Auditory Representation Learning

by

Hardik B. Sailor
201321002

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY
to
Dhirubhai Ambani Institute of Information and Communication Technology

August 2018

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Doctor of Philosophy at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

Hardik B. Sailor

Certificate

This is to certify that the thesis work entitled, "*Auditory Representation Learning*," has been carried out by *Hardik B. Sailor* for the degree of *Doctor of Philosophy* at *Dhirubhai Ambani Institute of Information and Communication Technology* under my supervision.

Prof. Hemant A. Patil
Thesis Supervisor

Acknowledgments

First and foremost, I would like to express my gratitude and thank to my Ph.D. supervisor Prof. Hemant A. Patil. He constantly motivates and encourages me with helpful discussions for good quality research followed by publications in top journals and conferences. He is completely dedicated towards his students in the manuscript and thesis corrections, research meetings and feedback on the research works. I am also very thankful to him for providing me opportunities for presenting my research work in various summer schools at DA-IICT, IIIT-Vadodara, and Nirma University. The book titled “Auditory Neuroscience: Making Sense of Sound" gifted by him is a great asset to me in the field of auditory representation learning.

I acknowledge my Research Progress Seminar (RPS) and Ph.D. synopsis committee members Prof. M. V. Joshi, Prof. Aditya Tatu, Prof. Sanjeev Gupta, Prof. Manish Narwaria, and authorities of DA-IICT. Special thanks to Prof. M. V. Joshi for significant discussion on deep learning and its mysteries. I acknowledge my thesis examiners, namely, Prof. Douglas O'Shaughnessy (INRS, Montreal, Canada) and Prof. Chandra Sekhar Seelamantula (IISc, Bangalore) for their excellent thesis reviews, corrections, and technical suggestions. I also acknowledge Prof. Suman Mitra (Dean Academic Programs at DA-IICT) as a one of the Ph.D. defense committee members. I am also thankful to anonymous reviewers of our research papers for their valuable feedback and suggestions. I also acknowledge the funding and support from the Ministry of Electronics and Information Technology (MeitY) through two consortium projects: TTS Phase-II and ASR Phase-II.

My heartfelt gratitude and thanks to all the members of Speech Research Lab at DA-IICT. The technical discussions with Maulikbhai are really helpful to me for thesis and understanding speech signal processing. Special thanks to Nirmesh who is also my good friend, co-author and earlier roommate for the technical discussions, and joyful conversations. I am very benefited from the presence of Mohammadi Zaki with whom we started an initial work of deep learning that really helped me in my further studies. I acknowledge my co-authors, Madhu, Dharmesh, and Avni for exploiting the filterbank learning framework in their re-

search areas followed by publications. I am very much thankful to all the project staff members of TTS and ASR project at DA-IICT for the submission of project deliverables. I also acknowledge all the members of Speech Research Lab for their help in data collection and accommodation in various regions of the Gujarat state. There are many other friends and colleagues who supported me and made my Ph.D. days enjoyable at DA-IICT.

With deep respect and regard, I am very much thankful to my parents. Words are not sufficient for their enormous love and care. Special appreciation for my lovely and helpful wife Hetal who supported me significantly throughout my Ph.D. studies and always stood with me in the rainbow of life. I also thank my brother Jugal who supported our parents and allow me to do research here at DA-IICT. I am also thankful to my parents-in-laws for their patience, trust and support throughout my Ph.D. studies.

Contents

Abstract

Representation learning (RL) or feature learning has a huge impact in the field of signal processing applications. The goal of the RL approaches is to learn the meaningful representation directly from the data that can be helpful to the pattern classifier. Specifically, the unsupervised RL has gained a significant interest in the feature learning in various signal processing areas including the speech and audio processing. Recently, various RL methods are used to learn the auditory-like representations from the speech signals or its spectral representations.

In this thesis, we propose a novel auditory representation learning model based on the Convolutional Restricted Boltzmann Machine (ConvRBM). The auditory-like subband filters are learned when the model is trained directly on the raw speech and audio signals with arbitrary lengths. The learned auditory frequency scale is also nonlinear similar to the standard auditory frequency scales. However, the ConvRBM frequency scale is adapted to the sound statistics. The primary motivation for the development of our model is to apply in the Automatic Speech Recognition (ASR) task. Experiments on the standard ASR databases show that the ConvRBM filterbank performs better than the Mel filterbank. The stability analysis of the model is presented using Lipschitz continuity condition. The proposed model is improved by using annealing dropout and Adam optimization. Noise-robust representation is achieved by combining ConvRBM filterbank with an energy estimation using the Teager Energy Operator (TEO). As a part of the research work for the MeitY, Govt. of India sponsored consortium project, the ConvRBM is used as a front-end for the ASR system in the speech-based access for the agricultural commodities in the Gujarati language. Inspired by the success in the ASR task, we applied our model in three audio classification tasks, namely, Environmental Sound Classification (ESC), synthetic and replay Spoof Speech Detection (SSD) in the context of the Automatic Speaker Verification (ASV), and Infant Cry Classification (ICC). We further propose the two layer auditory model by stacking two ConvRBMs. We refer it as an Unsupervised Deep Auditory Model (UDAM) and it performed well compared to the single layer ConvRBM in the ASR task.

List of Acronyms

AGC	Automatic Gain Control
AI	Artificial Intelligence
AM	Amplitude Modulation
ANF	Auditory Nerve Fibers
ANN	Artificial Neural Networks
ARL	Auditory Representation Learning
ASR	Automatic Speech Recognition
ASV	Automatic Speaker Verification
BLSTM	Bidirectional Long Short-Term Memory
BM	Basilar Membrane
BPTT	Back-Propagation Through Time
CD	Contrastive Divergence
CD-HMM	Continuous Density Hidden Markov Models
CF	Center Frequencies
CI	Conditional Independence
CNN	Convolutional Neural Networks
CNS	Central Nervous System
ConvRBM	Convolution Restricted Boltzmann Machine
CRF	Conditional Random Fields
CW	Context Window
DCT	Discrete Cosine Transform
DET	Detection Error Tradeoff
DGM	Directed Graphical Models
DNN	Deep Neural Networks
DST	Deep Scattering Transform
EBM	Energy Based Models

EER	Equal Error Rate
ERB	Equivalent Rectangular Bandwidth
ESC	Environmental Sound Classification
FAR	False Acceptance Rate
FFNN	Feed-Forward Neural Networks
FM	Frequency Modulation
FRR	False Rejection Rate
FST	Finite State Transducer
FT	Fourier Transform
GBRBM	Gaussian-Bernoulli Restricted Boltzmann Machine
GF	Gammatone Filterbank
GMM	Gaussian Mixture Models
GPU	Graphical Processing Unit
HAS	Human Auditory system
HMM	Hidden Markov Models
IC	Inferior colliculus
ICA	Independent Component Analysis
ICC	Infant Cry Classification
IHC	Inner Hair Cells
ILSL	Indian Language Speech Sound Label Set
IVRS	Interactive Voice Response System
LDA	Linear Discriminant Analysis
LF-MMI	Lattice-free Maximum Mutual Information
LIF	Leaky Integrate-and-Fire
LLR	Log-Likelihood Ratio
LM	Language Model
LReLU	Leaky Rectifier Linear Units
LSTM	Long Short-Term Memory
LVCSR	Large Vocabulary Continuous Speech Recognition
MBR	Minimum Bayes Risk
MCMC	Markov Chain Monte Carlo
MFCC	Mel Frequency Cepstral Coefficients
MIS	Mandi Information System
MLE	Maximum Likelihood Estimation

MLLT	Maximum Likelihood Linear Transform
MLP	Multilayer Perceptrons
MRF	Markov Random Fields
NReLU	Noisy Rectified Linear Units
OHC	Outer Hair Cells
PCA	Principal Component Analysis
PDF	Probability Distribution Function
PER	Phone Error Rate
PGM	Probabilistic Graphical Model
PLP	Perceptual Linear Prediction
PNS	Power Normalized Spectra
POI	Probability Of Improvement
PReLU	Parametric Rectifier Linear Units
PRI	Primary Rate Interface
RBM	Restricted Boltzmann Machine
ReLU	Rectifier Linear Units
RF	Receptive Fields
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Networks
SGD	Stochastic Gradient Descent
SMVN	Spectral Mean Variance Normalization
SNN	Stochastic Neural Networks
SSD	Spoof Speech Detection
SSU	Stepped Sigmoid Units
STFT	Short-Time Fourier Transform
STRF	Spectro-Temporal Receptive Field
TDNN	Time-Delay Neural Networks
TEO	Teager Energy Operator
TESC	Teager Energy Spectral Coefficients
TFS	Temporal Fine Structure
TRF	Temporal Receptive Field
UDAM	Unsupervised Deep Auditory Model
UGM	Undirected Graphical Model
WER	Word Error Rate

List of Symbols

$p(\cdot)$	Probability distribution
$P(\cdot)$	Probability of an event
Z	Partition function
$E(\cdot)$	Energy function of an energy-based models
$\prod$	Notation for multiplication
$\sum$	Notation for summation
θ	Model parameters
$\mathbf{x}$	Input signal or visible units to the learning algorithm
$\mathbf{h}$	Hidden units
$\mathbf{W}$	Weights of a model (DNN, RBM, ConvRBM)
$\mathrm{sigmoid}(\cdot)$	Sigmoid function
$\mathcal{N}(\mu, \sigma^2)$	Gaussian distribution with mean μ and variance σ^2
$*$	Convolution operation
$\oplus$	System combination
$\odot$	Elementwise multiplication
$\approx$	Approximately
$\log(\cdot + \epsilon)$	Stabilized logarithm with base e and $\epsilon = 0.0001$
Δ	Delta or dynamic features
$\Delta\Delta$	Delta-Delta or double delta or acceleration features
$\langle \cdot \rangle$	Sample average
$\mathbb{E}[\cdot]$	Expectation operator
ϵ	Learning rate parameter
η	Momentum parameter
$\lVert \cdot \rVert_p$	L^p-norm, where $p = 2$ for L^2-norm
$Bernoulli(\cdot)$	Bernoulli distribution
$\Psi\{\cdot\}$	Teager Energy Operator

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Motivation

“Artificial Intelligence (AI) is the new electricity" according to Andrew Ng, a professor at Stanford University. Today AI is transforming every sector of industry, academics, and business. From smartphones to supercomputers, the main medium of operating and communicating with humans is through AI. Specifically, voice-based access in digital devices, such as smartphones, is equipped with an AI-enabled personal assistant, e.g., OK Google, Apple Siri, Microsoft Cortana, Amazon Echo, etc. Along with AI, there are rapid advancements in the Internet of Things (IoT) where many electronic devices are connected through the Internet. Since an AI-enabled IoT framework is used in real-life noise scenarios, we need a knowledge of both signal processing and machine learning to propose a feasible solution to develop a noise robust voice-based access system in the IoT. Hence, there is a great demand for speech and audio technology in the near future since voice is the most important form of human communication and possibly human-machine interactions.

One of the important aspects of speech and audio processing applications is to come up with a better representation of the sounds so that variabilities in the signals are largely reduced. For example, in the speech recognition task, we need a feature representation that reduces the variability in speakers, dialects, microphones, and provide robustness to the background noise [37]. It is shown in research studies that the representative features obtained using human auditory-based models are more successful in speech recognition, sound classification and synthesis [38]. The traditional approaches (as shown in Figure 1.1) to extract the auditory features are either based on computational models that are based on perceptual or physiological experiments or mathematical models that are based on the mathematical representation of underlying transformations in the auditory system [39].

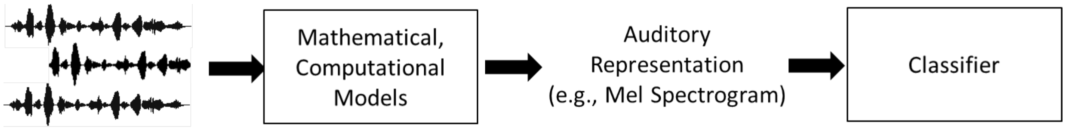

Mathematical, Computational Models
Auditory Representation (e.g., Mel Spectrogram)
Classifier

Figure 1.1: Traditional approach for speech signal processing.

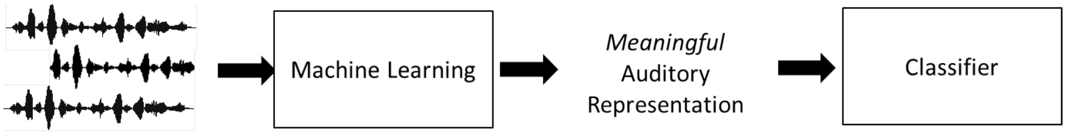


Figure 1.2: Representation learning for speech signal processing.

Recently, representation learning (RL) has gained a significant interest for feature learning in various signal processing areas including speech processing [40]. In the speech processing literature, it is also called Automatic Speech Analysis (ASA) that involves the extraction of meaningful information using computers from the speech signals [41]. The goal of the RL is to extract meaningful auditory representations directly from the raw signals as shown in Figure 1.2. As discussed in [42], features for human speech perception, vision, and other cognition tasks are learned from experience, simulating the human learning mechanism (to a certain extent) as we grow. Unsupervised learning is one of the important forms of representation learning since many human learning tasks are unsupervised [43]. For example, we listen to many sounds as we grow and we are usually not told every time about the type of sound, speech and their sources (e.g., speaker-specific aspects, such as gender, age, etc.). Another example is language acquisition by infants during initial stages of their growth, which is also a type of unsupervised learning [44].

Motivation for this thesis is the first notable study conducted by Lewicki to show that the human auditory system (HAS) is adapted to sound statistics in [12]. The data-driven model based on the information-theoretic criteria is trained with three sound categories, namely, environmental sounds, animal vocalizations, and speech signals. The subband filters obtained from these three categories are shown in Figure 1.3. The experimental results show that the optimal auditory codes are different according to the statistics of the sounds, such as the wavelet-like basis for the environmental sounds, a Fourier-like basis for animal vocalizations and a mixture of the wavelet and Fourier-like basis for the speech signals. It is also shown that most of the learned subband filters are similar to the physiological auditory filters estimated through the experiments on animals as shown in Figure 1.4.

The objective of this thesis is to propose a novel auditory representation learn-

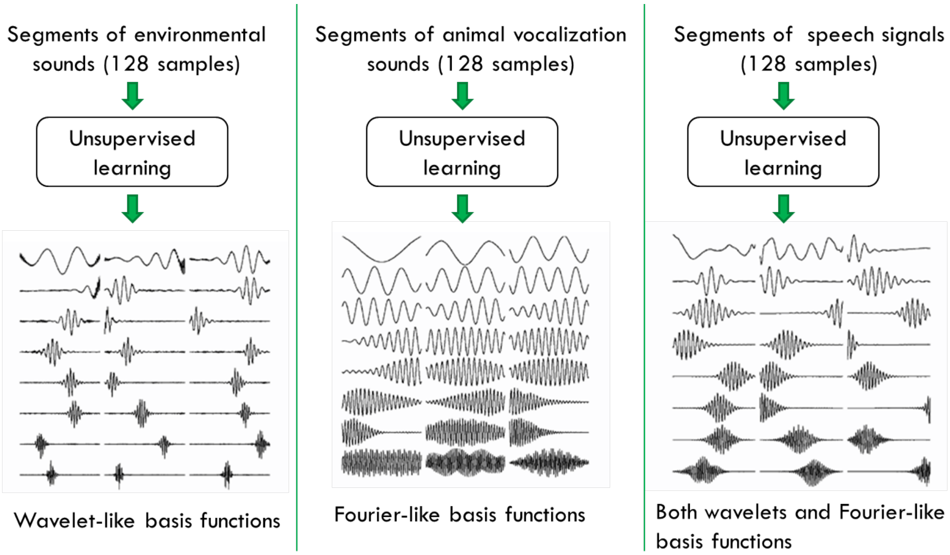

Segments of environmental sounds (128 samples)
Unsupervised learning
Wavelet-like basis functions
Segments of animal vocalization sounds (128 samples)
Unsupervised learning
Fourier-like basis functions
Segments of speech signals (128 samples)
Unsupervised learning
Both wavelets and Fourier-like basis functions

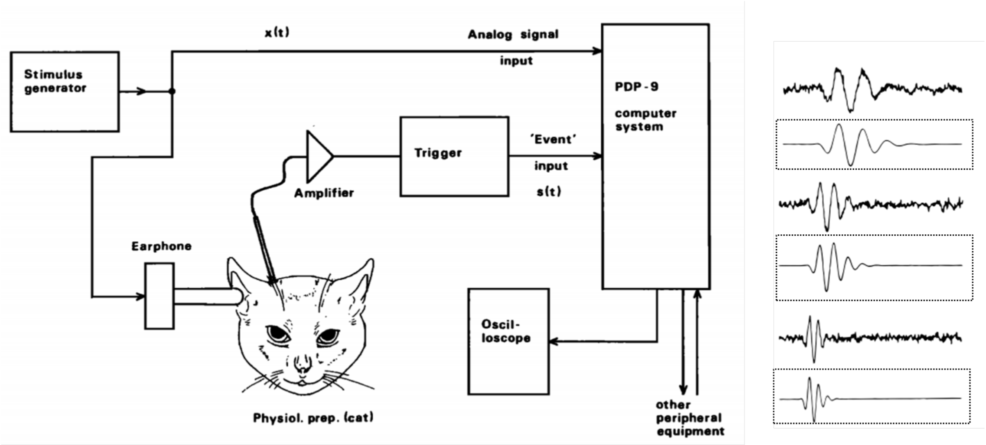
Figure 1.3: Summary of the remarkable study conducted by Lewicki using efficient auditory coding. After [12].

ing model that can be used in various speech and audio processing applications. In this thesis, we have developed unsupervised auditory filterbank learning using a Convolutional Restricted Boltzmann Machine (ConvRBM) directly from the raw speech and audio signals of arbitrary lengths. Our proposed ConvRBM model has been successfully applied in automatic speech recognition (ASR) [2–4, 10, 11], Environmental Sound Classification (ESC) [5], spoof speech detection (SSD) [6], [45], and infant cry classification (ICC) [46]. Next, we discuss the key research challenges in the development of auditory-based models followed by the contributions in this research area from this thesis.

1.2 Key Research Challenges

The HAS is one of the engineering masterpieces of the human body that is unique and distinct from other animals. The auditory processing on the incoming sound is based on various physiological aspects of the human ear. The key research challenges in developing the human auditory model are as follows:

- The HAS is highly complex containing several layers of nonlinear transformations and physiological effects, many of which are still not clearly understood. Many auditory models still use linear approximations and a simplified mathematical formulation to mimic the HAS. The main research challenge is how to best approximate the functioning of the HAS. To answer this question, many researchers from different fields, such as auditory neuroscience, speech processing, applied mathematics, psychoacoustics, machine



(a) (b)

Figure 1.4: (a) Experimental setup for the physiological auditory filter estimation, and (b) comparison with the data-driven auditory filters (shown in the boxes). After [13], [12].

learning, etc. are working and proposed various models/methods to represent the HAS. Recently, the research in this direction is growing significantly due to availability of data, techniques and demand for the development of an area called the *machine perception* or *machine hearing* [47].

- There is a significant importance of the temporal structures in sounds. Most of the auditory models in speech and audio applications have used windowing for the quasi-stationary assumptions. The windowing of the speech and audio in the auditory feature representation introduces artifacts, and also we are ignoring the nonstationary nature of the sounds [48]. One of the ways to get high temporal resolution is to use subband processing using the auditory filterbanks. However, due to high dimensionality of the subband responses windowing is performed at a later stage. Hence, how to preserve the temporal structures in the sounds is another open research issue.

- The standard auditory representations use a fixed auditory frequency scale and filter shapes for a variety of applications. However, the auditory system is continuously adapting to the natural sound statistics [12]. Hence, machine learning methods have emerged to learn the subband filters and STRFs in the auditory processing. However, still there are linear time-invariant (LTI) system assumptions (e.g., convolution model of the filterbank) and hence, many complex adaptations (e.g., automatic gain control (AGC) and synaptic plasticity) are ignored.

- It is observed that the type of auditory representations used are selective for the particular applications. For example, the speech processing applications use the Mel filterbank while audio classification generally uses gammatone and wavelet filterbanks. In many audio processing applications, the STFT is also used. Hence, there is a need of a generalized auditory representation that is common across the tasks.

- Understanding the cortical representation of sounds is currently an active research area in auditory neuroscience. Due to highly complex nature of the auditory cortex, many computational and machine learning models are still at an elementary-level. Moreover, the existing models produce the cortical representation that is very high-dimensional (e.g., 4-D responses in the STRF-based models [49]). Hence, understanding cortical processing of sounds and how to represent it in a concise form is a further research issue.

1.3 Contributions from the Thesis

The main contribution of the thesis is to propose a novel model of the auditory representation learning that tries to address few of the research challenges mentioned above. The model is based on ConvRBM, an unsupervised probabilistic graphical model (PGM). Following are the key contributions in this thesis using our proposed ConvRBM model:

1.3.1 Proposed Model for Auditory Representation Learning

Compared to the earlier work of using ConvRBM applied to model the spectrograms with sigmoid units [50], we proposed to model the raw speech signals of an arbitrary lengths and thus, avoiding the need of windowing. We also propose to use noisy rectified linear units (NReLU) for inference in ConvRBM. The mathematical derivations for the ConvRBM architecture, and an algorithm to train the model are developed. We further improved our proposed ConvRBM using an annealing dropout and the Adam optimization technique in ConvRBM training. For noise-robust ASR task, a novel auditory-based feature representation is proposed using ConvRBM and the energy estimation using the Teager Energy Operator (TEO). A unsupervised deep auditory model (UDAM) is proposed by stacking the two ConvRBMs using a greedy layer-wise training. The first ConvRBM learns auditory filterbank from the raw speech and audio signals. The second ConvRBM learns the temporal modulation information. Hence, the proposed UDAM can be seen as a simplified model of the deep auditory processing in humans.

1.3.2 Analysis of the Model and Representation

The subband filters, frequency scale, and the hidden unit representations of the ConvRBM are analyzed in this thesis. The analysis of the learned frequency scales of ConvRBM are compared with the standard auditory frequency scales. The shapes of the subband filters are analyzed and compared with the physiological auditory filters estimated from the human auditory nerve fibers (ANF) and standard auditory filters, such as a Gammatone filterbank. The comparative analysis of the subband filters and the frequency scales obtained using various sound categories are also provided. The cross-domain experiments are performed on the ASR task to justify that ConvRBM can learn general representation across various databases of the speech signals. The mathematical justification of improved performance in the noise-robust ASR task is given using the Lipschitz continuity conditions derived for the ConvRBM. The modulation information extracted by the temporal receptive fields (TRF) in the UDAM is also analyzed.

1.3.3 Applications

The first motivation to develop the ConvRBM is to use it as a front-end in the automatic speech recognition (ASR). The experimental results on the standard ASR datasets shows that our proposed ConvRBM-based features perform better than the Mel filterbank. As a part of the Ministry of Electronics and Information Technology (MeitY), Govt. of India sponsored consortium project at DA-IICT, ConvRBM is also applied in the development of a speech-based access system for the agricultural commodities in the Gujarati language. Later, the ConvRBM is applied in a variety of speech and audio processing applications, namely, Environmental Sound Classification (ESC), Spoof Speech Detection (SSD), and Infant Cry Classification (ICC). In all these applications, our proposed model gave consistently better performance compared to the respective baselines. The auditory frequency scales and subband filters are adapted automatically throughout the learning of ConvRBM on a particular database.

The overall contributions of this thesis are summarized in Figure 1.5.

1.4 Organization of the Thesis

The organization of the chapters in the thesis is shown in Figure 1.6.

The required background studies and literature is discussed in Chapter 2. We discussed the fundamentals of human auditory processing and probabilistic

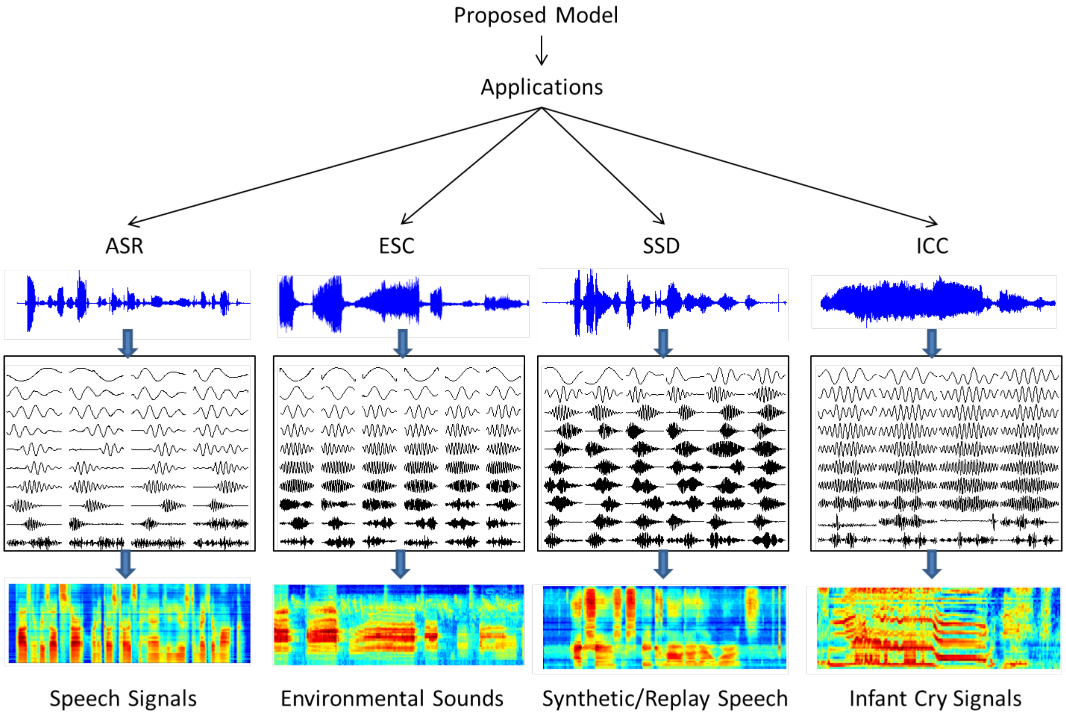


Figure 1.5: Pictorial representation of proposed model applied in different applications along with the subband filters learned for particular sound categories.

graphical models. Since the major application is the ASR task, we briefly described the background of the ASR task. The literature survey of the auditory representation learning is also discussed in this chapter.

The detailed description of the architecture of our proposed ConvRBM model, the learning algorithm, and the feature extraction technique are presented in Chapter 3. The analysis of filterbank and stability of a convolution operation and the rectified non-linearity are also discussed. The experiments on the ASR task using the standard datasets are presented to evaluate the proposed model.

Chapter 4 discusses the approaches to improve the proposed model using advanced optimization and regularization techniques. Specifically, the noise-robust auditory representations are obtained using the Teager Energy Operator and ConvRBM feature representation. The improved model is evaluated on the noisy ASR task.

As a special case study in the ASR, the speech-based access system for the agricultural commodities in the Gujarati language is presented in Chapter 5. The data collection and transcription for the ASR in the Gujarati language presented. The experimental results with the proposed feature representation is discussed.

Three audio classification tasks are presented in Chapter 6. The ConvRBM is applied to the diverse complex sounds other than speech in the environmen-

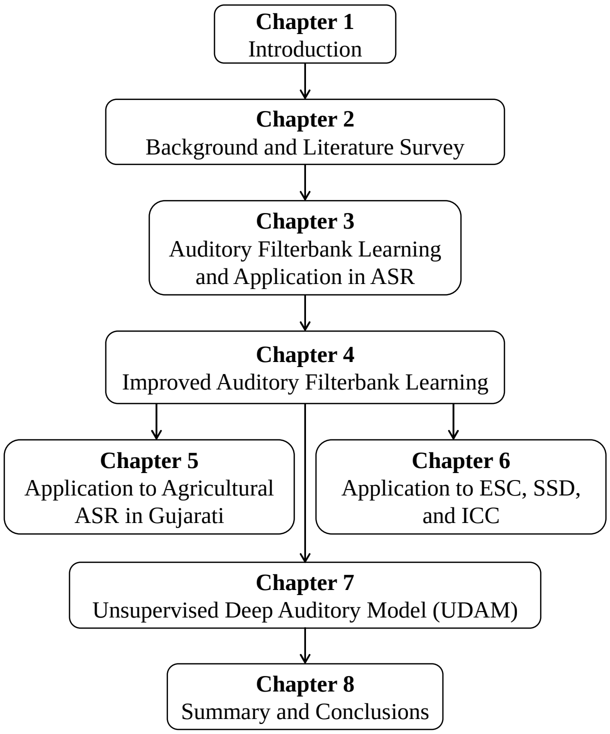

Chapter 1
Introduction
Chapter 2
Background and Literature Survey
Chapter 3
Auditory Filterbank Learning and Application in ASR
Chapter 4
Improved Auditory Filterbank Learning
Chapter 5
Application to Agricultural ASR in Gujarati
Chapter 6
Application to ESC, SSD, and ICC
Chapter 7
Unsupervised Deep Auditory Model (UDAM)
Chapter 8
Summary and Conclusions

Figure 1.6: Organization of the chapters in the thesis.

tal sound classification. To improve the security of the voice biometrics, a spoof speech detection task is also discussed. Finally, a socially relevant problem of infant cry classification is discussed where we show that ConvRBM can be trained even on the small amount of database.

Chapter 7 discusses the proposed deep model UDAM and its application to ASR task. The overall summary of the thesis and future research directions are presented in Chapter 8.

1.5 Chapter Summary

In this Chapter, an introduction to the problem of auditory representation learning in this thesis is presented. The motivation for our research work in this thesis is discussed that is based on significance of data-driven techniques for auditory representations. The major contributions in the thesis include a novel model of auditory representation learning, analysis of the filterbank, and applications to ASR, ESC, SSD, and ICC. The organization of the chapters in the thesis is also presented. In the next Chapter, we discuss the background studies and the literature corresponding to the auditory representation learning.

CHAPTER 2

Background and Literature Survey

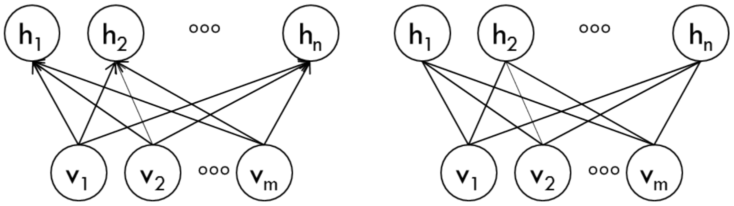
2.1 Introduction

To understand our proposed approach of the auditory representation learning, the required background topics are presented in this Chapter. Since our proposed model is probabilistic in nature, the fundamentals of the probabilistic graphical models (PGM) are discussed in Section 2.2. In Section 2.3, we will discuss how the Boltzmann machines are formulated from the Hopfield networks followed by its restricted version called as restricted Boltzmann machine (RBM) in Section 2.4. Introducing a convolution as a strong prior in RBM, the Convolutional RBM is presented in Section 2.5. The Section 2.6 describes various stages of the human auditory processing. The fundamentals of deep learning is presented in Section 2.7. Since our early motivation for the auditory representation learning was the ASR, the fundamentals of the ASR is presented in Section 2.8. Finally, the literature on the auditory modeling is presented in Section 2.9.

2.2 Probabilistic Graphical Models (PGM)

PGM combine the graph theory and probability theory in a efficacious approach for the statistical modeling. PGM has several advantages, such as [14]:

- Using the graph theory, PGMs provide a simple way to visualize the structure of a probabilistic model and its variables. Visualization property also helps to make any changes in the design and create new models.
- Probabilistic properties of the model, such as conditional independence, and dependence can directly be observed from the structure of the graph.
- Graphical manipulations are helpful to understand the complex computations required to perform inference and learning in the model.



(a) (b)

Figure 2.1: Examples of graphical models: (a) directed, and (b) undirected.

A graph consists of the nodes (also called as vertices) connected by the edges (called as links or arcs) [14]. In PGM, each node represents a random variable (or sometimes a group of random variables), and edges represent probabilistic relationships between these variables. The graph describes the approach in which the joint distribution over all of the random variables can be decomposed into a product of factors each depending only on a subset of the variables. There are two types of PGMs depending on the directionality of the connections. The first one is directed graphical models (DGM) where the edges in a graph have a particular direction indicated by the arrows between the nodes. They are also called Bayesian networks [14]. The second category is undirected graphical models (UGM), where there are *no* arrows between nodes. They are also called as Markov Random Fields (MRF) [14]. Examples of graphical models are shown in Figure 2.1. The main advantages of UGM over DGM are [51]: (1) they are symmetric and therefore more "natural" for certain domains, such as spatial or relational data, and (2) discriminative UGM (known as conditional random fields, or CRFs), which define conditional densities of random variables work better than the discriminative DGM. The main disadvantages of the UGM compared to the DGM are [51]: (1) the parameters are less interpretable and less modular, (2) estimation of parameters is computationally more expensive and hence, it is generally based on statistical sampling techniques. The Markov Random Fields or Markov networks or UGM specify both conditional independence and factorization properties.

2.2.1 Conditional Independence (CI)

The CI property of random variables is defined as [14]:
Definition: Let A, B, and C be discrete random variables. We say that A and C are conditionally independent given B, written as $A \perp\!\!\!\perp C|B$, if

$$P(A = a, C = c|B = b) = P(A = a|B = b)P(C = c|B = b), \forall a, b, c. \tag{2.1}$$

The simple example of illustrating the CI is shown in Figure 2.2, where $A \perp\!\!\!\perp C|B$.

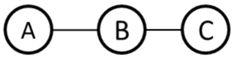


Figure 2.2: An example of a simple graph to illustrate the CI property.

The intuitive meaning of the CI is that once you know B, C provides no extra information about A. One can also directly observe or encode CI between a pair of random variables given the rest of the variables in the graph by not connecting with the edge between variables, and hence,

$$\text{No edge between } A \text{ and } C \Leftrightarrow A \perp\!\!\!\perp C|\text{rest}, \tag{2.2}$$

where the rest refers to all other variables in the graph besides A and C. This type of graph is called a pairwise Markov graph or having the Global Markov property.

2.2.2 Factorization Property

If the two nodes x_i and x_j are conditionally independent then their joint probability distribution factorizes as follows [14]:

$$p(x_i, x_j|\mathbf{x}_{\backslash\{i,j\}}) = p(x_i|\mathbf{x}_{\backslash\{i,j\}})p(x_j|\mathbf{x}_{\backslash\{i,j\}}), \tag{2.3}$$

where $\mathbf{x}_{\backslash\{i,j\}}$ denotes the set $\mathbf{x}$ of all the variables in the graph with x_i and x_j removed. The factorization property of an MRF involves expressing the joint distribution as a product of functions defined over sets of variables that are local to the graph [14]. A clique is defined as a subset of the nodes in a graph such that there exists a link between all pairs of nodes in the subset [14]. The set of nodes in a *clique* is fully connected. A clique is called *maximal clique* if no node can be added such that the resulting set is still a clique [14], [52]. The example for explaining this concept is shown in Figure 2.3. This graph has five cliques one of which is shown by a dot-dash line, and maximal clique is shown by a dashed line in Figure 2.3.

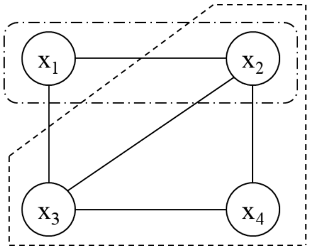


Figure 2.3: An example of a four-node undirected graph showing a clique (dot-dash lines) and a maximum clique (dashed lines). After [14].

The factors in decomposition of the joint distribution can be defined as the functions of variables in the cliques. Without the loss of generality, we can also consider these factors as the maximal cliques since other cliques are subsets of it. Let us denote a clique by $\mathcal{C}$ and the set of variables in that clique by $\mathbf{x}_{\mathcal{C}}$. The joint distribution is now written as a product of non-negative potential functions (such that $p(\mathbf{x}) \geq 0$) $\psi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}})$ over the maximal cliques of the graph [14]:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{\mathcal{C}} \psi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}), \tag{2.4}$$

where Z is a normalization constant also called the partition function [14]. It ensures the overall distribution sums to 1, and is given by:

$$Z = \sum_{\mathbf{x}} \prod_{\mathcal{C}} \psi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}). \tag{2.5}$$

This framework is also applicable to a graph with continuous random variables, where instead of summation we use integration. The normalization constant is one of the major limitations of the UGM. If a model has M discrete nodes each having K states, then the normalization constant is obtained by summing K^M states, and hence is exponential in the size of the model [14]. The CI of random variables and the factorization properties of the joint probability distribution are closely related. The connection between them is given by the *Hammersley-Clifford Theorem* [14], [52]. The theorem states that a strictly positive distribution p satisfies the Markov property w.r.t. an undirected graph if and only if p factorizes over a graph [52]. Since we are restricting the potential functions to be strictly positive, it is convenient to express them as exponential functions so that [14]

$$\psi_{\mathcal{C}}(\mathbf{x}_{\mathcal{C}}) = e^{-E(\mathbf{x}_{\mathcal{C}})}, \tag{2.6}$$

where $E(\mathbf{x}_{\mathcal{C}})$ is called an energy function. The high probability states correspond to low energy configuration and vice-versa. The exponential representation of the energy function is called the Boltzmann distribution or the Gibbs distribution [14], [52]. In the next section, we will discuss how the model that utilizes these concepts will emerge known as the Boltzmann Machine (BM). There is a deep connection between UGMs and mathematical models of statistical physics. Such models are also called energy-based models (EBM), and are generally used in statistical physics, biochemistry, as well as some branches of machine learning to learn representation from the data (in the form of probability distribution) [53].

2.3 From Hopfield Net to Boltzmann Machine (BM)

The first of EBM was the Hopfield network, which is a feedback (recurrent) neural network [25], [54]. The Hopfield network consists of N neurons, where each unit is connected to all other units except itself [54]. The connections between the units are symmetric and bidirectional with weights $w_{ij} = w_{ji}$ for connection between unit i and j. There are no self-connections and hence, $w_{ii} = 0, \forall i$. The biases w_{i0} may be included (this can be viewed as the weights from neuron '0' whose activity x_0 is permanently set to 1). The Hopfield network's activity rule is governed by a threshold activation function [15], [54]. Due to the feedback connections in the Hopfield network, the updates may be synchronous or asynchronous [15].

Definition: Let **W** denote the weight matrix of a Hopfield network with N units. Let **b** be the threshold of an N-dimensional row vector of units. The energy function $E(\mathbf{x})$ of a Hopfield network is given as follows [15]:

$$E(\mathbf{x}) = -\frac{1}{2}\mathbf{x}\mathbf{W}\mathbf{x}^{\mathrm{T}} + \mathbf{b}\mathbf{x}^{\mathrm{T}}. \tag{2.7}$$

The energy can also be written in variables form as follows [15]:

$$E(\mathbf{x}) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}x_i x_j + \sum_{i=1}^{N} b_i x_i. \tag{2.8}$$

The factor 1/2 is present for the mathematical convenience since the identical terms $w_{ij}x_ix_j$ and $w_{ji}x_jx_i$ are present in the double sum. It is always guaranteed to find local minima in the energy surface with the Hopfield network [15].

Proposition: The energy of a Hopfield network can only decrease or stay the same. If an update causes a unit to change sign, then the energy will decrease, otherwise it will stay the same.

A simple example of a Hopfield network is shown in Figure 2.4 as a flip-flop, a network with two units and a zero threshold. It has only two stable states (1,-1) and (-1,1). In any other state, one of the units forces the other unit to change its state to stabilize the network. The energy function of the flip-flop with weights $w_{12} = w_{21} = -1$ and two units with $b = 0$ is given by:

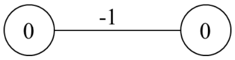


Figure 2.4: A flip-flop as a Hopfield network. After [15].

$$E(x_1, x_2) = x_1 x_2, \tag{2.9}$$

where x_1 and x_2 denote the states of first and second units, respectively. From the four discrete states (1,1), (1,-1),(-1,1), and (-1,-1), $E(x_1, x_2)$ has local minima at (1,-1) and (-1,1). If we choose, $x_1, x_2 = [-1, 1] \in \mathbb{R}$, the energy function can be visualized in Figure 2.5. One can see the two local minima at (1,-1) and (-1,1). The weights of the Hopfield network are learned using the Hebb rule [15], [54]:

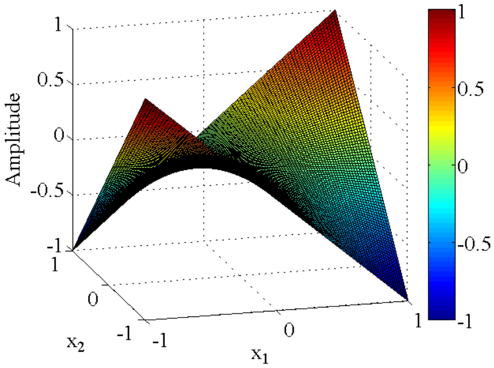


Figure 2.5: The energy function of a flip-flop. After [15].

$$\frac{\partial E(\mathbf{x})}{\partial w_{i,j}} = x_i x_j. \tag{2.10}$$

The major disadvantage of the Hopfield network is that with the increasing complexity of the problem, the number of local minima increases. Hence, the network is not able to stabilize in a correct state by always falling into local minima [54], [25]. One of the possible strategies to avoid many local minimas of the energy function consists in introducing the noise into the network training [54]. The best known model incorporating these concepts is the Boltzmann Machine [55].

Definition: A Boltzmann machine is a Hopfield network composed of N stochastic units with states, $x_1, x_2..., x_N$. The state of a unit i is updated asynchronously according to the rule [54]:

$$x_i = \begin{cases} 1 \text{ with probability } p_i, \\ 0 \text{ with probability } 1 - p_i, \end{cases} \tag{2.11}$$

where the probability p_i is given as [54]:

$$p_i = \frac{1}{1 + e^{-(\sum_{j=1}^{N} w_{ij} x_j - b_i)/T}}, \tag{2.12}$$

and T is a positive temperature constant. Here, w_{ij} denotes the weights of the network and b_i are biases. The temperature T is not actually a physical entity, and hence also called as a pseudo temperature due to an analogy of the Boltzmann machine with the statistical mechanics. The energy function of the Boltzmann machine is similar to the Hopfield network given in eq. (2.8). The difference between a Boltzmann machine and a Hopfield network is the stochastic activation of the units. Hence, the Boltzmann machine and related models are also called the stochastic neural networks (SNN). A simple example of a Boltzmann machine is given by a flip-flop network with $w_{ij} = -1$, and $b = -0.5$ as shown in Figure 2.6. The network states are binary with the energy values: $E_{00} = 0, E_{01} = -0.5, E_{10} = -0.5, E_{11} = 0$ [15]. In order to analyze the dynamics of a Boltzmann machine, we need to construct its matrix of transition probabilities.

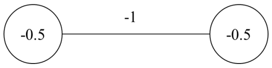


Figure 2.6: A flip-flop network as a Boltzmann machine. After [15].

Definition: The probability of a transition from the state i to state j in a single step at the temperature T in a Boltzmann machine with N units is given by a state transition matrix, $P_T = \{p_{ij}\}$ of size $2^N \times 2^N$.
A matrix with $0 < p_{ij} \leq 1$ and $\sum_{j=1}^{N} p_{ij} = 1$ (rows sums to 1) is called a stochastic matrix. Therefore, Boltzmann machines are examples of a *first-order Markov process*, as the transition probabilities of the future states depend only on the current state, not on the history of the system [15], [54]. The stochastic states of a Boltzmann machine thus construct a *Markov chain*. A fundamental result of the Markov chains theory states that a stable probability distribution function always exists, if all of the states can be reached from any other state in one or more steps, and with non-zero probability [15]. In a Boltzmann machine with $T > 0$, such a stable probability distribution function exists representing the thermal equilibrium of the network dynamics. We can now define transition probabilities in terms of the energies associated with each state.

Consider a system in thermal equilibrium with m different states and associated energies, $E_1, E_2, ..., E_m$. The probability p_{ij} of a transition from a state of energy E_i to another state of energy E_j is given by [15]:

$$p_{ij} = \frac{1}{1 + e^{(E_j - E_i)/T}}. \tag{2.13}$$

The probability distribution function of a physical system, governed by the energy transition, and reaching a stable state at thermal equilibrium, is known as the Boltzmann distribution [54].

Definition: The probability p_i of a system with energy E_i in state i during thermal equilibrium is given by a Boltzmann distribution:

$$p_i = \frac{e^{-E_i/T}}{Z}, \tag{2.14}$$

where $Z = \sum_{i=1}^{m} e^{-E_i/T}$ is a normalizing constant (also called as the partition function) so that distribution p_i sums to one. The Boltzmann machine is governed by eq. (2.13). If a transition from state α to β occurs, a unit k must have changed its state from x_k to $x_k^{'}$. The transition probability $p_{\alpha\beta}$ is given by [15]:

$$p_{\alpha\beta} = \frac{1}{1 + e^{-(\sum_{i=1}^{N} w_{ki}x_i - b_k)/T}}, \tag{2.15}$$

$$= \frac{1}{1 + e^{-(E_\beta - E_\alpha)/T}}, \tag{2.16}$$

where the difference between two energy functions is given by [15]:

$$E_\beta - E_\alpha = -\sum_{i=1}^{N} w_{ki}x_i + b_k. \tag{2.17}$$

Hence, eq. (2.13) and eq. (2.16) are equivalent. To model more complex distributions, hidden units (also known as the latent variables) are also added to the Boltzmann machine [55]. The example of the Boltzmann machine with hidden units is shown in Figure 2.7. The variables where input is connected are called the visible units (denoted as **x**) and the variables where pattern representation is learned are called the hidden units (denoted as **h**). The learning rule of the Boltzmann machine can be derived from the maximum likelihood principle [55]. The major disadvantage of the Boltzmann machine is that due to large number of connections, the inference is intractable. However, restricting the connection to only between visible and hidden units leads to an interesting probabilistic model called a Restricted Boltzmann Machine (RBM).

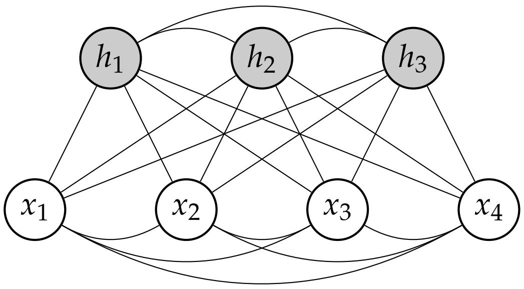


Figure 2.7: An example of Boltzmann machine with the hidden units.

2.4 Restricted Boltzmann Machine (RBM)

An RBM with the binary visible and hidden units is the first original variant of the Boltzmann machine family [56]. It was extended to model real variables in the visible units known as the Gaussian-Bernoulli RBM (GBRBM) [57]. Here, we describe an RBM with the real-valued visible units, $\mathbf{x} \in \mathbb{R}$ and the binary hidden units, $\mathbf{h} \in \{0, 1\}$. The energy function of the GBRBM is given as [57]:

$$E(\mathbf{x}, \mathbf{h}) = \frac{1}{2\sigma_x^2} \sum_{i=1}^{N} x_i^2 - \sum_{i=1}^{N} \sum_{j=1}^{M} x_i w_{ij} h_j - \sum_{i=1}^{N} c_i x_i - \sum_{j=1}^{M} b_j h_j, \tag{2.18}$$

where b_i and c_j are hidden and visible biases, respectively and σ_x is a variance of the visible units. An example of RBM with three hidden units and four visible units is shown in Figure 2.8. It is an MRF with a bipartite graph with the visible and hidden units forming two layers of the vertices in the graph (and no connection between the units of the same layer) [40].

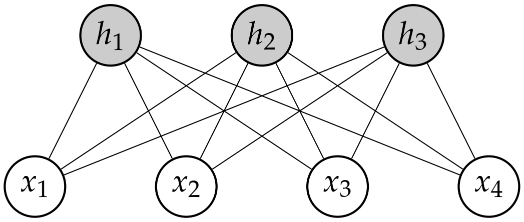


Figure 2.8: An example of RBM with hidden and visible units.

The joint probability distribution of the GBRBM is given as:

$$p(\mathbf{x}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{x}, \mathbf{h})}, \tag{2.19}$$

where Z is the partition function, $Z = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-E(\mathbf{x}, \mathbf{h})} \mathrm{d}\mathbf{x} \mathrm{d}\mathbf{h}$. Here, compared to a Boltzmann machine, the temperature parameter is set to 1. Since there are no connections between hidden-hidden and visible-visible units, hidden units are conditionally independent given the state of the visible units. Hence, the conditional distribution $p(\mathbf{h}|\mathbf{x})$ and $p(\mathbf{x}|\mathbf{h})$ factorize nicely [52]. The probability of the hidden units given the visible units is calculated as follows:

$$p(\mathbf{h}|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{h})}{p(\mathbf{x})}, \tag{2.20}$$

$$= \prod_{j=1}^{M} p(h_j|\mathbf{x}). \tag{2.21}$$

The probability that a hidden unit is ON (i.e., binary state 1) is given by

$$p(h_j = 1|\mathbf{x}) = \text{sigmoid}(\sum_{i=1}^{M} x_i w_{ij} + b_j), \tag{2.22}$$

where the sigmoid function is given as $\text{sigmoid}(x) = \frac{1}{1+e^{-x}}$. The probability of the visible units given the hidden units is calculated as follows:

$$\begin{aligned} p(\mathbf{x}|\mathbf{h}) &= \frac{p(\mathbf{x},\mathbf{h})}{p(\mathbf{h})}, && (2.23)\\ &= \prod_{i=1}^{N} p(x_i|\mathbf{h}), && (2.24)\\ &= \prod_{i=1}^{N} \mathcal{N}(\sum_{j=1}^{M} w_{ij}h_j + c_i, 1), && (2.25) \end{aligned}$$

where $\mathcal{N}(\sum_{j=1}^{M} w_{ij}h_j + c_i, 1)$ is a Gaussian distribution with mean $\sum_{j=1}^{M} w_{ij}h_j + c_i$ and a unit variance. The parameters of the RBM are learned using a contrastive divergence (CD) technique [24]. More detailed discussion on the RBM is presented in [52]. The CD learning technique is presented in detail in Chapter 3.

2.5 From RBM to Convolutional RBM

The RBMs are very successful to learn the feature representations in various signal processing applications including the speech processing [58]. However, they do not scale well as the dimensionality of the input increases. Another important drawback is they do not learn invariant representation, since it requires applying the transformations locally. The translation invariance (and to other groups to certain extent) is achieved by Convolutional Neural Networks (CNN), a state-of-the-art architecture in computer vision [43]. Lee et al. proposed a Convolutional Deep Belief Network by a stack of Convolutional RBMs incorporating ideas from the CNN and RBM [59]. First, we will discuss the difference between fully connected (dense) and having a convolution connection in a network.

2.5.1 Understanding Convolutional Connections

Let us take an example of a dense network of the input, $\mathbf{x} = [x_1, x_2, x_3]$ and output, $\mathbf{y} = [y_1, y_2, y_3, y_4]$ as shown in Figure 2.9 (for simplicity, we ignore the bias terms). If W_{ij} denotes the weights (for $i = \{1, 2, 3, 4\}$, and $j = \{1, 2, 3\}$), the output y_1, y_2, y_3, y_4 is given by:

$$y_1 = \text{sigmoid}(W_{11}x_1 + W_{12}x_2 + W_{13}x_3), \tag{2.26}$$

$$y_2 = \text{sigmoid}(W_{21}x_1 + W_{22}x_2 + W_{23}x_3), \tag{2.27}$$

$$y_3 = \text{sigmoid}(W_{31}x_1 + W_{32}x_2 + W_{33}x_3), \tag{2.28}$$

$$y_4 = \text{sigmoid}(W_{41}x_1 + W_{42}x_2 + W_{43}x_3). \tag{2.29}$$

It can also be written in a matrix form as follows:

$$\mathbf{y} = \text{sigmoid}(\mathbf{x}^T\mathbf{W}), \tag{2.30}$$

where the matrix $\mathbf{W}$ is written as:

$$\mathbf{W} = \begin{bmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \\ W_{41} & W_{42} & W_{43} \end{bmatrix}. \tag{2.31}$$

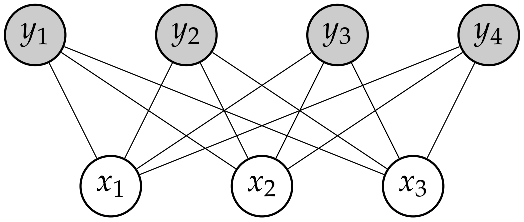


Figure 2.9: An example of a dense neural network.

Now consider a network shown in Figure 2.10. It can be seen that each output node is only connected to two input nodes and weights W_{11} and W_{12} are *shared* between the output nodes. One can also think of it as connecting each node *locally* to the input. The output equations (before nonlinearity) are written as:

$$y_1 = W_{11}x_1 + W_{12}x_2 + 0 \cdot x_3, \tag{2.32}$$

$$y_2 = 0 \cdot x_1 + W_{11}x_2 + W_{12}x_3. \tag{2.33}$$

Writing it in matrix form as $\mathbf{y} = \text{sigmoid}(\mathbf{x}^T\mathbf{W})$ (similar to a dense network),

$$\mathbf{W} = \begin{bmatrix} W_{11} & W_{12} & 0 \\ 0 & W_{11} & W_{12} \end{bmatrix}. \tag{2.34}$$

This is a convolution matrix with a band diagonal structure (also called as Toeplitz matrix). It is important to note that the weights must be flipped before processing in the network according to the definition of a convolution. It is a *valid length* convolution operation between the input and weights. For N-dimensional input and

l-dimensional weight vector, the length of output sequence is $N - l + 1$. Hence, there are only two outputs (3-2+1=2) for given a example under consideration. The actual definition of the discrete linear convolution (with finite impulse response) is given as:

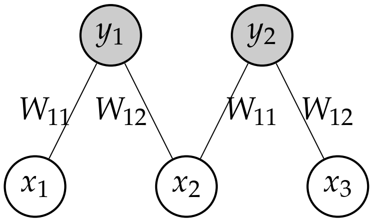


Figure 2.10: An example of simple 1-D CNN with valid length convolution.

$$y[n] = \sum_{k=0}^{N-1} x[n-k]w[k], \qquad \text{for } n = 0, .., M-1. \tag{2.35}$$

The length of the output sequence $y[n]$ is $N + M - 1$ and hence, the convolutional operation increases the output length. To incorporate the full length convolution in the convolutional network, zero padding is applied to the input as shown in Figure 2.11. In this example, the weight matrix can be written as:

$$\mathbf{W} = \begin{bmatrix} W_{11} & W_{12} & 0 & 0 & 0 \\ 0 & W_{11} & W_{12} & 0 & 0 \\ 0 & 0 & W_{11} & W_{12} & 0 \\ 0 & 0 & 0 & W_{11} & W_{12} \end{bmatrix}. \tag{2.36}$$

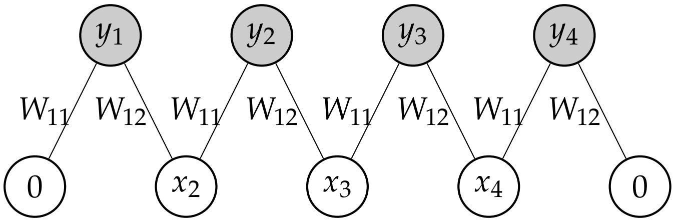


Figure 2.11: An example of simple 1-D CNN with full length convolution.

There are many notable differences between a dense and convolutional network. The first one is the local connections to the input layer that help to learn local structures in a signal. The second is a weight sharing that leads to reduced parameters compared to the dense network (as seen from matrix in eq. (2.34) with few zero entries). If we have K such groups of the hidden units and in each group neurons share weights, then any variable-sized input can be applied to the convolutional network.

2.5.2 Convolutional RBM

The convolutional RBM (denoted as ConvRBM or CRBM) was first proposed by Lee et al. in [59]. The ConvRBM has two layers, namely, the visible and hidden layers similar to the RBM. We describe ConvRBM with 1-D input signal $\mathbf{x} \in \mathbb{R}^{1\times N_x}$ in the visible layer and K number of groups in the hidden layer. The weights are denoted as $\mathbf{W} \in \mathbb{R}^{K\times m}$, where m is the convolution window length. The weights in the k^{th} group are denoted as $\mathbf{W}^k = w_r^k, r = 1, 2..., m$. The length of the hidden layer is determined as $N_h = N_x - m + 1$ (due to valid length convolution), and can change according to input length N_x. Hence, the ConvRBM has a flexibility of working with variable-sized input signals. Let us denote the visible bias as $c \in \mathbb{R}$ (since input is 1-D, visible bias is just a scalar) and hidden bias as $\mathbf{b} \in \mathbb{R}^{1\times K}$. The energy function of ConvRBM is given by [50]:

$$\begin{aligned} E(\mathbf{x},\mathbf{h}) = \frac{1}{2\sigma_x^2}\sum_{i=1}^{N_x} x_i^2 - \frac{1}{\sigma_x}\sum_{k=1}^{K}\sum_{j=1}^{l}\sum_{r=1}^{m}\left(h_j^k w_r^k x_{j+r-1}\right) \\ - \sum_{k=1}^{K} b_k \sum_{j=1}^{N_h} h_j^k - \frac{1}{\sigma_x^2} c \sum_{i=1}^{N_x} x_i, \end{aligned} \tag{2.37}$$

where σ_x is a standard deviation of the visible units. Generally, we normalize the input by performing a zero-mean and a unit standard deviation so that $\sigma_x = 1$ as suggested by Hinton in [60]. The equation for the joint probability distribution is same as eq. (2.19) of RBM. The model of ConvRBM proposed by Lee at el. in [59] also include probabilistic max-pooling after a convolution operation. Similar to CNN, probabilistic max-pooling reduces the ConvRBM responses by a constant factor. It helps to learn invariant representation and reduce computational cost for higher layers when used in belief network framework [59]. Probabilistic max-pooling is possible only when hidden units are binary. If hidden units are real-valued, it is difficult to introduce pooling while learning the ConvRBM. In that case, one can use the pooling on feature representation as used in our proposed work discussed in Chapter 3. In the next section, we will discuss the physiology of hearing from ear to the auditory cortex.

2.6 Auditory Processing

The human auditory system (HAS) is one of the engineering masterpieces of the human body, which is unique and distinct from other animals. In this Section, we review the HAS. Most of the discussion is motivated from the recent excellent book on auditory neuroscience [61].

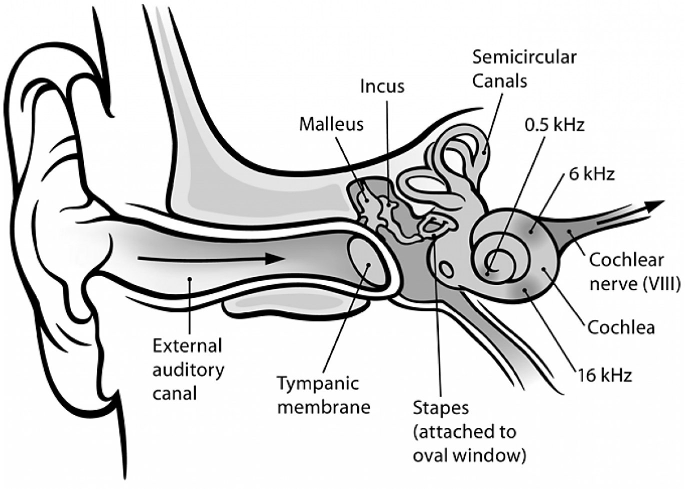


Figure 2.12: A cross-section inside the human ear to show the structures of the outer, middle, and the inner ear. Adapted from [16].

2.6.1 Early Auditory Processing of Sounds

The auditory processing on the incoming sound is based on various physiological aspects of the human ear. Such processing by the ear is also called as an early auditory processing [49], [62]. The anatomy of the human ear is shown in Figure 2.12. The hearing mechanism begins when the sound waves enter the ear canal and push against the eardrum. The eardrum separates the external (or outer) ear from the middle ear. The middle ear consists of three bones, known as the malleus, incus, and stapes [61]. The role of the middle ear is to transmit the tiny sound vibrations to the cochlea, where as the inner ear structure is responsible for sound encoding. The middle ear acts as a bridge between air-filled spaces of the outer and middle ear, and fluid-filled spaces of the cochlea. The cochlea is a coiled tube, a bone-like structure as shown in Figure 2.12 and is filled with the physiological fluids, specifically slightly salted water. The air propagated sound waves are too weak to impart similar size vibrations onto the fluid in the cochlea. To achieve an efficient transmission of sound from the air filled outer and middle ear to the cochlea, it is therefore necessary to concentrate the pressure of a sound wave onto a small spot, which is precisely the purpose of the middle ear [61].

There are two openings in the cochlea as shown in Figure 2.13 (b). The one is the oval window and another is the round window. Every time the stapes (a bone from the middle ear) pushes against the oval window, it increases the pressure in the fluid-filled spaces of the cochlea. The motion of the fluid column in the cochlea causes motion of the round window. A structure that subdivides the fluid-filled spaces in the cochlea is called the basilar membrane (BM), which runs along the entire length of the cochlea shown in Figure 2.13 (c) [61]. The BM has interesting mechanical properties. It is narrow and stiff at the basal end of the cochlea (i.e., near the oval and round window), however, wider and floppy at the other end of the cochlea (i.e., apex). The cochlea has two sources of mechanical resistance, one provided by the stiffness of the BM, and the other by inertia of the cochlear fluids [61]. Both resistances are graded along the cochlea. However, they run in the opposite directions. The inertial gradient increases as we move away from the oval window, but the stiffness gradient decreases and vice-versa. Since the inertial resistances are frequency-dependent, the path of the overall lowest resistance depends on the frequency [61]. It is long for low frequencies that are less affected by the inertia, and shorter for higher frequencies. Thus, if the stapes vibrates at low frequencies (order of few hundred Hz), it cause vibrations in the BM mostly at the apex. However, as we increase the frequency,the stapes causes vibrations towards the base end of the BM. Hence, the cochlea can act as a frequency analyzer. The cochlear frequency tuning characteristics are shown in

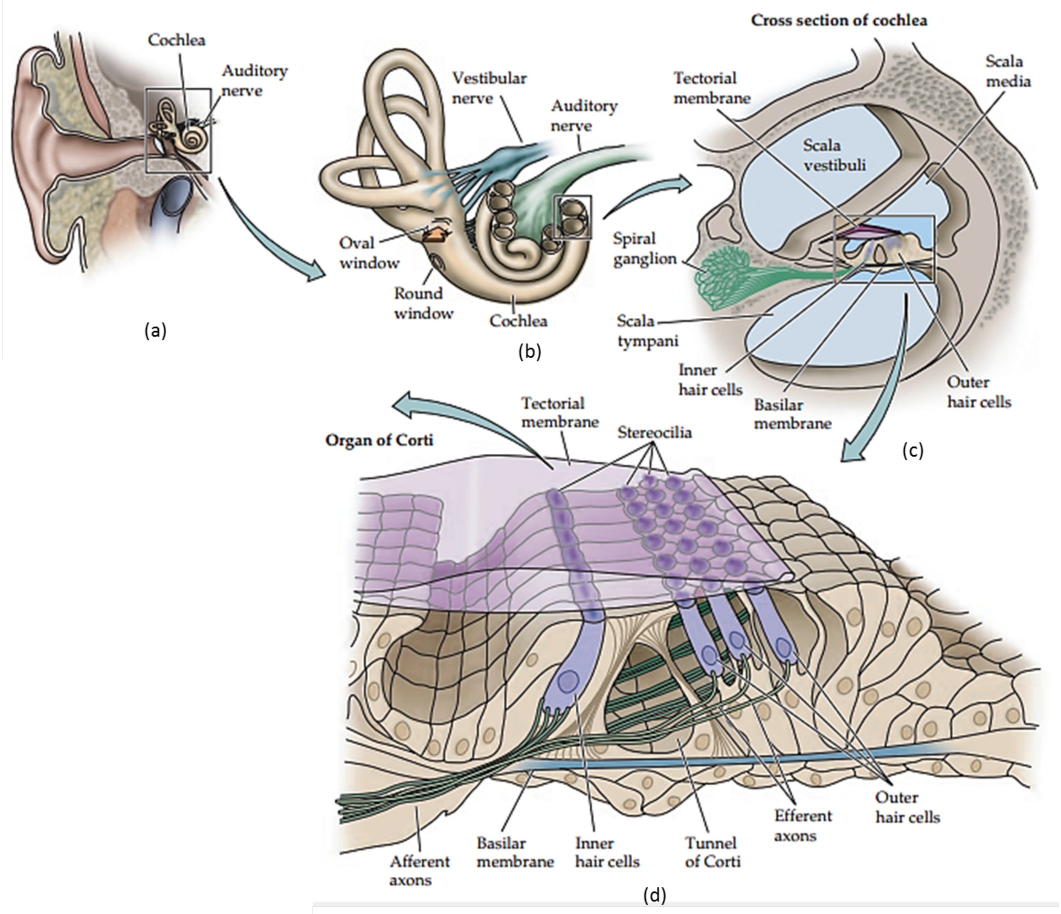


Figure 2.13: (a) Cross-section of human ear, (b) cochlea, (c) cross-section of cochlea, and (d) basilar membrane. Adapted from [17].

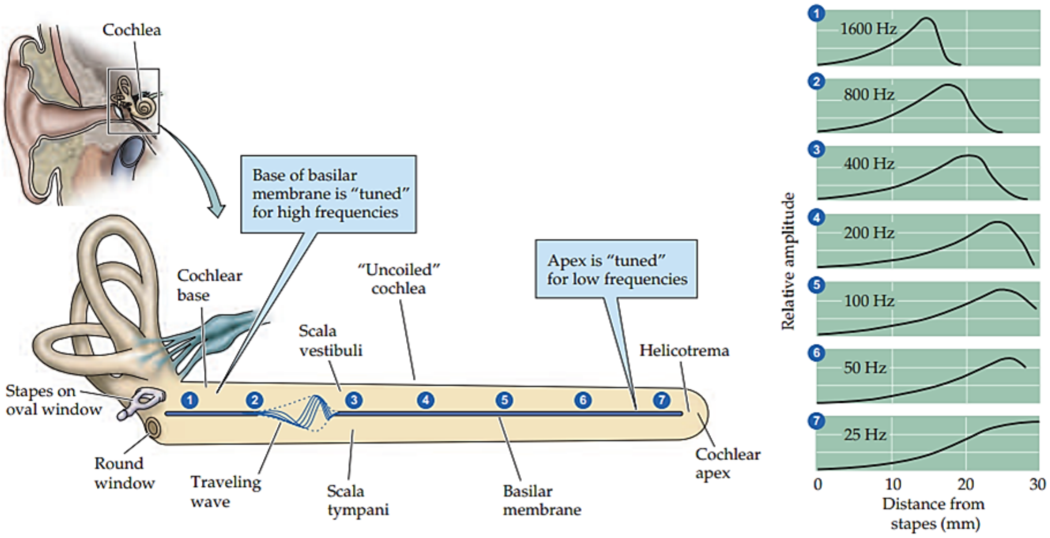


Figure 2.14: The traveling waves and frequency tuning in the cochlea. Adapted from [17].

Figure 2.14. Mathematically, it decomposes any complex sound into different frequency bands (also called the *subbands*). Each point of the BM has its own 'best frequency', a frequency that will make this point on the BM vibrate more than any other. Each small part of the BM, together with the fluid columns, forms a small mechanical filter element, each with its own resonance frequency (or center frequency). These resonance frequencies are determined by the membrane stiffness and fluid columns. One can compare the standard method of frequency analysis, the Fourier transform with the kind of transformation applied by the cochlea. However, unlike the frequency components of the Fourier transform, the cochlear filters are not placed at linear frequency intervals. Instead, their spacing is nonlinear, approximately logarithmic.

The mechanical filtering provided by the cochlea is neither linear, nor time-invariant. Still, a set of linear filters can provide a useful first-order approximation of the mechanical response of the BM. A set of auditory motivated filters is known as the gammatone filterbank as shown in Figure 2.15. The high frequency impulse responses are much faster than the low frequency ones, in the sense that they operate on a much smaller time window. The length of the temporal analysis window that is required to achieve a frequency resolution of about 12 % of the center frequency can be achieved with proportionally shorter time windows as the center frequency increases. This also explains why the impulse responses of the base, high frequency regions of the BM are shorter than those of apex, low frequency regions. After the decomposition of sound into different frequency components by the BM, the next stage is the conversion of mechanical vibrations of the BM into a

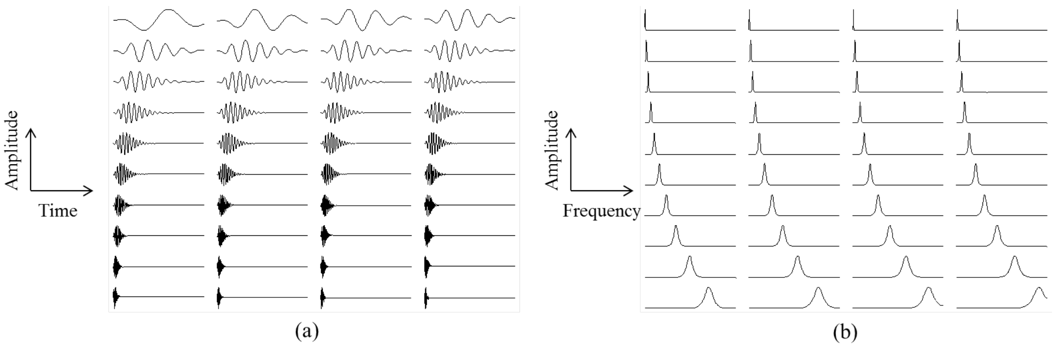


Figure 2.15: The gammatone filterbank: (a) time-domain impulse responses, and (b) corresponding frequency responses. After [18].

pattern of electrical excitaion. It can further be encoded by sensory neurons in the spiral ganglion of the inner ear for transformation to the auditory regions in the brain. The transformation from mechanical to electrical signals takes place in the organ of corti, a delicate structure in the BM. Figure 2.13 (d) shows the schematic drawing of a slice through the inner part of the cochlea. When the parts of the BM move up and down, the corresponding parts of the organ of corti will also move together with the membrane. The organ of corti has a folded structure as shown in Figure 2.13. There are rows of sensory hair cells on the foot of structure that sits directly on the BM. The organ of corti curves up and folds back over to form a little roof-like structure known as the tectorial membrane (shown by a purple glassy structure), which comes into close contact with the stercocillia (the hairs) on the sensory hair cells, inner and outer. The inner hair cells (IHC) form a single row of cells all along the BM, and they owe their name due to the fact that they are in inner parts of the organ of corti. Outer hair cells (OHC) form three rows of cells. The organization of the IHC and OHC along with the stercocillia is shown in Figure 2.13. The pattern of mechanical vibrations in the BM is converted into an analogous pattern of depolarizing current via IHC. The larger the deflection in the stereocilia, the greater the current. The amount of depolarizing current is in turn manifested in the IHC membrane potential. Hence, the voltage difference across the hair cell's membrane decreases and increases periodically in synchrony with the BM vibrations. At low frequencies, each cycle of stimulus is faithfully reflected in the sinusoidal change in the membrane potential. However, as the sound frequency increases, the individual cycles of the vibrations become less visible in the voltage response. The mathematical modeling of IHC is performed using the half-wave rectification.

While the job of the IHC is to communicate with other nerve cells, the job of the OHC is to vibrate in the sense of “moving in tune the rhythm of the sound".

The OHC provide a mechanical amplification of the vibrations in the BM produced by the incoming sound. The role of the OHC is very critical, and they can be easily damaged. The animals or people who suffer permanent damage to the OHCs are subsequently severely hearing impaired [61]. The careful psychoacoustical studies show that the OHC amplifies weaker sounds (low sound pressure level (SPL)) more strongly than the louder sounds. Mathematically, an operation that amplifies small values a lot, however, a large values only little bit, is called a “compressive nonlinearity". Hence, the role of OHC is represented by the compressive nonlinearity, such as logarithm or cubic nonlinearity.

The hair cells can be considered as neurons. Unlike typical neurons, they do not fire action potentials, and they have neither axons nor dendrites, however, they do have excitatory synaptic contacts with the neurons of the spiral ganglion [61]. These spiral ganglion neurons then form the long axons that travel through the auditory nerve to connect the hair cell receptors to the first auditory relay station in the brain called as the cochlear nucleus. The spiral ganglion cell axons are also known as auditory nerve fibers (ANF). The IHC and OHC are connected to different types of ANFs . More the particular region of the BM vibrates, the higher the firing rate of their ANFs. Furthermore, the BM tonotopy is preserved in the arrangements of the ANFs. Thus, the pattern of vibrations of membrane of the BM is translated into a neural “rate-place-code" in the auditory nerve. The firing rate distribution across the population of the ANFs produces a “neurogram" representation of sounds that resembles the filterbank responses of the sounds. The simplified stages of an early auditory processing are summarized in the Figure 2.16.

The hair cell membrane potentials encode low frequencies faithfully as analog, AC voltage signals. However, for frequencies higher than a few kHz, they switch into DC mode, i.e., not following individual cycles of the waveform. This behavior of the IHC is also reflected in the firing of the ANFs to which they connect. At low frequencies, as the IHC potential oscillates up and down in phase with the incoming sound, the probability of an action potential firing of ANFs also oscillates accordingly. Hence, for low stimulus frequencies, ANFs exhibit a phenomenon known as ‘phase locking’. The phase locking of a nerve fiber response is said to be stochastic, to reflect the residual randomness that arises because auditory nerve fibers may skip cycles, and their firing is not precisely time locked to the crest of the wave. The term “volly principle" is used to convey the idea, that if one fiber skips a particular cycle of sound stimulus, another nerve filter may mark it with a nerve impulse. This volly principle seems to make it possible for the auditory

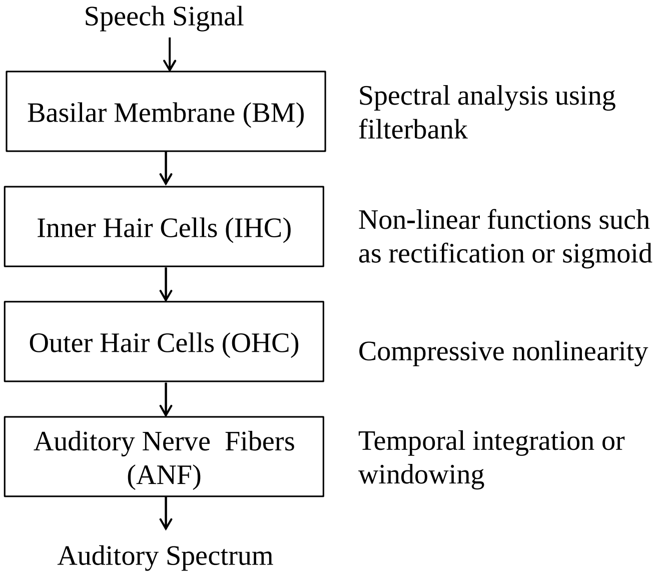

Speech Signal
Basilar Membrane (BM)
Spectral analysis using filterbank
Inner Hair Cells (IHC)
Non-linear functions such as rectification or sigmoid
Outer Hair Cells (OHC)
Compressive nonlinearity
Auditory Nerve Fibers (ANF)
Temporal integration or windowing
Auditory Spectrum

Figure 2.16: Simplified stages of an early auditory processing.

nerve to encode temporal fine structure (TFS) of the sounds at frequencies up to a few kHz. It is also said that high frequency ANFs are phase locked to the temporal envelope or amplitude modulations patterns, which ride on top of higher frequencies. All the outputs from the ANFs will reach the first major acoustic processing station of the mid-brain, the inferior colliculus (IC). Some of the auditory stations, not discussed here, play the central role in spatial hearing.

2.6.2 Cortical Representation of Sounds

The responses from the IC are sent to the auditory cortex as shown in Figure 2.17. The auditory cortex is also subdivided into a number of separate fields, some of which show clear tonotopic organization shown in Figure 2.17. The neurons in the auditory cortex process the temporal and spectral information jointly via their spectro-temporal receptive field (STRF). The STRF can be thought of as a two-dimensional impulse response of the neuron that characterizes it completely. Mathematically, it is denoted as $STRF(t, f)$. The linear response of an auditory neuron $r(t)$ is related to the input spectro-temporal representation $S(t, f)$ and $STRF(t, f)$ as follows [63]:

$$r(t) = \int \int STRF(t - \tau, f) \cdot S(t, f) d\tau df + \mathcal{E}(t), \tag{2.38}$$

where convolution is along the temporal dimension t and integration along the spectral dimension f. $\mathcal{E}(t)$ is the residual response not explained by the STRF

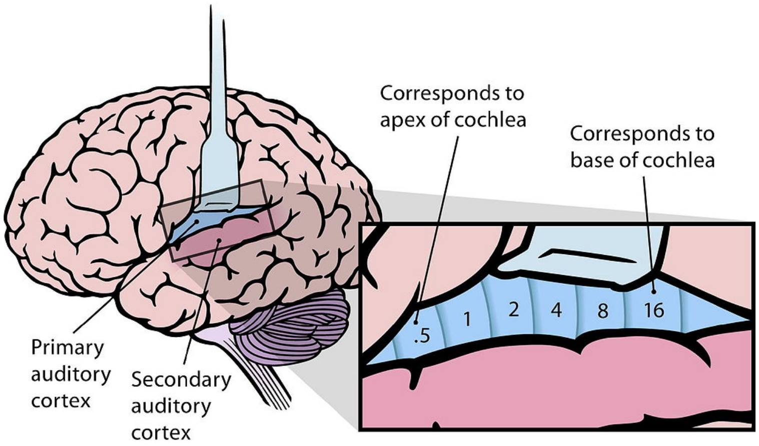


Figure 2.17: The lateral view of the human brain along with the auditory cortex areas exposed. An example of tonotopic organization of frequency tuning is shown here in the scale of 2. Adapted from [16].

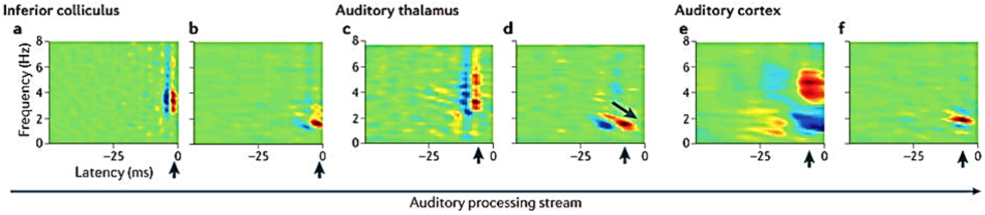


Figure 2.18: Examples of estimated STRFs at different auditory processing levels. Adapted from [19].

model. The STRF is computed from the responses to elementary *ripple* sounds, a family of sounds with the drifting sinusoidal spectral envelopes [63]. The collection of auditory neuron responses to all the elementary ripples is the estimated STRF [63]. The examples of STRFs estimated at different levels of auditory processing are shown in Figure 2.18. The red color shows the excitatory regions, i.e., it allows the corresponding spectro-temporal components in the neurons' responses. The blue color shows the inhibition regions that suppress the spectro-temporal components in the neuron's response. The complex spectro-temporal envelope of any dynamic sound can be expressed as the linear sum of individual ripples. Since the STRF can describe spectro-temporal responses, it is natural to think whether the spectral and temporal components are separable or not. It is observed that most of the STRF are either fully separable or quadrant separable [63]. The computational model to characterize the STRF is based on a 2-D Gabor/wavelet filterbank [49], [64]. In this thesis, we discuss our proposed model to learn the temporal receptive field (TRF), also known as the temporal response function (TRF). The TRF is estimated for each frequency subband, and hence, we can write the expression of the TRF from the STRF eq. (2.38) as [65]:

$$r(t) = \int TRF(t-\tau) \cdot S_i(t) df + \mathcal{E}(t), \tag{2.39}$$

where $S_i(t)$ is the spectral envelope in the i^{th} subband. A detailed review of the neural processing of sounds is given in [19]. In the next section, we will define deep learning and present the state-of-the-art deep learning architectures.

2.7 Deep Learning

Since 2006, the new area of machine learning has emerged that has significant impact in many signal processing applications, such as speech, images, biomedical, etc. Since the early techniques started with learning from unlabeled data [66], it is known as *representation learning*. Later impressive results also achieved using supervised learning techniques by using many parallel data processing units with nonlinearities. It is now called *deep learning* or *hierarchical learning*. There are various definitions of representation learning or deep learning.

Representation learning: It is defined as learning the representation of the data that makes it easier to extract the meaningful and useful information, when building classifiers or other predictors for signal processing applications [40].

Deep learning: Deep learning is a new area of machine learning research, which has been introduced with the objective of moving machine learning closer to one of its original goals: Artificial Intelligence (AI). Deep learning is about learning multiple levels of representation, and abstraction that helps to make sense of data, such as speech and audio, images, and text [67].

Comparison of classical machine learning and representation techniques is shown in Figure 2.19. In a classical or traditional machine learning approach, the input signal is first processed with signal processing techniques to obtain the meaningful features followed by a supervised classifier to obtain the mapping corresponding to the given task. Such features are called the handcrafted (or hand-designed) features [20]. In the representation learning approach, the meaningful features are learned from the data itself, without requiring any specific domain knowledge or the need of a specialized signal processing. If we increase the feature learning technique to learn low-level to high-level features through the multiple learning stages (either layerwise training or joint training), it is called deep learning as defined above. We discuss the three types of deep learning architectures based on artificial neural networks (ANN) that are used in this study either for acoustic modeling in the ASR task or for the audio classification task.

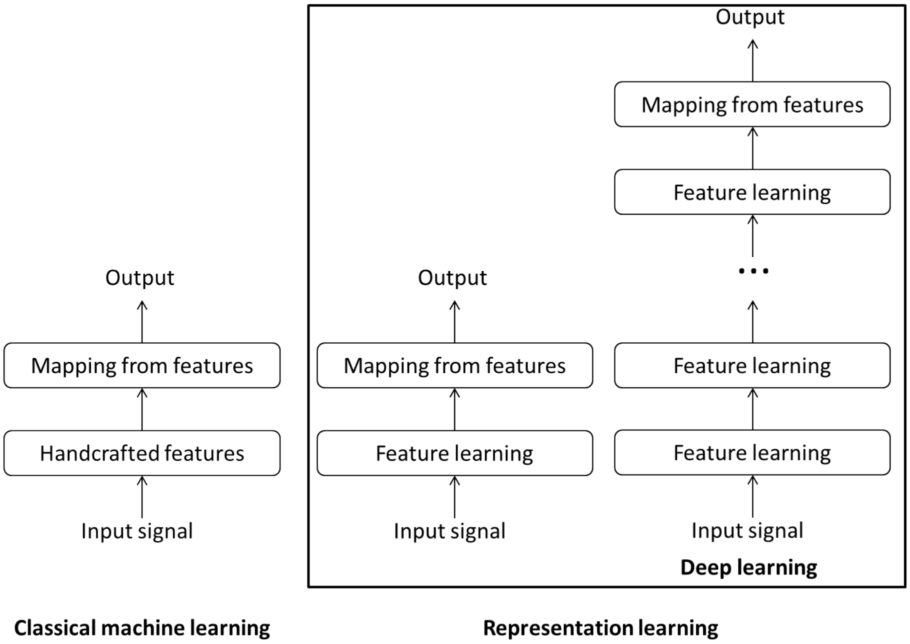


Figure 2.19: Comparison of classical machine learning and representation techniques. After [20].

2.7.1 Deep Neural Networks (DNN)

Feedforward neural networks (FFNN) or multilayer perceptrons (MLP) are used in signal processing tasks from the late 90s. However, for a large scale task, such as speech recognition, these neural networks requires a high computational power to train the models. The resurgence of neural networks is possible because of two key factors: (1) massive amount of data available, and (2) parallel processing using Graphical Processing Units (GPU). The goal of the FFNN is to approximate the function f. FFNN learns the value of parameters of the network θ, and finds an appropriate mapping, $\mathbf{y} = f(\mathbf{x};\theta)$. The network is called the MLP when we have more than one layer, and called DNN when it has more than two-three layers [20]. The example of a three layer DNN is shown in Figure 2.20. The input is connected to all the neurons in the first hidden layer followed by two more fully connected hidden layers (dense connections as discussed in Section 2.5.1). The outputs of neurons in the final layer are the posterior probability of a particular class.

For the input signal $\mathbf{x}$, the neurons' activations in the l^{th} hidden layer are calculated as [20]:

$$\mathbf{h}^{(l)} = g\left(\mathbf{W}^{(l)T}\mathbf{x} + \mathbf{b}^{(l)}\right), \tag{2.40}$$

where $g(\cdot)$ is the nonlinear activation function, $\mathbf{W}^{(l)}$ is a weight matrix, and $\mathbf{b}^{(l)}$ is a bias vector. The output neurons in the DNN use the softmax nonlinearity. The

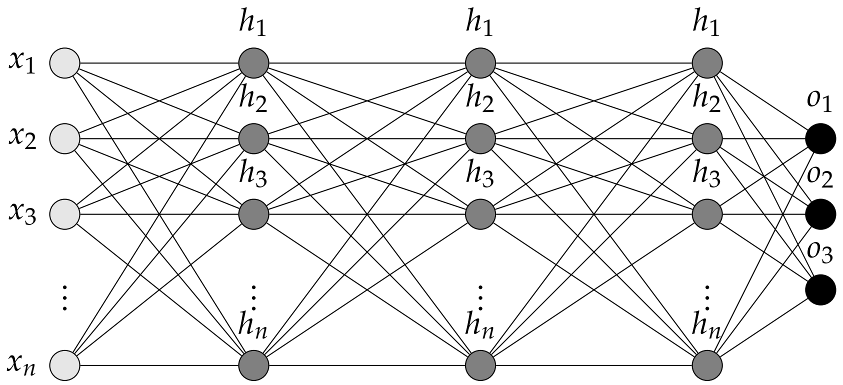


Figure 2.20: An example of the three layer DNN.

softmax function can be seen as a generalization of the sigmoid function, which is used to represent a probability distribution function over a binary variable. The softmax functions are used as the output of a DNN classifier, to represent the probability distribution function over n different classes. The expression for the softmax function is given by:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}, \tag{2.41}$$

where $\mathbf{z} = \mathbf{W}^T\mathbf{x} + \mathbf{b}$. From a neuroscience perspective, it is interesting to think of the softmax function as a way to create a form of competition between the units that participate in it [20]. The softmax outputs always sum to 1 and hence, an increase in the value of one unit necessarily corresponds to a decrease in the value of others [20].

The hidden units in the DNN have many choices to use activation functions. The older DNN networks use sigmoid function $g(\cdot) = \text{sigmoid}(\cdot)$ defined as [20]:

$$\text{sigmoid}(\mathbf{W}^{(l)T}\mathbf{x} + \mathbf{b}^{(l)}) = \frac{1}{1 + e^{-\left(\mathbf{W}^{(l)T}\mathbf{x}+\mathbf{b}^{(l)}\right)}}. \tag{2.42}$$

The sigmoid function is bounded in the range [0,1]. This sometimes causes the sigmoid function to go in the saturation regions, which leads to the vanishing gradient problem [20]. Hence, recently, rectifier linear units (ReLU) are more popular, which is defined as [20]:

$$g(\mathbf{W}^{(l)T}\mathbf{x} + \mathbf{b}^{(l)}) = \max(0, \mathbf{W}^{(l)T}\mathbf{x} + \mathbf{b}^{(l)}). \tag{2.43}$$

It can be seen that this suppresses the negative values of the neuron's input. The ReLU has a range [0,∞]. There are several advantages of ReLU over the sigmoid function as listed below:

- **Sparsity**: Since ReLU makes the negative values of the neuron's input to zero, it enforces sparsity in the neuron's activation values [68]. It is mathematically proved that ReLU can lead to sparsity in the DNN [69].
- **Fast convergence**: It has been empirically shown that ReLU leads to faster learning of DNN. In addition, the computations are also very cheap (economical): there is no need for computing the exponential function in activations as needed in the sigmoid units [68].
- **No need for pre-training**: Earlier in order to train DNN with more number of layers, DNNs are pre-trained using unsupervised learning techniques [60]. However, it is shown that when ReLU is used as an activation function, it achieves similar performance as pretrained DNN [68].
- **Biologically motivated**: ReLU is motivated from the biologically plausible leaky integrate-and-fire (or LIF) neuron activation function [68].

There are many variants of ReLU proposed to further improve the performance, such as leaky ReLU (LReLU), and parametric ReLU (PReLU). The parameters of the DNN are optimized using the back-propagation algorithm [70].

2.7.2 Convolutional Networks

The convolutional networks are a specialized kind of neural network for processing data that has known grid-like topological structures. If they are based on neural networks, it is known as convolutional neural networks (CNN), first proposed in [71] and implemented by [72]. The convolutional networks employ a mathematical operation called the convolution, a linear operator. CNNs are simply neural networks that use a convolution operation in place of general matrix multiplication compared to the DNN in at least one of their layers [20]. We have discussed the difference between dense connections *vs*. convolutional connection in Section 2.5.1. Here, we discuss specifically the architecture of the supervised CNN. The CNN architecture has a convolution layer followed by fully-connected layers. Various stages in the convolution layer are shown in Figure 2.21. The convolution stage computes the convolution between the input signal and weights known as *convolution kernals* or *filters*. The detection stage includes generally the ReLU nonlinearity. In the third stage, the dimensions of the output of a detection stage are reduced by a pooling or subsampling method. Two popular pooling operations are average and max-pooling defined as follows:

$$P_{avg} = \frac{1}{n_p} \sum_{i=1}^{n_p} \mathbf{x}_i, \tag{2.44}$$

$$P_{max} = \max_{i=1:n_p}(\mathbf{x}_i), \tag{2.45}$$

where $\mathbf{x}_i$ are the neuron's activation values, and n_p is the length of the pooling region. In the subsampling technique, the features are uniformly downsampled instead of taking an average or max operation.

The pooling stage helps to make the representation become approximately invariant to small translations of the input signal [20]. The invariance to translation means that if one translates the input signal by a small amount (either in time, frequency or both), the activation values of most of the pooled outputs remain changed [20]. The convolutional networks are inspired from the neuroscientific discovery of a simple and complex cells in the visual cortex by Hubel and Wiesel, which led them to win the Noble Prize. The role of a convolution layer is similar to simple cells while the detection and pooling layer function as complex cells [20].

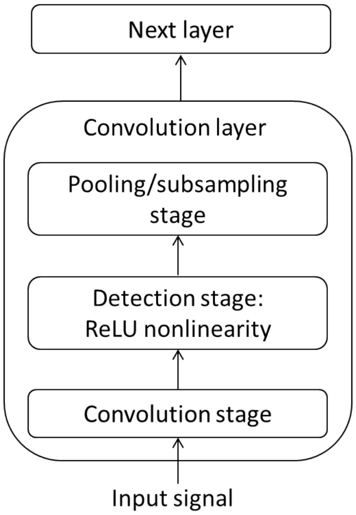


Figure 2.21: Various stages in the CNN architecture. After [20].

The convolution networks have a property called equivariance to translation [20] defined as follows [73]:

Definition: Let $f : (X, \rho) \rightarrow (X, \rho)$ be applied on the input signal x, where (X, ρ) is a metric space. Let a translation operator $\mathcal{T}$ on the function f be [73]:

$$\mathcal{T}(f(x)) = f(x - x^{'}), \quad x, x^{'} \in X. \tag{2.46}$$

The function f is said to be invariant to the translation if $\mathcal{T}(f(x)) = f(x)$ and equivariant if $f(\mathcal{T}(x)) = \mathcal{T}(f(x))$. The convolution operator is translation equivariant while the pooling stage provides some form of translation-invariance [20]. Earlier, Lang and Hinton [74] introduced the use of back-propagation to train time-delay neural networks (TDNNs). The TDNNs are actually one-dimensional convolutional networks applied to time series data [20]. Recently, the TDNN concept is revived in [75] and successfully applied in the ASR task. Next, we discuss the neural networks known as the recurrent neural networks (RNN).

2.7.3 Recurrent Neural Networks

To model sequential data, such as time series, speech, etc., recurrent neural networks (RNN) are the first choice. We can understand the RNN structure in a graphical form. Consider the classical form of a dynamical system [20]:

$$\mathbf{s}_{(t)} = f(\mathbf{s}_{(t-1)}; \theta), \tag{2.47}$$

where $\mathbf{s}_{(t)}$ is called as the *state* of the system. This equation is recurrent (occurring repeatedly or often) since the state $\mathbf{s}$ at time t refers back to the same definition at time $t-1$. The graph for this dynamical system can be unfolded by applying the definition $\tau - 1$ times for a finite number of time steps, τ. Unfolding the equation by repeatedly applying the definition eq. (2.47) has yielded an expression that does not involve recurrence. A directed acyclic computational graph is used to represent the above expression [20].

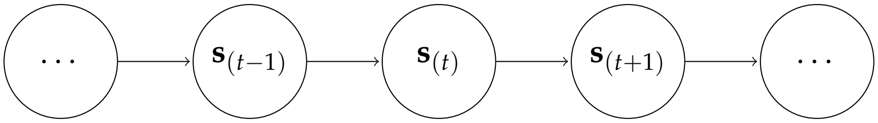


Figure 2.22: The graph representation of the dynamical system described by eq. (2.47). After [20].

Now, consider the dynamical system driven by an external signal $\mathbf{x}^{(t)}$:

$$\mathbf{s}_{(t)} = f(\mathbf{s}_{(t-1)}, \mathbf{x}_{(t)}; \theta), \tag{2.48}$$

where it can be seen that the state now contains information about the whole past sequence. In the context of RNN, we can rewrite equation (2.48) using the hidden variables $\mathbf{h}$ as follows:

$$\mathbf{h}_{(t)} = f(\mathbf{s}_{(t-1)}, \mathbf{x}_{(t)}; \theta). \tag{2.49}$$

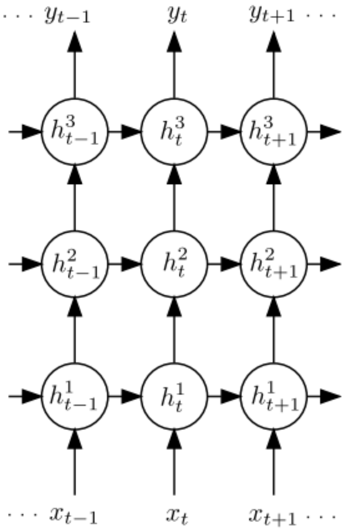


Figure 2.23: The unfolded deep RNN architecture including input, hidden and output nodes. Adapted from [20].

RNN also includes nodes to read information out of the state $\mathbf{h}^{(t)}$ to make the predictions. The example of the deep RNN in the form of an unfolded graph with the three hidden layers is shown in Figure 2.23. The use of back-propagation on the unrolled RNN is called the back-propagation through time (BPTT) algorithm [20]. The basic problem in the RNN training is that the gradients propagated over many stages tend to either vanish (most of the time) or explode, i.e., making weights and biases infinity (rarely, but with much damage to the optimization) [20]. The gradient vanishing problem arises due to large multiplications of the sigmoid functions in the RNN in gradient computation. Even if it is assumed that the parameters are such that the RNN will be stable (i.e., no gradient exploding), the difficulty with long-term dependencies arises from the exponentially smaller weights given to long-term interactions compared to the short-term ones.

The most effective and popular sequence models used in the practical applications are called gated RNNs. These include the long short-term memory (LSTM) and the bidirectional LSTM (BLSTM). The gated RNNs are based on the idea of creating paths through time that have derivatives that neither vanish nor explode [20]. The LSTM model is based on introducing self-loops to produce the paths, where the gradient can flow for longer durations. Using the gate controlled by the hidden unit, the time scale of integration can be changed dynamically. In this case, even for an LSTM with fixed parameters, the time scale of integration can change based on the input sequence lengths, since the time constants are output by the model itself [20]. The LSTM block diagram is shown in Figure 2.24.

An input feature is computed with a regular ANN unit such as sigmoid or hy-

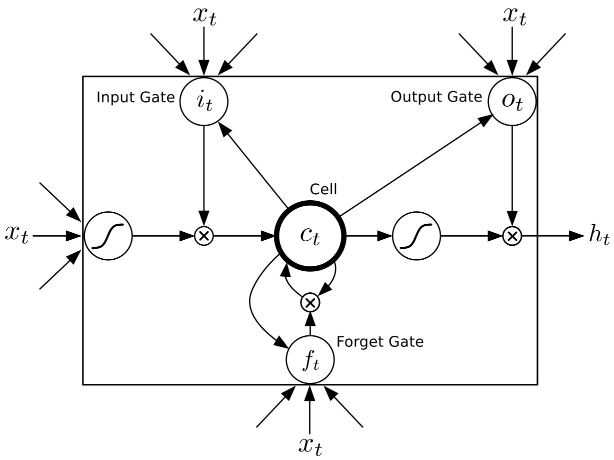


Figure 2.24: An example of LSTM cell. Adapted from [21].

perbolic tangent. Its value can be accumulated into the state if the sigmoidal input gate allows it. The state unit has a linear self-loop whose weight is controlled by the forget gate. The output of the cell can be shut off by the output gate. All the gating units have a sigmoid nonlinearity, while the input unit can have any squashing nonlinearity, such as sigmoid or hyperbolic tangent. The state unit can also be used as an extra input to the gating units. The equations for these units are written as follows [21]:

$$i_t = \tanh(W_{xi}x_t + W_{hi}h_{t-1} + b_i), \tag{2.50}$$

$$j_t = \sigma(W_{xj}x_t + W_{hj}h_{t-1} + b_j), \tag{2.51}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f), \tag{2.52}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o), \tag{2.53}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes j_t, \tag{2.54}$$

$$h_t = \tanh(c_t) \otimes o_t, \tag{2.55}$$

where **W** and **b** are the biases of the LSTM cell. Here, $\otimes$ denotes an element-wise multiplication. Instead of only using previous context for the sequence prediction, we can use the future context also in a form of bidirectional LSTM (BLSTM) [21]. The block diagram of BLSTM is shown in Figure 2.25. One can also combine different DNN along with the RNN models as done in [76].

2.8 Automatic Speech Recognition (ASR)

The task of an ASR system is to convert a speech signal into a sequence of words (or phonemes) in the text format with the help of a machine. The ultimate goal of the ASR task is to enable people to communicate with the machines in the form of

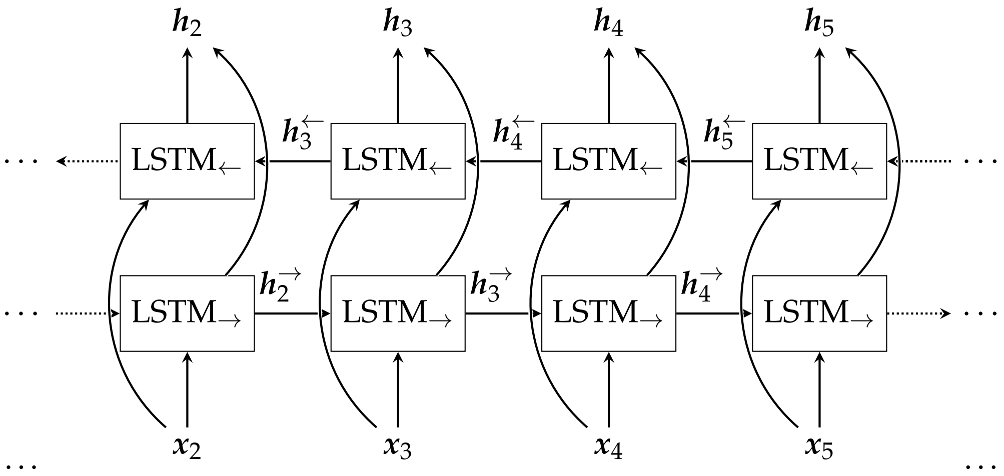


Figure 2.25: Architecture of the BLSTM network. After [21].

human-computer interaction. There are many potential applications of the ASR that includes call centers, voice dialing, data entry and dictation, command and control, computer-aided language learning, etc. In recent days, ASR along with text-to-speech synthesis (TTS) is used effectively in chat bots and in mobile phones as an intelligent personal assistant (e.g., Apple Siri, Ok Google, Amazon Echo). The modern ASR problem is based on statistical pattern recognition along with speech signal processing as a front end.

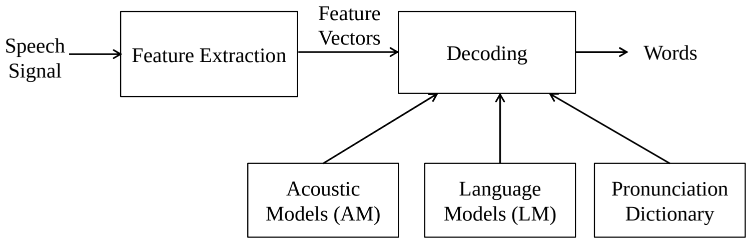


Figure 2.26: The architecture of the statistical ASR system. After [22].

The principal components of the ASR system are shown in Figure 2.26. The input speech signal is first converted into a sequence of feature vectors through feature extraction block. A typical feature extraction procedure is to transform a raw speech signal into a time-frequency representation via auditory filterbank or Short-Time Fourier Transform (STFT) followed by the auditory scale. Let us denote the feature vector sequence as $\mathbf{X} = \mathbf{x}_1, ..., \mathbf{x}_F$, where F is the number of frames. The goal of the decoder is to find the optimal word sequence $\tilde{\mathbf{Y}}$ through the fundamental equation of the ASR given by:

$$\tilde{\mathbf{Y}} = \arg\max_{\mathbf{Y}} P(\mathbf{Y}|\mathbf{X}). \tag{2.56}$$

However, in the generative models, it is difficult to find the posterior probability $P(\mathbf{Y}|\mathbf{X})$ directly. Hence, applying Bayes' rule to eq. (2.56), we get

$$\begin{aligned}\tilde{\mathbf{Y}} &= \arg\max_{\mathbf{Y}} \frac{P(\mathbf{Y})P(\mathbf{X}|\mathbf{Y})}{P(\mathbf{X})}, &(2.57)\\ &\approx \arg\max_{\mathbf{Y}} P(\mathbf{Y})P(\mathbf{X}|\mathbf{Y}). &(2.58)\end{aligned}$$

Since the maximization is done with a fixed observation $\mathbf{X}$, $P(\mathbf{X})$ will not take part in optimization. The likelihood $P(\mathbf{X}|\mathbf{Y})$ is determined by the acoustic model and the prior $P(\mathbf{Y})$ is determined by the language model. Apart from feature representation, the major challenge in statistical ASR is to build accurate acoustic and language models. We will discuss various components of the ASR briefly in the upcoming sub-sections.

2.8.1 Feature Extraction

The feature extraction stage generally converts the speech signal into a time-frequency representation using the perceptual/auditory filterbank. Typically, the Mel frequency scale is used for feature extraction; however, significant efforts are devoted to find better representation using speech signal processing and machine learning techniques [38], [39], [77]. Here, we discuss the very popular and state-of-the-art Mel filterbank first used in [78] for the ASR task. The Mel scale is a perceptual scale proposed by Stevens, Volkmann, and Newman in 1937 [79]. The name Mel comes from the word *melody* to indicate that this frequency scale is based on perceived pitch comparisons [79]. The mathematical formula to convert the frequency f_{Hz} in Hz to f_{Mel} in Mel is given as [79]:

$$f_{Mel} = 2595 \log_{10}\left(1 + \frac{f_{Hz}}{700}\right). \tag{2.59}$$

The nonlinear Mel frequency scale is shown in the Figure 2.27 along with the Equivalent Rectangle Bandwidth (ERB) and the Bark scales [39] defined as:

$$\begin{aligned}f_{Bark} &= \left[\frac{26.81 f_{Hz}}{(1960 + f_{Hz})}\right] - 0.53, &(2.60)\\ f_{ERB} &= 21.4 \log_{10}\left(1 + 0.00437 f_{Hz}\right). &(2.61)\end{aligned}$$

The Mel filterbank derived in [78] has the triangular-shaped filters in the frequency-domain as shown in Figure 2.28. One can see that the filters are becoming broader

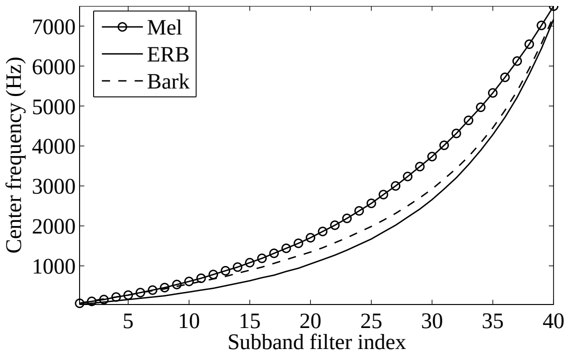


Figure 2.27: The comparison of standard auditory-based frequency scales.

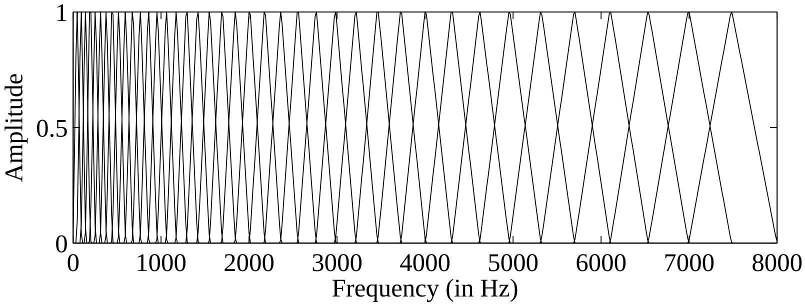


Figure 2.28: An example of a Mel filterbank with 40 subband filters.

as the frequency increases. This is due to the nonlinear frequency scale as shown in Figure 2.27. The Mel spectrogram is shown in Figure 2.29. The important aspect of the Mel spectrogram is that it reduces the spectral harmonics so that formants can be more effectively represented. However, the time-frequency resolution decreases as the frequency increases due to the averaging by the Mel subband filters.

To use Mel spectrogram-based features in the ASR task, a Discrete Cosine Transform (DCT) is applied to reduce the dimension and decorrelate the features. Further by selecting only few coefficients (generally 13), it also eliminates the source effect due to homomorphic nature of speech signal processing [80]. These reduced features are called as the Mel Frequency Cepstral Coefficients (MFCC), a state-of the-art features in many speech processing applications [78]. The dynamic features known as first-order delta and second-order delta features (also called delta-delta or acceleration coefficients) were also added to the static MFCC features. If the feature vector is $\mathbf{x}_t$ at frame index t, the first-order delta features $\Delta\mathbf{x}_t$ are calculated as follows [22]:

$$\Delta\mathbf{x}_t = \frac{\sum_{i=1}^{n} d_i(\mathbf{x}_{t+i} - \mathbf{x}_{t-i})}{2\sum_{i=1}^{n} d_i^2}, \tag{2.62}$$

where n is the window length and d_i are the regression coefficients. The second-order delta features denoted as ΔΔ are derived in a similar manner using the first-order delta $\Delta\mathbf{x}_t$ features. In the DNN-based approaches for ASR, it is shown that Mel spectrograms showed improved performance compared to the DCT-based MFCC features. It is shown that DNNs can better model correlated features [81]. The Mel spectrograms are now state-of-the-art features in the DNN-based ASR [58]. Other auditory-based features include perceptual linear prediction (PLP) [82] and gammatone-based models [83]. The PLP features are based on the Bark frequency scale while the gammatone filterbank uses the ERB scale. A detailed discussion on various features for the ASR task is presented in [38], [77], and [84]. Next, we discuss the details of statistical acoustic modeling for ASR.

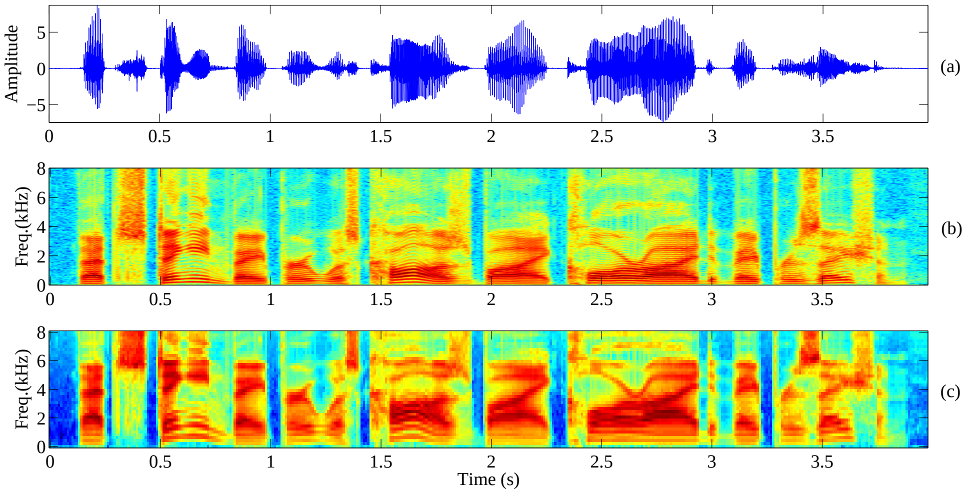


Figure 2.29: (a) Speech signal, (b) STFT, and (c) Mel spectrogram. The utterance is “she had your dark suit in greasy wash water all the year".

2.8.2 Acoustic Modeling

The role of acoustic model is to improve the recognition accuracy by combating variations in speakers, dialects, environment, and noise. It is believed in the research community that the acoustic model is the central part of any ASR system. Acoustic modeling of speech signals refers to the process of establishing the statistical representation for the feature vector sequences computed from the speech signals. The Hidden Markov Models (HMM) are the most popular type of statistical model used for acoustic modeling since for a long time [85], [86]. However, recently deep learning-based frameworks are becoming popular for acoustic modeling which will be discussed in Section 2.8.5. Acoustic modeling also includes “pronunciation modeling" that describes how a sequence or multi-sequence of the fundamental speech units (e.g., phones) is used to represent larger speech sound units, such as words or phrases.

2.8.2.1 Acoustic Modeling using GMM-HMM

Each spoken word w is decomposed into a sequence of N_w basic speech sound units called as *base phones*, a fixed set of basic sound units for a given language. This sequence is called its pronunciation denoted as $\mathbf{q}^w = q_1, q_2, ..., q_{N_w}$. Generally, this pronunciation of words is supplied via what is called the pronunciation dictionary, which contains the phonetic decomposition of words. Multiple pronunciations are allowed by computing the likelihood $P(\mathbf{X}|\mathbf{Y})$ that can be computed over multiple pronunciations as follows:

$$P(\mathbf{X}|\mathbf{Y}) = \sum_{\mathbf{Q}} P(\mathbf{X}|\mathbf{Q})P(\mathbf{Q}|\mathbf{Y}), \tag{2.63}$$

where the summation is all over the valid pronunciation sequences for $\mathbf{Y}$, and $\mathbf{Q}$ is a particular sequence of pronunciations. The probability $P(\mathbf{Q}|\mathbf{Y})$ is given by:

$$P(\mathbf{Q}|\mathbf{Y}) = \prod_{l=1}^{L} P(\mathbf{q}^{w_l}|w_l), \tag{2.64}$$

where each $\mathbf{q}^{w_l}$ is a valid pronunciation for the word w_l. Each base phone q is presented by a continuous density HMM as shown in Figure 2.30. It is assumed that the sequence of all observed features vectors are generated by a Markov chain. Generally, a left-to-right form of an HMM is used as shown in the Figure 2.30. The transition probabilities are denoted as $\{a_{ij}\}$ and the output emission probabilities are denoted as $\{b_j(\cdot)\}$. The probability of making a transition from state s_i to s_j is given by the transition probability a_{ij}. An HMM is a finite state machine (FSM) that changes a state once every time frame t when a state j is entered. Then the observation feature vector $\mathbf{x}_t$ is generated from the emission probability distribution $\{b_j(\mathbf{x}_t)\}$ [87]. This is due to the conditional independence assumptions in HMM [88], [22]:

- The HMM states are contitionally independent of all other states given the previous state.
- The observations are conditionally independent of all other observations given the HMM state that generated it.

In the left-to-right model of an HMM, two extra non-emitting states are also used called an entry state, and an exit state at the entry of speech feature vector genera-

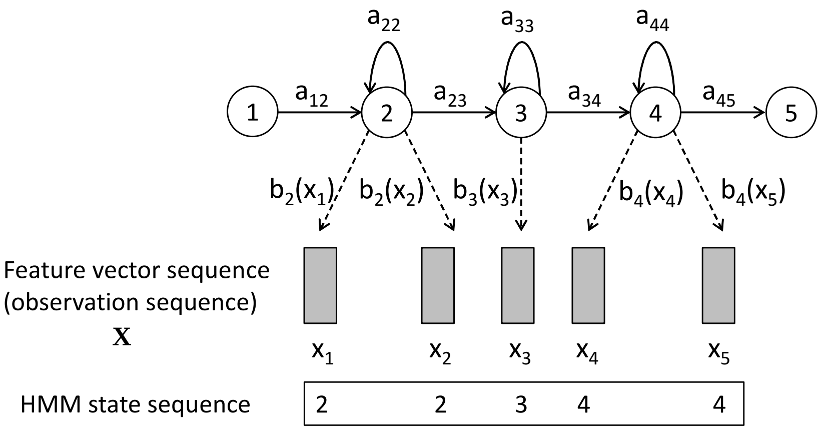


Figure 2.30: An example of HMM-based phone model. Adapted and modified from [22].

tion, and the exit of the generation process, respectively. The emitting probability density $b_j(\mathbf{x})$ describes the distribution of the observation vectors at the state j. In the continuous density HMM (CD-HMM), a Gaussian Mixture Model (GMM) is used to represent the emission probability density as follows [87]:

$$b_j(\mathbf{x}) = \sum_{m=1}^{M} c_{jm}\mathcal{N}\left(\mathbf{x}; \mu_{jm}, \Sigma_{jm}\right), \tag{2.65}$$

where the GMM is represented as:

$$\mathcal{N}\left(\mathbf{x}; \mu_{jm}, \Sigma_{jm}\right) = \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma_{jm}|^{\frac{1}{2}}} e^{-(\frac{1}{2})(\mathbf{x}-\mu_{jm})^T \Sigma_{jm}^{-1}(\mathbf{x}-\mu_{jm})} \tag{2.66}$$

is a multivariate Gaussian density for D-dimensional feature vector $\mathbf{x}$. $c_{mj}, \mu_{jm}, \Sigma_{jm}$ are the weight, mean, and covariance of the m^{th} Gaussian mixture component at state j, respectively. Since the dimensionality of the acoustic vector $\mathbf{x}$ is relatively high, the covariances Σ_{jm} are often constrained to be a diagonal matrix [22]. Such form of acoustic modeling is known as the GMM-HMM technique, where the GMM is used to estimate the feature vector probability density in the HMM and the Markovian nature of HMM is used for the temporal sequence modeling. A more detailed discussion on HMMs is given in an excellent tutorial by Rabiner in [89]. The mathematical treatment of the HMMs considering the statistical properties of random variables and conditional independence is given in [88]. Given the composite HMM, $\mathbf{Q}$ is formed by concatenating all of the base phones

$\mathbf{q}^{(w_1)}, ..., \mathbf{q}^{(w_L)}$. Then the acoustic likelihood is given by:

$$P(\mathbf{X}|\mathbf{Q}) = \sum_{\theta} P(\theta|\mathbf{X}, \mathbf{Q}), \tag{2.67}$$

where θ is a state sequence through the composite HMM model and

$$P(\theta|\mathbf{X}, \mathbf{Q}) = a_{\theta_0\theta_1} \prod_{t=1}^{T} b_{\theta_t}(\mathbf{x}_t) a_{\theta_t\theta_{t+1}}, \tag{2.68}$$

where θ_0 and θ_{T+1} correspond to the non-emitting entry and exit states shown in Figure 2.30, included to simplify the process of concatenating HMM phone models to make the words. The acoustic model parameters, a_{ij} and $b_j(\cdot)$ can be efficiently estimated from the feature vectors of the training utterances using the forward-backward algorithm [89], which is an example of the Expectation Maximization (EM) algorithm [90]. So far, we discussed how the speech feature vectors are represented by concatenating a sequence of HMM phone models together. However, we ignored the context-dependent variations in the speech. For example, the pronunciation of the vowel “a" in the words “bat" and “ball" are different. Such context-free phone models are referred to as monophones [22]. A simple way to incorporate the context in the phones is to use a unique HMM phone model for every possible pair of left and right neighbors of the phones. The resulting HMM models are called as *triphones* [22].

2.8.3 Language Modeling

The role of the language model in ASR is to provide the value $P(\mathbf{Y})$ in the fundamental equation of the ASR (eq. (2.58)). The probabilistic relationship between a sequence of words can be directly derived and modeled from a text corpus with a large number of words. These probabilistic models are called stochastic language models or N-grams. A language model can be formulated as a probability distribution $P(\mathbf{Y})$ over a word string $\mathbf{Y}$ that reflects how frequently a string $\mathbf{Y}$ occurs as a sentence [87]. The $P(\mathbf{Y})$ can be decomposed as:

$$P(\mathbf{Y}) = P(y_1, y_2, ..., y_N), \tag{2.69}$$

$$= \prod_{i=1}^{N} P(y_i|y_1, y_2, ..., y_{i-1}), \tag{2.70}$$

where $P(y_i|y_1, y_2, ..., y_{i-1})$ is the probability that y_i will follow given the previous word sequence, $y_1, y_2, ..., y_{i-1}$. For a large vocabulary continous speech recognition (LVCSR) task, the word history is truncated to $N-1$ words due to computational issues, which leads to an N-gram language model. If the current word depends on the previous word, we have the bi-gram model $P(y_i|y_{i-1})$, and if the word depends on two previous words, we have a trigram model $P(y_i|y_{i-2}, y_{i-1})$. The probabilities in the N-gram model are estimated from the training text corpus by counting N-gram occurrences to form the maximum likelihood estimates [22]. For example, in the case of a trigram model, the word probabilities can be estimated by the frequency of occurrences or the counts of the word pair $C(y_{i-2}, y_{i-1})$, and triplet $C(y_{i-2}, y_{i-1}, y_i)$ as follows:

$$P(y_i|y_{i-2}, y_{i-1}) = \frac{C(y_{i-2}, y_{i-1}, y_i)}{C(y_{i-2}, y_{i-1})}. \tag{2.71}$$

A more detailed treatment of language modeling can be found in the book [86]. Next, we discuss briefly about the decoding process of ASR.

2.8.4 Decoding

The role of the decoding process in ASR is to find a sequence of words whose corresponding acoustic and language models best match the input feature vector sequence [87]. The decoding process is also called a *search* process in the ASR literature [22], [86], [87]. An efficient way to solve the search problem is to use dynamic programming. Let $\phi_j^{(t)} = \max_\theta P(\mathbf{X}_{1:t}, \theta_t = s_j|\lambda_{HMM})$, i.e., the maximum probability of observing the partial sequence, $\mathbf{X}_{1:t}$, and then being in the state s_j at time t given the model parameters λ_{HMM}. This probability can be efficiently computed using the Viterbi algorithm as follows [22]:

$$\phi_j^{(t)} = \max\{\phi_i^{(t-1)} a_{ij}\} b_j(\mathbf{x}_t). \tag{2.72}$$

It is initialized by setting $\phi_0^{(t)} = 1$ for the initial non-emitting entry state and 0 for all other states. The probability of the most likely word sequence is then given by $\max_j \left(\phi_t^{(j)}\right)$. If every maximization decision from the decoding process is recorded, a traceback to all such paths will yield the required best matching state/word sequence [22]. One can also generate an N best set of hypothesis instead of one best hypothese as from the above eq. (2.72), where N is chosen between 100-1000. This is very much useful since it allows multiple passes over the data without the computational expense of repeatedly solving eq.(2.72) from scratch. A compact and efficient structure for storing these hypotheses is called the word lattice in the ASR literature [22].

2.8.5 Deep Learning for ASR

The GMM-HMM-based ASR has shown impressive results for the LVCSR task in the past. However, to date ASR is not a solved problem in any fundamental sense [91]. The Table 2.1 shows the 'pros' and 'cons' of an HMM for the ASR task. Many of the limitations of the GMM-HMM can be alleviated using the deep neural networks [58], earlier known as the *connectionist approach* [92]. Currently, the state-of-the-art ASR approaches are based on using DNN for the acoustic modeling and HMM for the sequence modeling and decoding. Such an approach is known as the hybrid DNN-HMM approach [81]. In the classical ASR, the likelihood probabilities $P(\mathbf{X}|\mathbf{Y})$ are estimated using the GMM-HMM from the acoustic feature vectors. The DNN can estimate posterior probabilities that are related to the emission probabilities and, hence, can be easily integrated with an HMM-based approach [92]. Hence, instead of the GMM, the DNN provides the emission probabilities. In particular, the DNNs can be trained to produce the posterior probability $P(\theta|\mathbf{X})$, i.e., the posterior probability of the HMM state given the acoustic feature vectors [92]. This is done by setting the DNN output layer as states of the HMM, and then it is converted to the emission probabilities using Bayes' rule [92]. Several researchers have shown that, when neural networks are used in the classification mode, the outputs of neural networks can be interpreted as the estimates of a posterior probabilities of the output classes conditioned on the input [92]. Applying the Bayes' rule to the DNN outputs, we obtain

Table 2.1: Advantages and shortcomings of the classical ASR approach using GMM-HMM. After [1].

Pros	Cons
Mathematically rich framework	Poor discrimination capability
Efficient learning and decoding methods	Requirement of distributional assumptions
Better at sequence learning considering the temporal aspects of the speech signal	Phones and subword are assumed to follow the Markov assumption
Flexible HMM topology for statistical phonetics and syntax	Assumption of uncorrelated acoustic features

$$P(\theta|\mathbf{X}) = \frac{P(\mathbf{X}|\theta)P(\theta)}{P(\mathbf{X})}, \tag{2.73}$$

where $P(\theta|\mathbf{X})$ is the posterior estimated using DNN, $P(\theta)$ is the class prior, namely, the relative frequencies of each class as determined from the class labels that are produced by a forced-alignment in the GMM-HMM training. The equation (2.73) can also be written as:

$$\frac{P(\mathbf{X}|\theta)}{P(\mathbf{X})} = \frac{P(\theta|\mathbf{X})}{P(\theta)}. \tag{2.74}$$

The scaled likelihood $P(\mathbf{X}|\theta)$ on the left hand side can be used as an emission probability for the HMM. We will not worry about $P(\mathbf{X})$, since it is a scaling factor and will not change the classification. An example of the hybrid DNN-HMM approach is shown in Figure 2.31 for a three-layer DNN.

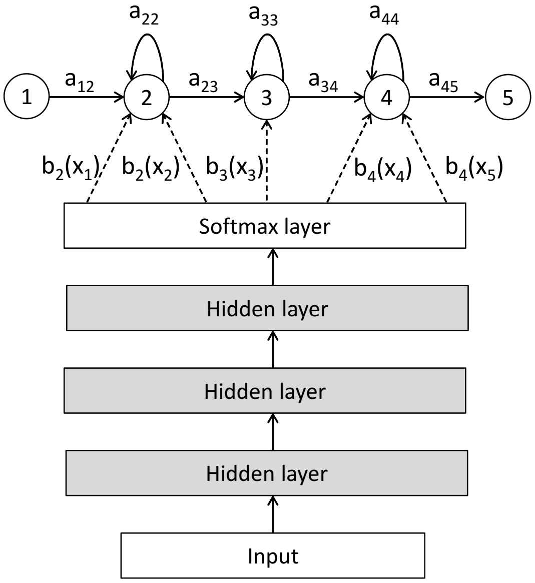


Figure 2.31: An example of hybrid DNN-HMM approach. Adapted and modified from [23].

The hybrid DNN-HMM has two advantages [58], [92]:

1. As mentioned the Table 2.1, the classical ASR system needs strong assumptions about the statistical nature of the input, such as parameterize the input densities as the mixtures of Gaussian densities (i.e., GMM). Furthermore, in order to obtain the emission probability from the GMM, feature vectors need to be uncorrelated (e.g., statistically independent). Such assumptions are not required with the DNN models. With DNN, one can directly use the Mel spectrogram or the STFT-based spectrogram, which yields better results than the standard uncorrelated MFCC feature vectors [81]. One can also effectively use two different types of feature vectors with DNN in a feature-level fusion framework [93].

2. DNN provides a simple mechanism to incorporate the contextual information in the acoustic feature vectors. If we take a context of c frames on the left

and right of the current frame of a feature vector, i.e., $X_{n-c}^{n+c} = \{x_{n-c}, ..., x_n, ..., x_{n+d}\}$, DNN will estimate, $P(\theta | X_{n-c}^{n+c})$.

3. DNNs are trained using discriminative criteria and hence, the posterior probabilities will be optimized to maximize the discrimination between classes of the acoustic sounds, rather than to most closely match the distributions with each acoustic sound class [92].

Earlier it was difficult to train deeper DNN-HMM networks and hence, the concept of pre-training emerged. Pre-training involves training the unsupervised model, such as stacks of RBM (i.e., DBN) or autoencoders using a large amount of unlabeled data [58]. The weights and biases of the pre-trained network are then used to initialize the parameters of the supervised DNN. There have been number of attempts to justify this approach including a remarkable mathematical approach presented in [94]. Recently, there are approaches called as end-to-end deep learning that do not require sequence modeling using HMMs in the ASR task or sequence classification in general. Such models are based on a variant of a RNN along with the connectionist temporal classification (CTC) loss function [95]. In this thesis, we used the hybrid DNN-HMM approach for the ASR task. Next, we will discuss the literature on approaches for the auditory modeling.

2.9 Literature on Auditory Modeling

The representation of a speech and audio signal based on sound perception in humans is of significant interest in developing features for speech and audio processing applications. The classical auditory models were developed during the 1980s to mimic the human auditory processing. However, such physiological models often do not reflect the full complexity of the HAS which, for example, is able to adapt readily to the variability in acoustic conditions. There are many approaches that are based on data-driven learning and/or optimization of parameters of auditory models. Data-driven learning or representation learning can be supervised (i.e., with label information) or unsupervised (where no such class labels are available). We review the literature of auditory models based on computational/mathematical models and machine learning-based approaches.

2.9.1 Computational and Mathematical Models

The Seneff, Ghitza, and Lyon's auditory models have made a huge impact on many recent computational auditory models as reviewed in [39]. These auditory models are based on mathematical modelling of auditory processing or psychophysical and physiological experiments. The auditory filterbank is a principal component of these models. It includes two kinds of models: (1) motivated by reproducing the observed behavior of the mechanical vibrations of the BM or the ANF responses and (2) motivated by psychoacoustic experiments, such as detection of tones in noise maskers [96]. The MFCC and PLP coefficients are the state-of-the-art auditory-based features for speech recognition that use a simplified auditory pipeline. The audio classification application also uses gammatone-based features. Such handcrafted features rely on simplified auditory models [38], [39]. The modulation representation is obtained using another transformation applied on spectrograms or any similar time-frequency representation. Such two-level auditory models are called deep auditory models (DAM) in this thesis. The deep auditory representation is obtained using S. Shamma's auditory model that consists of two layer wavelet transforms with many auditory nonlinearities [49]. Another deep auditory model was proposed by T. Dau that utilize the masking effect in the auditory processing [97]. Using similar approaches by S. Shamma and T. Dau's models, S. Mallat proposed the deep scattering transform (DST) that has nice intriguing mathematical properties [98]. The DST was successfully applied in various speech and audio processing applications [98].

2.9.2 Machine Learning-based Models

Supervised learning approaches for raw speech signals include work in [99–105] which the end-to-end approaches for acoustic modeling in ASR. In the case of audio processing, supervised learning approaches include end-to-end audio classification proposed in [106] and [107]. Most works on unsupervised learning for speech and audio signals are based on cochlear filterbank learning to model auditory processing. The first approach was to use Independent Component Analysis (ICA) as a learning model applied on the small windows of speech and audio signals [12, 108, 109]. To model a Mel-like filterbank, Nonnegative Matrix Factorization (NMF) was applied to the power spectra of speech signals [110]. In [111], nonlinearity associated with the auditory system is optimized using a data-driven method. Based on local geometry of the feature vector-domain and the perceptual auditory-domain, MFCC features were optimized in [112]. The RBM with ReLU was also used to learn features using segments of raw speech signals [113].

The unsupervised learning methods described above are based on processing small segments of speech and audio signals or operating on STFT of speech and audio signals. However, there are many disadvantages of such block-based (or window-based) signal processing as discussed in [114]. In particular, a speech signal has very brief transient sounds as well as quasi-periodic voiced sounds; fixed window segments of speech may smear these sounds. In addition, the representation is very sensitive to the temporal shifts of windows as experimentally proved in [114]. To avoid these problems, sparse spike coding is used to learn filterbanks from speech and audio signals [114], [115]. However, spike coding does not include any *nonlinearity* in the model and optimization is performed on the linear superposition of kernel functions [114]. To obtain a MFCC-like representation using time-domain formulation, the scattering transform was proposed [98]. Scattering wavelets on Mel scale are convolved with speech/audio signals and averaged later using a lowpass filter (that has a similar effect as that of the Mel scale averaging). However, deep scattering wavelets do not involve learning of subband filters [98]. A summary of the ASR literature is given in Table 2.2. A summary of the ESC, SSD and ICC literature is given in Table 2.3.

Table 2.2: Selected chronological progress using the representation learning for the ASR task

Models	**Input**	**Databases**	**Source**
LDA-RASTA	PLP	OGI-multilingual	S. Vuuren et al. 1997 [116]
Discriminative filters	Bell Lab feats	AURORA 2	B. Mak et al. 2003 [117]
LDA	MFCC	NUM-100	J. W. Hung et al. 2006 [118]
ConvRBM	STFT	TIMIT	H. Lee et al. 2009 [50]
Nonlinearity learning	STFT	TIMIT, TIDIGITS	S. Chatterjee et al. 2011 [119]
NMF	STFT	TIMIT	A. Bertrand et al. 2008 [110]
RBM	Speech	TIMIT	N. Jaitly and G. E. Hinton 2011 [113]
Autoencoder	STFT	TIMIT	N. Jaitly and G. E. Hinton 2013 [120]
DNN	Speech	Quaero	Z. Tuske et al. 2014 [105]
Gabor-CNN	FBANK	WSJ, AURORA 4	S. Chang et al. 2014 [121]
CNN	Speech	Google ASR	T. Sainath et al. 2015 [102]
CNN	Speech	Quaero	P. Golik et al. 2015 [103]
CNN	Speech	TIMIT, WSJ	D. Palaz et al. 2015 [104]
CNN	Speech	Switchboard, WSJ	P. Ghahremani et al. 2016 [122]
ConvRBM*	FBANK	TIMIT, WSJ	H. Sailor and H. Patil 2016 [10]
ConvRBM*	Speech	TIMIT, WSJ	H. Sailor and H. Patil 2016 [2]
ConvRBM*	Speech	TIMIT, WSJ, AURORA 4	H. Sailor and H. Patil 2016 [3]
Stacked ConvRBMs*	Speech	TIMIT, AURORA 4	H. Sailor and H. Patil 2016 [11]
ConvRBM and TEO*	Speech	AURORA 4	H. Sailor and H. Patil 2017 [4]
DNN	AMFB	CHiME2, REVERB, Librispeech	N. Moritz et al. 2016 [123]
DNN	STFT	ASJ, JNAS	H. Seki et al. 2017 [124]
Sparse representations	STFT, Speech	TIMIT, WSJ, AURORA 4	P. Sharma et al. 2017 [125]
ConvRBM	STFT	AURORA 4, REVERB	P. Agrawal and S. Ganapathy 2017 [126]
ConvRBM*	Speech	Gujarati Agri-ASR	H. Sailor and H. Patil 2017 [30]

* indicates proposed model in this thesis.

To alleviate the problems discussed above, this thesis proposes an unsupervised filterbank learning model which is shown to perform better than the MFCC and Mel filterbank features for the speech recognition task [2]. Later the model is applied on various audio classification tasks, such as Environmental Sound Clas-

Table 2.3: Selected chronological progress using representation learning for speech and audio processing applications

Models	Input	Domain	Databases	Source
Spherical k-means	FBANK	ESC	Urbansound8k	J. Salamon et al. 2015 [127]
Spherical k-means	DSS	ESC	Urbansound8k	J. Salamon et al. 2015 [128]
CNN	Audio	ESC	ESC-50	Y. Tokozume et al. 2017 [106]
CNN	Audio, video	ESC	Soundnet	Y. Aytar et al. 2016 [107]
ConvRBM*	Audio	ESC	ESC-50	H. Sailor, et al. 2017 [5]
DNN	FBANK	SSD	ASVSpoof 2015	N. Chen et al. 2015 [129]
DNN, RNN	FBANK	SSD	ASVSpoof 2015	Y. Qian et al. 2016 [130]
DNN bottleneck	FBANK	SSD	ASVSpoof 2015	M. Alam et al. 2016 [131]
CLDNN	Speech	SSD	BTAS 2016	H. Dinkel et al. 2017 [132]
ConvRBM*	Speech	SSD	ASVSpoof 2015	H. Sailor, et al. 2017 [6]
LCNN	STFT	SSD	ASVSpoof 2017	G. Lavrentyva et al. 2017 [133]
ConvRBM*	Speech	SSD	ASVSpoof 2017	H. Sailor, et al. 2017 [45]
DNN	Speech	SSD	ASVSpoof 2015	H. Yu et al. 2017 [134]
DNN, RNN	FBANK, CQCC	SSD	ASVSpoof 2015	Y. Qian et al. 2017 [135]
GMM	MFCC	ICC	Private dataset	H. Alaie et al. 2016 [136]
ConvRBM*	Cry signal	ICC	DA-IICT, Chillanto	H. Sailor, et al. 2017 [46]

* indicates proposed model in this thesis.

sification (ESC) [5], spoof speech detection (SSD) [6], and infant cry classification (ICC) tasks [46]. The novelty of the proposed model lies in learning directly from the speech and audio signals of any *arbitrary* lengths in order to alleviate artifacts of windowing. In addition, it includes nonlinearity in learning and the model is stochastic in nature. Our proposed model is based on a Convolutional Restricted Boltzmann Machine (ConvRBM), which was first proposed in [59] to improve the scalability of RBM. Earlier ConvRBM was applied on spectrograms of speech signals to model the TRFs in the auditory cortex [50]. Our work is highly motivated by success of RBM-based approaches in [50] and [232] for auditory representation learning. In [10], we have introduced noisy rectified linear units (NReLU) in ConvRBM to learn the TRFs. We developed ConvRBM to model auditory processing in the human ear using raw speech signals [2], [3]. We have used ReLU to increase the sparsity, and inference is based on NReLU. Compared to recent approaches for filterbank learning in convolutional networks [102–104], our model is unsupervised and probabilistic in nature. The two ConvRBMs are stacked together via layerwise training which we refer to as the unsupervised deep auditory model (UDAM), which is successfully applied for the ASR task [11]. Unsupervised filterbank learning along with the Teager Energy Operator (TEO) is applied for noise-robust speech recognition in [4].

2.10 Chapter Summary

In this Chapter, we presented the basic background for auditory signal processing and representation learning. In addition, the fundamentals of speech recognition were presently briefly. The discussion on Hopfield networks and Boltzmann machines makes a starting point to understand our proposed Convolutional RBM. The literature on auditory modeling is presented using both computational/mathematical methods and machine learning-based methods. In the next Chapter, we will present our proposed ConvRBM model in detail.

CHAPTER 3

Auditory Filterbank Learning

3.1 Introduction

In this chapter, we introduce our proposed model of unsupervised auditory filterbank learning using speech signals. The model is based on a Convolutional Restricted Boltzmann Machine, abbreviated as ConvRBM. The architecture of the proposed model is discussed in Section 3.2. Section 3.3 presents the training criterion of ConvRBM and an algorithm to update the model parameters in detail. After the model is trained, the stages of feature extraction are presented in Section 3.4. An analysis of the learned filterbanks and its comparison with the standard auditory frequency scales is discussed in Section 3.5. We have shown that the learned subband filters resemble the gammatone-like auditory filters. An application to the ASR task is presented in Section 3.6-3.7 for various databases.

3.2 Proposed Model for Filterbank Learning

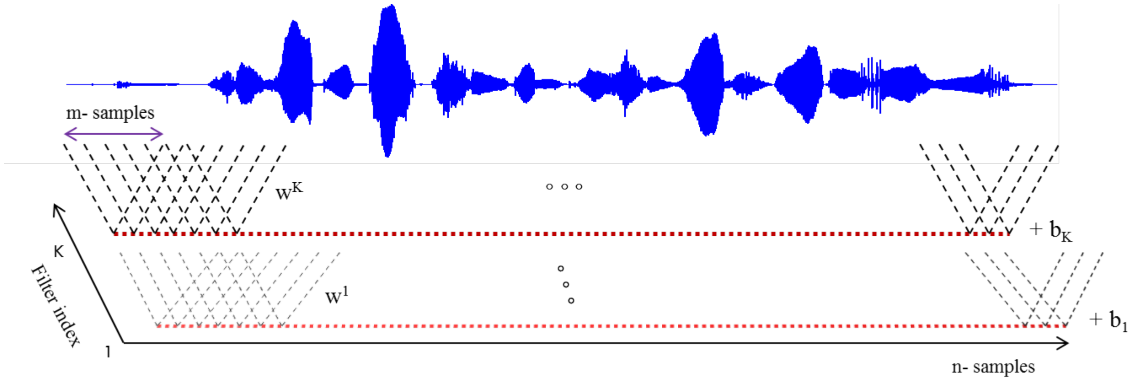


Figure 3.1: The arrangement of the hidden units in K groups, and the corresponding weight connections. The filter index-axis is perpendicular to the plane of this paper. Each hidden unit (red dots) in the k^{th} group is wired such that it results in a valid convolution between the speech signal and weights $\mathbf{W}^k$. After [3].

ConvRBM has two layers, namely, a visible layer and a hidden layer [2,3,50, 59]. The input to the visible layer (denoted as $\mathbf{x}$) is an entire speech signal of length n samples. The hidden layer (denoted as $\mathbf{h}$) consists of K-groups (i.e., number of subband filters) with filter length of m samples in each. Weights (also called as filters or in fact, subband filters w.r.t. the speech perception mechanism in human hearing [137]) are shared between the visible and hidden units amongst all the locations in each group [59]. Weight sharing reduces the number of parameters compared to the fully connected RBM, and helps the model to learn time-frequency structures in the speech signals as discussed later in this Section. Denoting b_k as the hidden bias for the k^{th} group, the response of the convolution layer for the k^{th} group is given as [3]:

$$\mathbf{I}_k = (\mathbf{x} * \tilde{\mathbf{W}}^k) + b_k, \tag{3.1}$$

For ConvRBM with the visible units $\mathbf{x}$ and hidden units $\mathbf{h}$, the energy function of the model is given as [2], [3]:

$$\begin{split} E(\mathbf{x},\mathbf{h}) = \frac{1}{2\sigma_x^2}\sum_{i=1}^{n} x_i^2 - \frac{1}{\sigma_x}\sum_{k=1}^{K}\sum_{j=1}^{l}\sum_{r=1}^{m}\left(h_j^k w_r^k x_{j+r-1}\right) \\ - \sum_{k=1}^{K} b_k \sum_{j=1}^{l} h_j^k - \frac{1}{\sigma_x^2} c \sum_{i=1}^{n} x_i, \end{split} \tag{3.2}$$

where c is a visible bias, which is also shared. We have used 'valid' length convolution (as discussed in Section 2.5.1, Chapter 2) and hence, the length of each group is $l = n - m + 1$. Each speech signal is normalized to the zero-mean and unit variance. Hence, variance (σ_x^2) in eq. (3.2) is set to 1 as suggested in [60]. The joint distribution function of the visible and hidden units is

$$p(\mathbf{x},\mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{x},\mathbf{h})}, \tag{3.3}$$

where Z is the partition function, $Z = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})} \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}$, which normalizes the energy, and thereby making it a probability distribution function (PDF). In case of sigmoid hidden units, the stochastic sampling equations of visible and hidden units are given as [50]:

$$\begin{gathered} \mathbf{h}^k \sim \text{sigmoid}((\mathbf{x} * \tilde{\mathbf{W}}^k) + b_k), \\ \mathbf{x}_{recon} \sim \mathcal{N}\left(\sum_{k=1}^{K}(\mathbf{h}^k * \mathbf{W}^k) + c, 1\right), \end{gathered} \tag{3.4}$$

where $\mathcal{N}(\mu, 1)$ is a Gaussian distribution with mean μ and variance 1. Generalization of binary hidden units is achieved by replacing each binary unit with infinite copies of binary units that all have the same weights and progressively more negative bias [138]. If the offsets of the sigmoid units are -0.5,-1.5,..., we obtain a set of sigmoid units referred to as the Stepped Sigmoid Units (SSU). The sum of probabilities of the copies of a single unit is extremely close to a form given by:

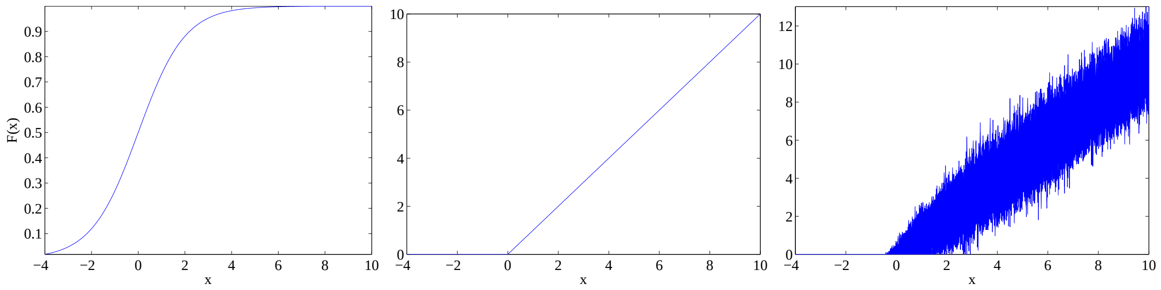
$$\lim_{N \to \infty} \sum_{i=1}^{N} \text{sigmoid}(\mathbf{I}_k - i + 0.5) = \log(1 + e^{\mathbf{I}_k}). \tag{3.5}$$

Hence, the total activities of the copies of a single unit act like a smoothed rectified linear unit known as the *softplus* function. The drawback of such an approach is that the sigmoid function needs to be used many times to get the probabilities required for sampling an integer value correctly. A fast approximation is possible where the sampled value of the ReLU is not constrained to be an integer. It is obtained by addition of a Gaussian noise whose variance is controlled by the sigmoid of the input. Since it is a stochastic version of the ReLU, it is known as Noisy ReLU (NReLU). With NReLU, following are the stochastic sampling equations for the hidden and visible units (use the reconstructed speech signal, $\mathbf{x}_{recon}$ to further update hidden units) [2], [3]:

$$\begin{aligned} \mathbf{h}^k &\sim \max(0, \mathbf{I}_k + \mathcal{N}(0, \text{sigmoid}(\mathbf{I}_k))), \\ \mathbf{x}_{recon} &\sim \mathcal{N}\left(\sum_{k=1}^{K} (\mathbf{h}^k * \mathbf{W}^k) + c, 1\right), \end{aligned} \tag{3.6}$$

where $\mathcal{N}(0, \text{sigmoid}(\mathbf{I}_k))$ is a Gaussian noise with mean zero and sigmoid of $\mathbf{I}_k$ as its variance. While calculating the relationship between the hidden and visible units, a deterministic ReLU (i.e., $max(0, \mathbf{I}_k)$) is used as an activation function of the hidden units (as shown in Algorithm 1 in Section 3.3). The example of various activation functions is illustrated in Figure 3.2 for activation function $F(x)$ applied on the input $x \in \mathbb{R}$. The NReLU activation function is plotted for 100 realizations of the Gaussian noise added to the input signal as shown in Figure 3.2 (c). The main difference between the ReLU and NReLU is the increased dynamic range of NReLU due to the addition of the Gaussian noise, which in turn leads to the stochastic nature of the hidden units.

With a convolution layer and ReLU nonlinearity, an example of processing stages is shown in Figure 3.3. The convolution with the subband filters (i.e., weights of the proposed model) decomposes the speech signal into different sub-



(a) (b) (c)

Figure 3.2: Example of activation functions for (a) Sigmoid, (b) ReLU, and (c) NReLU.

bands. We will see in Section 3.5 that such decomposition of the speech signal is due to using different subband filters that are localized in the frequency-domain. Learning of such subband filters is possible because of the weight sharing in ConvRBM and temporal information in the speech signals (see in Appendix A for detailed discussion). ReLU reduces the information by making the negative values to zero that leads to the *sparsity* in the hidden units.

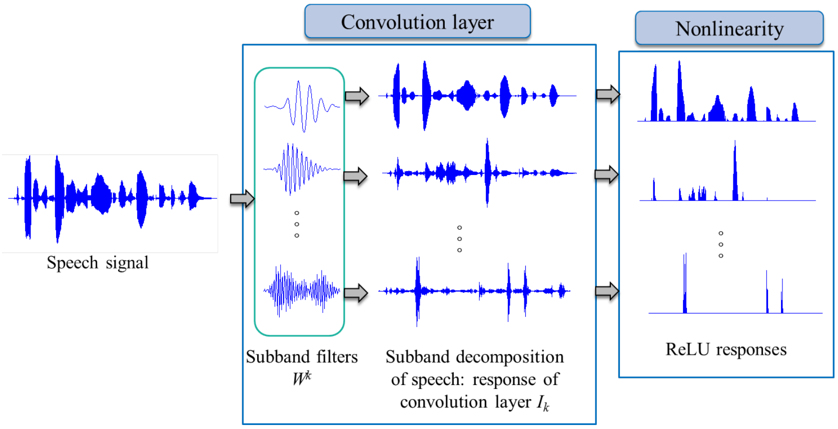


Figure 3.3: The example of decomposition of a speech signal using the weights of ConvRBM with a convolution layer followed by ReLU nonlinearity. After [3].

3.3 Model Learning

Unsupervised learning of a probabilistic model means adjusting the model parameters θ so as to maximize the likelihood of training data [52]. Let us assume that we have a set of D training examples (i.e., $\mathbf{X} = \{\mathbf{x}_d | d \in [1, 2, ..., D]$ sampled from some underlying function $f(\mathbf{x})$). Assuming that all the training examples are *i.i.d* (independent and identically distributed) sampled from the data distribution $p(\mathbf{x})$, we can write likelihood for the observed data as a multiplication of the probability densities of each training example [52]:

$$p(\mathbf{X};\theta) = \prod_{d=1}^{D} p(\mathbf{x}_d;\theta). \tag{3.7}$$

Generally, it is convenient to use log-likelihood and hence, products in eq. (3.7) can be written as sum of logarithms as follows [52]:

$$\begin{aligned} \ell(\mathbf{X};\theta) &= \log \prod_{d=1}^{D} p(\mathbf{x}_d;\theta), \qquad (3.8)\\ &= \sum_{d=1}^{D} \log p(\mathbf{x}_d;\theta). \qquad (3.9) \end{aligned}$$

Since logarithm is a monotonic function, maximization of likelihood will be intact in log-likelihood domain [14]. The details of ML optimization are given in Appendix F, Section F.1. For a single training example with the visible and hidden units $[\mathbf{x}, \mathbf{h}]$, the log-likelihood of the MRFs is given as [52]:

$$\begin{aligned} \ell(\mathbf{x};\theta) &= \log p(\mathbf{x};\theta), \qquad (3.10)\\ &= \log \int_{-\infty}^{\infty} p(\mathbf{x}, \mathbf{h};\theta) \mathrm{d}\mathbf{h}. \qquad (3.11) \end{aligned}$$

Using eq. (3.3), the log-likelihood can be written in terms of the energy function $E(\mathbf{x}, \mathbf{h})$ as follows:

$$\begin{aligned} \ell(\mathbf{x};\theta) &= \log \frac{1}{Z} \int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})} \mathrm{d}\mathbf{h}, \qquad (3.12)\\ &= \log \int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})} \mathrm{d}\mathbf{h} - \log Z, \qquad (3.13)\\ \ell(\mathbf{x};\theta) &= \log \int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})} \mathrm{d}\mathbf{h} - \log \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})} \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}, \qquad (3.14) \end{aligned}$$

where the partition function is given as $Z = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})} \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}$. The gradient of log-likelihood is calculated as the derivative of the log-likelihood function w.r.t. the model parameters, $\theta := \left(\mathbf{W}^k, b_k, c\right)$ [3], [52]:

$$
\begin{aligned}
\frac{\partial}{\partial\theta}\ell(\mathbf{x};\theta) &= \frac{\partial}{\partial\theta}\left(\log\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\mathrm{d}\mathbf{h}\right) - \frac{\partial}{\partial\theta}\left(\log\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}\right), \\
&= \frac{1}{\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\mathrm{d}\mathbf{h}}\int_{-\infty}^{\infty}\frac{\partial}{\partial\theta}\left(e^{-E(\mathbf{x},\mathbf{h})}\right)\mathrm{d}\mathbf{h} \\
&+ \frac{1}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{\partial}{\partial\theta}\left(e^{-E(\mathbf{x},\mathbf{h})}\right)\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}, \\
&= -\frac{1}{\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\mathrm{d}\mathbf{h}}\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\frac{\partial}{\partial\theta}E(\mathbf{x},\mathbf{h})\mathrm{d}\mathbf{h} \\
&+ \frac{1}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\frac{\partial}{\partial\theta}E(\mathbf{x},\mathbf{h})\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h}, \\
&= -\int_{-\infty}^{\infty} p(\mathbf{h}|\mathbf{x})\frac{\partial}{\partial\theta}E(\mathbf{x},\mathbf{h})\mathrm{d}\mathbf{h} + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(\mathbf{x},\mathbf{h})\frac{\partial}{\partial\theta}E(\mathbf{x},\mathbf{h})\mathrm{d}\mathbf{x}\mathrm{d}\mathbf{h},
\end{aligned}
\tag{3.15}
$$

where $p(\mathbf{h}|\mathbf{x})$ can be derived using the product rule of probability [14]:

$$
p(\mathbf{h}|\mathbf{x}) = \frac{p(\mathbf{x},\mathbf{h})}{p(\mathbf{x})} = \frac{e^{-E(\mathbf{x},\mathbf{h})}}{\int_{-\infty}^{\infty} e^{-E(\mathbf{x},\mathbf{h})}\mathrm{d}\mathbf{h}}. \tag{3.16}
$$

With the notations used in [52], we can write the log-likelihood in terms of expectations as [3]:

$$
\begin{aligned}
\frac{\partial}{\partial\theta}\ell(\mathbf{x};\theta) &= -\mathbb{E}_{p(\mathbf{h}|\mathbf{x})}\left[\frac{\partial}{\partial\theta}E(\mathbf{x},\mathbf{h})\right] + \mathbb{E}_{p(\mathbf{h},\mathbf{x})}\left[\frac{\partial}{\partial\theta}E(\mathbf{x},\mathbf{h})\right], \\
&\approx -\left\langle\frac{\partial}{\partial\theta}E(\mathbf{x},\mathbf{h})\right\rangle_{data} + \left\langle\frac{\partial}{\partial\theta}E(\mathbf{x},\mathbf{h})\right\rangle_{model},
\end{aligned}
\tag{3.17}
$$

where $\langle\cdot\rangle$ is the sample mean under distribution used to calculate expectations. Here, $\langle\cdot\rangle_{data}$ is the sample mean estimated, when the visible units are clamped to the speech signal (i.e., input data) and $\langle\cdot\rangle_{model}$ is the sample mean estimated when the visible and hidden units are sampled from a model distribution. The first part of eq. (3.17), for $\mathbf{W}^k$ as a model parameter, can be computed by taking a derivative of eq. (3.2) w.r.t. $\mathbf{W}^k$. The gradient for weights in each k^{th} group is given as [3]:

$$
\frac{\partial}{\partial w_r^k}E(\mathbf{x},\mathbf{h}) = -\frac{\partial}{\partial w_r^k}\left[\sum_{k=1}^{K}\sum_{j=1}^{l}\sum_{r=1}^{m}\left(h_j^k w_r^k x_{j+r-1}\right)\right]. \tag{3.18}
$$

For $r = 1$ to m, eq. (3.18) can be written as a set of equations as follows [3]:

$$\begin{aligned} \frac{\partial}{\partial w_1^k} E(\mathbf{x}, \mathbf{h}) &= \sum_{j=1}^{l} \left(h_j^k x_j \right), \\ &\vdots \\ \frac{\partial}{\partial w_m^k} E(\mathbf{x}, \mathbf{h}) &= \sum_{j=1}^{l} \left(h_j^k x_{j+m-1} \right). \end{aligned} \tag{3.19}$$

Since $\mathbf{W}^k = [w_1^k, w_2^k, ..., w_m^k]$ is a weight vector, we can write this as a gradient of weight vector, $\mathbf{W}^k$ [3]:

$$\therefore \left[\frac{\partial}{\partial w_1^k} E(\mathbf{x}, \mathbf{h}), \ldots, \frac{\partial}{\partial w_m^k} E(\mathbf{x}, \mathbf{h}) \right] = \frac{\partial}{\partial \mathbf{W}^k} E(\mathbf{x}, \mathbf{h}), \tag{3.20}$$

$$= - \sum_{j=1}^{l} h_j^k x_{j+r-1}, \tag{3.21}$$

$$= -conv(\mathbf{x}, \tilde{\mathbf{h}^k}). \tag{3.22}$$

where $\tilde{\mathbf{h}^k}$ is a *flipped* array to represent the linear convolution operation denoted as $conv(\cdot)$. The length of this valid convolution between the input of length n samples and the k^{th} hidden group of length, $l = n - m + 1$ (obtained from the convolution of input and weights), is $n - l + 1 = m$ samples. This term is easy to calculate. We clamp the visible units to the speech signal, $\mathbf{x}$, and find hidden unit activations. The hidden unit activations can be found by passing convolution responses, $\mathbf{I}_k$, from the deterministic ReLU nonlinearity. Then, the relationship between the hidden and visible units can be found using eq. (3.22). This is called the *positive phase* of CD learning [24].

The second term in eq. (3.17) requires samples from a model distribution, which is very difficult to obtain. Since we have conditional probabilities of both visible and hidden units, blocked Gibbs sampling is used to obtain samples from the model [51]. The Gibbs sampling is a type of Markov Chain Monte Carlo (MCMC) technique to draw samples from the distribution and to approximate the expectation operator in the sample average form [14], [51]. Ideally, we should run the chain to infinity to obtain the samples from the model distribution at equilibrium state [139]. We clamp the visible units to the input data, update the hidden units, and reconstruct back the visible units, and repeat this procedure infinite times as shown in Figure 3.4. Infinite steps in Gibbs sampling can be well approximated in finite time using a technique called the *Contrastive Divergence* (CD) [24].

Instead of sampling infinite times, we can sample only up to N times called as (CD-N) or it is shown in [24] that even a single step gives a good approximation called as *CD-1*. We have used a single step CD learning as shown in Figure 3.5. Updating hidden units using reconstructed speech signal is called the negative phase of CD learning [24]. The second term in eq. (3.17) can be written as:

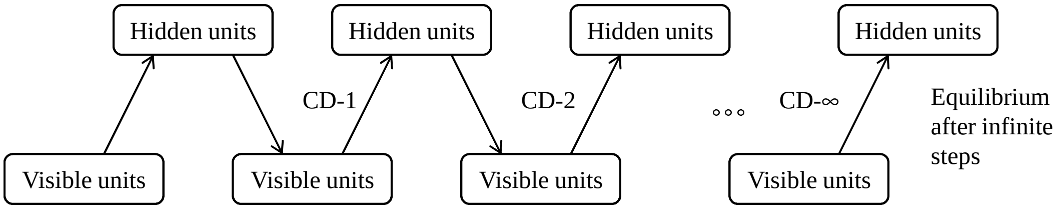


Figure 3.4: Gibbs sampling in ConvRBM. After [24], [25].

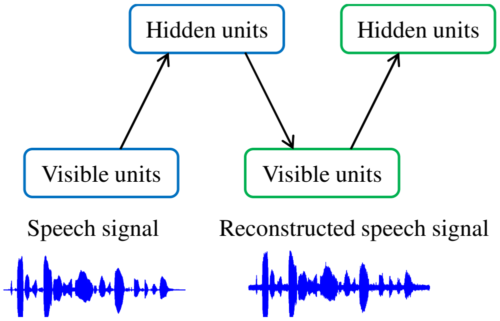


Figure 3.5: Demonstration of a CD-1 learning. After [3].

$$\frac{\partial}{\partial \mathbf{W}^k} E(\underline{\mathbf{x}}, \underline{\mathbf{h}}) = -conv(\underline{\mathbf{x}}, \underline{\tilde{\mathbf{h}^k}}), \tag{3.23}$$

where the underline symbol denotes visible ($\underline{\mathbf{x}} = \mathbf{x}_{recon}$), and the hidden states ($\underline{\tilde{\mathbf{h}^k}}$) in the CD-1 stage (negative phase). We obtain samples of the visible and hidden units using eq. (3.6). For the weights of model, eq. (3.17) can now be written as:

$$\begin{aligned} \frac{\partial}{\partial \mathbf{W}^k} \ell(\mathbf{x}; \theta) &= \mathbb{E}_{p(\mathbf{h}|\mathbf{x})}\left[conv(\mathbf{x}, \tilde{\mathbf{h}^k})\right] - \mathbb{E}_{p(\mathbf{h},\mathbf{x})}\left[conv(\underline{\mathbf{x}}, \underline{\tilde{\mathbf{h}^k}})\right], \\ &\approx \left\langle conv(\mathbf{x}, \tilde{\mathbf{h}^k})\right\rangle_{data} - \left\langle conv(\underline{\mathbf{x}}, \underline{\tilde{\mathbf{h}^k}})\right\rangle_{model}. \end{aligned} \tag{3.24}$$

The corresponding gradient update for weights is now written as [3]:

$$\nabla \mathbf{W}^k = \epsilon \left(\left\langle conv(\mathbf{x}, \tilde{\mathbf{h}^k})\right\rangle_{data} - \left\langle conv(\underline{\mathbf{x}}, \underline{\tilde{\mathbf{h}^k}})\right\rangle_{model} \right), \tag{3.25}$$

where ϵ is a learning rate parameter. For the hidden biases, b_k, we can write the gradient equation as:

$$\begin{aligned}\frac{\partial}{\partial b_k}\ell(\mathbf{x};\theta) &= -\mathbb{E}_{p(\mathbf{h}|\mathbf{x})}\left[\frac{\partial}{\partial b_k}E(\mathbf{x},\mathbf{h})\right] + \mathbb{E}_{p(\mathbf{h},\mathbf{x})}\left[\frac{\partial}{\partial b_k}E(\mathbf{x},\mathbf{h})\right],\\ &= \mathbb{E}_{p(\mathbf{h}|\mathbf{x})}\left[\sum_{j=1}^{l} h_j^k\right] - \mathbb{E}_{p(\mathbf{h},\mathbf{x})}\left[\sum_{j=1}^{l} \underline{\tilde{h}_j^k}\right],\\ &\approx \left\langle \sum_{j=1}^{l} h_j^k \right\rangle_{data} - \left\langle \sum_{j=1}^{l} \underline{\tilde{h}_j^k} \right\rangle_{model}.\end{aligned} \tag{3.26}$$

For visible bias c, we can write the gradient equation as:

$$\begin{aligned}\frac{\partial}{\partial c}\ell(\mathbf{x};\theta) &= -\mathbb{E}_{p(\mathbf{h}|\mathbf{x})}\left[\frac{\partial}{\partial c}E(\mathbf{x},\mathbf{h})\right] + \mathbb{E}_{p(\mathbf{h},\mathbf{x})}\left[\frac{\partial}{\partial c}E(\mathbf{x},\mathbf{h})\right],\\ &= \mathbb{E}_{p(\mathbf{h}|\mathbf{x})}\left[\sum_{i=1}^{n} x_i\right] - \mathbb{E}_{p(\mathbf{h},\mathbf{x})}\left[\sum_{i=1}^{n} \underline{x_i}\right],\\ &\approx \left\langle \sum_{i=1}^{n} x_i \right\rangle_{data} - \left\langle \sum_{i=1}^{n} \underline{x_i} \right\rangle_{model}.\end{aligned} \tag{3.27}$$

The gradient update equations for the hidden and the visible biases are given as [3]:

$$\begin{aligned}\nabla b_k &= \epsilon\left(\left\langle \sum_{j=1}^{l} h_j^k \right\rangle_{data} - \left\langle \sum_{j=1}^{l} \underline{\tilde{h}_j^k} \right\rangle_{model}\right),\\ \nabla c &= \epsilon\left(\left\langle \sum_{i=1}^{n} x_i \right\rangle_{data} - \left\langle \sum_{i=1}^{n} \underline{x_i} \right\rangle_{model}\right).\end{aligned} \tag{3.28}$$

The iterative updates for model parameters $\theta := \left(\mathbf{W}^k, b_k, c\right)$ are given as [52]:

$$\theta^{(t+1)} = \theta^{(t)} + \nabla\theta^{(t)} + \eta\theta^{(t-1)}, \tag{3.29}$$

where the momentum term with the parameter η helps against the *oscillatory* behavior in the parameter space, and accelerates the learning process [14], [52], [70]. In addition, we include the weight decay regularization term in the likelihood function in eq. (3.11) to combat overfitting in the ConvRBM training. The weight decay regularization is discussed in Appendix F. The steps for the model learning using CD-1 are described in Algorithm 1 [3].

Algorithm 1 The proposed algorithm for ConvRBM training applied on speech signals. After [3].

```
Input: Speech signals, $\mathbf{x}$, with arbitrary length of $n$ samples.
Output: Weights, $\mathbf{W}$, hidden biases $b$, and visible bias $c$.
for each training iteration $t$ do
    Use weights and biases updated during the last iteration $t - 1$
    for each training example $\mathbf{x}$ do
        for each $k^{th}$ group do
            Convolution response, $\mathbf{I}_k = (\mathbf{x} * \tilde{\mathbf{W}}^k) + b_k$
            $\mathbf{h}^k_{act} = \max(0, \mathbf{I}_k)$
            $\mathbf{h}^k_{sample} \sim \max(0, \mathbf{I}_k + \mathcal{N}(0, \text{sigmoid}(\mathbf{I}_k)))$
            $VH = \text{conv}(\mathbf{x}, \mathbf{h}^k_{act})$
            $H_{\sum} = \sum(\mathbf{h}^k_{act})$
        end for
        Sample the visible units (reconstruct speech signal) from the hidden units as:
        $\mathbf{x}_{recon} \sim \mathcal{N}\left(\sum_k(\mathbf{h}^k_{sample} * \mathbf{W}^k) + c, 1\right)$
        for each $k^{th}$ group do
            Convolution response, $\underline{\mathbf{I}}_k = (\mathbf{x}_{recon} * \tilde{\mathbf{W}}^k) + b_k$
            $\underline{\mathbf{h}}^k_{act} = \max(0, \underline{\mathbf{I}}_k)$
            $\underline{\mathbf{h}}^k_{sample} \sim \max(0, \underline{\mathbf{I}}_k + \mathcal{N}(0, \text{sigmoid}(\underline{\mathbf{I}}_k)))$
            $\underline{VH} = \text{conv}(\mathbf{x}_{recon}, \underline{\mathbf{h}}^k_{act})$
            $\underline{H_{\sum}} = \sum(\underline{\mathbf{h}}^k_{act})$
        end for
        $\nabla \mathbf{W}^{(t)} = [VH - \underline{VH}]\,/n$
        $\nabla \mathbf{b}^{(t)} = \left[H_{\sum} - \underline{H_{\sum}}\right]\,/n$
        $\nabla c^{(t)} = [\sum(\mathbf{x}) - \sum(\mathbf{x}_{recon})]\,/n$
        $\mathbf{W}^{(t+1)} \leftarrow \mathbf{W}^{(t)} + \epsilon \nabla \mathbf{W}^{(t)} + \eta \mathbf{W}^{(t-1)}$
        $\mathbf{b}^{(t+1)} \leftarrow \mathbf{b}^{(t)} + \epsilon \nabla \mathbf{b}^{(t)} + \eta \mathbf{b}^{(t-1)}$
        $c^{(t+1)} \leftarrow c^{(t)} + \epsilon \nabla c^{(t)} + \eta c^{(t-1)}$
    end for
end for
```

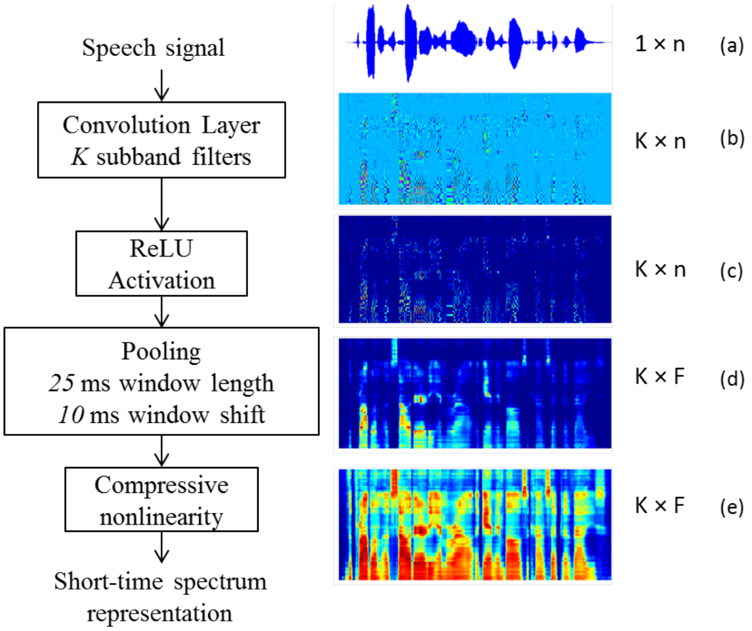


Figure 3.6: Block diagram of stages in the feature representation using ConvRBM: (a) speech signal, (b) and (c) responses from convolution layer, and ReLU, respectively, (d) pooling, and (e) logarithmic compression. After [2], [3].

3.4 Feature Extraction

After ConvRBM is trained, pooling is applied to reduce the representation of ConvRBM responses in the temporal-domain. Our proposed model is different from the one used in [50], where a probabilistic max-pooling was used in the inference stage itself for the binary hidden units. Our approach resembles the method used in [140], where the time-domain gammatone responses were reduced using average-based framing, which is a pooling-like operation. Such an approach is also used in [98], where after convolution with the scattering wavelets, averaging is performed using a lowpass filtering. Here, pooling in the time-domain is equivalent to short-time averaging in the spectral features, such as MFCC, and lowpass filtering in the scattering wavelets. For a speech signal with the sampling frequency, Fs = 16 kHz, pooling is applied using a 25 ms (i.e., 400 samples) window length (wl) and 10 ms (i.e., 160 samples) shift (ws). We used this setup to compare the MFCC extracted using the same windowing parameters. Pooling is performed across time and separately for each subband filter. The speech signal with n samples has $F = \frac{n-wl+ws}{ws}$ number of frames. We have experimented with both the average and max-pooling and found better experimental results with the average pooling. After the pooling, a stabilized logarithm $\log(\cdot + \delta)$ (with $\delta = 0.0001$) is applied as a compressive nonlinearity similar to [141].

The block diagram for the feature extraction procedure (described above) is shown in Figure 3.6. To obtain the same length as the speech signal, 'same' length convolution is used. During the feature extraction stage, we have used deterministic ReLU nonlinearity $\max(0, \mathbf{I}_k)$ as an activation function of the hidden units. The pooling operation reduces the temporal resolution from $K \times n$ samples to $K \times F$ frames. The logarithmic nonlinearity compresses the dynamic range of features, which was found to improve the performance in ASR tasks [141]. The feature extraction steps involved in this ordering resemble the early auditory processing [39], [62] (discussed in the Figure 2.16 in Section 2.6, Chapter 2). In the next Section, the analysis of the ConvRBM filterbank is presented.

3.5 Analysis of ConvRBM

3.5.1 Analysis of Learned Subband Filters

For the analysis of the subband filters, we computed the center frequencies (CFs) of the subband filters as described in [105]. We have analyzed the model with K=60 subband filters (i.e., 60 groups in the hidden layer). Examples of subband filters learned using ConvRBM on the TIMIT, WSJ1 and AURORA 4 databases are shown in Figure 3.7. Filters were arranged according to their increasing order of CFs. Weights of ConvRBM were initialized randomly, and there is no constraint on the filter shapes; still the model was able to learn meaningful representation from the speech signals. Weights of the model called impulse responses of subband filters in the time-domain are shown in Figure 3.7 (a)-(c). We can see that for all three databases, many subband filters are very similar to the auditory gammatone filters and physiological auditory filters (i.e., primarily motivated by the studies reported in [12], [115]). Unlike the filters derived using RBM [113], our learned subband filters resemble more closely the auditory subband filters for the speech signals [12]. This may be due to the fact that RBM was trained on randomly selected smaller windows of the speech signal and hence, they were in any random temporal phase [113]. We have trained our model on speech signals in the time-domain without windowing to learn subband filters, and all subband responses are pooled later to get the short-time spectral representation of the speech signal. Figure 3.7 (d)-(f) shows the frequency-domain representation of corresponding time-domain impulse responses. We can see that all the subband filters are localized in the frequency-domain with the different CFs. Filters with lower CFs are highly localized in the frequency-domain, while those with higher CFs are broader in terms of their bandwidth.

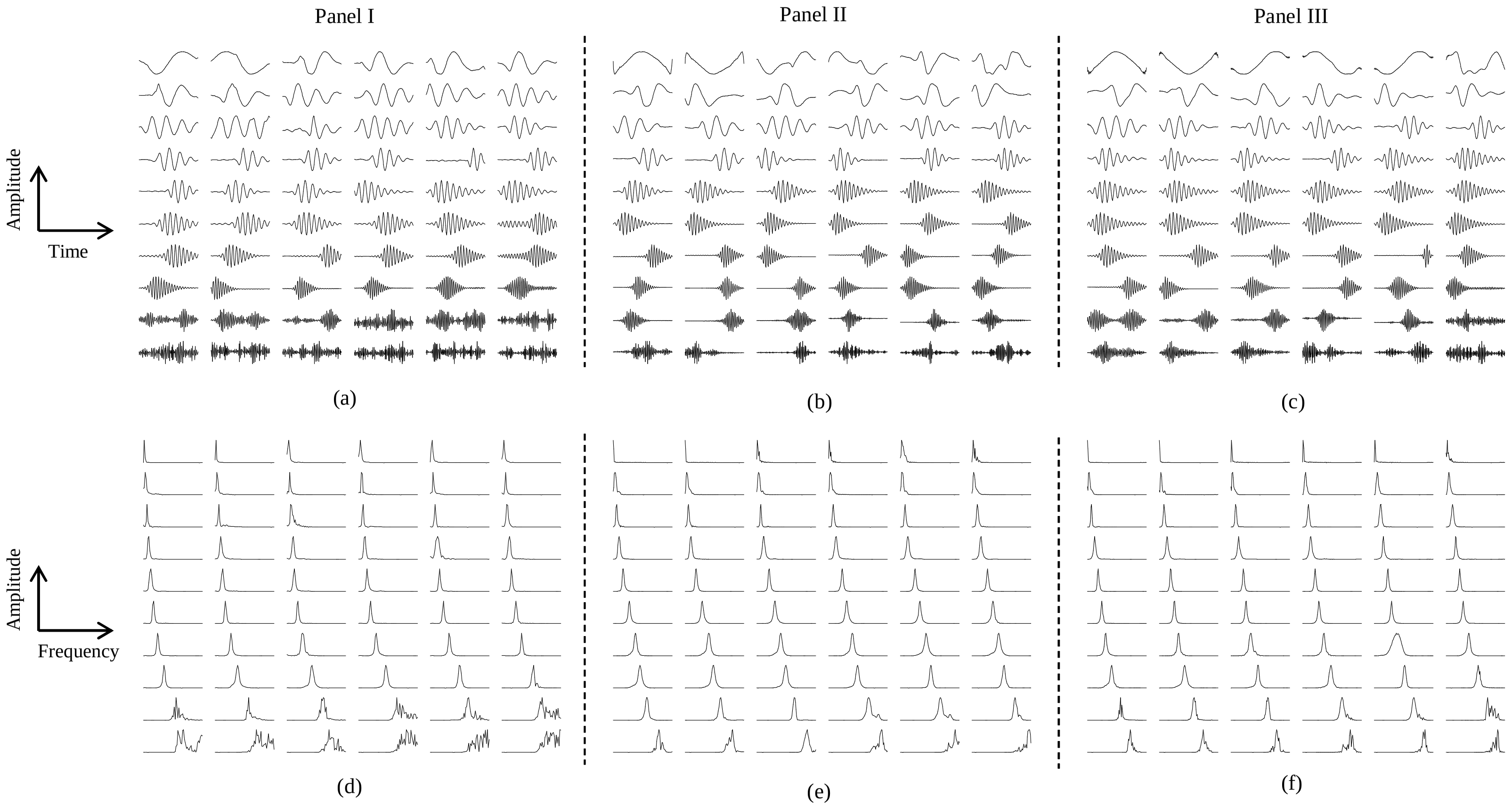


Figure 3.7: Examples of subband filters trained on the TIMIT (Panel I), WSJ1 (Panel II), and AURORA 4 (Panel III) databases, respectively:(a)-(c) subband filters in the time-domain (i.e., impulse responses), (d)-(f) subband filters in the frequency-domain (i.e., frequency responses). After [3].

Proposed subband filters can also accurately reconstruct a speech signal even after ReLU nonlinearity. A small segment of the original speech signal (about 500 samples) from the WSJ1 database, a segment of reconstructed speech from the model and the residual error are shown in Figure 3.8. From the residual error (RMSE = 0.032), we can see a very accurate reconstruction of the speech signal.

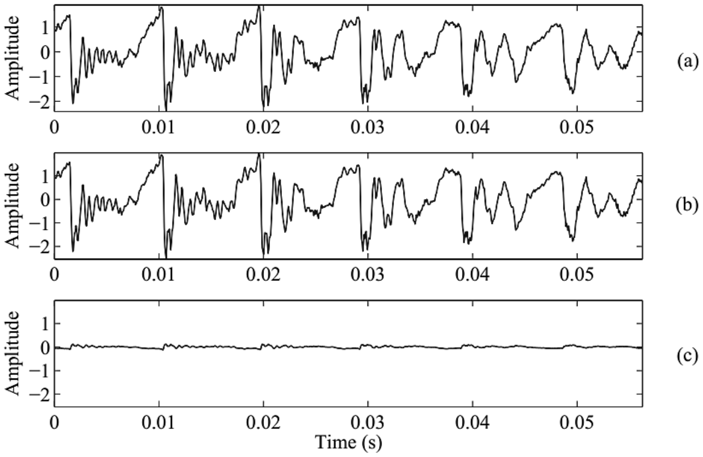


Figure 3.8: (a) Segment of speech signal, (b) reconstructed speech from the proposed model, and (c) residual error. Root Mean Squared Error (RMSE) between the original and reconstructed speech signal is 0.032. After [3].

3.5.2 Comparison with Standard Auditory Filterbanks

In order to compare a learned filterbank with the standard auditory filterbanks, we have shown a CF *vs.* subband filter index plot in Figure 3.9 for a filterbank learned on three databases. We can see that the ConvRBM filterbank has also a nonlinear relationship between CF and subband filter ordering similar to other auditory filterbanks (more closely with the Mel scale). This represents the placement of subband filters on the BM in the cochlea. Out of 60 subband filters, more than 40 subband filters have CFs below 4 kHz. Low frequency regions are represented by more number of subband filters learned by the model compared to the high frequency regions (similar to the Mel scale). Hence, the learned filters can represent frequency tuning in the human cochlea, which can be modeled more effectively using a bank of subband filters [61].

We have also computed Equivalent Noise Bandwidth (ENBW) of the ConvRBM subband filters as done in [105] for CNN filters. The scatter plot of bandwidth *vs.* CF is shown in Figure 3.10. The Gammatone filterbank (GTFB) with ERB scale is chosen as a reference since GTFB is more physiological filterbank. One can see that GTFB is perfectly a constant-Q filterbank since bandwidths of

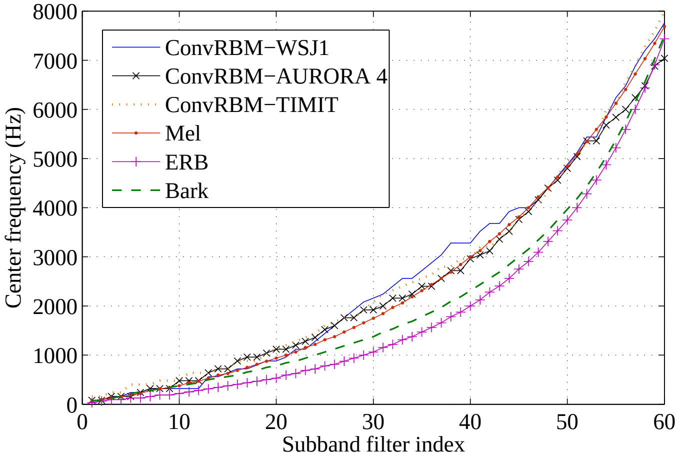


Figure 3.9: Comparison of filterbank learned using ConvRBM with auditory filterbanks. After [3].

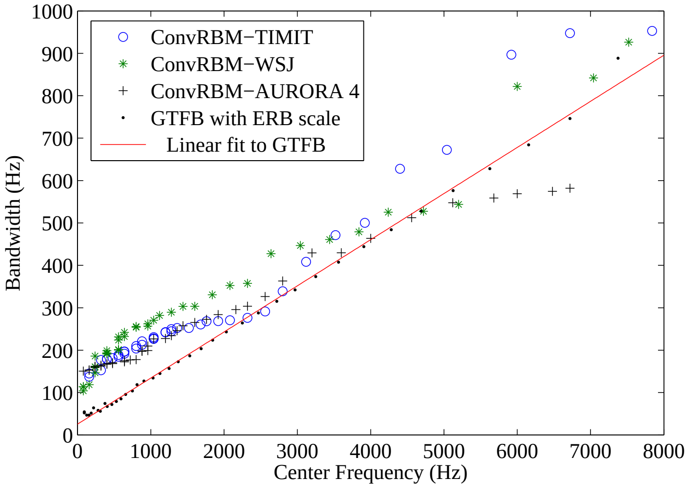


Figure 3.10: Bandwidth *vs.* CF for speech databases.

the Gammatone filters are progressively increasing as the increase in CF. ConvRBM filterbank for all the three databases also largely preserve the constant-Q nature. This observation is also consistent with the study presented in [12] for speech signals.

A detailed comparison of the filterbank is shown in Figure 3.11. In Figure 3.11 (a), filterbanks are compared with CFs up to 1 kHz. We can see that, in all the learned filterbanks, some of the subband filters have similar CFs. This redundancy is only observed for CFs up to 1 kHz and not in CFs above 1 kHz as shown in Figure 3.11 (b). This may be due to the lack of regularization in ConvRBM. We have also compared filterbanks trained on the clean WSJ0 database, and multi-condition training database AURORA 4 in Figure 3.11 (c). The difference between both the filterbanks can be seen after 2 kHz, since the filterbank learned from the

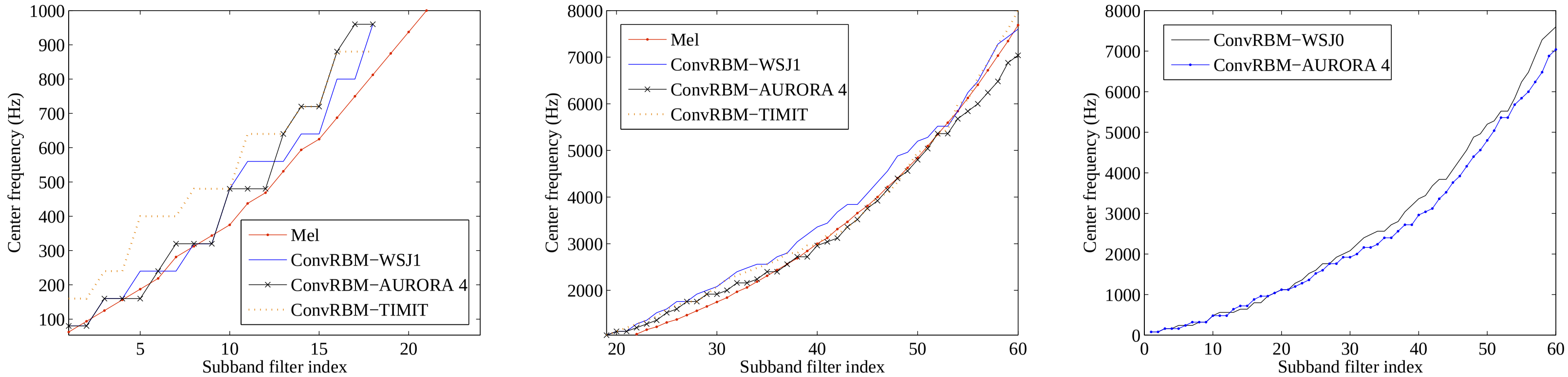


Figure 3.11: Comparison of a filterbank learned using ConvRBM with auditory filterbanks: (a) CF up to 1 kHz, (b) CF from 1 to 8 kHz and (c) between clean and multicondition training database. After [3].

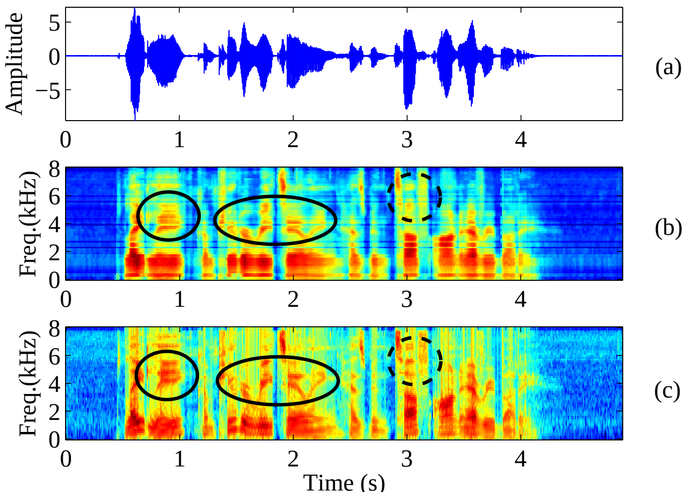


Figure 3.12: (a) Speech signal, (b) ConvRBM spectrogram, and (c) Mel spectrogram. Full line regions are marked to see similarities and the dotted circle indicates differences in both the spectrograms. After [3].

AURORA 4 database uses more subband filters compared to the clean WSJ0 in the low frequency regions. This observation is different from the one reported in [102], where the filterbank trained on a clean database uses more subband filters than the noisy database. However, the major difference is that the weights of our model were randomly initialized without any constraints, while in [102] weights were initialized using the gammatone filterbank.

The spectrum representation of the speech signal using subband filters is compared with the Mel spectrogram in Figure 3.12. Similar to a Mel spectrogram, a ConvRBM spectrogram indeed represents the spectrum information, such as formant contours, voiced, and the unvoiced sounds, etc. The regions marked using solid lines shows that learned subband filters are capturing spectrum information. However, the filterbank scale is slightly different from the Mel scale as seen from Figure 3.12. We have also noticed that for the ConvRBM filterbank, the resolution is slightly poor at the higher frequencies compared to the Mel spectrogram (e.g., the region marked by the dotted circles). In the next Section, we will discuss how ConvRBM subband filters can represent an optimal auditory code.

3.5.3 Optimal Auditory Code

The auditory code (also called as the auditory neural code) is an auditory representation obtained from the transformations applied on sounds, which encodes the unique characteristics of a particular sound [115]. It must represent a wide range of the auditory tasks that require the great sensitivity in time and frequency and be effective over the diverse nature of sounds present in the natural acoustic environments [115]. It has been suggested that our sensory systems might have evolved highly efficient coding strategies to maximize the information conveyed to the brain, while minimizing the required energy and neural resources [142], [143]. It was observed the first time by Lewicki that such auditory coding can be learned through the statistics of natural sounds [12]. The auditory code is optimal for the underlying natural sound categories and also resembles auditory *revcor* (reverse correlation) filters obtained from the auditory nerves of a cat [61]. The reverse correlation is a technique to estimate the auditory nerve impulse responses from the cochlea or STRFs from the auditory cortex [144]. The examples of auditory revcor filters are shown in Figure 3.13 obtained from the EarLab, Boston University, USA, [26], [13]. Comparison of our auditory subband filters in Figure 3.5.1 and Figure 3.13 concludes that our model can also learn the auditory-like codes similar to as obtained from the physiological experiments.

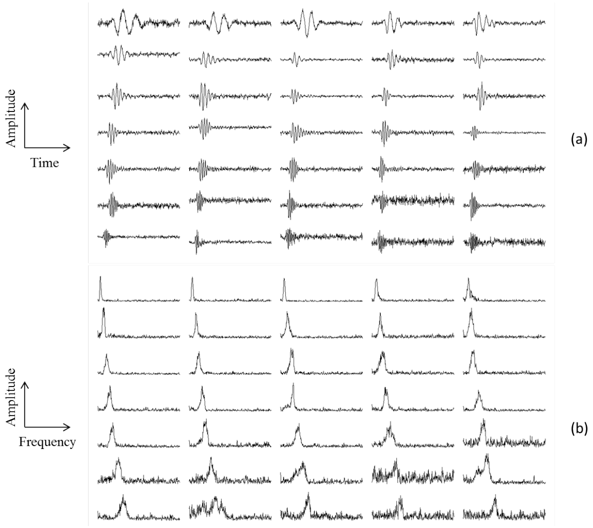


Figure 3.13: Examples of auditory revcor filters in (a) time-domain, and (b) frequency-domain. The **.mat* files are obtained from the website given in [26].

In this section, we will also investigate as to whether the learned subband filters represent an optimal auditory code (as investigated in other landmark studies [12], [115]). First, ConvRBM was trained on single speaker 'SLT' database taken from the CMU-ARCTIC database [145]. We have found that some subband filters are different from the ConvRBM trained on TIMIT or WSJ databases. The example of subband filters of ConvRBM trained on 'SLT' speaker is shown in Figure 3.14. We can see a *harmonic nature* of the subband filters (examples are marked by the dotted boxes) in both time and frequency-domains. This supports the findings reported in [12], and [109] that when a probabilistic model is trained on a single speaker database, subband filters represent formant contours as well as harmonic structures in the speech signal, which is speaker-specific (e.g., the fundamental frequency (F_0) and its harmonics or pitch contours). Such subband filters are not localized like the one that are trained on TIMIT and the WSJ database (as shown in Figure 3.5.1). Hence, we can say that the learned subband filters can best represent a single speaker or an *optimal auditory code* for that particular speaker. This may help for several speech processing applications where the speaker-specific information is required in the feature extraction stage.

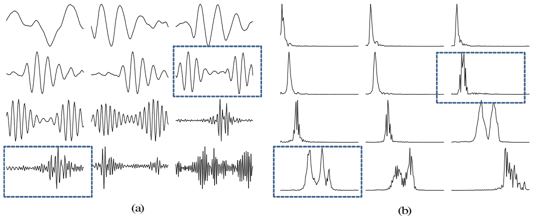


Figure 3.14: Training ConvRBM on a single speaker database: (a) subband filters in time-domain, and (b) corresponding frequency responses.

When a model is trained on a database with multiple speakers, it captures the *statistical properties* of speech signals, which may be speaker-independent [12]. In particular, such statistical properties can best represent the speech signal, which in turn leads to an *invariant representation* with respect to the speakers and environmental differences. Hence, the subband filters do not represent any harmonic structure, which is speaker-specific; rather it may represent vocal tract characteristics, i.e., formants (such as emphasis on lower formants F_1, and F_2 to aid for speech recognition tasks). Hence, these subband filters can be considered to be an optimal auditory code for a speech database prepared from multiple speakers.

3.5.4 Stability Analysis of ConvRBM to Additive Noise

Since the ConvRBM filterbank largely preserve constant-Q nature for speech signals, it can be shown that it is stable to time and frequency deformations similar as mathematically proved for constant-Q scattering wavelets [98]. Here, we will discuss the stability of the transformations in the ConvRBM w.r.t. the additive noise motivated by the studies in [98], [146]. Let T be the transformation applied on input $\mathbf{x} \in l^p$ that can be linear or nonlinear. Let us assume a bounded additive noise $\mathbf{n} = [n_1, n_2, ..., n_N]$, i.e., $|n_i| \leq M < \infty, \forall i$. We want to prove stability of ConvRBM for a signal with additive noise $\hat{\mathbf{x}} = \mathbf{x} + \mathbf{n}$. For T to be stable to the additive noise $\mathbf{n}$, the *Lipschitz continuity condition* needs to be satisfied for constant $\lambda > 0$, which is given as [146]:

$$\|T\mathbf{x} - T\hat{\mathbf{x}}\|_2 \leq \lambda \, \|\mathbf{x} - \hat{\mathbf{x}}\|_2 . \tag{3.30}$$

The definition of the Lipschitz continuity condition is given in Appendix D. This condition was derived for scattering convolutional networks [98]. Recently, it was proved for the supervised CNN with certain criteria, such as max-norm regularization for weights, ReLU nonlinearity, and max-pooling [147]. Our model also has the convolution and ReLU stages and hence, we can prove that ConvRBM is also stable to the additive noise.

3.5.4.1 Stability of Convolution in ConvRBM

The transformation T for the convolution operation in ConvRBM for the k^{th} group is $T\mathbf{x} = \mathbf{x} * \mathbf{W}^k$. For sequences $f_n \in l^p$ and $g_n \in l^g$ $(n \in \mathbb{N})$, Young's inequality is defined as follow:

$$\|f_n * g_n\|_r \leq \|f_n\|_p \|g_n\|_q , \tag{3.31}$$

where $1 \leq p, q, r \leq \infty$ and $\frac{1}{r} = \frac{1}{p} + \frac{1}{q} - 1$. Here, $\|f_n\|_p$ is defined for sequence $f_n = [f_1, f_2, ..., f_n]$ as $\|f_n\|_p = (\sum_{i=1}^{n} |f_i|^p)^{1/p}$. Young's inequality for the convolutions of the form, $T\mathbf{x} = \mathbf{x} * \mathbf{W}$ (with $\mathbf{W}^k \in l^p, \forall k$) can be written as:

$$\|\mathbf{x} * \mathbf{W}\|_2 \leq \|\mathbf{W}\|_1 \|\mathbf{x}\|_2 , \tag{3.32}$$

where $p = 1, q = 2, r = 2$ satisfy $1 \leq p, q, r \leq \infty$ and $\frac{1}{r} = \frac{1}{p} + \frac{1}{q} - 1$. The Lipschitz continuity for convolutions can be written as:

$$\|\mathbf{x} * \mathbf{W} - \hat{\mathbf{x}} * \mathbf{W}\|_2 \leq \|\mathbf{W}\|_1 \|\mathbf{x} - \hat{\mathbf{x}}\|_2 . \tag{3.33}$$

Comparing eq. (3.30) and eq. (3.33), the Lipschitz constant $\lambda = \|\mathbf{W}\|_1$. Hence, stability analysis depends on the L^1-norm of weights of ConvRBM. This condition is similar to the stability condition in the LTI system that the impulse response of an LTI system is absolutely summable or integrable (i.e., $\|\mathbf{W}\|_1 << \infty$) [148]. However, the Lipschitz continuity requires that $\|\mathbf{W}\|_1$ should be as small as possible. In [147], weights are max-norm regularized to obtain the stability criteria. ConvRBM training includes weight decay, which penalizes the weights to be small and smooth [20]. For TIMIT and AURORA 4 databases, we have observed that $\|\mathbf{W}\|_1 \leq 3$, and $\|\mathbf{W}\|_1 \leq 2.5$, respectively. Hence, based on the derivation in [147], for convolution operation in ConvRBM, the following stability condition holds:

$$\left\|\mathbf{x} * \mathbf{W}^k - \hat{\mathbf{x}} * \mathbf{W}^k\right\|_2 \leq \lambda \|\mathbf{x} - \hat{\mathbf{x}}\|_2 , \tag{3.34}$$

where $\langle\lambda_k\rangle \leq 3$ for TIMIT and $\langle\lambda_k\rangle \leq 2.5$ for the AURORA 4 ($\langle\lambda_k\rangle = \frac{1}{K}\sum_{k=1}^{K} \lambda_k$).

3.5.4.2 Stability of Rectified Nonlinearity

As discussed in Section 3.4, we have used the deterministic ReLU for feature extraction. It is proved in [147] that the ReLU operation is also stable with λ=1. The stability condition for ConvRBM with response, $\mathbf{I}_k$ for clean and $\hat{\mathbf{I}}_k$ for the additive noise is given as:

$$\left\|max(\mathbf{I}_k, 0) - max(\hat{\mathbf{I}}_k, 0)\right\|_2 \leq \left\|\mathbf{I}_k - \hat{\mathbf{I}}_k\right\|_2 . \tag{3.35}$$

We can prove this with four cases depending on signs of $\mathbf{I}_k$ and $\hat{\mathbf{I}}_k$.
Case 1: $\mathbf{I}_k, \hat{\mathbf{I}}_k > 0$

$$\left\|max(\mathbf{I}_k, 0) - max(\hat{\mathbf{I}}_k, 0)\right\| = \left\|\mathbf{I}_k - \hat{\mathbf{I}}_k\right\| . \tag{3.36}$$

Case 2: $\mathbf{I}_k, \hat{\mathbf{I}}_k < 0$

$$\left\|max(\mathbf{I}_k, 0) - max(\hat{\mathbf{I}}_k, 0)\right\| < \left\|\mathbf{I}_k - \hat{\mathbf{I}}_k\right\| . \tag{3.37}$$

Case 3: $\mathbf{I}_k > 0, \hat{\mathbf{I}}_k < 0$

$$\left\|max(\mathbf{I}_k, 0) - max(\hat{\mathbf{I}}_k, 0)\right\| = \|\mathbf{I}_k\| < \left\|\mathbf{I}_k - \hat{\mathbf{I}}_k\right\| \tag{3.38}$$

Case 4: $\mathbf{I}_k < 0, \hat{\mathbf{I}}_k > 0$

$$\left\|max(\mathbf{I}_k, 0) - max(\hat{\mathbf{I}}_k, 0)\right\| = \left\|\hat{\mathbf{I}}_k\right\| < \left\|\mathbf{I}_k - \hat{\mathbf{I}}_k\right\| \tag{3.39}$$

Hence, from all the four cases, eq. (3.35) is verified. The rectifier nonlinearity can also be viewed as a mapping $T : \mathbb{R} \rightarrow \mathbb{R}^+$ that has *fixed points* for $x > 0$ [149]. The stability of ConvRBM to the additive noise resulted in an improved performance in the AURORA 4 speech recognition task.

3.6 Experimental Setup

3.6.1 Speech Databases

3.6.1.1 Small Vocabulary Speech Database

We have used the TIMIT database for the phone recognition task [150]. In the TIMIT database, all SA category sentences (that are spoken by all the speakers) were removed as they may bias the speech recognition performance. Training data contain utterances from the 462 speakers. The Development set and test set contain utterances from 50 and 24 speakers, respectively.

3.6.1.2 Large Vocabulary Speech Databases

Large Vocabulary Continuous Speech Recognition (LVCSR) tasks were performed using the Wall Street Journal (WSJ) databases [151]. Both WSJ0 SI-84 (the subset of WSJ) and the full WSJ corpus (also known as WSJ1) are used for the experiments. The WSJ0 SI-84 training data consist of 14 hours of speech data, which include 7138 utterances spoken by the 84 speakers. Two Nov'92 evaluation sets, namely, 5K-word and 20K-word vocabulary (denoted as Eval92_5K and Eval92_20K, respectively), were used for testing. The WSJ1 training database is of 81 hours. Notations of the development and the evaluation sets for WSJ1 are as follows: D1 and D2 for the development sets Dev93 and Dev93_5k, E1 and E2 for the evaluation sets Eval93 and Eval93_5k, E3 and E4 for the evaluation sets Eval92 and Eval92_5k, respectively.

3.6.1.3 Noisy Speech Database

We have also used the AURORA 4 database (obtained from the WSJ0 database), which was created using six different types of additive noises, namely, car, a crowd of people (babble), restaurant, street, airport, and train station [152]. The multi-condition training database was prepared with 7138 utterances from the WSJ0 database with half of them recorded with a Sennheiser microphone and the other half recorded with a second microphone. The type of noise is randomly chosen out of six noises in total and at a randomly chosen SNR between 10 dB and 20 dB. A set of 330 utterances has been designated to perform a baseline recognition on the 5K word vocabulary. The test set consists of 14 subsets, each with 330 utterances denoted as T1 to T14. The test sets are grouped into four categories, namely, A: clean (set T1), B: noisy (set T2 to set T7), C: clean with channel distortion (set T8), and D: noisy with channel distortion (set T9 to T14).

3.6.2 Training of ConvRBM and Feature Extraction

We have trained ConvRBM on each individual speech databases. Each speech signal after mean-variance normalization was applied to the ConvRBM. The weight decay parameter was set to 0.001 (from the range 0.01-0.00001). The learning rate was empirically chosen to be 0.005 (from the range 0.01-0.0001), which was fixed for the first 10 epochs and decayed later at each epoch for stable learning of the model parameters. We observed that, with ReLUs, only 25-35 training epochs were found to be sufficient. For the first five training epochs, momentum was set to 0.5 and after that, it was set to 0.9. We have trained the model with different lengths of ConvRBM filters, and with different number of subband filters. Since the average phoneme duration in speech signal is less than or equal to 10 ms [153], the window duration is selected in the range 8-10 ms. After the model was trained, features were extracted from the speech signal as shown in Figure 3.6. To reduce the dimension of a feature vector and to compare proposed feature set with the MFCC, the DCT was applied and only first 13-D were retained. Delta and delta-delta features were also appended resulting in 39-D cepstral feature vector (indicated as ConvRBM-CC). We have not used DCT for filterbank experiments, and use only 40 subband filters of ConvRBM (indicated as ConvRBM-BANK) similar to the 40 subbands in Mel filterbanks.

3.6.3 ASR System Building

Baseline monophone GMM-HMM systems and hybrid DNN-HMM systems were built using 39-D MFCC and 120-D FBANK feature vectors, respectively, for all the databases used in this Chapter. The MFCC feature vectors were extracted from the windowed speech signal with a 25-ms length window, and a 10-ms window shift similar to the parameters of pooling. For the TIMIT database, 48 phones were used for training and mapped to 39 phones during the scoring [154]. The LM was performed using a bi-gram language model. For WSJ databases (WSJ0 and WSJ1), 5K and 20K tri-gram LMs were used using 46 phones. The 5K bi-gram LM and tri-gram LM were used for AURORA 4 test sets. In this thesis, all the ASR systems were built using the KALDI speech recognition toolkit [155]. We also experimented with the hybrid DNN-HMM system using the forced-aligned labels obtained from the corresponding GMM-HMM systems. The results are reported using DNN with 3 hidden layers, an 11-frame context-window, and 3000 hidden units. The DNN-HMM system combination is performed using the Minimum Bayes Risk (MBR) technique [156]. Lattices generated by N different systems are combined to get the optimal word sequence as follows [156]:

$$W^* = \arg\min_{W} \left\{ \sum_{i=1}^{N} \lambda_i \sum_{W'} P_i(W'|\mathbf{O}) L(W, W') \right\}, \tag{3.40}$$

where $L(W, W')$ is the Levenshtein edit distance between two word sequences, $P_i(W'|\mathbf{O})$ is the posterior probability of the word sequence W' given the acoustic observation sequence $\mathbf{O}$, and λ_i is the weight assigned to the i^{th} system.

3.7 Experimental Results

The significance of the ConvRBM filterbanks using various datasets is verified using a phone recognition task, an LVCSR task, and ASR in degraded conditions. We will first fine-tune the parameters of the model for each individual database and use the optimal set of parameters in corresponding ASR experiments.

3.7.1 Experiments on TIMIT Database

In this Section, the effect of a number of subband filters (K), filter length (m), and pooling type is verified through the experiments on the TIMIT database using GMM-HMM systems, and results are reported in Table 3.1 [2], [3]. We can see that the optimal filter length corresponding to the least Phone Error Rate (PER) (see Appendix B.1) is 128 samples on the development (Dev) and test set. A filter length of 128 samples (i.e., 8 ms) is sufficient to capture the small temporal variations in the speech signals [12]. In our case, average pooling works better than the max-pooling. Since we are using the rectifier nonlinearity, it eliminates the cancellations between neighboring filter outputs, when combined with the average pooling [157]. Hence, we achieved good performance with the average pooling. Best performance is obtained with 60 subband filters, 128-sample filter length, and using the average pooling.

Table 3.1: % PER for comparison of the number of subband filters (K), filter length (m), and pooling type on the TIMIT database. After [2], [3].

K	m	Pooling type	Dev	Test
40	128	Avg	32.0	32.6
60	128	Avg	**31.2**	**31.8**
80	128	Avg	31.5	31.9
60	96	Avg	31.4	32.5
60	160	Avg	31.7	33.0
60	256	Avg	32.8	33.5
60	128	Max	32.6	33.5

Avg=Average Pooling, Max=Maximum Pooling

The experimental results are reported in % PER and % relative improvement (in the parenthesis) in Table 3.2 [2], [3]. The relative improvements due to the ConvRBM-CC and ConvRBM-FBANK are shown w.r.t. the MFCC and FBANK, respectively. We can see that the ConvRBM-CC perform better than the MFCC giving an absolute reduction of 1.5 % in PER on the development set, and 1.7 % on the test set. Table 3.2 shows that for DNN-HMM systems, there is an absolute reduction of 1.1 % in PER using the ConvRBM-CC feature set, and 0.7 % using ConvRBM-BANK on the development set. We achieved an absolute reduction of 0.7 % (3.15 % relative) in PER using ConvRBM-CC, and 0.6 % (2.56 % relative) using the ConvRBM-BANK on the test set. Combining systems (denoted as $\oplus$) trained using both the filterbank features gave an absolute reduction of 1 % in PER compared to the ConvRBM-BANK and 1.7 % PER compared to the FBANK. The comparison of a supervised CNN trained on the raw speech signals shows that unsupervised ConvRBM-based features indeed perform better on the small size datasets. However, later on, we observed in Section 3.7.3 that a supervised CNN performs well with larger datasets, such as WSJ.

Table 3.2: % PER and relative improvements for TIMIT database. After [2], [3]

Feature Set	System	Dev	Test
MFCC	GMM-HMM	32.7	33.5
ConvRBM-CC	GMM-HMM	**31.2** (4.59)	**31.8** (5.07)
MFCC	DNN-HMM	23.0	24.0
ConvRBM-CC	DNN-HMM	**21.9** (4.78)	**23.3** (2.92)
A:FBANK	DNN-HMM	22.2	23.4
B:ConvRBM-BANK	DNN-HMM	**21.5** (3.15)	**22.8** (2.56)
A $\oplus$ B	DNN-HMM	**20.5** (7.66)	**21.7** (7.26)
CNN with the raw speech [158]		-	29.9

$\oplus$ denotes system combination experiments

3.7.2 Experiments on WSJ0 Database

The effects of parameters of ConvRBM were tested on the WSJ0 database and results are reported in Table 3.3. We can see a similar set of parameters as TIMIT that resulted in a lower % Word Error Rate (WER) (see Appendix B.1). The results of ASR experiments are reported in Table 3.4 in terms of % (WER) [2], [3]. There is an absolute reduction of 0.99 % WER on the eval92_5K test set, and 1.92 % WER on the eval92_20K test set over MFCC using the GMM-HMM system. Significant absolute reduction of 2.3 % WER is obtained for the 20K test set using the DNN-HMM systems. The lowest WER 5.85 % (3.6 % relative improvement) for the 5K test is achieved with the ConvRBM-BANK, while improvement is less using the ConvRBM-CC. There is a relative improvement of 14.6 % over the MFCC and 5.6 % over the FBANK for the 20K test set. The ConvRBM-CC and ConvRBM-BANK yielded almost similar WER for the 20K test set. This may be due to different numbers of the subband filters in ConvRBM feature set ($K = 60$ followed by 13-D DCT) and the ConvRBM-BANK ($K = 40$) to compare results with the MFCC and

FBANK, respectively. However, compared to the FBANK, an absolute reduction of 1.24 % (8.66 % relative) in WER for the 20K test set, and 0.67 % (11.40 % relative) for the 5K test set was achieved by the MBR system combination.

Table 3.3: % WER for comparison of number of subband filters (K), filter length (m) and pooling type on the WSJ0 database. After [2], [3]

K	m	Pooling type	Eval92_5K	Eval92_20K
40	128	Avg	13.49	26.21
60	128	Avg	12.96	25.80
80	128	Avg	13.41	25.66
60	96	Avg	13.97	25.94
60	160	Avg	13.25	26.1
60	256	Avg	13.75	27.01
60	128	Max	13.50	26.80

Table 3.4: % WER and relative improvements for the the WSJ0 database. After [2], [3]

Feature Set	System	Eval92_5K	Eval92_20K
MFCC	GMM-HMM	13.95	27.72
ConvRBM-CC	GMM-HMM	**12.96** (7.09)	**25.80** (6.93)
MFCC	DNN-HMM	6.30	15.70
ConvRBM-CC	DNN-HMM	**6.05** (3.97)	**13.40** (14.65)
A:FBANK	DNN-HMM	6.07	14.32
B:ConvRBM-BANK	DNN-HMM	**5.85** (3.62)	**13.52** (5.59)
A $\oplus$ B	DNN-HMM	**5.40** (11.04)	**13.08** (8.66)

$\oplus$ denotes system combination experiments

3.7.3 Experiments on WSJ Database

With the parameters of ConvRBM obtained from the WSJ0, we experimented on the full WSJ database (i.e., WSJ1). The results are reported in Table 3.5 in terms of % Word Error Rate (WER). We can see that our ConvRBM-CC feature set gave an improvement on all the test sets. Using the GMM-HMM systems, we achieved a relative improvement of 8 % (3-1.85 % absolute) on the development sets, 1.9-5.2 % on the evaluation set: E1 and E2 and 10.7-12.3 % (2.85-1.72 % absolute) on the evaluation set: E3 and E4. Using DNN-HMM systems, ConvRBM-CC gives relative improvement of 9.65-13.42 % on the development sets, 7.21-17.8 % on the evaluation sets, E1 and E2 and 12.49-13.16 % on the evaluation sets, E3 and E4. Experiments on the filterbank feature sets also show improvements (1.35-6.82 %

relative improvements) using ConvRBM-BANK compared to the FBANK except on the test set E4. System combination of both the filterbank-based feature sets gives further improvements with an absolute reduction of 1.44 %, 1.45 % and 0.95 % in WER for the test set D1, E1 and E3, respectively, compared to the FBANK feature set. The comparison with supervised CNN (for test set E4) trained on the raw speech signals shows that on larger data sets the supervised method performs slightly better compared to our proposed unsupervised method.

Table 3.5: % WER and % relative improvements (shown in the brackets) for the WSJ1 LVCSR task. After [3]

Feature Set	D1	D2	E1	E2	E3	E4
GMM-HMM system						
MFCC	37.45	23.04	30.95	17.87	26.60	13.94
ConvRBM-CC	**34.37**	**21.19**	**29.35**	**17.53**	**23.75**	**12.22**
	(8.22)	(8)	(5.17)	(1.9)	(10.71)	(12.34)
DNN-HMM system						
MFCC	20.94	11.40	17.75	9.76	13.93	5.47
ConvRBM-CC	**18.92**	**9.87**	**16.47**	**8.02**	**12.19**	**4.75**
	(9.65)	(13.42)	(7.21)	(17.82)	(12.49)	(13.16)
A:FBANK	18.70	9.66	17.31	8.10	12.26	4.39
B:ConvRBM-BANK	**17.96**	**9.53**	**16.13**	**7.92**	**11.80**	4.91
	(3.96)	(1.35)	(6.82)	(2.22)	(3.75)	(-11.84)
A ⊕ B	**17.26**	**9.04**	**15.86**	**7.76**	**11.31**	**4.22**
	(7.7)	(6.42)	(8.38)	(4.2)	(7.75)	(3.87)
ConvRBM-BANK with CD-DNN-HMM, bi-gram 5k LM						6.4
CNN with raw speech, bi-gram 5k LM [104]						5.6

⊕ denotes system combination experiments

3.7.4 Experiments on the AURORA 4 Database

Multi-condition AURORA 4 training data is used for the ASR system building. ConvRBM parameter tuning experiments on AURORA 4 database are shown in Table 3.6. Since we need robustness against the signal degradation conditions, we choose % WER of test sets, B and D as ConvRBM parameter selection criteria. The ConvRBM with filter lengths 160 and 60 number of filters, is found to perform relatively best for test sets B and D (however, the difference in % WER using both the sets of parameters is very small).

The comparison of different feature sets are given in Table 3.7. Performance on test sets with channel distortions is improved using context-independent DNN-HMM (CI-DNN-HMM) systems. We obtained a relative reduction of 6.7 % WER on test set B, and an absolute reduction of 4.5 % and 3.05 % in WER for test sets C and D, respectively. The ConvRBM-BANK gave an absolute reduction of 1.25 % on test set B and 1.73 % on test set D over the FBANK feature set. For test sets A and C, an absolute reduction of 1.57 % and 3.85 %, respectively, is achieved using ConvRBM-BANK compared to the FBANK feature set. High absolute reduction in % WER is obtained using system combination S1 $\oplus$ S2 of FBANK and ConvRBM-BANK trained systems, respectively.

Table 3.6: % WER for comparison of number of subband filters (K), filter Length (m) and pooling type on AURORA 4 database. After [3]

K	m	Pooling type	A	B	C	D	Avg
40	128	avg	21.65	35.70	38.71	51.65	36.92
60	128	avg	22.53	32.77	**36.63**	49.04	35.24
60	128	max	21.48	**32.72**	37.34	49.08	35.15
80	128	avg	21.95	34.2	36.93	50.53	35.90
60	160	avg	**21.02**	32.76	37.08	**48.95**	**34.95**

We have also reported results on the CD-DNN-HMM with the forced-aligned labels obtained from the triphone GMM-HMM system. In both bi-gram and tri-gram 5K LM cases, the ConvRBM-BANK showed improvements compared to the FBANK. With bi-gram 5K LM, 3 % (the relative improvement) is achieved for the channel distortion test sets. With tri-gram 5K LM, 1.4-12.94 % relative improvements are achieved for test sets (less improvements for the test set B and set D). System combination for both LMs gives significant improvements compared to the baseline FBANK systems. This shows that complementariness of both the filterbanks-based features further helps for a robust ASR task.

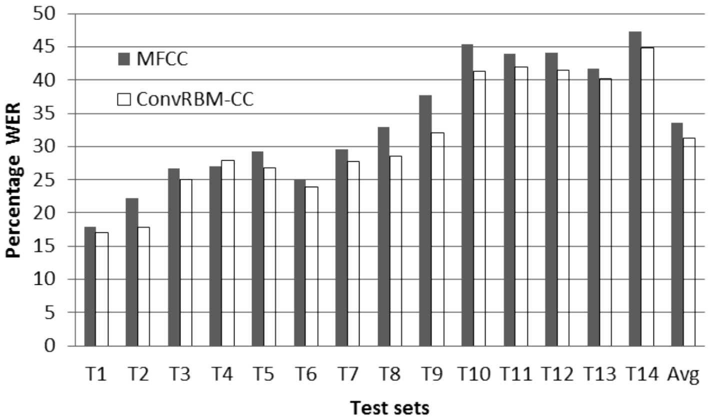


Figure 3.15: Detailed evaluation of AURORA 4 test sets using MFCC and ConvRBM-CC feature sets. After [3].

Table 3.7: % WER and % relative improvements (as shown in the brackets) for AURORA 4 database. After [3]

Feature Set	A	B	C	D	Avg
GMM-HMM system tri-gram 5k LM					
MFCC	22.62	33.21	39.4	49.51	36.18
ConvRBM-CC	**21.02**	**32.76**	**37.08**	**48.95**	**34.95**
	(7.07)	(1.35)	(5.89)	(1.13)	(3.39)
CI-DNN-HMM system with tri-gram 5k LM					
MFCC	17.92	26.63	32.97	43.36	30.22
ConvRBM-CC	**17.06**	**24.84**	**28.47**	**40.31**	**27.67**
	(4.8)	(6.72)	(13.65)	(7.03)	(8.44)
S1:FBANK	12.33	21.59	29.35	38.57	25.46
S2:ConvRBM-BANK	**10.76**	**20.34**	**25.5**	**36.84**	**23.36**
	(12.73)	(5.79)	(13.12)	(4.49)	(8.24)
S1 $\oplus$ S2	**10.65**	**19.22**	**26.42**	**36.38**	**23.17**
	(13.63)	(10.98)	(10)	(5.68)	(8.99)
CD-DNN-HMM system with bi-gram 5k LM					
S3:FBANK	10.61	14.85	20.38	30.71	19.12
S4:ConvRBM-BANK	**9.68**	**14.81**	**19.58**	**29.69**	**18.44**
	(8.77)	(0.3)	(3.9)	(3.32)	(3.6)
S3 $\oplus$ S4	**9.47**	**13.91**	**18.85**	**28.52**	**17.69**
	(10.74)	(6.33)	(7.5)	(7.31)	(7.48)
CD-DNN-HMM system with tri-gram 5k LM					
S5:FBANK	5.62	9.29	15.15	24.27	13.58
S6:ConvRBM-BANK	**4.89**	**9.15**	**13.86**	**23.93**	**12.95**
	(12.98)	(1.5)	(8.5)	(1.4)	(4.6)
S5 $\oplus$ S6	**4.71**	**8.43**	**13.53**	**22.74**	**12.35**
	(16.19)	(9.26)	(10.69)	(6.3)	(9.06)

$\oplus$ denotes system combination experiments

Detailed evaluations of AURORA 4 test sets are shown in Figure 3.15 and Figure 3.16 for the CI-DNN-HMM systems. From Figure 3.15, we can see that, in all the test conditions, ConvRBM-CC are performing better than the MFCC feature set except the T4 test set. ConvRBM-FBANK also performs better than the FBANK feature set except for restaurant noise conditions (i.e., test sets T4 and T11). To justify the improvements in ASR task using the AURORA 4 database, we have investigated log-spectrum amplitude variations during the time for three test conditions (namely, babble, street, and babble+distortion) against a clean log-spectrum as a reference. A log-spectrum for a subband filter with CF 2.16 kHz is plotted for ConvRBM in Figure 3.17 (a)-(c) and for the Mel spectrum in Fig-

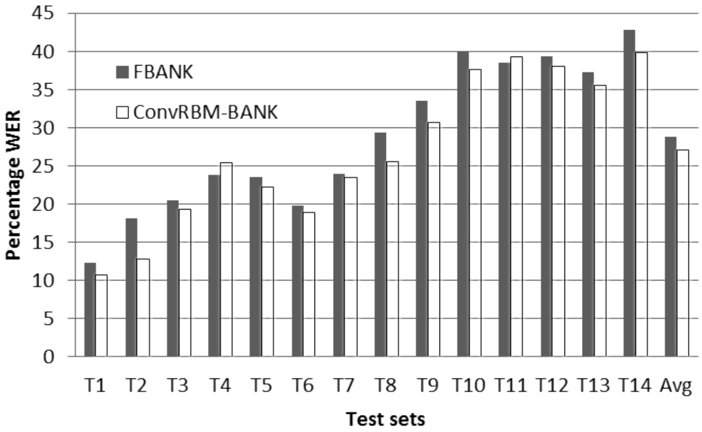


Figure 3.16: Detailed evaluation of AURORA 4 test sets using FBANK and ConvRBM-BANK feature sets. After [3].

ure 3.17 (d)-(f). Since we have not applied any mean-variance normalization on the spectrum, there is a difference in amplitude level in all the noisy spectra. It is clearly seen that the Mel spectrum is very much affected by the noise (examples of selected regions are marked in Figure 3.17 (d)-(f)) in all the test conditions compared to the ConvRBM spectrum. Hence, the ConvRBM trained filterbank is likely to reduce the noise distortions, which may improve the ASR performance in the degraded conditions as well. Comparison of our approach using a bi-gram 5K LM with other approaches (specifically, convolutional networks) is given in Table 3.8. Our supervised back-end is DNN with 3 hidden layers as we have discussed in Section 3.6.3. Many recent architectures, such as [159] and [160] are able to perform quite well for AURORA 4 task.

Table 3.8: Comparison of % WER for AURORA 4 database using different approaches. After [3]

Approaches	A	B	C	D	Avg
Our approach (S4, 3 layers)	9.68	14.81	19.58	29.69	18.44
Our approach (S3 $\oplus$ S4, 3 layers)	9.47	13.91	18.85	28.52	17.69
DNN (5 layers) [161]	10.6	16.4	15.8	26.6	20.3
CNN (5 layers, 1-D filters) [161]	9.5	14.8	14.6	23.6	18.2
PNS-CNN (2-D filters) [121]	7.4	13.4	12.8	24.7	17.8
CNN (2-D filters) [159]	5.1	8.8	8.5	20.1	13.4
AD Maxout CNN [160]	4.0	7.8	6.7	14.9	10.5

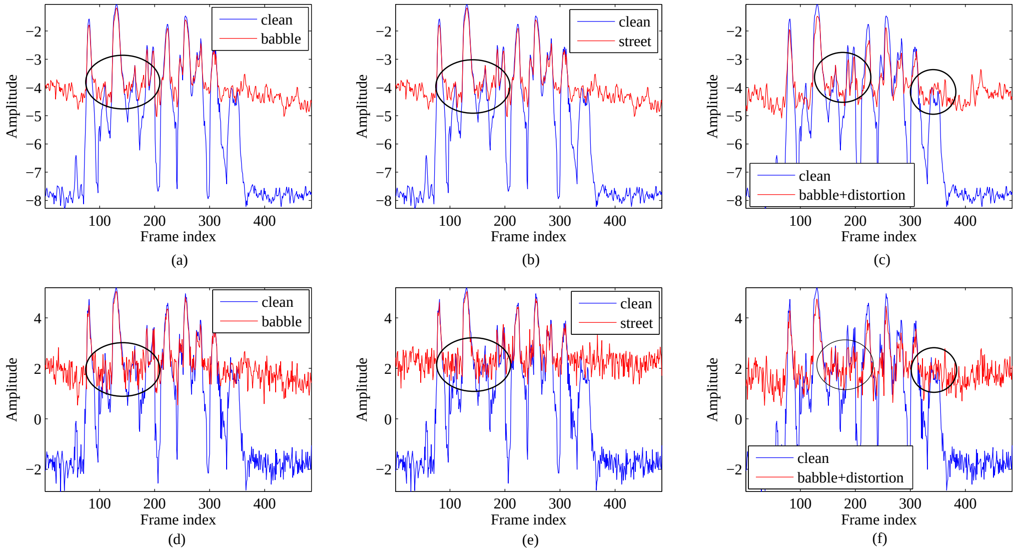


Figure 3.17: Comparison of spectrum for one subband filter on AURORA 4 test sets. (a)-(c) ConvRBM generated spectrum, (d)-(f) Mel spectrum. Highlighted regions show the examples of regions distorted due to the noise. After [3].

3.7.5 Cross-domain Experiments

We have also experimented using a TIMIT-trained ConvRBM filterbank for the AURORA 4 speech recognition task, and vice-a-versa to see whether ConvRBM subband filters are a generalized representation of auditory processing? We have changed the filterbanks in the front-end to extract features from the TIMIT, and AURORA 4 databases. Once the features were extracted, acoustic modeling was performed as per the TIMIT and AURORA 4 tasks. Following are the results of cross-domain experiments.

Table 3.9: Results of the TIMIT phone recognition task using the AURORA 4 trained ConvRBM in % PER. After [3]

ConvRBM training database	Test
TIMIT	22.8
AURORA 4	23.6

Table 3.10: Results of the AURORA 4 task using the TIMIT-trained ConvRBM. After [3]

ConvRBM Training Database	A	B	C	D	Avg
AURORA 4	9.68	14.81	19.58	29.69	18.44
TIMIT	9.9	14.92	19.3	29.18	18.52

Table 3.9 shows that relative % PER of subband filters of the AURORA 4 database is 3.4 % higher (an absolute difference of 0.8 %) compared to the subband filters of the TIMIT database. However, there is no significant difference in % PER, when we have used subband filters trained on the different database (along with different training conditions). Table 3.10 shows that the % WER of all AURORA 4 test sets are similar for subband filters of both the databases. These results also explain that, even with a small amount of the TIMIT training data, we can achieve similar gains on the larger AURORA 4 task. These experiments also suggest that we can train ConvRBM on larger and more diverse datasets to use it for smaller ASR task or different application in the form of transfer learning.

3.8 Chapter Summary

In this chapter, we have described the architecture of the proposed model using ConvRBM for auditory filterbank learning. The proposed model can take arbitrary length speech signals as an input to the ConvRBM. The detailed theory of the model learning and algorithm to train the model using speech signals is presented. The feature extraction from the trained ConvRBM is discussed. We have shown that the proposed model is able to learn the auditory-like subband filters from the raw speech signals. Comparisons with standard auditory frequency scales further justify the proposed model's capability for the auditory representation learning. Results are shown for the ASR task on the phonetically balanced TIMIT database, and statistically meaningful LVCSR databases, WSJ and AURORA 4. In the next chapter, we will discuss the improved version of the ConvRBM for filterbank learning.

CHAPTER 4

Improved Auditory Model

4.1 Introduction

Earlier we described our proposed model for filterbank learning from speech signals in Chapter 3. In this Chapter, we present our improved auditory model using an annealed dropout for regularization and Teager Energy Operator (TEO) as a noise-robust energy estimation technique. The ConvRBM training using an annealed dropout is discussed in Section 4.2. The effect of an Adam optimization algorithm applied on the ConvRBM is discussed in Section 4.3. The application of TEO on the subbands of ConvRBM is presented in Section 4.4. The proposed feature representation is discussed and analyzed in Section 4.5. The ASR experiments and the statistical significance of the results are presented in Section 4.6.

4.2 Dropout Convolutional RBM

Dropout is a stochastic regularization technique that prevents a network from overfitting by preventing the co-adaptation of weights in the network. The term dropout refers to randomly dropping out neurons (i.e., assigning the zero value) in a neural network with a probability p. In Chapter 3, we presented our proposed model of auditory processing using the ConvRBM. In ConvRBM training, a dropout is applied before sampling the hidden units in both the positive and negative phase of CD-1 learning. Applying a dropout to the ConvRBM can be thought of as multiplying each unit in the k^{th} group with a binary mask (called the *dropout mask*). The dropout mask for the k^{th} group is defined as random variables drawn from the Bernoulli distribution, i.e.,

$$\mathbf{m}_k = Bernoulli(p), \tag{4.1}$$

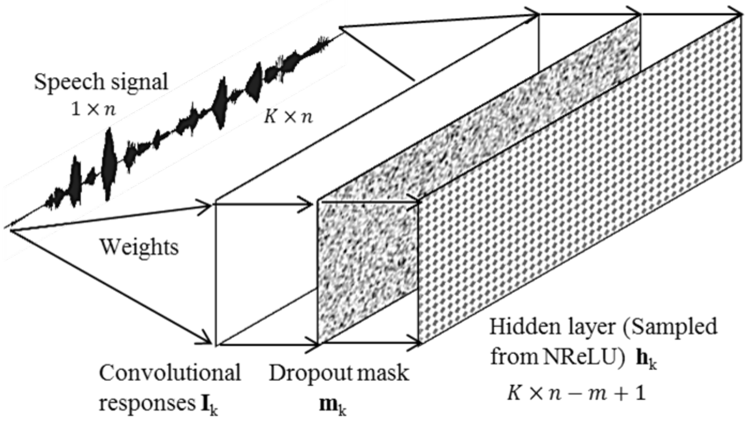

Speech signal
$1 \times n$
$K \times n$
Weights
Convolutional responses $\mathbf{I}_k$
Dropout mask $\mathbf{m}_k$
Hidden layer (Sampled from NReLU) $\mathbf{h}_k$
$K \times n - m + 1$

Figure 4.1: The block diagram of dropout ConvRBM model. After [4].

where $P(m_k = 0) = p$ and $P(m_k = 1) = 1 - p$. The sampling equations for the hidden and visible units in the dropout ConvRBM are given as:

$$\begin{aligned} \mathbf{h}^k &\sim \max(0, \mathbf{m}_k \odot \mathbf{I}_k + \mathcal{N}(0, \text{sigmoid}(\mathbf{m}_k \odot \mathbf{I}_k))), \\ \mathbf{x}_{recon} &\sim \mathcal{N}\left(\sum_{k=1}^{K}(\mathbf{h}^k * \mathbf{W}^k) + c, 1\right). \end{aligned} \tag{4.2}$$

Here, $\odot$ indicates an elementwise multiplication. The block diagram of our proposed ConvRBM architecture is shown in Figure 4.1. In [4], we explored an annealed dropout training of ConvRBM that was proposed for supervised deep networks in [162]. In an annealed dropout, the dropout probability of the units in the network is gradually decreased over the training period. We have used the following annealing dropout schedule as suggested in [162]:

$$P[t] = \max\left(0, \left(1 - \frac{t}{N}\right) P[0]\right), \quad t \in [0, N], \tag{4.3}$$

where $P[0]$ is the initial dropout rate at training iteration, $t = 0$. The dropout rate is decayed from $P[0]$ to a small value or zero for $t = N$ iterations. After N iterations, $P[t]$ is kept zero. In the next section, we discuss the improved ConvRBM parameter updates using Adam optimization.

4.3 ConvRBM Training with Adam Optimization

Until now, we used the stochastic gradient descent (SGD) algorithm to update the ConvRBM parameters. In this Section, the novel stochastic optimization method called Adam is used, which is based on first-order gradient-based optimization of the ConvRBM parameters. The Adam computes individual adaptive learning rates for different parameters from the estimates of first and second moments of the gradients [163]. The name Adam is derived from adaptive moment estimation [163]. The Adam optimization has several advantages, such as the magnitudes of parameter updates being invariant to rescaling of the gradient, its step sizes are approximately bounded by the step size hyperparameter, it does not require a stationary objective, it works with sparse gradients, and it naturally performs a form of step size annealing [163]. The pseudo-code of the Adam optimization is shown in Algorithm F.4. The effect of Adam optimization on the RMSE between the original and reconstructed speech signals (averaged over the entire database of AURORA 4) is shown in Figure 4.2. It can be seen that the RMSE curve over the training iterations for the Adam optimization is significantly lower compared to the SGD optimization in the ConvRBM.

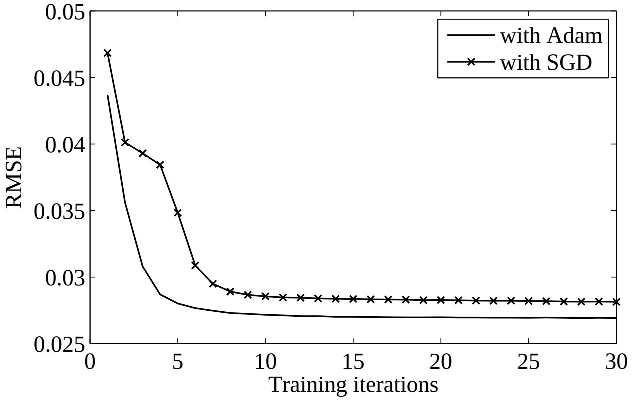


Figure 4.2: The comparison of SGD and Adam optimization in ConvRBM training on the RMSE during training iterations.

In next the section, we discuss the noise-robust energy estimation using the Teager Energy Operator (TEO).

4.4 Representing Energy in the Auditory System

Among many acoustic and perceptual features of the speech signal, temporal modulations are one of the important parametric representations of the speech signal. Temporal modulations describe changes of a speech signal in terms of amplitude modulation (AM) and frequency modulation (FM) [164]. The acoustic-phonetic analysis examines the AM/FM in the speech signal for detecting and characterizing speech sounds [61]. It is observed that AM and FM always *co-occur* and are inseparable features of speech signals [164]. The AM-FM responses can be obtained from the auditory filterbank. However, instead of separating AM and

FM responses after a filterbank, we consider here using an operator that can track the running estimate of energy jointly contributed by both AM and FM. The non-linear operator known as the Teager Energy Operator (TEO) introduced by Kaiser can effectively estimate the “true" energy of a signal or the energy required to produce a signal [165]. The discrete-time version of the TEO applied on the AM-FM signal of the form, $s[n] = a[n]cos(\phi[n])$ is defined as [166]:

$$\Psi\{s[n]\} := s^2[n] - s[n-1]s[n+1] \approx a^2[n]\omega^2[n], \tag{4.4}$$

where $a[n]$ and $\omega[n] = \frac{d}{dn}\phi[n]$ are discrete time-varying amplitude and instantaneous frequency (the derivative of instantaneous phase, $\phi[n]$), respectively. According to the noise suppression capability of the TEO, for a signal with additive noise, i.e., $\hat{s}[n] = s[n] + v[n]$, the TEO of $\hat{s}[n]$ is given as [167]:

$$\Psi\{\hat{s}[n]\} = \Psi\{s[n]\} + \Psi\{v[n]\} + 2\tilde{\Psi}\{s[n], v[n]\}, \tag{4.5}$$

where $s[n]$ is a clean signal and $v[n]$ is a zero-mean additive noise. Using eq. (4.4), $\tilde{\Psi}\{s[n], v[n]\}$ can be derived as:

$$\tilde{\Psi}\{s[n], v[n]\} = s[n]v[n] - (1/2)s[n-1]v[n+1] - (1/2)s[n+1]v[n-1]. \tag{4.6}$$

Here, $s[n]$ and $v[n]$ are assumed to be zero-mean independent stochastic processes and hence, the expected value of cross-TEO terms $\tilde{\Psi}\{\cdot\}$ are zero [167]. In addition, $\mathbb{E}[\Psi\{v[n]\}]$ is approximately zero since $\mathbb{E}[\Psi\{v[n]\}] = R_{vv}(0) - R_{vv}(2) \approx 0$. This can also be explained by observing the power spectral density (PSD) of a noise before and after applying the TEO as shown in Figure 4.3. We can see that, the PSD of three different types of noises are lower when TEO is applied and hence, the spectral power of a noise is reduced. Since the babble noise is made of random speech signals (e.g., people talking in the background), the PSD is similar to without applying the TEO after 4 kHz. Still for the lower frequencies, the PSD is lower after applying the TEO. Hence, $\mathbb{E}[\Psi\{\hat{s}[n]\}] \approx \mathbb{E}[\Psi\{s[n]\}]$, where $\mathbb{E}[\cdot]$ is an expectation operator [167]. This shows the significance of the TEO as a noise-robust energy estimator that has noise suppression capability. The detailed derivation of noise suppression capability of the TEO is given in Appendix C. Application of the TEO on the responses of subband signals represents the noise-robust energy estimation of the basilar membrane (BM) (as done in [168]).

Despite recent breakthrough studies using deep learning for ASR, it is shown that feature engineering indeed helps for the robust ASR task [161]. One of the approaches for the robust feature representation is to use the Teager Energy Op-

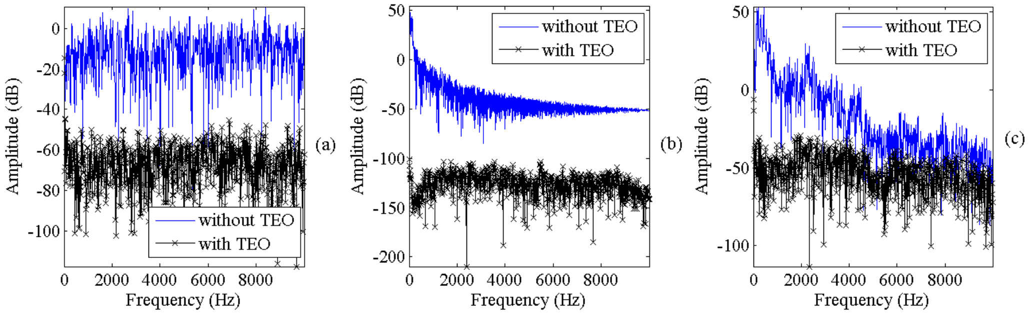


Figure 4.3: The effect of TEO on the PSD of noises: (a) white, (b) car, and (c) babble.

erator (TEO) [165]. TEO is used as an energy estimation along with an auditory feature processing pipeline as proposed in [168]. Compared to earlier works in the ASR with TEO-based feature sets [167, 168], in this work, we have reported results on the AURORA 4 database using filterbank learning as a front-end and various deep networks as a back-end. The ConvRBM model of filterbank learning and feature extraction is similar to an early auditory processing stage in many auditory models. The TEO-based energy estimation on the subband filtered signals from the ConvRBM filterbank is discussed in the next section.

4.5 Proposed Feature Representation

The block diagram of our proposed feature extraction method is shown in Figure 4.4. Inspired by state-of-the-art auditory models [169], a lowpass filter with cutoff frequency 1 kHz is also used after the ConvRBM responses. The lowpass filtering retains the temporal fine structure (TFS) of the subband signals (as discussed in Appendix A) at low frequencies and extracts the envelope of the signal at high frequencies (that corresponds to the phase-locking phenomenon) [61], [169]. The energy estimates using TEO for each subband were pooled later to obtain the short-term features followed by a logarithmic compressive nonlinearity. The average pooling is used based on our experiments reported in [3].

The subband filters trained using ConvRBM with and without dropout are shown in Figure 4.5. We can see the higher number of low frequency subband filters when ConvRBM is trained with annealed dropout as shown in Figure 4.5. The high-frequency subband filters (Figure 4.5 (b)) are less noisy when using the annealed dropout due to the dropout regularization effect. To analyze how an annealed dropout training affects the nonlinear relationship between the CF and filter ordering, the frequency scales of ConvRBM are compared with standard au-

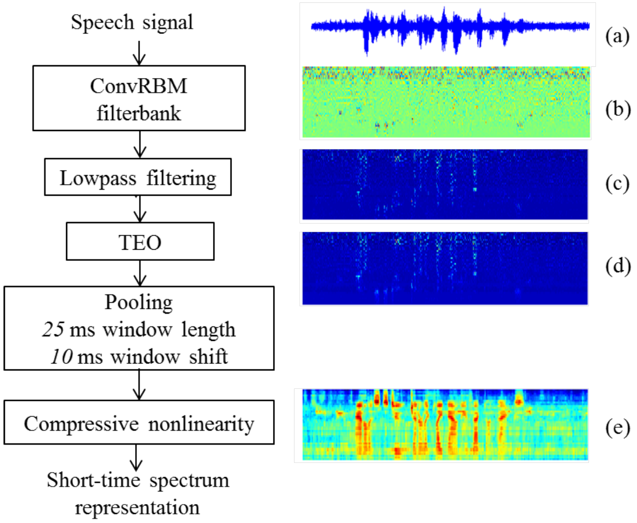


Figure 4.4: Block diagram of the proposed feature extraction method: (a) speech signal, (b) responses from the filterbank (c) lowpass filtering, (d) energy estimation using TEO, and (e) short-time spectrum representation.

ditory filterbanks in Figure 4.6. The ConvRBM trained with an annealed dropout uses more subband filters in the frequency range 500-5500 Hz compared to the ConvRBM without dropout. It follows the ERB scale up to 1 kHz, the Bark scale from 1 kHz-2.5 kHz and after that it is in between the Mel and the Bark scale. The bandwidth *vs*. CF scatter plot is shown in Figure 4.7. Due to regularization effect of annealing dropout, the ConvRBM filterbank trained on AURORA 4 database obtained better constant-Q nature compared to the ConvRBM trained without dropout.

The spectrogram representation of the proposed approach (Figure 4.4) is compared with the Mel and ConvRBM spectrograms in Figure 4.8. The ConvRBM filterbank is able to suppress the noise compared to the Mel filterbank. An application of the TEO on the ConvRBM filterbank suppresses the noise much better compared to the ConvRBM alone. The formant transitions are clearly visible in the TEO applied ConvRBM. One can also see that the noise in the silence regions (before and after the utterance) is completely eliminated after the TEO applied. The significance of a lowpass filtering is shown in Figure 4.9 for the same utterance taken in Figure 4.8. The TEO supressed the noise at the cost of reducing the spectral energies in the lower frequencies (below 4 kHz). Since this lowpass filter is introduced to mimic phase synchrony, it extracts the envelope of the signal at the higher frequencies (above 1 kHz). Hence, application of the TEO on the lowpass filtered subbands preserves the spectral energies for subbands above 1 kHz. We have also verified this with reduced % WER in Section 4.6.

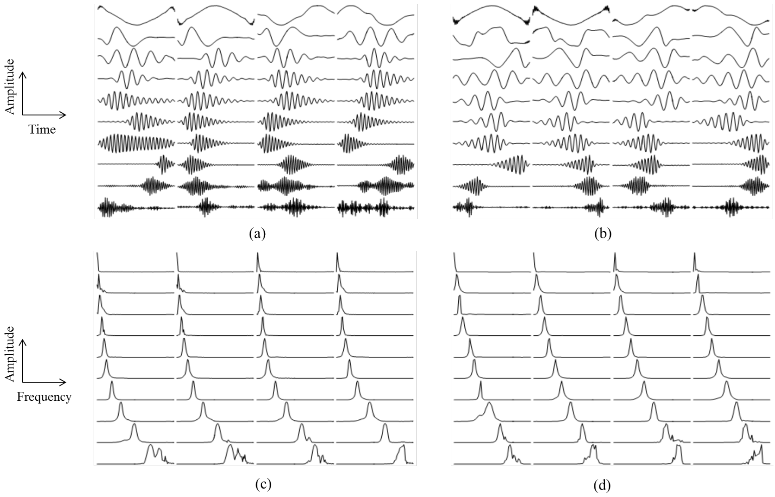


Figure 4.5: The subband filters trained on the AURORA 4 database: time and frequency-domain subband filters using ConvRBM without dropout ((a) and (c)) and ConvRBM with annealed dropout ((b) and (d)).

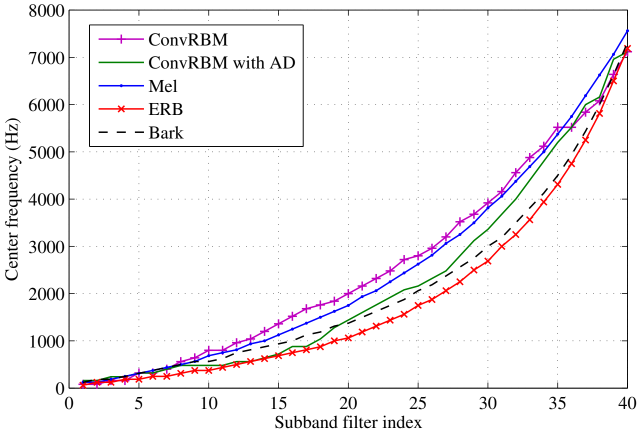


Figure 4.6: Comparison of filterbank scale learned using ConvRBM (with and without annealed dropout (AD)) with auditory filterbanks.

4.6 Experimental Setup and Results

4.6.1 ASR System Building

We have used the AURORA 4 multicondition training database as described in Section 3.6.1, Chapter 3. The triphone GMM-HMM systems were built using 39-D MFCC feature vectors to obtain the forced-aligned labels. MFCC and Mel filterbank (FBANK) feature sets were used in the GMM-HMM and DNN-HMM system building, respectively. The WSJ0 bi-gram 5K language model was used for AURORA 4 test sets. All the ASR systems were built using the KALDI toolkit [155]. Three types of neural networks were used in this study, namely, CNN [170],

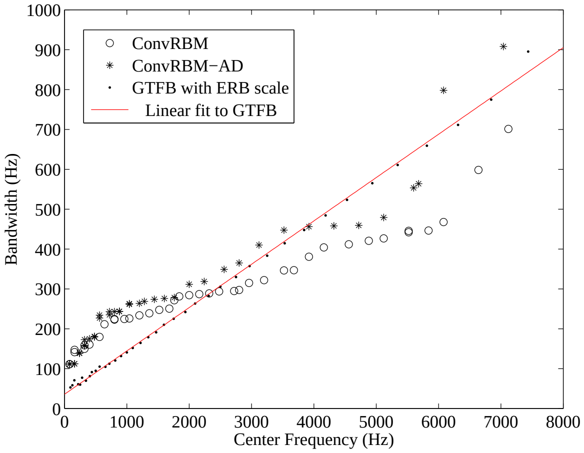


Figure 4.7: Comparison of bandwidth *vs*. CF for filterbank learned using ConvRBM (with and without annealed dropout (AD)).

TDNN [75], and BLSTM networks [21]. The results are reported using (1) CNN with 150 and 300 feature maps in the first and second layer and 1024 hidden units in other two layers, (2) TDNN with 5 hidden layers and 1024 hidden units, and (3) BLSTM networks with 3 hidden layers and 800 cells. The system combination is performed using the MBR technique [156] as discussed in Chapter 3, Section 3.6.

4.6.2 Training of ConvRBM and Feature Extraction

We have trained ConvRBM with an annealed dropout (AD) using $P[0] = 0.3, 0.4, 0.5$ decayed to $P[N] = 0$ and a fixed dropout (FD) using $P[t] = 0.3, \forall t \in [0, N]$. The learning rate was chosen to be 0.01 (two times larger than the one used in [3]), which was fixed for first 10 epochs and decayed later at each epoch. For the first five training epochs, momentum was set to 0.5 and after that, it was set to 0.9. The model is trained with 40 subband filters (i.e., K) and a convolution window of length (i.e., m) 128 samples. After the model was trained, features were extracted from the speech signal as discussed in Section 4.5. The delta and delta-delta features were also appended resulting in a 120-dimensional (120-D) feature vector. The notations for different feature sets (120-D) are given in Table 4.1.

4.6.3 Experimental Results

The results of the ASR experiments using different configurations of the ConvRBM filterbank and TEO are shown in Table 4.2 in terms of % word error rate (WER). The performance of the AD-CBANK is better than the CBANK and FD-CBANK on average. The AD-CBANK has low WER for test sets C and D com-

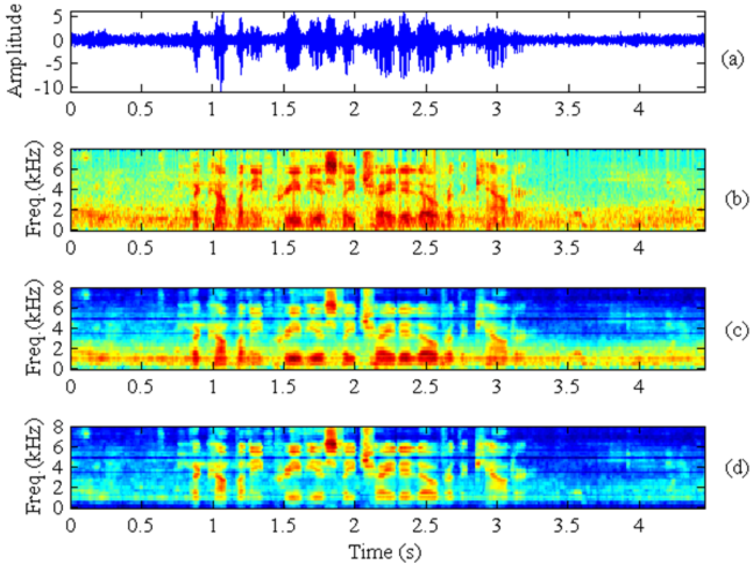


Figure 4.8: Comparison of spectrograms: (a) speech signal, (b) Mel spectrogram, (c) ConvRBM spectrogram, and (d) TEO applied ConvRBM spectrogram.

Table 4.1: Notations of different feature sets used in this study

Feature Set	Description
FBANK	Mel filterbank
CBANK	ConvRBM filterbank
FD-CBANK	ConvRBM filterbank learned using FD
AD-CBANK	ConvRBM filterbank learned using AD
TEO-CBANK	TEO applied on CBANK
TEO-FD-CBANK	TEO applied on FD-CBANK
TEO-AD-CBANK	TEO applied on AD-CBANK

pared to the FBANK. Application of the TEO on the CBANK without half-wave rectification (HWR) (with the method similar to as in [168]) resulted in improvements with all the three types of ConvRBM configurations. Thus, the noise suppression capability of the TEO indeed helps to reduce % WER. FD-CBANK did not perform well compared to the CBANK and AD-CBANK, which shows the significance of the annealing dropout technique. Our proposed TEO-AD-CBANK resulted in reduced % WER compared to the FBANK in the channel distortion test sets (i.e., C and D) as can be seen from Table 4.2. TEO-AD-CBANK resulted in WER of 15.43 % and 25.83 % for test sets C and D, respectively. Hence, an absolute reduction of 1.43-2.49 % in WER was achieved using the TEO-AD-CBANK compared to the FBANK. We also investigated the effectiveness of the HWR nonlinearity with the proposed feature pipeline (Figure 4.4). TEO-AD-CBANK with HWR got improvements only for the test sets A and B. Hence, TEO directly applied on

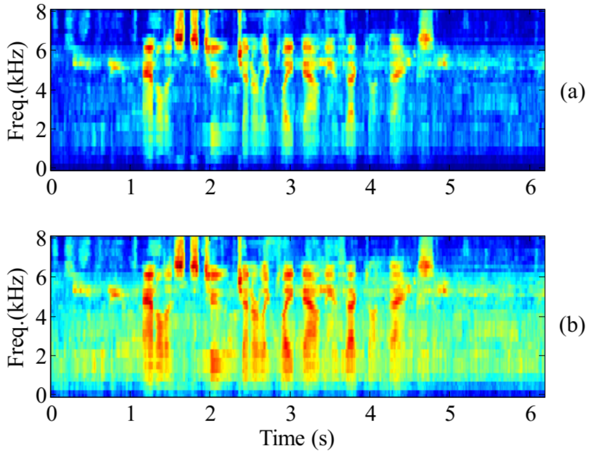


Figure 4.9: Effect of lowpass filtering: TEO-ConvRBM spectrogram (a) without lowpass filtering, and (b) with lowpass filtering.

CBANK performs well. Furthermore, TEO-AD-CBANK feature extraction without lowpass filtering (LPF) did not perform well. TEO-AD-CBANK without HWR and using LPF proved to be a better feature representation for the AURORA 4 task and is used for other experiments. The experiments show that $P[0] = 0.3$ performs well compared to other $P[0]$ values as shown in Table 4.2. We have also shown the results for all the 14 test sets for different noise types in Figure 4.10 using TDNN models. The proposed TEO-AD-CBANK feature set performs better than the FBANK except for the comparable performance in test sets T2, T5, and T13, as can be observed from Figure 4.10.

Table 4.2: % WER for the AURORA 4 test sets using TDNN models. Here, $P[0]$ indicates dropout probability and ✓ means the corresponding technique is applied vice-versa. After [4]

Feature Set	LPF	HWR	$P[0]$	A	B	C	D	Avg
FBANK	-	-	-	11.48	15.21	17.92	27.26	20.3
CBANK	-	✓	-	11.17	16.04	17.4	28.14	20.9
FD-CBANK	-	✓	0.3	10.63	15.61	18.42	27.35	20.48
AD-CBANK	-	✓	0.3	12.06	15.29	17.62	27.16	20.31
TEO-CBANK	✓	×	0.3	10.5	15.50	16.38	25.99	19.70
TEO-FD-CBANK	✓	×	0.3	10.83	15.45	17.4	26.79	20.12
TEO-AD-CBANK	✓	×	0.3	10.89	15.17	**15.43**	**25.83**	**19.45**
TEO-AD-CBANK	✓	✓	0.3	**10.44**	**15.04**	16.73	26.64	19.80
TEO-AD-CBANK	×	×	0.3	11	15.16	16.62	26.40	19.78
TEO-AD-CBANK	✓	×	0.4	10.44	15.33	16.66	26.49	19.86
TEO-AD-CBANK	✓	×	0.5	11.13	15.27	16.84	26.56	19.92

We have used TDNN (in Table 4.2) to generate the alignments for all the ex-

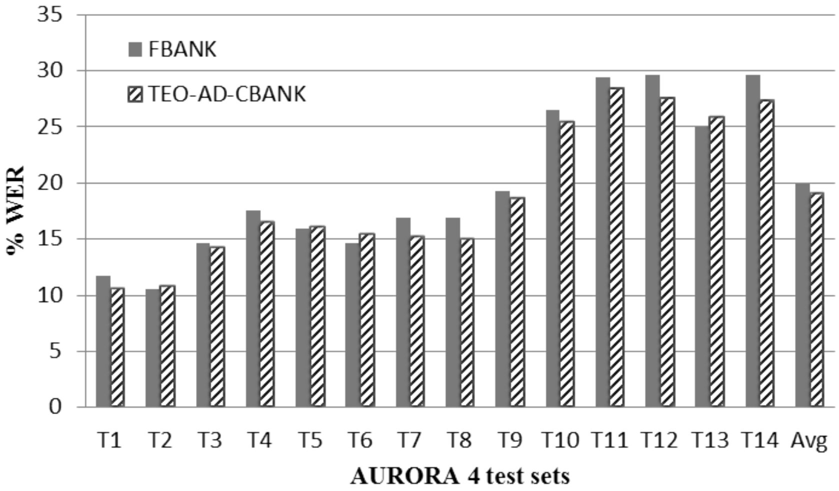


Figure 4.10: Detailed comparison of FBANK and TEO-AD-CBANK features on the AURORA 4 test sets using TDNN models. After [4].

periments in Table 4.3. The CNN models did not perform well compared to the TDNN and BLSTM, which exploits the temporal context better compared to the traditional CNN. Using TDNN, TEO-AD-CBANK features gave relative improvements of 2.59-11.63 % for test sets A to D compared to the FBANK. With BLSTM models, TEO-AD-CBANK gave relative improvements of 1.26-6.87 % for test sets A, C, and D, respectively, compared to the FBANK. The system combination (denoted as ⊕) of both feature sets resulted in a significant improvement for both TDNN and the BLSTM models. The best results were achieved using S3 ⊕ S4 based on the BLSTM model that gave an absolute reduction of 1.56-2.82 % and 1.18-2.22 % in WER compared to the FBANK and TEO-AD-CBANK, respectively.

Table 4.3: % WER for the AURORA 4 test sets using various deep networks. After [4]

Feature Set	Model	A	B	C	D	Avg
FBANK	CNN	10.55	14.19	18.14	27.70	20
TEO-AD-CBANK	CNN	10.55	14.5	18.4	27.82	20.1
S1:FBANK	TDNN	11.72	15.04	16.86	26.56	19.87
S2:TEO-AD-CBANK	TDNN	10.55	14.65	14.9	25.54	19.04
S3:FBANK	BLSTM	9.65	14.62	15.73	25.91	19.18
S4:TEO-AD-CBANK	BLSTM	9.27	14.80	14.65	25.42	18.94
S1 ⊕ S2	TDNN	9.49	13.05	13.39	23.86	17.45
S3 ⊕ S4	BLSTM	**8.09**	**12.73**	**12.91**	**23.20**	**16.90**

The significance of Adam optimization in ConvRBM training is shown in Table 4.4. The filterbank obtained by training ConvRBM using Adam optimization followed by the TEO is used in the AURORA 4 task with the BLSTM acoustic models. Compared to the SGD optimization in ConvRBM, Adam optimization provides an absolute reduction of 1 % in WER on the noisy test sets B and D. Compared to the baseline FBANK feature set, there is a relative reduction of 5-6.6 % in WER on the AURORA 4 test sets. Furthermore, the system combination with FBANK significantly reduces % WER compared to FBANK alone. Use of an Adam optimization in TEO-AD-CBANK improves performance compared to the TEO-AD-CBANK with SGD optimization.

Table 4.4: % WER task for significance of Adam optimization in ConvRBM

Feature Set	Optimization	A	B	C	D	Avg
S1:FBANK	-	9.65	14.62	15.73	25.91	19.18
S2:TEO-AD-CBANK	SGD	9.27	14.80	14.65	25.42	18.94
S3:TEO-AD-CBANK*	Adam	9.13	13.89	14.69	24.37	18.1
S1 $\oplus$ S2	-	8.09	12.73	12.91	23.20	16.90
S1 $\oplus$ S3*	-	**8.07**	**12.35**	13.04	**22.66**	**16.51**

*These results were obtained later and not part of our research study reported in [4].

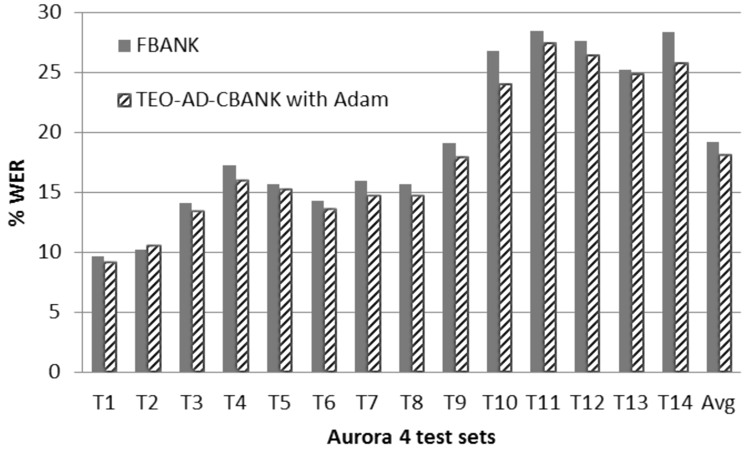


Figure 4.11: Detailed comparison of FBANK and TEO-AD-CBANK with Adam optimization on AURORA 4 test sets using BLSTM models.

4.6.4 Statistical Significance of ASR Results

The statistical significance of the ASR results was assessed using the bootstrap algorithm discussed in Appendix B. Here, we describe the bootstrap algorithm for the ASR task. It is assumed that the test corpus contains N number of sentences for which the recognition result is independent, and the number of errors can thus be evaluated independently. For each sentence i, we record the number of words,

n_i, and the errors e_i as follows:

$$X = \{(n_1, e_1), ..., (n_N, e_N)\}. \tag{4.7}$$

Generate the bootstrap sample for $b = 1, ..., B$ such as given below:

$$X^{*b} = \{(n_1^{*b}, e_1^{*b}), ..., (n_N^{*b}, e_N^{*b})\}. \tag{4.8}$$

The sample will contain several of the original sentences multiple times, while others are missing. Then, we calculate the WER on this sample as follows [171]:

$$WER^{*b} := \frac{\sum_{i=1}^{N} e_i^{*b}}{\sum_{i=1}^{N} n_i^{*b}}. \tag{4.9}$$

The WER^{*b} are called the bootstrap replications of WER. They can be thought of as samples of the WER from an ensemble of virtual test sets. The bootstrap estimate of the WER is given by [171]:

$$WER_{boot} := \langle WER^{*} \rangle \approx \frac{1}{B} \sum_{b=1}^{B} WER^{*b}. \tag{4.10}$$

Given the two ASR systems with the WER counts e_i^{ASR1} and e_i^{ASR2}, the difference in WER is [171]:

$$\Delta WER := WER^{ASR1} - WER^{ASR2} = \frac{\sum_{i=1}^{N}(e_i^{ASR1} - e_i^{ASR2})}{\sum_{i=1}^{N} n_i}. \tag{4.11}$$

The difference in number of errors is calculated on the identical bootstrap samples. The bootstrap estimate of probability of error reduction is defined as [171]:

$$\begin{aligned} \text{POI} &:= P(\Delta WER^{*} < 0), \\ &= \langle \Theta(-\Delta WER^{*}) \rangle, \\ &\approx \frac{1}{B} \sum_{b=1}^{B} \Theta(-\Delta WER^{*b}), \end{aligned} \tag{4.12}$$

where $\Theta(x)$ is the step function, which is one for $x > 0$. This statistical measure is called the *Probability of Improvement (POI)*. The steps in POI computation are summarized in Algorithm 2. We used the method proposed in [171] for quoting the statistical significance test that is based on the bootstrap technique.

To show how our proposed feature set TEO-AD-CBANK performs compared

Algorithm 2 The bootstrap algorithm for the ASR task

```
Input: Original test set $X$ and two trained ASR systems, namely, ASR1 and ASR2
Output: The POI estimate
for each bootstrap interval, $b = 1, 2, ..., B$ do
    Generate a random test set with replacement $X_b^*$ of size $X$
    Compute the bootstrap WER for both system $WER^{ASR1*}$ and $WER^{ASR2*}$
    Compute the difference in the bootstrap WER:
    $\Delta WER^{*b} := WER^{ASR1*} - WER^{ASR2*}$
end for
POI = $\frac{1}{B}\sum_{b=1}^{B}\Theta(-\Delta WER^{*b})$
```

to the FBANK, we have found the % POI as formulated in [171] to compare the two systems. The statistical significance of the AURORA 4 test sets is shown in Figure 4.12 for all the 14 test sets. It can be seen that for the TDNN acoustic models, except test sets T2, T6 and T13, % POI values are significantly higher (with 100 % POI in some of the cases). For BLSTM acoustic models, except test sets, T2, T4, and T5, % POI values are more than 25 % for all the test sets. Hence, a statistical significance test shows that the TEO-AD-CBANK features perform better than the FBANK (with significant % POI in many test sets) for both the acoustic models (i.e., TDNN and BLSTM). The POI values for the FBANK and TEO-AD-CBANK are summarized in terms of standard AURORA 4 test sets in Table 3. One important aspect of the TEO-AD-CBANK feature set is the high % POI values for channel distortion test sets (i.e., C and D). This observation is in line with the reduction of % WER in channel distortion test sets. Hence, the TEO-AD-CBANK performs well compared to the FBANK with 75.09 % and 55.84 % POI for the TDNN and BLSTM models, respectively, indicating the statistical significance of the results for the proposed approach.

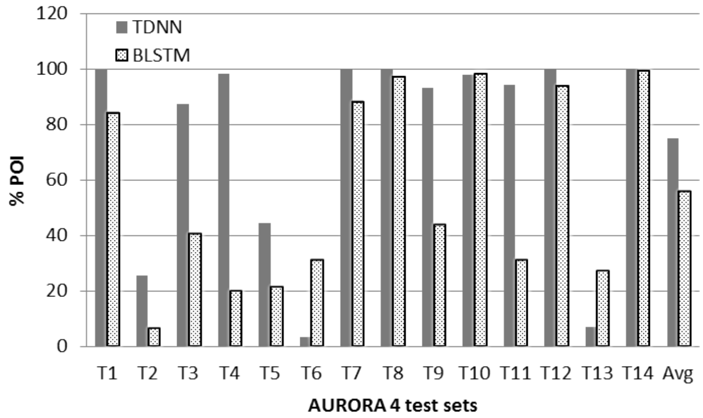


Figure 4.12: The % POI values for AURORA 4 test sets.

Table 4.5: % POI using the TEO-AD-CBANK over FBANK for AURORA 4 test sets. After [4]

Model	A	B	C	D	Avg
TDNN	99.59	59.89	99.95	82.07	75.09
BLSTM	84.01	34.47	97.29	65.61	55.84

4.6.5 Comparison with the ASR Literature

The comparison of the proposed features with the literature is given in Table 4.6. We have also compared the BLSTM model trained with the Teager Energy Spectral Coefficients (TESC) implemented using the Gammatone filterbank (GTFB) [168]. Application of the TEO on GTFB reduces % WER compared to the FBANK. However, it did not perform well compared to the TEO-AD-CBANK. This indicates that applying TEO on the learned ConvRBM filterbank indeed improves the performance in the ASR task. The power normalized spectra (PNS) include more complex auditory processing stages [121]. Hence, there is an absolute 1 % difference in WER compared to the TEO-AD-CBANK. However, the proposed system combination performs similar and/or better to the PNS-CNN [121]. The study reported in [161] used various features including the auditory-motivated Normalized Modulation Coefficients (NMC) and Damped Oscillator Coefficients (DOC). The % WER for NMC and DOC is lower compared to our proposed work. This may be due to the generation of the alignments for DNN models from the fMLLR-based speaker adaptation techniques in the GMM-HMM pipeline as opposed to our speaker-independent GMM-HMM models (in which no such speaker adaptation is used). The results are not compared with the studies, such as in [162], that used the clean training alignments.

Table 4.6: Comparison of our proposed feature representation approach and the system combination model with different feature sets in the ASR literature

Feature Set	A	B	C	D	Avg
Proposed: TEO-AD-CBANK [4]	9.27	14.80	14.65	25.42	18.94
Proposed: TEO-AD-CBANK (Adam)	9.13	13.89	14.69	24.37	18.1
FBANK $\oplus$ TEO-AD-CBANK (Adam)	**8.07**	**12.35**	**13.04**	**22.66**	**16.51**
TESC	9.44	14.58	15.06	25.57	18.96
Gabor-DNN [121]	8.4	14.2	14.3	25.8	18.8
PNS-CNN [121]	7.4	13.4	12.8	24.7	17.8
NMC [161]	8.4	11.9	11.4	21.3	15.64
DOC [161]	9.0	11.9	11.8	21.8	15.93

$\oplus$ represents the system combination experiments

4.7 Chapter Summary

In this Chapter, our efforts to improve our proposed model for the noise-robust ASR task are presented. We applied an annealed dropout as a regularization technique in ConvRBM training, where the dropout probability is reduced gradually. To improve the gradient-based optimization, an Adam optimization technique is explored. Furthermore, the noise-robust energy estimation approach based on the TEO is applied on each subband signal of the ConvRBM. Experiments on the AURORA 4 database were conducted using various deep networks and compared the results with the ASR literature. The statistical significance tests using the bootstrap technique show the efficacy of the proposed auditory feature representation. In the next Chapter, we apply ConvRBM on the agricultural speech database for the ASR task in the Gujarati language (an Indian language).

Chapter 5

ASR in the Agricultural Domain

5.1 Introduction

In this Chapter, an ASR system for the Gujarati language is discussed for the development of a speech-based access system in the agricultural domain. The research study in this Chapter is a part of the MeitY, Govt. of India sponsored consortium project, namely, "Speech-Based Access of Agricultural Commodity Prices and Weather Information in 12 Indian Languages". We are developing this system in the Gujarati language at Speech Research Lab, DA-IICT. The system architecture, data collection, and transcription are discussed in Section 5.2. The experimental setup and results for the ASR task are given in Section 5.3 and Section 5.4, respectively. The Chapter is summarized in Section 5.5.

5.2 Speech-Based Access for Agricultural Commodity

5.2.1 The Need of a Speech-Based Access System for Agriculture

Gujarat is one of the states that provide the highest grossing in India's agricultural progress. With farming land of 98 lakh hectares, Gujarat is a major agricultural crop provider with major possible crops being cotton, groundnut, castor, etc. Several websites are maintained to provide current prices of agricultural commodities [172], [28]. Thus, it helps farmers to know the better prices to sell their crops. However, socio-economic status and variation in educational backgrounds make them less aware and accessible to the agricultural commodity prices, weather forecasts and various Government schemes for the benefits of farmers.

To provide information to the farmers, we are developing a speech-based access system in Gujarati (an Indian language) called the Mandi Information System (MIS). In this Chapter, mandi means the marketplace in India. The project is based on using an Interactive Voice Response System (IVRS) and an ASR system trained on voices recorded from the farmers in the districts of Gujarat. The farmers need to call a toll-free number and follow the call flow to get information regarding the prices of commodities and the weather. This telephone-based system is helpful to all the farmers even with those who have different educational backgrounds, since they can get information in their native regional language just by making a telephone call. Earlier, the MIS were built for the six Indian languages [27, 173–177].

5.2.2 System Architecture

There are three major building blocks of the system, namely, information source, IVRS, and ASR system as shown in Figure 5.1. Everyday information regarding the agricultural commodities is fetched from the AGMARKNET webportal maintained by the Government of India [28]. The information regarding the weather forecast is fetched from the IMD website maintained by the Ministry of Earth Science, Government of India [29]. Snapshots of the AGMARKNET and IMD websites are shown in Figure 5.2 and Figure 5.3, respectively. Our local agricultural database is updated based on a webcrawler program that automatically updates the AGMARKNET and IMD data related to the Gujarat state. An IVRS is used to record the speech signals from the farmers via a telephone line (called the Primary Rate Interface (PRI) line). Based on the information from the database, the response is given to the farmer for a query recognized by the ASR system.

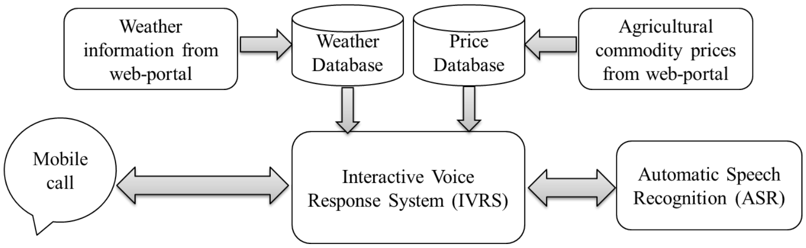
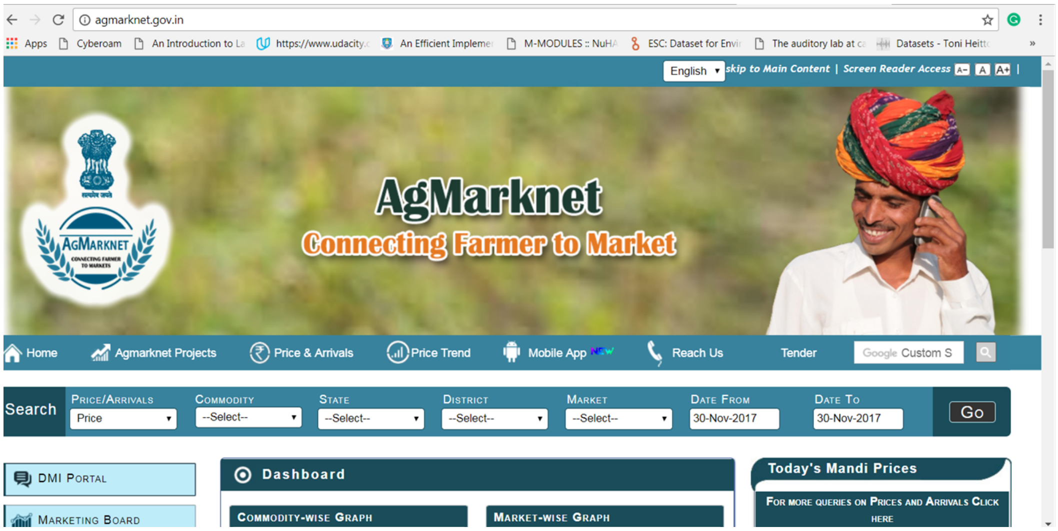

Weather information from web-portal
Weather Database
Price Database
Agricultural commodity prices from web-portal
Mobile call
Interactive Voice Response System (IVRS)
Automatic Speech Recognition (ASR)

Figure 5.1: Block diagram of a speech-based access system for an agricultural commodity. After [27].

The ASR system is one of the major components in a speech-based access system for an agricultural commodity. It is this component that identifies the required query, which is then passed to the IVRS. The success of the response to a farmer's call is based on the accuracy of the ASR system.

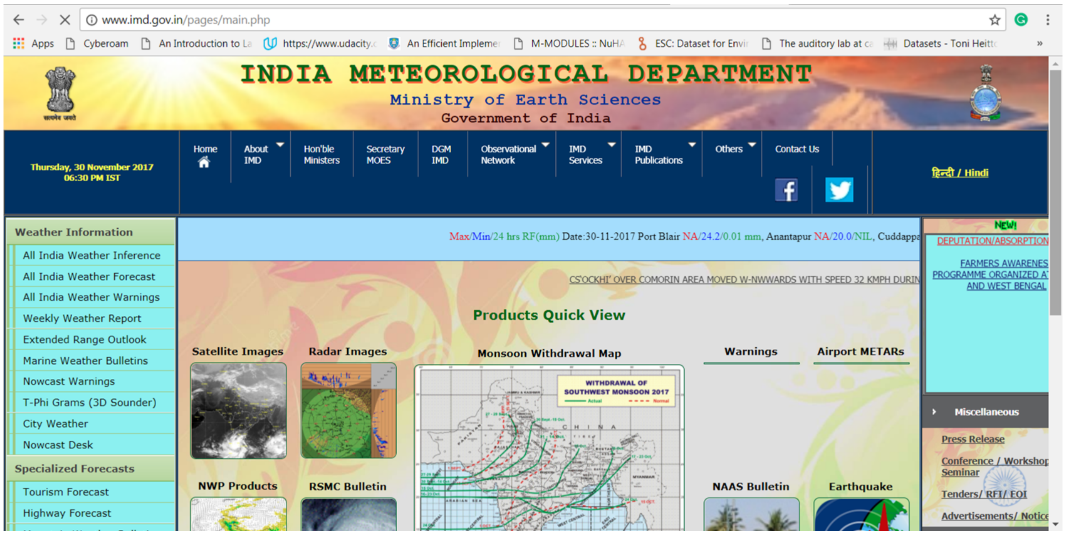
5.2.3 Data Collection

Data collection from the farmers in villages is an important task of this project work. The total number of districts in the Gujarat are 33 (out of which 27 regis-



Price/Arrivals	Commodity	State	District	Market	Date From	Date To
Price	--Select--	--Select--	--Select--	--Select--	30-Nov-2017	30-Nov-2017

Figure 5.2: The snapshot of the AGMARKNET. Adapted from [28].

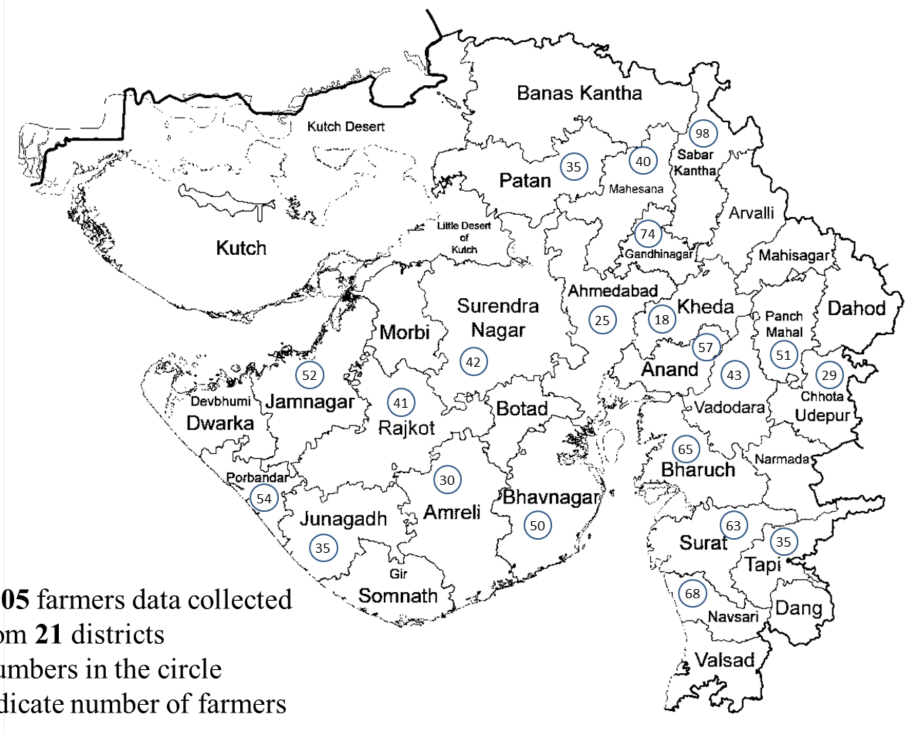


Max/Min/24 hrs RF(mm) Date:30-11-2017 Port Blair NA/24.2/0.01 mm, Anantapur NA/20.0/NIL, Cuddapa

CS'OCKHI' OVER COMORIN AREA MOVED W-NWWARDS WITH SPEED 32 KMPH DURIN

Figure 5.3: The snapshot of the IMD website. Adapted from [29].

Figure 5.4: The photographs of data collection in the villages of Gujarat state by the project staff members and volunteers. After [30].



- **1005** farmers data collected from **21** districts
- Numbers in the circle indicate number of farmers

Figure 5.5: Agricultural speech data collection from the Gujarat districts.

tered with AGMARKNET website). The major dialects observed in Gujarat state are: Gamadia (Ahmedabad, Vadodari), Kakari, Kathiyawadi (Saurastra), Kharwa, Parsi, Standard Gujarati, Tarimuki, and Surati (South Gujarat). The project staff members and volunteers of the Speech Research Lab at DA-IICT have gone to the several villages belonging to different dialectal regions of Gujarat state to collect the speech data. They have used mobile phones to record speech signals via a toll-free number (**079-30515300**). The snapshots taken during the field recordings by the project staff and volunteers are shown in Figure 5.4. The recording of speech signals is based on the Asterisk server configuration and prompts are stored in the server, located at Speech Research Lab, DA-IICT. The database includes the names of agricultural crops, mandi, weather information, and yes/no type of questions. The data collection includes natural speaking styles and dialects of the farmers with real environmental noises, such as vehicles, animals, babble, etc. The dataset also includes channel mismatch conditions since the recording was done with several mobiles of different companies (that include different microphones). The data collection from **1005 farmers** has been completed covering **21 districts** of Gujarat (namely, Gandhinagar, Sabarkantha, Mehesana, Patan, Kheda, Panchamahal, Surat, Navsari, Surendranagar, Anand, Vadodara, Bharuch, Rajkot, Chhota Udepur, Jamnagar, Porbander, Junagadh, Amreli, Bhavnagar, Tapi, Ahmedabad). The distribution of the collected database is shown in

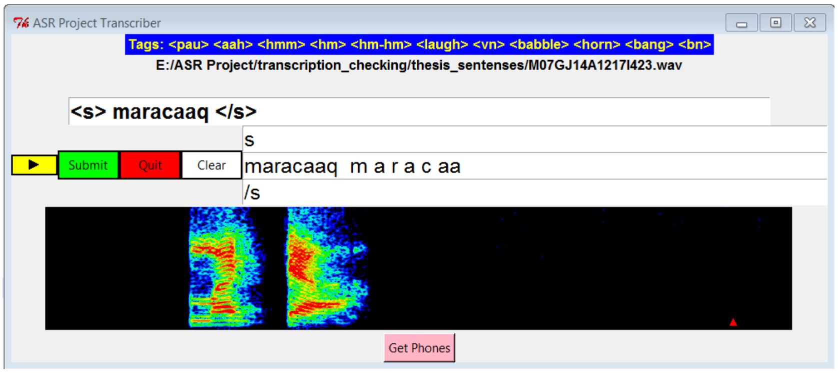
Label	IPA	Gujarati	Label	IPA	Gujarati	Label	IPA	Gujarati
a	/a/	અ	gh	/ɡʱ /	ઘ	dh	/d̪ʱ /	ધ
ax	/ɔ/	ઑ	ng	/ŋ /	ઙ	n	/n/	ન
aa	/aː /	આ	c	/tʃ/	ચ	p	/p /	પ
i	/ɪ /,/i/	ઇ	ch	/tʃʰ /	છ	ph	/pʰ /	ફ
ii	/iː /	ઈ	j	/dʒ/	જ	b	/b/	બ
u	/u/,/ʊ /	ઉ	jh	/dʒʱ /	ઝ	bh	/bʱ /	ભ
uu	/uː /	ઊ	nj	/ɲ/	ઞ	m	/m/	મ
ee	/eː /	એ	tx	/ʈ/	ટ	y	/j/	ય
ei	/ɛː /	ઐ	txh	/ʈʰ /	ઠ	r	/r/	ર
o	/o/	ઓ	dx	/ɖ /	ડ	l	/l/	લ
ae	/ae/	ઍ	dxh	/ɖʱ /	ઢ	lx	/ɭ/	ળ
ou	/oʊ /	ઔ	nx	/ɳ /	ણ	w	/ʋ /	વ
k	/k/	ક	t	/t̪/	ત	sh	/ʃ /	શ
kh	/kʰ /	ખ	th	/t̪ʰ /	થ	sx	/ʂ /	ષ
g	/g/	ગ	d	/d̪/	દ	s	/s /	સ
hq	-	◌ઃ	mq	-	◌ઁ	h	/ɦ /	હ

Figure 5.6: Transliteration for Gujarati UTF-8 to common phoneset. After [31].

Figure 5.5. There are a number of issues in data collection, such as explaining the farmers about the task, since they are hesitant to talk, and response to the IVRS, disfluencies in speech since many farmers are not much habituated to such mobile-based recording, field recordings, etc.

5.2.4 Transcription

We prepared the dictionary (along with transcription) containing the names of crops, mandis and districts. The Indian Language Speech sound Label set (ILSL) format has been used for transcription and dictionary preparation as shown in Figure 5.6 for the Gujarati language [31]. The dictionary contains different varieties of commodities, mandi, names of villages, and districts. There are 25 districts, 328 markets, and 159 unique commodities (excluding variations) in the lexicon. The lexicon contains 5387 words including varieties in commodities, speaking market names and yes/no utterances spoken in various dialectal manner from the farmers. Examples of dialect variations in speaking commodity names are shown in Table 5.1. The speech signals were transcribed using semi-supervised transcription tool, namely, "Indic Language Transliteration Tool", provided by the MeitY ASR Phase-II consortium. An example of the transcription tool is shown in Figure 5.7. The transcriber can play the audio to mark the labels. The spectrogram of the speech signal can also be seen in the tool, which helps the transcriber to mark the labels more correctly. After the transcription, validation of the dictionary is done via the same tool. The dictionary also shows variations in the spoken word in the transcription tool as shown in Figure 5.8, which helps transcribers to validate the transcription easily.



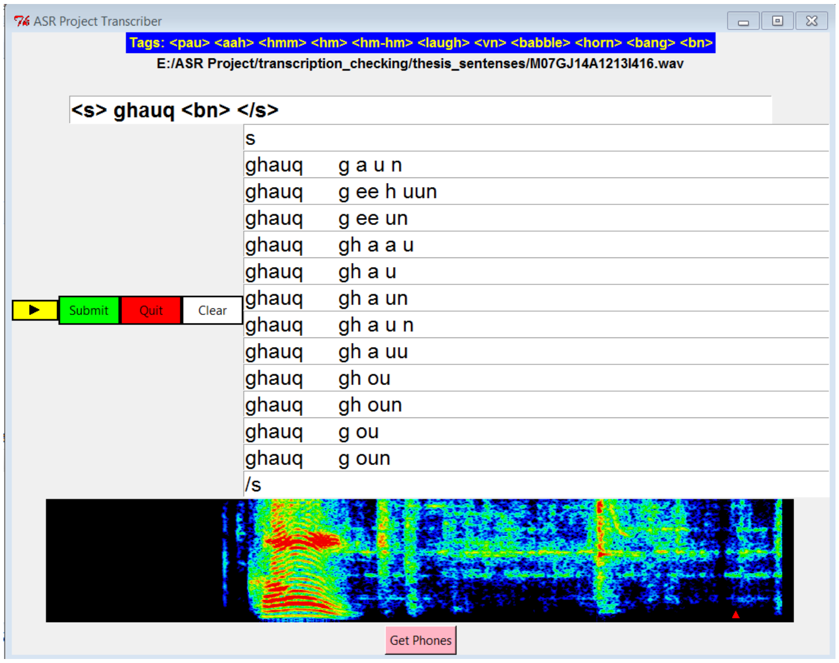
Tags: <pau> <aah> <hmm> <hm> <hm-hm> <laugh> <vn> <babble> <horn> <bang> <bn>

E:/ASR Project/transcription_checking/thesis_sentenses/M07GJ14A1217I423.wav

<s> maracaaq </s>

s
maracaaq m a r a c aa
/s

Figure 5.7: An example of a transcription tool developed by IIIT Hyderabad.



Tags: <pau> <aah> <hmm> <hm> <hm-hm> <laugh> <vn> <babble> <horn> <bang> <bn>

E:/ASR Project/transcription_checking/thesis_sentenses/M07GJ14A1213I416.wav

<s> ghauq <bn> </s>

s
ghauq g a u n
ghauq g ee h uun
ghauq g ee un
ghauq gh a a u
ghauq gh a u
ghauq gh a un
ghauq gh a u n
ghauq gh a uu
ghauq gh ou
ghauq gh oun
ghauq g ou
ghauq g oun
/s

Figure 5.8: An example of a validation tool developed by IIIT Hyderabad.

5.2.5 Analysis of Filterbank

The subband filters learned using ConvRBM are shown in Figure 5.9. The time-domain subband filters (weights of the model) are shown in Figure 5.9 (a)-(c) and corresponding frequency-domain subband filters (obtained by applying Fourier transform) are shown in Figure 5.9 (d)-(f). We can see that the model is able to learn auditory-like filters similar to the one reported in [2]. Some of the subband

Table 5.1: An example of a word presented in the dictionary with different phonetic pronunciation

Word	Phonetic description (spoken form)
batxaakaa	b a t aa k aa
batxaakaa	b a tx aa k aa
batxaakaa	b a tx aa k u
batxaakaa	b a tx aa tx aa
batxaakaa	b a tx ae k aa

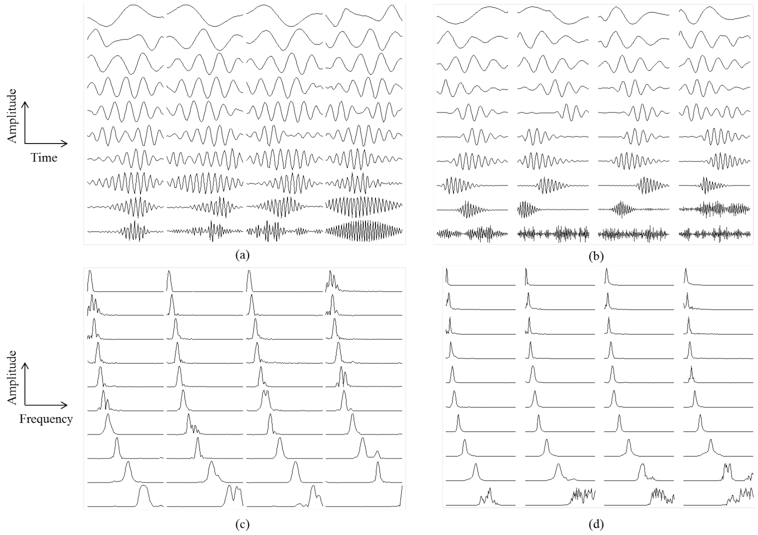


Figure 5.9: The subband filters trained on Gujarati ((a) and (c)) and the TIMIT database ((b) and (d)): (a)-(b) subband filters in time-domain, (c)-(d) subband filters in the frequency-domain.

filters are not localized compared to the filters trained on the TIMIT database. We would like to emphasize here an important point that speech signals for this project are recorded over PRI telephone line via mobile phone. Hence, the sampling frequency of speech is limited to 8 kHz. This may restrict the model to learn subband filters for high frequencies as we can see this difference in Figure 5.9 (a) and (b). We believe that the model may have learned Gujarati language-specific structures as well as the real environmental scenarios in speech signals, the analysis of which is an important open question for future research direction. The Mel filterbank and ConvRBM filterbank are compared in Figure 5.10. The speech signal is shown for a crop named “turmeric" (h lx d lx in English transliteration format) in the Gujarati language spoken along with babble noise as seen in Figure 5.10. We can see from the plots that formant structures are more clearly visible using the ConvRBM filterbank compared to the Mel filterbank (highlighted by the red circle). The babble noise is suppressed in the ConvRBM filterbank, while high

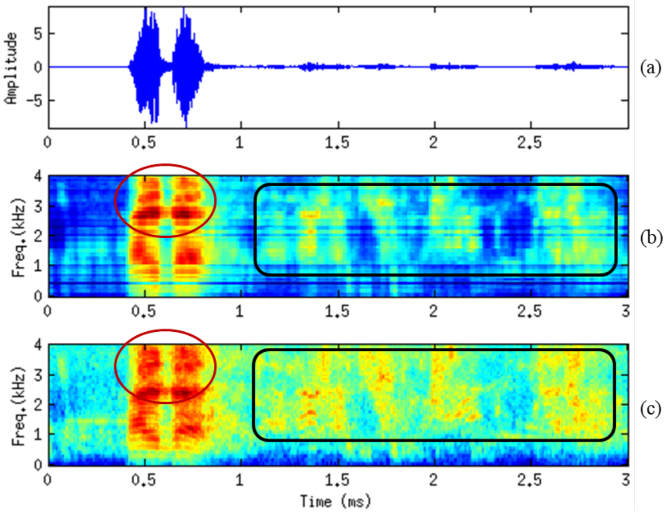


Figure 5.10: Comparison of filterbanks: (a) speech signal for the word “turmeric" (h lx d lx in English transliteration format) in the Gujarati language, (b) ConvRBM filterbank, and (c) Mel filterbank.

energy regions are still visible in the Mel filterbank as seen in rectangular regions of noise in Figure 5.10.

5.3 Experimental Setup

5.3.1 ConvRBM Training and Feature Extraction

The parameters of ConvRBM, learning rate, weight decay, etc. are similar as selected in Chapter 4. The ConvRBM is trained by 40 filters and an 8 ms convolution window length. To train the GMM-HMM systems, DCT-based 39-D ConvRBM-CC features are used. For the neural network training, 120-D ConvRBM-BANK features are used.

5.3.2 ASR System Building

The CD-GMM-HMM system was built from the MFCC feature set by varying the number of Gaussians and senones. We have used the finite state transducer (FST)-based LM (using the recipe provided by the IIT Madrass ASR team) trained from the agricultural commodity text data. We have applied linear discriminant analysis (LDA) after the GMM-HMM triphone system. LDA is used to reduce the dimensionality of the context-based cepstral features, e.g., context of 7 frames of MFCC with 39 coefficients (39×7=273) to 40-D using discriminative training. To decorrelate the features, the feature space transformation technique called Maximum Likelihood Linear Transform (MLLT) is applied on the LDA-based features. It is a feature orthogonalizing transform that makes the features more accurately modeled by diagonal-covariance Gaussians.The LDA-MLLT system was trained with the context of 7 frames on top of the CD-GMM-HMM system. The alignments obtained from the LDA-MLLT system were used in the hybrid DNN-HMM training in all the experiments. Here, we have explored the recently proposed Lattice-free Maximum Mutual Information (LF-MMI) in the HMM framework for sequence-to-sequence learning [178]. The sequence learning framework is later used for the hybrid DNN-HMM training in the KALDI toolkit. The LSTM-TDNN and BLSTM models were used for acoustic modeling. The LSTM-TDNN has 3 LSTM layers and TDNN layers (with a context of {-13,9}) in between while the BLSTM has three layers. The number of hidden units and layers in the BLSTM were chosen based on the % WER. The deep networks were trained using the ConvRBM filterbank and the Mel filterbank (denoted as FBANK). The systems were combined using the MBR decoding with the fusion factor, λ =0.5.

5.4 Experimental Results

Since there is no standard recipe for this ASR in Gujarati task, we varied the number of Gaussians and senones for GMM-HMM systems. The results are summarized in Table 5.2 for MFCC and ConvRBM-CC feature sets. Using the CD-GMM-HMM system, best results are obtained using 1800 senones and 12 Gaussians for both the feature sets. The ConvRBM-CC performs better than the MFCC using the CD-GMM-HMM system with an absolute reduction of 1.09 % in WER. The ConvRBM-CC also significantly performs better than the MFCC using the LDA-MLLT system with an absolute reduction of 1.53 % in WER. Hence, ConvRBM-CC improved the ASR performance compared to the baseline MFCC feature set using the CD-GMM-HMM and LDA-MLLT systems. The alignments generated from the respective LDA-MLLT systems (for MFCC and ConvRBM-CC) are used in the hybrid DNN-HMM systems.

The experiments on the hybrid DNN-HMM models are reported in Table 5.3. The ConvRBM-BANK perform better compared to the FBANK with both DNN models. Using the TDNN-LSTM models, the best results were achieved using 800 hidden units for the FBANK and 900 hidden units for the ConvRBM-BANK. There is an absolute reduction of 0.8 % in WER when the ConvRBM-BANK feature set is used in the TDNN-LSTM models. The performance of both feature sets improved when BLSTM models are used for acoustic modeling. However, due to increased

Table 5.2: The summary of results using GMM-HMM systems in % WER for 1005 speakers

Feature Set	Acoustic model	Test
MFCC	Triphone (CD-GMM-HMM)	30.36
ConvRBM-CC	Triphone (CD-GMM-HMM)	29.27
MFCC	LDA-MLLT	26.98
ConvRBM-CC	LDA-MLLT	**25.45**

Table 5.3: The experimental results using various parameters of hybrid DNN-HMM models in % WER for 1005 speakers

Feature Set	DNN Model	Hidden Units	Hidden Layers	Test
FBANK	TDNN-LSTM	700	7	21.74
FBANK	TDNN-LSTM	800	7	21.01
FBANK	TDNN-LSTM	900	7	21.42
ConvRBM-BANK	TDNN-LSTM	800	7	20.83
ConvRBM-BANK	TDNN-LSTM	800	7	20.42
ConvRBM-BANK	TDNN-LSTM	900	7	20.21
FBANK	BLSTM	700	3	21.28
FBANK	BLSTM	700	4	20.59
FBANK	BLSTM	800	3	21.11
FBANK	BLSTM	800	4	20.62
FBANK	BLSTM	900	3	20.28
FBANK	BLSTM	900	4	20.45
ConvRBM-BANK	BLSTM	700	3	20.90
ConvRBM-BANK	BLSTM	700	4	20.56
ConvRBM-BANK	BLSTM	800	3	**19.55**
ConvRBM-BANK	BLSTM	800	4	20.00
ConvRBM-BANK	BLSTM	900	3	20.10
ConvRBM-BANK	BLSTM	900	4	19.72

Table 5.4: The summary of results using hybrid DNN-HMM models along with the system combination in % WER for 1005 speakers

Feature Set	DNN Model	Test
FBANK	TDNN-LSTM	21.01
ConvRBM-BANK	TDNN-LSTM	20.21
A:FBANK	BLSTM	20.28
B:ConvRBM-BANK	BLSTM	19.55
A $\oplus$ B	BLSTM	**16.65**

complexity, increasing number of layers in BLSTM did not improve the results. The best results for BLSTM models were achieved using 3 layers and 900 hidden units for the ConvRBM-BANK feature set. There is an absolute reduction of 0.73 % in WER using ConvRBM-BANK compared to FBANK feature set using BLSTM models. The system combination of the ConvRBM-BANK and FBANK shows the significant reduction of 2.9 % in the WER.

Since the Gujarati ASR database is recorded in real environments, TEO-based auditory representation presented in Section 4.4, Chapter 4 is also used here. The TEO and ConvRBM-based auditory feature representation is denoted as TEO-ConvRBM-BANK in this Chapter. The experiments are performed using the similar LF-MMI and BLSTM models by varying the number of hidden units and hidden layers. The ASR results are shown for the optimal BLSTM parameters in Table 5.5. The TEO-ConvRBM-BANK perform better than FBANK and ConvRBM-BANK with a relative reduction of 5.8 % and 2.3 % in WER, respectively. Hence, the TEO along with ConvRBM indeed helps to supress real environmental noise in the speech recordings. The system combination of FBANK and TEO-ConvRBM-BANK (S1$\oplus$S3) performed significantly better than FBANK, ConvRBM-BANK, TEO-ConvRBM-BANK, and S1$\oplus$S2. Compared to FBANK and ConvRBM-BANK, S1$\oplus$S3 gives an absolute reduction of 4.45 % and 3.72 % in WER, respectively. The S1$\oplus$S3 also gives an absolute reduction of 0.82 % in WER over S1$\oplus$S2. This shows the strong complementary information of TEO-ConvRBM-BANK that is helpful in the ASR task. The final system combination of all the three feature sets S1$\oplus$S2$\oplus$S3 further reduce % WER.

Table 5.5: The summary of results using the TEO-based auditory representation along with the system combination in % WER for 1005 speakers

Feature Set	Hidden Units	Hidden Layers	Test
S1:FBANK	900	3	20.28
S2:ConvRBM-BANK	800	3	19.55
S3:TEO-ConvRBM-BANK	900	3	19.10
S1 $\oplus$ S2	-	-	16.65
S1 $\oplus$ S3	-	-	15.83
S1 $\oplus$ S2 $\oplus$ S3	-	-	**15.21**

5.5 Chapter Summary

In this Chapter, the first attempt of its kind for development of an ASR system for a speech-based access system for an agricultural commodity in Gujarati is presented. The data collection of farmers from the Gujarat state and transcription techniques are discussed. The ConvRBM is used as a front-end to learn features from the raw speech signals recorded from the realistic noisy scenarios. The ASR experiments using the TDNN-LSTM and BLSTM systems show that the proposed front-end provides lower % WER compared to the Mel filterbank feature set. In the next Chapter, we will discuss the application of ConvRBM in various audio classification tasks, such as environmental sound classification (ESC), spoof speech detection (SSD) task, and infant cry classification (ICC).

CHAPTER 6

Application to Audio Classification

6.1 Introduction

In the Chapters 3-5, we have discussed our proposed model of auditory filterbank learning and applied on speech signals for the ASR task. However, our auditory system is able to discriminate the variety of natural sounds, such as human speech *vs.* dog barking [19], [179]. In this Chapter, to explore the potential of our proposed model to represent different variety of natural sounds, we have considered the environmental sound classification (ESC) task in Section 6.2. The experimental setup for the ESC task is discussed in Section 6.3. The analysis of the ConvRBM filterbank presented in Section 6.4 revealed that optimal auditory codes to represent the environmental sounds are different from the speech signals. Section 6.5 discusses the study of the spoof speech detection (SSD) task. The experimental setup for the SSD task is given in Section 6.6. The analysis and results of the SSD task is presented in Section 6.7. This study also presents other insights of the synthetic speech signals in terms of the spectral and temporal content compared to the natural speech signals. In addition, ConvRBM is also applied in the replay SSD task discussed in Section 6.8-6.10. We have also applied our feature learning framework to the socially-relevant problem of infant cry classification (ICC) task presented in 6.11-6.14, where it is shown that our model is able to learn filterbank even from very small size of database.

6.2 Environmental Sound Classification (ESC)

Environmental sound classification is a growing research problem in multimedia applications. Environmental sounds are a very diverse group of everyday audio events that cannot be described as only speech or music [180]. Environmental sounds are important for understanding the content of the multimedia. Therefore, ESC technology development is better for characterizing the essential role of environmental sounds in many multimedia applications, such as audio scene classification [181], audio surveillance systems [182], hearing aids [183], smart room monitoring [184], and video content highlight generation [185], etc. The literature of using representation learning for the ESC task is given in Table 2.2, Chapter 2. In this Chapter, we propose to exploit ConvRBM as a front-end for filterbank learning from raw audio signals. Here, we have used an Adam optimization [163] along with an annealed dropout technique [162] as discussed in Chapter 5.

6.3 Experimental Setup for the ESC task

6.3.1 Database

We have used the publicly available ESC-50 database [180] for the ESC task. The ESC-50 consists of 2000 short (5 seconds) environmental recordings. These recordings are divided into 50 equally balanced classes. These 50 classes form five major groups, namely, animals, natural soundscapes and water sounds, human non-speech sounds, interior/domestic sounds and exterior/urban noises. The audio files are prearranged in 5-fold cross-validation format. Due to this reason, the results of the experiments can be directly compared to the baseline results and with the previous approaches.

6.3.2 Training of ConvRBM and Feature Extraction

We have trained ConvRBM with an annealed dropout using $P[0] = 0.3$, and $P[0] = 0.5$, that is decayed to zero during training (discussed in Section 4.2, Chapter 4). The learning rate was chosen to be 0.001, and decayed according to the learning rate schedule as suggested in [163]. The moment parameters of Adam optimization were chosen to be β_1=0.5 and β_2=0.999. We have trained the model with 60 subband filters (i.e., K) with different convolution window lengths (i.e., m=132, 176, 220 samples). The delta features were also appended resulting in two channels (60-dimensional each) for the CNN classifier.

6.3.3 CNN Classifier

The CNN classifier with the architecture as proposed in [32] was used for the ESC task. Earlier, feature extraction for the CNN classifier, we first pre-processed the audio signal. All the audio files were downsampled to 22.05 kHz. The audio files were divided into frames by using a 25-ms Hamming window with 50 % overlap.

Then, we applied a silence removal algorithm. For silence removal, we first check for more than three consecutive silence frames (approximately, 50 ms duration). If silence is present in more than three frames, then we remove the silence frames else we keep those frames. A Simple energy thresholding algorithm was used to remove the silence regions. The Mel Filterbank (FBANK) is used as the baseline features. We have also used an auditory inspired Gammatone filterbank. Short segments of 41 frames were used as the input to the CNN. The segments were extracted with 50 % overlap from the audio files.

Figure 6.1 shows the details of each layer in the CNN architecture that we have used in the ESC task. The network was implemented using Keras [186] with *theano* back-end. A mini-batch implementation with a 200 batch size was used to train the network. The network hyperparameters were similar to the one used in [32]. At the testing time, the class of the test audio files was decided using the probability prediction scheme [32]. The performance of the classifier was evaluated using % classification accuracy (defined in Appendix B.2). We have also done the score-level fusion of different feature sets as used in [187].

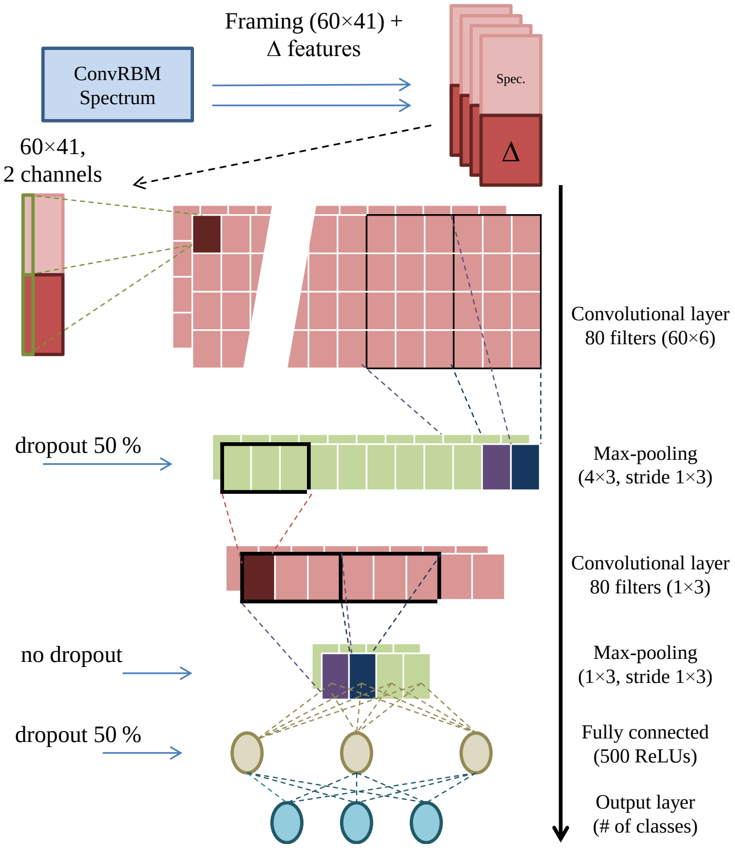


Figure 6.1: The CNN architecture for ESC task. After [5], [32], and [33].

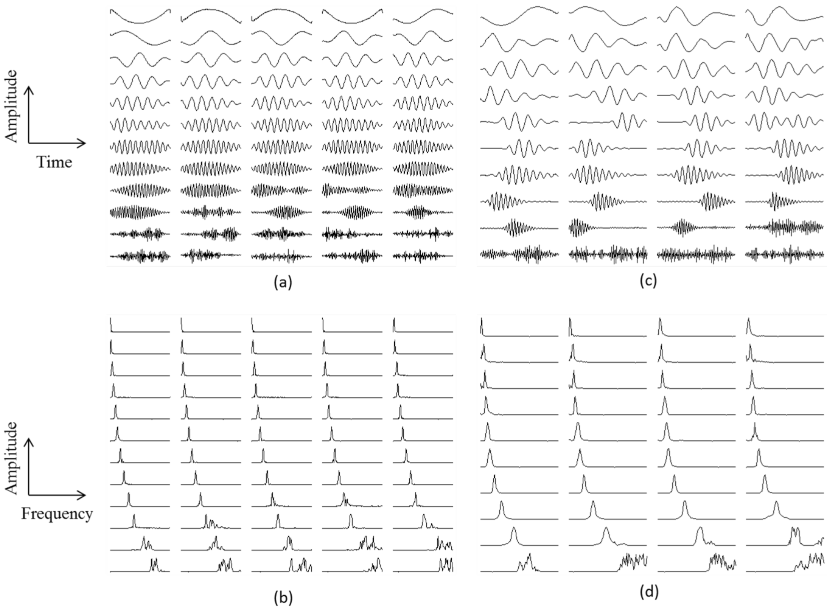


Figure 6.2: Filterbank learned using ESC-50 and TIMIT databases: (a), (c) subband filters in the time-domain (b), (d) corresponding frequency responses of subband filters in the frequency-domain. After [5].

6.4 Experimental Results of the ESC Task

In this Section, the filterbank learned using ConvRBM is analyzed first, followed by a discussion on the experimental results.

6.4.1 Analysis of Subband Filters

The impulse responses of the subband filters and their corresponding frequency responses are shown in Figure 6.2. It can be seen that many of the subband filters are Fourier-like basis functions that represent harmonic sounds, such as animal vocalizations. Lower frequency subband filters are gammatone-like basis functions. From Figure 6.2 (b), we can see that most of the subband filters are highly localized in the frequency-domain. The frequency responses of the higher frequency subband filters are not localized, which represent noise-like sound classes, such as rain, airplane, and thunderstorm. Similar insights have been discussed in [12], [115], [188], where the filterbanks were learned using an efficient coding principle. The work of [12], [115] analyze the filterbank on a separate database of the animal vocalizations and environmental sounds. Here, the ESC-50 database is a mixture of both of these categories. The subband filters are also different than the one we obtained, when ConvRBM is trained on the speech signals [2], [3] (as shown in Chapter 3, Section 3.5). The study in [189] also revealed that the auditory cortex regions that are sensitive to speech and other sounds are different. Hence, to completely characterize auditory processing, we should consider using statistics of speech and environmental sounds together. This approach can be helpful as the *transfer of knowledge* from the speech to the ESC tasks as done in [190].

The spectrogram obtained from the ConvRBM filterbank is shown in Figure 6.3 along with the Mel spectrograms using 60 subbands. The time-domain audio signals of the three environmental sounds category, namely, hen, rain, and mouse click are shown in Figure 6.3 (a)-(c). The ConvRBM spectrograms reveal time-frequency patterns similar to the Mel spectrograms. However, the intensity of the subbands are more dominant in the ConvRBM spectrograms than the Mel spectrograms (see, for example, rain and mouse click sounds in Figure 6.3).

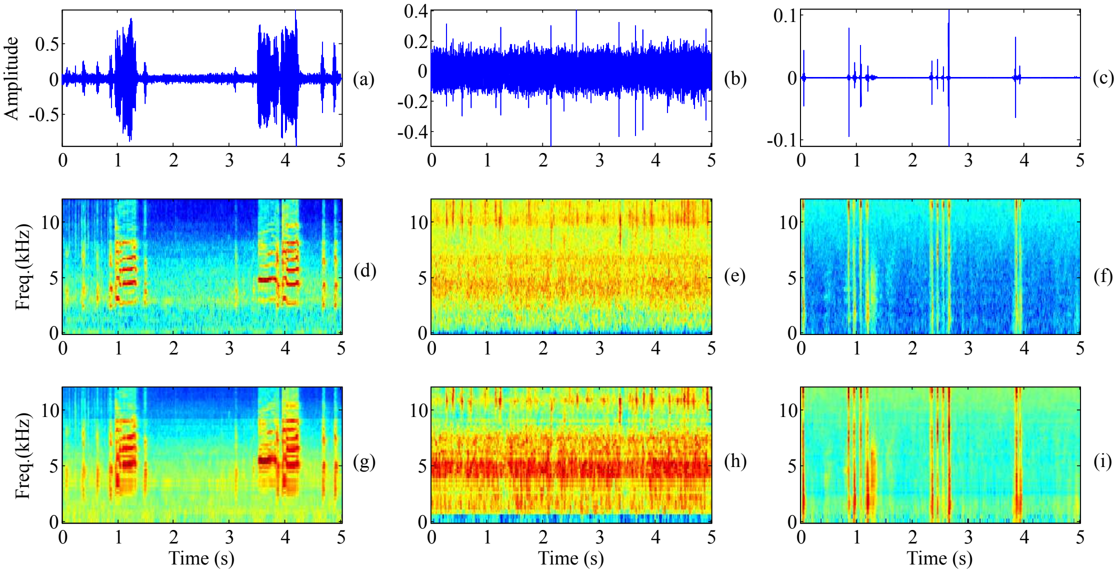


Figure 6.3: The examples of environmental sounds from the ESC-50: (a)-(c) audio signals, (d)-(f) Mel spectrograms, and (g)-(i) ConvRBM spectrograms.

6.4.2 Analysis of Filterbank Scale and Bandwidth

In order to compare the learned filterbank with the standard auditory filterbanks, we have shown a CF *vs.* subband filter index plot in Figure 6.4. We have also compared two ConvRBMs, the one that is trained using SGD, without dropout and the other that is trained using AD and Adam optimization. Both ConvRBM filterbanks have a nonlinear relationship between CF and filter ordering similar to the other auditory filterbanks. The ConvRBM trained with AD and Adam optimization uses more subband filters in the frequency range 1.5-8 kHz (similarly observed in [124], when using Adam optimization in DNN). Since the ESC-50 dataset contains harmonics, transients, and noise-like sound classes, the frequency scale learned is also similar, when ConvRBM is trained with the speech signals [3]. However, the shape of the subband filters are different compared to the speech signals [3], which indicates a different optimal auditory code for the environmental sounds. The scatter plot of bandwidth *vs*. CF is shown in Figure 6.5 for two ESC databases, namely ESC-50 and UrbanSound8k for better understanding and two ASR databases. The Q-factors for ESC sounds are different than speech sounds and did not preserve the constant-Q characteristics. However, the Q-factors still increasing as CF increasing. This may be due to the fact that ESC databases also contain mixture of harmonic, transient, and noisy sounds.

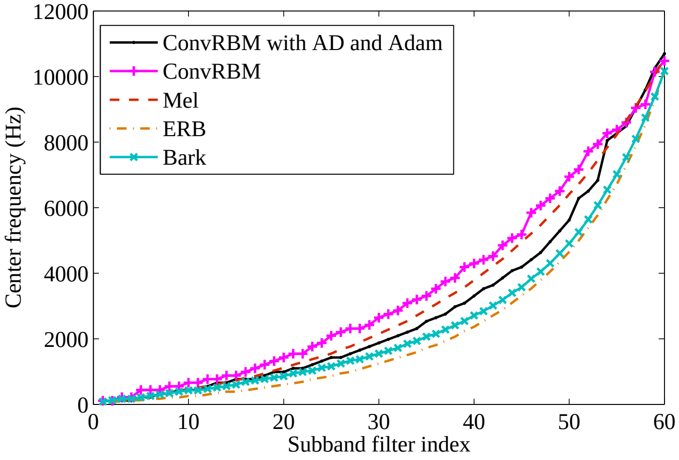


Figure 6.4: Comparison of ConvRBM filterbank and standard auditory filterbanks on the ESC-50 dataset. Here, AD represents annealed dropout. After [5].

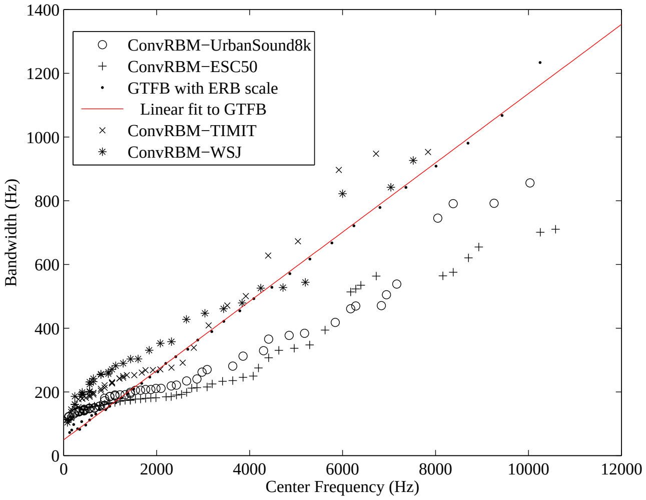


Figure 6.5: Comparison of bandwidth *vs*. CF for ESC and ASR databases.

6.4.3 Classification Results

To evaluate the performance of the proposed filterbank with different parameters, 5-fold cross-validation was performed on the ESC-50 database as shown in Table 6.1. We observed that a filter length of 132 samples (i.e., 6 ms) gave better performance. In all the cases, Adam optimization performed better than the stochastic gradient descent (SGD) in the ConvRBM training. From Table 6.1, it can be seen that the max-pooling with dropout significantly works better than the average-pooling in ConvRBM for the ESC task. This observation is different from what we observed in the ASR task in Section 3.7.1, Chapter 3, where average-pooling performed well [2]. We also performed the experiments with different dropout probabilities (P) for filterbank learning. The annealing dropout with probability 0.5 performed better than 0.3 with the same configurations of ConvRBM. Hence, we have selected ConvRBM with filter length of 132 samples, dropout probability of 0.5, Adam optimization, and max-pooling for rest of the experiments.

Table 6.1: % Classification accuracy using ConvRBM-BANK features with different tuning parameters. Here, m is the ConvRBM filter length and $P[0]$ is the annealed dropout probability. After [5]

m	Optimizer	$P[0]$	Pooling	Accuracy (%)
132	SGD	-	average	59.85
132	SGD	-	max	76.95
132	ADAM	-	average	66.55
132	ADAM	-	max	76.15
132	ADAM	0.3	average	67.45
132	ADAM	0.3	max	78.15
132	ADAM	0.5	max	**78.45**
176	ADAM	0.3	average	57.40
176	ADAM	0.3	max	74.90
176	ADAM	0.5	max	75.30
220	ADAM	0.5	max	73.25

We compared the performance of ConvRBM-BANK with FBANK and Gammatone Spectral Coefficients (GTSC). The overall results of the proposed method and baseline feature sets are summarized in Table 6.2 with the CNN classifier. ConvRBM-BANK performs significantly better than the FBANK with an absolute improvement of 10.65 % in the classification accuracy. The gammatone filterbank is inspired from auditory physiology [191], whereas ConvRBM filterbank is learned from the raw audio signals with the randomly initialized weights. Interestingly, it gives a comparable classification accuracy with GTSC (79.10 % *vs.*

78.45 %). The score-level fusion of the ConvRBM-BANK with the FBANK and GTSC improves the performance. However, the score-level fusion of ConvRBM-BANK (78.45 %) and FBANK (67.80 %) achieved the best accuracy of 86.50 % in this study. This shows that the proposed ConvRBM-BANK contains highly complementary information over the Mel filterbank, which is helpful in the ESC task.

Table 6.2: % Classification accuracy with different feature sets. Here, $\oplus$ and α indicate score-level fusion, and fusion factor, respectively. After [5]

Feature Sets	α	Accuracy (%)
FBANK	-	67.80
GTSC	-	79.10
ConvRBM-BANK	-	78.45
FBANK $\oplus$ ConvRBM-BANK	0.5	**86.50**
GTSC $\oplus$ ConvRBM-BANK	0.5	83.00

Our proposed work is also compared with other studies in the literature in Table 6.3. The ConvRBM-BANK performs significantly better than the CNN with FBANK [32], [106]. In [106], the filterbank is learned from the raw audio signal using the CNN as an end-to-end system. The EnvNET [106] performs better compared to the FBANK, when combining with log Mel CNN. However, our proposed ConvRBM-BANK outperforms EnvNET [106] even without the system combination. This shows the significance of unsupervised generative training using ConvRBM. In the next Section, we will present another audio classification problem of the SSD task.

Table 6.3: Comparison of classification accuracy of the ESC-50 dataset in the literature. The $\otimes$ sign indicated system combination before soft-max. After [5]

Feature Sets	Accuracy (%)
ConvRBM-BANK (proposed)	**78.45**
FBANK $\oplus$ ConvRBM-BANK (proposed)	**86.50**
Piczak FBANK-CNN [32]	64.50
Human [180]	81.30
EnvNET [106]	64.00
logmel-CNN [106]	66.5
logmel-CNN $\otimes$ EnvNet [106]	71.00

6.5 Spoof Speech Detection (SSD)

Automatic Speaker Verification (ASV) or voice biometrics is the task of verifying the claimed identity of a person from his or her voice with the help of machines [192]. However, practical ASV systems are vulnerable to the biometric attacks, also known as the voice presentation attacks, according to the ISO/IEC standard 30107-1:2016 [193]. The major voice attacks include voice conversion (VC) [194], speech synthesis (SS) [195], replay [196], and impersonation [197], which are known to degrade the performance of ASV systems [192]. Hence, a speaker verification system also includes the SSD system as a countermeasure along with the ASV system as shown in Figure 6.6. The general countermeasure approach is one of the solutions to focus on feature representation and statistical pattern recognition techniques. In particular, feature representation forms a key task for the SSD task. The aim is to distinguish between genuine and impostor speech by capturing the key discriminative features between two speech signals. This might suggest that the design of spoofing countermeasures should better focus on feature representation, rather than on advanced or complex classifiers [198, 199]. The details of various approaches used for the SSD task both for the ASVspoof 2015 challenge, and post evaluation results are given in [200]. The literature of using representation learning for the SSD task is given in Table 2.3, Chapter 2. In this Chapter, we describe our approach of the unsupervised filterbank learning using ConvRBM for the SSD task [6].

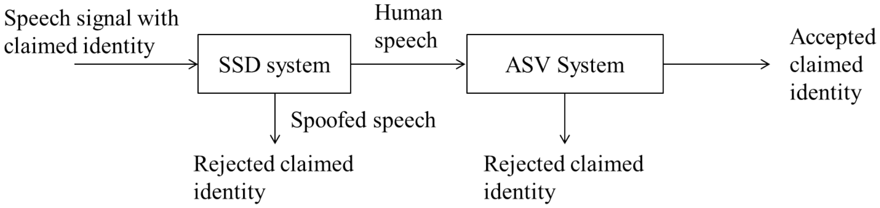


Figure 6.6: A speaker verification system along with the spoof speech detection system as a countermeasure.

6.6 Experimental Setup

6.6.1 ASVspoof 2015 Challenge Database

The experiments are conducted on the ASVspoof Challenge 2015 database [192]. It consists of speech data without channel or background noise collected from the 106 speakers (45 male and 61 female). It is divided into three subsets, namely, training, development, and evaluation set. The description of the database is given in Table 6.4.

Table 6.4: Number of speakers and utterances in different datasets

Subset	# Speakers		# Utterances	
	Male	Female	Genuine	Spoofed
Training	10	15	3750	12625
Development	15	20	3497	49875
Evaluation	20	26	9404	184000

6.6.2 Training of ConvRBM and Feature Extraction

The ConvRBM is trained on the training set of the ASVspoof Challenge 2015 database. Each speech signal after mean-variance normalization was applied to the ConvRBM. The filter length is chosen to be m=128 samples (i.e., 8 ms), similar to as in [3]. The learning rate was chosen to be 0.0001 and decayed at each epoch according to the learning rate scheduling as suggested in [163]. The moment parameters of an Adam optimization were chosen to be β_1=0.5 and β_2=0.999. We have trained the model with different numbers of ConvRBM filters, with average and max-pooling. After the model was trained, the ConvRBM-CC features were extracted from the speech signals.

6.6.3 Model Training and Score-Level Fusion

We used the GMM with 512 mixtures for modeling the two classes, in which the classes correspond to the genuine and impostor class in ASVspoof 2015 database. The GMMs are trained with the training set of the database. The use of a GMM classifier has been shown to perform best in the detection of genuine *vs.* impostor speech in the ASVspoof 2015 challenge [200]. Final scores are represented in terms of the log-likelihood ratio (LLR). The decision of the test speech being genuine or impostor is based on the LLR, i.e.,

$$LLR = \log \frac{p(\mathbf{X}|H_0)}{p(\mathbf{X}|H_1)}, \tag{6.1}$$

where $p(\mathbf{X}|H_0)$ and $p(\mathbf{X}|H_1)$ are the likelihood scores from the GMM for the genuine and impostor trials (with hypothesis H_0 and H_1), respectively, for feature vectors $\mathbf{X}$. To obtain the complementary information of the MFCC and ConvRBM-CC feature sets, we use their score-level fusion, i.e.,

$$LLR_{combine} = (1-\alpha)LLR_{feature1} + \alpha LLR_{feature2}, \tag{6.2}$$

where $LLR_{feature1}$ is the log-likelihood score of MFCC, and $LLR_{feature2}$ is the score for ConvRBM-CC, respectively. The weights of the scores are decided by the fusion parameter of α.

6.7 Experimental Results of the SSD Task

In this section, first we will analyze the subband filters of ConvRBM followed by the experimental results.

6.7.1 Analysis of Subband Filters

The ConvRBM is trained using the training set of the ASVspoof Challenge 2015 database [201]. Figure 6.7 shows the subband filters learned using ConvRBM trained on the entire training set (denoted as ConvRBM-TrainingAll), synthetic speech from the training set (denoted as ConvRBM-TrainingSyn) of the ASVspoof 2015 database and the TIMIT database (denoted as ConvRBM-TIMIT). The model is analyzed with $K = 40$ subband filters for all the cases. The subband filters in the time-domain as shown in Figure 6.2 (a)-(c), and the corresponding frequency responses are shown in Figure 6.2 (d)-(f). The subband filters of ConvRBM-TrainingAll and ConvRBM-TrainingSyn are different from the ConvRBM-TIMIT. However, the filterbanks of ConvRBM-TrainingAll, and ConvRBM-TrainingSyn includes more lower frequency subband filters with many of the subband filters being wavelet-like basis functions (e.g., Figure 6.7 (a), (b) shows the short duration impulses responses). We can also see that all the subband filters are localized in the frequency-domain with different CFs except in the synthetic speech case as shown in Figure 6.2 (e). These observations are also reflected in the ConvRBM spectrogram (as shown in Figure 6.8). The ConvRBM spectrogram shows more emphasis in the lower frequency subbands compared to the Mel spectrograms. The training set of the ASVspoof 2015 database contains 3750 utterances of natural speech and 12625 utterances of synthetic speech. Hence, the ConvRBM subband filters trained on the training set (that includes both the natural and synthetic speech) adapted more towards representing the synthetic speech signals (since the model is biased towards them). From the frequency responses of filters, we can see that it also limits the model to represent higher frequencies that are difficult to model in the synthetic speech signals, such as fricative and transient sounds.

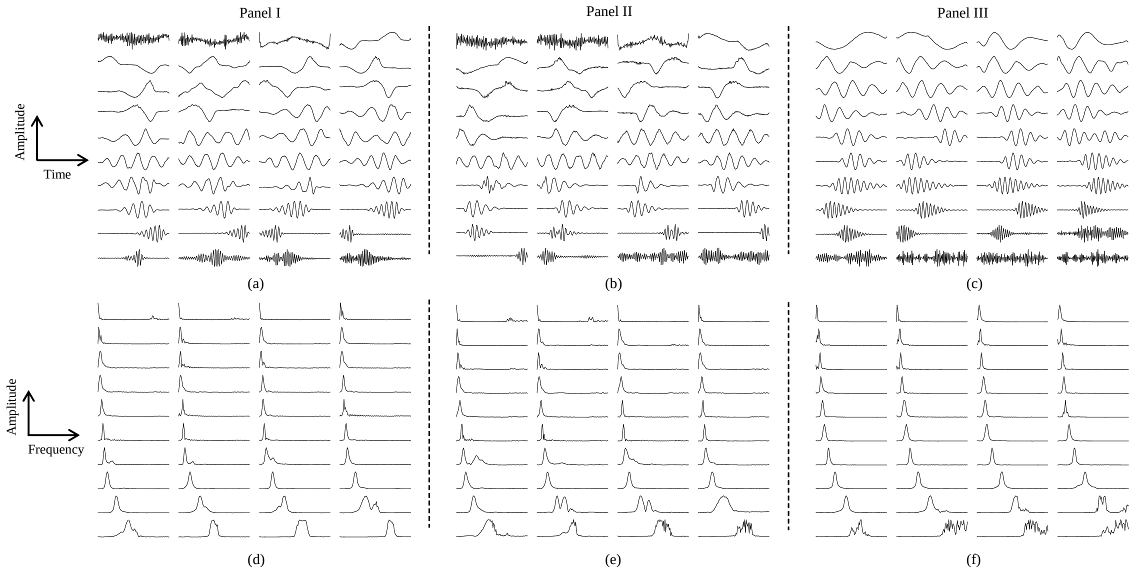


Figure 6.7: The subband filters trained on the training set of ASVspoof 2015 (Panel I), synthetic speech of ASVspoof 2015 (Panel II), and the TIMIT (Panel III) databases, respectively: subband filters in (a)-(c) time-domain, (d)-(f) frequency-domain.

6.7.2 Filterbank Scale Analysis

The CF *vs.* subband filter index plot is shown in Figure 6.9. The filterbank learned with ConvRBM-TrainingAll, ConvRBM-TrainingSyn, and ConvRBM-TrainingNat (natural speech) uses more lower-frequency subband filters compared to the rest of the filterbanks. The frequency scale of ConvRBM-TrainingNat is slightly different in the frequency range 1-3 kHz compared to ConvRBM-TrainingAll and ConvRBM-TrainingSyn. However, the frequency scales of ConvRBM-TrainingNat and ConvRBM-TIMIT alone are significantly different after 1 kHz. Since ConvRBM is a statistical model, it better learns the subband filters with more diverse databases and encodes the statistical properties of the underlying database (such as 462 speakers in the TIMIT database *vs.* 25 speakers in the training set). We also observe that the model is biased towards synthetic speech due to the large number of examples compared to the natural speech in the training set.

6.7.3 Experimental Results on Development Dataset

The results on the development set for the individual performance of the 39-D MFCC and ConvRBM-CC with different parameters (number of subband filters *K*,

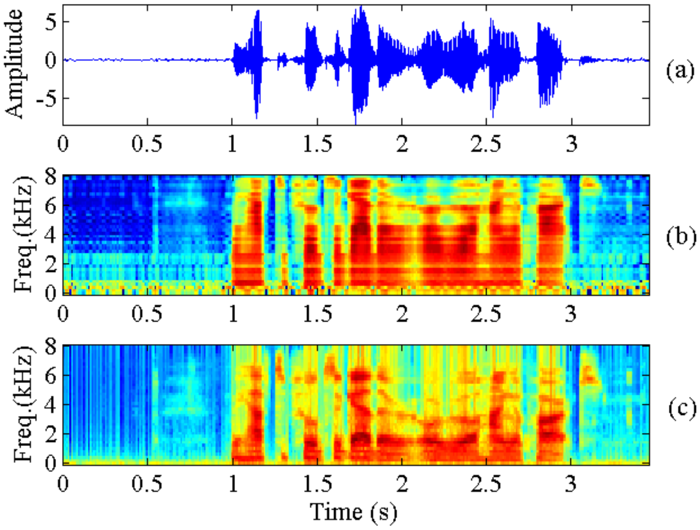


Figure 6.8: Spectrogram analysis: (a) speech signal, (b) ConvRBM spectrogram, and (c) Mel spectrogram. The utterance is: “We have to pull together or we will hang apart".

and pooling techniques) are shown in Table 6.5. It is observed that the ConvRBM-CC feature set (3.71-2.53 % EER) gives relatively better performance compared to the MFCC (6.14 % EER). However, an increasing number of subband filters (i.e., K=60) does not improve the performance of classification compared to the smaller number of subband filters (i.e., K=40). The lowest % EER is achieved using max-pooling and 40 subband filters. We have used 40 subband filters for rest of the experiments. The score-level fusion of ConvRBM-CC (K=40, average and max-pooling) with the MFCC further reduces % EER. This shows that the ConvRBM-CC feature set contains complementary information that was not evident from the MFCC alone. The DET curves of the two GMM systems using the MFCC and ConvRBM-CC feature sets, respectively, are shown in Figure 6.10.

Table 6.5: The results of different parameters of ConvRBM-CC features on the development set in % EER. After [6]

Feature Set	No. of Filters (K)	Pooling	% EER
MFCC	40	-	6.14
ConvRBM-CC	60	average	3.71
A:ConvRBM-CC	40	average	3.18
B:ConvRBM-CC	40	max	2.53
A⊕MFCC	40	-	2.80
B⊕MFCC	40	-	2.31

⊕ indicates score-level combination

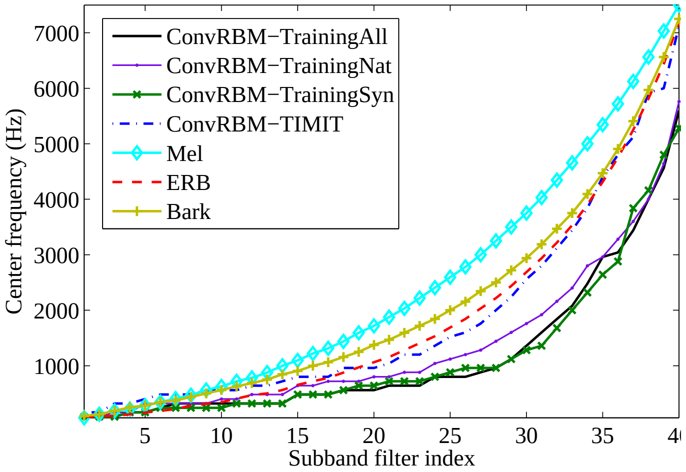


Figure 6.9: Comparison of the filterbank learned using ConvRBM with auditory filterbanks. After [6].

6.7.4 Experimental Results on the Evaluation Dataset

Table 6.6 shows the performance of the ConvRBM-CC feature set for each of the different spoofing attacks grouped into known and unknown attacks with their average EERs for known and unknown spoofing attacks. It is observed that ConvRBM-CC performs better than the MFCC on the evaluation set in all the spoofing attacks. The ConvRBM-CC with the max-pooling performs better than the average-pooling for known attacks (S1-S5). However, in the case of unknown attacks, specifically S10 (unit selection speech synthesis), the average-pooling performs better (22.64 %) than the max-pooling (33.20 % EER). Due to the dominance of S10 % EER, the average pooling gave the lowest EER of 5.87 % in unknown attacks compared to the max-pooling (7.69 %) and MFCC (13.76 %). Compared to the development set, ConvRBM-CC with the average-pooling performs better than the max-pooling on the evaluation set with the lowest EER 3.90 % on average. To observe whether any complementary information is being captured in the average and max-pooling of ConvRBM-CC, score-level fusion is performed. It resulted in the reduction of % EER in a few cases and increased % EER (including S10). We have also performed the score-level fusion of the MFCC and ConvRBM-CC with the average-pooling. Here, the fusion reduces the % EER for all the attacks except S2, S6 and S10 (which are having significantly higher % EER). Hence, the individual ConvRBM-CC with average pooling performed well in the SSD task. A comparison of the proposed feature representation with the literature (state-of-the-art as well as the feature learning methods) is shown in Table 6.7. Compared to the supervised features obtained using DNN with linear discriminant analysis (LDA) and GMM classifiers [130], [131], our unsupervised

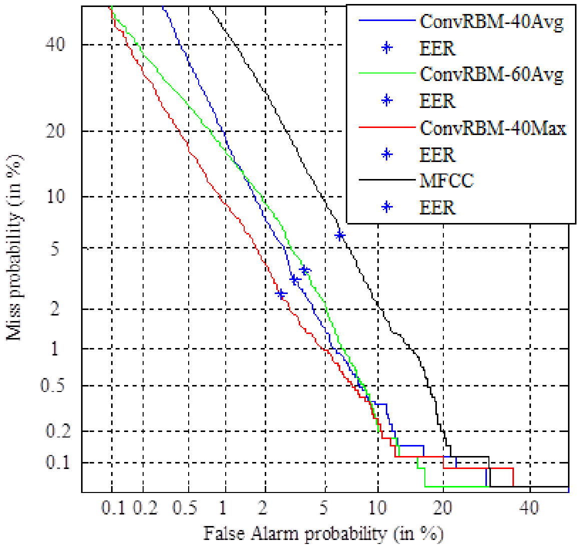


Figure 6.10: The DET curve of ConvRBM-CC (60 and 40 subband filters with max, and average pooling), and MFCC on the development set. After [6].

Table 6.6: Results on the evaluation dataset for each spoofing attack in terms of % EER. After [6]

Feature Set	Known Attacks						Unknown Attacks						All
	S1	S2	S3	S4	S5	Avg.	S6	S7	S8	S9	S10	Avg.	Avg.
MFCC	0.78	9.68	**0.00**	**0.00**	7.42	3.57	7.45	1.82	0.17	1.80	57.57	13.76	8.66
A:ConvRBM-CC (avg)	**0.00**	5.68	**0.00**	**0.00**	3.97	1.93	3.26	1.60	**0.00**	1.88	**22.64**	5.87	3.90
B:ConvRBM-CC (max)	**0.00**	**3.61**	**0.00**	**0.00**	2.82	1.26	2.15	1.69	**0.00**	1.45	33.20	7.69	4.47
A⊕B	0.35	3.70	0.16	0.21	**2.50**	**1.13**	**2.13**	1.13	**0.00**	1.20	24.49	**5.79**	**3.46**
MFCC⊕A	**0.00**	4.13	**0.00**	**0.00**	2.79	1.38	2.39	**0.68**	**0.00**	**0.00**	54.16	11.44	6.41

⊕ indicates score-level combination

filterbank learned using ConvRBM performs better in S10 class and similar % EER on an average in the unknown attacks. It also performs better than the supervised spectro/CNN [202] in S10 and resulted in similar % EER for unknown attacks. The CQCC feature set gave the lowest results achieved on the ASVspoof 2015 databases.

6.8 Replay Spoof Speech Detection

Among all the spoofing attacks, the replay attacks are a major threat to the ASV systems since they can be easily performed (using playback of recorded voice) [204]. A simple example is to use a device (either in smartphone or standalone) to replay a recording of a target speaker's speech to unlock a smartphone that uses the ASV-based access control [205]. To promote the research in development of countermeasures for the replay SSD, the ASVspoof 2017 Challenge was orga-

Table 6.7: Comparison of various feature sets in the SSD literature in terms of feature vector dimension (D), classifier, S10 class, unknown attacks and all the attacks. After [6]

Feature Set	D	Classifier	S10	Unknown	All
ConvRBM-CC	39	GMM	**22.64**	**5.87**	**3.90**
CQCC [203]	38	GMM	1.07	0.46	0.26
Best DNN [130]	96	LDA	25.5	5.1	2.6
Best RNN [130]	96	LDA	10.7	2.5	1.4
DMCC-BNF [131]	64	GMM	21.47	-	2.15
DPSCC-DNN [131]	60	DNN	12.86	-	2.18
Spectro/CNN [202]	128	CNN	26.83	5.83	3.07
Spectro/RNN [202]	128	RNN	17.97	4.05	2.46
Spectro/CNN+RNN [202]	128	RNN	14.27	3.33	1.86

nized as a part of a special session at INTERSPEECH 2017 [205]. The goal of the challenge is to develop replay SSD using the only acoustic characteristics of the utterances [205]. The replayed speech may contain unknown background noises, reverberation, channel noise, etc. In addition, the recordings made from very high-quality devices may also be close to the genuine speech.

It is observed in the literature that low frequency information is useful to detect synthetic speech. As discussed above, our ConvRBM learns more lower frequency subband filters naturally. In the replay speech detection, high frequency information is important [206], [207]. Hence, in this task, we used pre-emphasized speech signals to train the ConvRBM as discussed below:

6.8.1 Filterbank Learning From Pre-emphasized Speech

In order to learn subband filters that are more localized towards the high frequency components, we used pre-emphasized speech signals to train ConvRBM. The speech spectrogram (STFT and Mel) shows large intensity for lower frequency regions below 1 kHz. Perceptual experiments suggest that some aspects of the weak energy in the high frequency range is also important [153]. In particular, the center frequencies of the second and third formants (i.e., F_2 and F_3) of the sonorant sounds or correspondingly the lowest resonance of the obstruent sounds are very important and must be modeled well in speech analysis [153]. The pre-emphasis is a technique for flattening the magnitude spectrum and balancing the low and high frequency components. The pre-emphasis also models the combined effect of the glottal flow waveform and the lip radiation. It is mathematically modeled using a smooth highpass filter given as:

$$H(z) = 1 - \alpha z^{-1}, \tag{6.3}$$

where $\alpha = 0.97$ [153]. We observed in Section 6.10 that the subband filters learned in this way represent the high frequency components much better.

6.8.2 Feature Normalization

The mismatch between training and testing data is a major source of error for many speech processing applications including ASV [208]. The mismatch conditions include differences in background noises, communication channel, recording equipment, etc. In the case of the replay SSD task, mismatch includes different replay conditions for the training, development, and evaluation data [205]. A simple technique to reduce the channel mismatch is the cepstral mean normalization (CMN) [208], also known as the cepstral mean subtraction (CMS) [209]. The basic principle behind CMN is based upon the behavior of the cepstrum under the convolutional distortions [210]. The assumption is that the channel impulse response $h[n]$ is linear time-invariant, i.e., it does not vary significantly over the duration of the utterance [210].

The channel distortion $h[n]$ may also include the distortions due to recording/playback device ($h_1[n]$) and communication channel ($h_2[n]$), i.e., $h[n] = h_1[n] * h_2[n]$ [211]. Let us denote the speech signal prior to the channel distortion $h[n]$ as $x[n]$, then the corrupted speech signal is $y[n]$ given by:

$$y[n] = x[n] * h[n], \tag{6.4}$$

where $*$ denotes the convolution operation. The convolution in the time-domain becomes additive in the cepstral-domain [209]. Hence, eq. (6.4) can be written as [210]:

$$\mathbf{c}_y = \mathbf{c}_x + \mathbf{c}_h, \tag{6.5}$$

where $\mathbf{c}_y, \mathbf{c}_x$ and $\mathbf{c}_h$, denotes the cepstrum representation of eq. (6.4). Now, consider taking the mean of the cepstrum in eq. (6.5),

$$\mathbb{E}[\mathbf{c}_y] = \mathbb{E}[\mathbf{c}_x] + \mathbb{E}[\mathbf{c}_h], \tag{6.6}$$

where $\mathbb{E}[\cdot]$ is the expected value and can be calculated as sample average. Since it is assumed that the channel does not vary over the duration of an utterance,

$\mathbb{E}[\mathbf{c}_h] = \mathbf{c}_h$. If the distribution and variety of sounds in $x[n]$ is such that the average spectrum over the utterance is relatively flat, then $\mathbb{E}[\mathbf{c}_x] \approx 0$. As we discussed in Section 6.8.1, the pre-emphasis produces a flat spectrum and hence, $\mathbb{E}[\mathbf{c}_y] = \mathbf{c}_h$. Thus, we can reduce the channel distortion by subtracting the cepstral mean $\mathbb{E}[\mathbf{c}_y]$ from the cepstra of a corrupted speech signal in eq. (6.5) [210], i.e.,

$$\mathbf{c}_y = \mathbf{c}_y - \mathbb{E}[\mathbf{c}_y], \tag{6.7}$$

$$= \mathbf{c}_x. \tag{6.8}$$

It is also known that normalizing the variance of the cepstrum known as cepstral mean variance normalization (CMVN) further helps in adverse conditions [208]. However, when we remove all the cues from the replayed speech apart from the channel mismatch, then it becomes difficult to distinguish from the natural speech. Our experiments (in Section 6.10) show that CMVN degrades the performance severely. CMVN is not directly related to reduce the channel distortion and the reason for robustness is not well understood. Hence, we only use the CMN as a feature normalization technique. Compared to our earlier feature representation using ConvRBM, here we used pre-emphasis as pre-processing on the speech signals and CMN as post-processing on the frame-level ConvRBM-CC feature set.

6.9 Experimental Setup for Replay SSD

6.9.1 ASVspoof 2017 Challenge Database

The ASVspoof 2017 Challenge database is based on the RedDots corpus and its replayed speech, which is a text-dependent database [212]. The spoofed data were recorded through a variety of different environments in the ongoing H2020-funded OCTAVE project2 [205]. The RedDots corpus was developed through different replay configurations consisting of varied playback devices, recording devices, and loudspeakers. The number of speakers and utterances in each subset are summarized in Table 6.8.

In contrast to the training and development sets, spoofed data in the evaluation set are generated in accordance with the intentionally recorded and replayed in different unseen environmental and channel conditions to encourage the research towards the generalized spoofing countermeasures. Only some of the replay conditions are the same as those in the development and training sets.

Table 6.8: Statistics of the ASV Spoof 2017 challenge corpus

Subset	No. of Speakers	No. of Utterances	
		Genuine	Spoofed
Training	10	1508	1508
Development	8	760	950
Evaluation	24	1298	12922
Total	42	3566	15380

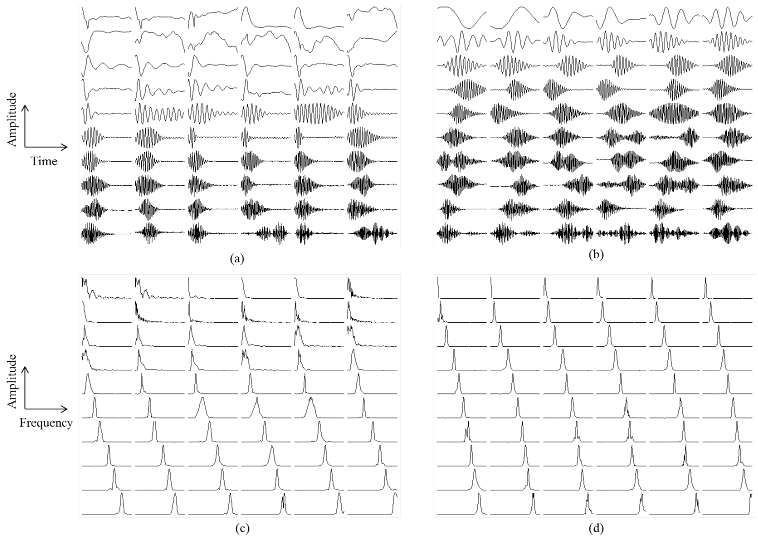


Figure 6.11: The subband filters trained on the training set of ASVspoof 2017 Challenge: (a) and (c) without and with pre-emphasis speech signals, respectively, (b) and (d) are the corresponding frequency responses. After [7].

6.9.2 Feature Extraction and Classifier

The speech signals are pre-emphasized prior to applying to ConvRBM followed by mean-zero, unit variance normalization. The rest of the parameters of ConvRBM are similar to those used in the synthetic spoof speech detection. The CMN is applied on the ConvRBM-CC feature set. The number of subband filters and dimension of the cepstral coefficients and type of pooling are decided by the performance of ConvRBM-CC features in the SSD task. The GMM classifier with the same parameters as used in synthetic SSD task are used here. The baseline GMM system was built using the CQCC features (90-D) with implementation provided by the ASVspoof 2017 Challenge organizers. Another GMM baseline system with CQCC was built by applying the pre-emphasis on speech signals and feature normalization using CMN.

6.9.3 Analysis of the ConvRBM Filterbank

The subband filters learned from the ASVspoof 2017 Challenge training database are shown in Figure 6.11. We have compared the subband filters trained with (denoted as ConvRBM-PEtraining) and without pre-emphasized speech signals. The difference between learned subband filters can be clearly seen in Figure 6.11. The filterbank learned without pre-emphasized speech signals contains many irregular low-frequency subband filters (Figure 6.11 (a)). Since the pre-emphasis increases the intensity of high frequency components, the filterbank learned with the pre-emphasized speech signals contains relatively few lower frequency filters, while many subband filters represent relatively high frequency components in the spectrum (Figure 6.11 (c)). The frequency responses of the subband filters in Figure 6.11 (b) and (d) show that ConvRBM trained on pre-emphasized speech signals learned more localized subband filters.

A comparison of frequency scales obtained for the pre-emphasized speech signals from the ASVspoof 2017 Challenge database is shown in Figure 6.12. The frequency scale obtained from the ConvRBM-PEtraining model is significantly different from other auditory scales as well as ConvRBM trained without pre-emphasized speech signals. Since the pre-emphasis performs flattening of the spectrum, the frequency scale obtained is more linear compared to the nonlinear scale obtained without pre-emphasis. It uses progressively more subband filters to represent higher frequencies. To represent the frequencies above 2 kHz, the ConvRBM-PEtraining model uses double the number of subband filters (45 *vs.* 20) compared to the other auditory scales and ConvRBM trained without pre-emphasized speech signals.

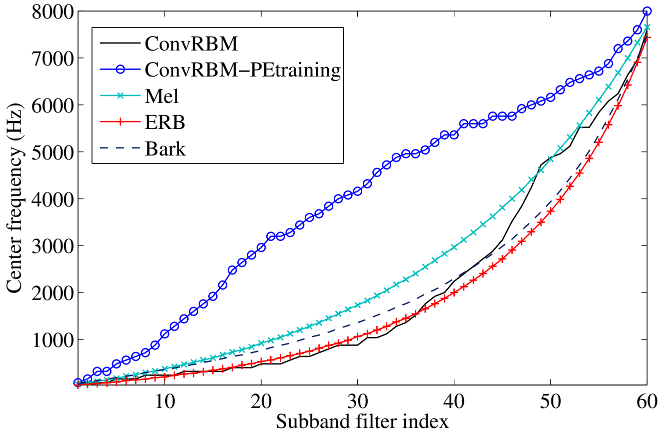


Figure 6.12: The comparison of ConvRBM filterbank scales (with and without pre-emphasized speech signals) for the ASVspoof 2017 Challenge database along with different auditory frequency scales. After [7].

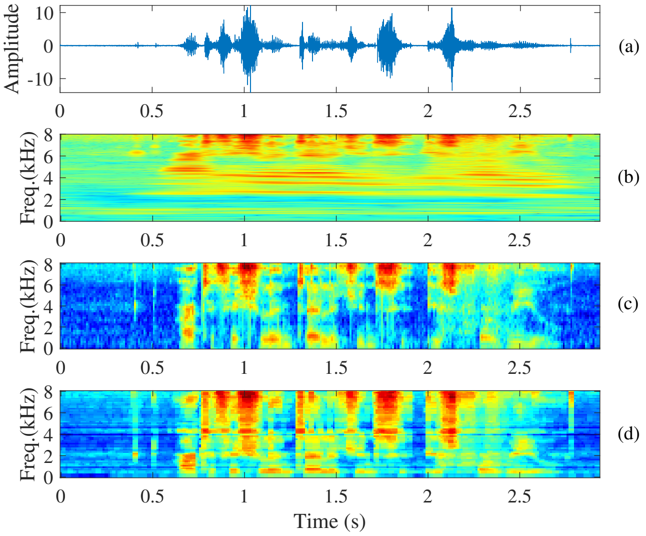


Figure 6.13: Comparison of spectrograms: (a) pre-emphasized speech signal, (b) constant-Q spectrogram, (c) Mel spectrogram, and (d) ConvRBM-PEtraining spectrogram. The utterance is: “Artificial intelligence is for real".

The spectrogram representation of the proposed filterbank is shown in Figure 6.13. We have also shown the constant-Q spectrogram (from the CQCC feature representation) and the Mel spectrogram in Figure 6.13. The ConvRBM spectrogram is obtained using 60 subband filters followed by the max-pooling operation. The constant-Q spectrogram has several subband filters representing the low frequencies. Hence, when the pre-emphasized speech signal is used in the constant-Q spectrogram, the intensities of the lower frequency components reduced significantly as can be seen from Figure 6.13 (b). The difference (w.r.t. the time-frequency representation) between the Mel and ConvRBM spectrogram is due to the frequency scale learned by the ConvRBM using pre-emphasized speech signals. The higher frequency components are more clearly visible than the Mel spectrograms. Since Mel filterbank filters are placed according to the Mel scale, it will not represent high frequency components even if we use the pre-emphasized speech signals (Figure 6.13 (b)). The ConvRBM subband filters are optimized in such a way that it represents the frequency range of the pre-emphasized speech signals.

6.10 Experimental Results

The effect of ConvRBM parameters in replay detection is shown in Table 6.9 on the development set in terms of % EER. It is observed from the previous studies that the ASVspoof 2017 task requires more number of cepstral coefficients. Hence, we compare different ConvRBM configurations with the 90-D feature vector (30 (static)+30 (Δ)+30 ($\Delta\Delta$)). The 90-D was chosen to compare the results with baseline 90-D CQCC features. The ConvRBM-CC obtained from the pre-emphasized filterbank resulted in an improved performance compared to the ConvRBM-CC feature set without a pre-emphasized filterbank. The use of NLReLU as hidden unit activations in ConvRBM improved the performance even further along with pre-emphasis (2.2 % absolute reduction). The significance of max-pooling can be seen in Table 6.9 compared to the average pooling (0.91 % absolute reduction in EER) for $K = 60$ and $m = 128$. Reducing the number of filters from $K = 60$ to $K = 40$ and increasing to $K = 80$ did not help in reduction of % EER. The filter length $m = 160$ and $m = 96$ also did not result in an improved performance. The lowest EER obtained is 9.48 % using $K = 60$, $m = 128$ and max-pooling with ConvRBM-CC features obtained from the ConvRBM-PEtraining.

Table 6.9: The analysis of various parameterization of ConvRBM-CC (90-D) in terms of % EER on the development set (Dev). Here, K and m represent the number of subband filters and filter length, respectively. After [7].

K	m	Pre-emphasis	Hidden units	Pooling	Dev
60	128	No	NReLU	avg	14.89
60	128	No	NReLU	max	12.39
60	128	Yes	NReLU	max	11.68
60	128	Yes	NLReLU	max	**9.48**
60	128	Yes	NLReLU	avg	9.70
40	128	Yes	NLReLU	max	11.93
80	128	Yes	NLReLU	max	9.80
60	160	Yes	NLReLU	max	11.46
60	96	Yes	NLReLU	max	12.42

avg= average pooling, max= max pooling, Dim= dimension of feature vector

Further analysis in terms of the pre-emphasis and feature normalization for the proposed ConvRBM-CC feature set along with CQCC is shown in Table 6.10. The pre-emphasis on the CQCC slightly reduces the % EER. However, applying the CMN further drops % EER to 9.61 % (absolute reduction of 0.74 %). Applying pre-emphasis and the CMN together to the CQCC reduces the EER from 10.35 % to 9.61 %. The feature normalization using the CMN significantly reduce % EER (from 12.39 to 9.86 %) even without using the ConvRBM-PEtraining feature set. Using only the ConvRBM-PEtraining feature set improves the performance in ConvRBM-CC with 0.69 % absolute reduction in % EER. However, the lowest EER of **8.79** % on the development set was achieved using the pre-emphasis

Table 6.10: The effect of pre-emphasis and feature normalization on CQCC (90-D) and ConvRBM-CC (90-D) in % EER on the development set. After [7].

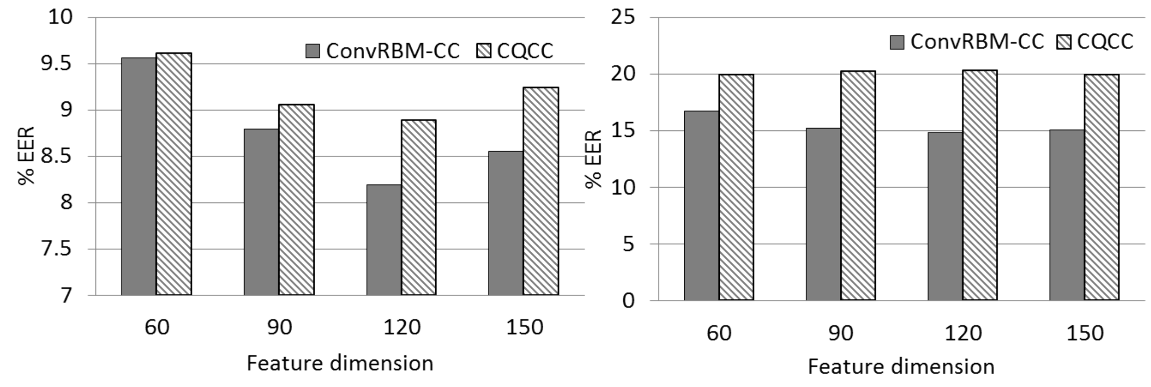
Feature Set	Pre-emphasis	Normalization	Dev
CQCC	No	No	10.35
CQCC	Yes	No	10.28
CQCC	Yes	CMN	9.61
ConvRBM-CC	No	No	12.39
ConvRBM-CC	No	CMN	9.86
ConvRBM-CC	Yes	No	9.48
ConvRBM-CC	Yes	CMN	**8.79**
ConvRBM-CC	Yes	CMVN	13.94

CMN=cepstral mean normalization,
CMVN=cepstral mean variance normalization

trained ConvRBM-CC along with the CMN. The CMVN did not perform well and increases the % EER. Hence, for our proposed feature set pre-emphasis trained ConvRBM and CMN resulted in an improved performance on the development set.

The effects of feature dimension in the ConvRBM-CC and CQCC are shown in Figure 6.14a and Figure 6.14b for the development and evaluation set, respectively. For different feature dimensions, the ConvRBM-CC feature set performs better than the CQCC feature set on both the development and evaluation set. In the case of the development set, 120-D feature dimension is optimal for both the CQCC and ConvRBM-CC, with the later performing relatively better. An absolute improvement of 0.7 % is obtained on the development set using the ConvRBM-CC over CQCC. In the case of the evaluation set, there is not much difference in % EER for CQCC feature set with minimum % EER for 60-D and 150-D. For ConvRBM-CC,better performance is obtained with 120-D, similar to the development set. ConvRBM-CC performs significantly better compared to the CQCC on the evaluation set with 5.07 % absolute reduction in % EER.

The final results using the proposed feature representation are summarized in Table 6.11. The baseline system using CQCC provided by the ASVspoof 2017 Challenge organizer has much higher % EER on the evaluation set compared to the baseline system we built using the CQCC feature set with pre-emphasis and CMN. In order to see the complementary information from CQCC and ConvRBM-CC, the score-level fusion is employed using eq. (5). The fused scores show significant reduction in % EER on the development set (5.90 %). However, for the evaluation set the reduction in % EER is not much significant. The performance analysis is also shown by the DET curve in the Figure 6.15a for the development



(a) (b)

Figure 6.14: The effect of feature dimension on (a) the development set and (b) evaluation set in % EER.

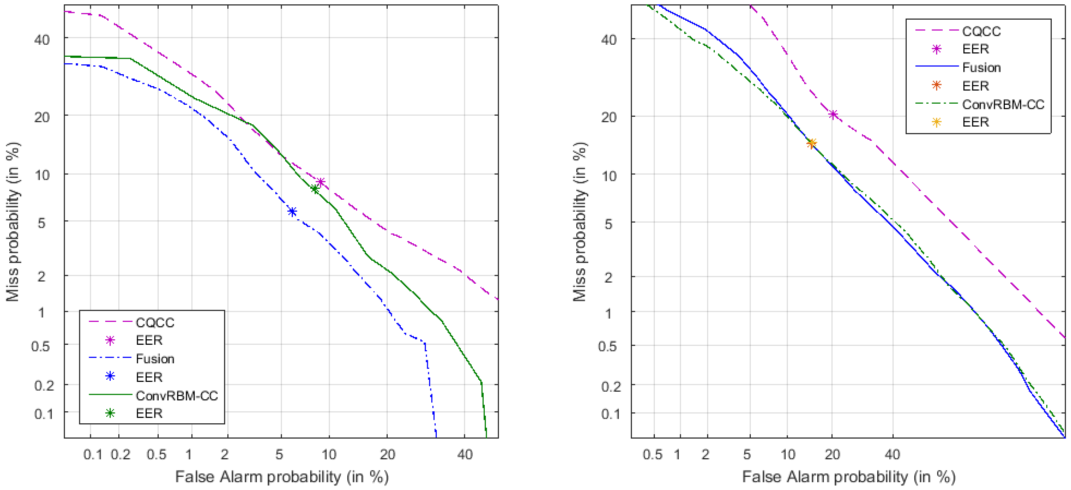
set and in the Figure 6.15b for the evaluation set. As seen from Figure 6.15a, except for a few operating points of the DET curve, ConvRBM-CC has lower false alarms and miss probabilities in the DET curve. The DET curve of score-level fusion is clearly distinct at *all* the operating points of the DET curve for the development set. For the evaluation set, the ConvRBM-CC features have significantly lower false alarms and miss probabilities in the DET curve compared to the CQCC. However, the score-level fusion does not have much benefit and also can be seen from the DET curve.

Table 6.11: The results on development and evaluation sets in % EER. After [7].

Feature Set	Dev	Eval
Challenge baseline [205]	10.35	30.17
S1:CQCC	9.61	19.93
S2:ConvRBM-CC	**8.19**	**14.86**
S1 $\oplus$ S2	**5.90**	**14.56**

$\oplus$ represents the score-level combination

The proposed feature set is compared to various feature sets proposed in the ASVspoof 2017 Challenge that was published in the special session during INTERSPEECH 2017. Our proposed feature set performs better on the evaluation set than the existing features based on using the high frequency information (HFCC and CQCC (i.e., for 6-8 kHz)). Most of the approaches used a GMM-based classifier. Few approaches used a deep learning framework either for classification or for feature learning followed by the GMM classifier [133]. Our proposed feature set resulted in lower % EER compared to the few approaches that use deep learning-based complex architectures except the work in [133]. The best results



(a) (b)

Figure 6.15: The DET curve for ConvRBM-CC, CQCC and their score-level fusion on (a) development set, and (b) evaluation set. After [7].

on the development set were achieved by the study [213] that incorporated the pre-emphasis technique. On the evaluation set, the system in [133] performed the best with the lowest 7.37 % EER that used feature normalization in both spectrum and cepstral-domain.

In the next section, we discuss classification of healthy *vs*. pathology infant cry classification.

6.11 Infant Cry Classification (ICC)

Humans cry to express a range and degree of emotions, such as from happiness after passing a tough exam or meeting a beloved one to grief after the death of a person or difficult situations in life [218]. On the whole, crying is not just a simple reaction to any feeling or emotional state but rather a multifaceted behavior that can offer clues to how we process and regulate our feelings, and how we experience the world around us [218]. The evolutionary background of crying is discussed in a book [219], where it is shown that only humans have the ability to cry, not other mammals. In humans, infants communicate their need, such as feeding, distress or pain by crying [220]. Intra-individual variation in infant cries is known to encode qualitative and quantitative information on the condition, needs, emotional status and the degree of urgency. Infant cry carries multiple levels of information as shown in Figure 6.16. Based on the perception of the cry, parents or caretakers empirically try to understand the reason for the crying and even

Table 6.12: The comparison of proposed features with various features proposed in the ASVspoof 2017 Challenge

Feature Set	Classifier	Dim	Dev	Eval
Challenge baseline [205]	GMM	90	10.35	30.60
Proposed:ConvRBM-CC	GMM	120	**8.19**	**14.86**
Proposed fusion	GMM	120	**5.90**	**14.56**
VESA-IFCC [214]	GMM	120	4.61	14.06
HFCC [206]	GMM	30	5.9	23.9
CQCC [215]	DNN	90	5.18	19.41
CQCC [215]	ResNet	90	5.05	18.79
CQCC (6-8 kHz) [207]	GMM	90	5.13	17.31
DA-CQCC [216]	GMM	60	7.01	19.18
DA-CQCC [216]	ResNet	60	6.32	23.14
MFCC [215]	ResNet	90	10.95	16.26
SFFCC-D [213]	GMM	30	**2.35**	20.2
SFFCC-D [213]	BLSTM	30	3.66	22.4
SCMC [217]	GMM	120	9.32	11.49
RFCC [217]	GMM	60	6.91	11.90
MFCC [217]	GMM	90	7.76	27.12
LCNN (FFT) [133]	GMM	-	4.53	**7.37**
LCNN (CQT) [133]	GMM	-	4.80	16.54

identify their newborn [220]. Recently, there is an increasing effort to investigate the reasons for Sudden Infant Death Syndrome (SIDS) [34] through the analysis of infant cry signals. Infant cry analysis is also valuable in the clinical diagnostics in order to know whether a disease to the newborn is due to the central nervous system (CNS) [221]. From a signal processing perspective, our goal is to classify whether the infant is crying due to pain, hunger or some medical diseases. The detailed discussion on the topic of infant cry and the literature of infant cry classification are found in [34] and in [222] to the best of authors' knowledge, the first Ph.D. thesis in this area from India. To date, there is no standard publicly available database for infant cry classification. Many researchers collected their own data including our Speech Research Group at DA-IICT [8], [223]. Other studies include the work in [136, 224–227]. In this chapter, we used the Baby Chillanto infant cry database, which is a property of INAOE-CONACyT, Mexico [228]. The literature of using representation learning for the ICC task is given in Table 2.3, Chapter 2. These techniques used handcrafted features such as the MFCC for acoustic analysis. Here, we used our proposed auditory filterbank learning framework using ConvRBM for feature learning from infant cry signals. Next, we discuss the

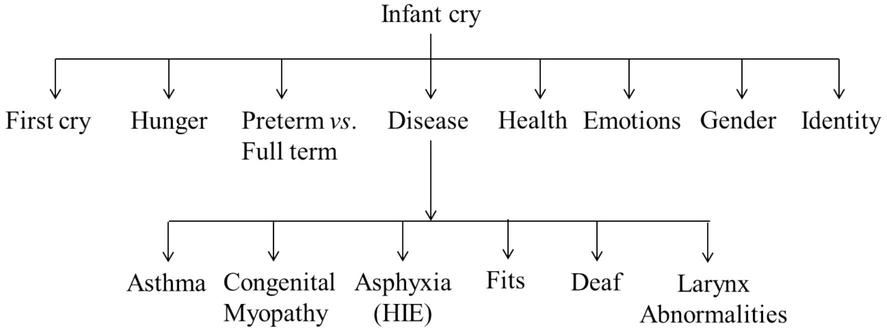


Figure 6.16: Multiple levels of information present in the infant cry signal. Adapted from [34].

database, the experimental setups, and results.

6.12 Experimental Setup

6.12.1 Databases

The DA-IICT Infant Cry database was collected as a part of a B.Tech project work, Ph.D. thesis work, and the DST fast track award for young scientists to Prof. Hemant A. Patil for the project, "Development of Infant Cry Analyzer using Source and System Features" [8]. The infant cry data was collected from three hospitals in Visakhapatnam, namely, 1. King George Hospital, 2. Prabha Nursing Home, and 3. Child Clinic. The sampling frequency of the original recordings was 12 kHz, quantized at 16-bit PCM. For our experiments, we downsample it to 11.025 kHz since at a later stage, we will compare the experimental results with another database. The statistics of the DA-IICT Infant Cry database is shown in Table 6.13. The healthy cry signals consist of normal and hunger cry signals. The pathology cry includes two types of pathologies, namely, asphyxia (also called Hypoxic Ischemic Encephalopathy (HIE)) and Asthma.

Table 6.13: Description of DA-IICT Infant Cry database. After [8]

Class	Category	No. of samples
Healthy	Normal, hunger	793
Pathology	Asphyxia	215
	Asthma	182

The Baby Chillanto infant cry database was developed by recordings conducted by medical doctors. The infant cry signals were carefully labeled at the time of the recording with the references, such as reason for crying, sick or not,

Table 6.14: Description of infant cry database.

Class	Category	No. of samples
Healthy	Normal	507
	Hungry	350
	Pain	192
Pathology	Asphyxia	340
	Deaf	879

and, infant age. Each cry signal was segmented into one second duration (that represents one sample) and is grouped into five categories as shown in Table 6.14. Since the sampling rate of cry signals is different in all the categories, we kept the sampling rate of 11.025 kHz for all the categories. Two groups were formed for binary classification of healthy *vs.* pathology. Healthy cry signals include three categories, namely, normal, hungry, and pain resulting in 1049 cry samples. Pathology cry signals include two categories, namely, asphyxia (also called as Hypoxic Ischemic Encephalopathy (HIE)) and deaf resulting in 1219 cry samples.

6.12.2 Feature Extraction and GMM Training

ConvRBM is trained with infant cry signals with parameters similar to those used in the other audio classification task. The model is trained with 40 subband filters (i.e., K) with convolution window length m= 88 samples (i.e., 8 ms). After the model was trained, the features were extracted from the infant cry signals. The 39-D cepstral features ConvRBM-CC are used in the GMM classifier. Since the Baby Chillanto infant cry database is very small in size (37 min and 50 second duration), the GMM is used for binary classification. Healthy cry features belong to one class and pathology cry features belong the other class. The GMMs with different mixture components were trained using the MFCC and ConvRBM-CC feature sets. The results are predicted using log-likelihood scores with 10-fold cross-validation. In each fold, the numbers of healthy and pathology cry samples were 945 and 1098, respectively, for training and the remaining (104 and 121 for healthy and pathology, respectively) for testing. For each fold, we noted % classification accuracy and % EER. The performance of the ICC task is evaluated using various performance measures obtained from the confusion matrix as discussed in Appendix B.3.

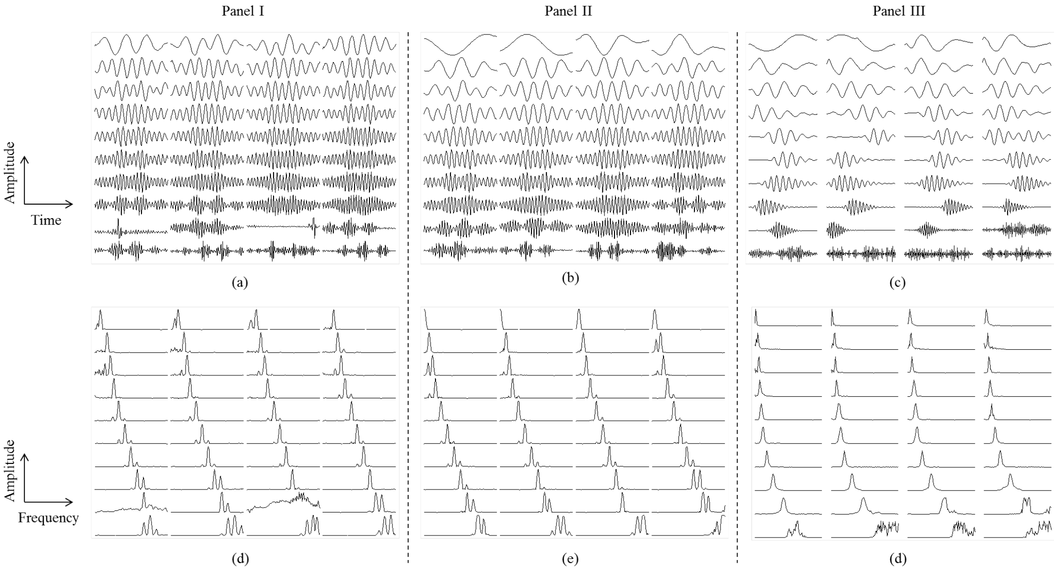


Figure 6.17: The subband filters trained on DA-IICT Infant Cry (Panel I), Baby Chillanto (Panel II), and TIMIT (Panel III) databases, respectively:(a)-(c) in the time-domain, (d)-(f) corresponding frequency responses. After [9].

6.13 Analysis of Infant Cry Signals

6.13.1 Analysis of Subband Filters and Frequency Scale

The subband filters learned from the DA-IICT Infant Cry and Baby Chillanto databases are shown in Figure 6.17. We have also shown the subband filters obtained from the TIMIT speech database. It is very interesting to note an intriguing observation that these subband filters were learned from only 37 minutes and 50 seconds duration of cry signals from the Baby Chilanto and 30 minutes of cry signals from the DA-IICT Infant Cry database (such scarcity of larger databases is all the more the case in medical scenarios). Thus, it shows the applicability of our proposed model even in very small database scenarios. The time-domain subband filters are significantly different from the one for the normal adult TIMIT speech database. The subband filters of the infant cry databases contain more Fourier-like basis functions due to the harmonic nature of the infant cry signals as shown in Figure 6.18. The analysis of the frequency-domain subband filters revealed that many subband filters are not localized and contain harmonic structures. This may be due to more harmonic content present in the infant cry signals. On comparing the subband filters learned from the two different databases, the subband filters from the baby Chillanto database has more lower frequency filters. However, the filter shapes of most of the subband filters are similar.

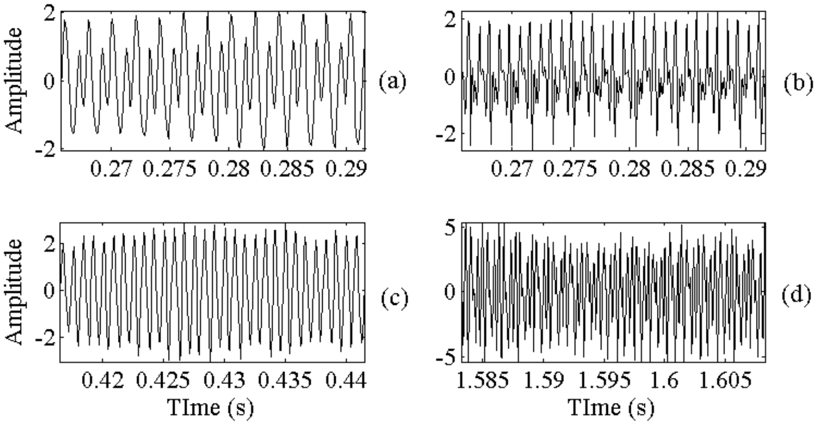


Figure 6.18: Segments of the infant cry signals showing the harmonic nature of the cry signals for (a) normal, (b) deaf, (c) asphyxia, and (d) asthma.

The frequency scales obtained using ConvRBM are compared with the standard auditory frequency scales in Figure 6.19. Unlike the scale obtained through the speech database [3], here we observed two linear segments in the frequency scale, from 0 to 1 kHz and from 1 kHz to 3 kHz. After 3 kHz, it is nonlinear and follows the ERB and Bark scales. However, the frequency scale from the DA-IICT Infant Cry database is more away from the standard scales. It has minimum center frequency of 500 Hz and after 4 kHz it follows the other frequency scales. The difference in the frequency scales of both the databases may be due to variability in the cry signal production mechanism through language perception (Indian languages *vs.* English in the Baby Chillanto), data recording conditions, background noise, channel characteristics, microphone specifications, etc. The scatter plot of bandwidth *vs.* CF is also shown in Figure 6.20. Due to harmonic nature of cry sounds, Q factors of ConvRBM filterbank are not constant specifically up to 4 kHz. The bandwidth of ConvRBM subband filters is constant around 150 Hz for CF up to 4 kHz. However, after 4 kHz bandwidth value increases as the CF also increases. This may be due to the fact that many of pathological cry signals contain distorted harmonics, transient, and noisy-like sounds. Hence, it leads to learn few large bandwidth subband filters at higher frequencies.

6.13.2 Analysis of ConvRBM Spectrograms

In this section, the spectrogram representation for the cry signals using the ConvRBM filterbank is presented in detail.

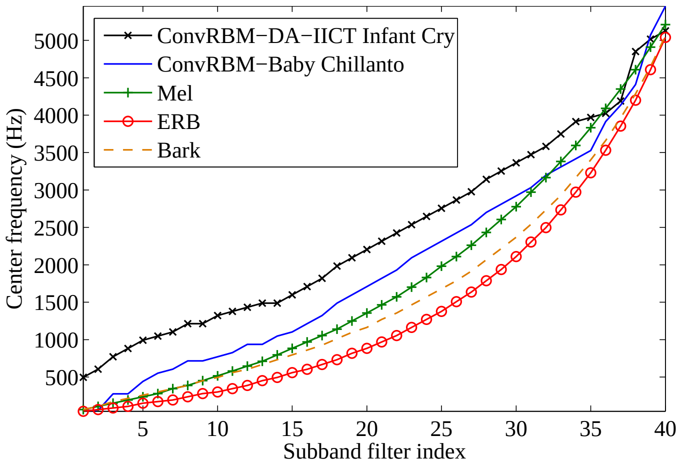


Figure 6.19: Comparison of the filterbanks learned using the ConvRBM with the standard auditory filterbanks.

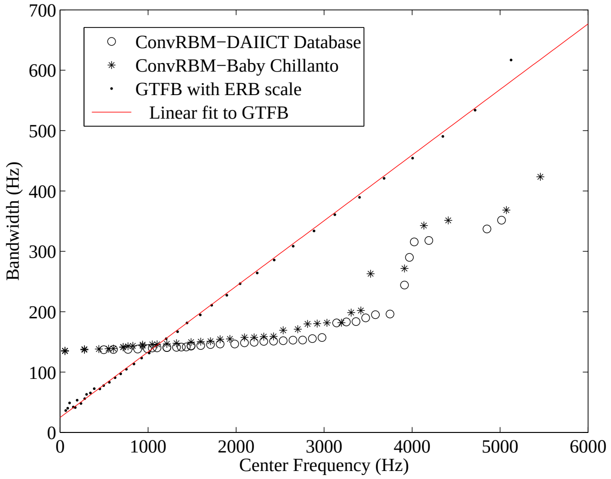


Figure 6.20: Comparison of bandwidth *vs*. CF for infant cry databases. After [9].

6.13.2.1 Normal Infant Cry Signals

The spectrograms from three normal infant cry signals taken from the Baby Chilanto database are shown in Figure 6.21. A better time-frequency resolution is obtained using the ConvRBM filterbank as marked in the spectrograms, specifically in the high frequency regions. We can see the slowly-varying harmonic structures and some noise (this is predominantly due to the turbulent excitation source and not due to the environmental noise) in the normal cry signals that are related to the *cry modes* as observed in [8], [222]. Figure 6.21 (a) is an example of falling, (b) is an example of flat, and (c) is an example of rising with the vibration cry mode. We can also observe the dysphonation cry mode in Figure 6.21 (a) after 0.2 seconds along with the falling cry mode. The spectrograms from the three

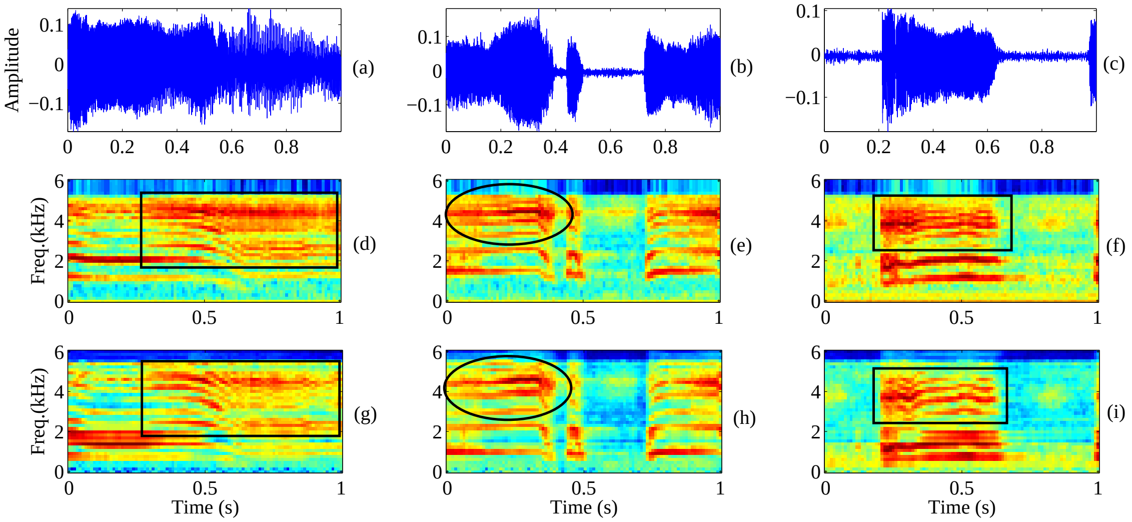


Figure 6.21: Comparison of the spectrograms for normal cry from the Baby Chillanto database: (a)-(c) time-domain signals, (d)-(f) Mel spectrograms, and (g)-(i) ConvRBM spectrograms. The rectangular and circular regions indicate differences in the two spectrograms.

normal infant cry signals taken from the DA-IICT Infant Cry database are shown in Figure 6.22. The resolution of the ConvRBM spectrograms is higher than the Mel spectrograms, as shown in marked regions in Figure 6.22. The harmonics are clearly resolved in the ConvRBM spectrograms. The cry modes, such as series of rising, falling, and flat, can be observed in Figure 6.22 (d) and (g). The dysphonation cry mode is observed in Figure 6.22 (e), (h) with the harmonic vibration mode (shown by a circle). Our observations for the normal cry signals are similar to those observed in [8], [222] for the normal infant cry signals.

6.13.2.2 Asphyxia Infant Cry Signal

The asphyxia or HIE is a disease caused in the newborn due to the lack of supply of oxygen or blood to the brain that arises due to abnormal breathing. In very serious conditions, asphyxia can cause coma or even death. The infants suffering from asphyxia are not able to produce a normal cry that results in pathological signs in the cry signals. The spectrograms from three asphyxia infant cry signals taken from the Baby Chillanto database are shown in Figure 6.23. The time-frequency resolution is significantly better compared to the Mel spectrograms as can be seen from Figure 6.23. The difference between normal and asphyxia cry is clearly visible from the spectrograms. There are no continuous harmonic structures present in the asphyxia cry; rather, it is of very short duration and noisy. This is due to the infant is not able to vocalize due to an inadequate supply of

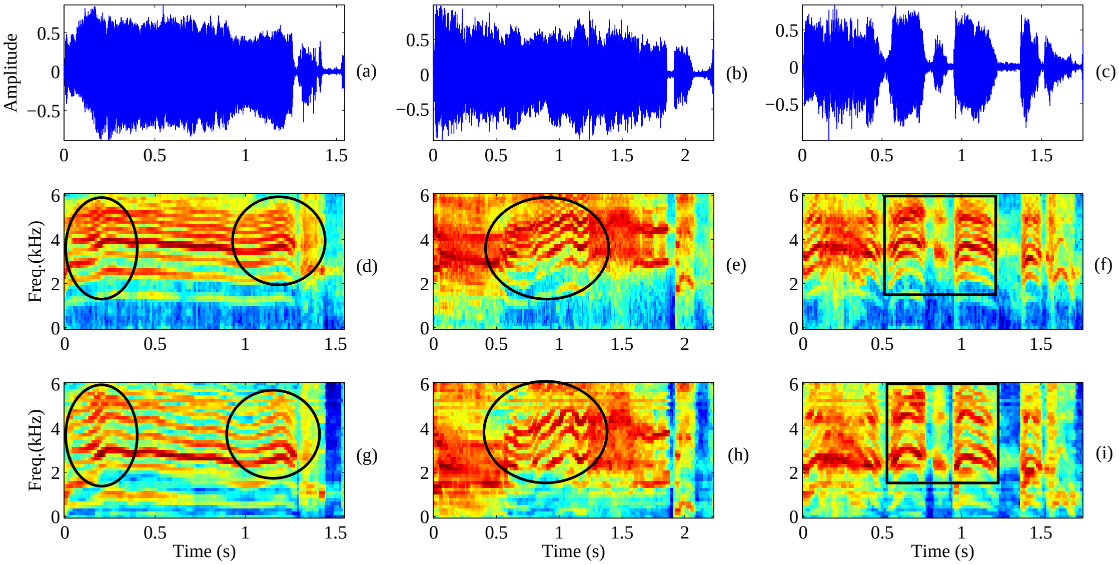


Figure 6.22: Comparison of spectrograms for normal cry signals from the DA-IICT Infant Cry database: (a)-(c) time-domain signals, (d)-(f) Mel spectrograms, and (g)-(i) ConvRBM spectrograms. The rectangular and circular regions indicate the differences in two spectrograms.

oxygen or blood to his/her brain. Many of the cry modes related to harmonics are absent in asphyxia cry. The blurred harmonics can be seen from asphyxia cry signals in Figure 6.23 (a)-(c). The ConvRBM spectrogram can show continuous dysphonation cry mode for one of the asphyxia cry signals in Figure 6.23 (i) which is not revealed by the Mel spectrogram in Figure 6.23 (f). Similar observations are made from the asphyxia cry signals taken from the DA-IICT Infant Cry database as shown in Figure 6.24. The continuous dysphonation cry mode is present in the asphyxia cry signals shown in Figure 6.24 (g). One can see that the Mel spectrograms are not able to resolve leading and trailing harmonics on both sides of dysphonation cry mode. The asphyxia cry signals from the DA-IICT Infant Cry database also show much less spectral energies or dysphonation cry mode in the spectrograms.

6.13.2.3 Deaf Infant Cry

There are several reasons for deafness in newborns and in many cases they become deaf early in life. It is not always possible to identify the reason for such cases; however, there are two possible cases, namely, pre-natal and post-natal causes [229]. Pre-natal cases include genetic reasons, complications during pregnancy, illnesses, such as rubella, cytomegalovirus (CMV), toxoplasmosis and her-

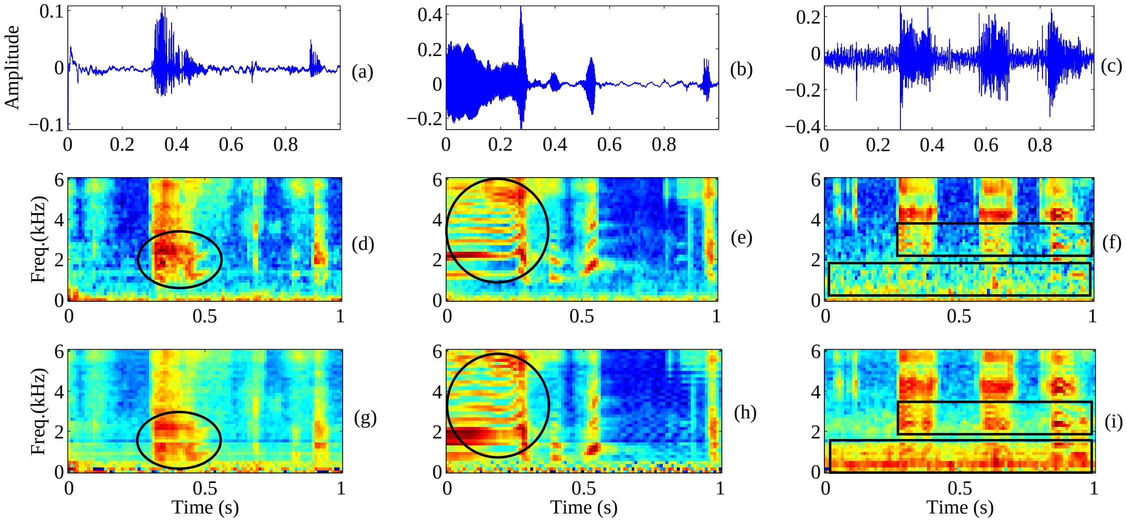


Figure 6.23: Comparison of spectrograms for asphyxia cry signals from the Baby Chillanto database: (a)-(c) time-domain signals, (d)-(f) Mel spectrograms, and (g)-(i) ConvRBM spectrograms. The rectangular and circular regions indicate the differences in two spectrograms.

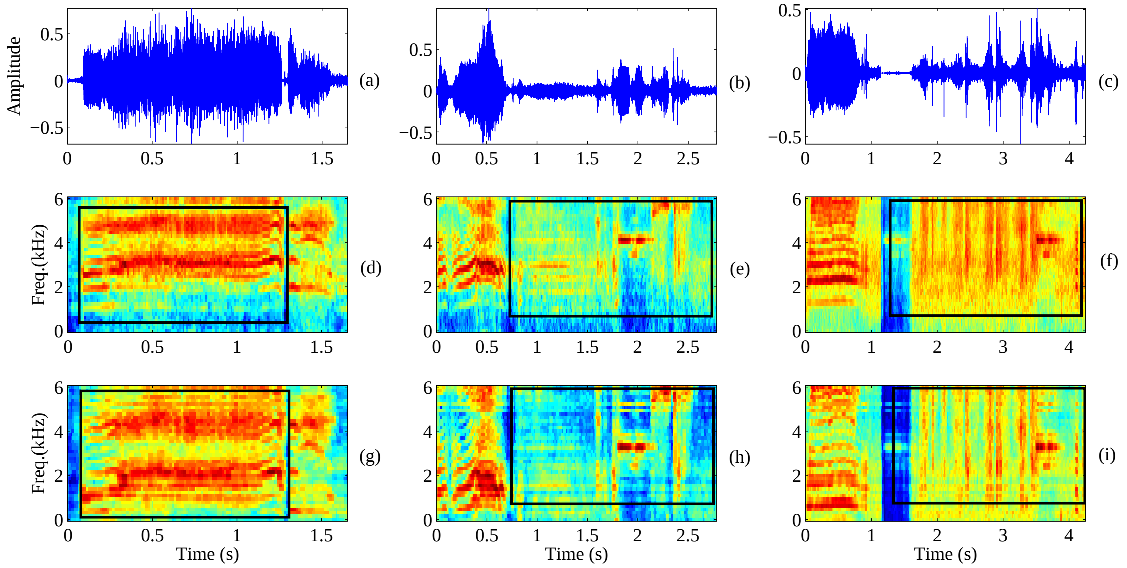


Figure 6.24: Comparison of spectrograms for asphyxia cry signals from the DA-IICT Infant Cry database: (a)-(c) time-domain signals, (d)-(f) Mel spectrograms, and (g)-(i) ConvRBM spectrograms. The rectangular regions indicate the differences in two spectrograms.

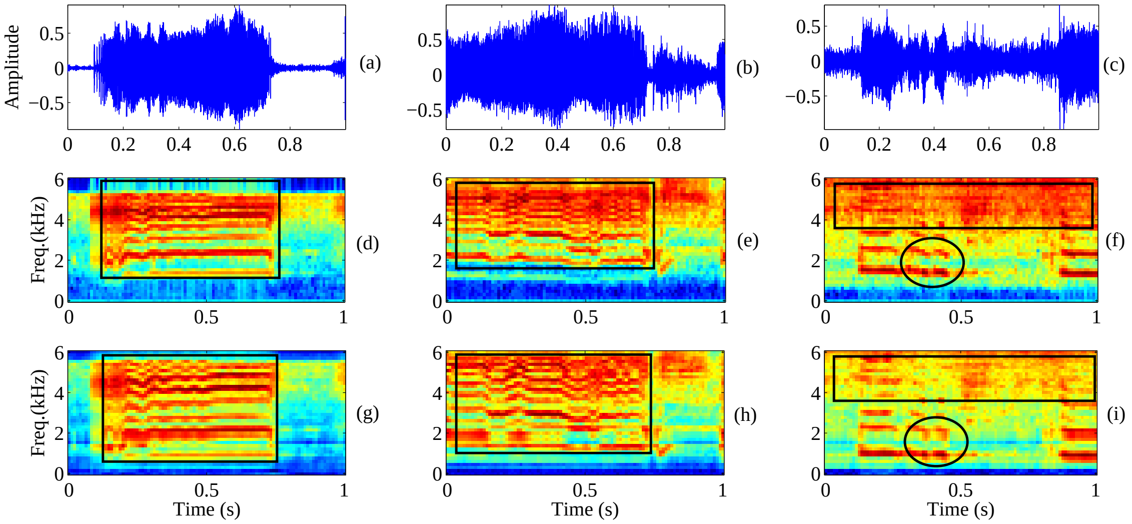


Figure 6.25: Comparison of spectrograms for deaf cry signals from the Baby Chillanto database: (a)-(c) time-domain signals, (d)-(f) Mel spectrograms, and (g)-(i) ConvRBM spectrograms. The rectangular and circular regions indicate the differences in two spectrograms.

pes can cause newborn to be deaf [229]. Post-natal causes include infection, specifically in prematurely born babies and exposure to loud noise [229]. The deaf infant's cry signals differ from the normal infant cry. The onset of crying or canonical babbling is delayed in deaf infants and cry signals differ in duration and timing [230]. Moreover, vocal cry inventories are very limited in the deaf infants. The deaf infants rely on only sounds that are visually prominent, such as /ba/ and /ma/. It has significant impact on the acquisition of language where sound perception plays a critical role [44]. Hence, early detection of deafness in infancy may help in providing a hearing aid that benefit for the better development of infants. Spectrograms from the three deaf infant cry signals from the baby Chillanto database are shown in Figure 6.25. One can see more resolved harmonics in the high frequency regions in ConvRBM spectrograms (as marked in Figure 6.25) compared to the Mel spectrograms. In all the deaf cry samples, dysphonation cry mode is present in the high frequency regions. There are vibration cry modes also present as seen in the cry signals in Figure 6.25) (a) and Figure 6.25) (b).

6.13.2.4 Asthma Cry

Asthma is a chronic inflammatory disease that inflames and narrows the airways. These airways allow air to come in and out of the lungs. Asthma causes recurring periods of wheezing (a whistling sound when you breathe), shortness of breath

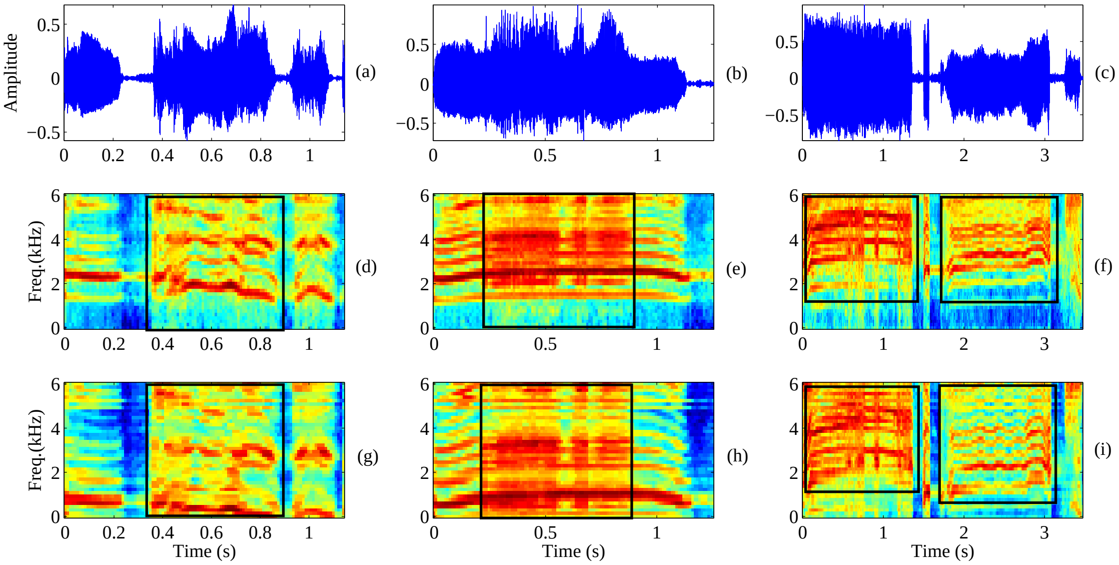


Figure 6.26: Comparison of spectrograms for asthma cry signals from the DA-IICT Infant Cry database: (a)-(c) time-domain signals, (d)-(f) Mel spectrograms, and (g)-(i) ConvRBM spectrograms. The rectangular regions indicate the differences in two spectrograms.

(i.e., difficulty in breathing), chest tightness, and coughing. The symptoms of asthma are seen in people of all ages, but it most often starts during childhood or in the infant stage. Asthma is thought to be caused by a combination of genetic and environmental factors that include allergens or air pollution. There is no cure for asthma till now; however, early symptoms can be prevented by avoiding triggers, such as allergens and irritants, etc. An infant suffering from asthma faces difficulties in breathing and hence, proper treatment must be conducted to reduce the symptoms. The spectrograms from the three deaf infant cry signals from the DA-IICT Infant Cry database are shown in Figure 6.26. Due to frequent inhalation, distorted harmonic structures are seen in the spectrograms in Figure 6.26 (d) and Figure 6.26 (g). Abrupt dysphonation cry modes are present in Figure 6.26 (e) and Figure 6.26 (h). Due to breathing difficulty, sometimes acoustic energy level, and harmonic frequency range changes abruptly Figure 6.26 (f) and Figure 6.26 (i).

6.14 Experimental Results

In this section, the classification results and evaluation using various performance measures are presented.

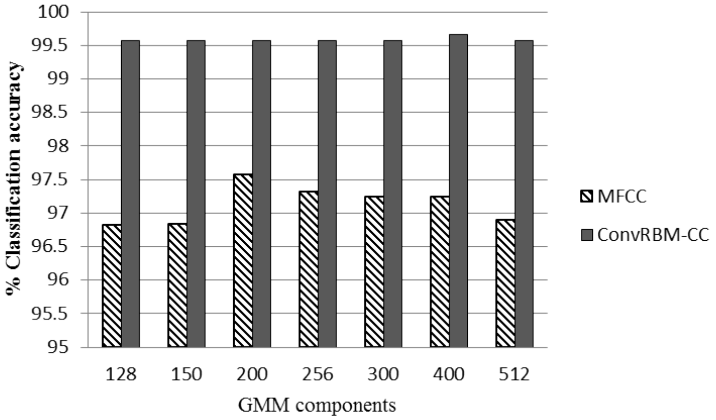


Figure 6.27: The % classification accuracy using for various GMM components the DA-IICT Infant Cry database. After [9].

6.14.1 Results on the DA-IICT Infant Cry Database

The classification accuracies for the DA-IICT Infant Cry database using the MFCC and ConvRBM-CC feature sets are shown in Figure 6.27. The ConvRBM-CC obtained higher % classification accuracy compared to the MFCC for all the GMM components. For the MFCC, optimal results are obtained using 200 GMM components. For the ConvRBM-CC, the optimal results are obtained using 400 GMM components. We achieved an absolute improvement of 2 % in the classification accuracy compared to the MFCC feature set. The confusion matrices for the classification experiment are shown in Figure 6.28. The FP and FN rate of the MFCC is quite high compared to the ConvRBM-CC feature set. From Figure 6.28 (b), it can be seen that the ConvRBM-CC has no FP and only 4 FN compared to the MFCC with 21 FN (Figure 6.28 (a)). Hence, with the ConvRBM-CC, there is no chance that normal cry signal is considered as a pathological cry signal.

	Healthy	Pathology
Healthy	791	8
Pathology	21	378

(a)

	Healthy	Pathology
Healthy	799	0
Pathology	4	395

(a)

Figure 6.28: Confusion matrices for experiments on the DA-IICT Infant Cry database using: (a) MFCC, and (b) ConvRBM-CC. After [9].

The performance measures of the classification experiments on the DA-IICT Infant Cry database are shown in Table 6.15. The ConvRBM-CC obtain significantly high values for all the measures compared to the MFCC. Since F-measures do not consider the true negatives, the values of an F-measure are very similar for both the feature sets. The MCC and the J-statistic values are higher for the ConvRBM-CC compared to the MFCC. The % accuracy does not consider false positives and false negatives. From Table 6.15, one can see that the difference in MCC and J-statistic for the MFCC and ConvRBM-CC is higher compared to % accuracy. Hence, MCC and the J-statistic are more meaningful performance measures than just % classification accuracy. Hence, ConvRBM-CC discriminates normal and pathology classes in the DA-IICT Infant Cry database more significantly than the MFCC.

Table 6.15: Performance measures for the classification experiments on the DA-IICT Infant Cry database. After [9].

Feature Set	MCC	F-measure	J-statistic
MFCC	0.945	0.963	0.937
ConvRBM-CC	**0.993**	**0.995**	**0.99**

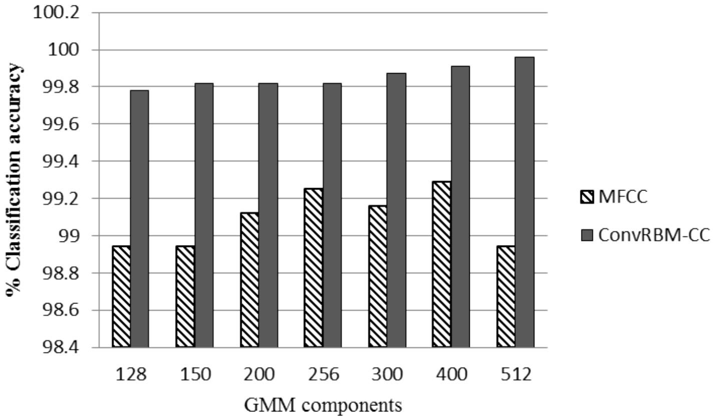


Figure 6.29: The % classification accuracies using for various GMM components of the Baby Chillanto database. After [9].

6.14.2 Results on the Baby Chillanto Database

The experimental results using the Baby Chillanto Database are shown in Figure 6.29 for the MFCC and ConvRBM-CC with different GMM mixture components. Compared to the DA-IICT Infant Cry database, both the feature sets were able to perform well in the classification of normal and pathology cry signals. However, ConvRBM-CC consistently performs better than the MFCC for all the GMM mixture components. The best classification accuracy of 99.87 % was achieved using ConvRBM-CC (0.58 % absolute improvement compared to the MFCC) obtained with 300 GMM mixture components. The confusion matrices for both feature sets are shown in Figure 6.30. The false positive rate of the MFCC is quite higher than the ConvRBM-CC (15 *vs.* 1), while there are no false negatives when the ConvRBM-CC is used in the classification task. Hence, with the ConvRBM-CC feature set, all the cry samples are correctly classified with only one false negative. The significance of this improvement using the ConvRBM-CC feature set can also be seen from the performance measures in Table 6.16. Here, again the F-measure is similar for both ConvRBM-CC and the MFCC. The MCC and the J-statistic are quite high for the ConvRBM-CC with value 0.999 (close to 1). The difference in values of the MCC and the J-statistic indicates that the ConvRBM-CC performs better than the MFCC even though % accuracy is quite similar.

	Healthy	Pathology
Healthy	1034	15
Pathology	1	1218

(a)

	Healthy	Pathology
Healthy	1048	1
Pathology	0	1219

(a)

Figure 6.30: Confusion matrices for experiments on the Baby Chillanto database using: (a) MFCC, and (b) ConvRBM-CC. After [9].

Table 6.16: Performance measures for the classification experiments on the baby Chillanto database. After [9].

Feature Set	MCC	F-measure	J-statistic
MFCC	0.986	0.994	0.985
ConvRBM-CC	**0.999**	**0.999**	**0.999**

6.15 Chapter Summary

The application of ConvRBM on audio classification tasks, namely, the ESC, SSD, and ICC task are presented in this Chapter. The experiments on the ESC task demonstrate that it performs significantly better than the baseline system and end-to-end CNN-based system. The experiments on the SSD task (synthetic and replay) show that the proposed ConvRBM-based features performed well compared to the MFCC-based baseline. Finally, we have also shown the improved performance in the ICC task by learning the filterbank from a very small amount of database. In the next Chapter, we will present our unsupervised deep auditory model obtained by stacking two ConvRBMs.

CHAPTER 7

Unsupervised Deep Auditory Model (UDAM)

7.1 Introduction

In Chapter 3, we have introduced our proposed model for auditory filterbank learning using ConvRBM. In this Chapter, we extend our proposed approach to model time-frequency representation of the sound in the higher-levels of the auditory processing. Since auditory processing is hierarchical in nature, we propose the two layer auditory representation learning model using stacks of the ConvRBMs. We first describe our proposed model of learning temporal modulation representation in Section 7.2. Our initial attempt of using the Mel spectrogram as an input to ConvRBM is discussed in Section 7.3-7.4. In Section 7.5, we will show that deep auditory representation can be achieved by taking spectrum representation of the ConvRBM trained on the speech signals as an input to another ConvRBM. The ASR experiments and results are given in Section 7.6. The improved UDAM model is presented along with experimental results in Section 7.7.

7.2 Temporal Modulations in Speech using ConvRBM

The ConvRBM filterbank responses show the temporal modulations at various temporal scales as discussed in Appendix A. Temporal modulations in speech can be modeled by applying a wavelet or Fourier transform on subbands of spectro-temporal representation of a speech signal. Hence, in order to use ConvRBM to model temporal modulations, the earlier study used PCA whitened spectrograms as an input to ConvRBM [50]. The ConvRBM is applied along each frequency subband in a similar manner to the modulation spectrum computation [231]. The difference between modulation spectrum and our proposed model is that we learn multiple modulation filters ($\mathbf{W} \in \mathbb{R}^{m \times S \times K}$, where m, S, K are the modulation filter length, number of subbands, and number of groups, respectively) for each subband. In our initial attempt, we proposed to use Mel spectrograms in ConvRBM with NReLUs [10]. Later, we used the ConvRBM filterbank that is learned from the raw speech signals to model temporal modulations [11]. In this section, we describe the architecture and training procedure of the model proposed in [10] and [11].

The input to the ConvRBM is a spectro-temporal representation of a speech signal. Let us denote the input $\mathbf{x} \in \mathbb{R}^{S \times F}$ as the spectro-temporal representation (also called time-frequency representation), where S and F are the numbers of subbands (also called as *channels* in terms of CNN terminology [20]) and frames, respectively. The hidden layer has K groups. The temporal convolution operation is applied on each subband individually from all the K groups. The convolutional responses for the k^{th} group in the hidden layer are given as:

$$\begin{aligned} \mathbf{y}_1^k &= \mathbf{x}_1 * \tilde{\mathbf{w}}_1{}^k, \\ \mathbf{y}_2^k &= \mathbf{x}_2 * \tilde{\mathbf{w}}_2{}^k, \\ \vdots &= \quad \vdots \\ \mathbf{y}_S^k &= \mathbf{x}_S * \tilde{\mathbf{w}}_S{}^k, \end{aligned} \tag{7.1}$$

where $\tilde{\mathbf{w}}_s{}^k$ are weights of the k^{th} group in $s = 1, .., S$ subbands. The convolutional responses are added together in the respective groups followed by addition of biases in each group as shown below:

$$\mathbf{I}_k = \sum_{s=1}^{S} \left(\mathbf{x}_s * \tilde{\mathbf{w}}_s{}^k \right) + b_k, \tag{7.2}$$

where b_k is the hidden bias in the k^{th} group. The sampling equation of hidden units in the k^{th} group is similar to what we presented in Chapter 3:

$$\mathbf{h}^k \sim \max(0, \mathbf{I}_k + \mathcal{N}(0, \text{sigmoid}(\mathbf{I}_k))). \tag{7.3}$$

In the negative phase, the reconstructed subband signals are given as:

$$\begin{aligned} \mathbf{x}_1 &\sim \mathcal{N}\left(\sum_{k=1}^{K} \left(\mathbf{h}^k * \mathbf{w}_1^k \right) + c_1, \right), \\ \mathbf{x}_2 &\sim \mathcal{N}\left(\sum_{k=1}^{K} \left(\mathbf{h}^k * \mathbf{w}_2^k \right) + c_2 \right), \\ \vdots & \quad \vdots \\ \mathbf{x}_S &\sim \mathcal{N}\left(\sum_{k=1}^{K} \left(\mathbf{h}^k * \mathbf{w}_S^k \right) + c_S \right), \end{aligned} \tag{7.4}$$

where $\mathbf{c} = [c_1, c_2, ..., c_S]$ are the visible biases that are also shared among all the visible units in each subband. The energy function of ConvRBM applied on the spectro-temporal representation of speech signals is given as:

$$
\begin{aligned}
E(\mathbf{x}, \mathbf{h}) = \frac{1}{\sigma_x^2} \sum_{s=1}^{S} \sum_{i=1}^{F} x_{s,i}^2 - \frac{1}{\sigma_x} \sum_{s=1}^{S} \sum_{k=1}^{K} \sum_{j=1}^{l} \sum_{r=1}^{m} \left(h_j^k w_{s,r}^k x_{s,j+r-1} \right) \\
- \sum_{k=1}^{K} b_k \sum_{j=1}^{l} h_j^k - \frac{1}{\sigma_x^2} \sum_{s=1}^{S} \sum_{i=1}^{F} c x_{s,i}.
\end{aligned}
\tag{7.5}
$$

The variance σ_x is set to one, when the PCA whitening or spectral mean variance normalization (SMVN) is applied on the input as suggested in the practical guide of training RBM [60]. The gradient of weights in the k^{th} group is given as:

$$
\frac{\partial}{\partial w_{s,r}^k} E(\mathbf{x}, \mathbf{h}) = \frac{\partial}{\partial w_{s,r}^k} \left[\sum_{s=1}^{S} \sum_{k=1}^{K} \sum_{j=1}^{l} \sum_{r=1}^{m} \left(h_j^k w_{s,r}^k x_{s,j+r-1} \right) \right]. \tag{7.6}
$$

For subbands, $s = 1, 2, ..., S$, eq. (7.6) can be written as a set of equations similarly as derived in filterbank learning ConvRBM in Chapter 3:

$$
\begin{aligned}
\frac{\partial}{\partial w_{1,r}^k} E(\mathbf{x}, \mathbf{h}) &= \mathrm{conv}(\mathbf{x}_1, \tilde{\mathbf{h}}^k), \\
\frac{\partial}{\partial w_{2,r}^k} E(\mathbf{x}, \mathbf{h}) &= \mathrm{conv}(\mathbf{x}_2, \tilde{\mathbf{h}}^k), \\
\vdots \quad &= \quad \vdots \\
\frac{\partial}{\partial w_{S,r}^k} E(\mathbf{x}, \mathbf{h}) &= \mathrm{conv}(\mathbf{x}_S, \tilde{\mathbf{h}}^k).
\end{aligned}
\tag{7.7}
$$

It can also be written in matrix form as follows, where each $\mathbf{W}^k \in \mathbb{R}^{m \times S}$:

$$
\frac{\partial}{\partial \mathbf{W}^k} E(\mathbf{x}, \mathbf{h}) = \left[\frac{\partial}{\partial w_{1,r}^k} E(\mathbf{x}, \mathbf{h}), \frac{\partial}{\partial w_{2,r}^k} E(\mathbf{x}, \mathbf{h}), ..., \frac{\partial}{\partial w_{S,r}^k} E(\mathbf{x}, \mathbf{h}) \right]. \tag{7.8}
$$

The gradient update rule for weights in the k^{th} group is given as:

$$
\nabla \mathbf{W}^k = \epsilon \left(\left\langle \mathrm{conv}(\mathbf{x}_s, \tilde{\mathbf{h}^k}) \right\rangle_{data} - \left\langle \mathrm{conv}(\underline{\mathbf{x}_s}, \underline{\tilde{\mathbf{h}^k}}) \right\rangle_{model} \right), \quad s = 1, 2, ..., S. \tag{7.9}
$$

With similar notations, the gradient update equations for hidden and visible biases are given as:

$$\begin{aligned} \nabla b_k &= \epsilon \left(\left\langle \sum_{j=1}^{l} h_j^k \right\rangle_{data} - \left\langle \sum_{j=1}^{l} \underline{\tilde{h^k}} \right\rangle_{model} \right), \\ \nabla c_s &= \epsilon \left(\left\langle \sum_{i=1}^{n} x_{s,i} \right\rangle_{data} - \left\langle \sum_{i=1}^{n} \underline{x_{s,i}} \right\rangle_{model} \right), \quad s = 1, 2, ..., S. \end{aligned} \tag{7.10}$$

In the next section, we will describe modulation representation learning using the Mel Spectrograms.

7.3 ConvRBM Applied on Mel Spectrograms

An input to the ConvRBM are Principal Component Analysis (PCA) whitened Mel spectrograms extracted from the speech signals. Whitening the data using PCA gives an approximation to sub-cortical processing, which was observed in the auditory cortex [232]. In this study, we also analyzed the sigmoid and ReLU activation functions in the ConvRBM. For analysis of the proposed model, the TIMIT database is used to train the model. We have used a system combination framework for the Mel filterbank and modulation features learned by the ConvRBM. The ASR experiments were carried out by using the TIMIT and WSJ0 databases.

7.3.1 Analysis of Learned Subband Filters

The filters learned in ConvRBM are visualized by applying the inverse of PCA whitening on the ConvRBM weights. Since convolution is applied in the temporal-domain (for each subband), subband filters represent Temporal Receptive Fields (TRFs) (see Section 2.6.2, Chapter 2 for details) [50]. Examples of the TRFs learned on the TIMIT database are shown in Figure 7.1, where each block represents one TRF.

Unlike the cells in the visual cortex (known as V1), all the Receptive Fields (RFs) in the auditory cortex (known as A1) are not localized [233]. Receptive fields in A1 exhibit multiple characteristics as certain cells demonstrate responses from the multiple frequencies [232]. Here, we observe similar behavior of the TRFs. From Figure 7.1, it can be seen that some of the modulation filters are highly localized along the Mel frequency-axis (e.g., A1 and A2), while some modulation filters are broadly distributed (e.g., B1, B2 and B3). Some modulation filters have strong localized excitatory and inhibitory regions (e.g., C1 and C2) while few have

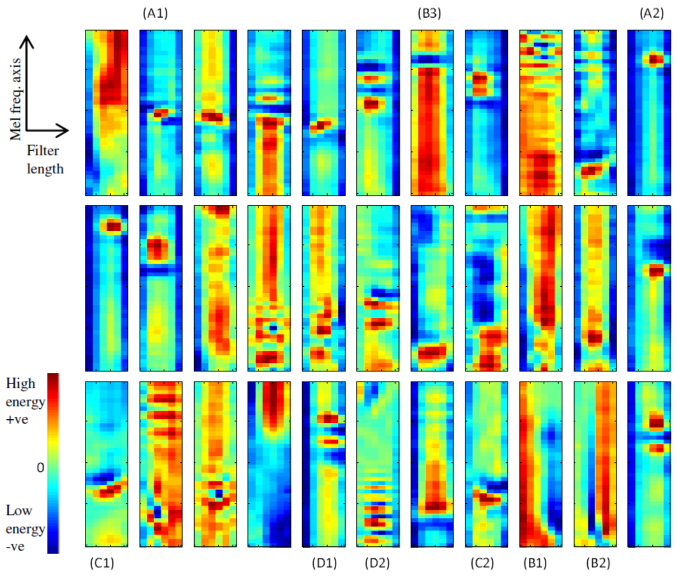


Figure 7.1: Examples of ConvRBM modulation filters. After [10].

a checkerboard-like pattern (e.g., D1 and D2). Similar patterns of RFs were also found in STRFs in the auditory cortex, when PCA whitened spectrograms were applied on the sparse coding algorithm [234]. Hence, ConvRBM filters capture the temporal modulation information with different modulation frequencies in each subband from the Mel spectrograms [235]. As shown in [50], each modulation filter may represent the temporal variations in different phonetic units.

7.3.2 Rectified Linear Units (ReLU) in ConvRBM

We justify the use of ReLUs in ConvRBM (compared to the first model proposed in [50]) by visualizing the reconstruction from the model using both nonlinearities as shown in Figure 7.2. It can be observed that the reconstruction from the sigmoid units is more noisy (as shown via dotted circles) compared to the original spectrogram and the one reconstructed using the ReLU. This noise is due to the saturation of neurons and the vanishing gradient effect in the case of a sigmoid nonlinearity, which may affect the ASR performance. In the case of the ReLUs, the hidden units are not binary; rather neurons can take any value from 0 to ∞ and hence, can better represent the input speech signal.

7.3.3 Feature Extraction and System Combination for ASR

Since the ConvRBM filters capture the temporal modulation information, we can use this along with the standard spectral features, Mel frequency filterbanks (denoted as FBANK). We trained both features on separate DNNs and use the system

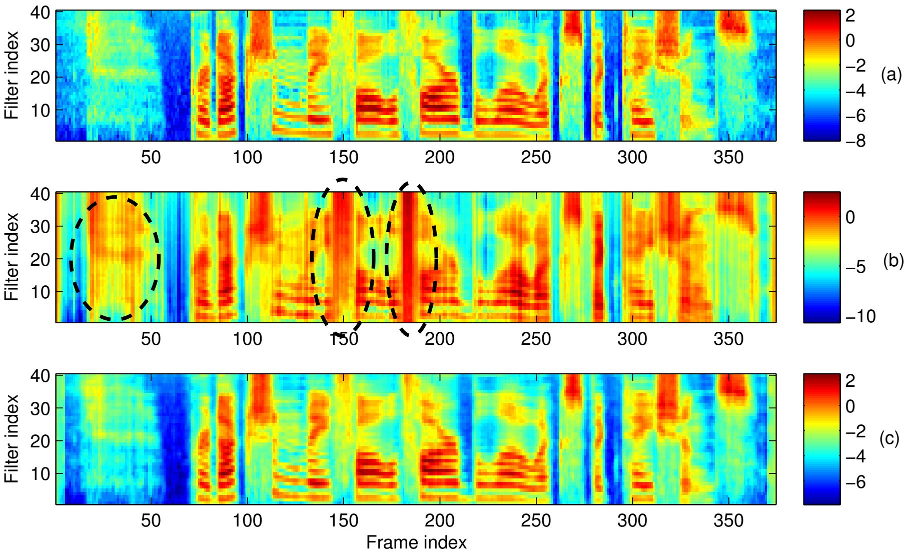


Figure 7.2: the Mel-spectrograms (a) original, (b) reconstructed using sigmoid units, (c) reconstructed using ReLUs. Dotted regions shows the saturation effect in the sigmoid units. After [10].

combination technique. We have used the MBR technique for system combination (discussed in Section 3.6.3, Chapter 3), which is very helpful when two different feature streams may represent complementary information [156]. Our feature extraction and system combination method is shown in Figure 7.3. The first pipeline is to extract FBANK features and their delta features to train DNN, which we denote as the spectral feature-trained DNN. The second pipeline is to extract the temporal modulation features from the trained ConvRBM and train DNNs on these feature sets, which we call modulation feature-trained DNN. Generated lattices from both the DNN systems are combined using the MBR decoding and then used for scoring.

7.4 Experimental Setup and Results

7.4.1 Training of ConvRBM

The Mel spectrograms were obtained from the speech signals by framing with a window length of 25 ms and a shift of 10 ms using 40 Mel subband filters. The PCA whitening was applied on all the concatenated Mel spectrograms. The learning rate was chosen to be 0.01 and was decayed at each epoch. The weight decay factor of 0.01 was used for the regularization. For the first five training epochs, momentum was set to 0.5 and after that it was set to 0.9. We trained the model on

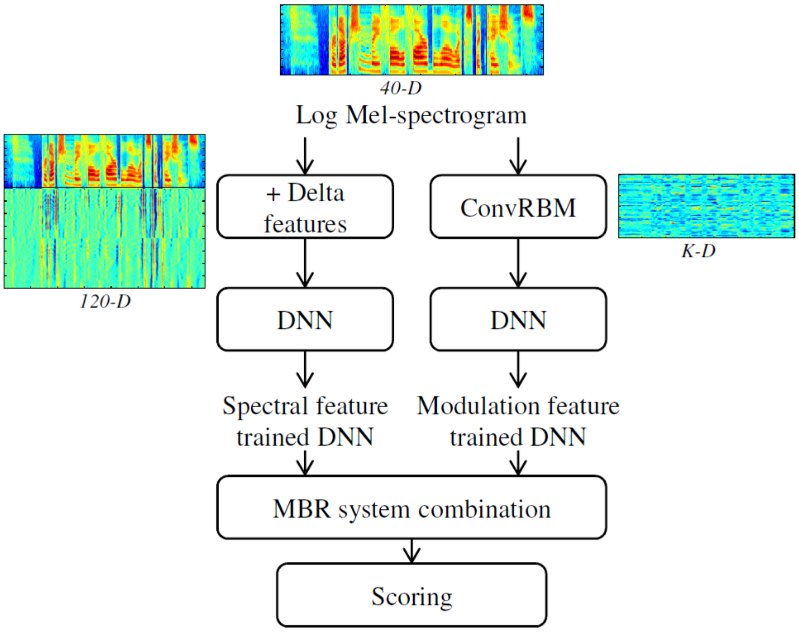


Figure 7.3: Block diagram for feature extraction and the system combination framework. After [10].

different numbers of subband filters 60, 80 and 120 with different filter lengths, namely, 6, 8, and 10. These parameters were optimized based on performance in the ASR experiments. The modulation filters were used to extract the features from the Mel spectrograms. Notations for ConvRBM using sigmoid and ReLU hidden units are ReLU-ConvRBM and Sigmoid-ConvRBM, respectively.

7.4.2 Hybrid DNN-HMM Systems

The monophone GMM-HMM systems were used to generate forced-aligned transcriptions for both the databases using the MFCC feature set. Acoustic modeling was performed with the DNN-HMM system in KALDI using Dan's recipe [155]. The CI-DNN-HMM systems were trained for the TIMIT and WSJ0 databases with DNN output labels and the LM described in Section 3.6, Chapter 3.

7.4.3 ConvRBM Parameter Tuning

Parameters of ConvRBM were optimized using a single-layer neural network trained on ReLU-ConvRBM features with 1500 hidden units and a CW of 11 frames. The parameters of ConvRBM include the number of filters and length of each filter. The results of these experiments are reported in Table 7.1 on the TIMIT and WSJ0 databases in % PER and % WER, respectively. From Table 7.1, for the TIMIT database 120 filters and a filter length of 6 frames gave the lowest

% PER. For the WSJ0 database, 60 filters and a filter length of 6 frames gave the lowest % WER. In the case of large databases, 60 and 120 filters yield almost the same WER and 60 filters are sufficient compared to double the number of filters for the TIMIT database.

Table 7.1: Results of ConvRBM parameter-tuning experiments on the TIMIT and WSJ0 databases. After [10]

Number of filters	Filter length	Dev (% PER)	Test (% PER)	Eval92_5K (% WER)
120	6	**24.3**	**25.6**	7.14
80	6	24.5	25.6	7.57
60	6	24.6	25.8	**7.10**
120	8	24.5	25.7	7.15
120	10	24.9	25.8	7.20
60	8	24.8	25.9	7.15

7.4.4 Results on the TIMIT Database

Two DNNs are trained using the FBANK (120-D) and ConvRBM (120-D) feature sets and the results are reported with the three hidden layers, 1500 hidden units and a context window of 11 frames. The performance of the ConvRBM-based feature set alone and with our system combination setup is reported in Table 7.2. The ReLU-ConvRBM feature set, which represents the modulation information, performs similar to spectral features, namely, FBANK. ConvRBM with the sigmoid units (denoted as Sigmoid-ConvRBM) did not perform well compared to the ReLU units. The system combination (denoted using the $\oplus$ symbol) of DNNs trained on ReLU-ConvRBM and FBANK feature sets gave relative improvement of 7.73 % and 5.93 % on the development and test sets, respectively. System combination using Sigmoid-ConvRBM has very low improvement on the development set (4.1% relative improvement) compared to the ReLU. This shows that the DNNs trained on the FBANK and ReLU-ConvRBM contain highly complementary information.

7.4.5 Results on the WSJ0 Database

Following the results of Table 7.1, we have used parameters of ConvRBM for WSJ0 experiments. FBANK and ConvRBM feature sets were trained using DNNs with three layers, 1500 hidden units, and a CW of 11 frames. The experimental results are reported in Table 7.3 in % WER. From the results, we can see that the

Table 7.2: Results on the TIMIT database in % PER. Numbers in the brackets indicate relative improvement over FBANK-DNN. After [10]

DNN System	Dev	Test
A: FBANK	22.0	23.6
B: ReLU-ConvRBM	22.3 (-1.36)	24.0 (-1.69)
C: Sigmoid-ConvRBM	23.4 (-6.36)	25.4 (-7.63)
A $\oplus$ B	**20.3 (7.73)**	**22.2 (5.93)**
A $\oplus$ C	21.1 (4.09)	22.6 (4.23)

$\oplus$ represents the system combination experiments

ReLU-ConvRBM features perform better than the Sigmoid-ConvRBM. The system combination of the FBANK and the ReLU-ConvRBM yield relative improvement of 4.3 % for the 5K word test set and 3.63 % for the 20K word test set over FBANK. System combination of the FBANK and the Sigmoid-ConvRBM also improved over FBANK. However, the improvement is less compared to the ReLU-ConvRBM with relative improvement of 1.48 % on 5K test set and 2.09 % on 20K test set. Hence, both ASR experiments show that the ReLU-ConvRBM performs better than Sigmoid-ConvRBM and performance is improved using system combination with the FBANK.

Table 7.3: Results in % WER and % relative improvements on WSJ0 database. After [10]

DNN System	Eval92_5K	Eval92_20K
A: FBANK	6.07	14.32
B: ReLU-ConvRBM	6.52 (-7.4)	15.15 (-5.7)
C: Sigmoid-ConvRBM	7.44 (-22.57)	16.16 (-12.84)
A $\oplus$ B	**5.81 (4.3)**	**13.80 (3.63)**
A $\oplus$ C	5.98 (1.48)	14.02 (2.09)

$\oplus$ represents the system combination experiments

7.5 Unsupervised Deep Auditory Model (UDAM) using Stacks of ConvRBM

We described the ConvRBM to model 1-D signals, such as speech in Chapter 3. It can be extended to ConvRBM with different subbands, where the input to the visible unit is 2-D as discussed in Section 7.2. The second ConvRBM is stacked on top of first ConvRBM to model a 2-D time-frequency (T-F) representation (i.e., subband filterbank) obtained from the first ConvRBM. The block diagram of our proposed UDAM architecture is shown in Figure 7.4. Let C1 and C2 denote ConvRBMs for first and second layer, respectively. Both the ConvRBMs are trained using CD-1 learning [24] in a greedy layerwise manner. We can write the generalized energy function, hidden, and visible unit activations that represent both the ConvRBMs as follows:

$$\mathbf{I}_k = \sum_{s=1}^{S} \left(\mathbf{x}_s * \tilde{\mathbf{W}}_s{}^{k} \right) + b_k, \tag{7.11}$$

$$\begin{aligned} E(\mathbf{x}, \mathbf{h}) = \frac{1}{\sigma_x^2} \sum_{s=1}^{S} \sum_{i=1}^{n_X} x_{s,i}^2 - \frac{1}{\sigma_x} \sum_{s=1}^{S} \sum_{k=1}^{K} \sum_{j=1}^{l} \sum_{r=1}^{n_W} \left(h_j^k w_{s,r}^k x_{s,j+r-1} \right) \\ - \sum_{k=1}^{K} b_k \sum_{j=1}^{l} h_j^k - \frac{1}{\sigma_x^2} \sum_{s=1}^{S} \sum_{i=1}^{n_X} c x_{s,i}. \end{aligned} \tag{7.12}$$

$$\mathbf{h}^k \sim \max(0, \mathbf{I}_k + \mathcal{N}(0, \sigma(\mathbf{I}_k))). \tag{7.13}$$

In the negative phase, the reconstructed signals are given as:

$$\mathbf{x}_s \sim \mathcal{N} \left(\sum_{k=1}^{K} \left(\mathbf{h}^k * \mathbf{W}_s^k \right) + c_s \right), \text{where} \quad s = 1, ..., S. \tag{7.14}$$

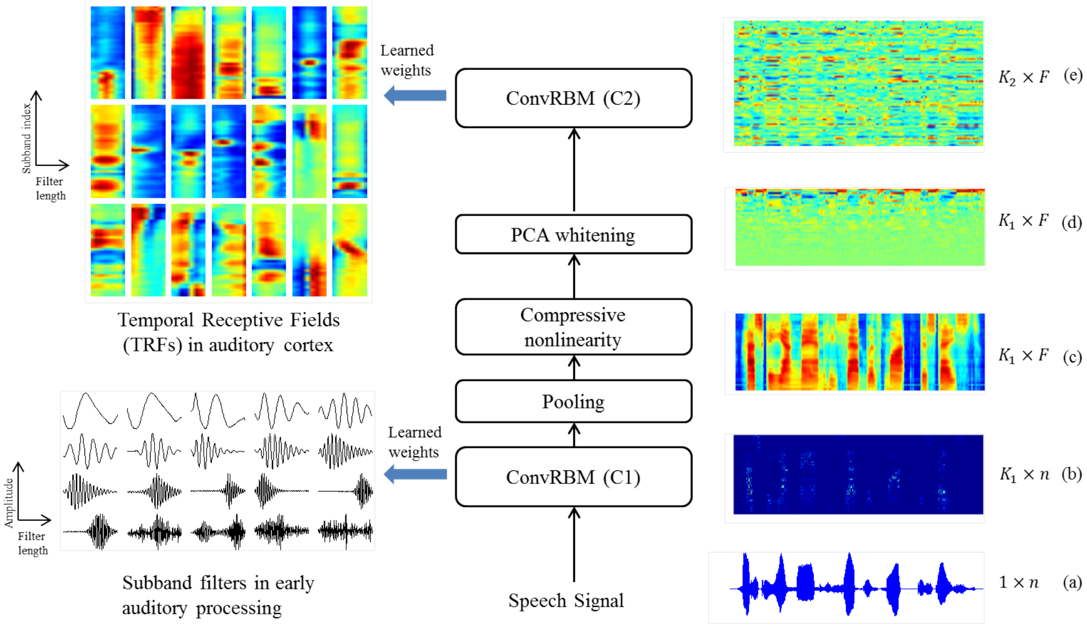


Figure 7.4: Block diagram of the proposed UDAM using ConvRBMs: (a) speech signal, (b) learned subband features of C1, (c) pooled subband signals followed by the compressive nonlinearity, (d) PCA whitening, and (e) learned modulation representation. After [11].

7.5.1 ConvRBM to Model Speech Signals

The input to the ConvRBM (C1) is an entire speech signal of length n samples (i.e., $n_X = n$). Weights of the C1 with length m_1 samples in each are also called subband filters with respect to the speech perception mechanism in hearing [2]. The energy function, activations of hidden and visible units are given by equations (7.11)-(7.14) with $s = 1$, i.e., single channel input to the C1. Convolution with $K = K_1$ subband filters decompose the speech signal into different subbands. Subbands are ordered according to the center frequencies of subband filters (examples are shown in Figure 7.4). The output of C1 is pooled according to a 25-ms window length and 10 ms window shift followed by compressive nonlinearity as shown in Figure 7.4 [2]. The short-time spectral representation is now given as input to C2, which is $K_1 \times F$-dimensional (where F is the number of frames).

7.5.2 ConvRBM to Model the Subband Filterbank

The input $\mathbf{x}$ to the ConvRBM C2 is a T-F representation of a speech signal with $K_1 = S$ subbands, and $n_X = F$ frames, pooled from the C1 responses. Before passing the input to the C2, PCA as a whitening transform is applied as done in [50]. The weights of the C2 are having length m_2 frames. Pooling is not performed after the C2 since we want to keep same number of frames to use as the features concatenated with C1 (i.e., feature-level fusion). The hidden layer has $K = K_2$ groups, which is two times overcomplete (i.e., $K_2 = 2K_1$). Hence, if $K_1 = 40$ subbands, then $K_2 = 80$ groups in C2 resulting in a 120-D feature representation (we kept this to compare the standard 120-D Mel filterbank with 40 filters and their delta features). The summary of notations and the corresponding configurations for both the layers are given in Table 7.4.

Table 7.4: Notations of the UDAM architecture

ConvRBM	Input $\mathbf{x}$	Channels	n_X	n_W	K
C1	speech	$s = 1$	n samples	m_1 samples	K_1
C2	filterbank	$s = 1, ..., S$	F frames	m_2 frames	K_2

The weights learned in C2 are visualized by applying the inverse of PCA whitening on the C2 weights. Examples of TRFs learned on the AURORA 4 database are shown in Figure 7.4, where each block represents one TRF. ConvRBM subband filters capture temporal modulation information with the different subband modulation frequencies from the first layer filterbank. Each subband filter represents the temporal variations in different phonetic units similar to delta features ($\Delta + \Delta\Delta$) of filterbanks.

7.6 Experimental Setup and Results using UDAM

The ASR experiments were performed with the clean and multicondition training databases described in the following sub-sections.

7.6.1 Training of ConvRBMs and Feature Representation

The training parameters for C1 are the same as that used in Chapter 3. The training method of C2 is different from the one used in Section 7.4. For C2, the learning rate was empirically chosen to be 0.005, which is fixed for first 20 epochs and decayed later. Compared to work in [10] and [50], we have not used sparsity regularization, since our model uses ReLUs, which provide sparsity in the hidden units (forcing negative activations to zero). Weights are regularized using weight decay with a factor of 0.0001 (empirically chosen from the range 0.01-0.00001). For comparison with the standard 120-D FBANK feature set, we restrict ourselves to the 40-D filterbank in C1 and 80-D features in C2 giving a 120-D feature vector. The notations for different feature sets are given in Table 7.5.

Table 7.5: Notations of different features used in this study

Description	Notation of Features
Mel filterbank with delta features	FBANK (120-D)
Filterbank from C1	C1 (40-D)
Modulation features from C2	C2 (80-D)
Feature-level fusion of C1 and C2	C1+C2 (120-D)

7.6.2 ASR System Building

The monophone GMM-HMM systems were built using 39-D MFCC for both the databases to generate forced-aligned labels. Language modeling is performed using bi-grams for TIMIT and tri-grams for AURORA 4. In this Chapter, all ASR systems were built using the KALDI speech recognition toolkit [155]. Hybrid DNN-HMM systems were built using fast implementation of p-norm DNNs with $p = 2$ [236] (different from our recent works [2] and [10], where we used DNN with the sigmoid units). ASR system combination (denoted as the $\oplus$ symbol) is performed using the MBR decoding [156] discussed in Section 3.6.3, Chapter 3.

7.6.3 Results on the TIMIT Database

The parameters of the C1 layer are the same as that tuned in [2] with the filter length of m_1=128 samples. To analyze the significance of the second layer C2, we have compared the performance of a single layer C1 filterbank with feature fusion of C1 and C2. The results of these experiments are reported in Table 7.6 in % PER using the hybrid p-norm DNN with parameters (based on a KALDI standard recipe): 2000 hidden units, group size of 5, 2 hidden layers, and context window of 9 frames. From Table 7.6, we can see that by adding delta features in the filterbank features extracted from the C1, we obtained a small relative improvement of 0.9 % compared to the C1. The second layer feature set C2 alone performs better or comparable to the C1 along with their delta features. The filter length of 8 frames in the C2 works better than 6 and 10 frames, when added to C1. It gives a relative improvement of 3.6 % compared to only C1 and 2.73 % compared to the C1 along with their delta features. The FBANK are compared with deep feature set C1+C2 using the same hybrid p-norm DNN of the three hidden layers in Table 7.7. The C1+C2 feature set gives a relative improvement of 5.36 % (1.2 % absolute) on the development set and 2.56 % on the test set compared to the FBANK. The system combination improves performance on the test set only, which is 5.13 % relative to the FBANK.

Table 7.6: % PER for comparison of filter length in C2 and comparison with first-layer features on the TIMIT development set (denoted as Dev). After [11]

ConvRBM Features	Filter Length in C2 (m_2)	Dev
C1 (40-D)	-	22.2
C1+$\Delta + \Delta\Delta$ (120-D)	-	22.0
C2 (80-D)	6	22.0
C2 (80-D)	8	21.8
C1+C2 (120-D)	6	22.1
C1+C2 (120-D)	8	**21.4**
C1+C2 (120-D)	10	21.8

$+$ represents the feature-level fusion experiments

Table 7.7: Results on the TIMIT database in % PER. After [11]

Feature Set	Dev	Test
A: FBANK (120-D)	22.4	23.4
B: C1+C2 (120-D)	21.2	22.8
A $\oplus$ B	**21.2**	**22.2**

$\oplus$ represents the system combination experiments

7.6.4 Results on the AURORA 4 Database

The % WER of experiments for the AURORA 4 task are reported in Table 7.8 using a hybrid p-norm DNN with parameters, namely, 2000 hidden units, group size of 5, 2 hidden layers, and a 9-frame context window. For C2, a filter length of 10 frames performs better compared to the 8 frames in the TIMIT database. The use of the second-layer feature set C2 improves performance compared to the C1 alone as well as addition of delta features in the C1. Specifically, for the noisy test sets (i.e., C and D), using the C1+C2, there is an absolute reduction of 1.12-1.42 % in WER over C1 and 0.63-1.22 % in WER over C1+$\Delta + \Delta\Delta$. Finally, the C1+C2 feature set is compared to the FBANK in Table 7.9 with three layer p-norm DNN. An absolute reduction of 1-2 % in WER is obtained using the C1+C2 feature set compared to FBANK. The fusion of both C1 and C2 feature sets performs better than the C1 and C2 alone. System combination further reduces WER (except in test set A), with significant reduction for the test sets C and D compared to the FBANK and C1+C2 feature sets. Hence, both the feature sets contain complementary information.

Table 7.8: Comparison of filter length in C2 for the AURORA 4 database in % WER. After [11]

Features Set	A	B	C	D	Avg
C1	9.36	17.90	22.64	34.66	21.14
C1+$\Delta + \Delta\Delta$	9	17.05	22.44	33.19	20.42
C1+C2, m_2=6	9.25	17.18	22.08	33.29	20.45
C1+C2, m_2=8	**8.91**	17.25	22.17	33.47	20.45
C1+C2, m_2=10	9.1	**16.97**	**21.22**	**32.54**	**19.96**
C2, m_2=10	8.87	18.37	23.48	34.4	21.28

$+$ represents the feature-level fusion experiments

Table 7.9: Results on the AURORA 4 database in % WER. After [11]

Features Set (120-D)	A	B	C	D	Avg
A: FBANK	10.41	18.16	22.45	34.09	21.28
B: C1+C2	**8.37**	16.89	20.96	33.04	19.82
A $\oplus$ B	8.56	**16.14**	**19.73**	**32.07**	**19.12**

$\oplus$ represents the system combination experiments

7.7 Improved UDAM

To improve the UDAM model, we used annealing dropout and Adam optimization in both the ConvRBMs. Another change we have made is to replace PCA whitening with the mean-variance normalization across subbands. This works very well with an Adam optimization and avoids an extra PCA whitening processing step. Specifically, for the AURORA 4 task, we used the TEO-based ConvRBM representation presented in Section 4.5, Chapter 4. For the WSJ task, we used a ConvRBM feature representation without TEO. The subband filters learned from the AURORA 4 database are shown in Figure 7.5. One can see that the TRFs exhibits similar patterns as discussed in Section 7.3.1.

7.7.1 Experimental Setup

The ASR experiments were performed using the BLSTM acoustic models with 800 hidden units and 3 hidden layers (based on a standard recipe in KALDI and earlier experiments in Chapter 4 and Chapter 5). In addition, based on very recent KALDI setups, we used LF-MMI for a sequence-to-sequence learning in the hybrid DNN-HMM framework. The initial alignments were obtained from the LDA+MLLT+fMLLR transform as per the KALDI recipe for the AURORA 4 and WSJ tasks. The language models were the same used in both the ASR tasks in Chapter 3. The BLSTM models are used in the LF-MMI sequence learning framework with similar parameters as used previously.

7.7.2 Experimental Results

The results on the AURORA 4 database are shown in Table 7.10 for BLSTM models. The first-layer feature set C1+$\Delta + \Delta\Delta$ is compared with the second-layer feature set C2 with various filter lengths (in frames). The filter length of 3 frames performed better compared to other lengths. This shows that second-layer ConvRBM requires small filter length to extract the temporal modulation information. The feature-level fusion of C1+C2 performed better compared to the C1+$\Delta + \Delta$-Δ, except the test set C. Hence, an improved UDAM performed slightly better in the ASR task instead of using only the filterbank obtained from the single ConvRBM.

The results on the AURORA 4 task using the LF-MMI-based BLSTM models are shown in Table 7.11. Compared to our model in Chapter 3, an improved ConvRBM (discussed in Chapter 4) performed significantly better than FBANK

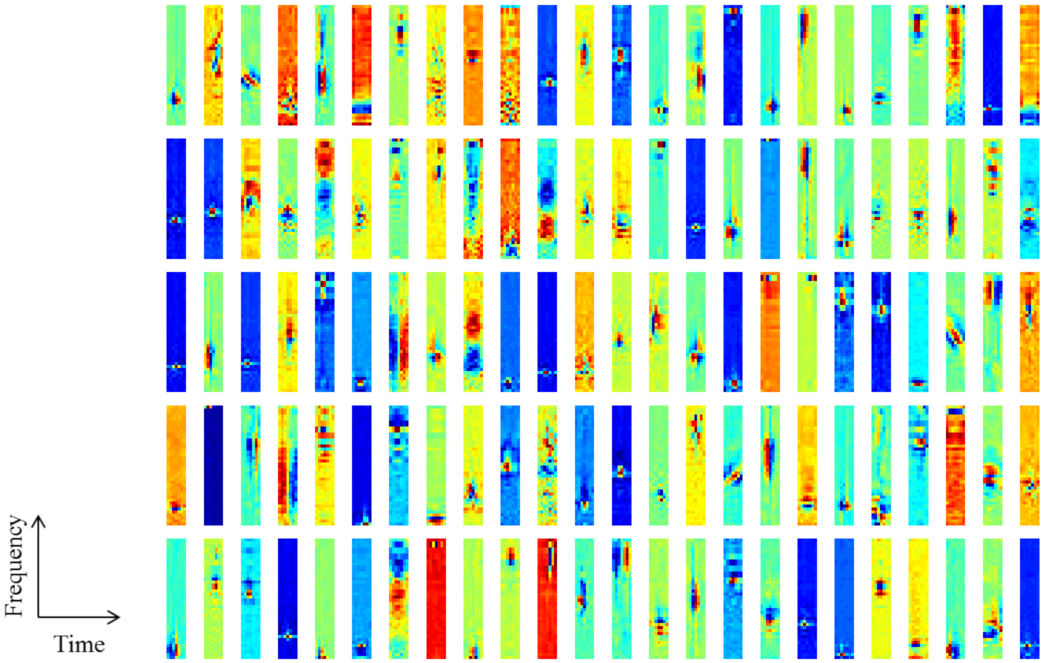


Figure 7.5: The subband filters trained in the second ConvRBM in the UDAM.

Table 7.10: The experiments using improved UDAM and hybrid HMM-BLSTM models on the AURORA 4 database in % WER

Features Set	A	B	C	D	Avg
C1+$\Delta + \Delta\Delta$	9.46	14.13	14.34	24.60	18.30
C2, m_2=3	9.79	14.06	14.39	24.38	18.19
C2, m_2=6	10.31	14.02	14.17	24.77	18.38
C2, m_2=10	10.02	14.25	**14.12**	24.71	18.42
C1+C2 m_2=3	**9.2**	**13.79**	14.52	**24.38**	**18.06**

$+$ represents the feature-level fusion experiments

(relative reduction of 1.2-8.84 % in WER). The second layer C2 is trained using fixed 120 groups (i.e., number of filters) and filter size is varied as $m_2 = 3, 5, 7$. The smaller filter size $m_2 = 3$ of C2 performed better compared to the larger one, which is a different observation shown in Table 7.6 and Table 7.8. Hence, C2 with $m_2 = 3$ is used for feature fusion C1+C2 and system combination experiments. The system combination of FBANK and C1+$\Delta + \Delta\Delta$ (S1$\oplus$S2) significantly reduces % WER (12.1-23.11 relative) compared with both the systems individually. Specifically, on the noisy test sets C and D, S1$\oplus$S2 give an absolute reduction of 2.58 and 3.53 in % WER, respectively. The system combination of C1 and C2 (S2$\oplus$S3) also significantly reduces % WER (12.01-24.88 relative) compared to both the systems individually. On the noisy test sets C and D, S2$\oplus$S3 gives an absolute reduction of 2.47 and 3.8 in % WER, respectively. On an average, S1$\oplus$S2 and S2$\oplus$S3 give relative reductions of 14.75 and 15.43, respectively. The system combination of FBANK and C1+C2 did not perform well compared to S1⊕S2 and S2⊕S3. Hence, the system combination S2⊕S3 performed better compared to all the systems.

Table 7.11: The experiments using improved UDAM and LF-MMI-based BLSTM models on the AURORA 4 database in % WER

Features Set	A	B	C	D	Avg
S1:FBANK	8.48	11.57	12.76	23.44	16.52
C1+$\Delta + \Delta\Delta$ (Chapter 3 model)	8.14	12.73	13.19	23.77	17.16
S2:C1+$\Delta + \Delta\Delta$	7.73	11.43	12.22	22.42	15.93
S3:C2, m_2=3	7.58	12.15	11.9	23.07	16.49
C2, m_2=5	7.57	12.42	11.94	23.70	16.87
C2, m_2=7	7.14	12.49	11.64	23.36	16.71
S4:C1+C2, m_2=3	7.36	11.92	11.97	23.08	16.38
S1⊕S2	6.52	10.17	**10.18**	19.91	14.08
S2⊕S3	**6.37**	10.18	10.29	**19.64**	**13.97**
S1⊕S4	6.69	**10.07**	10.7	20.56	14.37

+ represents the feature-level fusion experiments
⊕ represents the system combination experiments

Table 7.12: The experiments using improved UDAM and LF-MMI-based BLSTM models on the full WSJ database in % WER

Feature Set	D1	D2	E1	E2	E3	E4
S1:FBANK	9.30	3.81	7.13	3.14	6.70	1.57
S2:C1+$\Delta + \Delta\Delta$	9.12	3.73	8.94	2.99	6.47	1.48
S3:C2	8.97	3.85	7.92	3.45	6.03	1.77
S4:C1+C2	8.65	3.41	7.57	2.99	6.36	1.57
S1⊕S2	8.31	**3.21**	7.28	3.01	6.22	**1.27**
S2⊕S3	8.23	3.23	7.16	3.04	5.97	1.46
S1⊕S4	**8.21**	3.44	**6.84**	**2.78**	**5.94**	1.38

+ represents the feature-level fusion experiments
⊕ represents the system combination experiments

The improved UDAM is trained using the full WSJ training database. The first layer C1 has 40 subband filters each with a length of 128 samples. The second layer C2 has 120 modulation filters each with a length of 3 frames (chosen based on experiments on the AURORA 4 database). The experimental results on the full WSJ database are shown in Table 7.12. The C1+$\Delta + \Delta\Delta$ feature set obtained reduced % WER compared to the FBANK except the test set E1. The C2 feature set reduce the % WER on the 20K test sets compared to the FBANK and C1+$\Delta + \Delta\Delta$ feature sets. The feature-level combination C1+C2 performed better or equal to the C1+$\Delta + \Delta\Delta$, C2 and FBANK. The system combination of FBANK with C1 (S1$\oplus$S2) and C1+$\Delta + \Delta\Delta$ with C2 (S2$\oplus$S3) performed better than the individual systems except E1 and E2 test sets. The system combination of FBANK with C1+C2 (S1$\oplus$S4) performs significantly well in 20K vocabulary test sets, D1, E1 and E3, respectively. S1$\oplus$S4 gives a relative reduction of 11.72, 4.24, and 11.34 % in WER for D1, E1, and E3 test sets, respectively compared to the FBANK. Hence, the UDAM-based features C1+C2 when used in a system combination framework performs well in all the test sets compared to the FBANK.

7.8 Chapter Summary

In this chapter, we discussed our proposed model for the temporal modulation filterbank learning. Our initial attempt is to use the Mel spectrogram as an input to ConvRBM. Analysis of the modulation filters shows that it represents temporal receptive fields. Inspired by this model, we proposed a two-layer deep auditory model using the stack of ConvRBM, where first one learns the filterbank from the speech signals and later one learns TRFs from the filterbank of the first ConvRBM. The experiments on the ASR task showed the improved performance compared to the baseline system. In the next chapter, we summarize the entire thesis and present some future research directions.

CHAPTER 8
Summary and Conclusions

In this chapter, a summary of this thesis work is presented along with the limitation of the current work and future research directions.

8.1 Summary of the Thesis

The following is a summary of the research work done in the entire thesis:

- In this thesis work, a novel auditory representation learning framework is presented. The model is based on unsupervised representation learning using a Convolutional Restricted Boltzmann Machine (ConvRBM). In particular, we proposed to model raw speech signals with arbitrary lengths. Compared to earlier works on feature learning using ConvRBM, we proposed to use noisy rectified linear units (NReLU) for inference from the hidden units. The model is successfully applied for various speech and audio processing applications.

- The background studies required to understand our proposed model and applications are discussed in Chapter 2. The detailed architecture of the proposed model and the mathematical derivation for learning parameters are presented in Chapter 3. The model is first trained with standard databases in the ASR task. The subband filter shapes (i.e., impulse responses) and frequency responses are analyzed and compared with the physiological filters. The frequency scale obtained from the ConvRBM is similar to the standard auditory frequency scales. However, the subband filter shapes and center frequencies are adapted according to the statistics of the database. The features extracted from the ConvRBM filterbank are applied for the clean and noisy ASR tasks. The proposed feature set performed very well compared to the Mel filterbank (which is a standard auditory-based feature). The performance improvements in the noisy ASR task were justified using the Lipschitz continuity condition applied on ConvRBM for stability to additive noise. The convolution and ReLU transformations in ConvRBM are stable to additive noise.

- To further improve our proposed model, we presented approaches to enhance the training of ConvRBM using Adam optimization in Chapter 4. The annealing dropout regularization was also added to prevent co-adaptation of weights in ConvRBM. We observed that the filterbank learned using an Adam and dropout are more smooth and averaged RMSE were lower during ConvRBM training. For the noise-robust ASR task, nonlinear energy estimation using Teager Energy Operator (TEO) was applied in the ConvRBM-based feature representation. The proposed amalgamation of signal processing and machine learning for auditory feature representation improved the ASR task in noisy environments. The statistical significance of the proposed approach was evaluated using the bootstrap method to estimate the % probability of improvement (POI).

- As a special case study, we applied our ConvRBM filterbank learning model in ASR for the agricultural-domain in the Gujarati language, a MeitY, Government of India supported consortium project. The speech data is collected from the 21 districts of Gujarat state with other project staff members and volunteers from the Speech Research Lab, DA-IICT. The ConvRBM is trained with the speech database collected from the real noisy environments and various dialectal regions of Gujarat state. The ConvRBM filterbank-based features significantly reduce the % WER in this ASR task.

- After successful application of ConvRBM filterbank learning framework using speech signals, we have also applied in other audio processing applications. To show our model's capability to adapt diverse sound classes, it is applied in the Environmental Sound Classification (ESC) task. The proposed features perform significantly better compared to the Mel filterbank and earlier approaches using supervised CNNs. The next application was the spoof speech detection (SSD) task, which is a part of the recent ASV system. Two different spoof speech types were considered, namely, synthetic and replay speech that were based on the ASVSpoof challenges organized in 2015 and 2017. Specifically, for the replay speech detection, we applied pre-emphasized speech signals in the ConvRBM training. Pre-emphasis leads to the more subband filters representing high frequency regions. Such an approach significantly reduced equal error rate compared to the baseline and many approaches in the literature. Finally, ConvRBM is applied in a socially-relevant problem of Infant Cry Classification (ICC). Our proposed ConvRBM-based features are able to better classify healthy *vs*. pathological cry signals using various performance measures even with the filterbank learned using 30 minutes of the cry signals.

- Finally, an extension of the ConvRBM to learn temporal receptive fields is presented by stacking the two ConvRBMs together. We refer to this new model as the unsupervised deep auditory model (UDAM). UDAM is applied to the ASR task and showed improvements compared to the Mel filterbank as well as single layer ConvRBMs. Hence, our proposed ARL model learns subband filters similar to the cochlear filter responses from the speech signals and when stacked by another ConvRBM, it learns the temporal receptive field that extracts the modulation information.

The MATLAB codes for ConvRBM and some pre-trained filterbanks (weights of ConvRBM) can be obtained from my homepage:

URL: https://sites.google.com/site/hardik89sailor/publications

8.2 Limitations of the Current Work

Our proposed model is one of the contributions towards the research in the auditory representation learning. However, there are certain limitations of our approach as described below:

- First of all, compared to the physiological auditory models [49], [169], a few aspects of the auditory processing are either approximated by the linear system or ignored. We have assumed the cochlear signal processing as an LTI system and hence, a convolutional model is used in ConvRBM. Auditory nonlinearities, such as sound level-dependent gain control or automatic gain control (AGC) [237], are not incorporated in our model.

- In central auditory processing, our model currently represents only temporal modulations. However, spectro-temporal receptive fields (STRFs) are not considered in our model, ignoring spectral processing on the subbands in the second ConvRBM.

- The inference in ConvRBM is based on the reliability of samples estimated from the Gibbs sampling technique in the CD-1 stage. However, it may be possible that the CD-1 estimate of the gradients does not always follow the actual gradient direction. Hence, there is a need of using Bayesian techniques, such as variational-likelihood, to be incorporated in the ConvRBM training.

8.3 Future Research Directions

Future research directions include possible solutions to the above mentioned limitations and further advancements in our proposed model as described below:

- One of the important extensions of our model is to represent the binaural sounds, i.e., mimicking the auditory processing that involves two ears. Such modeling will be helpful in spatial hearing tasks, such as sound localization, source separation, etc. We can also extend it to model the multiple channels of speech for application in the distant ASR task, where a microphone array is used. For example, in our very recent work, we developed ConvRBM to model the two-channel speech signals. The model is developed in a spirit of acoustic beamforming, i.e., combining two channels into a single channel using the weights learned in ConvRBM. The examples of subband filters trained using channels 1 and 2 from the CHiME 4 database are shown in Figure 8.1. The difference in the amplitude and phase can be seen in time-domain impulse responses. In binaural speech terminology, they are called the Interaural Time Differences (ITD) and Interaural Level Differences (ILD) [238]. The ITD is related to the difference in the arrival time of speech signals in the ears that results in a delay in the BM vibrations in one ear relative to the other [238]. The ILD is related to differences in amplitude for frequencies higher than 2 kHz that result in differences in intensity of the speech signals [238]. Our future work is to develop binaural ConvRBM that can be possibly used for noise-robust speech recognition, speech separation and sound localization.

- In order to make our model more close to the human auditory processing, our future work is to incorporate nonlinear auditory processing stages, such as Automatic Gain Control (AGC) and synaptic depression, in the ConvRBM training as done in [239].

- Spectro-temporal Receptive Field (STRF) learning in the ConvRBM to jointly

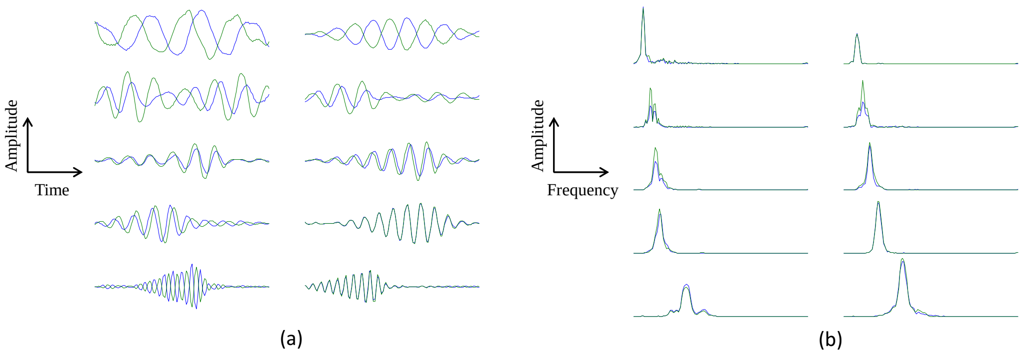


Figure 8.1: Subband filters from the two channel ConvRBM: (a) time-domain impulse responses, and (b) frequency-domain responses.

represent spectro-temporal modulations inspired by the phenomenal auditory model [49].

- We can improve the learning in ConvRBM using many recent advancements for RBM, such as stochastic spectral descent for parameter updates, variational inference instead of CD-1 learning, etc.

- Our auditory system is continuously adapting to a variety of different sounds from simple tones to more complex sounds, such as songs (that has music and human vocals). As a generalized auditory model, we would like to train ConvRBM on the ensemble of sounds that includes speech, environmental sounds, and music. The analysis of the filterbanks trained from different categories of sounds may reveal interesting auditory representations.

Appendix A. Significance of Temporal Information in Filterbank Learning

In order to justify the significance of temporal context in learning the ConvRBM, we used an audio scrambling technique with the MATLAB code obtained from [240]. It works by scrambling an audio file by moving around short, 50 % overlapping windows within a local window. They can be used to create new versions of existing recordings that preserve the spectral content over longer time scales, but remove the structure at shorter time scales. This can be useful, e.g., for making the speech unintelligible. The example of a scrambled speech signal from the TIMIT database is shown in Figure A.1. One can see from Figure A.1 that the scrambled speech signal looks like random noise and spectral content in the shorter time scales is destroyed. The ConvRBM is trained with such scrambled speech signals taken from the TIMIT database. The resulting subband filters are shown in Figure A.2. The lower frequency subband filters are impulse-like signals while higher frequency filters are wavelet-like basis functions. Since the temporal structure is

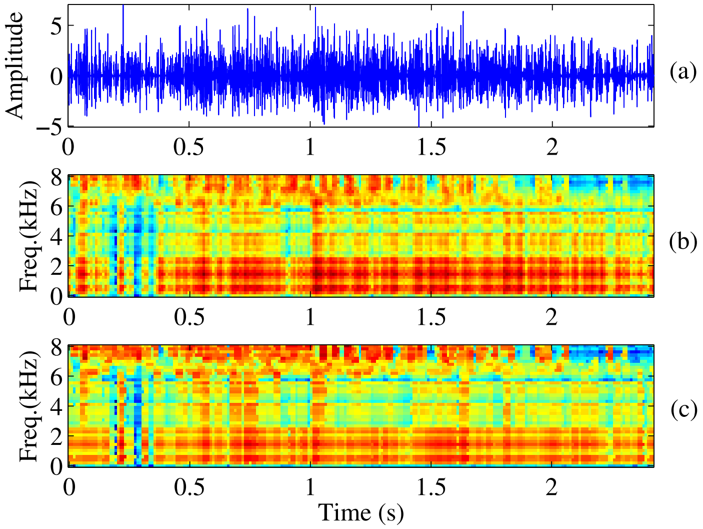


Figure A.1: (a) Time-scrambled speech signal, (b) Mel spectrogram, and (c) ConvRBM spectrogram. The utterance is: “Spring Street is straight ahead".

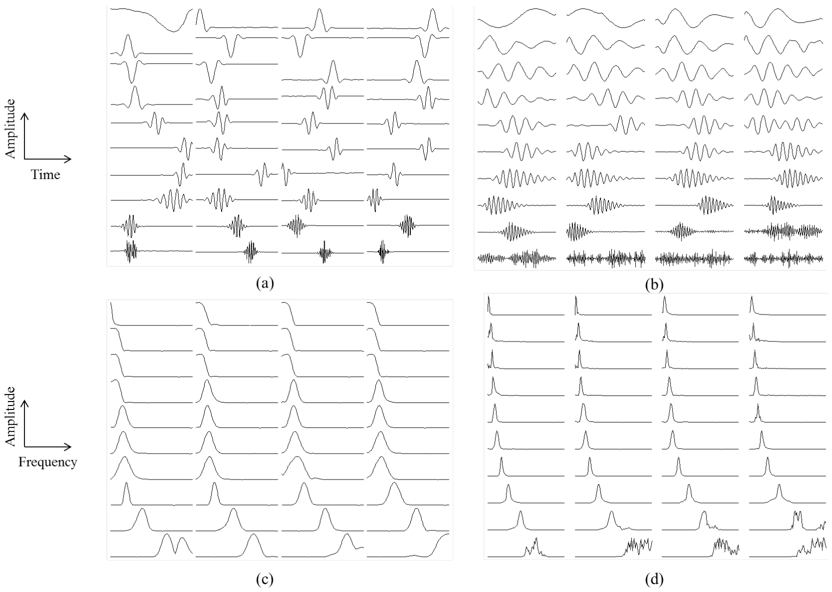


Figure A.2: (a), (c) Subband filters trained on time-scrambled speech signals and (b), (d) trained without time-scrambled speech signals.

destroyed, the speech signals have random transient-like sound events abruptly occurring at many places. Hence, the model tries to fit the optimal weights for such signals. Since there are short duration wavelet-like impulse responses, the frequency responses show that the subband filters are not localized as compared to the subband filters obtained from without scrambling. Hence, the temporal information in the speech signals such as the context of phonetic sounds, formant contours, etc. has a significant effect on filterbank learning.

To show that ConvRBM subbands represent the temporal modulations, we have shown the three kinds of temporal modulations at various scales in Figure A.3. The speech signal is convoluted with the ConvRBM subband filter with the center frequency 700 Hz as shown in Figure A.3 (a). The segments in the subband filtered signal roughly correlate with the different syllabic segments of an utterance. The envelope obtained from the short-time averaging (as done in our proposed feature representation in Section 3.4, Chapter 3) or by taking the Hilbert transform, called the slow temporal modulations. At the intermediate temporal scale, the temporal modulations (sharp peaks) due to the interharmonic interactions reflect the fundamental frequency (F_0) of the signal, as shown in Figure A.3 (b). The rapid temporal modulations at very fine scale in Figure A.3 (c) are due to the frequency component driving this subband best around 700 Hz. These are also called the Temporal Fine Structure (TFS). This analysis of the ConvRBM filterbank has similar insights as obtained in [49]. Hence, when we abruptly destroy such an important temporal information, the ConvRBM is not able to learn auditory-like subband filters as shown in Figure A.2.

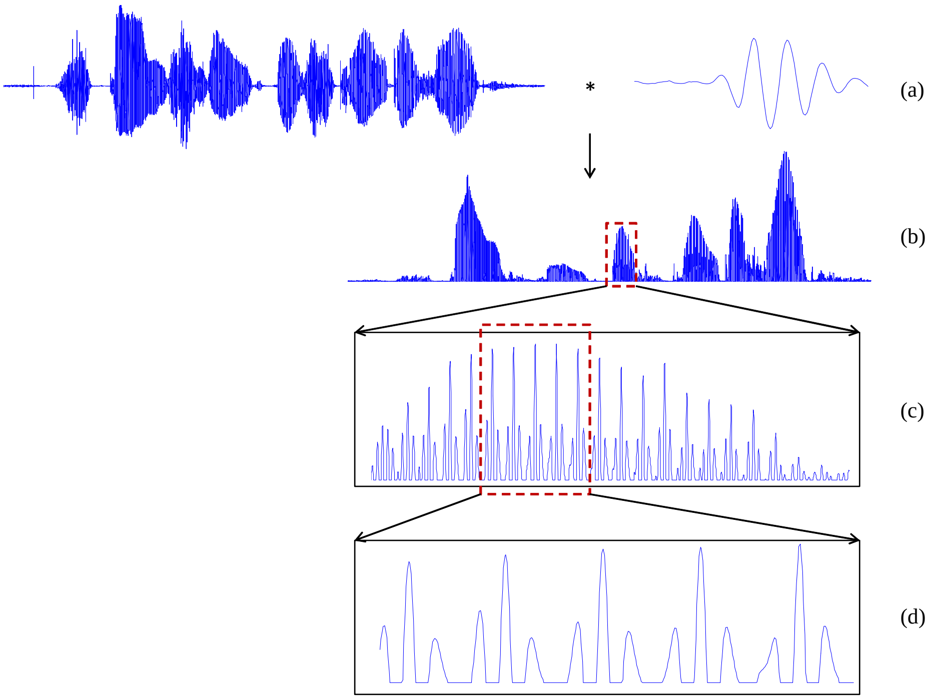


Figure A.3: The temporal modulations in the speech signal (a) convolution of speech signal with the ConvRBM subband filter, (b) subband filtered signal that has slow temporal modulations, (c) temporal modulations due to interharmonic interactions, and (d) Fast temporal modulations.

Appendix B. Performance Measures

B.1 Word Error Rate (WER)

The standard performance metric for automatic speech recognition (ASR) systems is the Word Error Rate (WER) [86]. The WER is computed for the decoded word sequence in the ASR output against the reference transcription. The % WER is defined as follows [86]:

$$WER = \frac{S + D + I}{N} \times 100, \tag{B.1}$$

where
S = Number of substitutions (one word is replaced with another one),
D = Number of deletions (word is missed out),
I = Number of insertions (word is added),
N = Total number of words in the reference transcription

In the case of the phone recognition task, the reference is the phonetic transcription (not word-level) and the ASR decoder also produces phone sequences in the output. In such a case, the same performance measure is applied; however, instead of words, we use phones. Hence, it is also called the % Phone Error Rate (PER).

B.2 Classification Accuracy

The performance of the classification task is measured by the classification accuracy. If $\hat{z}_i$ is the predicted value of the i^{th} sample, and z_i is the corresponding true value, then the % classification accuracy (the fraction of correct prediction) over a total of N samples is defined as [241]:

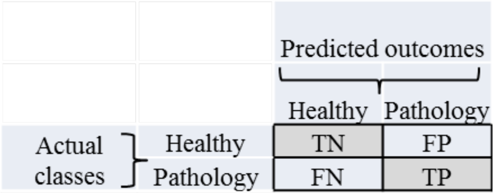
$$\text{\% Classification Accuracy} = \frac{1}{N} \sum_{i=0}^{N-1} \mathbb{I}(\hat{z}_i = z_i) \times 100, \tag{B.2}$$

where $\mathbb{I}(\cdot)$ is an indicator function with $\mathbb{I}$=1 when $\hat{z}_i = z_i$, otherwise $\mathbb{I}$=0.

B.3 Performance Measures from Confusion Matrix

The confusion matrix of a binary classification task shows how errors are distributed across the classes [242]. The example of a confusion matrix for a classification task is shown in Figure B.1 for healthy *vs.* pathological infant cry. The rows indicate the actual classes and columns indicate the predicted outcome of the pattern classifier [242]. Since our task is to detect the pathology in an infant cry, we denote the results associated with pathology as positive and healthy as negative. Given the labels of actual and predicted classes by the classifier, there are four outcomes possible [242]:

- True positive (TP): Actual class is pathology and predicted pathology
- True negative (TN): Actual class is healthy and predicted healthy
- False positive (FP): Actual class is healthy and predicted pathology
- False negative (FN): Actual class is pathology and predicted healthy



		Predicted outcomes	
		Healthy	Pathology
Actual classes	Healthy	TN	FP
	Pathology	FN	TP

Figure B.1: The details of a confusion matrix for the binary classification task.

In the case of k-fold CV, we find the combined confusion matrix (i.e., all the entries in the matrix are summed for all the folds). Various other performance measures can be obtained from the confusion matrix. The numbers along the major diagonal indicates (TP and TN) the correct decisions made by the classifier [242]. The classification accuracy can also be obtained from TP, TN and a total number of instance of both the classes (i.e., P+N) as follows [242]:

$$\text{Classification accuracy } (\%) = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}}. \tag{B.3}$$

Another important performance measure is the F1-score, also known as F-measure. The range of F-measure is between 1 and 0, where 1 represents the perfect prediction and 0 means the worst. The F-measure is defined as follows [242]:

$$\text{F-measure} = \frac{2\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}}. \tag{B.4}$$

The F-measure does not take TN into account. Hence, we also used another performance measure called Youden's J-statistic or informedness [243]. The range of the J-statistic is between -1 and +1, where -1 indicates no agreement between the observation and the prediction, and +1 represents a perfect prediction. The J-statistic estimates the probability of an informed decision and is given by [243]:

$$\text{J-statistic} = \frac{\text{TP}}{\text{TP} + \text{FN}} + \frac{\text{TN}}{\text{TN} + \text{FP}} - 1. \tag{B.5}$$

Another important performance measure is the Matthews Correlation Coefficient (MCC) [244]. It takes into account TP, TN, FP, FN and is generally regarded as a balanced measure that can be used even if the classes are of very different sizes. The range of MCC is between -1 and +1, where +1 indicates a perfect prediction, 0 means no better than just a random prediction, and -1 indicates a total disagreement between the observation and the prediction. MCC is expressed as [244]:

$$\text{MCC} = \frac{(\text{TP} \times \text{TN}) - (\text{FP} \times \text{FN})}{\sqrt{(\text{TP} + \text{FP})(\text{TN} + \text{FN})(\text{TP} + \text{FN})(\text{TN} + \text{FP})}}. \tag{B.6}$$

B.4 Equal Error Rate (EER)

A detection task or classification task can also be viewed as involving a trade-off between the two types of errors, namely, miss detection and false alarm. The miss detection or the False Rejection Rate (FRR) is the probability that the classifier fails to detect a match between the input pattern and a matching class in the database [35]. FRR measures the percentage of valid inputs that are incorrectly rejected in the classification task. The false alarm or the False Acceptance Rate (FAR) is the probability that the classifier incorrectly matches the input pattern to a non-matching class in the database [35]. FAR measures the percent of invalid inputs that are incorrectly accepted in the classification task. A detection error trade-off (DET) graph is a graphical plot of error rates for binary classification systems, plotting the FRR *vs*. FAR [35]. Since FAR and FRR are opposite functions (when one monotonically increases, the other decreases and vice-versa), there is a trade-off in the error reduction in the detection task and hence the name DET curve. The point where FRR and FAR are equal is called as the % Equal Error Rate (EER), which is generally used as the performance measure. An example of the DET curve is shown in Figure B.2 for the SSD task. A lower FAR means higher security against spoof speech. A lower FRR means higher convenience of the system performance.

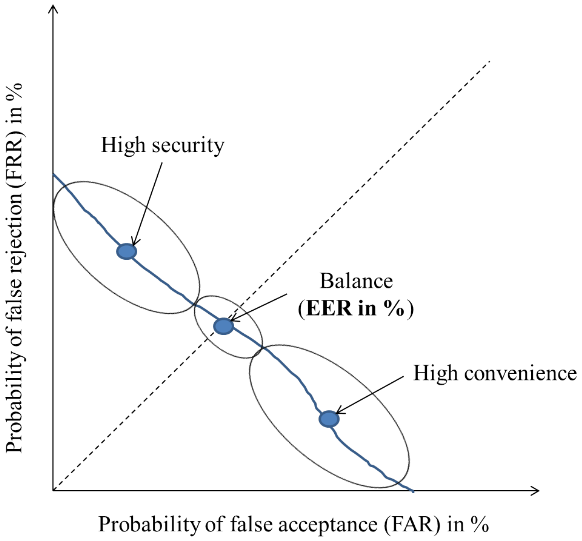


Figure B.2: An example of the DET curve. After [35].

B.5 Statistical Significance of Results Using Bootstrap

Statistical performance analysis is a challenging task, when the numbers of observations are small, and repeating the experiments is difficult. The test sets for the ASR task are fixed (e.g., WSJ and AURORA 4 test sets were decided by the ARPA evaluation framework), and repeating experiments on LVCSR task is time consuming. The bootstrap is a technique that does with a computer what the experimenter would do in practice if it were feasible: repeating the experiment several times [36]. The two main advantages of a bootstrap technique are: (1) recomputations are done for a large number of times, (2) does not require large number of observations, since it create them. The core idea of the bootstrap is to create the replication of the statistic by a random sampling from the dataset with replacement. The principle of the non-parametric bootstrap is presented in Figure B.3. The basic principle of the bootstrap technique is presented in Algorithm 3.

Algorithm 3 The bootstrap algorithm. After [36].

```
Input: Original data samples $X$
Output: Bootstrap estimates
for each bootstrap interval, $b = 1, 2, ..., B$ do
    Generate a random sample with replacement $X_b^*$ of size $X$
    Compute the estimate $\hat{\theta}_b^*$ for each bootstrap sample
end for
$\hat{\theta}_b^*$ for $b = 1, 2, ..., B$ are the bootstrap estimates
Estimate confidence intervals, histogram of estimates, probability statements
```

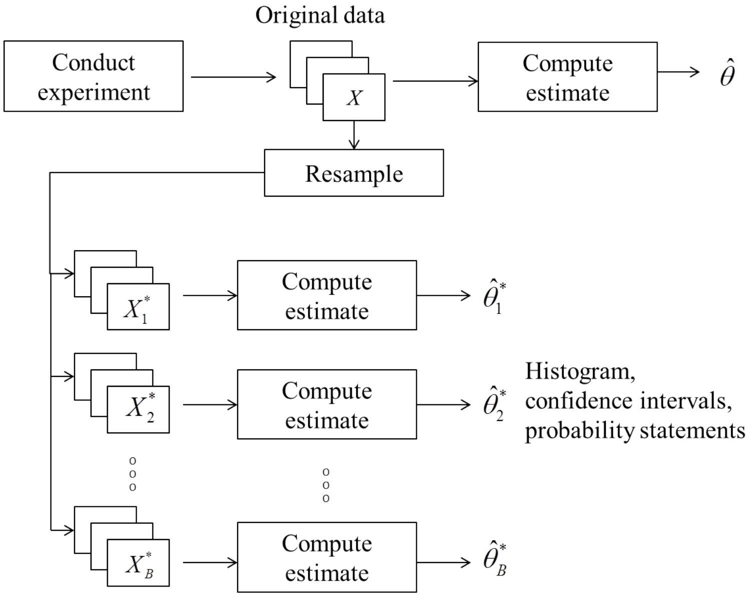


Figure B.3: Principle of the non-parametric bootstrap technique. Adapted from [36].

Appendix C. Noise Suppression Using the Teager Energy Operator (TEO)

The Teager Energy Operator (TEO) has the noise suppression capability first analyzed in [167] for speech recognition applications in the car noise scenario in [245] for epoch estimation using various noises, and for person recognition in noisy environments [246]. Here, we discuss the noise suppression capability of TEO for the additive noise case. Let $x[n]$ and $\hat{x}[n] = x[n] + v[n]$ be clean, and noisy speech signal, where $v[n]$ is a zero-mean additive noise signal. The TEO profiles for $x[n]$ and $v[n]$ are given as:

$$\Psi\{x[n]\} = x^2[n] - x[n-1]x[n+1], \tag{C.1}$$

$$\Psi\{v[n]\} = v^2[n] - v[n-1]v[n+1]. \tag{C.2}$$

The TEO profile for the noisy speech signal $\hat{x}[n]$ is calculated as:

$$\begin{aligned}\Psi\{\hat{x}[n]\} &= \hat{x}^2[n] - \hat{x}[n-1]\hat{x}[n+1],\\ &= (x[n]+v[n])^2 - (x[n-1]+v[n-1])(x[n+1]+v[n+1]),\\ &= x^2[n] + 2x[n]v[n] + v^2[n] - x[n-1]x[n+1] - x[n-1]v[n+1]\\ &\quad - v[n-1]x[n+1] - v[n-1]v[n+1].\end{aligned} \tag{C.3}$$

Rearranging the above terms and using eq. (C.1) and eq. (C.2), we get

$$\Psi\{\hat{x}[n]\} = \Psi\{x[n]\} + \Psi\{v[n]\} + 2\hat{\Psi}\{x[n], v[n]\}, \tag{C.4}$$

where $\hat{\Psi}\{x[n], v[n]\}$ is called the cross-TEO between $x[n]$ and $v[n]$, which is given by:

$$\hat{\Psi}\{x[n], v[n]\} = x[n]v[n] - \frac{1}{2}x[n-1]v[n+1] - \frac{1}{2}x[n+1]v[n-1]. \tag{C.5}$$

Considering $x[n]$ and $v[n]$ as random variables, the expected value of the TEO is given as:

$$\mathbb{E}\left[\Psi\{\hat{x}[n]\}\right] = \mathbb{E}\left[\Psi\{x[n]\}\right] + \mathbb{E}\left[\Psi\{v[n]\}\right] + 2\mathbb{E}\left[\hat{\Psi}\{x[n], v[n]\}\right], \tag{C.6}$$

where $\mathbb{E}[\cdot]$ is an expectation operator. Since $v[n]$ is a zero-mean additive noise, and $x[n]$ and $v[n]$ are statistically independent so that $\mathbb{E}\left[\hat{\Psi}\{x[n], v[n]\}\right] = 0$ and hence:

$$\begin{aligned}\mathbb{E}\left[\hat{\Psi}\{x[n], v[n]\}\right] = \mathbb{E}\left[x[n]v[n]\right] - \frac{1}{2}\mathbb{E}\left[x[n-1]v[n+1]\right] \\ - \frac{1}{2}\mathbb{E}\left[x[n+1]v[n-1]\right],\end{aligned} \tag{C.7}$$

Here, $\mathbb{E}\left[x[n]v[n]\right] = \mathbb{E}\left[x[n]\right]\mathbb{E}\left[v[n]\right] = 0$, since $\mathbb{E}[v[n]] = 0$, and similarly for other two terms in the above equation. Hence,

$$\mathbb{E}\left[\Psi\{\hat{x}[n]\}\right] = \mathbb{E}\left[\Psi\{x[n]\}\right] + \mathbb{E}\left[\Psi\{v[n]\}\right]. \tag{C.8}$$

The expected values in eq. (C.8) can also be represented in terms of autocorrelation as follows:

$$\mathbb{E}\left[\Psi\{\hat{x}[n]\}\right] = R_{xx}(0) - R_{xx}(2) + R_{vv}(0) - R_{vv}(2), \tag{C.9}$$

where $R_{xx}(\tau) = \mathbb{E}[x[n]x[n-\tau]]$ and $R_{vv}(\tau) = \mathbb{E}[v[n]v[n-\tau]]$ are autocorrelation functions of clean and noise signals, respectively. It is experimentally verified in [167] and [245] that, when the TEO is applied on the noise signal, $R_{vv}(0) - R_{vv}(2) \approx 0$. Hence, it can be proved that:

$$\mathbb{E}\left[\Psi\{\hat{x}[n]\}\right] \approx \mathbb{E}\left[\Psi\{x[n]\}\right]. \tag{C.10}$$

The eq. (C.10) indicates that TEO when applied on the noisy signal, with the additive zero-mean noise, can suppress the noise and hence, TEO has the *noise suppression* capability.

Appendix D. Lipschitz Continuity Condition

In order to define the Lipschitz continuity condition, the basic definitions of continuous mappings between two metric spaces are given. The continuity for the real-valued functions of a real variable are defined as follows [149]:

Definition 1: A mapping T from a metric space X to a metric space Y is said to be continuous at the point $x \in X$ provided for any sequence $\{x_n\}$ in X,

$$\text{if } \{x_n\} \to x, \text{ then } \{T(x_n)\} \to T(x). \tag{D.1}$$

The mapping T is said to be continuous if it is continous at every point in X.

The $\epsilon - \delta$ criterion for continuity: A mapping T from a metric space (X, ρ) to (Y, σ) is continous at the point $x \in X$ if and only if for every $\epsilon > 0$, $\exists\ \delta > 0$ for which if $\rho(x, x^{'}) < \delta$, then $\sigma(T(x), T(x^{'})) < \epsilon$, that is,

$$T(B(x, \delta)) \subseteq B(T(x), \epsilon), \tag{D.2}$$

where $B(x, \delta)$ and $B(T(x), \epsilon)$ are the **open balls** centered at x and $T(x)$ of radius r and ϵ, respectively.

Definition 2: A mapping $T : (X, \rho) \to (Y, \sigma)$ is said to be uniformly continuous, provided for every $\epsilon > 0$, $\exists\ \delta > 0$ such that for $u, v \in X$:

$$\text{if } \rho(u, v) < \delta, \text{ then } \sigma(T(u), T(v)) < \epsilon. \tag{D.3}$$

Uniformly continuous mapping is continuous; however, the converse is not true.

Definition 3: A mapping $T : (X, \rho) \to (Y, \sigma)$ is said to be Lipschitz provided there is a $\lambda \geq 0$ such that $\forall u, v \in X$,

$$\sigma(T(u), T(v)) \leq \lambda \rho(u, v), \tag{D.4}$$

where λ is called the Lipschitz constant. This is a very general definition of the Lipschitz continuity. A Lipschitz mapping is uniformly continuous, since it satisfies the criterion for uniform continuity that for any $\epsilon > 0$, we always find δ

such that $\delta = \frac{\epsilon}{\lambda}$. This definition also holds for a mapping T from a metric space (X, ρ) into itself, i.e., $T : (X, \rho) \rightarrow (X, \rho)$.

For our analysis of ConvRBM in Chapter 3, Section 3.5.4, we have used the following Lipschitz continuity condition defined on the L^2 norm and in the metric space of m-dimensional real numbers. Let the mapping (or the transformation) $T : \mathbb{R}^m \rightarrow \mathbb{R}^m$. Given an open set $B \subseteq \mathbb{R}^m$, we say that T is Lipschitz continuous on the open subset B if $\exists$ a constant $\lambda > 0$ such that

$$\|T(x) - T(y)\| \leq \lambda \|x - y\|, \quad \forall x, y \in B, \tag{D.5}$$

where λ is the Lipschitz constant of T on B, which depends on the choice of a norm. The mapping is locally Lipschitz continuous if λ depends on the input signal and globally Lipschitz continuous on all of the space, $\mathbb{R}^m$. **Important Remarks**:

Definition 4: A point $x \in X$ is called a *fixed point* of the mapping $T : X \rightarrow X$ provided $T(x) = x$.

A fixed point of a real-valued function of a real variable corresponds to a point in the place at which the graph of the function intersects the diagonal line $y = x$.

1. If λ=1, the mapping T is called a short map or firmly non-expansive mapping or *weak contraction*. Such functions do not increase or decrease distance between metric spaces; for example, the rectifier nonlinear function.

2. If $0 \leq \lambda < 1$, the mapping is called a *contraction* [149], which has exactly one fixed point. The contraction mapping implies Lipschitz continuous and hence, uniformly continuous.

3. The usual definition of continuity in the calculus is purely qualitative since it requires to define ϵ and δ, while Lipschitz continuity is quantitative in terms of the Lipschitz constant.

4. Lipschitz continuity is a special case of Hölder continuity, defined as [149]:

Definition 5 The mapping, $T : \mathbb{R}^m \rightarrow \mathbb{R}^m$, is Hölder continuous, if $\exists$ positive constant λ and α, such that

$$\|T(x) - T(y)\| \leq \lambda \|x - y\|^{\alpha}, \tag{D.6}$$

where the constant $\alpha \in \mathbb{R}$ is called the Hölder exponent [149]. If $\alpha = 1$, T is said to be Lipschitz continuous.

Appendix E. Agri-ASR System Building in Kaldi

E.1 Data Preparation

- First get the wave and label files for both the training and testing set.
- Prepare wav.scp, spk2utt, utt2spk and text files for both the train and test set as follows:

```
ls *.wav | cut -d "." -f 1 >../cname.txt (instead of ls *.wav> ../cname.txt
because we dont need .wav extension)
ls -d $PWD/*.wav > ../cpath.txt
cd ..
paste cname.txt cpath.txt > wav.scp
paste cname.txt cname.txt > spk2utt
paste cname.txt cname.txt > utt2spk
ls *.lab | cut -d "." -f 1 > ../labname.txt
cat *.lab > ../labcont.txt
cd..
paste labname.txt labcont.txt >text
```

- In kaldi/egs/ make a ASR_Guj directory In ASR_Guj directory make versions of your system, e.g., s1, s2... etc. Here, we make s1 directory inside ASR_Guj directory.
- Create a data directory in ASR_Guj. Make train and test folders inside data directory.
- Place the training and testing files wav.scp, spk2utt, utt2spk and text in data/train and data/test directory, respectively.

E.2 Language Model Preparation

- Create a folder named dict in the /s1/data/local. Put following files in this folder:lexicon.txt, nonsilence_phones.txt, optional_silence.txt, silence_phones.txt

- From the text of test set, find unique words and store them in words.txt

```
awk ’print $2’ < text > testwords.txt
sort -u testwords.txt > words.txt
```

- Use fst.sh and generate_bigram.py files provided by the IIT-Madras. The content of the fst.sh is as follows:

```
# ./fst.sh contents
#!/bin/bash
. ./cmd.sh
. ./path.sh
main_dir=/home/daiict/kaldi/egs/ASR_Guj/s1/
data_dir=$main_dir/data
dict_dir=$main_dir/data/local/dict
tmp_dir=$main_dir/data/tmp
lang_dir=$main_dir/data/lang
mkdir -p $tmp_dir
utils/prepare_lang.sh $dict_dir ’!SIL’ $data_dir/local/actual_overal
$main_dir/data/lang | | exit 1;

python local/generate_bigram.py $tmp_dir/words.txt
> $tmp_dir/wp_gram.txt

local/make_rm_lm.pl $tmp_dir/wp_gram.txt > $tmp_dir/G.txt

fstcompile --isymbols=$lang_dir/words.txt
--osymbols=$lang_dir/words.txt --keep_isymbols=false
--keep_osymbols=false $tmp_dir/G.txt > $lang_dir/G.fst
utils/validate_lang.pl $lang_dir
```

- generate_bigram.py program is used to generate the wp_gram from the district or commodity list. The content of this file is as follows:

```
#!/usr/bin/env python
import sys
from collections import defaultdict
word_list = [xx.strip().split() for xx in open(sys.argv[1])]
word_list = [ ["SENTENCE-END"] + xx + ["SENTENCE-END"] for xx in
word_list ]
suc_list = defaultdict(set)
for line in word_list:
for w1, w2 in zip(line[:-1], line[1:]):
suc_list[w1].add(w2)
list_of_keys = suc_list.keys()
list_of_keys.sort()
for ww in list_of_keys:
print ">" + ww
for ss in suc_list[ww]:
print " " + ss
```

- Change the paths in the fst.sh file and execute it; make sure to clear all the errors in this step. If it runs successfully, then you have no errors of mismatch in label files and lexicon. Do check G.fst file for binary file and not of very small size (in few bytes).

E.3 Feature Extraction

In this section, we will extract the MFCC feature set that will be used to build the GMM-HMM systems and the Mel filterbank (FBANK) feature set that will be used to build the hybrid DNN-HMM systems. Here, “nj 10" indicates the number of jobs to extract the features in parallel.

- The MFCC feature set is obtained as follows:

```
mfccdir=mfcc
for x in test train; do
   steps/make_mfcc.sh --cmd "$train_cmd" --nj 10 $datadir/$x
   exp/makemfcc/$x $mfccdir || exit 1;
   steps/compute_cmvn_stats.sh $datadir/$x exp/makemfcc/$x $mfc-
cdir || exit 1; done
```

- The FBANK feature set is obtained as follows:

```
fbankdir=fbank
for x in test train; do
    steps/make_fbank.sh --cmd "$train_cmd" --nj 10 $datadir/$x
    exp/makefbank/$x $fbankdir || exit 1;
    steps/compute_cmvn_stats.sh $datadir/$x exp/makefbank/$x
$fbankdir || exit 1; done
```

E.4 Acoustic Modeling GMM-HMM

In this Section, we will show how to build GMM-HMM system in KALDI.

- Monophone GMM-HMM system can be build by the following commands:

```
expdir=mono_mfcc
steps/train_mono.sh --nj "$train_nj" --cmd "$train_cmd" $datadir/train
data/lang exp/$expdir || exit 1;
utils/mkgraph.sh --mono data/lang exp/$expdir exp/$expdir/graph ||
exit 1;
steps/decode.sh --nj "$decode_nj" --cmd "$decode_cmd"
exp/$expdir/graph $datadir/test exp/$expdir/decode || exit 1;
local/score.sh --cmd run.pl $datadir/test exp/$expdir/graph
exp/$expdir/decode || exit 1;
```

- The triphone GMM-HMM system will be built from the alignments generated from the monophone system. Here, we have option to vary the number of senones and Gaussians in the triphone trees. The triphone GMM-HMM system can be built by the following commands:

```
expdir=mono_mfcc
tridir=tri_mfcc
  steps/align_si.sh --boost-silence 1.25 --nj "$train_nj" --cmd "$train_cmd"
   $datadir/train data/lang exp/$expdir exp/$expdir_ali | | exit 1;
for sen in 1800 2000 2200 2500; do
for gauss in 12 14 16; do
  gauss=$(($sen * $gauss))
  steps/train_deltas.sh --cmd "$train_cmd" $sen $gauss $datadir/train
  data/lang exp/$expdir_ali exp/$tridir_$sen_$gauss | | exit 1;
  utils/mkgraph.sh data/lang exp/$tridir_$sen_$gauss
  exp/$tridir_$sen_$gauss/graph | | exit 1;
  steps/decode.sh --nj "$decode_nj" --cmd "$decode_cmd"
  exp/$tridir_$sen_$gauss/graph $datadir/test
exp/$tridir_$sen_$gauss/decode | | exit 1;
```

- The triphone system with the lowest % WER is selected for the LDA+MLLT system building. For example, here a system with 2000 senones and 12 Gaussians is selected.

```
steps/align_si.sh --nj "$train_nj" --cmd "$train_cmd" data/train data/lang
exp/tri_mfcc_2000_24000 exp/tri_mfcc_2000_24000_ali | | exit 1;
for sen in 2000 2500 3000; do
for gauss in 12 16; do
  gauss=$(($sen * $gauss))
  steps/train_lda_mllt.sh --cmd "$train_cmd" --splice-opts
  "--left-context=3 --right-context=3" $sen $gauss data/train data/lang
  exp/tri_mfcc_2000_24000_ali   exp/$tridir2_$sen_$gauss | | exit 1;
  utils/mkgraph.sh data/lang exp/$tridir2_$sen_$gauss
  exp/$tridir2_$sen_$gauss/graph2 | | exit 1;
  steps/decode.sh --nj "$decode_nj" --cmd "$decode_cmd"
  exp/$tridir2_$sen_$gauss/graph2 data/test
exp/$tridir2_$sen_$gauss/decode3 | | exit 1;
done
done
```

E.5 Acoustic Modeling using DNN-HMM

- The LDA-MLLT system with the lowest % WER is selected for the hybrid DNN-HMM experiments. First generate the alignments from the LDA-MLLT system as follows:

```
expdir=exp/tri2_mfcc_2500_40000
steps/align_si.sh --nj 8 --cmd "$train_cmd"  data/train data/lang
exp/$expdir exp/$expdir_ali
```

- To train the DNN-HMM system, the Mel filterbank features are extracted as follows:

```
fbankdir=fbank
for x in test train; do
  steps/make_fbank.sh --cmd "$train_cmd" --nj 10 $datadir/$x
  exp/makefbank/$x $fbankdir || exit 1;
  steps/compute_cmvn_stats.sh $datadir/$x exp/makefbank/$x
  $fbankdir || exit 1;
done
```

- The hybrid DNN-HMM system using nnet3 setup in the KALDI toolkit. We show a demo of building TDNN system with different numbers of hidden units as follows:

```
for x in 500 600 700 800 900; do
  datadir=nnet2_data_$features
  nndir=tri2_fbank40_TDNN_$x
  steps/nnet3/tdnn/train.sh --relu-dim $x $datadir/train
  data/lang $expdir_ali exp/tdnn/$nndir_nnet3 || exit 1;

  steps/nnet3/decode.sh $expdir/graph $datadir/test
  exp/tdnn/$nndir_nnet3/decode

  local/score.sh --cmd run.pl $datadir/test $expdir/graph
  exp/tdnn/$nndir_nnet3/decode
done
```

Appendix F. Miscellaneous

F.1 Maximum Likelihood Estimation (MLE)

Mathematical modeling represents an underlying process (such as hearing as a physiological auditory processing) by means of certain parameters of the model that completely characterize the model. There are two general methods for estimation of the parameters, namely, least squares estimation (LSE) and maximum likelihood estimation (MLE). The LSE is popular in statistical model fitting techniques, such as linear regression and root mean squared error (RMSE), which unlike MLE does not require distributional assumptions [247]. MLE is considered as a general approach for parameter optimization, which has the following nice properties [14], [247]:

- **sufficiency**: complete information about the parameters in the MLE,
- **efficiency**: lowest possible variance of the parameter estimates achieved asymptotically,
- **consistency**: for data of sufficiently large samples, the true parameter value that generated the data recovered asymptotically,
- **parameterization invariance**: same MLE solution obtained independent of the parametrization used.

Let $\mathbf{X} = \{\mathbf{x}_d | d \in \{1, 2, ..., D\}\}$ be independent and identically distributed (i.i.d.) samples with probability density function (pdf) $p(\mathbf{x}_d; \theta)$, where θ are the model parameters that characterize $p(\mathbf{x}_d; \theta)$. The likelihood function is defined as the joint probability density $p(\mathbf{x}_1, \mathbf{x}_1, ..., \mathbf{x}_D; \theta)$ treated as a function of the parameters θ:

$$L(\theta | \mathbf{x}_1, \mathbf{x}_1, ..., \mathbf{x}_D) = p(\mathbf{x}_1, \mathbf{x}_1, ..., \mathbf{x}_D; \theta) = \prod_{d=1}^{D} p(\mathbf{x}_d; \theta). \tag{F.1}$$

To simply the notation, it can be written in a vector form as:

$$L(\mathbf{X}|\theta) = p(\mathbf{X};\theta). \tag{F.2}$$

There is an important difference between pdf $p(\mathbf{X};\theta)$ and likelihood function. Both functions are defined on different scales and hence, they are not directly comparable as shown in [247]. The $p(\mathbf{X};\theta)$ is a function of data given a particular set of parameters defined on the *data scale* [247]. On the other hand, the likelihood function $L(\mathbf{X}|\theta)$ is a function of the parameters given a particular set of observed data, defined on the *parameter scale* [247]. We are interested in estimating the optimal value of θ such that it maximizes the likelihood of the observed data. The principle of MLE, originally developed by R.A. Fisher (1920), states that the desired probability distribution is the one that makes the observed data “most likely", which means that one must obtain the value of the parameters that maximizes the likelihood function $L(\mathbf{X}|\theta)$ [247]. The ML estimate denoted as θ^* is given as:

$$\theta^* = \arg\max_{\theta} L(\mathbf{X}|\theta). \tag{F.3}$$

It is often very difficult to solve eq. (F.3). Hence, for mathematical convenience, we maximize the log-likelihood $\log L(\mathbf{X}|\theta)$ given as:

$$\log L(\mathbf{X}|\theta) = \log\left(\prod_{d=1}^{D} p(\mathbf{x}_d;\theta)\right) = \sum_{d=1}^{D} \log p(\mathbf{x}_d;\theta). \tag{F.4}$$

Since logarithm is a monotone function, the optimal value θ^* remains the same in the optimization. The monotone function is defined as follows:

Definition: Let $T : \mathbb{R} \to \mathbb{R}$ be a function. T is a monotonically increasing (decreasing) on $\mathbb{R}$, if $\forall x, y \in \mathbb{R}$, $x \leq y$ one has $T(x) \leq T(y)$ (respectively, $T(x) \geq T(y)$). The properties of monotone functions are given by following theorems [149].

Theorem: Let T be a monotone function on the open interval (a, b). Then, T is continuous except possibly at a countable number of points in (a, b).

Lebesgue’s theorem: If the function T is monotone on the open interval (a, b), then it is differential almost everywhere (that is, except for a set of points that has Lebesgue measure zero) on (a, b).

Since ML optimization involves finding derivatives of log-likelihood function, the optimal θ can be found by differentiating the log-likelihood, and solving the first-order conditions (known as the likelihood equation) as follows:

$$\frac{\partial}{\partial\theta}\log L(\mathbf{X}|\theta) = 0, \tag{F.5}$$

$$\frac{1}{L(\mathbf{X}|\theta)}\frac{\partial}{\partial\theta}L(\mathbf{X}|\theta) = 0, \tag{F.6}$$

$$\frac{\partial}{\partial\theta}L(\mathbf{X}|\theta) = 0. \tag{F.7}$$

A necessary condition for the existence of an MLE estimate is represented by the likelihood equation (F.7). Since the first-order derivatives cannot reveal that the function is maximum or minimum, we need additional conditions for the ML estimate. This can be done by checking second-order derivatives as follows:

$$\frac{\partial^2}{\partial\theta^2}\log L(\mathbf{X}|\theta) < 0. \tag{F.8}$$

In many cases, it is difficult to find the value of parameters analytically using equations (F.7) and (F.8). In that case, the usual way to find the parameters is to use gradient-based techniques, such as gradient-ascent [14], [25]. This corresponds to iteratively updating the parameters $\theta^{(t)}$ to $\theta^{(t+1)}$ based on the gradient of $L(\mathbf{X}|\theta)$. The update rule for the gradient-ascent is given as follows [25], [14]:

$$\theta^{(t+1)} = \theta^{(t)} + \nabla\theta^{(t)}, \tag{F.9}$$

where $\nabla\theta^{(t)} = \epsilon\frac{\partial}{\partial\theta^{(t)}}\log L(\mathbf{X}|\theta^{(t)})$ is a gradient of log-likelihood, and ϵ is called the learning rate parameter. The learning rule can also be extended by including the momentum term as follows:

$$\theta^{(t+1)} = \theta^{(t)} + \nabla\theta^{(t)} + \eta\theta^{(t-1)}, \tag{F.10}$$

where $\eta \in [0,1)$ is called as the momentum parameter that helps against oscillatory behavior in the iterative updates, and can speed up the learning process [52], [70].

F.2 Relationship Between KL-Divergence and ML

The goal of probabilistic graphical models (PGM) is to learn the model distribution $p(\mathbf{x};\theta)$ that approximates the true distribution of the data points $f(\mathbf{x})$ [14]. The relative entropy or Kullback-Leibler (KL) divergence is used to find the difference between two probability distributions, and it is defined as follows [54], [248]:

$$\mathrm{KL}(f(\mathbf{x})||p(\mathbf{x};\theta)) = \int_{-\infty}^{\infty} f(\mathbf{x}) \log\left(\frac{f(\mathbf{x})}{p(\mathbf{x};\theta)}\right) d\mathbf{x}, \tag{F.11}$$

$$= \int_{-\infty}^{\infty} f(\mathbf{x}) \log f(\mathbf{x}) d\mathbf{x} - \int_{-\infty}^{\infty} f(\mathbf{x}) \log p(\mathbf{x};\theta) d\mathbf{x}, \tag{F.12}$$

$$= \mathbb{E}_{f(\mathbf{x})}\left[\log f(\mathbf{x})\right] - \mathbb{E}_{f(\mathbf{x})}\left[\log p(\mathbf{x};\theta)\right], \tag{F.13}$$

where $\mathbb{E}[\cdot]$ is the expectation operator over the distribution $f(\mathbf{x})$. The first term $\mathbb{E}_{f(\mathbf{x})}\left[\log f(\mathbf{x})\right]$ is constant (negative of the entropy), and will not take part in optimization of model parameters θ. Hence, KL-divergence is minimum, when the second term $\mathbb{E}_{f(\mathbf{x})}\left[\log p(\mathbf{x};\theta)\right]$ is maximum (due to negative sign). The second term is just the expected log-likelihood, $\log p(\mathbf{x};\theta)$ as given in eq. (3.11) in Chapter 3. The $\mathbb{E}_{f(\mathbf{x})}\left[\log p(\mathbf{x};\theta)\right]$ can be calculated as sample mean:

$$\mathbb{E}_{f(\mathbf{x})}\left[\log p(\mathbf{x};\theta)\right] \approx \frac{1}{D}\sum_{d=1}^{D} \log p(\mathbf{x};\theta). \tag{F.14}$$

Hence, eq. (F.13) can be written as:

$$\arg\min_{\theta} \mathrm{KL}(f(\mathbf{x})||p(\mathbf{x};\theta)) \Leftrightarrow \arg\min_{\theta}\left[-\frac{1}{D}\sum_{d=1}^{D} \log p(\mathbf{x};\theta)\right], \tag{F.15}$$

$$\Leftrightarrow \arg\max_{\theta}\left[\frac{1}{D}\sum_{d=1}^{D} \log p(\mathbf{x};\theta)\right], \tag{F.16}$$

$$\Leftrightarrow \arg\max_{\theta}\left[\sum_{d=1}^{D} \log p(\mathbf{x};\theta)\right], \tag{F.17}$$

$$\Leftrightarrow \arg\max_{\theta} L(\mathbf{X}|\theta). \tag{F.18}$$

This proves that minimizing the KL-divergence (which is an information-theoretic measure) is equivalent to maximizing the log-likelihood function.

F.3 Weight Decay Regularization in ConvRBM

One of the key problems in representation learning is how to make a model that will perform well not just on the training data, but also on new inputs in the test data. One also has to be careful such that the model will not overfit either due to more parameters or less training data. Many strategies used in machine learning are explicitly designed to reduce the overfitting and test error. These strategies are known collectively as the regularization techniques [20]. The norm penalties are the most common and the simplest regularization techniques. They are based on limiting the capacity of models, such as linear regression, logistic regression, and neural networks, by adding a parameter norm penalty $\Omega(\theta)$ to the objective function. In the case of ConvRBM, norm penalty $\Omega(\mathbf{W})$ is used for the weights $\mathbf{W}$ and added to the follows

$$\tilde{\ell}(\mathbf{x};\theta) = \ell(\mathbf{x};\theta) + \lambda_d \Omega(\mathbf{W}), \tag{F.19}$$

where $\lambda_d \in [0, \infty)$ is a regularization parameter that weights the relative contribution of the norm penalty term relative to the standard objective function. Larger values of λ_d correspond to more regularization effect [20]. Generally for the neural networks, we penalize only the weights of the model at each layer and no regularization for the biases. The biases typically require less data to learn than the weights to fit accurately. Each weight specifies how two variables (in the hidden and visible layer) interact. Fitting the weight well requires observing both variables in a variety of conditions as shown in eq. (3.25) in Section 3.3, Chapter 3. Each bias controls only a single variable (as derived in eq. (3.26) and eq. (3.27) in Section 3.3, Chapter 3). This means that we do not induce too much variance by leaving the biases unregularized. Also, regularizing the bias parameters can introduce a significant amount of underfitting [20]. Hence, we regularize the weights of the ConvRBM only.

The simplest norm penalty is the L^2-norm regularization. It is also known as the weight decay in the neural network literature [20]. In other academic communities it is also known as Tikhonov regularization or ridge regression [20]. The log-likelihood-function along with the weight decay is written as:

$$\tilde{\ell}(\mathbf{x};\theta) = \ell(\mathbf{x};\theta) + \frac{\lambda_d}{2} \|\mathbf{W}\|_2 , \tag{F.20}$$

where the factor $\frac{1}{2}$ is only for the mathematical convenience. The gradient of weights is then given as:

$$\frac{\partial}{\partial \mathbf{W}} \tilde{\ell}(\mathbf{x};\theta) = \frac{\partial}{\partial \mathbf{W}} \ell(\mathbf{x};\theta) + \lambda_d \mathbf{W} \tag{F.21}$$

The weight decay regularization prevents the weights from growing too large and forces them to be near to zero. Hence, this introduces sparsity in the model. One can also use L^1 regularization to prevent overfitting and introduce sparsity in the model. Since ConvRBM uses ReLU activations, it natural introduces sparsity in the hidden units. Our experiments with the L^1-regularization show that ConvRBM did not work well with the ReLU activations since training such ConvRBM makes more weights and the hidden units to zero. Hence, we prefer weight decay, i.e., L^2-regularization in the ConvRBM training along with the ReLU activations.

F.4 Adam Optimization Algorithm

Algorithm 4 The Adam optimization algorithm. After [163].

```
Input: Stochastic objective function $f(\theta)$ with parameters $\theta$, exponential decay
    rates for the moment estimates $\beta_1, \beta_2 \in (0, 1]$, step size $\alpha$
    Initialize first moment vector: $m_0 \leftarrow 0$
    Initialize second moment vector: $v_0 \leftarrow 0$
    Initialize training iteration: $t \leftarrow 0$
Output: Optimized parameters $\theta_t$
while $\theta_t$ not converged do
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla f_t(\theta_{t-1})$
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) g_t^2$ (Update biased second raw moment estimate)
    $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\hat{v}_t + \epsilon)$ (Update the parameters of the model)
end while
```

Bibliography

[1] N. Morgan and H. Bourlard, "An introduction to hybrid HMM/connectionist continuous speech recognition," *IEEE Signal Processing Magazine*, vol. 12, no. 3, pp. 25–42, May 1995.

[2] H. B. Sailor and H. A. Patil, "Filterbank learning using convolutional restricted Boltzmann machine for speech recognition," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, China, 20-25 March 2016, pp. 5895–5899.

[3] H. B. Sailor and H. A. Patil, "Novel unsupervised auditory filterbank learning using convolutional RBM for speech recognition," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 24, no. 12, pp. 2341–2353, Dec. 2016.

[4] H. B. Sailor and H. A. Patil, "Auditory feature representation using convolutional restricted Boltzmann machine and Teager energy operator for speech recognition," *Journal of Acoustical Society of America Express Letters (JASA-EL)*, vol. 141, no. 6, pp. EL500–EL506, June. 2017.

[5] H. B. Sailor, D. M. Agrawal, and H. A. Patil, "Unsupervised filterbank learning using convolutional restricted Boltzmann machine for environmental sound classification," in *INTERSPEECH*, Stockholm, Sweden, 2017, pp. 3107–3111.

[6] H. B. Sailor, M. R. Kamble, and H. A. Patil, "Unsupervised representation learning using convolutional restricted Boltzmann machine for spoof speech detection," in *INTERSPEECH*, Stockholm, Sweden, 2017, pp. 2601–2605.

[7] H. B. Sailor, M. R. Kamble, and H. A. Patil, "Auditory filterbank learning for temporal modulation features in replay spoof speech detection," *to appear in INTERSPEECH, Hyderabad*, Sept. 2018.

[8] N. Buddha and H. A. Patil, "Corpora for analysis of infant cry," in *Int. Conf. on Speech Databases and Assessments, Oriental COCOSDA, Hanoi, Vietnam*, Dec. 2007, pp. 43 – 48.

[9] H. B. Sailor and H. A. Patil, "Auditory filterbank learning using ConvRBM for infant cry classification," *to appear in INTERSPEECH, Hyderabad*, Sept. 2018.

[10] H. B. Sailor and H. A. Patil, "Unsupervised learning of temporal receptive fields using convolutional RBM for ASR task," in *European Signal Processing Conference (EUSIPCO), Budapest, Hungary*, 29 Aug. - 2 Sept. 2016, pp. 873–877.

[11] H. B. Sailor and H. A. Patil, "Unsupervised deep auditory model using stack of convolutional RBMs for speech recognition," in *INTERSPEECH*, San Francisco, California, USA, 8–12 September 2016, pp. 3379–3383.

[12] M. S. Lewicki, "Efficient coding of natural sounds," *Nature Neuroscience*, vol. 5, no. 4, pp. 356–363, 2002.

[13] L. H. Carney and T. Yin, "Temporal coding of resonances by low-frequency auditory nerve fibers: single-fiber responses and a population model," *Journal of Neurophysiology*, vol. 60, no. 5, pp. 1653–1677, 1988.

[14] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, First Edition, 2007.

[15] R. Rojas, *Neural Networks - A Systematic Introduction*. New York, NY, USA: Springer-Verlag, First Edition, 1996.

[16] L. Chittka and A. Brockmann, "Perception space: The final frontier," *Public Library of Science (PLOS) Biology*, vol. 3, no. 4, pp. 1–5, 2005.

[17] D. Purves, G. J. Augustine, D. Fitzpatrick, L. C. Katz, A.-S. LaMantia, J. O. McNamara, and S. M. Williams, *Neuroscience*. Sunderland (MA), Sinauer Associates, Third Ddition, 2001.

[18] D. P. W. Ellis, "Gammatone-like spectrograms," URL: http://www.ee.columbia.edu/~dpwe/resources/matlab/gammatonegram/, {Last Accessed on 20 December, 2017}.

[19] F. E. Theunissen and J. E. Elie, "Neural processing of natural sounds," *Nature Reviews Neuroscience*, vol. 15, pp. 355–366, 2014.

[20] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, First Edition, 2016.

[21] A. Graves, N. Jaitly, and A. R. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Olomouc, Czech Republic, 2013, pp. 273–278.

[22] M. Gales and S. Young, *The Application of Hidden Markov Models in Speech Recognition*. Hanover, MA, USA: Foundations and Trends in Signal Processing, Now Publishers Inc., First Edition, 2007.

[23] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[24] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[25] S. S. Haykin, *Neural Networks and Learning Machines*. Pearson Education, Third Edition, 2009.

[26] B. U. EarLab, "The auditory revcor database," URL: http://earlab.bu.edu/databases/collections/Default.aspx, {Last Accessed: 20 December 2017}.

[27] J. Basu, S. Khan, R. Roy, and M. S. Bepari, "Commodity price retrieval system in Bangla: An IVR-based application," in *Proceedings of the Asia-Pacific Conference on Computer Human Interaction*, Bangalore, India, 2013, pp. 406–415.

[28] AGMARKNET, "Ministry of agriculture and farmers welfare, government of India," URL: http://agmarknet.dac.gov.in/, {Last Accessed on 22 December 2017}.

[29] IMD, "India meteorological department (IMD), ministry of earth sciences, government of india," URL: http://www.imd.gov.in/pages/main.php, {Last Accessed: 22 December 2017}.

[30] H. B. Sailor and H. A. Patil, "Representation learning for speech recognition system in agricultural commodity for Gujarati (poster presentation)," in *Global Conference on Cyberspace (GCCS), Organized by MeitY, Govt. of India under National e-Governance Division (NeGD), New Delhi, India*, 2017.

[31] K. Samudravijaya and H. A. Murthy, "Indian language speech sound label set (ILSL12), version v2.1.3," *Indian Language TTS Consortium and ASR Consortium*, pp. 1–14, 2013.

[32] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in 25th *IEEE Int. Workshop on Machine Learning for Signal Processing (MLSP)*, Boston, MA, USA, 2015, pp. 1–6.

[33] D. M. Agrawal, H. B. Sailor, M. H. Soni, and H. A. Patil, "Novel TEO-based gammatone features for environmental sound classification," in *European Signal Processing Conf. (EUSIPCO)*, Kos island, Greece, August 28 – 2 September 2017.

[34] H. A. Patil, "Cry baby: Using spectrographic analysis to assess neonatal health status from an infant's cry," in *Advances in Speech Recognition Mobile Environments, Call Centers and Clinics*. A. Neustein, (Ed.), Springer, 2010, pp. 323–348.

[35] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, "The DET curve in assessment of detection task performance," in *EUROSPEECH*, Rhodes, Greece, 1997, pp. 1895–1898.

[36] A. M. Zoubir and D. R. Iskander, *Bootstrap Techniques for Signal Processing*. Cambridge University Press, First Edition, 2004.

[37] M. Benzeghiba, R. D. Mori *et al.*, "Automatic speech recognition and speech variability: A review," *Speech Communication*, vol. 49, no. 10, pp. 763 – 786, 2007.

[38] R. Stern and N. Morgan, "Hearing is believing: Biologically inspired methods for robust automatic speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 34–43, Nov 2012.

[39] R. M. Stern and N. Morgan, "Features based on auditory physiology and perception," in *Techniques for Noise Robustness in Automatic Speech Recognition*. T. Virtanen, B. Raj, and R. Singh, (Eds.) John Wiley and Sons, Ltd, New York, NY, USA, 2012, pp. 193–227.

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

[41] Z. Zhang, N. Cummins, and B. Schuller, "Advanced data exploitation in speech analysis: An overview," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 107–129, July 2017.

[42] G. Hinton, "Where do features come from?" *Cognitive Science*, vol. 38, no. 6, pp. 1078–1101, 2014.

[43] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[44] P. K. Kuhl, "Early language acquisition: cracking the speech code," *Nature Reviews Neuroscience*, vol. 5, no. 11, pp. 831–843, Nov. 2004.

[45] H. B. Sailor, M. R. Kamble, and H. A. Patil, "Novel unsupervised filterbank learning using convolutional restricted Boltzmann machine for replay spoof speech detection," *submitted in Pattern Recognition Letters (PRL)*, Sept. 2017.

[46] H. B. Sailor and H. A. Patil, "Unsupervised auditory filterbank learning for infant cry classification," in *Voice Technologies for Reconstruction and Enhancement*. Hemant A. Patil and Neustein, Amy,(Eds.), De Gruyter Series in Speech Technology and Text Analytics in Medicine and Healthcare, 2018, pp. 1–18.

[47] R. F. Lyon, "Machine hearing: An emerging field," *[Exploratory DSP], IEEE Signal Processing Magazine*, vol. 27, no. 5, pp. 131–139, Sept. 2010.

[48] E. Smith and M. S. Lewicki, "Efficient coding of time-relative structure using spikes," *Neural Comput.*, vol. 17, no. 1, pp. 19–45, Jan. 2005.

[49] T. Chi, P. Ru, and S. A. Shamma, "Multiresolution spectrotemporal analysis of complex sounds," *The Journal of the Acoustical Society of America (JASA)*, vol. 118, no. 2, pp. 887–906, 2005.

[50] H. Lee, P. T. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *23^{rd} Annual Conference on Neural Information Processing Systems (NIPS), Canada, 7-10 December*, 2009, pp. 1096–1104.

[51] K. P. Murphy, *Machine Learning : A Probabilistic Perspective*. The MIT Press, First Edition, 2013.

[52] A. Fischer and C. Igel, "An introduction to restricted Boltzmann machines," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Alvarez L., Mejail M., Gomez L., Jacobo J. (Eds.), Springer, 2012, pp. 14–36.

[53] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," in *Predicting Structured Data*. G. Bakir and T. Hofman and B. Scholkopf and A. Smola and B. Taskar, (Eds.), The MIT Press, 2006, pp. 1–59.

[54] D. J. C. MacKay, *Information Theory, Inference & Learning Algorithms*. New York, NY, USA: Cambridge University Press, First Edition, 2002.

[55] G. E. Hinton and T. J. Sejnowski, "Learning and relearning in Boltzmann machines," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*. Cambridge, MA, USA: Rumelhart, David E. and McClelland, James L. and PDP Research Group, (Eds.), CORPORATEMIT Press, 1986, pp. 282–317.

[56] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*, D. E. Rumelhart, J. L. McClelland, and C. PDP Research Group, Eds. Cambridge, MA, USA: MIT Press, 1986, pp. 194–281.

[57] M. Welling, M. Rosen-Zvi, and G. E. Hinton, "Exponential family harmoniums with an application to information retrieval," in *Advances in neural information processing systems (NIPS), Vancouver, Canada*, 2005, pp. 1481–1488.

[58] G. Hinton, L. Deng *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[59] H. Lee, R. B. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *26^{th} Annual International Conference on Machine Learning, (ICML), Canada, June 14-18*, 2009, pp. 609–616.

[60] G. E. Hinton, "A practical guide to training restricted Boltzmann machines," in *Neural Networks: Tricks of the Trade*. Montavon G., Orr and G.B., Muller KR. , (Eds.), Springer, 2012, pp. 599–619.

[61] J. W. Schnupp, I. Nelken, and A. J. King, *Auditory Neuroscience: Making Sense of Sound*. The MIT Press, First Edition, 2012.

[62] X. Yang, K. Wang, and S. Shamma, "Auditory representations of acoustic signals," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 824–839, March 1992.

[63] D. A. Depireux, J. Z. Simon, D. J. Klein, and S. A. Shamma, "Spectro–temporal response field characterization with dynamic ripples in ferret primary auditory cortex," *Journal of Neurophysiology*, vol. 85, no. 3, pp. 1220–1234, 2001.

[64] A. Qiu, C. E. Schreiner, and M. A. Escabí, "Gabor analysis of auditory midbrain receptive fields: Spectro-temporal and binaural composition," *Journal of Neurophysiology*, vol. 90, no. 1, pp. 456–476, 2003.

[65] K. C. Puvvada and J. Z. Simon, "Cortical representations of speech in a multitalker auditory scene," *Journal of Neuroscience*, vol. 37, no. 38, pp. 9189–9196, 2017.

[66] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[67] L. Deng and D. Yu, *Deep Learning: Methods and Applications*. Foundations and Trends in Signal Processing, First Edition, 2014.

[68] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *International Conference on Artificial Intelligence and Statistics (AISTATS), Ft. Lauderdale, FL, USA*, 2011, pp. 315–323.

[69] J. Li, T. Zhang, W. Luo, J. Yang, X. T. Yuan, and J. Zhang, "Sparseness analysis in the pretraining of deep neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 6, pp. 1425–1438, June 2017.

[70] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, First Edition, 1995.

[71] K. Fukushima, S. Miyake, and T. Ito, "Neocognitron: A neural network model for a mechanism of visual pattern recognition," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, no. 5, pp. 826–834, Sept. 1983.

[72] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[73] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond Euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, July 2017.

[74] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 3, pp. 328–339, Mar. 1989.

[75] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *INTERSPEECH, Dresden Germany*, 2015, pp. 2440–2444.

[76] V. Peddinti, Y. Wang, D. Povey, and S. Khudanpur, "Low latency acoustic modeling using temporal convolution and LSTMs," *IEEE Signal Processing Letters*, vol. PP, no. 99, pp. 1–5, 2017.

[77] H. Hermansky, J. Cohen, and R. Stern, "Perceptual properties of current speech recognition technology," *Proce. of the IEEE*, vol. 101, no. 9, pp. 1968–1985, Sept. 2013.

[78] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. on Acoust., Speech, and Signal Process.*, vol. 28, no. 4, pp. 357–366, 1980.

[79] S. S. Stevens, J. Volkmann, and E. B. Newman, "A scale for the measurement of the psychological magnitude pitch," *The Journal of the Acoustical Society of America*, vol. 8, no. 3, pp. 185–190, 1937.

[80] A. Oppenheim and R. Schafer, "Homomorphic analysis of speech," *IEEE Transactions on Audio and Electroacoustics*, vol. 16, no. 2, pp. 221–226, Jun 1968.

[81] A. R. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 14–22, Jan. 2012.

[82] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *The Journal of the Acousti. Soc. of America (JASA)*, vol. 87, no. 4, pp. 1738–1752, 1990.

[83] R. Schluter, I. Bezrukov, H. Wagner, and H. Ney, "Gammatone features and feature combination for large vocabulary speech recognition," in *Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Honolulu, Hawaii, USA, 2007, pp. 645–649.

[84] D. O'Shaughnessy, "Acoustic analysis for automatic speech recognition," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1038–1053, May 2013.

[85] F. Jelinek, *Statistical Methods for Speech Recognition*. Cambridge, MA, USA: MIT Press, First Edition, 1997.

[86] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., First Edition, 1993.

[87] X. Huang and L. Deng, "An overview of modern speech recognition," in *Handbook of Natural Language Processing, Second Edition*. Nitin Indurkhya and Fred J. Damerau, (Eds.), Chapman and Hall/CRC, 2010, pp. 339–366.

[88] J. A. Bilmes, "What HMMs can do," *IEICE Trans. Inf. Syst.*, vol. E89-D, no. 3, pp. 869–891, Mar. 2006.

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

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

[91] L. Deng and X. Li, "Machine learning paradigms for speech recognition: An overview," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 5, pp. 1060–1089, May 2013.

[92] H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Norwell, MA, USA: Kluwer Academic Publishers, First Edition, 1993.

[93] M. Zaki, H. B. Sailor, and H. A. Patil, "Analysis of hierarchical bottleneck framework for improved phoneme recognition," in *IEEE International Conference on Signal Processing and Communications (SPCOM), IISc, Bangalore*, June 2016, pp. 1–5.

[94] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio, "Why does unsupervised pre-training help deep learning?" *J. Mach. Learn. Res. (JMLR)*, vol. 11, pp. 625–660, Mar. 2010.

[95] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *International Conference on Machine Learning (ICML)*, New York, USA, 2006, pp. 369–376.

[96] R. F. Lyon, *Human and Machine Hearing: Extracting Meaning from Sound*. Cambridge University Press, First Edition, 2017.

[97] M. L. Jepsen, S. D. Ewert, and T. Dau, "A computational model of human auditory signal processing and perception," *The Journal of the Acoustical Society of America (JASA)*, vol. 124, no. 1, pp. 422–438, 2008.

[98] J. Anden and S. Mallat, "Deep scattering spectrum," *IEEE Transactions on Signal Processing*, vol. 62, no. 16, pp. 4114–4128, Aug. 2014.

[99] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning (ICML), Beijing, China*, 2014, pp. 1764–1772.

[100] D. Amodei, R. Anubhai, E. Battenberg *et al.*, "Deep speech 2: End-to-end speech recognition in English and Mandarin," in *arXiv preprint arXiv:1512.02595*, 2015.

[101] Y. Miao, M. Gowayyed, and F. Metze, "EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding," in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Scottsdale, AZ, USA*, Dec. 2015.

[102] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform CLDNNs," in *INTERSPEECH*, Dresden, Germany, 6–10 Sept. 2015, pp. 1–5.

[103] P. Golik, Z. Tüske, R. Schlüter, and H. Ney, "Convolutional neural networks for acoustic modeling of raw time signal in LVCSR," in *INTERSPEECH*, Dresden, Germany, 6-10 Sep. 2015, pp. 26–30.

[104] D. Palaz, M. Magimai-Doss, and R. Collobert, "Convolutional neural networks-based continuous speech recognition using raw speech signal," in *40^{th} International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, QLD*, 19–24 April 2015, pp. 4295–4299.

[105] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for LVCSR," in *INTERSPEECH*, Singapore, 14–18 Sep. 2014, pp. 890–894.

[106] Y. Tokozume and T. Harada, "Learning environmental sound with end-to-end convolutional neural network," in *IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, New Orleans, USA, 2017, pp. 2721–2725.

[107] Y. Aytar, C. Vondrick, and A. Torralba, "Soundnet: Learning sound representations from unlabeled video," in *Advances in Neural Information Processing Systems (NIPS)*, Barcelona, Spain, 2016, pp. 892–900.

[108] J. Lee and et. al, "Speech feature extraction using independent component analysis," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Istanbul, Turkey*, vol. 3, 2000, pp. 1631–1634.

[109] M. S. Lewicki and T. J. Sejnowski, "Learning overcomplete representations," *Neural Comput.*, vol. 12, no. 2, pp. 337–365, Feb. 2000.

[110] A. Bertrand, K. Demuynck, V. Stouten, and H. V. hamme, "Unsupervised learning of auditory filter banks using non-negative matrix factorization," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, (ICASSP) 2008, Las Vegas, Nevada, USA*, 2008, pp. 4713–4716.

[111] Y.-H. Chiu, B. Raj, and R. Stern, "Learning-based auditory encoding for robust speech recognition," in *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), Dallas, Texas, USA*, March 2010, pp. 4278–4281.

[112] S. Chatterjee and W. Kleijn, "Auditory model-based design and optimization of feature vectors for automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 6, pp. 1813–1825, Aug. 2011.

[113] N. Jaitly and G. Hinton, "Learning a better representation of speech soundwaves using restricted Boltzmann machines," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Prague, Czech Republic*, 22–27 May 2011, pp. 5884–5887.

[114] E. Smith and M. S. Lewicki, "Efficient coding of time-relative structure using spikes," *Neural Comput.*, vol. 17, no. 1, pp. 19–45, Jan. 2005.

[115] E. C. Smith and M. S. Lewicki, "Efficient auditory coding," *Nature*, vol. 439, no. 7079, pp. 978–982, 2006.

[116] S. van Vuuren and H. Hermansky, “Data-driven design of RASTA-like filters,” in *Eurospeech, Rhodes, Greece*, vol. 1, 1997, pp. 1607–1610.

[117] B. Mak, Y.-C. Tam, and R. Hsiao, “Discriminative training of auditory filters of different shapes for robust speech recognition,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing(ICASSP), Hong Kong, China*, vol. 2, April 2003, pp. 45–48.

[118] J.-W. Hung and L.-S. Lee, “Optimization of temporal filters for constructing robust features in speech recognition,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 3, pp. 808–832, May 2006.

[119] S. Chatterjee and W. Kleijn, “Auditory model-based design and optimization of feature vectors for automatic speech recognition,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 6, pp. 1813–1825, Aug 2011.

[120] N. Jaitly and G. E. Hinton, “Using an autoencoder with deformable templates to discover features for automated speech recognition,” in *INTERSPEECH 2013, Lyon, France, August 25-29*, 2013, pp. 1737–1740.

[121] S.-Y. Chang and N. Morgan, “Robust CNN-based speech recognition with Gabor filter kernels,” in *INTERSPEECH, Singapore*, 14–18 Sept. 2014, pp. 905–909.

[122] P. Ghahremani, V. Manohar, D. Povey, and S. Khudanpur, “Acoustic modelling from the signal domain using CNNs,” in *INTERSPEECH, San Francisco, California, USA*, 2016, pp. 3434–3438.

[123] N. Moritz, B. Kollmeier, and J. Anemuller, “Integration of optimized modulation filter sets into deep neural networks for automatic speech recognition,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 12, pp. 2439–2452, Dec. 2016.

[124] H. Seki, K. Yamamoto, and S. Nakagawa, “A deep neural network integrated with filterbank learning for speech recognition,” in *Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, USA, 2017, pp. 5480–5484.

[125] P. Sharma, V. Abrol, and A. K. Sao, “Deep sparse representation based features for speech recognition,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 11, pp. 2162–2175, Nov. 2017.

[126] P. Agrawal and S. Ganapathy, “Unsupervised modulation filter learning for noise-robust speech recognition,” *The Journal of the Acoustical Society of America*, vol. 142, no. 3, pp. 1686–1692, 2017.

[127] J. Salamon and J. P. Bello, “Unsupervised feature learning for urban sound classification,” in *Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, Australia, 2015, pp. 171–175.

[128] J. Salamon and J. P. Bello, “Feature learning with deep scattering for urban sound analysis,” in 23^{r}*d European Signal Processing Conference (EUSIPCO), Nice, France*, Aug. 2015, pp. 724–728.

[129] N. Chen, Y. Qian, H. Dinkel, B. Chen, and K. Yu, “Robust deep feature for spoofing detection-the SJTU system for ASVspoof 2015 challenge.” in *INTERSPEECH*, Dresden, Germany, 2015, pp. 2097–2101.

[130] Y. Qian, N. Chen, and K. Yu, "Deep features for automatic spoofing detection," *Speech Communication, Elsevier*, vol. 85, pp. 43–52, 2016.

[131] M. J. Alam, P. Kenny, V. Gupta, and T. Stafylakis, "Spoofing detection on the ASVspoof2015 challenge corpus employing deep neural networks," in *Odyssey 2016*, Bilbao, Spain, 2016, pp. 270–276.

[132] H. Dinkel, N. Chen, Y. Qian, and K. Yu, "End-to-end spoofing detection with raw waveform CLDNN," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA*, 2017, pp. 4860–4864.

[133] G. Lavrentyeva, S. Novoselov, E. Malykh, A. Kozlov, O. Kudashev, and V. Shchemelinin, "Audio replay attack detection with deep learning frameworks," *Proc. INTERSPEECH, Stockholm, Sweden*, pp. 82–86, 2017.

[134] H. Yu, Z. H. Tan, Y. Zhang, Z. Ma, and J. Guo, "DNN filter bank cepstral coefficients for spoofing detection," *IEEE Access*, vol. 5, pp. 4779–4787, March 2017.

[135] Y. Qian, N. Chen, H. Dinkel, and Z. Wu, "Deep feature engineering for noise robust spoofing detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 10, pp. 1942–1955, Oct. 2017.

[136] H. F. Alaie, L. Abou-Abbas, and C. Tadj, "Cry-based infant pathology classification using GMMs," *Speech Communication*, vol. 77, no. 1, pp. 28 – 52, 2016.

[137] J. B. Allen, "How do humans process and recognize speech?" *IEEE Transactions on Speech and Audio Processing*, vol. 2, no. 4, pp. 567–577, Oct 1994.

[138] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *27th International Conference on Machine Learning (ICML), Haifa, Israel*, 21–24 June 2010, pp. 807–814.

[139] G. Casella and E. I. George, "Explaining the Gibbs sampler," *The American Statistician*, vol. 46, no. 3, pp. 167–174, 1992.

[140] J. Qi, D. Wang, Y. Jiang, and R. Liu, "Auditory features based on gammatone filters for robust speech recognition," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS), Beijing, China*, 19–23 May 2013, pp. 305–308.

[141] Y. Hoshen, R. J. Weiss, and K. W. Wilson, "Speech acoustic modeling from raw multichannel waveforms," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brisbane, Queensland*, 19–24 April 2015, pp. 4624–4628.

[142] H. B. Barlow, "Possible principles underlying the transformations of sensory messages," *Sensory Communication*, vol. 35, no. 8, pp. 217–234, 1961.

[143] S. B. Laughlin and T. J. Sejnowski, "Communication in neuronal networks," *Science*, vol. 301, no. 5641, pp. 1870–1874, 2003.

[144] J. J. Eggermont, P. I. M. Johannesma, and A. M. H. J. Aertsen, "Reverse-correlation methods in auditory research," *Quarterly Reviews of Biophysics*, vol. 16, no. 3, p. 341–414, 1983.

[145] J. Kominek and A. W. Black, "The CMU-ARCTIC speech databases," in *Fifth ISCA Workshop on Speech Synthesis*, Pittsburgh, PA, USA, 2004.

[146] J. Bruna and S. Mallat, “Invariant scattering convolution networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1872–1886, Aug. 2013.

[147] R. Yeh, M. H. Johnson, and M. N. Do, “Stable and symmetric filter convolutional neural network,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2016*, Shanghai, China, March 2016, pp. 2652–2656.

[148] A. V. Oppenheim, A. S. Willsky, and S. H. Nawab, *Signals & Systems*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., Second Edition, 1996.

[149] H. Royden and P. Fitzpatrick, *Real Analysis*. Pearson Education, Forth Edition, 2015.

[150] Garofolo *et al.*, “DARPA TIMIT acoustic-phonetic continuous speech corpus CD-ROM. NIST speech disc 1-1.1,” *NASA STI/Recon Technical Report N*, vol. 93, 1993.

[151] D. B. Paul and J. M. Baker, “The design for the Wall Street Journal-based CSR corpus,” in *Proc. of the Workshop on Speech and Natural Language*, ser. HLT ’91. Stroudsburg, PA, USA: Association for Computational Linguistics (ACL), 1992, pp. 357–362.

[152] N. Parihar and J. Picone, “AURORA working group: DSR front end LVCSR evaluation AU/384/02,” *Tech. Rep., Inst. for Signal and Information Process, Mississippi State University*, 2002.

[153] L. Deng and D. O’Shaughnessy, *Speech Processing: A Dynamic and Optimization Oriented Approach*. Marcel Dekker Inc., First Edition, June 2003.

[154] K.-F. Lee and H.-W. Hon, “Speaker-independent phone recognition using hidden Markov models,” *IEEE Transactions on Acoustic, Speech and Signal Processing*, vol. 37, no. 11, pp. 1641–1648, Nov. 1989.

[155] D. Povey *et al.*, “The Kaldi speech recognition toolkit,” in *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU), Big Island, Hawaii, USA*, Dec. 2011, pp. 1–4.

[156] H. Xu, D. Povey, L. Mangu, and J. Zhu, “Minimum Bayes risk decoding and system combination based on a recursion for edit distance,” *Computer Speech & Language*, vol. 25, no. 4, pp. 802 – 828, 2011.

[157] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition?” in *IEEE International Conference on Computer Vision, Kyoto, Japan*, 29 Sept. - 2 Oct. 2009, pp. 2146–2153.

[158] D. Palaz, R. Collobert, and M. M. Doss, “Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks,” in *INTERSPEECH, Lyon, France*, 25–29 August 2013, pp. 1766–1770.

[159] J.-T. Huang, J. Li, and Y. Gong, “An analysis of convolutional neural networks for speech recognition,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, Queensland*, 19–24 April 2015, pp. 4989–4993.

[160] S. J. Rennie, V. Goel, and S. Thomas, “Annealed dropout training of deep networks,” in *IEEE Spoken Language Technology Workshop (SLT), Lake Tahoe, California*, 7–10 Dec. 2014, pp. 159–164.

[161] V. Mitra and et. al., "Evaluating robust features on deep neural networks for speech recognition in noisy and channel mismatched conditions," in *INTERSPEECH, Singapore*, 14–18 Sept. 2014, pp. 895–899.

[162] S. J. Rennie, V. Goel, and S. Thomas, "Annealed dropout training of deep networks," in *IEEE Spoken Language Technology Workshop (SLT), South Lake Tahoe, California and Nevada*, 2014, pp. 159–164.

[163] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR), San Diego*, 2015, pp. 1–11.

[164] H. Luo, Y. Wang, D. Poeppel, and J. Z. Simon, "Concurrent encoding of frequency and amplitude modulation in human auditory cortex: MEG evidence," *Journal of Neurophysiology*, vol. 96, no. 5, pp. 2712–2723, 2006.

[165] J. F. Kaiser, "On a simple algorithm to calculate the energy of a signal," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Albuquerque, New Mexico, USA, 1990, pp. 381–384.

[166] P. Maragos, J. F. Kaiser, and T. F. Quatieri, "On amplitude and frequency demodulation using energy operators," *IEEE Trans. on Signal Processing*, vol. 41, no. 4, pp. 1532–1550, 1993.

[167] F. Jabloun, A. E. Cetin, and E. Erzin, "Teager energy based feature parameters for speech recognition in car noise," *IEEE Signal Processing Letters*, vol. 6, no. 10, pp. 259–261, 1999.

[168] D. Dimitriadis, P. Maragos, and A. Potamianos, "Auditory Teager energy cepstrum coefficients for robust speech recognition." in *INTERSPEECH*, Lisbon, Portugal, 2005, pp. 3013–3016.

[169] M. L. Jepsen, S. D. Ewert, and T. Dau, "A computational model of human auditory signal processing and perception," *The Journal of the Acous. Soc. of Amer (JASA)*, vol. 124, no. 1, pp. 422–438, 2008.

[170] O. Abdel-Hamid, A. R. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Trans. on Audio, Speech, and Lang. Process.*, vol. 22, no. 10, pp. 1533–1545, Oct. 2014.

[171] M. Bisani and H. Ney, "Bootstrap estimates for confidence intervals in ASR performance evaluation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Montreal, Que., Canada*, vol. 1, 17–21 May 2004, pp. I–409–12 vol.1.

[172] Agricoop, "Ministry of Agriculture and Farmers Welfare, Government of India," URL: http://agricoop.nic.in, {Last Accessed on 22 December 2017}.

[173] A. Mohan, R. Rose, S. H. Ghalehjegh, and S. Umesh, "Acoustic modelling for speech recognition in Indian languages in an agricultural commodities task domain," *Speech Communication*, vol. 56, pp. 167–180, 2014.

[174] G. Mantena, S. Rajendran, B. Rambabu, S. Gangshetty, S. Yegnanarayana, and K. Prahallad, "A speech-based conversation system for accessing agricultural commodity prices in Indian languages," in *IEEE Joint Workshop on Hand-free Speech Communication and Microphone Arrays (HSCMA)*, India, 2011, pp. 153–154.

[175] S. Shahnawazuddin, D. Thatoppa, B. D. Sarma, A. Deka, S. R. M. Prasanna, and R. Sinha, "Assamese spoken query system to access the price of agricultural commodities," in *National Conference on Communication (NCC)*, India, March 2013, pp. 1–5.

[176] T. Godambe and K. Samudravijaya, "Speech data acquisition for voice-based agricultural information retrieval," in *Proc. of 39th All India DLA Conference*, Patiala, India, 2011, pp. 1–5.

[177] T. G. Yadava and H. S. Jayanna, "A spoken query system for the agricultural commodity prices and weather information access in Kannada language," *International Journal of Speech Technology (IJST), Springer*, vol. 20, no. 3, pp. 635–644, Sep. 2017.

[178] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA*, 8-12 September 2016, pp. 2751–2755.

[179] S. M. N. Woolley, T. E. Fremouw, A. Hsu, and F. E. Theunissen, "Tuning for spectro-temporal modulations as a mechanism for auditory discrimination of natural sounds," *Nature Neuroscience*, vol. 8, pp. 1371–1379, 2005.

[180] K. J. Piczak, "ESC: Dataset for environmental sound classification," in 23rd *Int. Conf. on Multimedia*, Brisbane, Australia, 2015, pp. 1015–1018.

[181] D. P. Ellis, "Prediction-driven computational auditory scene analysis," Ph.D. dissertation, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology (MIT), USA, 1996.

[182] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance of roads: a system for detecting anomalous sounds," *IEEE Trans. on Intell. Transp. Syst.*, vol. 17, no. 1, pp. 279–288, 2016.

[183] E. Alexandre, L. Cuadra, M. Rosa, and F. Lopez-Ferreras, "Feature selection for sound classification in hearing aids through restricted search driven by genetic algorithms," *IEEE Trans. on Audio, Speech, and Lang. Process.*, vol. 15, no. 8, pp. 2249–2256, 2007.

[184] M. Vacher, J.-F. Serignat, and S. Chaillol, "Sound classification in a smart room environment: an approach using GMM and HMM methods," in *IEEE Conference on Speech Technology and Human-Computer Dialogue (SpeD), Publishing House of the Romanian Academy (Bucharest)*, vol. 1, 2007, pp. 135–146.

[185] L. Ballan, A. Bazzica, M. Bertini, A. Del Bimbo, and G. Serra, "Deep networks for audio event classification in soccer videos," in *IEEE Int. Conf. on Multimedia and Expo (ICME)*, New York, USA, 2009, pp. 474–477.

[186] F. Chollet, "Keras," https://github.com/fchollet/keras { Last Accessed on 22 December, 2017}.

[187] J. Li, W. Dai, F. Metze, S. Qu, and S. Das, "A comparision of deep learning methods for environmental sound detection," in *IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, New Orleans, USA, 2017, pp. 126–130.

[188] R. G. Erra and J. Gervain, "The efficient coding of speech: Cross-linguistic differences," *Public Library of Science (PLOS) ONE*, vol. 11, no. 2, pp. 1–18, 2016.

[189] T. Overath, J. H. McDermott, J. M. Zarate, and D. Poeppel, "The cortical analysis of speech-specific temporal structure revealed by responses to sound quilts," *Nature Neuroscience*, vol. 18, no. 6, pp. 903–911, 2015.

[190] H. Lim, M. J. Kim, and H. Kim, "Cross-acoustic transfer learning for sound event classification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China*, March,2016, pp. 2504–2508.

[191] B. R. Glasberg and B. C. Moore, "Derivation of auditory filter shapes from notched-noise data," *Hearing Research*, vol. 47, no. 1, pp. 103–138, 1990.

[192] Z. Wu, N. Evans, T. Kinnunen, J. Yamagishi, F. Alegre, and H. Li, "Spoofing and countermeasures for speaker verification: a survey," *Speech Communication*, vol. 66, no. Supplement C, pp. 130–153, 2015.

[193] ISO/IEC Information Technology Task Force (ITTF), "Information technology – biometric presentation attack detection," URL: https://www.iso.org/standard/53227.html, 2016, {Last Accessed: 22 December 2017}.

[194] Y. Stylianou, "Voice trasformation: a survey," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Taipei, Taiwan, 2009, pp. 3585–3588.

[195] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.

[196] F. Alegre, A. Janicki, and N. Evans, "Re-assessing the threat of replay spoofing attacks against automatic speaker verification," in *IEEE International Conference of the Biometrics Special Interest Group (BIOSIG)*, Darmstadt, Germany, 2014, pp. 1–6.

[197] Y. W. Lau, M. Wagner, and D. Tran, "Vulnerability of speaker verification to voice mimicking," in *IEEE International Symposium on Intelligent Multimedia, Video and Speech Processing*, Hong Kong, 2004, pp. 145–148.

[198] C. Hanilçi, T. Kinnunen, M. Sahidullah, and A. Sizov, "Classifiers for synthetic speech detection: a comparison,," in *INTERSPEECH*, Dresden, Germany, 2015, pp. 2057–2061.

[199] M. Sahidullah, T. Kinnunen, and C. Hanilçi, "A comparison of features for synthetic speech detection," in *INTERSPEECH*, Dresden, Germany, 2015, pp. 2087–2091.

[200] Z. Wu, J. Yamagishi, T. Kinnunen, C. Hanilçi, M. Sahidullah, A. Sizov, N. Evans, and M. Todisco, "ASVspoof: The automatic speaker verification spoofing and countermeasures challenge," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 4, pp. 588–604, June 2017.

[201] Z. Wu, T. Kinnunen, N. W. D. Evans, J. Yamagishi, C. Hanilçi, M. Sahidullah, and A. Sizov, "ASVspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge,," in *INTERSPEECH*, Dresden, Germany, 2015, pp. 2037–2041.

[202] C. Zhang, C. Yu, and J. H. L. Hansen, “An investigation of deep learning frameworks for speaker verification anti-spoofing,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 4, pp. 684–694, 16 January 2017.

[203] M. Todisco, H. Delgado, and N. Evans, “A new feature for automatic speaker verification anti-spoofing: Constant Q cepstral coefficients,” in *Speaker Odyssey Workshop, Bilbao, Spain*, vol. 25, 2016, pp. 249–252.

[204] J. Gałka, M. Grzywacz, and R. Samborski, “Playback attack detection for text-dependent speaker verification over telephone channels,” *Speech Communication*, vol. 67, no. Supplement C, pp. 143 – 153, 2015.

[205] T. Kinnunen, N. Evans, J. Yamagishi, K. A. Lee, M. Sahidullah, M. Todisco, and H. Delgado, “ASVspoof 2017: Automatic speaker verification spoofing and countermeasures challenge evaluation plan,” in *INTERSPEECH, Stockholm, Sweden*, 2017.

[206] P. Nagarsheth, E. Khoury, K. Patil, and M. Garland, “Replay attack detection using dnn for channel discrimination,” *Proc. INTERSPEECH, Stockholm, Sweden*, pp. 97–101, 2017.

[207] M. Witkowski, S. Kacprzak, P. Zelasko, K. Kowalczyk, and J. Gałka, “Audio replay attack detection using high–frequency features,” *Proc. INTERSPEECH, Stockholm, Sweden*, pp. 27–31, 2017.

[208] S. Molau, F. Hilger, and H. Ney, “Feature space normalization in adverse acoustic conditions,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Hong Kong, China*, vol. 1, April 2003, pp. 656–659.

[209] M. Westphal, “The use of cepstral means in conversational speech recognition,” in *European Conference on Speech Communication and Technology (EUROSPEECH), Rhodes, Greece*, 1997, pp. 1143–1146.

[210] A. A. Garcia and R. J. Mammone, “Channel–robust speaker identification using modified–mean cepstral mean normalization with frequency warping,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (ICASSP), Phoenix, Arizona, USA*, 1999, pp. 325–328.

[211] P. G. Radadia and H. A. Patil, “A cepstral mean subtraction based features for singer identification,” in *International Conference on Asian Language Processing (IALP), Kuching, Malaysia*, Oct. 2014, pp. 58–61.

[212] T. Kinnunen, M. Sahidullah *et al.*, “RedDots replayed: A new replay spoofing attack corpus for text-dependent speaker verification research,” in *IEEE Int. Conf. on Acoust., Speech and Sig. Process. (ICASSP), New Orleans, LA, USA*, 2017, pp. 1–5.

[213] K. R. Alluri, S. Achanta, S. R. Kadiri, S. V. Gangashetty, and A. K. Vuppala, “SFF anti-spoofer: IIIT–H submission for automatic speaker verification spoofing and countermeasures challenge 2017,” *Proc. INTERSPEECH, Stockholm, Sweden*, pp. 107–111, 2017.

[214] H. A. Patil, M. R. Kamble, T. B. Patel, and M. Soni, “Novel variable length Teager energy separation based instantaneous frequency features for replay detection,” *Proc. INTERSPEECH, Stockholm, Sweden*, pp. 12–16, 2017.

[215] Z. Chen, Z. Xie, W. Zhang, and X. Xu, "ResNet and model fusion for automatic spoofing detection," *Proc. INTERSPEECH, Stockholm, Sweden*, pp. 102–106, 2017.

[216] W. Cai, D. Cai, W. Liu, G. Li, and M. Li, "Countermeasures for automatic speaker verification replay spoofing attack: On data augmentation, feature representation, classification and fusion," *Proc. INTERSPEECH, Stockholm, Sweden*, pp. 17–21, 2017.

[217] R. Font, J. M. Espín, and M. J. Cano, "Experimental analysis of features for replay attack detection–results on the ASVspoof 2017 challenge," *Proc. INTERSPEECH, Stockholm, Sweden*, pp. 7–11, 2017.

[218] O. Aragón, "Why do we cry?" *Scientific American Mind*, vol. 28, no. 2, p. 74, April 2017.

[219] A. Vingerhoets, *Why Only Humans Weep: Unravelling the mysteries of tears*. Oxford University Press, First Edition, 2013.

[220] E. Gustafsson, F. Levréro, D. Reby, and N. Mathevon, "Fathers are just as good as mothers at recognizing the cries of their baby," *Nature Communications*, vol. 4, no. 1698, pp. 1–6, 2013.

[221] O. Wasz-Höckert and et. al., "Twenty five years of Scandinavian cry research," in *Infant Crying: Theoretical and Research Perspectives*. C.F.Z. Boukydis and B.M. Lester, (Eds.), Springer, 1985, pp. 83–104.

[222] A. Chittrora, "Crying for a reason: A signal processing based approach for infant cry analysis and classification," *Ph.D. Thesis, Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT), Gandhinagar, Gujarat, India*, 2017.

[223] A. Chittora and H. A. Patil, "Data collection of infant cries for research and analysis," *Journal of Voice, Elsevier*, vol. 31, no. 2, pp. 252.e15 – 252.e26, 2017.

[224] R. Prescott, "Infant cry sound: developmental features," *The Journal of the Acoustical Society of America (JASA)*, vol. 57, no. 5, pp. 1186–1191, 1975.

[225] T. Etz, H. Reetz, C. Wegener, and F. Bahlmann, "Infant cry reliability: Acoustic homogeneity of spontaneous cries and pain-induced cries," *Speech Communication*, vol. 58, no. 1, pp. 91 – 100, 2014.

[226] S. Orlandi, C. A. R. Garcia, A. Bandini, G. Donzelli, and C. Manfredi, "Application of pattern recognition techniques to the classification of full-term and preterm infant cry," *Journal of Voice, Elsevier*, vol. 30, no. 6, pp. 656–663, 2016.

[227] L. Abou-Abbas, C. Tadj, C. Gargour, and L. Montazeri, "Expiratory and inspiratory cries detection using different signals' decomposition techniques," *Journal of Voice, Elsevier*, vol. 31, no. 2, pp. 259.e13 – 259.e28, 2017.

[228] A. Rosales-Pérez, C. A. Reyes-García, J. A. Gonzalez, O. F. Reyes-Galaviz, H. J. Escalante, and S. Orlandi, "Classifying infant cry patterns by the genetic selection of a fuzzy model," *Biomedical Signal Processing and Control*, vol. 17, no. 1, pp. 38 – 46, 2015.

[229] N. D. C. Society, "Causes of deafness," URL: http://www.deafchildworldwide.info, {Last Accessed on 20 December, 2017}.

[230] P. K. Kuhl and A. N. Meltzoff, "Infant vocalizations in response to speech: Vocal imitation and developmental change," *Journal of Acoustical Society of America (JASA)*, vol. 100, pp. 2425–2438, Oct. 1996.

[231] H. Hermansky, "History of modulation spectrum in ASR," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Dallas, Texas, USA, 15–19 March 2010, pp. 5458–5461.

[232] A. M. Saxe *et al.*, "Unsupervised learning models of primary cortical receptive fields and receptive field plasticity," in *25th Annual Conference on Neural Information Processing Systems, 12-14 December, Granada, Spain.*, 2011, pp. 1971–1979.

[233] H. Terashima and M. Okada, "The topographic unsupervised learning of natural sounds in the auditory cortex," in *26th Annual Conference on Neural Information Processing Systems 2012, December 3-6, 2012, Lake Tahoe, United States*, pp. 2321–2329.

[234] D. J. Klein, P. König, and K. P. Körding, "Sparse spectrotemporal coding of sounds," *EURASIP J. Adv. Sig. Proc.*, vol. 2003, no. 7, pp. 659–667, 2003.

[235] C. Lee, F. K. Soong, and K. Paliwal, *Automatic Speech and Speaker Recognition: Advanced Topics*. Springer Science & Business Media, First Edition, 2012.

[236] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Florence, Italy, May 2014, pp. 215–219.

[237] X. Zhang, M. G. Heinz, I. C. Bruce, and L. H. Carney, "A phenomenological model for the responses of auditory-nerve fibers: I. nonlinear tuning with compression and suppression," *The Journal of the Acoustical Society of America*, vol. 109, no. 2, pp. 648–670, 2001.

[238] B. Grothe, M. Pecka, and D. McAlpine, "Mechanisms of sound localization in mammals," *Physiological Reviews, American Physiological Society*, vol. 90, no. 3, pp. 983–1012, 2010.

[239] W. Zhang, H. Li, M. Yang, and N. Mesgarani, "Synaptic depression in deep neural networks for speech processing," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Shanghai, China*, March 2016, pp. 5865–5869.

[240] D. P. W. Ellis, "Time-domain scrambling of audio signals in MATLAB," URL: http://www.ee.columbia.edu/~dpwe/resources/matlab/scramble/, {Last Accessed on 20 December, 2017}.

[241] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, Second Edition, 2004.

[242] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters, Elsevier*, vol. 27, no. 8, pp. 861–874, 2006.

[243] W. J. Youden, "Index for rating diagnostic tests," *Cancer, Wiley Subscription Services, Inc., A Wiley Company*, vol. 3, no. 1, pp. 32–35, 1950.

[244] B. Matthews, "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochimica et Biophysica Acta (BBA), Protein Structure, Elsevier*, vol. 405, no. 2, pp. 442–451, 1975.

[245] H. A. Patil and S. Viswanath, "Effectiveness of Teager energy operator for epoch detection from speech signals," *Int. J. Speech Technol., Springer*, vol. 14, no. 4, pp. 321–337, Dec. 2011.

[246] H. A. Patil and M. C. Madhavi, "Combining evidences from magnitude and phase information using VTEO for person recognition using humming," *Computer Speech and Language, Elsevier*, 2017.

[247] I. J. Myung, "Tutorial on maximum likelihood estimation," *J. Math. Psychol.*, vol. 47, no. 1, pp. 90–100, Feb. 2003.

[248] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, Second Edition, 2006.

List of Publications from Thesis

Journal Papers

1. Hardik. B. Sailor and Hemant. A. Patil, “Novel unsupervised auditory filterbank learning using convolutional RBM for speech recognition,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 24, no. 12, pp. 2341-2353, Dec. 2016.
2. Hardik B. Sailor and Hemant A. Patil, "Auditory feature representation using convolutional restricted Boltzmann machine and Teager energy operator for speech recognition", The Journal of the Acoustical Society of America Express Letters (JASA-EL), vol. 141, no. 6, pp. EL500–EL506, June 2017.

Book Chapters

1. Hardik B. Sailor and Hemant A. Patil, “Unsupervised auditory filterbank learning for infant cry classification," *submitted* in Voice Technologies for Reconstruction and Enhancement, H. A. Patil and A. Neustein, (Eds.), De Gruyter Series in Speech Technology and Text Analytics in Medicine and Healthcare, 2018, pp. 1–18.

Conference Papers

1. Hardik. B. Sailor, Madhu R. Kamble, and Hemant A. Patil, “Auditory filterbank learning for temporal modulation features in replay spoof speech detection," to appear in INTERSPEECH, Hyderabad, Sept. 2018.
2. Hardik. B. Sailor and Hemant A. Patil, “Auditory filterbank learning using ConvRBM for infant cry classification," to appear in INTERSPEECH, Hyderabad, Sept. 2018.
3. Hardik. B. Sailor and Hemant. A. Patil, “Representation learning for speech recognition system in agricultural commodity for Gujarati", in Global Conference on Cyberspace (GCCS), MeitY, Govt. of India under National e-Governance Division (NeGD), New Delhi, India, 23-24 Nov. 2017.
4. Hardik B. Sailor, Madhu R. Kamble and Hemant. A. Patil, “Unsupervised representation learning using convolutional restricted Boltzmann machine

for spoof speech detection", in INTERSPEECH 2017, Stockholm, Sweden, pp. 2601-2605.

5. Hardik B. Sailor, Dharmesh M. Agrawal and Hemant. A. Patil, “Unsupervised filterbank learning using convolutional restricted Boltzmann machine for environmental sound classification", in INTERSPEECH 2017, Stockholm, pp. 3107-3111.

6. Dharmesh M. Agrawal, Hardik B. Sailor, Meet H. Soni, and Hemant A. Patil, "Novel TEO-based gammatone features for environmental sound classification", in European Signal Processing Conference (EUSIPCO), 2017, Kos Island, Greece, pp. 1809-1813.

7. Hardik. B. Sailor and Hemant. A. Patil, “Unsupervised deep auditory model using stack of convolutional RBMs for speech recognition,” in INTERSPEECH 2016, San Francisco, California, USA, September 2016, pp. 3379-3383.

8. Avni Rajpal, Tanvina B. Patel, Hardik B. Sailor, Maulik C. Madhavi, Hemant A. Patil and Hiroya Fujisaki, “Native language identification using spectral and source-based features," in INTERSPEECH 2016, San Francisco, USA, September 2016, pp. 2383-2387.

9. Hardik. B. Sailor and Hemant. A. Patil, “Filterbank learning using convolutional restricted Boltzmann machine for speech recognition,” in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Shanghai, China, Mar. 2016, pp. 5895-5899.

10. Hardik. B. Sailor and Hemant. A. Patil, “Unsupervised learning of temporal receptive fields using convolutional RBM for ASR task,” in European Signal Processing Conference (EUSIPCO), Budapest, Hungary Aug./Sep. 2016, pp. 873-877.

11. Mohammadi Zaki, Hardik B. Sailor and Hemant A. Patil, “Analysis of hierarchical bottleneck framework for improved phoneme recognition," accepted in International Conference on Signal Processing and Communications (SPCOM), IISc Bangalore, India, 12-15 June, 2016, pp. 1-5.

12. Anshu Chittora, Hemant A. Patil and Hardik B. Sailor, “Spectro-temporal Analysis of HIE and Asthma Infant Cries Using Auditory Spectrogram," in International Conference on BioSignal Analysis, Processing and System (ICBAPS), Kuala Lumpur, Malaysia, on 26-28 May 2015.

Brief Biography

Hardik B. Sailor received B.E. degree from Government Engg. College (GEC), Surat in 2010. In 2013, he received M.Tech degree from Dhirubhai Ambani Institute of Information and Communication Technology (DA-IICT), Gandhinagar. Currently he is a doctoral student under a supervision of Prof. Hemant A. Patil at DA-IICT, Gandhinagar. He was a project staff member of MeitY, Govt. of India sponsored project Automatic Speech Recognition for Agricultural Commodities Phase-II (April 2016 - April 2018). At DA-IICT, He was also a project staff member of MeitY, Govt. of India sponsored project on Development of Text-to-Speech (TTS) Synthesis Systems for Indian languages Phase-II, from May 2012 - March 2016.

He has published 25 research papers in top conferences and peer-reviewed journals. His research area includes representation learning, deep learning, auditory processing, and Automatic Speech Recognition (ASR). His main research is focused on developing representation learning techniques to model the auditory processing. He is a student member of IEEE, IEEE Signal Processing Society, and International Speech Communication Association (ISCA). He is also an invited reviewer for IEEE/ACM Transactions on Audio, Speech, and Language Processing, IEEE Signal Processing Letters, IEEE Access, and Applied Acoustics, Elsevier. He received ISCA student grant of 650 Euros to present his three research papers during INTERSPEECH 2018, Hyderabad.