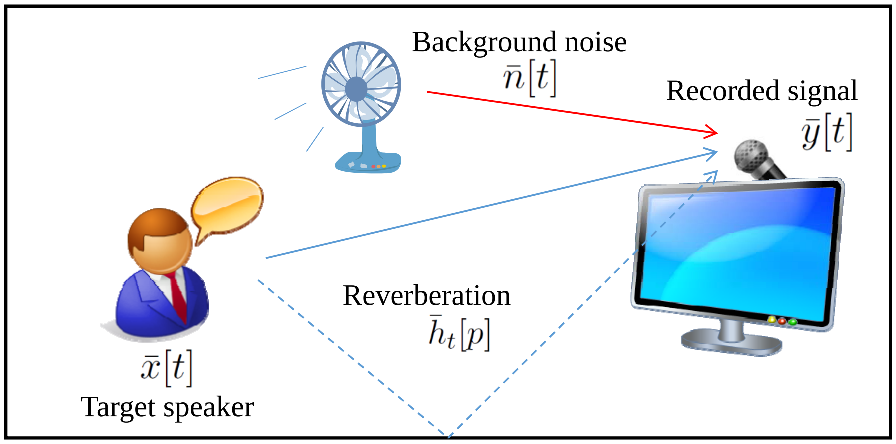


Figure 2.1: Reverberant noisy environment.

of two components, the reverberant speech signal $\bar{s}[t]$ and background noise $\bar{n}[t]$ as given by

$$\bar{y}[t] = \bar{s}[t] + \bar{n}[t]. \tag{2.1}$$

Let $\bar{h}_t[p]$ represent the room impulse response (RIR) from the target speaker to the microphone at the time index t with the corresponding tap index $p \in \{0, 1, \cdots\}$. Then, the reverberant speech signal results from the convolution of the source speech signal with the time-variant RIR $\bar{h}_t[p]$, i.e.,

$$\bar{s}[t] = \sum_{p=0}^{\infty} \bar{h}_t[p]\bar{x}[t-p]. \tag{2.2}$$

The noise signal includes all the reverberant background noise signals which originate from noise sources as well as inherent microphone noise. The three components, $\bar{x}[t]$, $\bar{h}_t[p]$ and $\bar{n}[t]$, may be modeled as independent random processes. The microphone signal is passed to an ASR system which is expected to estimate the word sequence spoken by the target speaker.

In our work, we focus on the mel frequency cepstral coefficient (MFCC) which is one of the predominant feature parameters for the state-of-the-art ASR system. General MFCC feature extraction process can be shown in Fig. 2.2. The time signal $\tilde{y}[t]$ which is obtained after pre-emphasis of the captured speech signal $\bar{y}[t]$ is framed and weighted by a Hamming analysis window function $\tilde{w}_a[t]$ of finite length L_w to obtain the frame-dependent windowed signal segments:

$$\tilde{y}_m[l_w] = \tilde{w}_a[l_w]\tilde{y}[l_w + mB] \tag{2.3}$$

in which $m \in \{0, 1, \cdots\}$, $l_w \in \{0, 1, \cdots, L_w - 1\}$ and B denote the frame index, time index within the segment and the length of frame shift, respectively. The windowed signal segments are subsequently transformed to the frequency domain by applying the discrete Fourier transform (DFT), resulting in the short-time discrete Fourier transform (STDFT) representations given by

$$\tilde{Y}_m[f] = \sum_{l_w=0}^{L_w-1} \tilde{y}_m[l_w] \exp\left(-\mathrm{j}\frac{2\pi f}{\mathrm{N}_f} l_w\right) \tag{2.4}$$

where $f \in \{0, 1, \cdots, \mathrm{N}_f - 1\}$ is the frequency bin index, N_f denotes the number of frequency bins and j represents imaginary unit $\sqrt{-1}$. Since the $\tilde{Y}_m[f]$ is complex value, we can get the power spectral coefficients by taking the square of the magnitude. And then mel power spectral coefficient (MPSC) is obtained by applying a bank of Q overlapping triangular filters Λ_q, $q \in \{0, 1, \cdots, Q - 1\}$, which are equally spaced on the mel scale, according to

$$Y_{m,q} = \sum_f |\tilde{Y}_m[f]|^2 \Lambda_q[f]. \tag{2.5}$$

The MPSC is computed by taking the natural logarithm as

$$y_{m,q} = \ln(Y_{m,q}) \tag{2.6}$$

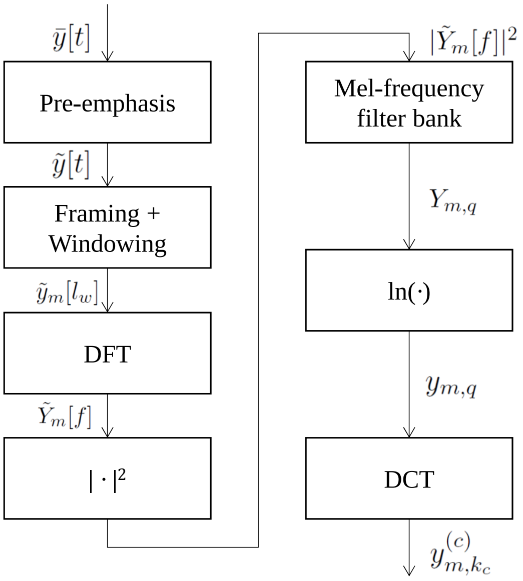


Figure 2.2: Feature extraction.

where $y_{m,q}$ represents the log MPSC (LMPSC) at the q-th mel band.

Finally, the $y_{m,q}$ is decorrelated by applying the discrete cosine transform (DCT) to obtain the well-known MFCC as follows:

$$y^{(c)}_{m,k_c} = \sum_{q=0}^{Q-1} y_{m,q} \cos\left(\frac{k_c \pi}{\mathrm{N}_c}\left(q + \frac{1}{2}\right)\right) \tag{2.7}$$

where $k_c \in \{0, 1, \cdots, \mathrm{N}_c - 1\}$ and N_c denote the MFCC index and the overall number of MFCC components.

2.2 Relationship between Clean and Distorted Speech in Feature Domain

In this section, we describe the relationship between the clean and distorted speech in reverberant noisy environment in detail. For that, by substituting (2.2) into (2.1), we can rewrite the reverberant speech signal in time domain as

$$\bar{y}[t] = \sum_{p=0}^{\infty} \bar{h}_t[p]\bar{x}[t-p] + \bar{n}[t]. \tag{2.8}$$

This relationship can be converted to MPSCs as follows:

$$Y_{m,q} = \left(\sum_{\tau=0}^{L} X_{m-\tau,q} H_{m,\tau,q} + N_{m,q} \right) + \mathcal{E}_{m,q} \tag{2.9}$$

where $X_{m,q}$, $N_{m,q}$, $Y_{m,q}$ and $\mathcal{E}_{m,q}$ respectively represent the MPSCs of the clean, background noise, corrupted speech signal and approximation error at the m-th frame for the q-th mel band, and $H_{m,\tau,q}$ denotes the average RIR magnitudes per mel band which is assumed to have finite length $(L+1)$, i.e., $H_{m,\tau,q} = 0$ for all $\tau > L$. Based on (2.9), the relationship between the corresponding LMPSCs is given by

$$y_{m,q} = \ln\left(Y_{m,q}\right) \tag{2.10}$$

$$= \ln\left(\sum_{\tau=0}^{L} X_{m-\tau,q} H_{m,\tau,q} + N_{m,q} \right) + \ln\left(1 + \frac{\mathcal{E}_{m,q}}{\sum_{\tau=0}^{L} X_{m-\tau,q} H_{m,\tau,q} + N_{m,q}} \right) \tag{2.11}$$

$$= \ln\left(\sum_{\tau=0}^{L} \exp\left(x_{m-\tau,q} + h_{m,\tau,q}\right) + \exp\left(n_{m,q}\right) \right) + v_{m,q} \tag{2.12}$$

where the logarithmic mel magnitude spectral-like representation of the RIR $h_{m,\tau,q}$ and the error term $v_{m,q}$ are respectively given by

$$h_{m,\tau,q} = \ln\left(H_{m,\tau,q}\right) \tag{2.13}$$

$$v_{m,q} = \ln\left(1 + \frac{\mathcal{E}_{m,q}}{\sum_{\tau=0}^{L} X_{m-\tau,q} H_{m,\tau,q} + N_{m,q}} \right). \tag{2.14}$$

Let $\mathbf{y}_m$, $\mathbf{x}_m$, $\mathbf{n}_m$ and $\mathbf{v}_m$ respectively denote the Q-dimensional LMPSC vectors of the reverberant noisy speech, clean speech, background noise and approximation error of the observation model in (2.12) at the m-th frame. We also let $\mathbf{h}_{m,\tau}$ represent the Q-dimensional vector which reflects the time-variant log frequency response of the reverberant acoustic path from the speaker to the microphone, which is specified at a frame index m for a tap index τ. These vectors are defined in the following way:

$$\mathbf{y}_m = [y_{m,0} \quad y_{m,1} \quad \cdots \quad y_{m,Q-1}]' \tag{2.15}$$

$$\mathbf{x}_m = [x_{m,0} \quad x_{m,1} \quad \cdots \quad x_{m,Q-1}]' \tag{2.16}$$

$$\mathbf{n}_m = [n_{m,0} \quad n_{m,1} \quad \cdots \quad n_{m,Q-1}]' \tag{2.17}$$

$$\mathbf{v}_m = [v_{m,0} \quad v_{m,1} \quad \cdots \quad v_{m,Q-1}]' \tag{2.18}$$

$$\mathbf{h}_{m,\tau} = [h_{m,\tau,0} \quad h_{m,\tau,1} \quad \cdots \quad h_{m,\tau,Q-1}]' \tag{2.19}$$

with the prime denoting a matrix or a vector transpose. When the background noise and acoustic reverberation exist simultaneously, the relation shown in (2.12) can be written in a vector form as follows:

$$\mathbf{y}_m = \ln\left(\sum_{\tau=0}^{L} \exp\left(\mathbf{x}_{m-\tau} + \mathbf{h}_{m,\tau}\right) + \exp\left(\mathbf{n}_m\right)\right) + \mathbf{v}_m \tag{2.20}$$

where the functions $\exp(\cdot)$ and $\ln(\cdot)$ are applied component-wisely and we assume that the approximation error distribution is given by

$$\mathbf{v}_m \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{v}}, \boldsymbol{\Sigma}_{\mathbf{v}}\right) \tag{2.21}$$

in which $\mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$ indicates a Gaussian probability density function with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

2.3 Database

In this section, the database (DB) used in this thesis is described. As mentioned above, the target environments are reverberant noisy. For that, we choose two kinds of DBs widely used in robust speech recognition area: TI digits corpus and Aurora-4 DB.

2.3.1 TI Digits Corpus

TI digits corpus has been collected for use in designing and evaluating algorithms for speaker independent recognition of connected digit sequences [6]. This dialect balanced DB consists of more than 25 thousand digit sequences spoken by over 300 men, women and children. The data were collected in a quiet environment and digitized at 20 kHz. The number of speakers contributing data for the DB is 326 in total. Eleven digits were used: “zero”, “one”, “two”, ..., “nine” and “oh”. Seventy-seven sequences of these digits were collected from each speaker, and consisted of the following types.

1. 22 isolated digits (two tokens/digit)
2. 11 two digit sequences
3. 11 three digit sequences
4. 11 four digit sequences
5. 11 five digit sequences
6. 11 seven digit sequences

Hence each speaker provided 253 digits and 176 digit transitions. Each speaker was seated in an acoustically treated sound room (Tracoustics, Inc., Model RE-244B acoustic enclosure), with the microphone placed 2-4 inches in front of the speaker’s mouth. The microphone was Electro-Voice RE-16 Dynamic Cardioid.

TI digits corpus have been collected in clean environment. In order to apply the reverberant noisy environment, we made a simulation DB. A small rectangular room of dimensions 6 m × 4 m × 3 m (length × width × height) was configured as shown

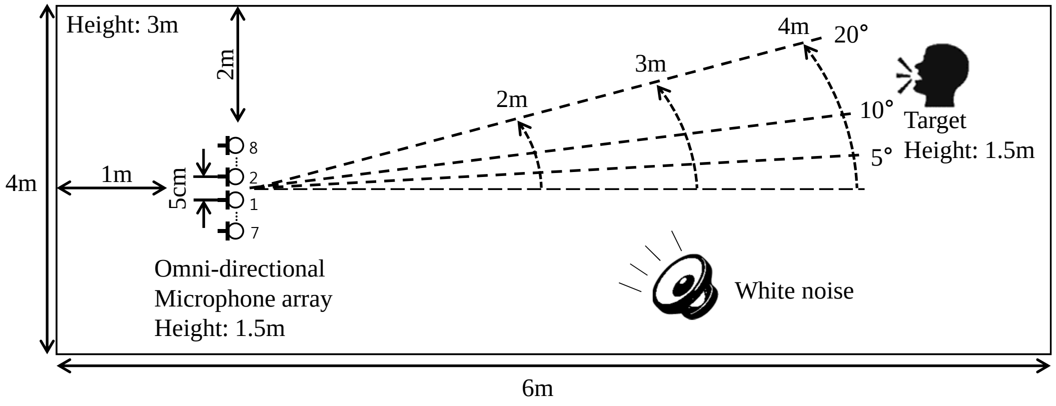


Figure 2.3: Simulated reverberant noisy environment in the presence of microphone array.

in Fig. 2.3. In our experiments, we used eight omni-directional microphones placed at the height of 1.5 m. The distance between the microphones was fixed at 5 cm. For convenience, we numbered each microphone from the center of the microphones in order. For the background noise, white noise was used as the diffuse noise source. In order to simulate various environmental conditions, we varied the distance between the target speaker and microphones from 2 m to 4 m. The target speaker was positioned at 5° to 20° deviated from the center of the microphones. Totally, the target speaker was located at nine positions according to the distances and separation angles. The RIRs were simulated by Allen and Berkley's image method [7] using Habets's software [8]. The reverberation range was varied from 300 to 700 ms and all the conditions were tested in the signal-to-noise ratios (SNRs) between 0 to 20 dB. For the results of single-channel techniques, the microphone was positioned at the center of microphone 1 and 2.

2.3.2 Aurora-4 DB

Aurora-4 DB [9] was made using 5k-word vocabulary based on the Wall Street Journal (WSJ) DB. The WSJ data were recorded with a primary Sennheiser microphone and with a secondary microphone in parallel. The recordings with the secondary microphone are used for enabling recognition experiments with different frequency characteristics in the transmission channel. An additional filtering is applied to consider the realistic frequency characteristics of terminals and equipment in the telecommunication area. Two standard frequency characteristics are used which have been defined by the ITU. The abbreviations G.712 and P.341 have been introduced as reference to these filters. The G.712 characteristic is defined for the frequency range of the usual telephone bandwidth up to 4 kHz and has a flat characteristic in the range between 300 and 3400 Hz. P.341 is defined for the frequency range up to 8 kHz and represents a band pass filter with a very low cut off frequency at the lower end and a cut off frequency at about 7 kHz at the higher end of the bandpass. These two filters can be applied to data sampled at 8 or 16 kHz, respectively. We use the 16 kHz sampled data.

The corpus has two training sets: clean- and multi-condition. Both clean- and multi-condition sets consist of the same 7138 utterances from 83 speakers. The clean-condition set consists of only the primary Sennheiser microphone data. One half of the utterances in the multi-condition set were recorded by the primary Sennheiser microphone and the other half were recorded using one of 18 different secondary microphones. Both halves include a combination of clean speech and speech corrupted by one of six different types of noises (car, babble, restaurant, street, airport and train station) at a range of SNRs between 10 and 20 dB. These noises represent

Table 2.1: Aurora-4 DB (m: male, f: female).

	Training data	Development data	Evaluation data
Hour	15.1471	8.9694	9.4026
Utterance	7138	4620	4620
Speaker	83 (m: 42, f: 41)	10 (m: 6, f: 4)	8 (m: 5, f: 3)

realistic scenarios of application environments for mobile telephones. Some noises are fairly stationary like e.g. the car noise. Others contain non-stationary segments like e.g. the recordings on the street and at the airport. The SNR was defined as the ratio of signal to noise energy after filtering both speech and noise signals with P.341 filter characteristic.

The evaluation was conducted on the test set consisting of 330 utterances from 8 speakers. This test set was recorded by the primary microphone and a number of secondary microphones. These two sets were then each corrupted by the same six noises used in the training set at SNRs between 5 and 15 dB, creating a total of 14 test sets. These 14 sets were then grouped into 4 subsets based on the type of distortions: none (clean speech), additive noise only, channel distortion only and noise + channel distortion. For convenience, we denote these subsets by Set_A, Set_B, Set_C and Set_D, respectively. Note that the types of noises are common across training and test sets but the SNRs of the data are not.

For the validation test, we used the development set in Aurora-4 DB consisting of 330 utterances from 10 speakers not included in the training and test set speakers. A total of 14 sets with the same conditions as the test set were constructed. More detail information for Aurora-4 DB is given in Table 2.1.

Chapter 3

Previous Robust ASR Approaches

The ASR system has poor performance when the training and test condition of speech recognition are mismatched. Since the acoustic model of ASR is usually trained in clean training data, mismatch between acoustics is inevitable in real environment and the compensation is needed. To alleviate the performance degradation, feature or model domain techniques have proposed. In the feature domain approaches, the distorted input features are compensated before being decoded using the acoustic recognition models that were trained on clean speech. The model adaptation approaches aim at reducing mismatch between the trained speech recognition models and the input speech by adapting the model parameters of the recognizer to the distorted environments.

An alternative approach is to train the acoustic model of ASR as a multi-condition training which trains the acoustic model with all available noisy speech data. This approach has better performance than above mentioned techniques but it has some disadvantages. Since the characteristic of the background noise is very diverse. All kinds of the background noise cannot be covered in training condition. A bunch of data is helpful but collecting the multi-condition database is difficult and expensive.

In this chapter, we discuss the conventional environment-robust techniques for robust speech recognition. Typical IMM-based feature compensation and its variation algorithm in single- and multi-channel scenarios are described using clean-condition GMM-HMM system. Under the multi-condition training of the acoustic model, DNN-HMM hybrid system is introduced.

3.1 IMM-Based Feature Compensation in Noise Environment

The conventional feature compensation algorithms are usually performed in log mel frequency domain [10]. If we assume that the target environment has only the background noise, we can derive the relationship between the noisy speech and the clean speech as (2.1). Here, the reverberant speech signal $\bar{s}[t]$ is treated as the clean speech signal. In (2.5), $|\tilde{Y}_m[f]|^2$ can be rewritten as following:

$$\begin{aligned} |\tilde{Y}_m[f]|^2 &= |\tilde{S}_m[f]|^2 + |\tilde{N}_m[f]|^2 + 2|\tilde{S}_m[f]||\tilde{N}_m[f]|\cos\theta_f && (3.1) \\ &\approx |\tilde{S}_m[f]|^2 + |\tilde{N}_m[f]|^2 && (3.2) \end{aligned}$$

where $\tilde{S}_m[f]$, $\tilde{N}_m[f]$ and θ_f indicate the spectrum of windowed $\bar{s}[l]$ and $\bar{n}[l]$ and the random angle between two complex variables $\tilde{S}_m[f]$ and $\tilde{N}_m[f]$. Statistically, the expected value of the last term in (3.1) is treats as zero since $\bar{x}[l]$ and $\bar{n}[l]$ are statistically independent. After substituting (3.2) into (2.5), we can rewrite the (2.6) as following:

$$Y_{m,q} \approx S_{m,q} + N_{m,q} \tag{3.3}$$

$$y_{m,q} \approx \ln\left(S_{m,q} + N_{m,q}\right) \tag{3.4}$$

$$= \ln\left(\exp(s_{m,q}) + \exp(n_{m,q})\right) \tag{3.5}$$

where $S_{m,q}$, $N_{m,q}$, $s_{m,q}$ and $n_{m,q}$ denote MPSCs and LMPSCs of channel distorted clean speech and background noise at the q-th mel band, respectively. The vectors are defined in (2.15), (2.17) and the following way:

$$\mathbf{s}_m = [s_{m,0}\ s_{m,1}\ \cdots\ s_{m,Q-1}]' \tag{3.6}$$

in which prime indicates the transpose of a vector or matrix. The relation between channel distorted clean speech and noisy LMPSCs can be rewritten as a vector form:

$$\mathbf{y}_m \approx \ln\left(\exp(\mathbf{s}_m) + \exp(\mathbf{n}_m)\right) = f(\mathbf{s}_m, \mathbf{n}_m). \tag{3.7}$$

The functions $\exp(\cdot)$ and $\ln(\cdot)$ in (3.7) are applied component-wisely.

The goal of IMM-based feature compensation technique is to estimate the channel distorted clean feature vector sequence $\mathbf{s}_0^T = [\mathbf{s}_0, \mathbf{s}_1, \cdots, \mathbf{s}_T]$ which is referred as clean feature to be estimated given noisy feature sequence $\mathbf{y}_0^T = [\mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_T]$, where $x_{m1}^{m2} = [x_{m1}, x_{m1+1}, \cdots, x_{m2}]$ denotes a subsequence of vectors from frame index $m1$ to $m2$. The relation between $\mathbf{y}_m$ and $\mathbf{s}_m$ is nonlinear as shown in (3.7). To estimate the clean feature, the probability density function of it is assumed as a mixture of Gaussian distributions such that

$$p(\mathbf{s}_m) = \sum_{k=0}^{K-1} p(k)\mathcal{N}\left(\mathbf{s}_m; \boldsymbol{\mu}_{s,k}(m), \boldsymbol{\Sigma}_{s,k}(m)\right) \tag{3.8}$$

where $p(k)$, $\boldsymbol{\mu}_{s,k}$, $\boldsymbol{\Sigma}_{s,k}$ and K represent the weight, mean vector, covariance matrix of the k-th Gaussian distribution and the number of Gaussians, respectively. As for the distribution of background noise, we employ a single Gaussian background noise model as given by

$$\mathbf{n}_m \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{n}}(m), \boldsymbol{\Sigma}_{\mathbf{n}}(m)\right) \tag{3.9}$$

where the mean vector $\boldsymbol{\mu}_{\mathbf{n}}(m)$ and covariance matrix $\boldsymbol{\Sigma}_{\mathbf{n}}(m)$ are unknown and should be estimated during the environment compensation procedure.

It is difficult to estimate directly the clean and noise feature vector $\mathbf{s}_m$ and $\mathbf{n}_m$ due to its nonlinearity in (3.7). To alleviate its difficulty, we apply the piecewise linear approximation to the given nonlinear function by using vector Taylor series (VTS) expansion [11]. In the k-th mixture component, the nonlinear function (3.7) is approximated by

$$\mathbf{y}_m \approx A_k\mathbf{s}_m + B_k\mathbf{n}_m + C_k \tag{3.10}$$

where A_k, B_k and C_k mean constant matrices of dimension $\{Q \times Q\}$, $\{Q \times Q\}$ and $\{Q \times 1\}$ respectively. The estimation of these constant matrices can be obtained like

$$\{\hat{A}_k, \hat{B}_k, \hat{C}_k\} = \underset{\{A_k, B_k, C_k\}}{\arg\min}\; E\left[||\tilde{f}(\mathbf{s}_m, \mathbf{n}_m) - (A_k\mathbf{s}_m + B_k\mathbf{n}_m + C_k)||^2\right] \tag{3.11}$$

where $\{\hat{A}_k, \hat{B}_k, \hat{C}_k\}$ are estimated constant matrices and $\tilde{f}(\mathbf{s}_m, \mathbf{n}_m)$ denotes the function obtained by truncating the VTS expansion of $f(\mathbf{s}_m, \mathbf{n}_m)$ up to a finite order. The more detailed derivation is given in [11]. To estimate the environmental parameter, we assume that the environmental characteristic of background noise slowly changes among the consecutive frames. With this assumption, we employ the noise evolution process by

$$\mathbf{n}_{m+1} = \mathbf{n}_m + w_m \tag{3.12}$$

$$w_m \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{w_m}) \tag{3.13}$$

where $\mathbf{0}$ and $\mathbf{\Sigma}_{w_m}$ indicate zero mean vector and fixed covariance matrix of Gaussian process w_m.

To update the slowly time-varying background noise model parameter $\lambda_{\mathbf{n}_m} = \{\boldsymbol{\mu}_{\mathbf{n}_m}, \mathbf{\Sigma}_{\mathbf{n}_m}\}$ and for a number of practical reasons, we consider a causal estimation scheme which is desired for sequential filtering. One of the efficient methods for the sequential estimation is the IMM technique. Using (3.10) and (3.12), we can construct the multiple state space models according to the mixture components. Here $\mathbf{n}_m$ is treated as a state vector. To solve these linear state space models, the extended Kalman filtering approach based on IMM is applied. The whole procedure has four steps: mixing, Kalman update, probability calculation and output generation steps. The detailed information is given as follows.

1) Mixing step: The initial statistics of the state are constructed by mixing the corresponding estimates at the previous time. Let us define the initial statistics as

$$\boldsymbol{\mu}_{\mathbf{n}}^{0}(m-1|j) = E[\mathbf{n}_{m-1}|k_m = j, \mathbf{y}_0^{m-1}] \tag{3.14}$$

$$\approx E[\mathbf{n}_{m-1}|\mathbf{y}_0^{m-1}] = \hat{\boldsymbol{\mu}}_{\mathbf{n}}(m-1) \tag{3.15}$$

$$\mathbf{\Sigma}_{\mathbf{n}}^{0}(m-1|j) = \mathrm{Cov}[\mathbf{n}_{m-1}|k_m = j, \mathbf{y}_0^{m-1}] \tag{3.16}$$

$$\approx \mathrm{Cov}[\mathbf{n}_{m-1}|\mathbf{y}_0^{m-1}] = \hat{\mathbf{\Sigma}}_{\mathbf{n}}(m-1) \tag{3.17}$$

where $E[\cdot]$ and $\mathrm{Cov}[\cdot]$ represents the expectation and covariance. The k_m indicates the mixture component at m-th frame. Since $\mathbf{n}_m$ is independent of k_m with our assumption, the initial statistics can be approximated such as the last terms of (3.15) and (3.17). The $\hat{\boldsymbol{\mu}}_{\mathbf{n}}(m-1)$ and $\hat{\mathbf{\Sigma}}_{\mathbf{n}}(m-1)$ mean the final estimated statistics from IMM algorithm at $(m-1)$-th frame.

2) Kalman update step: The conventional Kalman update is carried out based on the initial statistics from the mixing step. Let $\boldsymbol{\mu}_{\mathbf{n}}^{p}(m|j)$ and $\mathbf{\Sigma}_{\mathbf{n}}^{p}(m|j)$ be the mean

and covariance of the one-step-ahead predictive state estimate in the j-th mixture component at frame m. The one-step-ahead predictive state estimate is calculated by using the state evolution model (3.12),

$$\boldsymbol{\mu}_{\mathbf{n}}^{p}(m|j) = \boldsymbol{\mu}_{\mathbf{n}}^{0}(m-1|j) \tag{3.18}$$

$$\boldsymbol{\Sigma}_{\mathbf{n}}^{p}(m|j) = \boldsymbol{\Sigma}_{\mathbf{n}}^{0}(m-1|j) + \boldsymbol{\Sigma}_{w_m}. \tag{3.19}$$

And then the innovation $e(m|j)$, its covariance $R_e(m|j)$ and Kalman gain $K_f(m|j)$ are computed by using linearized observation model (3.10) and predictive state estimates (3.18) and (3.19).

$$e(m|j) = \mathbf{y}_m - A_j\boldsymbol{\mu}_{s,j}(m) - B_j\boldsymbol{\mu}_{\mathbf{n}}^{p}(m|j) - C_j \tag{3.20}$$

$$R_e(m|j) = A_j\boldsymbol{\Sigma}_{s,j}(m)A_j' + B_j\boldsymbol{\Sigma}_{\mathbf{n}}^{p}(m|j)B_j' \tag{3.21}$$

$$K_f(m|j) = \boldsymbol{\Sigma}_{\mathbf{n}}^{p}(m|j)B_j'R_e^{-1}(m|j). \tag{3.22}$$

With $e(m|j)$, $R_e(m|j)$ and $K_f(m|j)$, we can compute $\hat{\boldsymbol{\mu}}_{\mathbf{n}}(m|j)$ and $\hat{\boldsymbol{\Sigma}}_{\mathbf{n}}(m|j)$ by the use of conventional measurement update scheme shown below:

$$\hat{\boldsymbol{\mu}}_{\mathbf{n}}(m|j) = \boldsymbol{\mu}_{\mathbf{n}}^{p}(m|j) + K_f(m|j)e(m|j) \tag{3.23}$$

$$\hat{\boldsymbol{\Sigma}}_{\mathbf{n}}(m|j) = \boldsymbol{\Sigma}_{\mathbf{n}}^{p}(m|j) - K_f(m|j)B_j\boldsymbol{\Sigma}_{\mathbf{n}}^{p}(m|j). \tag{3.24}$$

For a number of experiments we have observed that large variation of estimated parameter over frame is not effective in speech recognition performance. For that reason, as proposed in [10], the Kalman gain shrink factor is applied.

$$K_f^*(m|j) = \alpha K_f(m|j) \tag{3.25}$$

where $K_f^*(m|j)$ represents the shrinked Kalman gain and α refers the shrinking factor between (0, 1]. By substituting $K_f(m|j)$ with $K_f^*(m|j)$ in (3.23) and (3.24), we can get the slowly varying parameter estimates.

3) Probability calculation step: In this step, the posterior probability corresponding to each mixture component is computed. Since the mixture component is assumed to be independent of the previous observations, we have $\gamma_j(m)$ at j-th mixture such as

$$\begin{aligned}\gamma_j(m) &= p(k_m = j|\mathbf{y}_0^m) && (3.26)\\ &= p(k_m = j|\mathbf{y}_m, \mathbf{y}_0^{m-1}) && (3.27)\\ &= \frac{p(\mathbf{y}_m|k_m = j, \mathbf{y}_0^{m-1})p(k_m = j)}{\sum_{i=0}^{K-1} p(\mathbf{y}_m|k_m = i, \mathbf{y}_0^{m-1})p(k_m = i)} && (3.28)\end{aligned}$$

where $p(k_m = j)$ is the prior probability of the j-th mixture component and denominator of (3.28) is the normalization factor. The $p(\mathbf{y}_m|k_m = j, \mathbf{y}_0^{m-1})$ represents the one-step-ahead predictive likelihood of the observation within the j-th Kalman filter, which is calculated during the Kalman update step.

4) Output generation step: All the estimates from the mixture components are combined with the posterior probabilities of probability calculation step to obtain the final estimated statistics $\hat{\boldsymbol{\mu}}_{\mathbf{n}}(m)$ and $\hat{\boldsymbol{\Sigma}}_{\mathbf{n}}(m)$ at m-th frame and then their values are propagated to the next frame.

$$\begin{aligned}\hat{\boldsymbol{\mu}}_{\mathbf{n}}(m) &= \sum_{j=0}^{K-1} \gamma_j(m)\hat{\boldsymbol{\mu}}_{\mathbf{n}}(m|j) && (3.29)\\ \hat{\boldsymbol{\Sigma}}_{\mathbf{n}}(m) &= \sum_{j=0}^{K-1} \gamma_j(m)[\hat{\boldsymbol{\Sigma}}_{\mathbf{n}}(m|j)\\ &\quad + (\hat{\boldsymbol{\mu}}_{\mathbf{n}}(m|j) - \hat{\boldsymbol{\mu}}_{\mathbf{n}}(m))(\hat{\boldsymbol{\mu}}_{\mathbf{n}}(m|j) - \hat{\boldsymbol{\mu}}_{\mathbf{n}}(m))']. && (3.30)\end{aligned}$$

3.2 Single-Channel Reverberation and Noise-Robust Feature Enhancement Based on IMM

We consider a typical hands-free scenario for ASR as shown in Fig. 2.1. We consider the single-channel scenario where only one microphone is used. The relationship between the reverberant noisy speech and the clean speech in LMPSC domain can be written by (2.20).

Our goal is to estimate the clean speech feature sequence $\mathbf{x}_0^m = [\mathbf{x}_0\ \mathbf{x}_1 \cdots \mathbf{x}_m]$ given the observation sequence $\mathbf{y}_0^m = [\mathbf{y}_0\ \mathbf{y}_1 \cdots \mathbf{y}_m]$ where $\mathbf{x}_{m_1}^{m_2} = [\mathbf{x}_{m_1}\ \mathbf{x}_{m_1+1}\ \cdots\ \mathbf{x}_{m_2}]$ denotes a subsequence of vectors from frame index m_1 to m_2. For a number of practical reasons, we consider a causal estimation scheme which is desired for sequential filtering. One of the efficient methods for the sequential estimation is the IMM technique [10] which is based on a mixture of multiple dynamic models adopting Kalman filtering techniques. In the IMM framework, a linear dynamic model is specified for each mixture component and the state estimates obtained from all the mixture components are merged together to produce a single estimate which is propageted to the next time frame.

In this work, the clean speech component and the log frequency response are jointly handled as the state vector

$$\mathbf{z}_m = [\mathbb{x}_m\ \mathbb{h}_m]' \tag{3.31}$$

where the prime denotes the transpose of a vector or matrix, $\mathbb{x}_m = \left[\mathbf{x}'_m\ \mathbf{x}'_{m-1}\ \cdots\ \mathbf{x}'_{m-L}\right]$ and $\mathbb{h}_m = \left[\mathbf{h}'_{m,0}\ \mathbf{h}'_{m,1}\ \cdots\ \mathbf{h}'_{m,L}\right]$ respectively denote the local clean speech trajectory and log frequency response LMPSCs consisting of $(L+1)$ consecutive frames. The a priori distribution of the state vector is assumed to be a mixture of K Gaus-

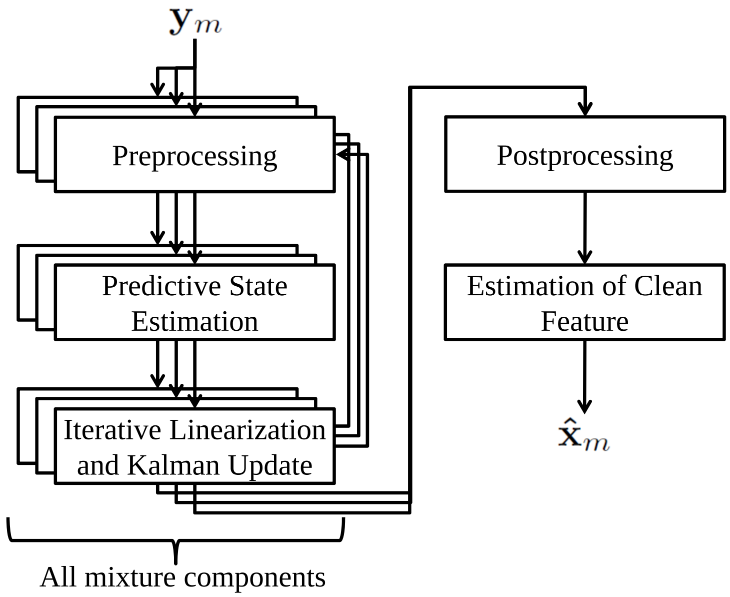

$\mathbf{y}_m$
Preprocessing
Predictive State Estimation
Iterative Linearization and Kalman Update
All mixture components
Postprocessing
Estimation of Clean Feature
$\hat{\mathbf{x}}_m$

Figure 3.1: Single-channel feature enhancement based on IMM.

sians. This prior distribution is obtained from the training database of the clean features $\mathbb{x}_m$ and a simple random walk model for $\mathbb{h}_m$ [12]. Based on this prior modeling, we can construct a switching linear dynamic model with K mixture components. Since the observation function in (2.20) is nonlinear with respect to the specified state variables, it is important to linearize it for each mixture component by means of the vector Taylor series approach [11].

The feature enhancement procedure based on the IMM technique is given in Fig. 3.1. The procedures consist of five steps and are performed at each frame. In the pre-processing step, the initial estimates of the state are constructed by mixing the corresponding estimates at the previous frame. And then in the predictive state estimate step, one-step-ahead estimates of the state are calculated from the state transition model. In the iterative linearization and Kalman update step, the linearized observation model is iteratively updated and then the usual Kalman filtering is performed. In this step, the Kalman gains and the innovations are computed and then the state estimates are updated. In the post-processing step, the final state estimate is computed by merging the estimates produced by all the mixture components.

3.3 Multi-Channel Feature Enhancement for Robust Speech Recognition

Multi-channel approaches use the benefits of the additional informations carried out by the presence of multiple speech observations. In most cases the target speech and noise or other interference sources reside in different spatial locations. Multiple microphone arrays are useful to extract the desired signal especially when each sound source is separated spatially. The microphone array system is theoretically able to obtain a significant gain over single-channel approaches, since it may exploit the spatial diversity.

In ASR scenario, beamforming techniques are employed as pre-processing stage. Beamforming is a method by which signals from several sensors can be combined to emphasize a desired source and to suppress all other noise and interference. Beamforming begins with the assumption that the positions of all sensors are known, and that the positions of the desired sources are known or can be estimated as well. The simplest of beamforming algorithms, the delay and sum beamformer (DSB), uses only this geometrical knowledge to combine the signals from several sensors. The theory of DSB originates from narrowband antenna array processing, where the plane waves at different sensors are delayed appropriately to be added exactly in phase. In this way, the array can be electronically steered towards a specific direction. This principle is also valid for broadband signals, although the directivity will then be frequency dependent. A DSB aligns the microphone signals to the direction of the speech source by delaying and summing the microphone signals. This kind of beamformer is proved to perform well when the number of microphones is relatively high, and when the noise sources are spatially white. On the contrary, performance degrade since noise reduction is strongly dependent on the direction of arrival of the noise signal. As a consequence, DSB performance on reverberant environments is poor.

During the past several decades multi-channel based beamforming techniques such as the GSC [3], [4] have been proposed to attenuate the coherent interfering sources and acoustic reverberation. However, in case of incoherent or diffuse noise fields, beamforming alone does not provide sufficient noise reduction, and postfiltering is normally required. Postfiltering includes signal detection, noise estimation, and spectral enhancement. The performance is sensitive to the estimation of TF or TOA between the microphone array and the fixed or moving speaker target. So, the estimation of TF or TOA still remains a challenging task.

Multi-channel based criteria are also directly applied in the feature domain for robust speech recognition [5], which are extended from the single-channel algorithms. However, these algorithms focus on additive background noise conditions only.

3.4 DNN-Based Robust Speech Recognition

Recently, ASR is playing an important role in a huge number of applications such as smart phones, car navigations and wearable devices. We have witnessed a great advance in ASR technology due to the development of new machine learning algorithms and the increased computing power. One of the most notable technological advances recently achieved in the ASR society is the rediscovery of the DNNs. For a long time since their introduction, DNNs were found difficult to be applied to various practical tasks. This difficulty mostly came from two reasons: lack of efficient training algorithms and computational complexity. These issues were resolved to some extent by the introduction of the layer-by-layer pre-training algorithm for the restricted Boltzmann machines (RBMs) [13] and the parallel processing approach using graphic processing units. With these approaches, DNNs can be successfully trained and have shown notable performances in various research areas covering speech, audio, text and image processing. The remarkable performance of the DNN is attributed to its capability in automatically learning complicated nonlinear relationship between the input and target values. Particularly in the area of speech recognition, DNN-HMM systems have outperformed those that are based on the GMM-HMM [14], [15]. If a sufficient amount of training data is available, more complicated input-target relationship can be easily learned by using wider and deeper neural network architectures.

The interest on the DNN's capability in efficient learning has been also expanded to the robust speech recognition area. Conventional speech enhancement or feature compensation techniques [10], [16], [17] applied to robust ASR usually require some specific models or formulations to account for the nonlinear relationship between the clean and noisy speech processes in an appropriate signal domain. Even stereo data-based feature mapping algorithms such as SPLICE [18] and switching linear dynamic system (SLDS) [19] are also designed based on some simplified models of the speech and noise generation processes. In contrast, the DNNs have the advantage that they can directly train an arbitrary unknown relationship between the noisy input and target values. For training the DNN in adverse environments, not only the clean features but also the corrupted features are used as an input of the DNN, which is common in the robust ASR area. In a sense, this approach can be regarded as a multi-condition training technique.

DNN is a multi-layer perceptron with many hidden layers, which consists of the stacked RBMs. The DNN training is divided by the unsupervised pre-training and supervised fine-tuning. In the pre-training, unsupervised greedy layer-wise training procedures are performed using the contrastive divergence algorithm to explain the nonlinear representation of the input data in stacked RBM networks. The trained parameters can be used as the initial variables of a deep network. In the fine-tuning, the output layer is added on the top of the pre-trained deep networks. The error between DNN network output and reference target value is calculated according to the objective function. The type of the output layer and the objective function are chosen based on the tasks. For the regression tasks, a linear layer is typically used to generate the output vector and a mean square error is used as the objective function. For the multi-class classification tasks, each output neuron represents a class. To serve as a valid multinomial probability distribution, the output vector should satisfy the sum of all the output neuron equal to 1. This can be done by normalizing the excitation with a softmax function. The errors between the softmax outputs and the target labels are calculated by a cross entropy. By using back-propagation algorithm with a stochastic gradient descent method, the DNN network parameters are updated.

There are also several DNN structures to be exploited for the robust speech recognition, i.e., recurrent neural network or long short term memory recurrent networks. In this thesis, we focus on improving the performance of the recognition results by complementing the disadvantages of the DNN.

Chapter 4

Multi-Channel IMM-Based Feature Enhancement for Robust Speech Recognition

4.1 Introduction

The performance of an ASR system is usually degraded when the input speech is distorted by background noise or acoustic reverberation. In order to alleviate this performance degradation in adverse environments, a variety of techniques have been developed e.g., speech enhancement, feature compensation and model adaptation algorithms [1], [2], [12], [20]–[24]. Though separate algorithms perform differently, their ultimate goal is nothing but to reduce the mismatch between the degraded input signal and the trained recognition model parameters. In this chapter, we focus on the feature compensation techniques which directly enhance the distorted input feature vectors to match the characteristic of the training data before being decoded

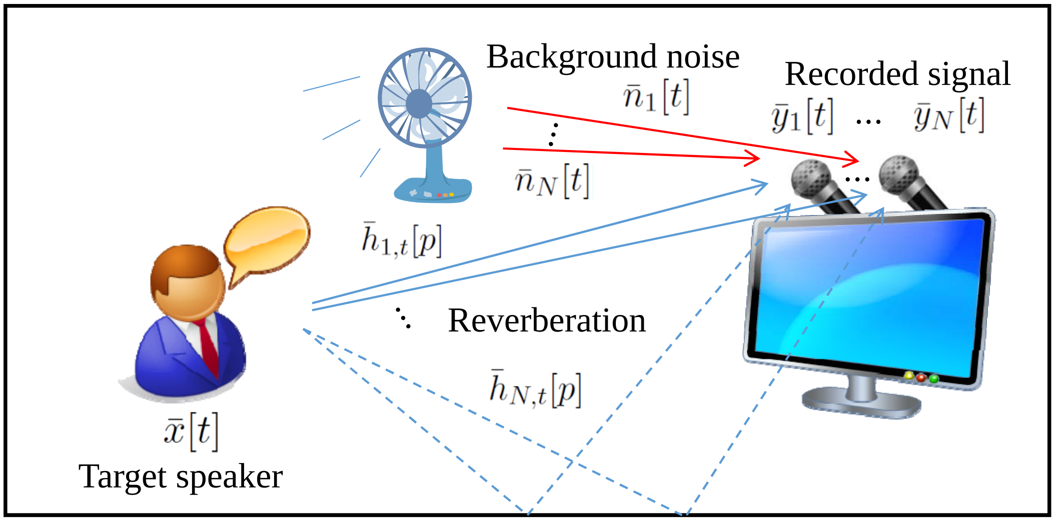


Figure 4.1: Reverberant noisy environment in multi-channel scenario.

by the acoustic recognition model.

In most cases the target speech and noise or other interference sources reside in different spatial locations. Multiple microphone arrays are useful to extract the desired signal especially when each sound source is separated spatially. During the past several decades multi-channel based beamforming techniques such as the GSC [3], [4] have been proposed to attenuate the coherent interfering sources and acoustic reverberation. However, the performance is sensitive to the estimation of TF or TOA between the microphone array and the fixed or moving speaker target. So, the estimation of TF or TOA still remains a challenging task. Multi-channel based criteria are also directly applied in the feature domain for robust speech recognition [5], which however focus on additive background noise conditions only.

In this chapter, we propose a novel multi-channel feature enhancement technique applied in the log-spectral domain. In the proposed approach, we extend the IMM algorithm [12] originally designed in the single-channel scenario so that it can fit to the multi-channel processing in reverberant noisy environments. Basically we assume that the geometric specification of the microphone array and the position of the target speaker are unknown, which doesn't matter for enhancing the features in the proposed technique. The proposed method has mainly two advantages. First, no *a prior* knowledge of the RIR is needed. Second, the parameters concerned with the acoustic reverberation and background noise are sequentially updated in a frame-by-frame manner instead of utterance-by-utterance basis for tracking their time-varying nature. This type of real-time update of the RIR parameters is very important in handling the possible movements of the talker or microphones.

One of the drawbacks of this multi-channel technique is that the computational amount increases rapidly as the number of microphones grows. In a synchronous data collection system such as the multiple sensor system or microphone array, a number of Kalman filtering algorithms with different processing structures e.g., parallel, sequential and data compression filters have been introduced in [25]. In the case of linear systems, all these processing structures are found equivalent and optimal. Motivated by this study, we also propose an algorithm which modifies the original multi-channel IMM approach for reducing computation.

4.2 Observation Model in Multi-Channel Reverberant Noisy Environment

We consider a typical hands-free scenario for ASR in which multiple microphones are used as shown in Fig. 4.1. The target speaker is located in a certain distance from the microphones in an enclosed room, which results in acoustic reverberation. Let $\bar{y}_i[t]$ be the signal obtained from the i-th microphone with $t \in \{0, 1, \cdots\}$ denoting the time index. If $\bar{x}[t]$ is the target speech signal and $\bar{h}_{i,t}[p]$ represents the RIR from the target speaker to the i-th microphone with corresponding tap index $p \in \{0, 1, \cdots\}$, then

$$\bar{y}_i[t] = \sum_{p=0}^{\infty} \bar{h}_{i,t}[p]\bar{x}[t-p] + \bar{n}_i[t] \tag{4.1}$$

where $\bar{n}_i[t]$ is the background noise added to the i-th microphone input.

By using the formulation presented in [12], we can rewrite the relation of (4.1) in the logarithmic mel magnitude spectral coefficient (LMMSC) domain as follow:

$$\mathbf{y}_{i,m} = \ln\left(\sum_{\tau=0}^{L} \exp\left(\mathbf{x}_{m-\tau} + \mathbf{h}_{i,m,\tau}\right) + \exp\left(\mathbf{n}_{i,m}\right)\right) + \mathbf{v}_{i,m} \tag{4.2}$$

where $\mathbf{y}_{i,m}$, $\mathbf{x}_m$, $\mathbf{h}_{i,m,\tau}$, $\mathbf{n}_{i,m}$ and $\mathbf{v}_{i,m}$ respectively denote the Q-dimensional LMMSC vectors of the reverberant noisy speech, clean speech, time-variant log frequency response of the reverberant acoustic path for a tap index τ which is assumed to have finite length $(L+1)$, i.e., $\mathbf{h}_{i,m,\tau} = -\infty$ for all $\tau > L$, background noise and approximation error of the observation model at the m-th frame which are collected at the i-th microphone. The only difference of (4.2) from the formulation derived in [12] is that we now add subscript i to identify each microphone. The functions $\exp(\cdot)$ and $\ln(\cdot)$ in (4.2) are applied component-wisely and we assume that the error distribution at each microphone is given by

$$\mathbf{v}_{i,m} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{v}_i}, \boldsymbol{\Sigma}_{\mathbf{v}_i}\right) \tag{4.3}$$

in which $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ indicates a Gaussian PDF with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

4.3 Multi-Channel Feature Enhancement in a Bayesian Framework

In our work, the clean speech component and the N log frequency responses are jointly handled as a state vector $\mathbb{z}_m$ at the m-th frame where N indicates the number of microphones. The core idea of our approach is to estimate the posterior probability $p(\mathbb{z}_m|\mathbb{y}_0^m)$ of the state vector

$$\mathbb{z}_m = [\mathbb{x}_m \; \mathbb{h}_{1,m} \; \mathbb{h}_{2,m} \; \cdots \; \mathbb{h}_{N,m}]' \tag{4.4}$$

$$\mathbb{x}_m = \left[\mathbf{x}_m' \; \mathbf{x}_{m-1}' \; \cdots \; \mathbf{x}_{m-L}'\right] \tag{4.5}$$

$$\mathbb{h}_{i,m} = \left[\mathbf{h}_{i,m,0}' \; \mathbf{h}_{i,m,1}' \; \cdots \; \mathbf{h}_{i,m,L}'\right] \tag{4.6}$$

conditioned on all the N observed reverberant noisy LMMSC vectors

$$\mathbb{y}_0^m = [\mathbb{y}_0 \; \mathbb{y}_1 \; \cdots \; \mathbb{y}_m] \tag{4.7}$$

$$\mathbb{y}_m = \left[\mathbf{y}_{1,m}' \; \mathbf{y}_{2,m}' \; \cdots \; \mathbf{y}_{N,m}'\right]' \tag{4.8}$$

where the prime denotes the transpose of a vector or matrix, $\mathbb{x}_m$ and $\mathbb{h}_{i,m}$ respectively mean a local clean speech and log frequency response LMMSC trajectories consisting of $(L+1)$ consecutive frames at the i-th microphone. In the above formulation, $\mathbf{x}_{m_1}^{m_2} = [\mathbf{x}_{m_1} \; \mathbf{x}_{m_1+1} \; \cdots \; \mathbf{x}_{m_2}]$ denotes a subsequence of vectors from frame index m_1 to m_2 and $\mathbb{y}_m$ means N observations at the m-th frame concatenated to a single vector.

A typical way to compute the posterior distribution of the state vector $\mathbb{z}_m$ based on a Bayesian inference is to recursively compute the predictive distribution $p(\mathbb{z}_m|\mathbb{y}_0^{m-1})$ and posterior distribution $p(\mathbb{z}_m|\mathbb{y}_0^m)$ given the previous reverberant noisy observations as follows:

$$\begin{aligned} p\left(\mathbb{z}_m|\mathbb{y}_0^{m-1}\right) &= \int p\left(\mathbb{z}_m|\mathbb{z}_{m-1},\mathbb{y}_0^{m-1}\right) \\ &\qquad\qquad \times p\left(\mathbb{z}_{m-1}|\mathbb{y}_0^{m-1}\right) d\mathbb{z}_{m-1} \qquad (4.9) \\ p\left(\mathbb{z}_m|\mathbb{y}_0^{m}\right) &= \frac{p\left(\mathbb{y}_m|\mathbb{z}_m,\mathbb{y}_0^{m-1}\right) p\left(\mathbb{z}_m|\mathbb{y}_0^{m-1}\right)}{\int p\left(\mathbb{y}_m|\mathbb{z}_m,\mathbb{y}_0^{m-1}\right) p\left(\mathbb{z}_m|\mathbb{y}_0^{m-1}\right) d\mathbb{z}_m}. \qquad (4.10) \end{aligned}$$

If both $p\left(\mathbb{z}_m|\mathbb{y}_0^{m-1}\right)$ and $p\left(\mathbb{z}_m|\mathbb{y}_0^{m}\right)$ are assumed to be Gaussian distributions, it is sufficient to revise the statistical moments up to the second-order which are defined as follows:

$$\begin{cases} \hat{\mathbb{z}}_{m|m-1} = E\left[\mathbb{z}_m|\mathbb{y}_0^{m-1}\right] \\ \hat{\mathbf{\Sigma}}_{\mathbb{z}_{m|m-1}} = E\left[\left(\mathbb{z}_m - \hat{\mathbb{z}}_{m|m-1}\right)\left(\mathbb{z}_m - \hat{\mathbb{z}}_{m|m-1}\right)' |\mathbb{y}_0^{m-1}\right] \end{cases} \qquad (4.11)$$

$$\begin{cases} \hat{\mathbb{z}}_{m|m} = E\left[\mathbb{z}_m|\mathbb{y}_0^{m}\right] \\ \hat{\mathbf{\Sigma}}_{\mathbb{z}_{m|m}} = E\left[\left(\mathbb{z}_m - \hat{\mathbb{z}}_{m|m}\right)\left(\mathbb{z}_m - \hat{\mathbb{z}}_{m|m}\right)' |\mathbb{y}_0^{m}\right] \end{cases} \qquad (4.12)$$

where $E[\cdot]$ indicates expectation. The mean vectors and covariance matrices in (4.11) and (4.12) are obtained through the IMM algorithm which will be described in Section 4.4.

When we assume that the frequency responses are independent of the process of generating the clean speech, the overall predictive distribution $p(\mathbb{z}_m|\mathbb{z}_{m-1},\mathbb{y}_0^{m-1})$ can be factorized, i.e.,

$$\begin{aligned} & p(\mathbb{z}_m|\mathbb{z}_{m-1},\mathbb{y}_0^{m-1}) \qquad (4.13) \\ & \approx p(\mathbb{x}_m|\mathbb{x}_{m-1},\mathbb{y}_0^{m-1})p(\mathbb{h}_m|\mathbb{h}_{m-1},\mathbb{y}_0^{m-1}). \end{aligned}$$

In (4.13), since $p(\mathbb{x}_m|\mathbb{x}_{m-1},\mathbb{y}_0^{m-1})$ and $p(\mathbb{h}_m|\mathbb{h}_{m-1},\mathbb{y}_0^{m-1})$ are posterior probabilities conditioned on the observation, they may have some degree of correlation with each other. However $\mathbb{x}_m$ and $\mathbb{h}_m$ are usually assumed to be independent when simulating reverberant environments [7]. For that reason and ease of implementation, we assume that they can be simply approximated as independent with each other.

Under the framework of environment compensation, our goal is to estimate the sequence of the local clean feature vector trajectory $\mathbb{x}_0^m$, log frequency response $\{\mathbb{h}_i\}_0^m$ and background noise $\{\mathbf{n}_i\}_0^m$ given a noisy feature $\mathbb{y}_0^m$. For this purpose, we propose in this section a variety of models for the clean speech, RIR and background noise by considering the characteristics of the individual components, and also present the methods of describing process evolution and function approximation necessary for an efficient estimation.

4.3.1 A Priori Clean Speech Model

In this section, the prior models for clean speech, RIR, and background noise are described. As mentioned in [12], the *a priori* clean speech distribution is assumed as a mixture of K Gaussians to approximate a high degree of speech dynamics as follows:

$$p(\mathbb{x}_m) = \sum_{j=0}^{K-1} p(\gamma_m = j)\mathcal{N}\left(\mathbb{x}_m; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right) \tag{4.14}$$

where $\gamma_m \in \{0, 1, \cdots, K-1\}$ denotes the index of the mixture component at the m-th frame, and $p(\gamma_m = j)$, $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ represent the weight, mean vector and covariance matrix of the j-th Gaussian distribution respectively. It is noted that the covariance matrix $\boldsymbol{\Sigma}_j$ in the above a priori model should be properly structured to incorporate the temporal and spectral correlations among the components of the local clean speech trajectory $\mathbb{x}_0^m$. Once (4.14) is employed, the clean speech term in

(4.13) can be written

$$\begin{aligned} & p(\mathbb{x}_m|\mathbb{x}_{m-1}, \mathbb{y}_0^{m-1}) \\ & = \sum_{j=0}^{K-1} p(\mathbb{x}_m|\mathbb{x}_{m-1}, \mathbb{y}_0^{m-1}, \gamma_m = j) p(\gamma_m = j|\mathbb{x}_{m-1}, \mathbb{y}_0^{m-1}). \end{aligned} \tag{4.15}$$

As in [26], we employ the approximation that

$$p(\mathbb{x}_m|\mathbb{x}_{m-1}, \mathbb{y}_0^{m-1}, \gamma_m = j) \approx p(\mathbb{x}_m|\mathbb{x}_{m-1}, \gamma_m = j) \tag{4.16}$$

$$p(\gamma_m = j|\mathbb{x}_{m-1}, \mathbb{y}_0^{m-1}) \approx \sum_{k=0}^{K-1} a_{jk} p(\gamma_{m-1} = k|\mathbb{y}_0^{m-1}) \tag{4.17}$$

where

$$a_{jk} = p(\gamma_m = j|\gamma_{m-1} = k) \tag{4.18}$$

denotes the time-invariant stat transition probability. This kind of prior model, known as the switching linear dynamic model (SLDM), explicitly considers correlations between successive speech feature vectors which are due to the speech production process on the one hand and the feature extraction process on the other. SLDMs have been successfully applied to noise robust speech recognition in the previous studies [26], [27], [28].

The parameters of an SLDM are generally learned from a set of clean speech training data through the well-known expectation maximization (EM) algorithm [29], which iteratively delivers improved parameter estimates obtained from maximizing the likelihood of the training data based on previous parameter estimates.

4.3.2 A Priori Model for RIR

We assume that the i-th log frequency response $\mathbb{h}_{i,m}$ at the m-th frame is statistically independent of the clean speech and background noise features. Since the

RIR is non-stationary, the parameter estimate for $\mathbb{h}_{i,m}$ must be constantly updated to track its time evolution. In our work, we exploit a random walk process which is the simplest solution for predicting the next state as given by

$$\mathbb{h}_{i,m} = \mathbb{h}_{i,m-1} + \mathbf{w}_{\mathbb{h},i,m} \tag{4.19}$$

$$\mathbf{w}_{\mathbb{h},i,m} \sim \mathcal{N}\left(\mathbf{0}_{Q(L+1)N}, \sigma_{\mathbb{h}}^2 \mathbf{I}_{Q(L+1)N}\right) \tag{4.20}$$

where $\mathbf{0}_d$ represents the zero vector with dimension d and $\mathbf{I}_d$ denotes the identity matrix of size $d \times d$. When $\sigma_{\mathbb{h}}^2$ is small, this model is well suited to a slowly evolving RIR environment.

4.3.3 A Priori Model for Background Noise

The characteristics of the background noise are very diverse. It is difficult to train all kinds of the background noise in advance. In a short period before active speech activity occurs, however, we can assume that the background noise only exists and its characteristic is stationary. Furthermore, the complexity of the background noise model needs to be quite low to allow a fast and computationally efficient online tracking [10]. For these reasons, we employ a single Gaussian background noise model as given by

$$\mathbf{n}_{i,m} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{n}_{i,m}}, \boldsymbol{\Sigma}_{\mathbf{n}_{i,m}}\right) \tag{4.21}$$

where the mean vector $\boldsymbol{\mu}_{\mathbf{n}_{i,m}}$ and covariance matrix $\boldsymbol{\Sigma}_{\mathbf{n}_{i,m}}$ are unknown and should be estimated during the environment compensation procedure.

4.3.4 State Transition Formulation

As in [12], the transition process of the state vector $\mathbf{z}_m$ for the j-th Gaussian mixture component can be simply structured as follows:

$$\mathbb{z}_m = \mathcal{A}^{(j)} \mathbb{z}_{m-1} + \mathbf{b}_m^{(j)} \tag{4.22}$$

with

$$\mathcal{A}^{(j)} = \left[\begin{array}{c:c} \begin{array}{cccc} & \mathbf{AB}^{-1} & & \mathbf{O}_Q \\ \mathbf{I}_Q & \mathbf{O}_Q & \cdots & \mathbf{O}_Q \\ \mathbf{O}_Q & \mathbf{I}_Q & \cdots & \mathbf{O}_Q \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{O}_Q & \cdots & \mathbf{I}_Q & \mathbf{O}_Q \end{array} & \mathbf{O} \\ \hdashline \mathbf{O}' & \mathbf{I}_{Q(L+1)N} \end{array} \right] \tag{4.23}$$

$$\mathbf{b}_m^{(j)} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{b}}^{(j)}, \boldsymbol{\Sigma}_{\mathbf{b}}^{(j)} \right) \tag{4.24}$$

where

$$\mathbf{A} = \left[\text{Cov}(\mathbf{x}_m, \mathbf{x}_{m-1}) \; \text{Cov}(\mathbf{x}_m, \mathbf{x}_{m-2}) \; \cdots \; \text{Cov}(\mathbf{x}_m, \mathbf{x}_{m-L}) \right] \tag{4.25}$$

$$\mathbf{B} = \begin{bmatrix} \text{Cov}(\mathbf{x}_{m-1}, \mathbf{x}_{m-1}) & \text{Cov}(\mathbf{x}_{m-1}, \mathbf{x}_{m-2}) & \cdots & \text{Cov}(\mathbf{x}_{m-1}, \mathbf{x}_{m-L}) \\ \text{Cov}(\mathbf{x}_{m-2}, \mathbf{x}_{m-1}) & \text{Cov}(\mathbf{x}_{m-2}, \mathbf{x}_{m-2}) & \cdots & \text{Cov}(\mathbf{x}_{m-2}, \mathbf{x}_{m-L}) \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}(\mathbf{x}_{m-L}, \mathbf{x}_{m-1}) & \text{Cov}(\mathbf{x}_{m-L}, \mathbf{x}_{m-2}) & \cdots & \text{Cov}(\mathbf{x}_{m-L}, \mathbf{x}_{m-L}) \end{bmatrix} \tag{4.26}$$

and $\text{Cov}(a, b)$ denotes the covariance between two vectors a and b. In addition,

$$\boldsymbol{\mu}_{\mathbf{b}}^{(j)} = \begin{bmatrix} \tilde{\boldsymbol{\mu}}_{\mathbf{b}} \\ \mathbf{0}_Q \\ \vdots \\ \mathbf{0}_Q \end{bmatrix}, \quad \boldsymbol{\Sigma}_{\mathbf{b}}^{(j)} = \left[\begin{array}{c:c} \begin{array}{cccc} \tilde{\boldsymbol{\Sigma}}_{\mathbf{b}} & \mathbf{O}_Q & \cdots & \mathbf{O}_Q \\ \mathbf{O}_Q & \mathbf{O}_Q & \cdots & \mathbf{O}_Q \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O}_Q & \mathbf{O}_Q & \cdots & \mathbf{O}_Q \end{array} & \mathbf{O} \\ \hdashline \mathbf{O}' & \sigma_{\mathbb{h}}^2 \mathbf{I}_{Q(L+1)N} \end{array} \right] \tag{4.27}$$

where

$$\tilde{\boldsymbol{\mu}}_{\mathbf{b}} = E[\mathbf{x}_m] - \mathbf{A}\mathbf{B}^{-1} \begin{bmatrix} E[\mathbf{x}_{m-1}] \\ E[\mathbf{x}_{m-2}] \\ \vdots \\ E[\mathbf{x}_{m-L}] \end{bmatrix} \tag{4.28}$$

$$\tilde{\boldsymbol{\Sigma}}_{\mathbf{b}} = \mathrm{Cov}(\mathbf{x}_m, \mathbf{x}_m) - \mathbf{A}\mathbf{B}^{-1}\mathbf{A}' \tag{4.29}$$

with $\mathbf{O}_Q$, $\mathbf{O}$ and $\mathbf{I}_d$ denoting a zero matrix with size $Q \times Q$, $Q(L+1) \times Q(L+1)N$ and the identity matrix of size $d \times d$. The above formulation given by (4.22)-(4.29) is an extension of the state transition model derived in [12] to the case of multiple RIRs.

4.3.5 Function Linearization

It is difficult to estimate directly the clean speech feature vector $\mathbf{x}_m$ and all the log frequency responses $\mathbf{h}_{i,m,\tau}$ of N-channel inputs from the speech distortion model (4.2) due to its nonlinearity. To alleviate its difficulty, we apply the piecewise linear approximation to the given nonlinear function by using Taylor series expansion. The first order form of Taylor series expansion at the i-th microphone input feature is given as in the following:

$$f_i\left(\mathbb{z}_m, \mathbf{n}_{i,m}\right) = \ln\left(\sum_{\tau=0}^{L} \exp\left(\mathbf{x}_{m-\tau} + \mathbf{h}_{i,m,\tau}\right) + \exp\left(\mathbf{n}_{i,m}\right)\right) \tag{4.30}$$

$$\approx \mathbf{G}_{i,m}\mathbb{z}_m + \mathbf{H}_{i,m}\mathbf{n}_{i,m} + \mathbf{q}_{i,m} \tag{4.31}$$

where

$$\mathbf{G}_{i,m} = \left[\tfrac{\partial f_i}{\partial \mathbb{x}_m}' \cdots \mathbf{0}'_{Q(N+1)} \cdots \tfrac{\partial f_i}{\partial \mathbb{h}_{i,m}}' \cdots \mathbf{0}'_{Q(N+1)} \cdots \right] \tag{4.32}$$

$$\mathbf{H}_{i,m} = \frac{\partial f_i}{\partial \mathbf{n}_{i,m}} \tag{4.33}$$

$$\mathbf{q}_{i,m} = f_i \left(\mathbb{z}_m^\circ, \mathbf{n}_{i,m}^\circ \right) - \mathbf{G}_{i,m} \mathbb{z}_m^\circ - \mathbf{H}_{i,m} \mathbf{n}_{i,m}^\circ \tag{4.34}$$

and $\mathbb{z}_m^\circ$ and $\mathbf{n}_{i,m}^\circ$ are constant vectors corresponding to the center of vector Taylor series expansion. In our work, we apply the statistical linear approximation [11] method for linear approximation. The matrix form of all the N observed reverberant noisy inputs can be shown as in (4.35).

4.4 Feature Enhancement Algorithm

At each frame m, the proposed feature enhancement procedure based on IMM technique conducts six steps: pre-processing, predictive state estimation, iterative linearization and Kalman update, background noise estimation, post-processing and

$$\begin{aligned} \mathbb{y}_m = \begin{bmatrix} \mathbf{y}_{1,m} \\ \mathbf{y}_{2,m} \\ \vdots \\ \mathbf{y}_{N,m} \end{bmatrix} &= \begin{bmatrix} \mathbf{G}_{1,m} \\ \mathbf{G}_{2,m} \\ \vdots \\ \mathbf{G}_{N,m} \end{bmatrix} \mathbb{z}_m + \begin{bmatrix} \mathbf{H}_{1,m} & \mathbf{0}_Q & \cdots & \mathbf{0}_Q \\ \mathbf{0}_Q & \mathbf{H}_{2,m} & \cdots & \mathbf{0}_Q \\ \mathbf{0}_Q & \mathbf{0}_Q & \ddots & \mathbf{0}_Q \\ \mathbf{0}_Q & \mathbf{0}_Q & \cdots & \mathbf{H}_{N,m} \end{bmatrix} \begin{bmatrix} \mathbf{n}_{1,m} \\ \mathbf{n}_{2,m} \\ \vdots \\ \mathbf{n}_{N,m} \end{bmatrix} + \begin{bmatrix} \mathbf{q}_{1,m} \\ \mathbf{q}_{2,m} \\ \vdots \\ \mathbf{q}_{N,m} \end{bmatrix} \\ &+ \begin{bmatrix} \mathbf{v}_{1,m} \\ \mathbf{v}_{2,m} \\ \vdots \\ \mathbf{v}_{N,m} \end{bmatrix} = \mathbb{G}_m \mathbb{z}_m + \mathbb{H}_m \mathbb{n}_m + \mathbb{q}_m + \mathbb{v}_m \end{aligned} \tag{4.35}$$

clean feature estimation steps.

1) Pre-processing: The initial statistics associated to the j-th iterated Kalman filter are constructed by mixing the corresponding estimates at the previous frame. Let us define the initial statistics as

$$\hat{\mathbb{z}}^{(0,j)}_{m-1|m-1} = E\left[\mathbb{z}_{m-1}|\gamma_m = j, \mathrm{y}_0^{m-1}\right] \tag{4.36}$$

$$\hat{\mathbf{\Sigma}}^{(0,j)}_{\mathbb{z}_{m-1|m-1}} = \mathrm{Cov}\left[\mathbb{z}_{m-1}|\gamma_m = j, \mathrm{y}_0^{m-1}\right]. \tag{4.37}$$

Then, by the IMM approximation [10], we can get

$$\hat{\mathbb{z}}^{(0,j)}_{m-1|m-1} = \sum_{k=0}^{K-1} \Lambda_m^{(j,k)} \hat{\mathbb{z}}^{(k)}_{m-1|m-1} \tag{4.38}$$

$$\begin{aligned}\hat{\mathbf{\Sigma}}^{(0,j)}_{\mathbb{z}_{m-1|m-1}} = \sum_{k=0}^{K-1} \Lambda_m^{(j,k)} \big[\hat{\mathbf{\Sigma}}^{(k)}_{\mathbb{z}_{m-1|m-1}} + \\ (\hat{\mathbb{z}}^{(0,j)}_{m-1|m-1} - \hat{\mathbb{z}}^{(k)}_{m-1|m-1})(\hat{\mathbb{z}}^{(0,j)}_{m-1|m-1} - \hat{\mathbb{z}}^{(k)}_{m-1|m-1})'\big]\end{aligned} \tag{4.39}$$

in which

$$\hat{\mathbb{z}}^{(k)}_{m-1|m-1} = E\left[\mathbb{z}_{m-1}|\gamma_{m-1} = k, \mathrm{y}_0^{m-1}\right] \tag{4.40}$$

$$\hat{\mathbf{\Sigma}}^{(k)}_{\mathbb{z}_{m-1|m-1}} = \mathrm{Cov}\left[\mathbb{z}_{m-1}|\gamma_{m-1} = k, \mathrm{y}_0^{m-1}\right] \tag{4.41}$$

$$\Lambda_m^{(j,k)} = P(\gamma_{m-1} = k|\gamma_m = j, \mathrm{y}_0^{m-1}). \tag{4.42}$$

In (4.42), $\Lambda_m^{(j,k)}$ denotes the probability that model k was active at the $(m-1)$-th frame given that model j is active at the m-th frame conditioned on the observation y_0^{m-1}. Based on (4.17) and (4.18) it can be shown that

$$\Lambda_m^{(j,k)} = \frac{1}{c_j} a_{jk} P^{(k)}_{m-1|m-1} \tag{4.43}$$

with

$$c_j = \sum_{k=1}^{K} a_{jk} P^{(k)}_{m-1|m-1} \tag{4.44}$$

where $P^{(k)}_{m-1|m-1} \equiv P(\gamma_{m-1} = k|\mathbb{y}_0^{m-1})$ is the a posteriori probability that model k is active at frame $(m-1)$ conditioned on the observations $\mathbb{y}_0^{m-1}$. Let $P^{(j)}_{m|m-1} \equiv P(\gamma_m = j|\mathbb{y}_0^{m-1})$ be the *a priori* model probability. Then from (4.17),

$$P^{(j)}_{m|m-1} = \sum_{k=0}^{K-1} a_{jk} P^{(k)}_{m-1|m-1}, \; 0 \leq j \leq K-1. \tag{4.45}$$

2) Predictive state estimation: The one-step-ahead statistics of the predictive state estimate in the j-th mixture component at frame index m based on the initial estimates computed from the previous step be defined by

$$\hat{\mathbb{z}}^{(j)}_{m|m-1} = E\left[\mathbb{z}_m|\gamma_m = j, \mathbb{y}_0^{m-1}\right] \tag{4.46}$$

$$\hat{\mathbf{\Sigma}}^{(j)}_{\mathbb{z}_{m|m-1}} = \mathrm{Cov}\left[\mathbb{z}_m|\gamma_m = j, \mathbb{y}_0^{m-1}\right]. \tag{4.47}$$

Then, by using the state evolution formulation of (4.22)-(4.29), we can derive

$$\hat{\mathbb{z}}^{(j)}_{m|m-1} = \mathcal{A}^{(j)} \hat{\mathbb{z}}^{(0,j)}_{m-1|m-1} + \boldsymbol{\mu}^{(j)}_{\mathbf{b}} \tag{4.48}$$

$$\hat{\mathbf{\Sigma}}^{(j)}_{\mathbb{z}_{m|m-1}} = \mathcal{A}^{(j)} \hat{\mathbf{\Sigma}}^{(0,j)}_{\mathbb{z}_{m-1|m-1}} (\mathcal{A}^{(j)})' + \mathbf{\Sigma}^{(j)}_{\mathbf{b}}. \tag{4.49}$$

3) Iterative linearization and Kalman update: The general approach of this step is similar to that proposed in [30] where the linearization and Kalman update are performed iteratively. The core idea of this approach is to find a more optimal center of vector Taylor series expansion for better linear approximation. Let R denote the total number of iterations and $r = 1, \cdots, R$ indicates an iteration index. As applied in single-channel algorithm, it can be easily extended to the multi-channel algorithm. Recall that $\mathbb{z}^\circ_m$ is the center of vector Taylor series expansion introduced in Subsection 4.3.5. At the r-th iteration we set

$$\mathbb{z}^\circ_m = \hat{\mathbb{z}}^{(r,j)}_{m|m} \tag{4.50}$$

and at the first iteration we set

$$\hat{\mathbb{z}}_{m|m}^{(1,j)} = \hat{\mathbb{z}}_{m|m-1}^{(j)}. \tag{4.51}$$

Let $\hat{\mathbb{y}}_m^{(r,j)}$ and $\hat{\mathbf{\Sigma}}_{\mathbb{y}_m}^{(r,j)}$ respectively represent the observation and the corresponding covariance matrix predicted based on $\hat{\mathbb{z}}_{m|m}^{(r,j)}$. Then from (4.3), (4.35) and (4.21), $\hat{\mathbb{y}}_m^{(r,j)}$ and $\hat{\mathbf{\Sigma}}_{\mathbb{y}_m}^{(r,j)}$ can be obtained by

$$\hat{\mathbb{y}}_m^{(r,j)} = \mathbb{G}_m \hat{\mathbb{z}}_{m|m-1}^{(j)} + \mathbb{H}_m \boldsymbol{\mu}_{\mathbb{n}_m} + \mathbb{q}_m + \boldsymbol{\mu}_{\mathbb{v}} \tag{4.52}$$

$$\hat{\mathbf{\Sigma}}_{\mathbb{y}_m}^{(r,j)} = \mathbb{G}_m \hat{\mathbf{\Sigma}}_{\mathbb{z}_{m|m-1}}^{(j)} \mathbb{G}_m' + \mathbb{H}_m \mathbf{\Sigma}_{\mathbb{n}_m} \mathbb{H}_m' + \mathbf{\Sigma}_{\mathbb{v}} \tag{4.53}$$

where

$$\boldsymbol{\mu}_{\mathbb{n}_m} = \begin{bmatrix} \boldsymbol{\mu}_{\mathbf{n}_{1,m}} \\ \boldsymbol{\mu}_{\mathbf{n}_{2,m}} \\ \vdots \\ \boldsymbol{\mu}_{\mathbf{n}_{N,m}} \end{bmatrix}, \tag{4.54}$$

$$\mathbf{\Sigma}_{\mathbb{n}_m} = \begin{bmatrix} \mathbf{\Sigma}_{\mathbf{n}_{1,m}} & \mathrm{Cov}(\mathbf{n}_{1,m},\mathbf{n}_{2,m}) & \cdots & \mathrm{Cov}(\mathbf{n}_{1,m},\mathbf{n}_{N,m}) \\ \mathrm{Cov}(\mathbf{n}_{2,m},\mathbf{n}_{1,m}) & \mathbf{\Sigma}_{\mathbf{n}_{2,m}} & \cdots & \mathrm{Cov}(\mathbf{n}_{2,m},\mathbf{n}_{N,m}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{Cov}(\mathbf{n}_{N,m},\mathbf{n}_{1,m}) & \mathrm{Cov}(\mathbf{n}_{N,m},\mathbf{n}_{2,m}) & \cdots & \mathbf{\Sigma}_{\mathbf{n}_{N,m}} \end{bmatrix}, \tag{4.55}$$

$$\boldsymbol{\mu}_{\mathbb{v}} = \begin{bmatrix} \boldsymbol{\mu}_{\mathbf{v}_1} \\ \boldsymbol{\mu}_{\mathbf{v}_2} \\ \vdots \\ \boldsymbol{\mu}_{\mathbf{v}_N} \end{bmatrix}, \quad \mathbf{\Sigma}_{\mathbb{v}} = \begin{bmatrix} \mathbf{\Sigma}_{\mathbf{v}_1} & \mathbf{O}_Q & \cdots & \mathbf{O}_Q \\ \mathbf{O}_Q & \mathbf{\Sigma}_{\mathbf{v}_2} & \cdots & \mathbf{O}_Q \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O}_Q & \mathbf{O}_Q & \cdots & \mathbf{\Sigma}_{\mathbf{v}_N} \end{bmatrix}. \tag{4.56}$$

Once these are completed, the innovation $\mathbb{e}_m^{(r,j)}$ and its covariance matrix $\mathbf{R}_{\mathbb{e}_m}^{(r,j)}$ are computed

$$\mathbb{e}_m^{(r,j)} = \mathbb{y}_m - \hat{\mathbb{y}}_m^{(r,j)} \tag{4.57}$$

$$\mathbf{R}_{\mathbb{e}_m}^{(r,j)} = \mathbb{G}_m \hat{\mathbf{\Sigma}}_{\mathbb{z}_{m|m-1}}^{(j)} \mathbb{G}_m' + \mathbb{H}_m \mathbf{\Sigma}_{\mathbb{n}_m} \mathbb{H}_m' + \mathbf{\Sigma}_{\mathbb{v}} \tag{4.58}$$

and the Kalman gain $\mathbb{K}_m^{(r,j)}$ is obtained as follows:

$$\mathbb{K}_m^{(r,j)} = \hat{\mathbf{\Sigma}}^{(j)}_{\mathbb{z}_{m|m-1}} \mathbb{G}'_m (\mathbf{R}^{(r,j)}_{\mathbb{e}_m})^{-1}. \tag{4.59}$$

With $\mathbb{e}_m^{(r,j)}$, $\mathbf{R}^{(r,j)}_{\mathbb{e}_m}$ and $\mathbb{K}_m^{(r,j)}$, we can update the center of Taylor series expansion in (4.50) by means of the conventional measurement-update scheme

$$\hat{\mathbb{z}}^{(r+1,j)}_{m|m} = \hat{\mathbb{z}}^{(j)}_{m|m-1} + \mathbb{K}_m^{(r,j)} \mathbb{e}_m^{(r,j)}. \tag{4.60}$$

After R iterative linearization and Kalman update, we can compute the mean vector and covariance matrix of the posterior distribution $p(\mathbb{z}_m|\gamma_m = j, \mathbb{y}_0^m)$ by

$$\hat{\mathbb{z}}^{(j)}_{m|m} = \hat{\mathbb{z}}^{(R+1,j)}_{m|m} \tag{4.61}$$

$$\hat{\mathbf{\Sigma}}^{(j)}_{\mathbb{z}_{m|m}} = (\mathbf{I}_{Q(L+1)(N+1)} - \mathbb{K}_m^{(R,j)} \mathbb{G}_m) \hat{\mathbf{\Sigma}}^{(j)}_{\mathbb{z}_{m|m-1}}. \tag{4.62}$$

4) Background noise estimation: Recall that the local clean trajectory and N log frequency responses are jointly handled as the state vector. At that time, the background noise is treated to the fixed estimated value at (4.35). In order to estimate the background noise, we assume that the background noise evolves according to the following process:

$$\mathbb{n}_m = \mathbb{n}_{m-1} + \mathbf{w}_m \tag{4.63}$$

where $\mathbf{w}_m$ is a zero mean i.i.d. Gaussian process with its covariance matrix $\mathbf{\Sigma}_{\mathbf{w}_m}$. The estimation of the background noise can be easily accomplished by an additional Kalman filtering using the statistics $\hat{\mathbb{z}}^{(j)}_{m|m}$ and $\hat{\mathbf{\Sigma}}^{(j)}_{\mathbb{z}_{m|m}}$ of the estimated state vector in (4.61) and (4.62). Using these statistics of the state vector, we can recalculate the observation and the corresponding covariance matrix of (4.52) and (4.53), which are denoted by $\check{\mathbb{y}}_m^{(r,j)}$ and $\check{\boldsymbol{\Sigma}}^{(r,j)}_{\mathbb{y}_m}$ as following

$$\check{\mathbb{y}}_m^{(r,j)} = \tilde{\mathbb{G}}_m \hat{\mathbb{z}}^{(j)}_{m|m} + \tilde{\mathbb{H}}_m \boldsymbol{\mu}_{\mathbb{n}_m} + \tilde{\mathbb{q}}_m + \boldsymbol{\mu}_{\mathbb{v}} \tag{4.64}$$

$$\check{\boldsymbol{\Sigma}}^{(r,j)}_{\mathbb{y}_m} = \tilde{\mathbb{G}}_m \hat{\boldsymbol{\Sigma}}^{(j)}_{\mathbb{z}_{m|m}} \tilde{\mathbb{G}}'_m + \tilde{\mathbb{H}}_m \boldsymbol{\Sigma}_{\mathbb{n}_m} \tilde{\mathbb{H}}'_m + \boldsymbol{\Sigma}_{\mathbb{v}} \tag{4.65}$$

Once these are completed, the innovation $\tilde{\mathbb{e}}_m^{(r,j)}$ and its covariance matrix $\tilde{\mathbf{R}}^{(r,j)}_{\tilde{\mathbb{e}}_m}$ are recomputed

$$\tilde{\mathbb{e}}_m^{(r,j)} = \mathbb{y}_m - \check{\mathbb{y}}_m^{(r,j)} \tag{4.66}$$

$$\tilde{\mathbf{R}}^{(r,j)}_{\tilde{\mathbb{e}}_m} = \tilde{\mathbb{G}}_m \hat{\boldsymbol{\Sigma}}^{(j)}_{\mathbb{z}_{m|m}} \tilde{\mathbb{G}}'_m + \tilde{\mathbb{H}}_m \boldsymbol{\Sigma}_{\mathbb{n}_m} \tilde{\mathbb{H}}'_m + \boldsymbol{\Sigma}_{\mathbb{v}}. \tag{4.67}$$

Let $\boldsymbol{\mu}^{(j)}_{\mathbb{n}_m}$ and $\boldsymbol{\Sigma}^{(j)}_{\mathbb{n}_m}$ be j-th mean and variance of background noise at m-th frame. The Kalman gain $\mathbb{K}^{(j)}_{\mathbb{n}_m}$ for background noise can be written by

$$\mathbb{K}^{(j)}_{\mathbb{n}_m} = \boldsymbol{\Sigma}^{(j)}_{\mathbb{n}_m} \tilde{\mathbb{H}}'_m (\tilde{\mathbf{R}}^{(R,j)}_{\tilde{\mathbb{e}}_m})^{-1}. \tag{4.68}$$

Then noise parameters are updated by means of the conventional measurement-update scheme

$$\hat{\boldsymbol{\mu}}^{(j)}_{\mathbb{n}_m} = \boldsymbol{\mu}^{(j)}_{\mathbb{n}_m} + \mathbb{K}^{(j)}_{\mathbb{n}_m} \tilde{\mathbb{e}}_m^{(R,j)} \tag{4.69}$$

$$\hat{\boldsymbol{\Sigma}}^{(j)}_{\mathbb{n}_m} = (\mathbf{I}_{QN} - \mathbb{K}^{(j)}_{\mathbb{n}_m} \tilde{\mathbb{H}}_m) \boldsymbol{\Sigma}^{(j)}_{\mathbb{n}_m}. \tag{4.70}$$

5) Post-processing: Let $P^{(j)}_{m|m}$ denote the a posteriori model probability. Then it can be computed as follows:

$$\begin{aligned} P^{(j)}_{m|m} &= \frac{1}{c} p(\mathbb{y}_m | \hat{\mathbb{z}}^{(j)}_{m|m-1}, \hat{\boldsymbol{\Sigma}}^{(j)}_{\mathbb{z}_{m|m-1}}) p(\gamma_m = j | \mathbb{y}_0^{m-1}) && (4.71) \\ &= \frac{1}{c} \mathcal{N}(\mathbb{y}_m; \hat{\mathbb{y}}_m^{(1,j)}, \hat{\boldsymbol{\Sigma}}^{(1,j)}_{\mathbb{y}_m}) P^{(j)}_{m|m-1} && (4.72) \end{aligned}$$

where the normalizing constant c is computed from

$$c = \sum_{j=0}^{K-1} \mathcal{N}(\mathbb{y}_m; \hat{\mathbb{y}}_m^{(1,j)}, \hat{\boldsymbol{\Sigma}}^{(1,j)}_{\mathbb{y}_m}) P^{(j)}_{m|m-1}. \tag{4.73}$$

The mean vector and covariance matrix of the posterior distribution $p(\mathbb{z}_m|\mathbb{y}_0^m)$ are computed through the linear combination of the state vectors obtained from all the mixture components.

$$\hat{\mathbb{z}}_{m|m} = \sum_{j=0}^{K-1} P^{(j)}_{m|m}\hat{\mathbb{z}}^{(j)}_{m|m} \tag{4.74}$$

$$\hat{\mathbf{\Sigma}}_{\mathbb{z}_{m|m}} = \sum_{j=0}^{K-1} P^{(j)}_{m|m}\big[\hat{\mathbf{\Sigma}}^{(j)}_{\mathbb{z}_{m|m}} + (\hat{\mathbb{z}}_{m|m} - \hat{\mathbb{z}}^{(j)}_{m|m})(\hat{\mathbb{z}}_{m|m} - \hat{\mathbb{z}}^{(j)}_{m|m})'\big]. \tag{4.75}$$

The mean vector and covariance matrix of the background noise are also calculated from model combination as given by

$$\hat{\boldsymbol{\mu}}_{\mathbb{n}_m} = \sum_{j=0}^{K-1} P^{(j)}_{m|m}\hat{\boldsymbol{\mu}}^{(j)}_{\mathbb{n}_m} \tag{4.76}$$

$$\hat{\mathbf{\Sigma}}_{\mathbb{n}_m} = \sum_{j=0}^{K-1} P^{(j)}_{m|m}\big[\hat{\mathbf{\Sigma}}^{(j)}_{\mathbb{n}_m} + (\hat{\boldsymbol{\mu}}_{\mathbb{n}_m} - \hat{\boldsymbol{\mu}}^{(j)}_{\mathbb{n}_m})(\hat{\boldsymbol{\mu}}_{\mathbb{n}_m} - \hat{\boldsymbol{\mu}}^{(j)}_{\mathbb{n}_m})'\big]. \tag{4.77}$$

6) Clean feature estimation: After the post-processing step, we can obtain the estimate for not only the clean speech feature but also RIR since

$$\hat{\mathbb{z}}_{m|m} = [\hat{\mathbb{x}}_{m|m}\ \hat{\mathbb{h}}_{1,m|m}\ \hat{\mathbb{h}}_{2,m|m} \cdots \hat{\mathbb{h}}_{N,m|m}]' \tag{4.78}$$

where $\hat{\mathbb{x}}_{m|m} = [\hat{\mathbf{x}}'_{m|m}\ \hat{\mathbf{x}}'_{m-1|m}\ \cdots\ \hat{\mathbf{x}}'_{m-L|m}]$ denotes the $(L+1)$ consecutive clean feature vectors given observations $\mathbf{y}_0^m$. The $\hat{\mathbf{x}}_{m|m}$ means the filtered estimate and all other $\hat{\mathbf{x}}_{m-l|m}$ for $l > 0$ are smoothed estimates. Since we are assuming the causal system, we choose the filtered estimate.

4.5 Incremental State Estimation

When we generally compare the single-channel techniques with multi-channel techniques, one of the issues is the computational complexity. As the number of

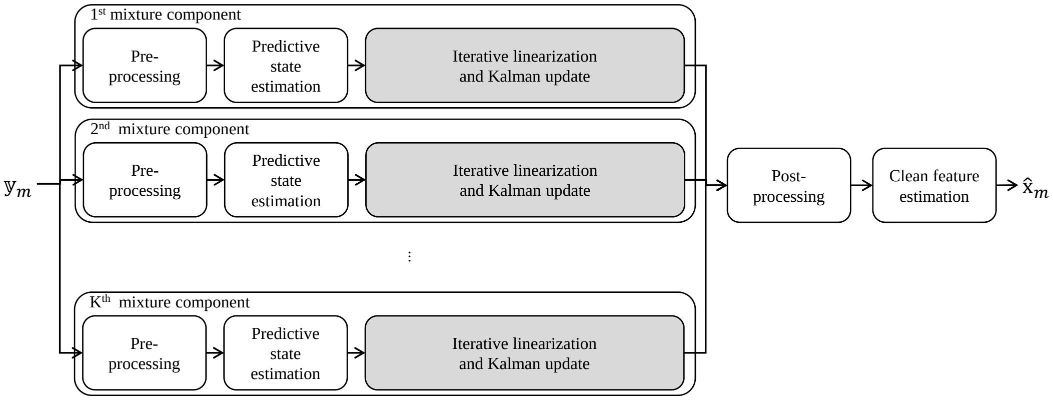
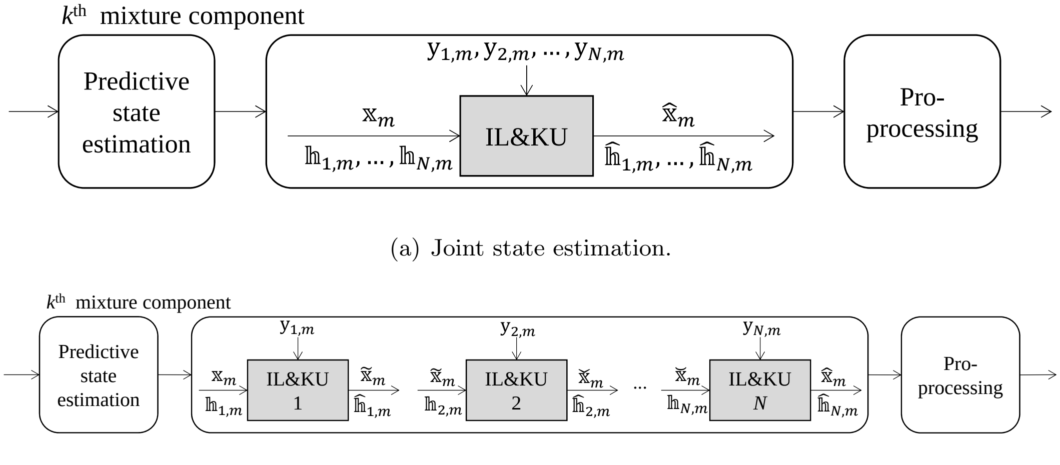


Figure 4.2: Feature enhancement algorithm.



(a) Joint state estimation.

(b) Incremental state estimation.

Figure 4.3: Types of state estimation of Kalman filter at k-th mixture component (IL&KU: iterative linearization and Kalman update).

channels grows, the computational complexity is also increased. In our case, when we consider all the observations simultaneously, the computational complexity is increased due to the large dimension of the state vector. To alleviate this computational complexity problem and make the algorithm more reasonable to be used in real applications, we propose a new state estimation method of the Kalman filtering motivated by [25]. Let the new state estimation method denote a incremental state estimation.

Basically, the feature enhancement procedures for the multi-channel technique are described in Fig. 4.2. The iterative linearization and Kalman update step requires a large amount of calculation according to the extended state vector. In Fig. 4.2, the iterative linearization and Kalman update step includes the background noise estimation step. Let the state estimate method in Fig. 4.2 denote a joint state estimation to distinguish it from the incremental state estimation. The main computation difference between the joint and incremental state estimation is occurred by modifying the iterative linearization and Kalman update step. The Fig. 4.3 shows the relationship between inputs and outputs of the iterative linearization and Kalman update step according to the joint and incremental state estimation. The joint state estimation is to update the estimates $[\hat{\mathbb{x}}_m \; \hat{\mathbb{h}}_{1,m} \; \hat{\mathbb{h}}_{2,m} \cdots \hat{\mathbb{h}}_{N,m}]$ of the state vector $\mathbb{z}_m$ using all the observations $[\mathbf{y}_{1,m}, \mathbf{y}_{2,m}, \cdots, \mathbf{y}_{N,m}]$ at one time while the incremental state estimation is to update the each estimate $[\hat{\mathbb{x}}_m \; \hat{\mathbb{h}}_{i,m}]$ of the state vector $\mathbb{z}_m$ according to the channel input $\mathbf{y}_{i,m}$ separately. Especially, in the incremental state estimation, the previous clean estimate $[\tilde{\mathbb{x}}_m, \breve{\mathbb{x}}_m, \cdots, \breve{\mathbb{x}}_m]$ of the Kalman filtering is used as an input of the next Kalman filtering and eventually the clean speech $\mathbb{x}_m$ is updated N times using all the observations. The log frequency response $\mathbb{h}_{i,m}$ is updated only once with the corresponding channel input. The reason why we can apply the incremental state estimation is that our main concern is only the clean speech component of the state vector and we can use the highly correlated sample data to find the optimal value of the clean speech at current frame.

Our multi-channel technique based on the mixture of multiple linear dynamic

Table 4.1: Average word accuracies (%) of the baseline system according to the distances and separation angles in the reverberation environment.

Reverb. (ms)	SNR (dB)					
	reverb.	20	15	10	5	0
300	96.34	81.75	64.43	39.82	21.87	14.39
500	84.28	60.17	43.20	26.37	16.75	12.03
700	70.57	45.42	31.87	20.43	14.28	10.50

models requires quite large computational complexity according to the several parameters such as K, L and N. As the value of the K grows, the computational complexity is increased by approximately K times to process the dynamic models. The feature dimension of the state vector is $(N+1)Q(L+1)$. In our algorithm, we assume that the clean speech and RIR are independent each other [7]. Given N channels, the log frequency response of the state vector is increased to $NQ(L+1)$ length vector. When we consider the increasing amount of computation according to the many matrix multiplications, the computational complexity of the joint state estimation is in proportion to about $\{(N+1)Q(L+1)\}^3$. On the other hand, the computational complexity of the incremental state estimation is in proportion to about $N\{2Q(L+1)\}^3$. From this analysis, we can know that the proposed incremental state estimation is more efficient than the joint state estimation in terms of computation.

Table 4.2: Average word accuracies (%) of the proposed system according to the distances and separation angles in the reverberation environment.

Reverb. (ms)	Method	SNR (dB)											
		reverb.		20		15		10		5		0	
		JNT	*INC*	*JNT*	*INC*	*JNT*	*INC*	*JNT*	*INC*	*JNT*	*INC*	*JNT*	*INC*
300	*IMM-1ch*	96.76	96.76	94.40	94.40	90.89	90.89	83.64	83.64	71.44	71.44	55.22	55.22
	IMM-2ch	96.42	96.74	95.52	95.38	93.37	93.38	88.63	88.81	79.02	78.98	63.92	63.81
	IMM-4ch	97.13	96.92	96.24	95.68	94.29	94.20	90.04	90.66	81.05	81.90	66.42	67.92
	IMM-8ch	97.38	97.59	96.67	96.20	95.20	94.98	91.87	91.96	84.44	83.54	71.60	69.95
500	*IMM-1ch*	82.98	82.98	81.44	81.44	76.87	76.87	69.73	69.73	59.11	59.11	46.05	46.05
	IMM-2ch	81.15	82.67	84.57	83.98	81.89	81.15	76.53	75.82	67.52	66.58	54.24	53.85
	IMM-4ch	85.07	83.53	87.35	84.95	84.78	82.52	79.56	77.92	70.50	69.48	57.15	58.01
	IMM-8ch	86.93	87.46	88.28	86.84	86.15	84.80	81.91	80.02	74.17	71.00	62.14	59.37
700	*IMM-1ch*	66.86	66.85	67.07	67.07	63.13	63.13	57.52	57.52	49.38	49.38	39.72	39.72
	IMM-2ch	63.83	66.43	71.34	70.91	68.92	67.92	64.33	63.22	56.75	55.95	46.53	46.35
	IMM-4ch	69.39	67.84	74.85	72.52	72.41	69.76	67.94	65.61	60.41	59.13	49.34	50.23
	IMM-8ch	72.32	73.75	76.87	75.44	74.72	72.78	70.66	68.20	63.84	60.84	53.99	51.45

4.6 Experiments

4.6.1 Simulation Data

The proposed approach was applied to a connected digit recognition task using TI digits corpus of which data set was usually used for feature compensation experiments. In our implementation, we employed the conventional front-end feature specified in the ETSI standard [31] and cepstral mean normalization was also applied. The baseline system of ASR was configured by using the scripts provided in [32], which was implemented by HTK software [33] for training and decoding. We assumed the clean training condition for the acoustic model of speech recognition in accordance with our purpose of estimating the clean feature vectors. The close talk baseline performance of the recognition system was obtained to the word accuracy 99.21 %.

The configuration for a reverberant noisy environment was shown in Fig. 2.3. In order to simulate various environmental conditions, the target speaker was located at nine positions according to the distances and separation angles. The RIRs were simulated by Allen and Berkley's image method [7] using Habets's software [8]. The reverberation range was varied from 300 to 700 ms and all the conditions were tested in the SNR range 0 to 20 dB. For the results of single-channel techniques, the microphone was positioned at the center of microphone 1 and 2. Due to the computational issue, we used $L = 2$ and $K = 16$ for all the following experiments. For convenience, we denote the baseline, single-, two-, three-, four- and eight-channel algorithm by *BAS*, *IMM-1ch*, *IMM-2ch*, *IMM-3ch*, *IMM-4ch* and *IMM-8ch*, respectively. All the experiments were performed according to the joint and incremental state estimations of Kalman filtering, which was denoted by *JNT* and *INC*, respectively. We compared our proposed multi-channel algorithm with the single-channel algorithm.

The Table 4.1 and 4.2 show the average word accuracies of the baseline and proposed system according to the distances and separation angles in the reverberation environment. The word accuracy is defined by

$$\text{Word accuracy (\%)} = \frac{\mathrm{N} - D - I - S}{\mathrm{N}} \times 100 \tag{4.79}$$

where N, D, I and S indicate the number of total words, word deletion errors, word insertion errors and word substitution errors, respectively. Here, *reverb.* indicates a noise-free condition which has only reverberation. The proposed methods

Table 4.3: Word accuracies (%) of the baseline and single-channel algorithm in the car environment.

	APL	*SVL*	*SVC*	*SVR*
Baseline	52.48	64.14	63.84	54.21
IMM-1ch	94.02	95.39	95.15	93.98

outperformed single-channel algorithm in many conditions. The *JNT* was better performance than *INC* in several conditions but the performance differences were quite comparable.

4.6.2 Live Recording Data

For the real environment test, we used a car environment speech corpora created by the SiTEC in the 1st year of the 5-year project for creation and distribution of standarized speech corpora [34]. The data consisted of spoken Korean words by 100 speakers recorded through 7 microphones and one handsfree simultaneously in driving environments. The target words were represented by car control or navigation commands, single digits, and connected digits with total 1451 words. We used the left, center, right microphones positioned at a sun visor with distance about 15 cm and one microphone attached at A-pillar of the passenger seat. For convenience, those microphones were denoted by *APL*, *SVL*, *SVC* and *SVR*, respectively. The acoustic model based on tied-state triphone was trained with SiTEC clean Korean word DB which was uttered by 407 people. Each state had eight Gaussian mixtures. The same feature as the TI digits corpus was exploited. The close talk baseline performance of the recognition system was obtained to the word accuracy 91.69 %.

Table 4.3 shows the word accuracies of the baseline and single-channel algorithm

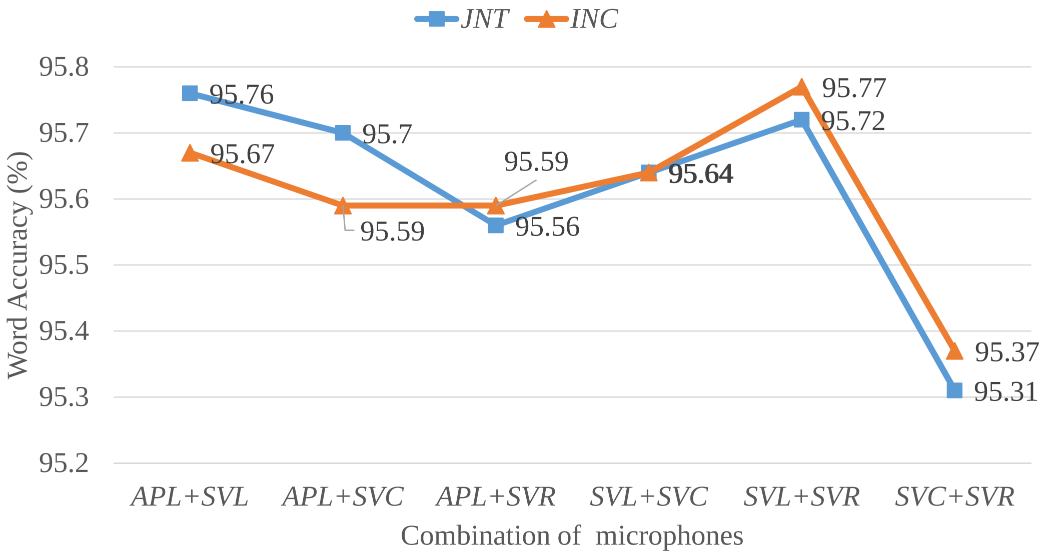


Figure 4.4: Word accurarices (%) on *IMM-2ch* according to different microphone combinations.

in the car environment. Since we assumed the hands-free scenario, four microphones *APL*, *SVL*, *SVC* and *SVR* were only considered. Fig. 4.4 and 4.5 show the word accuracies of the *IMM-2ch* and *IMM-3ch* according to the different microphone combinations, respectively. From the results, we can see that better performance was obtained when using *APL* and *SVL* microphones due to their high correlation.

4.6.3 Computational Complexity

In this work, we compared the proposed joint and incremental state estimation methods. The real time factors (RTFs) were measured using a server with Intel Xeon 3.33 GHz processor and 48 GB memory running on Ubuntu Linux 10.04 LTS. In Fig. 4.6, the RTFs of the joint and incremental state estimation methods are shown. The *INC* shows suitable results in spite of increase in the number of channels.

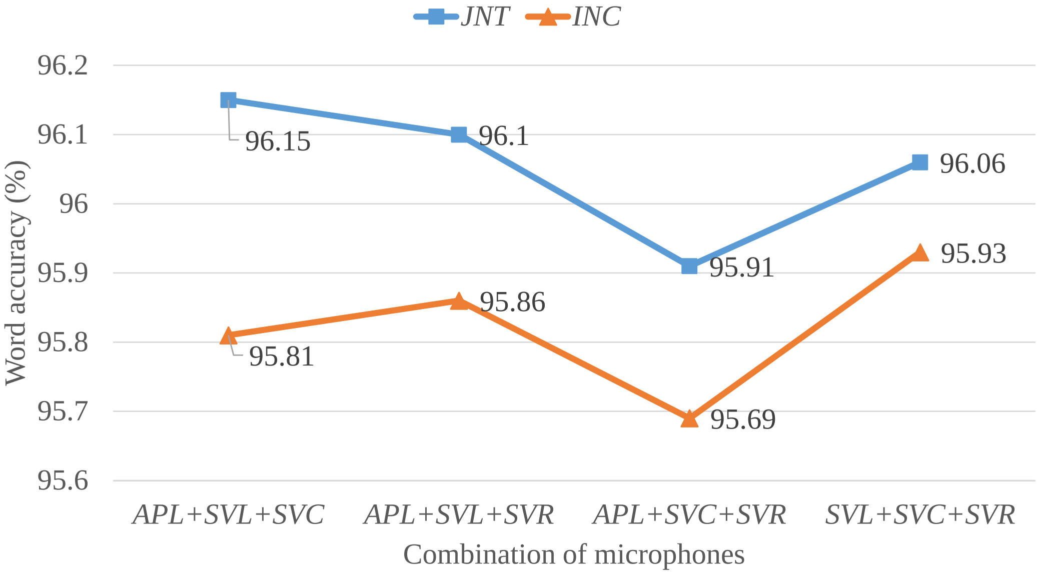


Figure 4.5: Word accurarices (%) on *IMM-3ch* according to different microphone combinations.

4.7 Summary

In this chapter, we have proposed a novel approach to estimate the clean feature vectors in multi-channel environment, which was obtained by extending the single-channel IMM algorithm to a multi-channel version. To reduce the computational complexity of the multi-channel technique, a new state estimation method of the Kalman filtering also has been described. From various experiments in reverberant noisy environments and real environment, it has been confirmed that the proposed algorithm has shown adequate results in terms of word accuracy and computational complexity.

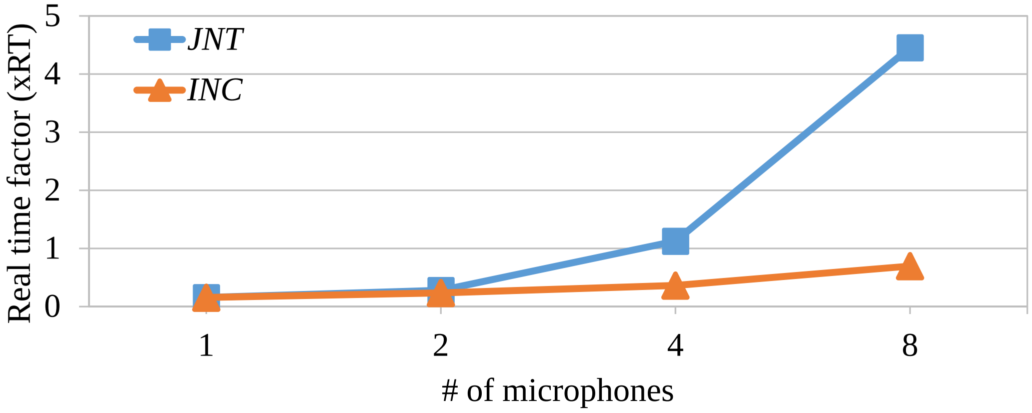


Figure 4.6: Real time factors according to the microphones.

Chapter 5

Supervised Denoising Pre-Training for Robust ASR with DNN-HMM

5.1 Introduction

Recently, DNNs have become one of the most popular techniques in the vast field of machine learning. Due to their powerful capability in nonlinear description between the input and the target values, the DNNs have outperformed many other conventional techniques in various tasks. This DNN's capability has also been applied to the environment-robust techniques for ASR. Particularly, in the robust ASR area, conventional environment-robust techniques usually necessitate some specific models or formulations to account for the nonlinear relationship between the clean and noisy speech processes in an appropriate signal domain. In constrast, the DNNs have the advantage that they can directly learn an arbitrary unknown relationship between the input and the target values without any specific assumption. Consequently, they have brought a better performance gain than the conventional approaches. A more complicated input-target relationship can be easily learned by using wider and deeper neural network architectures with a sufficient amount of training data.

Since DNN is a highly nonlinear and non-convex model, its performance usually depends on the initial parameter setting for training. This issue has been possibly resolved through a number of unsupervised or supervised pre-training methods. For the unsupervised methods, generative pre-training algorithm for the RBMs (RBM) [13], greedy layer-wise unsupervised pre-training using autoencoder [35] and stacked denoising autoencoder (SDAE) [36], [37] were proposed. A core idea of these algorithms is to learn a nonlinear representation of the input data one level at a time using unsupervised feature learning. In the case of SDAE, the pre-training module takes the noisy features as an input and then tries to recover the original clean features by minimizing the cross-entropy loss or the squared error loss between the reconstructed features and the original clean features.

In the class of the supervised methods, greedy layer-wise supervised pre-training (GLPT) [35] and discriminative pre-training (DPT) [38] methods were proposed. These methods first train the DNN with one hidden layer using the target labels discriminatively, then insert another hidden layer between the trained hidden layer and the output layer and again discriminatively train the network to convergence. This procedure is repeated until the desired number of hidden layers are all trained. A hybrid pre-training algorithm combining RBM and GLPT was also introduced [39]. These pre-training techniques can potentially bring the DNN weights to a relatively good initial point for converging to a better local optimum.

The above mentioned pre-training techniques can be also applied to robust ASR. In order to initialize the DNN in adverse environments, not only the clean features but also the corrupted features are used as an input of the DNN, which is common in the robust ASR area. In a sense, this approach can be regarded as a multi-condition training technique. The parameters of the DNN are learned to describe the hidden representation of the multi-condition data set. As the depth of the DNN gets deeper, more abstract features can appear at higher layers. More abstract concepts are generally considered more robust to most local variations of the inputs. Learning these invariant features has been a long-standing goal in pattern recognition [40].

In this chapter, we propose a novel supervised denoising pre-training technique for the DNN-HMM system for robust speech recognition in adverse environments. In the proposed approach, our aim is to initialize the DNN parameters such that they yield abstract features robust to acoustic environment variations. In order to achieve this, we first derive the abstract features from an early fine-tuned DNN model which is trained based on a clean speech database. By using the derived abstract features as the target values, the standard error back-propagation algorithm with the stochastic gradient descent method is performed to estimate the initial parameters of the DNN. The performance of the proposed algorithm was evaluated on Aurora-4 DB, and better results were observed compared to a number of conventional pre-training methods.

5.2 Deep Neural Networks

DNN is a multi-layer perceptron network with many hidden layers. A DNN consists of input, hidden and output layers as shown in Fig. 5.1. For simplicity, we denote the input layer as layer 0 and the output layer as layer L for an $(L+1)$-layer DNN.

The hidden representation of the DNN at the l-th layer can be written by

$$\mathbf{v}^l = \sigma(\mathbf{z}^l) = \sigma(\mathbf{W}^l \mathbf{v}^{l-1} + \mathbf{b}^l), \text{ for } 0 < l < L \tag{5.1}$$

where $\mathbf{v}^l = [v_1^l \ v_2^l \ \cdots \ v_{N_l}^l]'$, $\mathbf{z}^l = \mathbf{W}^l \mathbf{v}^{l-1} + \mathbf{b}^l = [z_1^l \ z_2^l \ \cdots \ z_{N_l}^l]'$, $\mathbf{W}^l$, $\mathbf{b}^l = [b_1^l \ b_2^l \ \cdots \ b_{N_l}^l]'$ and N_l denote the activation vector, excitation vector, weight matrix with size $N_l \times N_{l-1}$, bias vector and the number of neurons at the l-th layer, respectively. Here, the prime denotes the transpose of a vector or a matrix. In (5.1), $\sigma(x) = 1/(1+e^{-x})$ is the sigmoid function which is usually employed as an activation function in many applications. The function $\sigma(\cdot)$ is applied to the excitation vector element-wisely. At the 0-th layer, $\mathbf{v}^0 = [v_1^0 \ v_2^0 \ \cdots \ v_{N_0}^0]'$ is the input vector and N_0 is the input feature dimension.

The data type at the output layer is decided based on the target task. For a multi-class classification task, each output neuron represents a class membership for which the softmax function is applied to $\mathbf{z}^L$ as follows:

$$v_i^L = \text{softmax}_i(\mathbf{z}^L) = \frac{e^{z_i^L}}{\sum_{j=1}^{N_L} e^{z_j^L}} \tag{5.2}$$

$$\sum_{i=1}^{N_L} v_i^L = 1 \tag{5.3}$$

where v_i^L, z_i^L and N_L indicate the i-th component of the output activation, the i-th component of the excitation vector and the number of classes at the output layer, respectively.

For supervised fine-tuning, a labeled training set $(\mathbf{o}, \mathbf{d}) = \{(o_t, d_t) | 1 \leq t \leq T\}$ is needed where o_t represents the t-th observation vector, $d_t = [d_{t,1} \ d_{t,2} \ \cdots \ d_{t,N_L}]'$ is the corresponding target vector with size N_L and T denotes the number of training

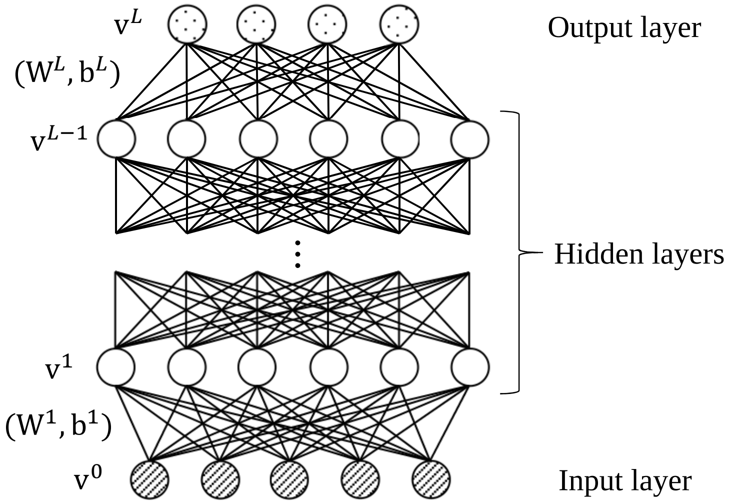


Figure 5.1: DNN structures.

samples. The DNN input $\mathbf{v}_t^0 = [v_{t,1}^0 \; v_{t,2}^0 \; \cdots \; v_{t,N_0}^0]'$ at time t usually consists of a number of concatenated observation vectors. During fine-tuning, the DNN parameters are updated by using the back-propagation procedure according to a proper objective function. For multi-class classification, the cross-entropy (CE) is usually adopted as an objective function as given by

$$J_{CE} = \frac{1}{T} \sum_{t=1}^{T} \left[- \sum_{i=1}^{N_L} d_{t,i} \log(v_{t,i}^L) \right] \tag{5.4}$$

where $d_{t,i}$ and $v_{t,i}^L$ indicate the i-th component of the desired target value and the i-th component of the generated DNN output value given the t-th observation. Basically, $d_{t,i}$ can be regarded as the posterior probability of the i-th output class.

5.3 Supervised Denoising Pre-Training

In this section, we propose a novel approach to initialize the DNN parameters particularly for robust ASR. The proposed approach is called a supervised denoising pre-training technique. In the proposed technique, the initial parameters of the DNN for noisy inputs are learned so as to describe the most abstract features obtained when the corresponding clean features are applied. If this is achieved, the DNN is capable of extracting abstract representations, i.e., hidden node values robust against the interfering noises.

For this approach, we need an auxiliary DNN with the same structure, which is fully-trained based solely on a clean speech database. It can be obtained using a set of clean training data and the corresponding target labels through the procedure described in Section 5.2. The hidden nodes of this auxiliary DNN are considered to form abstract representations of the clean speech features which are not affected by interfering noises. Since the nodes of the top hidden layer are considered to possess the most abstract characteristics of the clean speech features, we only focus on the top hidden layer of the auxiliary DNN when creating the target abstract representation in this work.

Let $\{\mathbf{W}_c^l, \mathbf{b}_c^l | (\mathbf{o}^c, \mathbf{d})\}$ denote the parameters of the auxiliary DNN estimated from a clean training data $\mathbf{o}^c$ with the corresponding target labels $\mathbf{d}$. Also let $\{\mathbf{W}_m^l, \mathbf{b}_m^l | (\mathbf{o}^m, \mathbf{d})\}$ be the parameters of the main DNN which will be trained based on a multi-condition data $\mathbf{o}^m$ with the target labels $\mathbf{d}$. Note that $\mathbf{o}^c$ and $\mathbf{o}^m$ form a stereo database, i.e., simultaneous recordings obtained in both the clean and corrupted conditions, and the desired target labels are the same in both data. In our approach to pre-training, the parameters $\{\mathbf{W}_m^l, \mathbf{b}_m^l\}$ are initialized such that they yield the abstract representation at the $(L-1)$-th hidden layer as close as possible to those obtained at the same layer of the auxiliary DNN which was fed with clean speech feature. Providing the last hidden layer values of the auxiliary DNN as the target enables to estimate $\{\mathbf{W}_m^l, \mathbf{b}_m^l\}$ with the use of the back-propagation algorithm. For back-propagating the errors between the activated values obtained from the main DNN and the target abstract features derived from the auxiliary DNN, we employ the mean square error (MSE) as the objective function. If the number of training samples is T, the objective function J_{MSE} is given by

$$J_{MSE} = \frac{1}{T}\sum_{t=1}^{T}\left[\frac{1}{2}||\mathbf{v}_{m,t}^{L-1} - \mathbf{v}_{c,t}^{L-1}||^2\right] \tag{5.5}$$

where $\mathbf{v}_{m,t}^{L-1}$ and $\mathbf{v}_{c,t}^{L-1}$ respectively indicate the t-th activation vectors obtained from the main and auxiliary DNN at the $(L-1)$-th layer, and $||\cdot||$ means Euclidean norm. It is very important to note that $\mathbf{v}_{m,t}^{L-1}$ in (5.5) is derived from $\mathbf{o}^m$ while $\mathbf{v}_{c,t}^{L-1}$ is derived from the auxiliary DNN when the clean speech feature $\mathbf{o}^c$ is applied. The proposed method can be modified to reproduce all the hidden node activations of the auxiliary DNN by treating each hidden activation as the target value. From a number of preliminary experiments, we have found that using only the last hidden layer as the target values shows a slightly better performance than using all the hidden layers.

After $\{\mathbf{W}_m^l, \mathbf{b}_m^l\}$ are initialized as above, a usual discriminative fine-tuning algorithm is performed with the objective function of (5.4) as described in Section 5.2.

5.4 Experiments

The performance of the proposed method was evaluated on Aurora-4 DB which is widely used in the robust speech recognition area. The proposed method was compared with the following conventional pre-training approaches: RBM [13], GLPT [35], DPT [38] and SDAE [37]. DPT is similar to GLPT but differs in that the latter only updates the newly added hidden layer while in the former all layers are jointly updated each time when a new hidden layer is added. The performance is measured with word error rate which is defined by

$$\text{Word error rate (\%)} = \frac{D + I + S}{\text{N}} \times 100 = 100 - Acc \tag{5.6}$$

where N, D, I, S and Acc indicate the number of total words, word deletion errors, word insertion errors, word substitution errors and word accuracy, respectively.

5.4.1 Feature Extraction and GMM-HMM System

We used the Kaldi speech recognition toolkit [41] for feature extraction, acoustic modeling of ASR, DNN training and ASR decoding. The feature was extracted with the default configuration of Kaldi. According to that configuration, 23-dimensional log mel filterbank (LMFB) features were calculated and 13-dimensional MFCCs (including C_0) with their first and second derivatives were extracted for the GMM-HMM recognizer. The cepstral mean normalization algorithm was applied for each speaker.

In order to provide the target alignment information for the discriminative DNN training, we built a clean-condition GMM-HMM system with 2009 senones and 15028 Gaussian mixtures in total. The target senone labels of the DNN-HMM system were obtained over the clean-condition training data. As for the language model, we applied the standard 5k open bi-gram model for decoding.

5.4.2 DNN Structures

For the auxiliary and main DNN training, we applied five hidden layers with 2048 nodes as proposed in [43]. As for the input features of the DNNs, we used the LMFB features due to their good performance demonstrated in the previous studies. The input features consisted of 11 frames (5 frames on each side of the current frame) context window of 23 dimensional LMFB features with their first and second order derivatives, which resulted in the input dimension of 759. The input features of the DNNs were normalized to have zero mean and unit variance.

For training the auxiliary DNN using the clean-condition training data, RBM was carried out to initialize the DNN parameters as described in [42] which is based on a greedy layerwise fashion [13]. The Gaussian-Bernoulli RBM was trained with an initial learning rate of 0.01 and the Bernoulli-Bernoulli RBMs with a rate of 0.4. The initial RBM weights were randomly drawn from a Gaussian $\mathcal{N}(0, 0.01)$; the hidden biases of Bernoulli units as well as the visible biases of the Gaussian units were initialized to zero, while the visible biases of the Bernoulli units were initialized as $log(p/1-p)$, where p was the mean output of a Bernoulli unit from previous layer. During pre-training, the momentum m was set to 0.9, which was accompanied by a rescaling of the learning rate using $1 - m$. Also the L2 regularization was applied to the weights, with a penalty factor of 0.0002. For the supervised fine-tuning, the initial learning rate of 0.008 with the same 256 minibatch size as the pre-training was used for the DNN training. The errors between the DNN outputs and the target senone labels were calculated according to (5.4).

For initializing the main DNN parameters using the proposed method, RBM was first conducted using the multi-condition training data for the main DNN and then supervised fine-tuning was performed using the abstract features derived from the auxiliary DNN as the target values with the initial learning rate of 0.0001. The errors between the last hidden node activations of the main DNN and the target abstract features derived from the auxiliary DNN were calculated as in (5.5). After initializing the parameters of the main DNN and adding an output layer on the top

Table 5.1: WERs (%) on the auxiliary DNN-HMM system.

Method	Set_A	Set_B	Set_C	Set_D	Average
RBM	7.12	47.55	42.91	65.98	52.23

Table 5.2: WERs (%) on the main DNN-HMM system according to various pre-training methods.

Method	Set_A	Set_B	Set_C	Set_D	Average
SDAE	7.77	11.88	12.42	23.72	16.70
DPT	8.00	11.92	12.68	23.29	16.57
GLPT	7.83	11.73	11.97	23.02	16.31
RBM	7.81	11.71	12.27	22.71	16.18
SDPT	7.42	10.93	11.86	22.56	**15.73**

of the network, the discriminative fine-tuning with the senone targets was performed with the initial learning rate of 0.008. In order to speed up training, we applied the learning rate scheduling scheme and stop criteria presented in [42].

5.4.3 Performance Evaluation

We compared our proposed method with the conventional pre-training approaches on Aurora-4 DB. For convenience, the proposed method was denoted by SDPT when demonstrating the experimental results. Table 5.1 shows the word error rates (WERs) obtained with the auxiliary DNN-HMM system which was used to generate the target abstract features. Table 5.2 shows the WERs of the main DNN-HMM system built with various pre-training techniques. From the results, we can see that SDPT outperformed all the other pre-training techniques in all the tested conditions. Consequently, the DNN parameters initialized by SDPT allowed the model to converge a better local optimum in adverse environments.

5.5 Summary

In this chapter, we have proposed a novel supervised denoising pre-training technique for the DNN robust to noisy input variations. The initial parameters of the DNN was obtained from the supervised training using the back-propagation algorithm. The target values were calculated from the auxiliary DNN which was fine-tuned using the clean training data and the corresponding target labels. From the experimental results, we have found that the proposed method was effective for enhancing the recognition performance in adverse conditions.

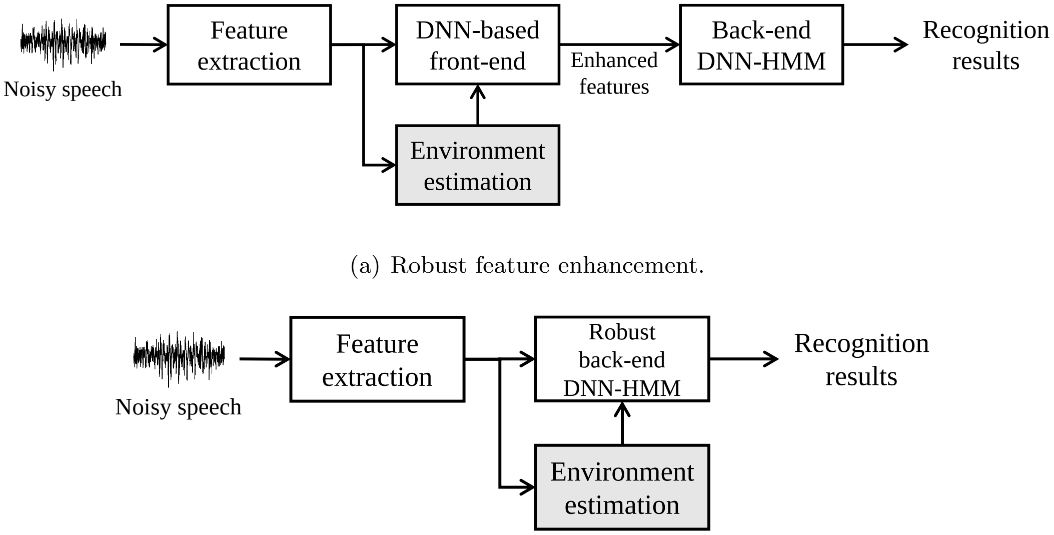
Chapter 6

DNN-Based Frameworks for Robust Speech Recognition Using Noise Estimates

6.1 Introduction

Recently, ASR has achieved a great success with the aid of the DNNs. The most salient feature of the DNN is its ability to learn an arbitrary unknown mapping from the input to the target values automatically. Interest in this DNN's capability has been also expanded to the area of robust speech recognition. The DNN-based algorithms have shown better performances recently than the conventional noise processing algorithms such as the speech enhancement, feature compensation and model adaptation techniques.

Despite its big success, the DNN still has some problems to be solved. First, since the DNN is a highly nonliner and non-convex model, its performance usually depends



(a) Robust feature enhancement.

(b) Robust model training.

Figure 6.1: DNN-based frameworks for robust ASR.

on the initial parameter setting for training. Second, the DNN has a training-test mismatch problem, i.e., the performance usually deteriorates for unseen data at the test stage. To alleviate these problems, pre-training with denoising autoencoder [35] and adaptation techniques such as noise-aware training [43] have been proposed. The former can potentially bring the DNN weights to a relatively good initial point for converging to a better local optimum by reconstructing the input from a corrupted version of it. The latter gives the background noise information as an additional input to the network such that the DNN training algorithm can automatically figure out how to adjust the model parameters to exploit the noise information in unknown conditions [44].

The DNN-based robust ASR can be implemented in two different ways as shown in Fig. 6.1: robust feature enhancement (RFE) and robust model training (RMT). In the scenario of RFE, DNN is applied to map the noisy input features to their uncorrupted clean versions without modifying the back-end recognizer which is assumed to be a DNN-HMM system in this work. On the other hand, in the RMT scheme, the parameters of the back-end DNN-HMM recognizer are directly learned to describe the relations between the input features and the desired target values while applying not only the clean speech features but also the corrupted features as the training data. In a sense, this approach can be regarded as a multi-condition training technique which is common in the area of robust ASR. Even though it has been generally known that the RMT approach usually outperforms the corresponding feature enhancement technique, the latter has the advantage that it can be performed separately from the back-end recognizer.

In this chapter, new DNN-based robust ASR approaches using noise estimates are proposed and then applied to both the RFE and RMT scenarios. A novel part of the proposed approaches is that the time-varying noise estimates are applied to the DNN as additional inputs. For this, we extract noise estimates in a frame-by-frame manner from the IMM algorithm which has been known to show good performance in tracking slowly-varying background noise [10]. The performance of the proposed approaches is evaluated on Aurora-4 DB and better performance is observed compared to the conventional DNN-based robust speech recognition algorithms [43].

6.2 DNN-Based Frameworks for Robust ASR

Our focus in this work is the environments where additive background noises exist. It is assumed that the distorted input signal is captured by a single microphone. As for the input features of the DNN, we use the LMFB features due to their good performance demonstrated in the previous studies. As mentioned in Section 6.1, we consider two DNN-based robust ASR frameworks: RFE and RMT. Now we describe each of these two frameworks below.

6.2.1 Robust Feature Enhancement

As shown in Fig. 6.1(a), we can divide the recognition process into two parts: DNN-based front-end processing and back-end DNN-HMM recognition system. In the DNN-based front-end processing, the noisy LMFB features are first enhanced before being fed to the back-end DNN-HMM system. In the back-end DNN-HMM system, the enhanced features are used to estimate the posterior probabilities of the tied-state triphones (senones) which are then converted to the emission probabilities of the HMM states.

The DNN-based front-end processing can be treated as a regression task where the target values are the clean speech features. The proposed DNN structure is given in Fig. 6.2(a) where the noise estimates are applied as inputs to the DNN in conjunction with the noisy features. Through the stacked hidden layers with many nodes, the complex mappings between the input vectors and the desired target values, e.g., the clean LMFB features are automatically learned based on a set of training DB. The input vector $\mathbf{v}_t$ for this DNN structure can be configured as follows:

$$\mathbf{v}_t = [\mathbb{y}_{t-\tau}, \mathbb{y}_{t-\tau+1}, \cdots, \mathbb{y}_t, \hat{\mathbb{n}}_t] \tag{6.1}$$

where $\mathbb{y}_t$ denotes the noisy LMFB feature with its first and second order derivatives extracted at the t-th frame and τ indicates the context window size. For practical purposes, we select the input vectors such that they enable a causal processing of the data, i.e., the output at a time only depends on the present or past inputs. In (6.1), $\hat{\mathbb{n}}_t$ denotes the estimated noise feature with its first and second order derivatives.

In our work, the noise estimates are obtained from the IMM-based technique [10] which also estimates the background noise components in a causal manner. With the assumption of slowly time-varying noise environments, we can track the variation of the background noise in a frame-by-frame basis and the estimated noises are directly applied to the DNN as additional inputs. Interested readers are referred to [10] for further information of the IMM algorithm.

The structure of the DNN used for the back-end DNN-HMM system is shown in Fig. 6.2(b) where we can see that the DNN estimates the senone posterior probabilities from the enhanced speech features. The input vector $\mathbf{v}_t$ of this DNN can be configured as

$$\mathbf{v}_t = [\mathbb{x}_{t-\tau}, \mathbb{x}_{t-\tau+1}, \cdots, \mathbb{x}_t] \tag{6.2}$$

where $\mathbb{x}_t$ denotes the estimated clean LMFB feature with its first and second order derivatives at the t-th frame. The back-end DNN in this framework is usually trained over a clean speech DB only though re-training is also possible with the enhanced features.

6.2.2 Robust Model Training

In the RMT scheme, as shown in Fig. 6.1(b), the robust back-end DNN-HMM system can estimate the senone posterior probabilities directly from the noisy features. The structure of the DNN used in the RMT scheme is different from that of the back-end DNN-HMM in Fig. 6.1(a) in that noise estimates are additionally applied as inputs to the network. This means that the robust back-end DNN-HMM system of RMT directly uses noisy features while the back-end DNN-HMM system of RFE uses clean or enhanced features. Since the input variations of RMT are very

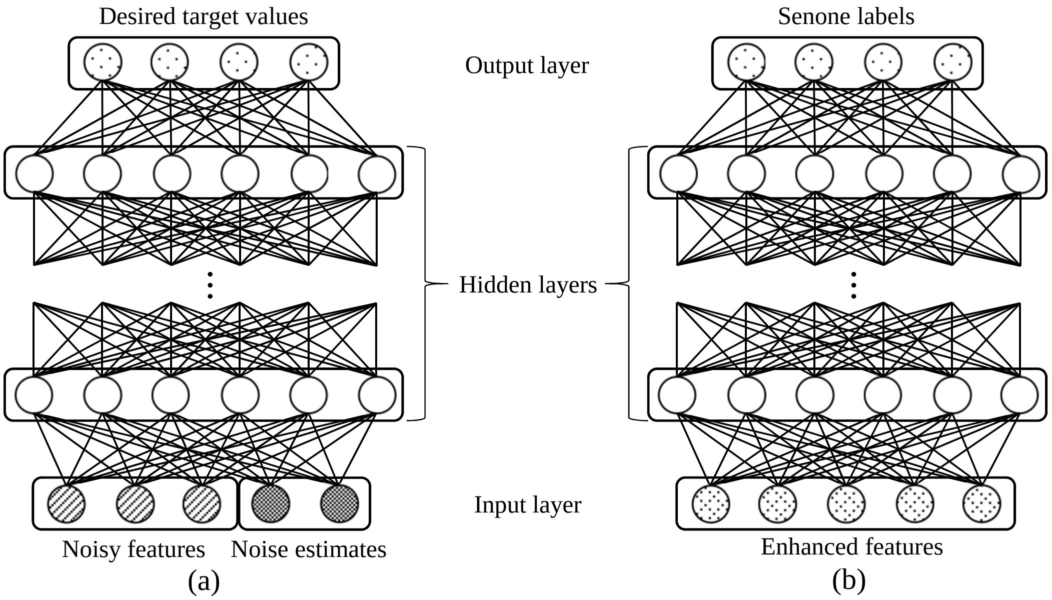


Figure 6.2: DNN structures (a) for the DNN-based front-end and robust back-end DNN-HMM system, (b) for the back-end DNN-HMM system.

diverse due to various noisy environments, the background noise estimates are considered to provide useful information concerned with the environment leading to a robust training of the DNN. The structure of the DNN is given in Fig. 6.2(a) where the target values now represent the senone posterior probabilities.

The advantage of RMT is its ability to learn arbitrary unknown relations between the noisy inputs and the senone labels directly using a deep network. Rather than two stage ASR as in the RFE scheme, lower recognition errors can be expected since a joint optimization of the parameters is performed instead of two separate optimizations.

6.3 IMM-Based Noise Estimation

In this section, we introduce the proposed noise estimation method based on IMM, which is the part of IMM-based feature compensation algorithm. The goal of IMM-based feature compensation technique is to estimate clean LMFB feature sequence $\mathbf{x}_0^T = [\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_T]$ given noisy LMFB feature sequence $\mathbf{y}_0^T = [\mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_T]$, where $x_{t1}^{t2} = [x_{t1}, x_{t1+1}, \cdots, x_{t2}]$ denotes a subsequence of vectors from frame index $t1$ to $t2$. Particularly, we consider a causal estimation scheme which is desired for sequential filtering. One of the most efficient methods for the causal estimation is the IMM technique which is based on the multiple dynamic models using Kalman filter. Specially, when the state transition and observation models for the multiple dynamic models evolved over frame are given, the states to be defined in the system are updated at all the dynamic models and propagated to the next frame.

In our work, the background noise feature is treated as a state vector. To estimate the noise features, we employ a single Gaussian background noise model as given by

$$\mathbf{n}_t \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{n}_t}, \boldsymbol{\Sigma}_{\mathbf{n}_t}\right) \tag{6.3}$$

where the mean vector $\boldsymbol{\mu}_{\mathbf{n}_t}$ and covariance matrix $\boldsymbol{\Sigma}_{\mathbf{n}_t}$ are unknown and should be estimated during the environment compensation procedure. As for the prior knowledge, the probability density function of the clean features is assumed as a mixture of Gaussian distributions such that

$$p(\mathbf{x}_t) = \sum_{k=1}^{N} p(k)\mathcal{N}\left(\mathbf{x}_t; \boldsymbol{\mu}_{\mathbf{x}_t}^k, \boldsymbol{\Sigma}_{\mathbf{x}_t}^k\right) \tag{6.4}$$

where $p(k)$, $\boldsymbol{\mu}_{\mathbf{x}_t}^k$, $\boldsymbol{\Sigma}_{\mathbf{x}_t}^k$ and N represent the weight, mean vector, covariance matrix of the k-th Gaussian distribution and the number of Gaussians, respectively.

We assume that the environmental characteristic of the background noise slowly changes among the consecutive frames. With this assumption, we employ the noise evolution process by

$$\mathbf{n}_{t+1} = \mathbf{n}_t + w_t \tag{6.5}$$

$$w_t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{w_t}) \tag{6.6}$$

where $\mathbf{0}$ and $\mathbf{\Sigma}_{w_t}$ indicate zero mean vector and fixed covariance matrix of Gaussian process w_t. To update the slowly time-varying background noise model parameters $\{\boldsymbol{\mu}_{\mathbf{n}_t}, \mathbf{\Sigma}_{\mathbf{n}_t}\}$, we can construct the N linear dynamic models according to the mixture components. Through the extended Kalman filtering approach based on IMM, the estimated noise model parameters $\{\hat{\boldsymbol{\mu}}_{\mathbf{n}_t}, \hat{\mathbf{\Sigma}}_{\mathbf{n}_t}\}$ are obtained at every frame. Interested readers are referred to [10] for further information.

The estimated noise means $\{\hat{\boldsymbol{\mu}}_{\mathbf{n}_t}\}$ are directly used to the DNN's additional inputs. In (6.1), $\hat{\mathbb{n}}_t$ is the feature including the dynamic component of $\hat{\boldsymbol{\mu}}_{\mathbf{n}_t}$.

6.4 Experiments

To confirm our proposed method, we evaluated the performance in noise matched and mismatched conditions. The noise matched conditions were obtained from Aurora-4 DB. The noise mismatched conditions were made using 100 nonspeech environmental sounds [45]. The features used in the experiments and the GMM-HMM system were described in Section 5.4.1.

6.4.1 DNN Structures

For performance comparison, we also implemented the DNN-based robust ASR technique proposed in [43] which used fixed noise estimates obtained from the initial duration of each utterance. We compared three different DNN input vectors: no noise estimate (NSY), fixed noise estimate (FIXN) and the noise estimates obtained from the IMM algorithm (ESTN). We used the context window size $\tau = 5$, which resulted in the input dimensions of 414, 483 and 483 for NSY, FIXN and ESTN, respectively. All the input features were normalized to have zero mean and unit variance.

For the DNN-based front-end, we applied three hidden layers with 2048 nodes as in [46]. For the back-end DNN-HMM system of RFE and RMT, seven hidden layers with 2048 nodes each were adopted as proposed in [47]. In RFE, 23-dimensional clean LMFB features were applied as the target vectors while 2009 senone labels were applied as the target values in the back-end DNN-HMM systems. Generative pre-training using the restricted Boltzmann machines [13] was carried out to initialize the DNN parameters as described in Section 5.4.2.

For supervised fine-tuning, the initial learning rates 0.001 and 0.008 with the same minibatch size as the pre-training were used for DNN-based front-end and the DNN training of all the back-end DNN-HMM systems respectively. For speeding up the training, we applied the learning rate scheduling and stop criteria as described in [42].

6.4.2 Performance Evaluations

In order to evaluate our proposed method, we also made the noise mismatched test sets with the clean speech of Set_A and Set_C on Aurora-4 DB. Six noises were chosen from 100 noise types: machine noise, siren, animal sound, water sound, wind and phone dialing. After downsampling to 16 kHz and applying P. 341 filter to maintain the same configuration as the Aurora-4 environment, the noises were added to the data in Set_A and Set_C at SNRs between 5 and 15 dB to form new

Table 6.1: WERs (%) of RFE according to the training conditions and input types.

Condition	Input types	Set_A	Set_B	Set_C	Set_D	Average
Clean	*Baseline*	7.23	52.47	43.53	69.33	55.83
	NSY	7.57	14.56	15.39	30.41	20.95
	FIXN	7.57	15.72	16.46	32.23	22.26
	ESTN	7.44	13.47	15.64	29.04	19.86
Re-trained multi	NSY	7.77	12.42	13.51	27.03	18.43
	FIXN	7.75	13.66	13.99	29.89	20.22
	ESTN	8.22	12.36	14.53	26.56	**18.31**

Table 6.2: WERs (%) of RMT according to the input types.

Condition	Input types	Set_A	Set_B	Set_C	Set_D	Average
Multi	NSY	7.96	12.45	13.79	26.35	18.18
	FIXN	8.33	13.63	14.55	28.31	19.61
	ESTN	7.64	12.22	13.21	25.56	**17.68**

Set_B and new Set_D, respectively.

First, we evaluated the performance in the RFE framework. The WERs obtained with different DNN input types and training conditions are shown in Table 6.1. The back-end DNN-HMM systems in the RFE framework were trained either over the clean-condition training set only or over all the multi-condition training set after being enhanced by the DNN-based front-end processing. We differentiate these two training conditions by denoting them respectively as clean-condition and re-trained multi-condition in the table. The *Baseline* in Table 6.1 means the baseline performance of the clean-condition back-end DNN-HMM system without any DNN-based

Table 6.3: WERs (%) of RMT on noise mismatched test conditions.

Condition	Input types	New Set_B	New Set_D	Average
Multi	NSY	27.50	43.08	35.29
	FIXN	30.99	45.26	38.13
	ESTN	24.95	39.16	**32.05**

front-end processing. From the results we can see that the DNN input type of ESTN, which applied the noise estimates obtained from the IMM algorithm, outperformed all the other input types. In addition, re-trained multi-condition training with the enhanced features further improved the performance of the ASR system.

Next, we evaluated the performance in the RMT framework. This time the back-end DNN-HMM systems were trained over all the multi-condition training set with different DNN input configurations. The results are shown in Table 6.2 from which we can see that the DNN input configuration of ESTN outperformed all the other input types. This observation once again confirms that the incorporation of the time-varying noise estimates obtained from the IMM algorithm is useful to provide the environment-related information to the DNN.

Finally, we evaluated the performance on the noise mismatched conditions. In this work, we just checked the WERs in the RMT framework. Table 6.3 shows the results according to the noise mismatched conditions in the RMT scenario. We can see that ESTN outperformed the other techniques and it is more effective in noise mismatched conditions.

6.5 Summary

In this chapter, we have proposed to incorporate the noise estimates in the DNN inputs for robust ASR. From the experimental results, we have found that the noise estimates obtained from the IMM algorithm are useful to enhance the recognition performance in both the RFE and RMT scenarios. Especially, we found that the proposed techniques were more effective to enhance the recognition performance in the noise mismatched condition.

Chapter 7

DNN-Based Robust Speech Recognition Using Soft Target Labels

7.1 Introduction

Recently, ASR has achieved a great success with the aid of the DNNs. The most salient feature of the DNN is its ability to automatically learn an arbitrary unknown mapping from the input to the target values. Interest in this DNN's capability has been also expanded to the area of robust speech recognition. The DNN-based algorithms have recently shown better performances than the conventional noise processing algorithms such as speech enhancement, feature compensation and model adaptation techniques. The most prominent advantage of this approach is that the parameters of the DNN-HMM recognizer can be directly trained to describe the relations between the possibly corrupted input features and the desired target values while applying not only the clean speech features but also the corrupted features as the training data. In a sense, this approach can be regarded as a multi-condition training technique which is common in the area of robust ASR.

The supervised learning algorithms used in the DNN framework usually require a large amount of labeled training data. Obtaining the correct labels is important for those training algorithms. If each training instance has multiple semantic meanings, multi-labels can be also applied to account for them, which has been tried in several application areas, e.g., image, text and language processing [48]. In the large vocabulary ASR area, the DNN estimates the posterior probabilities of the tied context-dependent acoustic states (senones) from the acoustic observations. In order to provide the target values for the DNN training, state level labels are usually obtained from the Viterbi alignment using the reference word transcript and a well-trained GMM-HMM system [15], [42], [44]. Since the quality of the target labels can affect the performance of the DNN training, it is important to train a good GMM-HMM system for state alignment. In [49], to further enhance the forced alignment, realignment is performed iteratively using the updated DNN-HMM parameters.

The state level alignments usually provide the senones' posterior probabilities. When applied to the DNN training, these are converted to the hard target labels, which means that the target value corresponding to the most probable senone is set to one and all the others are fixed to zero. In contrast, we can consider an alternative approach in which soft labels are used based on the senones' posterior probabilities. Soft labeling means that each target value is not restricted to 0 or 1 but takes non negative values in (0,1) and their sum equals 1. In adverse conditions, confining the target values only to a single senone is considered not proper since the distinction between different acoustic units becomes more ambiguous as the speech is degraded more severely.

In this chapter, we propose a DNN training method using soft target labels instead of hard labeling. The usage of the soft target labels at the DNN training can be also regarded as taking advantage of the correlations among the target labels. In a sense, this approach can be seen as a kind of structured learning technique [50]. In our work, the soft target labels are obtained from the forward-backward algorithm well-known in HMM training as applied in the handwriting recognition task [51]. The proposed method makes the DNN training be more robust in noisy and unseen conditions. The performance of the proposed approach was evaluated on Aurora-4 DB and various noise mismatched test conditions, and better results were observed compared to the conventional hard target labeling method.

7.2 DNN-HMM Hybrid System

The aim of a speech recognition system is to find the most likely word sequence $\mathbf{w} = \{w_1, ..., w_N\}$ given an observation sequence $\mathbf{x} = \{x_1, ..., x_T\}$, which can be formulated as

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmax}} \; P(\mathbf{w}|\mathbf{x}). \tag{7.1}$$

By applying Bayes rule and considering that the word sequence is independent of the marginal distribution of the observations $p(\mathbf{x})$, (7.1) can rewritten as

$$\begin{aligned} \hat{\mathbf{w}} &= \underset{\mathbf{w}}{\operatorname{argmax}} \; \frac{p(\mathbf{x}|\mathbf{w})P(\mathbf{w})}{p(x)} & (7.2) \\ &= \underset{\mathbf{w}}{\operatorname{argmax}} \; p(\mathbf{x}|\mathbf{w})P(\mathbf{w}) & (7.3) \end{aligned}$$

where $P(\mathbf{w})$ is the prior probability of a particular word sequence provided by a language model, and $p(\mathbf{x}|\mathbf{w})$ is calculated from an acoustic model. HMMs have been one of the most popular and successful acoustic models to date. The HMM parameters are basically estimated according to the maximum likelihood criterion. Given HMM parameters λ, (7.2) is converted to

$$\hat{\mathbf{w}} = \operatorname*{argmax}_{\mathbf{w}} p(\mathbf{x}|\mathbf{w}, \lambda)P(\mathbf{w}). \tag{7.4}$$

Since the observation sequence is generated along a state sequence $\mathbf{q} = \{q_1, ..., q_T\}$ of the HMM, the likelihood $p(\mathbf{x}|\mathbf{w}, \lambda)$ is calculated as an expectation over all possible hidden state sequences $\mathbf{q}$ associated with the hyphothesis $\mathbf{w}$

$$\begin{aligned} p(\mathbf{x}|\mathbf{w}, \lambda) &= \sum_{\mathbf{q}} p(\mathbf{x}, \mathbf{q}|\mathbf{w}, \lambda) & (7.5) \\ &= \sum_{\mathbf{q}} p(\mathbf{x}|\mathbf{q}, \mathbf{w}, \lambda)p(\mathbf{q}|\mathbf{w}, \lambda) & (7.6) \\ &= \sum_{\mathbf{q}} \pi_{q_1} \prod_{t=2}^{T} a_{q_{t-1}q_t} \prod_{t=1}^{T} p(x_t|q_t) & (7.7) \end{aligned}$$

where π_{q_t}, $a_{q_{t-1}q_t}$ and $p(x_t|q_t)$ denote the initial state probability, state transition probability between states q_{t-1} and q_t, and observation probability at state q_t for the observation x_t respectively.

In (7.7), the state likelihood $p(x_t|q_t)$ has been usually computed using GMM until recently. In DNN-HMM hybrid systems, this likelihood is obtained from the DNN. After the posterior probability $p(q_t|x_t)$ is estimated, it is converted to the state likelihood $p(x_t|q_t)$ after being divided by the prior probability $p(q_t)$. The DNN-HMM hybrid system takes advantage of the DNN's strong representation learning power and HMM's sequential modeling ability, and outperforms the conventional GMM-HMM system significantly in many large vocabulary continuous speech recognition tasks [15].

In DNN training, in order to estimate $p(q_t|x_t)$, the state level label information corresponding to the observation at each frame is needed. In the common framework for DNN-HMM training, the maximum likelihood of the observation sequence given one hidden state sequence is used to approximate the marginal likelihood over all possible state sequences and this process is called the Viterbi algorithm, i.e.,

$$p(\mathbf{x}|\mathbf{w},\lambda)P(\mathbf{w}) \approx \max_{\mathbf{q}} p(\mathbf{x},\mathbf{q}|\mathbf{w},\lambda)P(\mathbf{w}). \tag{7.8}$$

Given a training data and the corresponding transcript, 1-best state sequence is obtained and then hard target label according to each observation is applied as the target for the supervised learning of the DNN.

7.3 Soft Target Label Estimation

The core idea proposed in this work is to apply soft labels when training the DNN instead of the conventional hard labeling scheme. In previous ASR studies, in order to obtain the state level labels, the Viterbi alignment is used and it is one of the best alternatives for getting the labels in the sequential data. When we consider only one instance, the most probable state at that time cannot be chosen after back-tracking in the Viterbi algorithm. In the proposed approach, we regard all possible states at each instance.

The soft labels are derived from the posterior probability $\gamma_j(t)$ of the state j at time t given the observation sequence $\mathbf{x}$, correct word sequence $\mathbf{w}$ and HMM parameters λ. From the basic HMM formulation, it is easy to see that

$$\begin{aligned}\gamma_j(t) &= P(q_t = j|\mathbf{x},\mathbf{w},\lambda) && (7.9)\\ &= \sum_{\mathbf{q}\setminus q_t} P(q_1,\cdots,q_{t-1},q_t = j,q_{t+1},\cdots,q_T|\mathbf{x},\mathbf{w},\lambda) && (7.10)\end{aligned}$$

where $\mathbf{q}_{\setminus q_t} = \{q_1, \cdots, q_{t-1}, q_{t+1}, \cdots, q_T\}$ denotes a subsequence excluding q_t. This probability can be calculated from the well-known forward-backward algorithm as follows:

$$\gamma_j(t) = \frac{\alpha_j(t)\beta_j(t)}{p(\mathbf{x}|\mathbf{w}, \lambda)},\ j \in Q \tag{7.11}$$

where

$$\alpha_j(t) = p(x_1, \cdots, x_t, q_t = j|\mathbf{w}, \lambda), \tag{7.12}$$

$$\beta_j(t) = p(x_{t+1}, \cdots, x_T|q_t = j, \mathbf{w}, \lambda), \tag{7.13}$$

$$\sum_{j=1}^{Q} \gamma_j(t) = 1 \tag{7.14}$$

with Q denoting the set of all HMM states.

The posterior probabilities calculated from (7.11)-(7.14) can be directly used for the target values of the DNN in the soft labeling scheme. Similar approaches have been proposed in the sequence-discriminative training methods [42], [52], [53]. The aim of these approaches is to minimize the sentence level errors at the training step. When comparing the reference transcript with the competing hypotheses, all possible state sequences for the reference word sequence can be represented using the forward-backward algorithm. In our work, we focus on minimizing the frame level errors.

In order to further investigate the effect of soft target labels, we employ a control parameter ξ with which the soft target labels are modified to $\gamma'_j(t)$ as

$$\gamma'_j(t) = \frac{\gamma_j^{\xi}(t)}{\sum_{j=1}^{Q} \gamma_j^{\xi}(t)}. \tag{7.15}$$

If $\xi \to \infty$, $\gamma'_j(t)$ becomes the hard target labels and $\xi \to 0$ means equally distributed soft target labels. Note that it is not equal to the forced aligned hard target labels using the Viterbi algorithm when $\xi \to \infty$.

7.4 Experiments

We compared our proposed method with the conventional hard labeling approach in noise matched and mismatched test conditions. The noise matched conditions were built using the Aurora-4 DB. The noise mismatched conditions were made using 100 nonspeech environmental sounds [45]. The features used in the experiments and the GMM-HMM system were described in Section 5.4.1. Furthermore, the performance evaluation when training the DNN with various noise conditions was also investigated.

7.4.1 DNN Structures

For the DNN-HMM hybrid system, we applied five hidden layers with 2048 nodes as proposed in [43]. As for the input features of the DNN, we used the LMFB features due to their good performance demonstrated in the previous studies. The input features consisted of 11 frames (5 frames on each side of the current frame) context window of 23 dimensional LMFB features with their first and second order derivatives, which resulted in the input dimension of 759. The input features were normalized to have zero mean and unit variance.

Generative pre-training using the restricted Boltzmann machines [13] was carried out to initialize the DNN parameters as described in [42]. For supervised fine-tuning, the initial learning rate 0.008 with the same 256 minibatch size as the pre-training was used for the DNN training. The error between DNN output and reference target value was calculated using the cross entropy criterion. The DNN parameters were updated by using the back-propagation algorithm with a stochastic gradient descent method. In order to speed up the training, we applied the learning rate scheduling

Table 7.1: WERs (%) on Aurora-4 DB.

Model	Label	Set_A	Set_B	Set_C	Set_D	Average
Clean DNN-HMM	HARD	7.12	47.55	42.91	65.98	52.23
	SOFT	7.12	47.33	43.88	65.68	**52.07**
Multi DNN-HMM	HARD	7.81	11.71	12.27	22.71	16.18
	SOFT	7.53	11.53	11.66	22.67	**16.03**

Table 7.2: WERs (%) on noise mismatched test conditions.

Model	Label	New Set_B	New Set_D	Average
Clean DNN-HMM	HARD	47.33	75.94	61.64
	SOFT	47.24	75.92	**61.58**
Multi DNN-HMM	HARD	21.62	37.76	29.69
	SOFT	19.72	36.13	**27.93**

and stop criteria as described in Section 5.4.2.

7.4.2 Performance Evaluation

In order to evaluate the performance of our proposed method, we used Aurora-4 DB and noise mismatched data which was described in 6.4.2. Tables 7.1 and 7.2 show the WERs on Aurora-4 DB and noise mismatched test conditions. For convenience, the labeling methods using the conventional Viterbi alignment and the proposed method were denoted by HARD and SOFT, respectively. In this experiments, we set ξ in (7.15) to 1.0 for deriving the soft labels. From the results, we can see that after applying the soft target labels for the DNN training, all the average WERs were improved. Particularly, in noise mismatched test conditions, multi-condition

Table 7.3: WERs (%) on Aurora-4 DB according to the control parameter ξ.

Model	ξ	Set_A	Set_B	Set_C	Set_D	Average
Multi DNN-HMM	0.75	7.62	11.46	11.92	22.88	16.11
	1.0	7.53	11.53	11.66	22.67	**16.03**
	1.25	7.70	11.65	12.25	22.87	16.22
	1.5	7.75	11.64	12.01	23.06	16.28

Table 7.4: WERs (%) on noise mismatched test conditions according to the control parameter ξ.

Model	ξ	New Set_B	New Set_D	Average
Multi DNN-HMM	0.75	19.43	35.99	27.71
	1.0	19.72	36.13	27.93
	1.25	18.89	36.01	**27.45**
	1.5	19.38	36.21	27.79

DNN-HMM system showed great performance improvement with the usage of soft labels.

7.4.3 Effects of Control Parameter ξ

The performance of soft labeling was further evaluated with various ξ in the multi-condition DNN-HMM system. The control parameter ξ was varied from 0.75 to 1.5. $\xi = 1.0$ means that the posterior probabilities obtained from the forward-backward algorithm were directly applied. Tables 7.3 and 7.4 respectively show the WERs on Aurora-4 DB and noise mismatched test conditions with different ξ. In Aurora-4 DB, the best performance was obtained with $\xi = 1.0$. On the other hand,

Table 7.5: WERs (%) on Aurora-4 DB in the multi-condition DNN-HMM system.

Input type	Pre-training	Target labels	Set_A	Set_B	Set_C	Set_D	Average
NSY	RBM	HARD	7.81	11.71	12.27	22.71	16.18
		SOFT	7.53	11.53	11.66	22.67	16.03
	SDPT	HARD	7.42	10.93	11.86	22.56	15.73
		SOFT	7.23	10.90	11.75	22.33	**15.60**
ESTN	RBM	HARD	7.66	11.51	11.66	22.52	15.96
		SOFT	7.94	11.69	11.97	22.45	16.06
	SDPT	HARD	7.49	11.12	11.62	22.59	15.81
		SOFT	7.34	11.04	12.24	22.31	15.69

in noise mismatched conditions, $\xi = 1.25$ resulted in the best performance.

7.4.4 An Integration with SDPT and ESTN Methods

In this section, we evaluated the performance of the integration of the initialization technique (SDPT) in Chapter 5, RMT using noise estimates (ESTN) in Chapter 6 and the soft target labeling technique (SOFT) in this work. For all the experiments about the DNN structures, the DNN input features and GMM-HMM system, etc., we used the same setting in this work.

Table 7.5 and 7.6 show the WERs on matched and mismatched conditions in the multi-condition DNN-HMM system. In Table 7.5, the combination of NSY, SDPT and SOFT shows better performance results. We found that SDPT played an important role in the DNN training. It allowed the initial parameters of the DNN to converge better local optimum given any input types. In Table 7.6, however, the initialization of the DNN training using SDPT brought worse performance than RBM.

Table 7.6: WERs (%) on noise mismatched conditions in the multi-condition DNN-HMM system.

Input type	Pre-training	Target labels	New Set_B	New Set_D	Average
NSY	RBM	HARD	21.62	37.76	29.69
		SOFT	19.72	36.13	27.93
	SDPT	HARD	24.55	39.67	32.11
		SOFT	23.19	38.72	30.96
ESTN	RBM	HARD	19.41	33.87	26.64
		SOFT	18.94	33.42	**26.18**
	SDPT	HARD	22.86	38.08	30.47
		SOFT	21.18	35.99	28.59

This means that if the noise mismatched data was used as an input of the DNN initialized using SDPT, the mismatched problem occurred in SDPT. In contrast, the combinations of ESTN and SOFT robust to the mismatched conditions shows the most performance in all the experiments.

7.4.5 Performance Evaluation on Various Noise Types

In order to investigate the performance of the DNN according to the number of noise types, we made new three training sets using clean-condition training data on Aurora-4 DB. The Set_1 consisted of clean speech, crowd, traffic and car, bell and cough noises. The Set_2 had clap, snore, yawn and cry noises in addition to the noised of the Set_1. The Set_3 had shower, tooth brushing, foot step and door moving noises in addition to the noised of the Set_2. To sum up, Set_1, Set_2 and Set_3 had 5, 9 and 13 conditions. The size of three training sets was the same as

Table 7.7: WERs (%) according to the DNN structures on validation data of Set_3.

Hidden layer depth	Validation set	Number of parameters (M)
3	31.20	14.07
4	30.75	18.26
5	30.69	22.46
6	30.25	26.65
7	30.22	30.85
8	30.23	35.05
9	30.45	39.24

the clean-condition training data on Aurora-4 DB. For the evaluation, mismatched noise condition New Set_B was used.

To choose the adequate DNN structure on new training data sets, we fist evaluated the performance according to the DNN structures using Set_3. For training and validation, Set_3 was divided into 90% and 10% of data. Table 7.7 shows the WERs according to the DNN structures on validation data of Set_3. In our case, we fixed the number of hidden nodes to 2048. The DNN with 7 hidden layers showed better performance results. As one layer is piled up, the parameters of the DNN increase. Efficiently to train the deep networks, a large amount of data is required for training.

Table 7.8, 7.9 and 7.10 show the performance results on mismatched noise conditions using Set_1, Set_2 and Set_3, respectively. The performance was evaluated using the proposed SDPT, ESTN and SOFT techniques. In Table 7.8, 7.9 and 7.10, IDEN means that the correct noise values are used in the DNN training instead of the fixed or time varying noise estimates. In Set_3, the overall performance became

Table 7.8: WERs (%) on noise mismatched conditions using Set_1.

Training set	Pre-training	Input type	Target labels	New Set_B	RERR (%)
Set_1	RBM	NSY	HARD	31.75	-
			SOFT	31.84	-0.28
		FIXN	SOFT	28.89	9.00
		ESTN	SOFT	25.46	19.81
		IDEN	SOFT	23.57	25.76
	SDPT	NSY	SOFT	35.13	-10.64
		FIXN	SOFT	30.51	3.90
		ESTN	SOFT	30.96	2.48
		IDEN	SOFT	23.59	25.70

better than that of Set_1 and Set_2. As the number of noise types increases, the performance results were better. The integrated solution of the proposed methods using the ESTN and SOFT methods showed better performance results. In case of IDEN, particularly, SDPT also showed better performance results on contrary to those in 7.6. The training data with many noise conditions made the DNN to have more generalization power. From all the performance results, we can see that the proposed methods were effective to improve the recognition performance.

7.4.6 DNN Training and Decoding Time

DNNs have many hidden layers each of which has many neurons. This greatly increases the total number of parameters in the model and slows down both the training and decoding. The training speed may be boosted by using high-performance computing devices such as general purpose graphical processing units (GPGPUs).

Table 7.9: WERs (%) on noise mismatched conditions using Set_2.

Training set	Pre-training	Input type	Target labels	New Set_B	RERR (%)
Set_2	RBM	NSY	HARD	23.34	-
			SOFT	22.33	4.33
		FIXN	SOFT	22.03	5.61
		ESTN	SOFT	21.22	9.08
		IDEN	SOFT	16.49	29.35
	SDPT	NSY	SOFT	25.48	-9.17
		FIXN	SOFT	24.94	-6.85
		ESTN	SOFT	23.52	-0.77
		IDEN	SOFT	16.23	30.47

To compare the RTFs between CPU and GPU, we exploited the five hidden layers with 2048 nodes. The dimension of the DNN input was 759 LMFB features and 2009 senones were used. Table 7.11 shows the RTFs of DNN training and decoding using CPU and GPU. The RTFs for CPU were measured using Intel Xeon E5-2620 2.4 GHz processor and 16 GB memory running on Ubuntu Linux 14.04 LTS. The RTFs for GPU were measured using GeForce GTX 980 with 2048 cores. From the results, we have found that GPU performs significantly faster than CPU and are preferred platforms for training DNNs.

7.5 Summary

In this chapter, we have proposed the DNN-based robust speech recognition approach using soft target labels. From the experimental results, we have found that the

Table 7.10: WERs (%) on noise mismatched conditions using Set_3.

Training set	Pre-training	Input type	Target labels	New Set_B	RERR (%)
Set_3	RBM	NSY	HARD	20.82	-
			SOFT	20.08	3.55
		FIXN	SOFT	20.14	3.27
		ESTN	SOFT	20.93	-0.53
		IDEN	SOFT	16.01	23.10
	SDPT	NSY	SOFT	23.16	-11.26
		FIXN	SOFT	22.96	-10.25
		ESTN	SOFT	21.84	-4.88
		IDEN	SOFT	13.91	33.17

Table 7.11: RTFs of DNN training and decoding using CPU and GPU.

	Training (1 epoch)	Decoding
CPU (xRT)	0.833021	0.842849
GPU (xRT)	0.006995	0.325861

soft target labels were useful to enhance the recognition performance. Particularly, great performance improvement has been observed in unseen test conditions. The combinations of the SDPT and ESTN methods were also investigated. In the noise matched condition, SDPT brought better performance. In the noise mismatched conditions, the combination of ESTN and SOFT showed better performance results. Furthermore, the performance evaluation when training the DNN with various noise conditions was also investigated. The proposed methods have been effective to enhance the recognition performance once again.

Chapter 8

Conclusions

In this thesis, model-based and data-driven approaches for the environment-robust speech recognition have been proposed. The sequential characteristic of the speech was modeled by HMM. According to the way to calculate the emission probabilities of the HMM, the type of the speech recognition decoder was divided into GMM-HMM and DNN-HMM systems. In the GMM-HMM system, the acoustic model was trained using the clean-condition training data and model-based technique was proposed in order to match the reverberant noisy input features with the characteristic of the trained acoustic model. In the DNN-HMM system, the DNN was trained using the multi-condition training data to obtain the relationship between the input and the target labels. In accordance with these concepts, we proposed four techniques for the environment-robust speech recognition.

Firstly, we have proposed a novel approach to estimate the clean feature vectors in multi-channel environment, which was obtained by extending the single-channel IMM algorithm to a multi-channel version. To reduce the computational complexity of the multi-channel technique, a new state estimation method of the Kalman filtering also has been described. From various experiments in reverberant noisy environments and real environment, it has been confirmed that the proposed algorithm has shown adequate results in terms of accuracy and computational complexity.

Secondly, we have proposed a novel supervised denoising pre-training technique for the DNN robust to noisy input variations. The initial parameters of the DNN was obtained from the supervised training using the BP algorithm. The target values was calculated from the auxiliary DNN which was fine-tuned using the clean training data and the corresponding target labels. From the experimental results, we have found that the proposed method was effective to enhance the recognition performance in adverse conditions.

Thirdly, we have proposed to incorporate the noise estimates in the DNN inputs for robust ASR. From the experimental results, we have found that the noise estimates obtained from the IMM algorithm are useful to enhance the recognition performance in both the RFE and RMT scenarios. Especially, we found that the proposed technique was more effective to enhance the recognition performance in the noise mismatched condition. Especially, ESTN method brought better performance results in the noise mismatched conditions.

Finally, we have proposed the DNN-based robust speech recognition approach using soft target labels. From the experimental results, we have found that the soft target labels were useful to enhance the recognition performance. Particularly, great performance improvement has been observed in unseen test conditions. Furthermore, the combinations of the SDPT and ESTN methods were investigated. In the noise matched condition, SDPT brought better performance. In the noise mismatched conditions, the combination of ESTN and SOFT showed better performance results.

Bibliography

[1] M. Miyoshi and Y. Kaneda, “Inverse filtering of room acoustics,” *IEEE Trans. Acoust. Speech Signal Process.*, vol. 36, no. 2, pp. 145–152, Feb. 1988.

[2] H. G. Hirsch and H. Finster, “A new approach for the adaptation of hmms to reverberation and background noise,” *Speech Commun.*, vol. 50, no. 3, pp. 244–263, Mar. 2008.

[3] S. Markovich, S. Gannot, and I. Cohen, “Multichannel eigenspace beamforming in a reverberant noisy environment with multiple interfering speech signals,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 17, no. 6, pp. 1071–1086, Aug. 2009.

[4] M. Souden, J. Benesty, and S. Affes, “On optimal frequency-domain multichannel linear filtering for noise reduction,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 18, no. 2, pp. 260–276, Feb. 2010.

[5] R. Rotili, E. Principi, S. Cifani, S. Squartini, and F. Piazza, “Multichannel feature enhancement for robust speech recognition,” in *Speech Technologies*, I. Ipsic, Ed. India: InTech, 2011.

[6] R. G. Leonard, “A database for speaker-independent digit recognition,” in *Proc. ICASSP*, 1984, pp. 42.11.1–42.11.4.

[7] J. B. Allen and D. A. Berkley, “Image method for efficiently simulating small-room acoustics,” *J. Acoust. Soc. Amer.*, vol. 65, no. 4, pp. 943–950, Apr. 1979.

[8] E. Habets. (2010) Room impulse response (rir) generator. [Online]. Available: http://home.tiscali.nl/ehabets/rir_generator.html

[9] G. Hirsch, “Experimental framework for the performance evaluation of speech recognition front-ends on a large vocabulary task, version 2.0,” Tech. Rep., 2002.

[10] N. S. Kim, “Imm-based estimation for slowly evolving environments,” *IEEE Signal Process. Lett.*, vol. 5, no. 6, pp. 146–149, Jun. 1998.

[11] N. S. Kim, “Statistical linear approximation for environment compensation,” *IEEE Signal Process. Lett.*, vol. 5, no. 1, pp. 8–10, Jan. 1998.

[12] C. W. Han, S. J. Kang, and N. S. Kim, “Reverberation and noise robust feature compensation based on imm,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 21, no. 8, pp. 1598–1611, Aug. 2013.

[13] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[14] A. Mohamed, G. Dahl, and G. Hinton, “Acoustic modeling using deep belief networks,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 14–22, Jan. 2012.

[15] G. Dahl, D. Yu, L. Deng, and A. Acero, “Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 1, pp. 30–42, Jan. 2012.

[16] Y. Ephraim and D. Malah, “Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator,” *IEEE Trans. Acoust. Speech Signal Process.*, vol. 32, no. 6, pp. 1109–1121, Dec. 1984.

[17] D. Yu, L. Deng, J. Droppo, J. Wu, Y. Gong, and A. Acero, “Robust speech recognition using a cepstral minimum-mean-square-error-motivated noise suppressor,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 16, no. 5, pp. 1061–1070, Jul. 2008.

[18] L. Deng, J. Droppo, and A. Acero, “Recursive estimation of nonstationary noise using iterative stochastic approximation for robust speech recognition,” *IEEE Speech Audio Process.*, vol. 11, no. 6, pp. 568–580, Nov. 2003.

[19] N. S. Kim, T. G. Kang, S. J. Kang, C. W. Han, and D. H. Hong, “Speech feature mapping based on switching linear dynamic system,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 2, pp. 620–631, Feb. 2012.

[20] T. Nakatani, B. H. Juang, T. Yoshioka, K. Kinoshita, M. Delcroix, and M. Miyoshi, “Speech dereverberation based on maximum-likelihood estimation with time-varying gaussian source model,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 16, no. 8, pp. 1512–1527, Nov. 2008.

[21] M. Miyoshi, M. Delcroix, K. Kinoshita, T. Yoshioka, T. Nakatani, and T. Hikichi, “Inverse filtering for speech dereverberation without the use of room

acoustics information," in *Speech Dereverberation*, P. A. Naylor and N. D. Gaubitch, Eds. London: Springer-Varlag, 2010.

[22] K. Lebart, J. M. Boucher, and P. N. Denbigh, "A new method based on spectral subtraction for speech dereverberation," *Acta Acoustica*, vol. 87, no. 3, pp. 359–366, 2001.

[23] K. Kinoshita, M. Delcroix, and T. Nakatani, "Suppression of late reverberation effect on speech signal using long-term multiple-step linear prediction," *IEEE Trans. Audio, Speech, Language Process.*, vol. 17, no. 4, pp. 534–545, May 2009.

[24] E. Habets, S. Gannot, and I. Cohen, "Late reverberant spectral variance estimation based on a statistical model," *IEEE Signal Process. Lett.*, vol. 16, no. 9, pp. 770–773, Sep. 2009.

[25] D. Willner, C. B. Chang, and K. P. Dunn, "Kalman filter algorithms for a multi-sensor system," in *Proc. IEEE Conf. Decis. Control*, 1976, pp. 570–574.

[26] A. Krueger and R. Haeb-Umbach, "Model-based feature enhancement for reverberant speech recognition," *IEEE Trans. Audio, Speech, Language Process.*, vol. 18, no. 7, pp. 1692–1707, Sep. 2010.

[27] J. Droppo and A. Acero, "Noise robust speech recognition with a switching linear dynamic model," in *Proc. ICASSP*, 2004, pp. 953–956.

[28] N. S. Kim, W. Lim, and R. M. Stern, "Feature compensation based on switching linear dynamic model," *IEEE Signal Process. Lett.*, vol. 12, no. 6, pp. 473–476, Jun. 2005.

[29] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *J. R. Stat. Soc. Series B, Stat. Methodol.*, vol. 39, no. 1, pp. 1–38, 1977.

[30] L. Deng, L. J. Lee, H. Attias, and A. Acero, “Adaptive kalman filtering and smoothing for tracking vocal tract resonances using a continuous-valued hidden dynamic model,” *IEEE Trans. Audio, Speech, Language Process.*, vol. 15, no. 1, pp. 13–23, Jan. 2007.

[31] ETSI Std. Document, “Speech processing, transmission and quality aspects (STQ); distributed speech recognition; front-end feature extraction algorithm; compression algorithm,” Tech. Rep., 2003.

[32] D. Pearce and H. G. Hirsch, “The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions,” in *Proc. Int. Conf. Spoken Language Process.*, 2000, pp. 29–32.

[33] S. Y. et al., “The htk book,” Tech. Rep., 2006.

[34] Speech information technology and industry promotion center website. [Online]. Available: http://www.sitec.or.kr

[35] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, “Greedy layer-wise training of deep networks,” in *Proc. NIPS*, 2006, pp. 153–160.

[36] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010.

[37] X.-L. Zhang and J. Wu, "Denoising deep neural networks based voice activity detection," in *Proc. ICASSP*, 2013, pp. 853–857.

[38] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc. ASRU*, 2011, pp. 24–29.

[39] H. Larochelle and Y. Bengio, "Classification using discriminative restricted boltzmann machines," in *Proc. ICML*, 2008, pp. 536–543.

[40] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[41] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[42] A. Ghoshal and D. Povey, "Sequence-discriminative training of deep neural networks," in *Proc. Interspeech*, 2013, pp. 2345–2349.

[43] M. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Proc. ICASSP*, 2013, pp. 7398–7402.

[44] D. Yu and L. Deng, *Automatic Speech Recognition - A Deep Learning Approach.* London: Springer-Verlag, 2015.

[45] G. Hu. (2004) 100 nonspeech environmental sounds. [Online]. Available: http://web.cse.ohio-state.edu/pnl/corpus/HuNonspeech/HuCorpus.html

[46] J. Du, Q. Wang, T. Gao, Y. Xu, L.-R. Dai, and C.-H. Lee, "Robust speech recognition with speech enhanced deep neural networks," in *Proc. Interspeech*, 2014, pp. 616–620.

[47] D. Yu, M. L. Seltzer, J. Li, J. T. Huang, and F. Seide, "Feature learning in deep neural networks – studies on speech recognition tasks," in *Proc. Int. Conf. Learn. Represent.*, 2013.

[48] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1819–1837, Aug. 2014.

[49] C. Weng, D. Yu, S. Watanabe, and B.-H. F. Juang, "Recurrent deep neural networks for robust speech recognition," in *Proc. ICASSP*, 2014, pp. 5532–5536.

[50] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *J. Mach. Learn. Res.*, vol. 6, pp. 1453–1484, Dec. 2005.

[51] A. W. Senior and A. J. Robinson, "Forward-backward retraining of recurrent neural networks," in *Proc. NIPS*, D. Touretzky, M. Mozer, and M. Hasselmo, Eds., 1996, pp. 743–749.

[52] L. Bahl, P. Brown, P. de Souza, and R. Mercer, "Maximum mutual information estimation of hidden markov model parameters for speech recognition," in *Proc. ICASSP*, 1986, pp. 49–52.

[53] D. Povey and B. Kingsbury, "Evaluation of proposed modifications to mpe for large scale discriminative training," in *Proc. ICASSP*, 2007, pp. IV–321–IV–324.

요 약

본 논문에서는 주변 환경에 강인한 음성인식을 위한 모델 및 데이터기반 기법들을 제안한다. 모델기반 기법은 반향 및 잡음이 존재하는 환경에서 GMM-HMM 인식기를 대상으로 강인한 음성인식을 얻기 위한 특징 보상 기법이다. 이는 기존에 제안되었던 IMM 기반 단일 채널 알고리즘을 다중 채널로 확장하였다. 제안된 다중 채널 알고리즘은 Bayisian 관점에서 기술되고 시간에 따라 변하는 반향 및 잡음의 특성을 실시간으로 추정에 나가는 장점을 가진다. 그리고 다중 채널의 계산량을 줄이기 위한 기법도 같이 제안되었으며, 다중 채널 환경의 시뮬레이션 실험과 자동차 환경에서 녹음된 실제 데이터를 통해 성능이 검증되었다.

데이터기반 기법은 DNN-HMM 인식기를 대상으로 수행되며 세가지 기법이 제안된다. 첫 번째로 주변 환경에 강인한 음성인식 성능을 얻기 위해 잡음 환경에서 DNN의 파라미터들이 좀 더 나은 초기값을 가질 수 있도록 지도 학습 기반의 사전 학습을 제안한다. 제안된 알고리즘에서 잡음 데이터를 입력으로 하는 DNN의 초기값은 깨끗한 음성으로 구별 학습된 보조 DNN의 은닉층 노드의 값을 가지도록 역전파 알고리즘을 수행하여 초기값이 결정된다. 제안된 기법은 Aurora-4 DB에서 여러 사전학습 기법들과 비교되었으며 좀 더 나은 성능을 보였다.

두 번째는 DNN의 학습 및 테스트 환경에서의 잡음 환경의 불일치에 따른 성능 감소를 보완하기 위해 주변 잡음 추정치를 DNN의 입력으로 같이 사용하는 기법이다. 이를 통해서 DNN은 입력 데이터와 주변 잡음 추정치의 정보를 모두 활용하여 학습되고

테스트 시 학습에 쓰이지 않은 잡음 데이터가 입력으로 들어오더라도 주변 잡음 추정 정보를 통해 좀 더 강인한 인식 성능을 얻을 수 있도록 한다. 주변 잡음은 IMM 기반 알고리즘을 통해서 얻어진다. 제안된 기법은 Aurora-4 DB에서 기존에 고정된 잡음 추정치를 사용하는 기법과 비교되어 좀 더 나은 성능을 보였다.

마지막으로 DNN 학습에서의 대상 레이블 정보를 고정된 값을 사용하는 것이 아니라 확률 정보에 기반한 레이블을 사용하는 것이다. 이는 잡음 데이터가 주어질 경우에는 여러 입력에 대해 하나의 고정된 정답을 주는 것보다 가능한 정답을 모두 제공해 주는 것이 좀 더 학습의 강인함을 증가시킬 수 있기 때문이다. 확률 정보에 기반한 레이블은 forward-backward 알고리즘을 통해 얻어졌으며 기존 Viterbi 알고리즘을 통해 얻어지는 고정된 레이블을 사용하는 방법과 Aurora-4 DB에서 비교되었고 좀 더 나은 성능을 확인할 수 있었다.

게다가, 모델기반 기법과 데이터기반 기법에서 제안된 세가지의 기법들을 하나의 통합된 알고리즘으로 제안하였고 이는 학습과 테스트 시의 잡음 환경의 일치 및 불일치 조건에 대해서 평가되었다. 결과적으로 잡음 환경이 일치할 경우에는 초기값을 정하는 기법이 중요한 역할을 하였으며 잡음 환경이 불일치할 경우에는 잡음 추정치를 DNN의 입력으로 사용하는 기법과 확률에 기반한 대상 레이블을 같이 사용했을 때 가장 좋은 성능을 보였다.

주요어: 강인한 음성인식, 다중 채널, interacting multiple model (IMM), 반향 제거, 사전 학습, 잡음 제거, 잡음 추정, deep neural network (DNN), DNN 기반의 regression, 오류 역전파 알고리즘, 확률 기반 대상 레이블, Viterbi 알고리즘, forward-backward 알고리즘

학 번: 2010-30974