Multilayer Perceptron Based Hierarchical Acoustic Modeling for Automatic Speech Recognition

THÈSE N^{o} 4649 (2010)

PRÉSENTÉE LE

À LA FACULTÉ SCIENCES ET TECHNIQUES DE L'INGÉNIEUR

LABORATOIRE DE L'IDIAP

PROGRAMME DOCTORAL EN GÉNIE ÉLECTRIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Joel Praveen Pinto

acceptée sur proposition du jury:

Prof. J. R. Mosig, président du jury

Prof. Hervé Bourlard, directeur de thèse

Prof. Martin Hasler, rapporteur

Prof. Simon King, rapporteur

Dr. Ralf Schlueter, rapporteur

Lausanne, EPFL

2010

Résumé

Dans cette thèse, nous proposons une approche hiérarchique afin d'évaluer les probabilités conditionnelles des classes phonétiques utilisant des "Multilayer Perceptrons (MLP)", type de réseaux de neurones couramment utilisé. L'architecture choisie est composé de deux classificateurs MLP en cascade. Le premier classificateur est entraîné de façon standard en utilisant des paramètres acoustiques tenant compte du contexte temporel sur une duré d'environs 90 ms. Le deuxième classificateur MLP est entraîné à partir des probabilités conditionnelles des classes phonétiques (ou paramètres postérieurs) estimées par le premier classificateur, en tenant compte d'un contexte temporel plus long cette fois-ci, avoisinant les 150-250 ms.

Le choix de l'architecture hiérarchique est motivé par la possibilité d'exploiter l'information contextuelle présente dans la séquence des paramètres postrieurs, qui contient l'évolution des valeurs de probabilité dans un phonème (sub-phonemic) ainsi que la transition depuis ou vers les phonèmes voisins (sub-lexical). Etant donné que les paramètres postérieurs sont epars et simples, le deuxième classificateur est capable d'obtenir l'information contextuelle sur une durée de 250ms. Des manipulationa effectuées sur la reconnaissance des phonèmes, de même que sur la retranscription écrite de la parole lors de conversations, montrent que l'approche hiérarchique conduit à des performances significativement meilleures. L'analyse du second classificateur MLP utilisant des séries Volterra, montre que les paramètres phonétiques et temporels sont représents dans l'espace des paramètres postérieurs. Ces paramètres phonétiques et temporels capturent les erreurs de classifications des phonèmes à la sortie du premier classificateur, de même que les phonotactics du langage observés dans l'ensemble des donnes d'entrainement. De plus, nous montrons lors de ce travail, que le second classificateur MLP est simple puisqu'il contient un nombre limité de paramètres dans le modèle est peut être entraîné sur un ensemble plus petit de données.

L'utilité de l'approache hiérarchique proposée par ce travail, servant à modéliser les paramètres acoustiques lors de la reconnaissance automatique de la parole, est démontrée à travers deux applications : (a) l'adaptation de cette tâche en exploitant les MLPs entraînés sur une grande quantité de données pour d'autres nouvelles tâches et

(b) la reconnaissance automatique de la parole utilisant un large vocabulaire d'émissions d'information radiophoniques ou télévisées en Mandarin. La reconnaissance de mots isolés utilisant un vocabulaire limité, ainsi que les études d'adaptation des tâches ont été exécutées sur la base de données "Phonebook". La reconnaissance de la parole utilisant un vocabulaire plus dispersé a été experimentée sur la base de données de Mandarin "DARPA GALE".

Mots Clés : Multilayer perceptron, système hiérarchique, séries de Volterra.

Abstract

In this thesis, we investigate a hierarchical approach for estimating the phonetic class-conditional probabilities using a multilayer perceptron (MLP) neural network. The architecture consists of two MLP classifiers in cascade. The first MLP is trained in the conventional way using standard acoustic features with a temporal context of around 90 ms. The second MLP is trained on the phonetic class-conditional probabilities (or posterior features) estimated by the first classifier, but with a relatively longer temporal context of around 150-250 ms.

The hierarchical architecture is motivated towards exploiting the useful contextual information in the sequence of posterior features which includes the evolution of the probability values within a phoneme (sub-phonemic) and its transition to/from neighboring phonemes (sub-lexical). As the posterior features are sparse and simple, the second classifier is able to learn the contextual information spanning a context as long as 250 ms. Extensive experiments on the recognition of phonemes on read speech as well as conversational speech show that the hierarchical approach yields significantly higher recognition accuracies. Analysis of the second MLP classifier using Volterra series reveal that it has learned the phonetic-temporal patterns in the posterior feature space which captures the confusions in phoneme classification at the output of the first classifier as well as the phonotactics of the language as observed in the training data. Furthermore, we show that the second MLP can be simple in terms of the number of model parameters and that it can be trained on lesser training data.

The usefulness of the proposed hierarchical acoustic modeling in automatic speech recognition (ASR) is demonstrated using two applications (a) task adaptation where the goal is to exploit MLPs trained on large amount of data and available off-the-shelf to new tasks and (b) large vocabulary continuous ASR on broadcast news and broadcast conversations in Mandarin. Small vocabulary isolated word recognition and task adaptation studies are performed on the Phonebook database and the large vocabulary speech recognition studies are performed on the DARPA GALE Mandarin database.

Keywords: Multilayer perceptron, hierarchical system, Volterra series.

Acknowledgements

I express my sincere gratitude to my thesis director, Prof. Hervé Bourlard, for his guidance and constant encouragement. His insightful feedbacks have helped me greatly in improving the quality of my thesis. I am thankful to my former advisor, Prof. Hynek Hermansky, for the sound advice and encouragement. His push towards analysis more than better system performance helped me in finding an exciting research problem. I also thank Dr. Mathew Magimai.-Doss, my co-advisor during the latter part of my doctoral research, for all his help. My research and thesis benefited immensely from the numerous stimulating discussions I had with him. I thank the Swiss National Science Foundation for funding my research under the Indo-Swiss joint research program.

It was a pleasure working with several (former and present) researchers and visitors at Idiap Research Institute. I thank Andrew Lovitt, Deepu Vijayasennan, Fabio Valente, Guillermo Aradilla, Hamed Ketabdar, Hemant Misra, Jithendra Vepa, John Dines, Petr Motlicek, Phil Garner, Prasanna Sompura, Samuel Thomas, Sriram Ganapathy, and Yegnanarayana Bayya for their help at various levels. I also thank Sarah Favre for helping me with the abstract in French, Sivaram Garimella for the scientific collaboration, and Hari Parthasarathi for the technical discussions and commenting on my research articles.

I owe my deepest gratitude to my parents, Simon and Faustine Pinto, for their love and support throughout my career. In particular, I thank them for inspiring me to pursue research. I also thank my brothers, sisters and their families and my wife's family for their encouragement and support.

My wife, Shakila, has been a pillar of strength to me during the last four years. This thesis would not have been completed without her love, patience, encouragement, and numerous sacrifices. The sweetness of this success is even more sweeter as we share it with our one year old daughter Laurel, who is now learning to recognize speech.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Automatic speech recognition (ASR) refers to the process of automatically transcribing a spoken utterance into its corresponding text. Recognition is typically performed by integrating three sources of information. A language model that captures the grammar or syntax of a language, a pronunciation dictionary that maps the words into its constituent phonemes, and an acoustic model that captures the acoustical (*e.g.,* spectro-temporal) properties of speech for each of the phonemes. Acoustic modeling is the most challenging and extensively researched component of the system. The Hidden Markov model (HMM) has been the mainstream of acoustic modeling ever since its introduction in the seventies.

The state emission distribution of the HMM provides the link between the acoustic observations and the underlying linguistic units. Traditionally, Gaussian mixture models (GMM) have been used to model this distribution. With several refinements and extensions, the HMM/GMM modeling remains the predominant acoustic modeling technique in state-of-the-art ASR systems. The last two decades has seen the emergence of artificial neural networks, particularly the multilayer perceptron (MLP) for acoustic modeling in ASR.

The input to the MLP are standard acoustic features such as mel frequency cepstral coefficients and its output classes represent the subword units of speech such as phonemes. A well trained MLP classifier estimates the posterior probabilities of its output phonetic classes conditioned on the input features. The phonetic class-conditional probabilities estimated by the discriminatively trained MLP are typically used in HMM based ASR. In the hybrid HMM/MLP approach, the MLP

is used as a scaled likelihood estimator in place of the conventional GMM. In the Tandem approach, the output of the MLP is transformed appropriately and used as features to a standard HMM/GMM system. As phoneme posterior probabilities can also be viewed and used as local representation of speech in the same way as standard acoustic features, they are commonly referred to as posterior features.

1.1 Objective

The primary objective of this thesis is to investigate the presence of useful contextual information in the sequence of posterior features and to explore possible ways to exploit this information towards obtaining more accurate estimates of the phonetic class-conditional probabilities. The second objective is to come up with an analysis framework that enable us to interpret the functionality of the parameters, namely the weights and biases, of the MLP classifier trained to estimate the phonetic class-conditional probabilities.

1.2 Motivation

In the posterior feature space, each dimension corresponds to a phoneme. The posterior feature vector at a particular time instant is a point in the posterior feature space, representing the instantaneous soft-decision on the underlying phonemes. It carries useful information such as the probability mass assigned to the competing phonemes. The sequence of posterior feature vectors is a trajectory in the posterior feature space, and it carries additional contextual information such as the evolution of these probabilities within a phoneme (sub-phonemic level) and its transition to neighboring phonemes (sub-lexical level). Since the posterior features have a sparse distribution, we hypothesize that the contextual information spanning longer contexts can be effectively learned in the posterior feature space by training a second classifier. Furthermore, as the posterior features have lesser nonlinguistic variabilities such as speaker and environmental characteristics, we expect the second classifier to be able to be trained using lesser data.

Although MLP based acoustic modeling has shown to help in the improvement of speech recognition accuracies, once trained its parameters are not further analyzed. The goodness of the trained

model is typically evaluated by first estimating the phonetic class-conditional probabilities on the cross-validation or a test set, and then using measures such as frame-level phoneme classification accuracy, the cross-entropy between the labels and the estimated posterior probabilities, phonetic confusion matrices, or the final speech recognition accuracies. While these measures can indicate how well the model is trained, it does not reveal any information about the properties of speech such as spectro-temporal patterns that are learned by the trained parameters of the MLP for each of the phonemes. We believe that a better understanding of the functionality of the system can eventually lead to better feature extraction and modeling approaches.

If the MLP used in the acoustic modeling is analyzed as a standalone system, then its functionality will be revealed in terms of the input features (*e.g.,* cepstral parameters) which are not directly interpretable. On the other hand, if a part of the feature extraction is included into the analysis framework, then the functionality of the combined system is revealed in terms of more intuitive information such as spectro-temporal patterns.

1.3 Contribution

The contributions of this thesis are

- We propose a generic mathematical framework to represent a cascade of a linear time invariant system and a three-layered MLP using Volterra series. By incorporating the linear system, we can include a part of the feature extraction process into the analysis, and thereby interpret the model parameters in terms of spectro-temporal patterns. The major contributions of this work include: (a) development of a mathematical framework to apply Volterra series to a nonlinear dynamic system consisting of a cascade of a finite impulse response filter bank and a three-layered MLP classifier (b) calculation of the Volterra kernels of the above nonlinear dynamic system in terms of the parameters of the system (c) modifications to the Volterra kernels when the features to the MLP are normalized to zero-mean and unit-variance (d) handling the case where a linear transformation matrix precedes the FIR filter bank, and (e) demonstration of the applicability of the proposed framework to analyze MLP classifiers which are trained on mel filter bank energy features, multi-resolution relative spectra features, and the more conventional mel frequency cepstral features.

- We propose an MLP based hierarchical approach for estimating the phonetic class-conditional probabilities. The architecture consists of two MLP classifiers in tandem. The first classifier is trained in the conventional way using the standard acoustic features. The contextual information in the estimated posterior features is learned by training a second MLP classifier with a temporal context spanning 150-250 ms.

  Through extensive phoneme recognition studies and the analysis of second MLP in the hierarchical system using Volterra series, we show that (a) the hierarchical system yields higher phoneme recognition accuracies compared to a single MLP based system (b) the posterior features contain useful contextual information spanning around 150-230 ms of temporal context (c) the second MLP in the hierarchical system learns the phonetic-temporal patterns in the posterior features, which includes the phonetic confusion patterns at the output of the first classifier and to a certain extent the phonotactics of the language as observed in the training data, and (d) the classifier at the second stage of the hierarchy requires fewer number of parameters and lesser amount of training data.

- We investigate the application of the hierarchical system for task adaptation, where an MLP trained on a large amount of out-of-domain data is used at the first stage of the hierarchical system. The MLP at the second stage of the system is trained on the in-domain or adaptation data. Task adaptation is demonstrated by using an MLP trained on 232 hours of conversational telephone speech for recognition of isolated words on the Phonebook database. The hierarchical adaptation yields lower error rates even when compared to the matched conditions. In addition, we show that the second MLP can be simpler in terms of the number of parameters and lesser amount of adaptation data is sufficient.

- The effectiveness of the hierarchical approach in estimating phonetic class-conditional probabilities is investigated in Tandem based large vocabulary continuous speech recognition. On the challenging DARPA GALE Mandarin task, we show that the hierarchical Tandem system yields lower error rates when compared to the conventional single MLP based system on both broadcast conversations and broadcast news.

1.4 Organization

This thesis is organized as follows:

- Chapter 2 is an overview of automatic speech recognition in the context of the research carried out in this thesis. We introduce multilayer perceptron based acoustic modeling and discuss its advantages and application in hidden Markov model based ASR.
- In Chapter 3, we propose a generic mathematical framework to represent a cascade of a linear time invariant system and a three-layer MLP using Volterra series. We discuss the application of the proposed framework in interpreting the functionality of the MLP classifiers trained to estimate the phonetic class-conditional probabilities.
- In Chapter 4, we propose an MLP based hierarchical approach for estimating the phonetic class-conditional probabilities. The usefulness of the proposed approach is demonstrated in the recognition of phonemes. Furthermore, we investigate the reasons for the effectiveness of the hierarchical system and analyze the functionality of the second stage in hierarchical system using Volterra analysis discussed in Chapter 3.
- In Chapter 5, the hierarchical system is investigated in task adaptation, where the goal is to exploit MLP classifiers trained on a large amount of data and available off-the-shelf to new tasks or application scenarios.
- Chapter 6 discusses the application of the hierarchical system in large vocabulary speech recognition in Mandarin.
- Chapter 7 provides a short summary of the thesis and discusses future directions.

Chapter 2

Speech Recognition: An Overview

2.1 Introduction

Speech is the most natural and convenient mode of communication among humans. Figure 2.1 shows a block schematic of speech communication between two humans. The talker's brain gener-

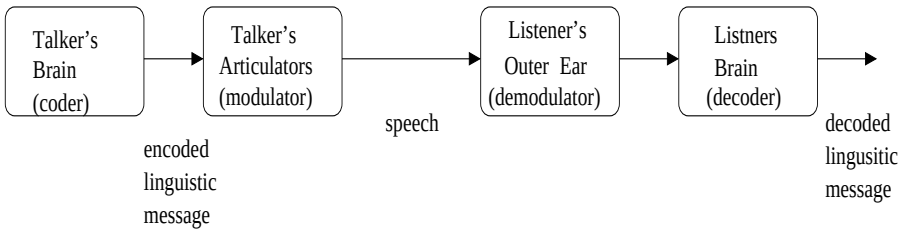


Figure 2.1. Block schematic of human speech communication.

ates the linguistic message that appropriately moves the articulators (vocal tract, lips, tongue etc) to generate the desired speech. The listeners outer ear acts as a frequency analyzer and converts the acoustic pressure wave to an intermediate representation, which is processed by the listener's brain to decode the intended linguistic message. This formulation can be viewed as a communication problem consisting of a coder and modulator on the talker's side and a demodulator and decoder on the listener's side.

The objective of automatic speech recognition (ASR) is to automatically transcribe speech (intended to another person or directed to a computer) into its corresponding text. In other words, the

goal is to partially[1] emulate the functionality of the listener in Figure 2.1. However, unlike electrical communication systems where the demodulator-decoder is essentially the reverse process of the coder-modulator, we do not have such an advantage in designing an ASR system. As Jelinek puts it in his book (Jelinek, 2001), "*We must make do with the coder-modulator that evolution has bequeathed to us: human language and speech.*" Hence, from a communication theoretic point of view, automatic speech recognition can be viewed as designing the decoder for the speech signal without precise knowledge of the coder-modulator. Because of this constraint, ASR has been typically approached as a statistical pattern recognition problem as shown in the following block schematic.

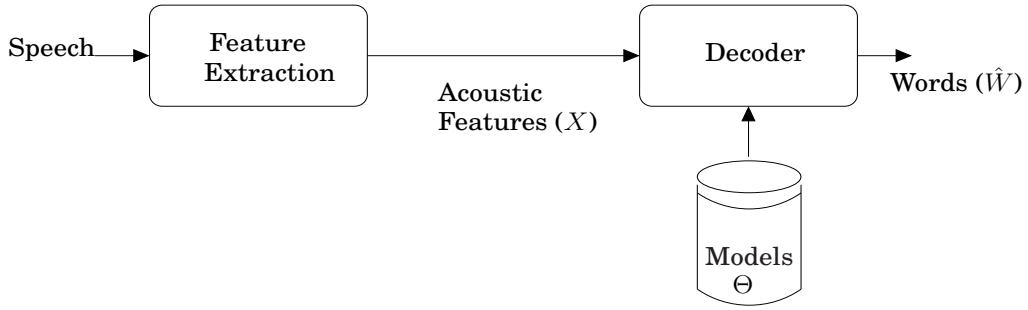


Figure 2.2. Block schematic of an automatic speech recognition system.

Speech propagates through the air as acoustic pressure waves. The transducers (*e.g.,* a microphone) converts the changes in the acoustic pressure into electrical signals. On a computer, the speech signal is first discretized by sampling in time typically at a frequency of 8000 Hz or 16000 Hz. It is then quantized and stored as a digital signal. Apart from the linguistic message, the speech signal also carries information which is irrelevant to automatic recognition of speech such as characteristics of speakers, channel, and the environment. The objective of feature extraction is to extract the useful acoustic correlates of the underlying linguistic message from the signal, while suppressing undesirable nonlinguistic information.

Given a sequence of acoustic feature vectors X and a trained ASR model Θ, the decoder attempts to find the sequence of words $\hat{W}$ which maximizes the *a posteriori* probability $P(W|X,\Theta)$. Mathematically, this can be written as

$$\hat{W} = \arg\max_{W} \quad P(W|X,\Theta)$$

[1]Understanding the linguistic message is beyond the scope of present day ASR systems.

By using Bayes' rule, the above expression can be written as

$$\hat{W} = \arg\max_{W} [p\,(X|W,\Theta) \quad P\,(W|\Theta)] \tag{2.1}$$

where, $p(X|W,\Theta)$ denotes the likelihood of the sequence of feature vectors conditioned on the sequence of words, and $P(W|\Theta)$ denotes the prior probability of the word sequence. The ASR model Θ is made up of two components - the acoustic model Θ_a and the language model Θ_l, which are estimated independently. With this assumption, (2.1) can be written as [2]

$$\hat{W} = \arg\max_{W} [p\,(X|W,\Theta_a) \quad P\,(W|\Theta_l)] \tag{2.2}$$

The acoustic model Θ_a is estimated on a training data set consisting of the spoken utterances and its corresponding transcription. The language model Θ_l helps in restricting the search space to grammatically well-formed and meaningful sentences. The language model is estimated from a large text corpus which is related to the task at hand. In the following subsections, we discuss feature extraction, language modeling, and acoustic modeling in detail.

2.2 Feature Extraction

Figure 2.3 shows the source-filter model (Fant, 1960) of human speech production followed by a linear model for transmission. According to the source-filter model, the production apparatus consists of an excitation source $e(n)$ which generates an impulse train or white noise depending on whether the underlying sound is voiced or not. The time varying filter $g(n)$ represents the vocal tract, and depending on the shape of the vocal tract, the speech signal $s(n)$ corresponding to different sounds is produced.

Figure 2.3 also shows a simplified model for the transmission of the speech from the talker's mouth to the computer for recognition. The channel effects, modeled by $h(n)$ include the room acoustics such as reverberation or the frequency response of the transmission line in the case of telephone speech. In addition, the speech could be corrupted by ambient noise which is additive in

[2] In practice, the total likelihood in the $\log$ domain is given by $\log p(X|\Theta_a) + \alpha \log P(W|\Theta_l) + \beta|W|$, where α is the language model scaling factor and β is the word insertion penalty, and $|W|$ denotes the number of words in the sequence W. The constants α and β are determined empirically on the development data.

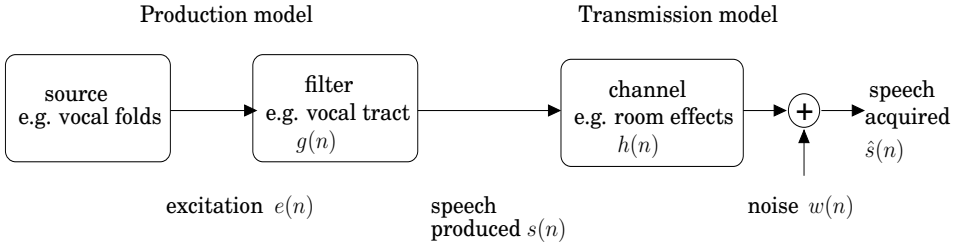


Figure 2.3. Block schematic of the source-filter model of speech production followed by a model for transmission.

nature. According to the above simplistic model, the speech acquired for ASR $\hat{s}(n)$ is given by

$$\hat{s}(n) = w(n) + h(n) * s(n), \quad \text{where} \quad s(n) = g(n) * e(n) \tag{2.3}$$

The acquired speech signal exhibits two important attributes, namely redundancy and variability. Redundancy in the speech signal is extremely useful in preserving the quality and naturalness of speech when it is played back. For automatic speech recognition, however, it is sufficient to extract the important acoustic correlates of the underlying linguistic message, while suppressing the redundant information. In this way, feature extraction can also be viewed as data compression.

The speech signal also exhibits a high degree of variability. Previous works in the literature (Zue, 1985; Klatt, 1985) provide in-depth insights on the variability in the speech signal. We review these here in relation to the simplified model of speech production/transmission discussed in Figure 2.3.

- Environmental variability: This includes the effect of the channel $h(n)$ and the additive noise $w(n)$. The environmental variability is typically compensated by post-processing the acoustic features. A commonly used technique is cepstral mean normalization (Atal, 1974), which has been shown to provide robustness against channel effects under certain conditions.

- Within-speaker variability: The characteristics of the same speech sound from a speaker can vary at different times due to the physiological conditions or the emotive state of the speaker. These conditions affect both the excitation signal $e(n)$ as well as the vocal tract response $g(n)$.

- Across-speaker variability: The characteristics of speech sounds from different speakers can vary due to the differences in the size of the vocal folds, size of the vocal tracts or even the differences in the socio-linguistic background of the speakers. These factors affect the characteristics of the filter $g(n)$ as well as the excitation source $e(n)$. The variability in speech due to

the size of the vocal tract is typically compensated using vocal tract length normalization (Lee and Rose, 1996).

- Coarticulatory effects: The acoustic characteristics of phonemes is affected by the neighboring phonetic context due to coarticulation. This variability is best handled by context dependent modeling.

In short, the goal of feature extraction is threefold: (a) extract useful acoustic correlates which are relevant to the identification of the underlying sound[3] (b) suppress the effect of nonlinguistic factors such as speaker and environmental variability and (c) achieve data compression.

Figure 2.4 is a contrastive block schematic of four feature extraction techniques which are commonly used in ASR (a) linear predictive cepstral coefficients (LPCC) (Makhoul, 1975), (b) mel frequency cepstral coefficients (MFCC) (Davis and Mermelstein, 1980), (c) perceptual linear predictive cepstral coefficients (PLP) (Hermansky, 1990) and (d) multi-resolution relative spectral (MRASTA) features (Hermansky and Fousek, 2005).

Pre-emphasis

The speech signal has an overall spectral slope of -6dB per octave due to the combined effect of glottal pulse roll-off (-12dB per octave) and the lip radiation (+6dB per octave). This slope can be compensated by performing pre-emphasis on the speech signal as $s'(n) = s(n) - \alpha s(n-1)$, where α typically takes values between 0.90 and 0.98. Moreover, pre-emphasis also helps removing the dc component in the speech signal.

Short time Fourier Analysis

Spectral analysis is the common stage in most of the state-of-the-art feature extraction techniques in ASR. As speech is quasi-stationary, its short-time Fourier magnitude spectrum is typically estimated using an analysis window size of 25 ms and a frame shift of 10 ms. The power spectrum is subsequently computed by squaring the magnitude spectrum. A plot of the power spectrum (in decibels) of speech as a function of time is also known as a spectrogram. The acoustic features for

[3]Although this information is mainly captured by the filter response $g(n)$ in Figure 2.3, studies have shown that the excitation signal $e(n)$ can contain additional complimentary information such as the pitch (Stephenson *et al.*, 2004). Appending pitch information to standard acoustic features has been found to be particularly useful in recognition of speech in tonal languages such as Mandarin (Lei *et al.*, 2006).

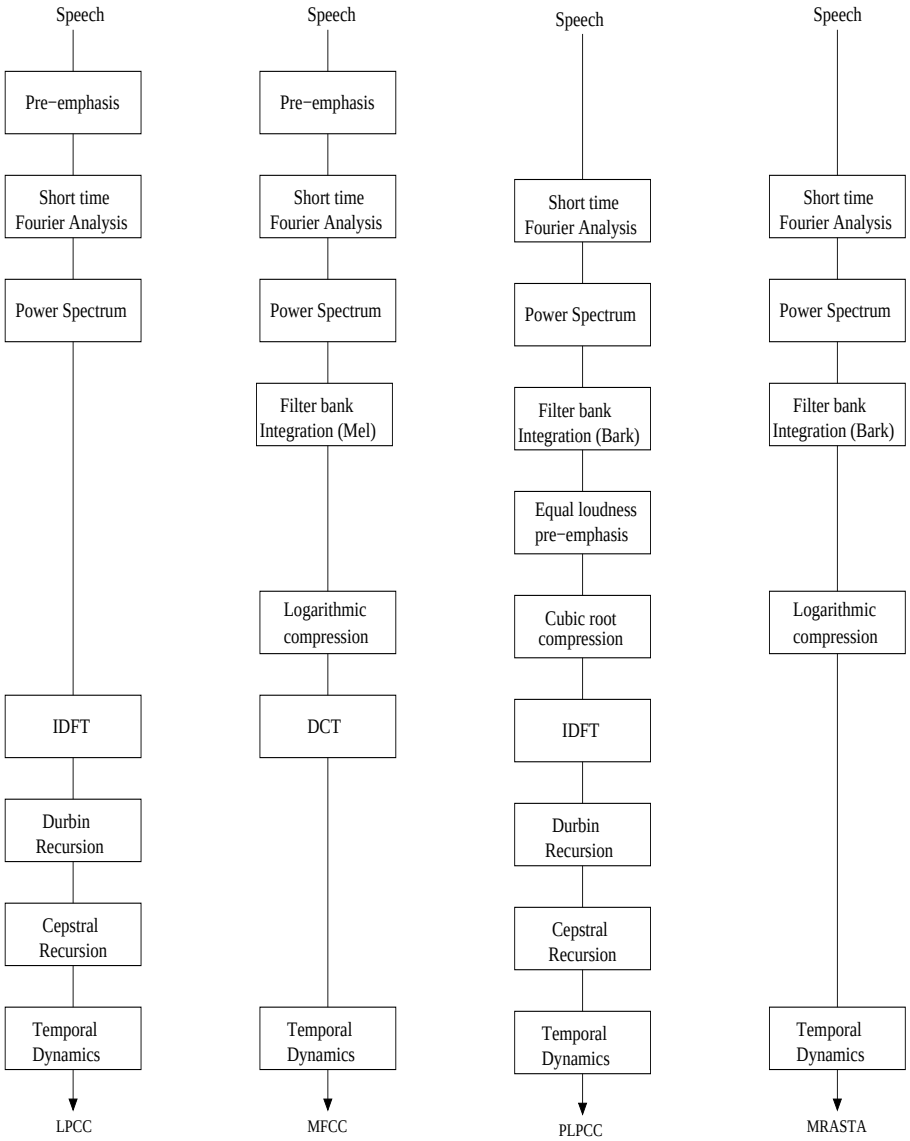


Figure 2.4. Comparison of different feature extraction techniques. The figure is motivated from a similar comparative block schematic Figure 22.4 in (Gold and Morgan, 1999).

ASR are derived from the spectrogram by processing it along both frequency and time as discussed in the following subsections.

Processing along Frequency

In this section, we discuss the processing along frequency for different feature extraction techniques

discussed in Figure 2.4.

- Linear predictive cepstral coefficients (LPCC):

  The linear predictive analysis of speech is based on the source-filter model for human speech production (Makhoul, 1975). The idea is to model the filter in Figure 2.3 as an all-pole model. The linear predictive coefficients parameterize the filter which carries information on the underlying speech sound, and this is the motivation for using them as features.

  The linear predictive coefficients can be estimated from the auto-correlation matrix, which can be estimated from the speech signal in the time domain. However, for comparative illustration with other feature extraction techniques, the auto-correlation is computed as the inverse Fourier transform of the power spectrum in Figure 2.4.

  It is well known that the useful information about the speech sounds lie in the gross shape of its spectrum, and not in the finer details. In this respect, the power spectrum of the filter (parameterized by the linear predictive coefficients) is a smoothed estimate of the power spectrum of the speech. In addition, auto-regressive modeling exhibits peak-hugging property where the peaks in the power spectrum are matched better in comparison with its valleys (Makhoul, 1975). The linear predictive cepstral coefficients are computed using the recursion formula described in (Markel and Gray, 1976).

- Mel frequency cepstral coefficients (MFCC):

  Spectral smoothing can also be achieved by integrating the power spectrum within overlapping critical band filters. In MFCC feature extraction, the frequency axis is first warped to the mel psychoacoustic scale, which is roughly linear below 1kHz and roughly logarithmic above this point. Triangular filters which are equally spaced in the mel scale are applied on the warped spectrum. The output of the filters are compressed using the logarithm function and cepstral coefficients are computed by applying the discrete cosine transformation (DCT). Further smoothing is achieved by dropping the higher order cepstral coefficients, which are known to contain mainly speaker specific information.

- Perceptual linear predictive cepstral coefficients (PLPCC):

  In the PLP feature extraction technique, the frequency axis is first warped to the Bark frequency scale. Trapezoidal shaped filters (motivated by the fact that they approximate the

power spectrum of the critical band masking curve from Fletcher (Fletcher, 1995)) equally spaced on the Bark frequency scale are applied. In PLP, pre-emphasis is performed in the frequency domain using a scaling function, which is based on the equal-loudness curve. The output of the filter bank is compressed using cubic root function, which is motivated by the power law relationship between the intensity and amplitude. The cepstral coefficients are estimated from the modified auditory spectrum by following the same steps as in LPCC.

- Multi-resolution relative spectra features (MRASTA):
  In MRASTA feature extraction, the log-energies in the Bark critical bands are used as features, but the important aspect is the processing of the trajectories in time. This is discussed in the following section.

Processing along Time

It is well known that important characteristics of speech sounds are also present in its dynamics (Furui, 1986a). The simplest and the most common way to capture these dynamics is to append the static cepstral features (LPCC, MFCC or PLPCC) with its first order time derivatives (delta cepstrum) and the second order time derivative (delta-delta cepstrum) (Furui, 1986b). The first order derivative is an estimate of the local slope, and is typically computed by applying an FIR filter with impulse response function given in Figure 2.5 (a) on the static features. The delta-delta features are computed using the impulse response function shown in Figure 2.5 (b).

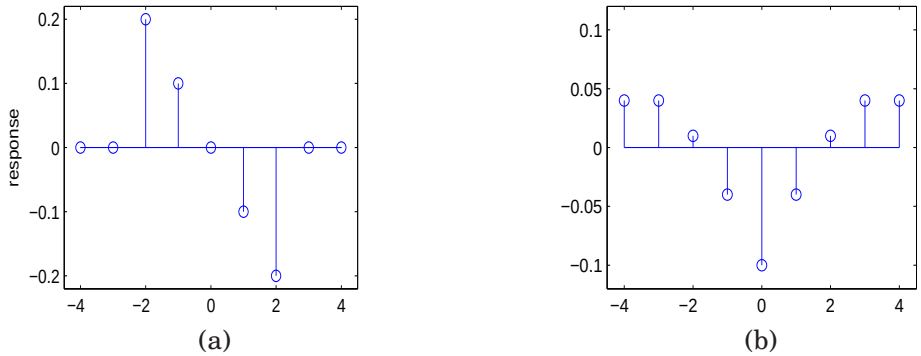


Figure 2.5. (a) A typical impulse response for computing delta features in hidden Markov model toolkit (HTK) (Young *et al.*, 2000). (b) Impulse response function for computing the delta-delta features.

Multi-resolution relative spectra feature (MRASTA) extraction technique is an extension to delta

features and RASTA filtering (Hermansky and Fousek, 2005). Here, the temporal information is integrated by filtering the log-energies in critical bands (auditory spectrum) using a bank of bandpass filters with varying resolutions. In the time-domain, these filters have the functional form of the first or second derivative of a Gaussian function as shown in Figure 2.6. In the frequency domain, the above operation can be viewed as filtering the modulation spectrum of speech. The bandwidth of the filters is controlled by the variance of the Gaussian function.

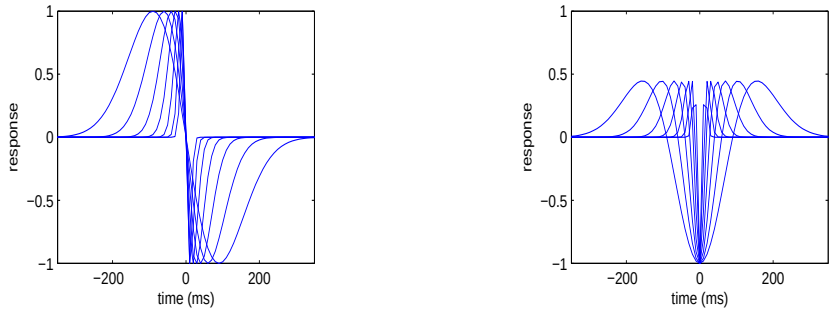


Figure 2.6. (a) Impulse response of MRASTA filters with the shape of first order Gaussian derivative. (b) Impulse response of MRASTA filters with the shape of second order Gaussian derivative. The standard deviation of the Gaussian function are 8, 12, 18, 27, 40, 60, and 90 ms. These filters span 70 frames or 700 ms of temporal context.

Cepstral Mean/Variance Normalization

Cepstral mean and variance normalization is a commonly used post-processing technique of the acoustic features. As shown in Figure 2.3, the channel $h(n)$ (room acoustics or the line characteristics in the case of telephone speech) has a convolutive effect on the speech in the time domain. In the cepstral domain, the effect of the channel is additive. If the characteristics of the channel varies slowly compared to that of the speech, its effect can be compensated by performing mean normalization in the cepstral domain. In addition, cepstral variance normalization has also been shown to provide some robustness against additive noise (Gales and Young, 2008).

In most state-of-the-art systems, cepstral mean/variance normalization is performed on per speaker basis to achieve speaker normalization. In the case of MRASTA feature extraction, the filters have a zero mean. Hence the features are inherently robust to linear distortion.

2.3 Language Modeling

The language model Θ_l estimates the joint probability of a sequence of words denoted by $W_1^K = \{W_1, W_2, \ldots W_K\}$ as

$$P(W_1^K|\Theta_l) = \prod_{k=1}^{K} P(W_k|W_1^{k-1}, \Theta_l) \tag{2.4}$$

In ASR, an N-gram statistical language model (Bahl *et al.*, 1983) is typically used, where it is assumed that the present word W_k is statistically independent of the preceding words W_1^{k-N}. In other words, the N-gram language model can be interpreted as a Markov model of order N-1. The language model probability is then given by

$$P(W_1^K|\Theta_l) = \prod_{k=1}^{K} P(W_k|W_{k-N+1}^{k-1}, \Theta_l) \tag{2.5}$$

Typical values of N are 2 (bigram) and 3 (trigram). The N-gram language model probabilities are estimated from a text corpus related to the recognition task at hand. The maximum likelihood estimate of the trigram probability is given by

$$\hat{P}(w_3|w_1, w_2) = \frac{C\left(w_1, w_2, w_3\right)}{C\left(w_1, w_2\right)} \tag{2.6}$$

where $C\left(w_1, w_2, w_3\right)$ and $C\left(w_1, w_2\right)$ respectively denote the number of times the word sequence $\{w_1, w_2, w_3\}$ and $\{w_1, w_2\}$ occur in the training text corpus. A major problem with the maximum likelihood estimation is data sparsity. Several algorithms have been proposed to assign non-zero probability mass to unseen events. For example, in the classical Katz smoothing technique (Katz, 1987), the trigram probability is estimated as follows.

$$\hat{P}(w_3|w_1, w_2) = \begin{cases} \frac{C(w_1,w_2,w_3)}{C(w_1,w_2)} & \text{if } C\left(w_1, w_2, w_3\right) \geq M \\ d\frac{C(w_1,w_2,w_3)}{C(w_1,w_2)} & \text{if } 1 \leq C\left(w_1, w_2, w_3\right) < M \\ \beta(w_1, w_2)\hat{P}(w_3|w_2) & \text{otherwise} \end{cases} \tag{2.7}$$

In other words, if the count $C\left(w_1, w_2, w_3\right)$ is sufficiently large, then the maximum likelihood

estimate is directly used. If the count is less than a certain constant M, then the maximum likelihood estimate is discounted by a factor d which is computed based on the Good-Turing theory. The total discounted mass is assigned to unseen events using the backoff technique. For example, the probability of the unseen trigram is estimated by backing off to the bigram estimate $\hat{P}(w_3|w_2)$, where $\beta(w_1, w_2)$ is the backoff weight. Detailed description of various language model smoothing techniques can be found in (Jelinek, 2001).

2.4 Acoustic Modeling

Hidden Markov models (HMMs) have been the mainstream of acoustic modeling in almost all practical large vocabulary ASR systems (Baker, 1975; Jelinek, 1976). In this section, we briefly discuss the theory and application of HMM in speech recognition. A detailed treatise on the theory of HMMs can be found in (Rabiner, 1989; Bilmes, 2006). The practical aspects or tricks of the trade of applying HMMs in ASR are discussed in (Gales and Young, 2008).

2.4.1 Hidden Markov Model

Let the likelihood of a sequence of acoustic feature vectors $X_1^T = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ conditioned on a word W and its model Θ_a be denoted by $p(X_1^T|W, \Theta_a)$. In HMMs we assume another discrete valued random variable $S_1^T = \{s_1, s_2, \ldots, s_T\}$, which denotes the hidden state sequence and write the above expression as

$$\begin{aligned} p(X_1^T|W, \Theta_a) &= \sum_{S_1^T} p(X_1^T, S_1^T|W, \Theta_a) \\ &= \sum_{S_1^T} p(X_1^T|S_1^T, W, \Theta_a)\ P(S_1^T|W, \Theta_a) \end{aligned} \tag{2.8}$$

The computation of (2.8) can be simplified greatly by the following assumptions which form the heart of the HMM theory.

- Markov Chain Assumption:

  We assume that the underlying state sequence is a first order Markov chain. In other words, the present state depends only on the immediately preceding state as $P(s_t|s_1, s_2, \ldots s_{t-1}) =$

$P(s_t|s_{t-1})$. As a result, the probability of the state sequence S_1^T in (2.8) can be written as

$$P(S_1^T|W,\Theta_a) = P(s_1)\prod_{t=2}^{T} P(s_t|s_{t-1}) \tag{2.9}$$

- Stationarity Assumption:

  The hidden state sequence is also assumed to be stationary. In other words, the state transition probabilities are independent of time t as

  $$P(s_t = j|s_{t-1} = i) = a_{i,j} \quad \forall \quad t \tag{2.10}$$

  Suppose that the states belong to a finite alphabet of size N, then state transitions are captured by the transition probability matrix of size $N \times N$ with elements $a_{i,j}$. The term $P(s_1)$ in (2.9) is the initial state occupancy probability and it is represented as $\pi_j = P(s_1 = j)$.

- Output Conditional Independence Assumption:

  The observation at time t depends only on the underlying state s_t. As a result, the probability of the observation vector X_1^T, given the corresponding state sequence S_1^T is given by

  $$p(X_1^T|S_1^T,W,\Theta_a) = \prod_{t=1}^{T} p(\mathbf{x}_t|s_t) \tag{2.11}$$

Furthermore, in a given state j, the features are assumed to be stationary and modeled using identical multivariate probability density functions. That is, $p(\mathbf{x}_t|s_t = j) = p_j(\mathbf{x}_t)$. To summarize, the acoustic model for the word W consists of a state transition probability matrix, an initial state occupancy probability vector, and the parameters of the output density function in each of the states. The likelihood of the feature vector sequence conditioned on the word and its acoustic model is obtained by combining (2.9) and (2.11) as

$$p(X_1^T|W,\Theta_a) = \sum_{S_1^T} P(s_1)p(\mathbf{x}_1|s_1)\prod_{t=2}^{T} p(\mathbf{x}_t|s_t)P(s_t|s_{t-1}) \tag{2.12}$$

The direct computation of the above equation grows exponentially with the number of frames T

in the utterance. However, this can be efficiently computed using the forward-backward recursion as described in (Rabiner, 1989). Alternatively, the above equation can be approximated by replacing the summation with a max operator as (2.13). In other words, by finding the state sequence that yields the maximum likelihood. The most likely state sequence can be efficiently computed using the Viterbi algorithm (Viterbi, 1967)

$$p(X_1^T|W,\Theta_a) \approx \max_{S_1^T} \; P(s_1)\; p(\mathbf{x}_1|s_1) \prod_{t=2}^{T} p(\mathbf{x}_t|s_t) P(s_t|s_{t-1}) \tag{2.13}$$

In large vocabulary ASR systems, it is not practical to train word based HMM models, mainly due to the data insufficiency problem. Moreover, it does not provide the flexibility to model new words previously unseen during the training process. To overcome this problem, a word is modeled as a sequence of phonemes and phoneme specific HMM models are trained. The mapping from words to the pronunciation lexicon is obtained from a pronunciation dictionary.

Pronunciation dictionaries map a given word to its corresponding sequence of phonemes. For example, in the TIMIT dictionary, the word “dog” is transcribed as /d/ /ow/ /g/. In practice, there can be variation in pronunciation due to the differences in the dialect or the influence of the native language. This is mitigated to a certain extent by including different plausible pronunciation variants. Alternatively, pronunciation dictionaries can also be obtained by using letter-to-sound rules.

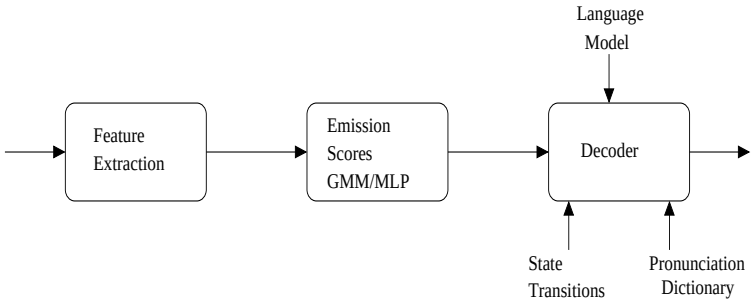


Figure 2.7. Detailed block schematic of an ASR system.

Figure 2.7 shows the detailed block diagram of an HMM based ASR system. The language model provides a grammar with words as units such that the search is biased towards grammatically well formed sentences. Each word in the language model is replaced by its pronunciation lexicon to derive a phonetic search network. Each phoneme is replaced by its corresponding HMM model

to derive a large search network, which is commonly known as a trellis. While decoding, given a sequence of acoustic feature vectors $X_1^T = \{\mathbf{x}_1, \ldots \mathbf{x}_t, \ldots, \mathbf{x}_T\}$, the emission scores $p_j(\mathbf{x}_t)$ in each of the HMM states j are first estimated. The Viterbi algorithm is subsequently applied on the trellis to decode the maximum likelihood state sequence, and thereby the best word sequence.

2.4.2 State Emission Modeling

The link between the acoustic observation and the trellis (prior model) is provided by the state emission modeling. The emission score in the state j of an HMM is denoted by $p_j(\mathbf{x}_t)$. In the following sections, we discuss various approaches for estimating the state emission scores.

Discrete Modeling

In this case, the acoustic feature vector sequence $X_1^T = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_t, \ldots \mathbf{x}_T\}$ is first quantized to discrete observation symbols $O_1^T = \{o_1, o_2, \ldots o_t, \ldots o_T\}$, where the symbols belong to a finite alphabet. This discretization is performed in an unsupervised fashion using standard vector quantization techniques such as k-means clustering. The emission probability of the observation symbol o_t in state j of the HMM is modeled as a discrete distribution $P_j(o_t)$.

Continuous Modeling

In discrete HMMs, there is a loss of information due to vector quantization. Alternatively, the acoustic feature vector in a state can be modeled as a continuous random variable using a continuous distribution function such as a Gaussian mixture model (GMM). The likelihood of the feature vector $\mathbf{x}_t$ in state j of the HMM is given by

$$p_j(\mathbf{x}_t) = \sum_{k=1}^{M} c_{j,k}\, \mathcal{N}(\mathbf{x}_t;\ \boldsymbol{\mu}_{j,k},\ \boldsymbol{\Sigma}_{j,k}) \tag{2.14}$$

where $c_{j,k}$ denotes the weight of the mixture component k in state j, and $\mathcal{N}(\mathbf{x}_t;\ \boldsymbol{\mu}_{j,k},\ \boldsymbol{\Sigma}_{j,k})$ denotes the normal distribution with a mean vector $\boldsymbol{\mu}_{j,k}$ and a covariance matrix $\boldsymbol{\Sigma}_{j,k}$ given by

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{j,k}, \boldsymbol{\Sigma}_{j,k}) = \frac{1}{(2\pi)^{\frac{D}{2}}\ |\boldsymbol{\Sigma}_{j,k}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{j,k})'\ \boldsymbol{\Sigma}_{j,k}{}^{-1}\ (\mathbf{x} - \boldsymbol{\mu}_{j,k})\right] \tag{2.15}$$

Semicontinuous Modeling

In continuous density modeling, there is a mean vector and a covariance matrix associated with each of the mixture component in a state. This will result in a large number of free parameters, which can lead to data insufficiency while training. In semicontinuous density modeling, the means and covariance matrices are shared among all the phonetic classes. This is achieved by clustering the data into a designated number of clusters M. The state emission likelihood in state j is then given by

$$p_j(\mathbf{x}_t) = \sum_{k=1}^{M} c_{j,k}\, \mathcal{N}(\mathbf{x}_t; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.16}$$

where $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ denotes the normal distribution representing the cluster k with a mean vector $\boldsymbol{\mu}_k$ and a covariance matrix of $\boldsymbol{\Sigma}_\mathbf{k}$. This approach can be seen as a hybrid between discrete and continuous modeling. It can seen that the acoustic features are clustered into M components (discrete modeling), and the likelihood for a phoneme is the linear combination of the likelihoods given each cluster (continuous modeling).

2.4.3 Training Criteria

In the discussion so far, we assumed a trained acoustic model Θ_a. In HMM based acoustic modeling, the acoustic model consists of an HMM for each of the basic modeling unit, *e.g.,* a phoneme, and its parameter set includes a state transition probability matrix and in the case of mixture of Gaussians emission distribution, a mean vector and a covariance matrix for each of the mixture component in a state. The parameters of the acoustic model are estimated to optimize certain training criterion on the training data. The most commonly used training criterion is maximum likelihood estimation which is discussed below.

Maximum Likelihood (ML) Estimation

Suppose that the training set contains R utterances. The sequence of acoustic feature vectors for utterance r is denoted by X_r and the corresponding sequence of words is denoted by W_r. If Θ (the subscript in Θ_a is dropped for notational simplicity) denotes the acoustic model, then the maximum

likelihood training criterion is given by

$$\mathcal{F}_{ML}(\Theta) = \sum_{r=1}^{R} \log p(X_r|W_r, \Theta) \tag{2.17}$$

The ML estimate of the acoustic model is given by $\Theta_{ML} = \arg\max_{\Theta} \mathcal{F}_{ML}(\Theta)$. The training can be efficiently performed using the Baum-Welch reestimation algorithm (Baum *et al.*, 1970), which is based on the principle of expectation maximization (Dempster *et al.*, 1977).

The ML training ensures that the likelihood of the observed data conditioned on the corresponding word sequence is maximized. This approach does not ensure the discrimination among the models. To this end, discriminative training criteria such as maximum mutual information (MMI) estimation and minimum Bayes' risk (MBR) have been found to be useful in ASR. We discuss them in the following subsections.

Maximum Mutual Information (MMI) Estimation

MMI estimation was one of the first discriminative training criteria to be investigated in ASR (Nadas, 1983; Bahl *et al.*, 1986). Here, the parameters of the acoustic model are optimized to maximize the mutual information between the acoustic feature vectors and the recognized word sequence. If X_r and W_r denote the sequence of feature vectors and words respectively for the utterance r, then the empirical MMI training criterion (Brown, 1987) is given by

$$\mathcal{F}_{MMI}(\Theta) = \frac{1}{R} \sum_{r=1}^{R} \log \frac{p(X_r, W_r|\Theta)}{P(W_r)p(X_r|\Theta)} \tag{2.18}$$

As the language model is independent of the acoustic model parameters parameters Θ, the training criterion can be equivalently written as

$$\begin{aligned} \mathcal{F}_{MMI}(\Theta) =& \frac{1}{R} \sum_{r=1}^{R} \log \frac{p(X_r|W_r, \Theta)P(W_r)}{\sum_W p(X_r|W, \Theta)P(W)} \\ =& \frac{1}{R} \sum_{r=1}^{R} \log P(W_r|X_r, \Theta) \end{aligned} \tag{2.19}$$

The MMI training is discriminative because the model parameters are optimized such that the likelihood of the correct model is maximized and the likelihood of the competing models is minimized.

It can be seen from (2.19) that the MMI training is equivalent to maximizing the posterior probability of the word sequence given the data. The training is performed using extended Baum-Welch algorithm or gradient descent based approaches.

It has been shown that when the prior distribution (*i.e.,* the language model) and the assumption on the parametric form of the distribution are correct, both ML and MMI are consistent estimators with the former yielding lower variance (Nadas, 1983). However, when the above assumptions are not valid, MMI training has been shown to yield lower word error rates.

Minimum Bayes' Error (MBR) Estimation

In MBR training (Kaiser *et al.*, 2002), the parameters of the acoustic model are optimized to minimize the expected loss during recognition $\mathcal{L}(W_r, W)$ between the recognized sequence of words W and the reference word sequence W_r as

$$\begin{aligned}\mathcal{F}_{MBR}(\Theta) &= \sum_{r=1}^{R} \frac{\sum_W p(X_r|W,\Theta)P(W)\ \mathcal{L}(W_r, W)}{\sum_W p(X_r|W,\Theta)P(W)} \\ &= \sum_{r=1}^{R} \sum_W P(W|X_r,\Theta)\ \mathcal{L}(W_r, W) \end{aligned} \tag{2.20}$$

The loss function $\mathcal{L}(W_r, W)$ can be the Levenshtein edit distance between the word sequences W_r and W. Alternatively, the distance can be computed between the phonetic sequences corresponding to the word sequences, which forms the basis of minimum phone error training (Povey and Woodland, 2002).

Artificial neural networks (ANN) form a prominent class of discriminatively trained models, which have been shown to be useful in ASR. We discuss this in the following section.

2.5 Artificial Neural Networks

An artificial neural network (ANN) is a mathematical model that attempts to emulate the structure and functionality of the biological neural network. It is a dense interconnection of simpler computational elements known as *artificial neurons*. Artificial neural networks are also known as connectionist models.

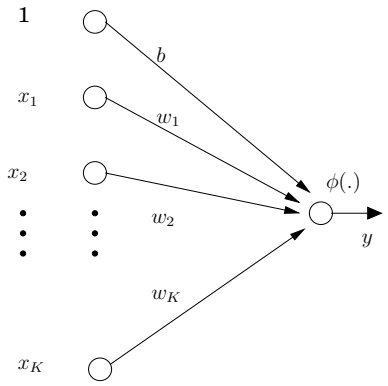


Figure 2.8. Block schematic of an artificial neuron.

Figure 2.8 shows the block schematic of an artificial neuron. Mathematically, it can be viewed as a multi-input single-output function. Suppose that $x_1, x_2, \ldots, x_K$ denote the input to the neuron, then the output of the neuron y is given by

$$y = \phi\left(b + \sum_{k=1}^{K} w_k x_k\right) \tag{2.21}$$

where $w_1, w_2, \ldots w_K$ denote the weights associated with the input $x_1, x_2, \ldots x_K$ respectively and b denotes the bias or the threshold value. The function $\phi(.)$ is sigmoidal or an "S" shaped nonlinear activation function. Typical examples include sigmoid and hyperbolic tangent functions.

Neural networks can find application in function approximation (regression) as well as pattern classification (recognition). They exhibit two important properties. Firstly, in the case of regression, it has been shown that a feedforward neural network with at least one hidden layer can approximate any continuous function to the desired level of accuracy (Hornik *et al.*, 1989). In the case of classification, it has been shown that the neural network classifiers with sufficient capacity and trained on sufficient amount of data can estimate the posterior probabilities of the output classes conditioned on the input features, provided the samples are drawn with the correct prior distribution (Richard and Lippmann, 1991). This coupled with efficient training algorithms have made artificial neural networks a popular choice in various machine learning applications.

The multilayer perceptron (MLP) represents the prominent and well researched class of artificial neural networks. It is a feedforward neural network where information flows in one direction only. The other classes of artificial neural networks include recurrent neural networks (with feed-

back mechanism), Kohonen self organizing maps, etc. (Haykin, 1998) provides a comprehensive description of artificial neural networks. The tricks of the trade of using neural networks are discussed in (LeCun *et al.*, 1998).

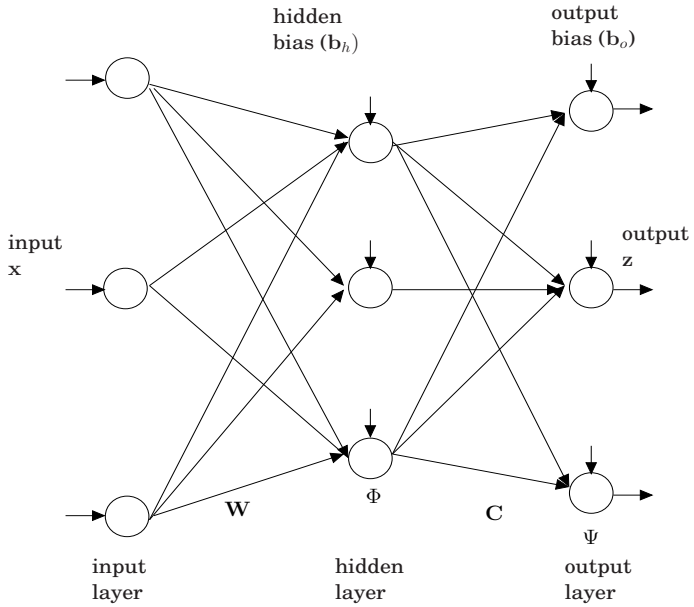


Figure 2.9. Multilayer perceptron with one hidden layer.

Figure 2.9 is a block schematic of an MLP with one hidden layer, where $\mathbf{W}$ denotes the weight matrix connecting the input layer to the hidden layer, $\mathbf{C}$ denotes the weight matrix connecting the hidden layer to the output layer, $\mathbf{b_h}$ and $\mathbf{b_o}$ denote the bias vectors at the hidden and output layers respectively, and $\Phi(.)$ and $\Psi(.)$ denote the vector valued activation functions at the hidden and output layer respectively. If $\mathbf{x}$ denotes the input to the MLP, then its output $\mathbf{z}$ is given by

$$\mathbf{z} = \Psi\left(\mathbf{b}_o + \mathbf{C}\Phi\left(\mathbf{b}_h + \mathbf{W}\mathbf{x}\right)\right) \tag{2.22}$$

The hidden activation function is typically sigmoid or hyperbolic tangent. In the case of regression, the output nonlinearity is dropped. In classification, the output nonlinear function is typically softmax, which can be interpreted as multi-dimensional extension of the sigmoid function (Bridle, 1990). As discussed previously, in a probabilistic interpretation, $\mathbf{z}$ represents a vector of the posterior probabilities of the output classes (*e.g.,* phonemes) conditioned on the input features $\mathbf{x}$.

2.5.1 MLP Based Acoustic Modeling

An area where MLPs have enjoyed considerable success is automatic speech recognition, particularly in the acoustic modeling of speech (Bourlard and Morgan, 1994; Zhu *et al.*, 2004; Morgan *et al.*, 2005; Stolcke *et al.*, 2006; Fousek *et al.*, 2008; Park *et al.*, 2009). Here, the MLP is typically trained using standard acoustic features such as MFCC or PLP coefficients. The output classes of the MLP represent the subword units of speech such as phonemes. The trained MLP estimates the posterior probabilities of the phonemes for every 10 ms of speech, which are subsequently used in ASR. In practice it is observed that taking an explicit temporal context of around 90 ms on these features is useful.

Training

Suppose that the MLP has K output phonetic classes. The output of the MLP is an estimate of the posterior probability of the output classes $q_k,\ k = 1, 2, \ldots K$ conditioned on the acoustic feature vector $\mathbf{x}$ and the MLP model Θ as $\hat{P}(q_k|\mathbf{x}, \Theta)$. The distribution of the underlying correct phonetic classes or the ground truth is denoted as $P(q_k|\mathbf{x})$ and is obtained by hand labeling or estimated by forced alignment. The ground truth labels are typically in the hard-target or one-hot format, where only one phonetic class is active at a given time instant. In classification problems, the MLP is usually trained using minimum cross-entropy error criterion, which can be expressed as

$$\begin{aligned}\mathcal{F}(\Theta) &= -\,E_X\left[\sum_k P(q_k|\mathbf{x}) \log \hat{P}(q_k|\mathbf{x}, \Theta)\right] \\ &= -\int_{\mathbf{x}} p_X(\mathbf{x}) \left[\sum_k P(q_k|\mathbf{x}) \log \hat{P}(q_k|\mathbf{x}, \Theta)\right] d\mathbf{x}\end{aligned} \tag{2.23}$$

Now, suppose that the training data $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ are drawn according to the distribution $p_X(\mathbf{x})$, then the empirical cross-entropy error criterion can be written as

$$\mathcal{F}(\Theta) \;\approx\; -\frac{1}{N}\left[\sum_n \sum_k P(q_k|\mathbf{x}_n) \log \hat{P}(q_k|\mathbf{x}_n, \Theta)\right] \tag{2.24}$$

The model parameters are optimized to minimize the above error criterion as $\Theta_{mlp} = \arg\min_\Theta \mathcal{F}(\Theta)$. Gradient descent approach is used for the optimization, and this can be efficiently implemented

using the back-propagation algorithm (Bishop, 1995). In the following sections, we discuss some of the information theoretic interpretations of minimum cross-entropy training:

- Relationship to Kullback-Leibler divergence: The cross-entropy training criterion (2.23) can be rearranged as

$$\begin{aligned}\mathcal{F}(\Theta) &= -\int_{\mathbf{x}} p(\mathbf{x}) \sum_{k=1}^{K} P(q_k|\mathbf{x}) \log P(q_k|\mathbf{x}) d\mathbf{x} + \int_{\mathbf{x}} p(\mathbf{x}) \sum_{k=1}^{K} P(q_k|\mathbf{x}) \log \frac{P(q_k|\mathbf{x})}{\hat{P}(q_k|\mathbf{x})} d\mathbf{x} \\ &= H(Q|X) + \mathcal{KL}_X(P||\hat{P}) \end{aligned} \tag{2.25}$$

  where $H(Q|X)$ is the conditional entropy of the phonetic labels given the acoustic features assuming the true distribution $P(q|x)$, and $\mathcal{KL}_X(P||\hat{P})$ denotes the average KL divergence between the reference distribution and the estimated distribution on the entire training data set. Hence minimizing the cross-entropy is same as minimizing the KL divergence between the reference and estimated distributions. For hard-target labeling strategy, the term $H(Q|X) = 0$, and cross-entropy is same as the KL divergence.

- Relationship to mutual information: The cross-entropy training criterion (2.23) can be rearranged using Bayes' rule as

$$\mathcal{F}(\Theta) = -\int_{\mathbf{x}} \sum_{k=0}^{K} p(\mathbf{x}, q_k) \log \frac{\hat{p}(\mathbf{x}, q_k)}{\hat{p}(\mathbf{x})\hat{P}(q_k)} d\mathbf{x} - \sum_{k}^{K} P(q_k) \log \hat{P}(q_k) \tag{2.26}$$

  It can be seen from the above equation that minimum cross-entropy training is equivalent to maximum mutual information training as the first term in the right hand side of the above equation is the expression for mutual information (Bridle, 1990). However, it should be noted that the pointwise mutual information $\log \frac{\hat{p}(\mathbf{x},q_k)}{\hat{p}(\mathbf{x})\hat{P}(q_k)}$ between the features and the labels is obtained from the probability distribution estimated by the MLP, but the expectation is with respect to the true distribution $p(\mathbf{x}, q)$. As a consequence, maximizing above form of mutual information is also equivalent to maximizing the classification accuracy.

- To obtain further insights, we write the expression for mutual information, where the expec-

tation is with respect to the joint distribution $\hat{p}(\mathbf{x}, q)$ estimated by the MLP as

$$\begin{aligned}\mathcal{MI}(Q, X) &= \int_{\mathbf{x}} \sum_{k=0}^{K} \hat{p}(\mathbf{x}, q_k) \log \frac{\hat{p}(\mathbf{x}, q_k)}{\hat{p}(\mathbf{x})\hat{P}(q_k)} d\mathbf{x} \\ &= -\sum_{k=0}^{K} \hat{P}(q_k) \log \hat{P}(q_k) + \int_{\mathbf{x}} p(\mathbf{x}) \sum_{k=0}^{K} \hat{P}(q_k|\mathbf{x}) \log \hat{P}(q_k|\mathbf{x}) d\mathbf{x} \\ &= \hat{H}(Q) - \hat{H}(Q|X) \end{aligned} \tag{2.27}$$

The term $\hat{H}(Q)$ is the entropy of the prior distribution of the labels as estimated by the MLP, and this can be estimated as the entropy of the average of the output of the MLP. [4] The term $\hat{H}(Q|X)$ is the conditional entropy of the phonetic symbols after observing the features, and this can be estimated as the average of the entropy of the output of the MLP as

$$\hat{H}(Q|X) \approx \frac{1}{N} \sum_{n=1}^{N} H(Q|\mathbf{x}_n) \quad \mathbf{x}_n \text{ drawn from from the distribution } p(\mathbf{x}) \tag{2.28}$$

The above expression for mutual information can be misleading as a higher mutual information does not always guarantee a higher classification accuracy. As an extreme example, consider a badly trained MLP which always yields erroneous decisions with high confidence (or zero entropy).

Relation to semicontinuous modeling

The architecture of a three layered MLP can be likened to semicontinuous density modeling discussed in Section 2.4.2 if the bias and activation function at the output layer is excluded. To illustrate this, we consider a semicontinuous Gaussian density model with M mixtures and diagonal covariance matrices. Let $\mu_{i,k}$ and $\sigma_{i,k}$ denote the mean and standard deviation of the feature component x_k in the mixture i. If $c_{j,i}$ denote the weight associated with mixture component i of the output class j, the likelihood of the feature vector $\mathbf{x} = [x_1, \ldots, x_k, \ldots x_K]'$ for the class j can be written from

[4] The average of the posterior probabilities estimated by the MLP on the training data is an estimate of the prior. Refer Appendix A.3 for the proof.

(2.16) as

$$p_j(\mathbf{x}) = \sum_{i=1}^{M} c_{j,i}\ \phi\left(b_i + \sum_{k=1}^{K}\left(\frac{x_k - \mu_{i,k}}{\sigma_{i,k}}\right)^2\right) \tag{2.29}$$

$$\text{where} \quad b_i = K\log(2\pi) + \sum_{k=1}^{K} 2\log\left(\sigma_{i,k}\right) \quad \text{and} \quad \phi(s) = \exp(-s/2) \tag{2.30}$$

To compare this, consider a three layer MLP with architecture $K \times M \times N$, where $w_{i,k}$ denotes the weight connecting the input node k to the hidden node i and $c_{j,i}$ denotes the weight connecting the hidden node i to the output node j. If μ_k and σ_k denotes the mean and standard deviation of the input feature component x_k and $\phi(.)$ denotes the sigmoid activation function at the hidden layer, the linear activation (without bias) at the output node j is given by

$$y_j = \sum_{i=1}^{M} c_{j,i}\ \phi\left(b_i + \sum_{k=1}^{K} w_{i,k}\ \frac{x_k - \mu_k}{\sigma_k}\right) \tag{2.31}$$

where b_i denotes the bias at the hidden node i and $\phi(s) = \frac{1}{1+\exp(-s)}$. It can be seen that in both the cases, the input is first projected to a hidden representation which is shared among all phonemes. In semicontinuous modeling, this mapping is captured by the mean and standard deviations of the mixture components. In the MLP, this mapping is learned by the input-to-hidden wights and feature normalization is not mixture-specific in this case. The hidden-to-output weights capture the mixture weights associated with the mixture components. The notable difference between the two modeling techniques is in the nonlinear kernel $\phi(.)$, which is exponential in the case of semicontinuous modeling and sigmoidal in the case of MLP. In addition, there can be a difference in the training criterion. The MLP is trained using the minimum cross-entropy criterion which is equivalent to maximum mutual information (MMI) training as discussed in The semicontinuous GMM model can be trained either using maximum likelihood criterion or discriminative training methods such MMI training.

Advantages of MLP based acoustic modeling

The MLP based acoustic modeling offers the following benefits. Some of these are also shared with other modeling techniques such as recurrent neural networks, support vector machines, Gaussian

mixture modeling etc.

- Provides a discriminative acoustic model.
- It can model feature vectors with a large dimensionality. As a result, temporal information in the features can be explicitly learnt by taking a temporal context on the features.
- It obviates the need for strong assumptions on the parametric form of the probability distribution function or the statistics of the input features.
- It obviates the need for statistical independence assumptions between feature streams in the case of feature combination. Consequently, feature combination can be achieved by simple concatenation (or early integration).[5]
- The output of the MLP are probabilities with useful properties such as positivity and summing up to one. As a result, multistream combination can be effectively achieved at the output of the MLP using late integration methods. (Kittler *et al.*, 1998).
- If the MLP is trained with a large amount of data from a diverse population of speakers and in different of environmental conditions, it has been shown to achieve invariance to speaker (Zhu *et al.*, 2004) and environmental specific information (Ikbal, 2004).
- Can be efficiently trained and is scalable with large amount of data. Using efficient toolkits such as Quicknet, it is possible to train MLPs on at least a few thousand hours of speech.

So far we discussed that the MLP estimates the posterior probabilities of phonemes conditioned on the acoustic features for every 10 ms of speech. In the following sections, we discuss the use of the estimated class-conditional probabilities in ASR.

2.5.2 Hybrid System

In the hybrid HMM/MLP system (Bourlard and Morgan, 1994), the MLP is used as a (scaled) likelihood estimator in place of the conventional GMM model. The state emission likelihood is obtained from the associated output of the MLP by normalizing the posterior probability by the respective

[5]If x_1 and x_2 denotes the feature vectors, the scaled likelihoods are obtained by $\frac{p(\mathbf{x}_1,\mathbf{x}_2|q_t=i,\Theta)}{p(\mathbf{x}_1,\mathbf{x}_2)} = \frac{P(q_t=i|\mathbf{x}_1,\mathbf{x}_2,\Theta)}{P(q_t=i)}$. In the case of parametric density modeling, the feature streams are typically assumed to be independent as $p(\mathbf{x}_1,\mathbf{x}_2|q_t=i,\Theta) \approx p(\mathbf{x}_1|q_t=i,\Theta)p(\mathbf{x}_2|q_t=i,\Theta)$

prior probability. For example, if $\mathbf{x}_t$ denotes the acoustic feature vector at time t, the state emission score in state k is the scaled likelihood given by

$$\frac{p(\mathbf{x}_t|q_k,\Theta_{mlp})}{p(\mathbf{x}_t)} = \frac{P(q_k|\mathbf{x}_t,\Theta_{mlp})}{P(q_k)} \tag{2.32}$$

The numerator of the right hand side term in (2.32) is obtained at the output of the MLP. The denominator denotes the prior probabilities of phonemes and it is estimated from the relative frequency of the phonemes in the label set. The output classes of the MLP could also represent the sub-phonemic states, for example, three output classes (states) per phoneme. This finer modeling of the output classes has been shown to yield higher accuracy in recognition of words (Fontaine *et al.*, 1996) as well as phonemes (Schwarz *et al.*, 2006; Pinto *et al.*, 2008).

A natural extension to this is context dependent modeling. In the case of conventional HMM/GMM modeling, it is well known that context dependent modeling yields significantly better performance as the probability density functions that are sharper and less overlapping than their context independent counterparts. In addition context dependent modeling helps in the decoding process by exploiting the sequence information. A major drawback of the hybrid system is that it cannot be easily extended to context dependent modeling as the output classes of the MLP can grow enormously. There have been attempts to indirectly estimate the posterior probabilities of context dependent phonemes as in (Bourlard *et al.*, 1992; Franco *et al.*, 1994; Fritsch *et al.*, 1997). Alternatively, the posterior probabilities of context independent phonemes can be used as features to a standard HMM/GMM based ASR system and this forms the basis of Tandem system, which is discussed in the following section.

2.5.3 Tandem System

The basic idea of the Tandem approach (Hermansky *et al.*, 2000) is to use the class-conditional probabilities estimated by the MLP as features to a standard HMM system in the same way as acoustic features such as MFCC. In this way, the application of the state-of-the-art HMM/GMM modeling techniques such as context dependent modeling, state tying, speaker adaptation, discriminative training, etc becomes straightforward.

The posterior probabilities of phonemes estimated by the MLP have a multinomial distribution,

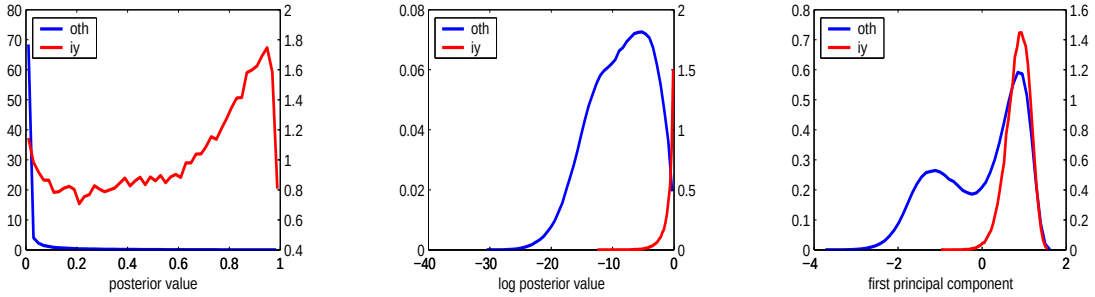


Figure 2.10. The probability density function of (a) posterior features (b) log posterior features and (c) maximum variance direction after Karhunen-Loeve transformation on the TIMIT database.

and hence they cannot be directly used as features to the HMM/GMM system. To address this problem, posterior features are first whitened by applying the logarithm and then decorrelated by using Karhunen-Loeve transformation (KLT).

Figure 2.10 (a) shows the distribution of the output of the MLP corresponding to the phoneme /iy/ (*e.g.,* f**ee**l) in two cases (i) when the underlying phoneme is /iy/ and (ii) all other phonemes in speech, denoted by the symbol /oth/. Figure 2.10 (b) shows the corresponding distribution of the log posterior probability values and Figure 2.10 (c) shows the distribution of the output of the KLT transformation with the maximum variance. This transformation allows us to model the posterior features using a GMM.

Taking a logarithm of the output of the MLP with a softmax output nonlinearity is equivalent to taking the linear activation values, except for a constant additive factor. [6] It has been shown that using the linear activation values and applying KLT is more effective than the standard Tandem approach (Hermansky *et al.*, 2000).

In the Tandem approach, the discriminatively trained MLP can be viewed as a nonlinear feature transformation, which retains (depending on the classification accuracy) the underlying linguistic information, while suppressing nonlinguistic variabilities such as speaker information. As a result, the posterior features are treated just like standard acoustic features, and state-of-the-art HMM/GMM modeling can be directly applied.

In a recent work (Aradilla, 2008), the output of the MLP were used directly as features (*i.e.,* without transformation), but the state emission distribution was appropriately assumed to be multino-

[6] If $\mathbf{y} = [y_1, y_2, \ldots y_i, \ldots y_N]$ denotes the linear activation vector at the output of the MLP, and $\mathbf{z} = [z_1, z_2, \ldots z_i, \ldots z_N]$ denotes the output after applying the softmax activation function. The log-posteriors can be written as $\log(z_i) = y_i - \log\left(\sum_{j=1}^{N} \exp(y_j)\right)$.

mial, and its parameters were learned. The parameters of the system (state transition matrix and the multinomial emission distribution) are estimated to minimize the Kullback-Leibler divergence between the state distribution and the distribution estimated by the MLP.

2.5.4 Scope for Improvement and Context of this Thesis

The MLP estimates the posterior probabilities of the phonemes conditioned on the input features x and the model Θ_{mlp} as $P(q = i|\mathbf{x}, \Theta_{mlp})$. There are three possible ways in which the MLP based acoustic modeling can be improved to obtain better ASR performance.

- Richer features x : This includes designing feature extraction techniques that can better model the spectral and temporal information in speech, and this is a dominant direction of research in the entire ASR community. The MLP provides the advantage in this aspect as it can effectively model features with very large dimensionality.

- Better modeling Θ_{mlp} : Better modeling can be achieved in two ways. Firstly by increasing the size of the training data to achieve better generalization. Secondly, increasing the capacity of the model by increasing the number of layers or the size of the hidden layer. However, this approach is often limited by the amount of training data available.

- Finer output classes q_i : The output classes of the MLP could represent finer classes such as sub-phonemic states.

The basic premise of this research is that there exists useful contextual information in the sequence of posterior features estimated by the MLP, and in this sparse feature space, contextual information spanning longer temporal contexts can be effectively modeled. To this end, we train a second classifier on the posterior features with a longer (than 90 ms which is typically applied on acoustic features) temporal context. With respect to the second classifier in this hierarchical architecture, this strategy can be seen as using richer features.

Chapter 3

Analysis of MLP Classifiers using Volterra Series

3.1 Introduction

Multilayer perceptron (MLP) neural network is being applied in a variety of real-world applications such as speech recognition, computer vision, bioinformatics, and computational finance, among many other fields. One area where MLPs have enjoyed considerable success is automatic speech recognition (ASR), particularly in acoustic modeling of speech, where it is typically used to estimate the posterior probabilities of phonemes conditioned on the acoustic features.

MLP based acoustic modeling has been shown to improve the performance of large vocabulary ASR systems. It has been further investigated for recognition of speech in languages such as Arabic (Park *et al.*, 2009) and Mandarin (Hwang *et al.*, 2007). The successes in practical ASR systems have spawned new research directions such as semi-supervised learning (Malkin *et al.*, 2009). One area which has received little or no attention is the analysis of the trained MLP classifiers to understand the properties of speech such as spectro-temporal patterns that are actually learned by the trained classifier.

The goodness of the trained MLP is typically evaluated by first estimating the phonetic class-conditional probabilities on the cross-validation or the test data set, and then using the ground

truth to compute one or more of the following measures: (a) frame-level phoneme classification error (b) cross-entropy between the estimated posterior probabilities and the phonetic labels, or (c) the final word error rate on the task. The above measures can indicate how well the MLP is trained in terms of the classification accuracy or its generalizing ability. In addition, the analysis of the phonetic confusion matrix can indicate the error patterns in phoneme classification. However, these measures do not reveal any information about the patterns in the input feature space such as spectro-temporal patterns that are learned by the trained system for each of the phonemes.

To analyze the functionality of an MLP, one has to interpret its trained parameters namely the weights and biases. In this chapter, we formulate a generic mathematical framework to apply the Volterra theory of nonlinear systems (Volterra, 1930; Boyd *et al.*, 1984) to interpret the functionality of MLP classifiers, which are trained to estimate the phonetic class-conditional probabilities.

The specific contributions of this work include: (a) development of a mathematical framework to apply Volterra series to a nonlinear dynamic system consisting of a cascade of a finite impulse response (FIR) filter bank and a three-layered MLP classifier (b) calculation of the Volterra kernels of the system in terms of the trained model parameters (c) modifications to the Volterra kernels when the features to the MLP are normalized to zero-mean and unit-variance, (d) addressing the scenario where a linear transformation matrix precedes the FIR filter bank, and (e) demonstration of the applicability of the proposed framework to analyze MLP classifiers which are trained using mel filter bank energies, mel frequency cepstral coefficients with delta and delta-delta parameters, and multi-resolution relative spectra features. Furthermore, we also compare the Volterra kernels obtained using an alternative analysis technique known as Weiner series.

3.2 Background

The MLP classifier can be analyzed in three broad ways by (a) deriving symbolic rules from the trained parameters (b) representing the input-output function as a power series and analyzing each term in the series and (c) presenting stimuli such as white noise at the input of the MLP and studying the correlation between the stimuli and the response. We discuss each of the analysis techniques in the following section.

3.2.1 Rule based analysis

There have been works in the literature where artificial neural networks such as MLPs have been analyzed by extracting symbolic representations or rules from the trained parameters (LiMin, 1994; Setiono and Liu, 1996; Benitez *et al.*, 1997; Setiono *et al.*, 2002). The rules are typically of the form *"If a set of positive antecedents are true and a set of negative antecedents are false, then the conclusion (or negated conclusion) holds."* For example, in the case of Fisher Iris data, where the features are simple measurements such as length and width of petals and sepals, a rule derived from the MLP could be of the form "*If (petal length* < 2.0 *cm* and *petal width* < 0.6 *cm), then the species is Iris setosa.*" In this section, we briefly discuss some of the rule based analysis techniques.

In the *knowledgetorn* approach (LiMin, 1994), each node in the output or hidden layer is represented as a concept. For each concept node, a set of positive and negative attributes (different from antecedents) are identified. A set of positive (or negative) attributes of a concept consists of nodes in the preceding layer with positive (or negative) weights connecting the concept node. The rules for a concept node in the output layer are identified by exploring all possible combinations of positive and negative attributes in a tree structured fashion, starting from the concept node in the output layer to the input layer. As exhaustive search for the combinations would grow exponentially with the number of layers in the network, the algorithm employs a set of heuristics to reduce the search space.

In the *NeuroRule* approach (Setiono and Liu, 1996), a trained three-layered neural network is analyzed in three stages. In the first stage, the redundant connections in the network are pruned depending on whether the absolute value of the weights is close to zero or not. After pruning, the neural network retains only the salient connections. In the second stage, the continuous-valued activation function in the hidden layer is quantized into a piece-wise constant function in its operating region. This is achieved by clustering the activation values on the function using the training data. In the third stage, a set of rules are derived which represents the relationship between the input and hidden layer, and another set of rules are derived for expressing the relationship between the hidden and the output layer. The two sets are appropriately merged to derive the complete set of rules. This work has been further extended in (Setiono *et al.*, 2002), where the activation function is approximated as a piece-wise (three-piece and five-piece) linear function.

A three-layered MLP can also be analyzed using fuzzy rules. For example, in a work by (Benitez

et al., 1997), fuzzy rules are derived from the network by applying the principle of f-duality on the hidden activation functions. The principle of f-duality states that if $f : X \mapsto Y$ is a one-to-one function, and $\oplus$ is an operator defined in its domain X, then there exists one and only operation $\otimes$ in its range Y such that $f(x \oplus y) = f(x) \otimes f(y)$.

For the sigmoid function $f(x) = \frac{1}{1+e^{-x}}$, the f-dual of addition operator ($+$) is given by $a \otimes b = \frac{ab}{(1-a)(1-b)+ab}$. In fuzzy logic literature, the operator $\otimes$ is known as interactive-or (*i-or*), and it is a hybrid between t-norm and the t-conorm (Benitez *et al.*, 1997). Let $x_1, x_2, \ldots$ denote the activation values at the input layer of the MLP, and $w_1, w_2, \ldots$ denote the weights connecting the input layer to a particular hidden node, then the activation value at the output of the hidden node can be decomposed using the *i-or* operator as

$$f\left(\sum_i w_i x_i\right) = \bigotimes\nolimits_i f(w_i x_i)$$

If the connecting weight is positive, the positive antecedent condition could be $f(w_i x_i) > 0.9$ or equivalently $x_i > 2.2/|w_i|$. If the connecting weight is negative, the negated antecedent could be $f(w_i x_i) < 0.1$ or equivalently $x_i < -2.2/|w_i|$. The individual antecedents (the above inequalities), connected using the interactive-or operator forms the final rule. For example, a rule could be "*if sepal-length is greater than 3 cm i-or petal length is not greater than 2 cm, then the species is setosa.*"

Rule based analysis is more effective when the MLP is analyzed as a standalone system, its input features represent simple measurements such as the length or width of the petals, sepals etc, and when the cardinality of the feature vector is small. In the following section, we discuss why rule based analysis of an MLP may not be applicable in the case of speech processing, and motivate the need for alternative analysis techniques.

3.2.2 Motivation

Figure 3.1 shows the block schematic of a typical usage of an MLP classifier for acoustic modeling in ASR, showing feature extraction as well as modeling. In Chapter 2, we discussed commonly used feature extraction techniques and their motivations. Here, we briefly revisit feature extraction and plot the signal representation at different stages of feature extraction for motivating the analysis framework. Figure 3.2 (a) is a spectrogram of the utterance "*artificial intelligence*" in the TIMIT

database. A trained human can accurately identify the underlying sequence of phonemes by carefully studying the spectrogram, even without the knowledge of the language (Zue, 1985). In ASR, however, the spectrogram is not directly used as a feature. Instead, it is processed to derive features which are more robust and suitable for the ensuing pattern classifier. The processing is performed along both frequency as well as time as discussed below:

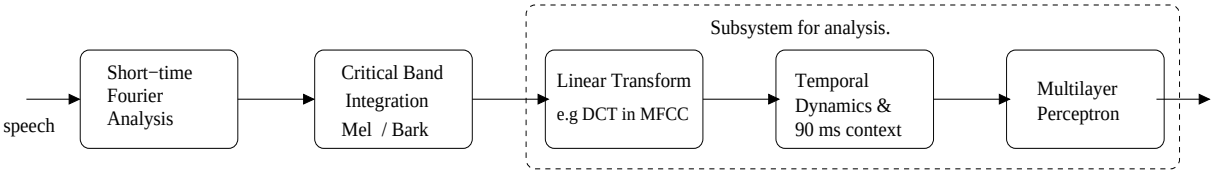


Figure 3.1. Block schematic of the feature extraction and the MLP classifier.

Processing along Frequency

Processing along frequency includes the application of mel or bark critical band filters on the Fourier power spectrum, followed by a nonlinear compression function. For example, in MFCC feature extraction, the frequency is warped to the mel psychoacoustic scale, triangular filters equally spaced in the mel scale are applied, and the output is compressed using the $\log$ function. Figure 3.2 (b) is a plot of the auditory spectrum of speech. It can be seen that it is a smoothed and frequency warped version of the spectrogram.

The $\log$-energies in the auditory filters can be used as acoustic features to the MLP. However, most often cepstral features are derived from the filter bank energies as discussed in the previous chapter. In MFCC feature extraction, cepstral coefficients are obtained by applying discrete cosine transform (DCT) on the auditory spectrum. Figure 3.2 (c) is a plot of the static cepstral coefficients as a function of time. It can be seen that the cepstral patterns are not as intuitive and interpretable as the raw spectrogram or the auditory spectrogram, but they have been shown to be more robust in ASR.

Processing along Time

The processing along time typically includes the computation of delta and delta-delta cepstral parameters. The dynamic cepstral coefficients are typically computed using FIR filters shown in Figure 2.5. In addition, the static and dynamic features are applied at the input of the MLP with

a temporal context of 9 frames. In the case of multi-resolution RASTA features, temporal information is integrated by filtering the auditory spectrum trajectories using a bank of multi-resolution bandpass filters as shown in Figure 2.6.

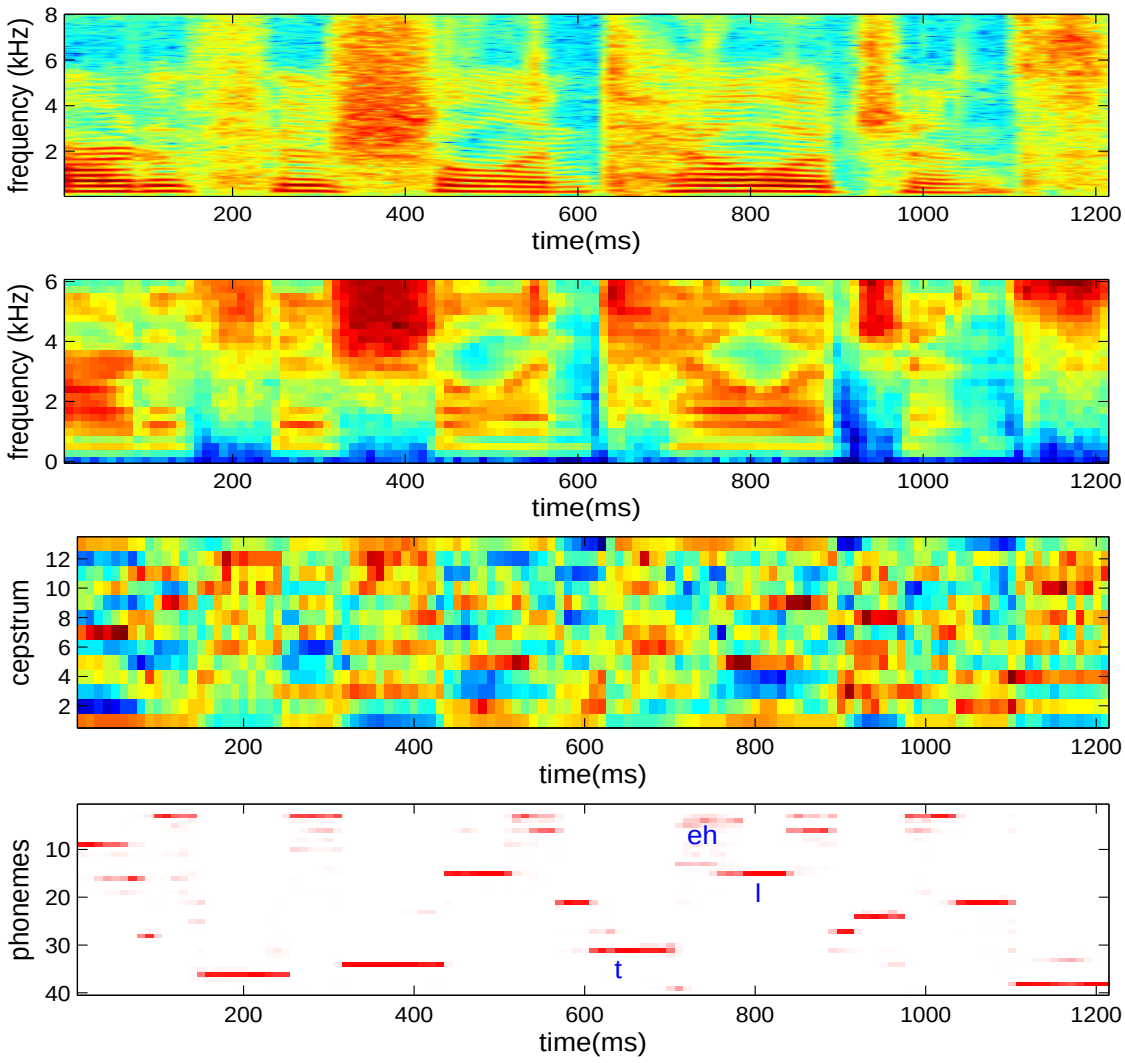


Figure 3.2. Different representations of the utterance *"artificial intelligence"* at different stages of feature extraction. (a) Fourier magnitude spectrum (b) mel auditory spectrum (c) mel frequency cepstral features and (d) phonetic class conditional probabilities. The utterance is transcribed as sequence of phonemes /ao/ /r/ /dx/ /ih/ /f/ /ih/ /sh/ /l/ /ih/ /n/ /t/ /eh/ /l/ /ih/ /d/ /jh/ /ih/ /n/ /s/ in the TIMIT database.

The computation of acoustic features from the auditory spectrum can be modeled as a linear time-invariant (LTI) system. For example, in the case of MFCC, the linear system consists of the DCT matrix and the FIR filters required to create the dynamic coefficients and a temporal context of 9 frames on the features. In the case of MRASTA feature extraction, the LTI system consists of a

bank of FIR filters with different time resolutions.

For the sake of completeness, in Figure 3.2 (d), we also plot the phonetic class conditional probabilities estimated by the MLP as a function of time. The plot shows the phoneme sequence /t/ /eh/ /l/, which is a part of the utterance. In the case of consonants /t/ and /l/, it can be seen that the classification is almost perfect. In the case of the vowel /eh/, the probability mass is spread across confusing vowels /ae/ and /aw/.

With this background, we discuss why rule based analysis of the MLP may not be applicable in the case of MLP classifier used for estimating the posterior probabilities:

- It can be seen from Figure 3.1 that, if the MLP is analyzed as a standalone system, then its functionality is revealed in terms of acoustic features (*e.g.,* cepstral parameters) which are not directly interpretable. In contrast, spectro-temporal patterns would be an ideal choice as they are intuitive and easily interpretable. More importantly, the spectro-temporal properties of speech sounds have been extensively researched in the context of human speech recognition.

- The features typically represent the temporal evolution of the spectral energies, and temporal information can be difficult to interpret in terms of parsimonious rules.

- In practical systems, the input features to the MLP have a large dimensionality typically in the range of 300-600. This will result in a large number of rules which are difficult to interpret.

- The derivation of rules often involves ad hoc heuristics, and the correctness of the derived rules obtained cannot be easily validated.

The above drawbacks can be overcome by incorporating a part of the feature extraction into the analysis framework as shown in Figure 3.1 and interpreting the system in terms of its input patterns. For example, in the case of MFCC feature extraction, if the DCT matrix and the FIR filters used in the computation of dynamic features are incorporated into the analysis, then the system can be interpreted in terms of auditory spectro-temporal patterns. The system under analysis is a cascade of a linear time-invariant (LTI) system and a static nonlinear system. Traditionally, nonlinear time-invariant systems have been analyzed using Volterra series.

3.2.3 Volterra Series

A linear time-invariant system is completely characterized by its impulse response function. If $x(t)$ denotes the input to an LTI system with impulse response function $h(t)$, then the output of the system $y(t)$ is given by

$$y(t) = \int_{\tau} h(\tau)x(t-\tau)d\tau \tag{3.1}$$

If the system is nonlinear and without memory (*i.e.,* static), then the input-output relationship of the system can be characterized using Taylor series. For example, suppose that $y(t) = f(x(t))$ is the nonlinear function, then the Taylor series expansion around a point b is given by

$$y(t) = \sum_{n=0}^{\infty} a_n \left[x(t) \ - \ b\right]^n \tag{3.2}$$

where the coefficients of the series $\{a_n\}$ are given by $a_n = \frac{1}{n!}f^{(n)}(b)$, where $f^{(n)}$ denotes the n^{th} order partial derivative of the function. If the Taylor series approximation of the transfer function converges for all values of the input $(-\infty, \infty)$, then the coefficients of the series $\{a_n\}_{n=1}^{\infty}$ about the point b can completely characterize the entire function.

Most systems in nature as well as engineering are nonlinear as well as dynamic, and analysis of such systems can be complicated. Vito Volterra (Allen, 1941) proposed an analog of the Taylor series representation to nonlinear dynamic systems, which is now popularly known as Volterra series. It combines the power series representation of a nonlinear system and the convolutional integral representation of an LTI system. For example, if $x(t)$ is the input to a nonlinear time-invariant (NLTI) system, then its output $y(t)$ can be expressed as an infinite series

$$y(t) = \sum_{n=0}^{\infty} \mathcal{G}_n \left[g_n, x(t)\right] \tag{3.3}$$

where $\{\mathcal{G}_n\}$ is the set of Volterra functionals, and $\{g_n\}$ is the set of Volterra kernels of the nonlinear system. For the purpose of illustration, the first three Volterra functionals are listed below.

$$\mathcal{G}_0 \left[g_0, x(t)\right] = g_0$$

$$\mathcal{G}_1\left[g_1, x(t)\right] = \int_{\tau} g_1(\tau)x(t-\tau)\ d\tau$$

$$\mathcal{G}_2\left[g_2, x(t)\right] = \int_{\tau_1}\int_{\tau_2} g_2(\tau_1,\tau_2)x(t-\tau_1)x(t-\tau_2)\ d\tau_1 d\tau_2$$

It can be seen that each term in the Volterra series is a multi-dimensional[1] convolution between the input to the system and its Volterra kernels. The Volterra kernels $\{g_0, g_1, g_2 \ldots g_\infty\}$ completely characterize the nonlinear time-invariant system. The first order Volterra functional G_1 is the linear convolutional integral, and its corresponding kernel g_1 is the most familiar time-domain description of an LTI system, *i.e.,* the impulse response function.

It can be seen that the n^{th} order Volterra functional is a homogeneous of order n. Consequently, the first order Volterra kernel represents the linear part of the nonlinear system, the second order kernel represents the quadratic part, and so on. In the following subsections, we discuss some of the properties of Volterra series such as its frequency domain representation and convergence. (Boyd *et al.*, 1984) provide an in-depth discussion on the analytical foundations of Volterra series.

Frequency Domain Representation

The input-output relationship of an LTI system in the time domain is given by (3.1). In the frequency domain, the input-output relationship can be written as $Y(\omega) = H(\omega)X(\omega)$, where $X(\omega)$, $Y(\omega)$, and $H(\omega)$ denote the Fourier transform of the signals $x(t)$, $y(t)$, and $h(t)$ respectively. The input-output of the NLTI system given by (3.3) can also be represented in the frequency domain. Let $G_n(\omega_1, \omega_2, \ldots, \omega_n)$ denote the Fourier transform of the n^{th} order Volterra kernel $g_n(\tau_1, \tau_2, \ldots, \tau_n)$, given by

$$G_n(\omega_1, \omega_2, \ldots, \omega_n) = \int_{\tau_1} \ldots \int_{\tau_n} g_n(\tau_1, \ldots, \tau_n)\ \exp(-j\omega_1\tau_1 - j\omega_2\tau_2 \ldots - j\omega_n\tau_n)\ d\tau_1, \ldots d\tau_n$$

[1]Although the system is single-input single-output, the convolution is multi-dimensional as for example the second order functional is a product of two terms $x(t-\tau_1)$ and $x(t-\tau_2)$. The third order functional is a product of three terms and so on.

By applying Fourier transform on (3.3), the input-output relationship of the NLTI system in the frequency domain can be represented as an infinite series as an infinite series as

$$Y(\omega) = g_0\delta(\omega) + G_1(\omega)X(\omega) + \int_{\omega_1} G_2(\omega_1, \omega-\omega_1)\, X(\omega)X(\omega-\omega_1)\, d\omega + \int_{\omega_1}\int_{\omega_2} G_3(\omega_1,\omega_2,\omega-\omega_1-\omega_2)\, X(\omega_1)X(\omega_2)X(\omega-\omega_1-\omega_2)\, d\omega_1 d\omega_2 + \ldots$$

Convergence of Volterra Series

A power series may converge for certain values of the input and may diverge for other values. Hence, it is important to ascertain the region of convergence of a power series. The Volterra series can be expressed compactly as

$$\begin{aligned} y(t) &= g_0 + \sum_{n=0}^{\infty} \int \ldots \int g_n(\tau_1, \ldots \tau_n)\, x(t-\tau_1)\ldots x(t-\tau_n)\, d\tau_1 \ldots d\tau_n \\ |y(t)| &\leq |g_0| + \sum_{n=0}^{\infty} \int \ldots \int |g_n(\tau_1, \ldots \tau_n)\, x(t-\tau_1)\ldots x(t-\tau_n)|\, d\tau_1 \ldots d\tau_n \\ &\leq |g_0| + \sum_{n=0}^{\infty} \|g_n\|\, \|x\|^n \end{aligned} \tag{3.4}$$

where, $\|g_n\|$ denotes the L^1 norm of the n^{th} order Volterra kernel, and $\|x\|$ denotes the L^∞ norm of the input, and defined by

$$\begin{aligned} \|g_n\| &\triangleq \int \ldots \int |g_n(\tau_1, \ldots \tau_n)|\, d\tau_1 \ldots d\tau_n \\ \|x\| &\triangleq \sup_t \{x(t)\} \end{aligned}$$

By applying the root test for convergence [2] (Rudin, 1976) on the Volterra series (3.4), it can be seen that the series is absolutely convergent in the region $\|x\| < \rho$, where the radius of convergence is given by

$$\rho = \left(\lim_{n\to\infty} \sup \left(\|g_n\|\right)^{\frac{1}{n}} \right)^{-1} \tag{3.5}$$

[2] A power series of the form $\sum_{n=0}^{\infty} a_n x^n$ is absolutely convergent in the region $\mid x \mid < \rho$, where the radius of convergence ρ is given by $\rho = \left(\lim_{n\to\infty} \sup (a_n)^{\frac{1}{n}}\right)^{-1}$.

Calculation of Volterra Kernels

We first discuss the calculation of Volterra kernels for a simple nonlinear time-invariant system shown in Figure 3.3. It consists of a cascade of an LTI system with impulse response function $h(t)$ and a polynomial nonlinear function given by $\phi(u) = a_0 + a_1 u + a_2 u^2$.

Figure 3.3. Block schematic of a simple nonlinear time invariant system.

If $x(t)$ is the input to the system, the output of the system is given by

$$y(t) = a_0 + a_1 \left(\int_\tau h(\tau) x(t-\tau) d\tau \right) + a_2 \left(\int_\tau h(\tau) x(t-\tau) d\tau \right)^2 \tag{3.6}$$

The Volterra series representation for a single-input single-output (3.3) can be expanded as

$$y(t) = g_0 + \int_\tau g_1(\tau) x(t-\tau) d\tau + \int_{\tau_1} \int_{\tau_2} g(\tau_1, \tau_2) x(t-\tau_1) x(t-\tau_2) d\tau_1 d\tau_2 + \dots \tag{3.7}$$

By comparing the (3.6) to (3.7), the Volterra kernels can be identified as

$$\begin{aligned} g_0 &= a_0 \\ g_1(\tau) &= a_1\, h(\tau) \\ g_2(\tau_1, \tau_2) &= a_2\, h(\tau_1)\, h(\tau_2) \\ g_n(\tau_1, \dots, \tau_n) &= 0 \quad \forall \quad n > 2 \end{aligned}$$

For the above example, the calculation of the Volterra kernels is straightforward as the nonlinearity is already in the form of a polynomial function. However, when the nonlinear function is more complex such as sigmoid (which is typically used as nonlinear activation function in an MLP), then it has to be approximated as a power series. Not all functions admit a power series approximation which is convergent for all values of the input. In this case, the approximation is done with respect to an error criterion within its operating interval. The calculation of Volterra kernels for an MLP are discussed in Section 3.3.

3.2.4 Wiener Series

The Volterra analysis of NLTI systems requires the knowledge about the system and its parameters (*e.g.,* the impulse response function $h(\tau)$ and the nonlinear function $\phi(.)$ in the example shown in Figure 3.3). If the functionality of the system is not known, then identification of Volterra kernels is not straightforward. It can be seen from (3.3) that the homogeneous functionals in the Volterra series are correlated. As a result, estimation of the individual Volterra kernels can become complicated as it can lead to simultaneously solving a set of integral equations. Alternatively, the Volterra series can be orthogonalized with respect to white Gaussian noise to derive the Wiener series (Wiener, 1958). The Wiener series for a single-input, single-output nonlinear system is given by

$$y(t) = \sum_{n=0}^{\infty} \mathcal{H}_n \left[h_n, x(t)\right] \tag{3.8}$$

where, $x(t)$ is the input, $y(t)$ is the output of the system, $\{h_n\}$ is the set of the Wiener kernels for the nonlinear system, and $\{\mathcal{H}_n\}$ is the complete set of orthogonal functionals, satisfying the property

$$E_X \left\{\mathcal{H}_m \left[h_m, x(t)\right] \mathcal{H}_n \left[h_n, x(t)\right]\right\} = 0 \qquad \text{if} \quad m \neq n \tag{3.9}$$

where $x(t)$ is white Gaussian noise as a function of time. $E\{.\}$ denotes the expected value with respect to the noise. If σ^2 denotes the variance of the white noise, the first four functionals in the Wiener series are given by

$$\begin{aligned}
\mathcal{H}_0 \left[h_0, x(t)\right] &= h_0 \\
\mathcal{H}_1 \left[h_1, x(t)\right] &= \int_{\mathbb{R}} h_1(\tau) x(t-\tau) d\tau \\
\mathcal{H}_2 \left[h_2, x(t)\right] &= \int_{\mathbb{R}^2} h_2(\tau_1\tau_2) x(t-\tau_1) x(t-\tau_2) d\tau_1\tau_2 - \sigma^2 \int_{\mathbb{R}} h_2(\tau,\tau) d\tau \\
\mathcal{H}_3 \left[h_3, x(t)\right] &= \int_{\mathbb{R}^3} h_3(\tau_1,\tau_2,\tau_3) x(t-\tau_1) x(t-\tau_2) x(t-\tau_3) d\tau_1\tau_2\tau_3 - 3\sigma^2 \int_{\mathbb{R}^2} h_3(\tau_1,\tau_2,\tau_2) x(t-\tau_1) d\tau_1 d\tau_2
\end{aligned}$$

A simple derivation of the Wiener series using Gram-Schmidt orthogonalization of the Volterra series is described in (Ogunfunmi, 2007). Due to the orthogonal property of the functionals, the Wiener kernels can be identified using the standard cross-correlation method (Lee and Schetzen,

1965). For this, we present the system with white Gaussian noise of variance σ^2 and find a correlation between the input and the output. The n^{th} order Wiener kernel is given by (Marmarelis and Naka, 1974) as

$$h_n(\tau_1, \tau_2 \dots \tau_n) = \frac{1}{n!\sigma^{2n}} E_X \left\{ \left[y(t) - \sum_{m=0}^{n-1} \mathcal{H}_m \left[h_m, x(t) \right] \right] x(t-\tau_1) \dots x(t-\tau_n) \right\} \tag{3.10}$$

In the Wiener analysis, the system under analysis is treated as a black box. The concept of Wiener analysis can be intuitively explained as follows. By using white noise as stimulus, the system is almost exhaustively tested for all possible combinations of the inputs, provided sufficiently large number of noise samples are generated. Cross-correlation can identify average patterns that the system responds to. However, unlike in Volterra analysis, the n^{th} order Wiener kernel does not necessarily give the total n^{th} order response of the system. It contains terms which represent the lower order parts of the system.

Wiener analysis has been particularly useful in the analysis of biological systems, where analytical calculation is not possible as the system cannot be represented mathematically, but the response of the system to pre designated stimulus can be estimated (Korenberg and Hunter, 1996; Klein *et al.*, 2000; Marmarelis, 2004).

3.3 Volterra analysis of MLP based acoustic modeling

The Volterra theory of nonlinear dynamic systems have been previously applied in the analysis of neural networks namely recurrent neural networks (Hakim *et al.*, 1991) and time-delay neural networks (Hakim *et al.*, 1991; Wray and Green, 1994). The mathematical formulation provided in these works for a time-delay neural network is briefly discussed below, as this chapter is an extension to these works.

A time-delay neural network (Waibel *et al.*, 1989) can be viewed as a cascade of a delay filter bank and an MLP. Suppose that the time-delay neural network is trained with L delayed time instants $t_1, t_2, \dots t_L$ (one present $t_1 = 0$ and $L-1$ past) of a vector valued input $\mathbf{x}_t = [x_1(t), \dots x_k(t) \dots x_K(t)]'$. The concatenated input to the MLP is given by $\mathbf{u}_t = [u_{1,1}(t), \dots u_{k,l}(t), \dots u_{K,L}(t)]'$, where $u_{k,l}(t) =$

$x_k(t - t_l)$. The vector valued linear activation at the output of the MLP is given by

$$\mathbf{y}_t = \mathbf{b}_o + \mathbf{C}\ \Phi(\mathbf{b}_h + \mathbf{W}\mathbf{u}_t) \tag{3.11}$$

where, $\mathbf{W}$ and $\mathbf{C}$ denote the weight matrices connecting input to hidden, and hidden to output layers of the MLP respectively, and $\mathbf{b}_h$ and $\mathbf{b}_o$ respectively denote the bias vectors at the hidden and output layer of the MLP. $\Phi(.)$ denotes the vector valued nonlinear activation function at the hidden layer of the MLP. Prior works (Hakim *et al.*, 1991; Wray and Green, 1994) discuss the Volterra series representation of the above multi-input multi-output nonlinear time-invariant system $(\mathbf{x}_t, \mathbf{y}_t)$ similar to (3.3).

The theory developed for the time delay neural network cannot be applied in the analysis of trained MLPs used in ASR as feature extraction cannot be included in the analysis. In Section 3.2.2, we discussed that the later stages (*e.g.,* auditory spectrum to cepstral transformation in MFCC, delta and delta-delta feature computation, etc) of feature extraction can be modeled as an LTI system. If Volterra analysis can be applied to the combined system consisting of a part of feature extraction and the MLP model, then the functionality of the system can be discovered in terms of more interpretable information such as spectro-temporal patterns in the auditory spectrum shown in Figure 3.2 (b).

With this motivation, we develop a generic mathematical framework to apply Volterra series to model a nonlinear dynamic system comprising of a linear time-invariant system followed by a three layered MLP. For mathematical convenience, the softmax nonlinear function at the output of the MLP is excluded from the analysis. This does not affect the interpretability as the output units are still phonemes, and the rank ordering of the estimates is not altered. The MLP is trained with features $u_{k,l}(t)$ which are obtained by convolving the input $x_k(t)$ with the impulse response function $h_l(t)$ of the linear system as $u_{k,l}(t) = x_k(t) * h_l(t)$. More specifically, we consider the following three systems as schematized in Figure 3.4.

System-1: A finite impulse response (FIR) filter bank followed by a three layered MLP as shown in Figure 3.4 (a). The filters can have arbitrary impulse response functions $h_l(t),\ l = 1, 2, \ldots L$. As a particular case, if the filters have a time-delayed impulse response functions $h_l(t) = \delta(t - t_l)$, then the Volterra kernels reduce to the solutions provided in (Hakim *et al.*, 1991; Wray and Green, 1994)

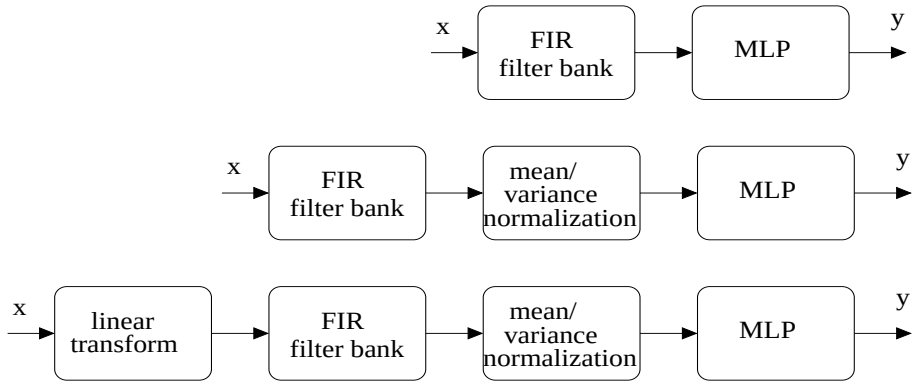


Figure 3.4. The Volterra theory of nonlinear systems is applied to the above three systems. (a) FIR filter bank followed by a three layer MLP (b) the features to the MLP are normalized to zero mean and unit variance (c) a linear transformation matrix preceding the FIR filter bank.

for the time delay neural network.

System-2: In practical applications, the input features to the MLP are normalized to zero mean and unit variance as shown in Figure 3.4 (b). Feature normalization mainly helps in achieving faster convergence of the back-propagation training (LeCun *et al.*, 1998) as well as addressing feature mismatch to a certain extent. In Section 3.3.2, we discuss the calculation of Volterra kernels when features are normalized.

System-3: In feature extraction techniques such as MFCC, there is a linear transformation matrix (discrete cosine transform) preceding the FIR filters as shown in Figure 3.4 (c). The analytical calculation of Volterra kernels for such a system is discussed in Section 3.3.3.

3.3.1 Calculation of Volterra Kernels: Three Layered MLP

Figure 3.5 is a detailed block schematic of a cascade of an FIR filter bank and a three layered MLP shown in Figure 3.4 (a). The vector $\mathbf{x}_t = [x_1(t), \ldots x_k(t), \ldots x_K(t)]'$ denotes the input to the system under analysis at time t. An FIR filter bank with impulse response function $h_l(t),\ l = 1 \ldots L$ is applied on each of the K inputs. The output of the filter bank is denoted by the vector $\mathbf{u}_t = [u_{1,1}(t), \ldots u_{k,l}(t), \ldots u_{K,L}(t)]'$, where $u_{k,l}(t)$ is given by the convolution between $x_k(t)$ and $h_l(t)$ as

$$u_{k,l}(t) = \int_{\tau} h_l(\tau) x_k(t-\tau) d\tau \tag{3.12}$$

The output of the filter bank forms the input to the MLP, whose trained parameter set is denoted

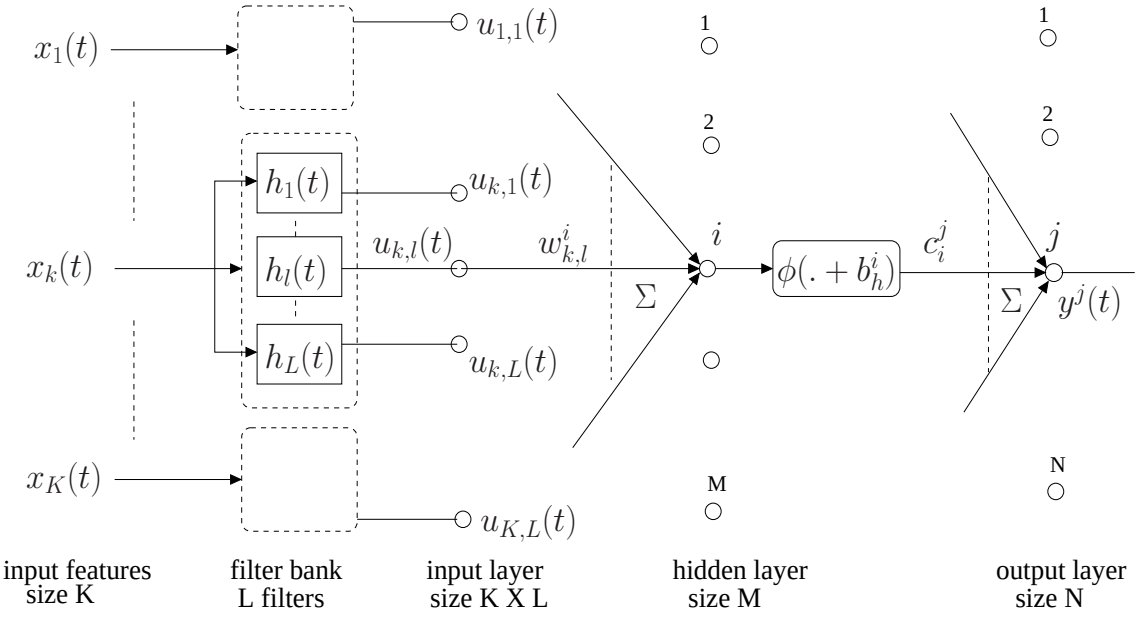


Figure 3.5. Block schematic of a cascade of an FIR filter bank and a three layered MLP.

by $\{\mathbf{W}, \mathbf{b}_h, \mathbf{C}, \mathbf{b}_o\}$. Here, $\mathbf{W}$ denotes the weight matrix connecting the input layer of size $K.L$ to the hidden layer of size M. An element $w^i_{k,l}$ in the matrix denotes the weight associated with the output of the filter bank $u_{k,l}(t)$ at node i in the hidden layer. The vector $\mathbf{b}_h = [b^1_h, \ldots, b^i_h, \ldots b^M_h]'$ denotes the bias vector at the hidden layer of the MLP. The matrix $\mathbf{C}$ denotes the weights connecting the hidden layer to the output layer of size N. An element c^j_i in the matrix denotes the weight between node i in the hidden layer and node j in the output layer of the MLP. The vector $\mathbf{b}_o = [b^1_o, \ldots, b^j_o, \ldots b^N_o]'$ denotes the bias vector at the output layer. Furthermore, if $\Phi(.)$ denotes the vector valued [3] sigmoidal function at the hidden layer of the MLP, then the linear activation vector at the output of the MLP $\mathbf{y}_t = [y^1(t), \ldots y^j(t), \ldots y^N(t)]'$ is given by (3.11).

The system schematized in Figure 3.5, and characterized by (3.12) and (3.11) can be viewed as a multi-input ($\mathbf{x_t}$), multi-output ($\mathbf{y_t}$), nonlinear time-invariant system. The FIR filter bank introduces memory in the system and the activation functions in the hidden layer introduces nonlinearity. Without loss of generality, the above system can be treated as N parallel, multi-input, single-output

[3] $\Phi(.)$ denotes the vector valued activation function given by $\Phi(.) = [\phi(.), \phi(.), \ldots, \phi(.)]'$, where $\phi(.)$ denotes the scalar valued activation function.

subsystems and analyzed independently. The linear output at node j in the output layer is given by

$$y^j(t) = b_o^j + \sum_{i=1}^{M} c_i^j \phi\left(b_h^i + \sum_{k=1}^{K}\sum_{l=1}^{L} w_{k,l}^i u_{k,l}(t)\right), \qquad j = 1, \ldots N \tag{3.13}$$

The system characterized by (3.12) and (3.13) is difficult to analyze in its present parametric form due to the presence of the nonlinear activation function $\phi(.)$. However, if the activation function can be approximated as a power series, then the same system can be alternatively characterized using Volterra series as

$$y^j(t) = g_0^j + \sum_{k_1=1}^{K} \int_{\tau_1} g_{k_1}^j(\tau_1)\, x_{k_1}(t-\tau_1)\, d\tau_1 +$$
$$\sum_{k_1=1}^{K}\sum_{k_2=1}^{K} \int_{\tau_1}\int_{\tau_2} g_{k_1 k_2}^j(\tau_1,\tau_2)\, x_{k_1}(t-\tau_1)\, x_{k_2}(t-\tau_2)\, d\tau_1 d\tau_2 + \ldots, \qquad j = 1, \ldots N \tag{3.14}$$

In this way, a set of Volterra kernels is identified for each of the N output classes of the MLP, given by $\{g_0^j,\ g_{k_1}^j(\tau_1),\ g_{k_1 k_2}^j(\tau_1,\tau_2), \ldots\}_{j=1}^N$. For the output class j, g_0^j is the zeroth order Volterra kernel, and it reveals the constant part of the nonlinear system. The first order Volterra kernels $g_{k_1}^j(\tau_1)$ reveal the linear part of the nonlinear system. Similarly, $g_{k_1 k_2}^j(\tau_1,\tau_2)$ is the second order Volterra kernel of the system, and reveals the quadratic part of the system. The variables $\tau_1, \tau_2 \ldots$ denote time, and $k_1, k_2 \ldots$ denote the indices of the input. The Volterra kernels can be identified in terms of the parameters of the MLP and the impulse response function of the filter bank. To identify the Volterra kernels, (3.13) is rewritten as

$$y^j(t) = b_o^j + \sum_{i=1}^{M} c_i^j\ \phi\left(b_h^i + s_i(t)\right) \tag{3.15}$$

where, $s_i(t)$ denotes the activation value to the nonlinear function at the hidden node i, given by

$$s_i(t) = \sum_{k=1}^{K}\sum_{l=1}^{L} w_{k,l}^i u_{k,l}(t) \tag{3.16}$$

Suppose that the nonlinear function $\phi(.)$ at the hidden layer is approximated as a power series

$$\phi\left(b_h^i + s_i(t)\right) \quad = \quad \sum_{n=0}^{\infty} a_{n,i}\ [s_i(t)]^n \tag{3.17}$$

where $a_{0,i}, a_{1,i} \ldots$ are the scalar coefficients of the series. By substituting (3.17) in (3.15), we obtain

$$y^j(t) = b_o^j + \sum_{i=1}^{M} c_i^j \sum_{n=0}^{\infty} a_{n,i} \left[s_i(t) \right]^n \tag{3.18}$$

By substituting (3.12) and (3.16) in (3.18), rearranging the terms, and comparing the resulting equation to the Volterra series equation (3.14), the first three Volterra kernels are identified as

$$g_0^j = b_o^j + \sum_{i=1}^{M} c_i^j \, a_{0,i} \tag{3.19}$$

$$g_{k_1}^j(\tau_1) = \sum_{i=1}^{M} c_i^j \, a_{1,i} \sum_{l_1=1}^{L} w_{k_1 l_1}^i h_{l_1}(\tau_1) \tag{3.20}$$

$$g_{k_1 k_2}^j(\tau_1, \tau_2) = \sum_{i=1}^{M} c_i^j a_{2,i} \sum_{l_1=1}^{L} \sum_{l_2=1}^{L} w_{k_1 l_1}^i w_{k_2 l_2}^i h_{l_1}(\tau_1) h_{l_2}(\tau_2) \tag{3.21}$$

The intermediate steps involved in the calculation of Volterra kernels are given in Appendix A.1 at the end of this thesis. The power series expansion of the nonlinear activation function at the hidden layer of the MLP is discussed in detail in Section 3.3.4.

3.3.2 Calculation of Volterra Kernels: Feature Normalization

In practical applications, the features to the MLP are normalized to zero mean and unit variance as shown in Figure 3.4 (b). In this section, we discuss the calculation of Volterra kernels when the features are normalized to zero mean and unit variance. Note that as the MLP is also trained using normalized features, the parameter set of the MLP $\{\mathbf{W}, \mathbf{b}_h, \mathbf{C}, \mathbf{b}_o\}$ is different from the one discussed above. Addressing variance normalization alone is straightforward as the feature variances just scale the weights connecting the input and the hidden layer. On the other hand, addressing mean normalization needs careful consideration as it is not a linear operation from a system theoretic point of view. [4]

Suppose that the feature vector component $u_{k,l}(t)$ has a mean $\mu_{k,l}$ and a standard deviation $\sigma_{k,l}$. The input to the MLP is given by $\left(u_{k,l}(t) - \mu_{k,l}\right)/\sigma_{k,l}$. By substituting the normalized features in

[4] It is important to distinguish the difference between a linear classifier, linear function, and a linear system. We illustrate this with single layer perceptron with weight matrix $\mathbf{W}$ and bias vector b as an example. If $\mathbf{x}$ denotes the input, the output $\mathbf{y}$ of the perceptron is given by $\mathbf{y} = \Psi(\mathbf{b} + \mathbf{W}\mathbf{x})$, where $\Psi(.)$ denotes the softmax function. It is a linear classifier as the decision boundary is a hyperplane. However, it is not a linear function and system. If the softmax function is dropped, then it becomes a linear function, but still not a linear system because of the bias. If $\mathbf{b} = \mathbf{0}$, then it is a linear system as well.

(3.16), we obtain

$$s_i(t) = \sum_{k=1}^{K}\sum_{l=1}^{L} w_{k,l}^{i}\ \frac{u_{k,l}(t) - \mu_{k,l}}{\sigma_{k,l}}$$

$$= \hat{s}_i(t)\ -\ \Delta_i \tag{3.22}$$

where,

$$\hat{s}_i(t) = \sum_{k=1}^{K}\sum_{l=1}^{L} \hat{w}_{k,l}^{i}\ u_{k,l}(t) \tag{3.23}$$

$$\hat{w}_{k,l}^{i} = \frac{w_{k,l}^{i}}{\sigma_{k,l}} \tag{3.24}$$

$$\Delta_i = \sum_{k=1}^{K}\sum_{l=1}^{L} w_{k,l}^{i}\ \frac{\mu_{k,l}}{\sigma_{k,l}} \tag{3.25}$$

If $a_{0,i}, a_{1,i}, a_{2,i}, \ldots$ denote the coefficients of the power series approximation (3.17) of the sigmoidal function at i^{th} node of the hidden layer, then the linear output at j^{th} node of the output layer can be written from (3.18) as

$$\begin{aligned} y^j(t) &= b_o^j + \sum_{i=1}^{M} c_i^j \sum_{n=0}^{\infty}\ a_{n,i}\ \left(\hat{s}_i(t) - \Delta_i\right)^n \\ &= b_o^j + \sum_{n=0}^{\infty}\sum_{i=1}^{M} c_i^j\ a_{n,i} \sum_{r=0}^{n} \binom{n}{r}\ \left(\hat{s}_i(t)\right)^r\ \left(-\Delta_i\right)^{n-r} \end{aligned}$$

The above equation can be expressed as an infinite series of functionals as

$$y^j(t) = \sum_{n=0}^{\infty} \mathcal{G}_n^j\left[g_n^j(\tau_1, \ldots \tau_n),\ x_1, x_2, \ldots x_K\right] \tag{3.26}$$

where the n^{th} order functional is given by

$$\mathcal{G}_n^j\left[g_n^j(\tau_1,\dots\tau_n),\ x_1,\dots x_K\right] = \sum_{r=0}^{n}\sum_{k_1=1}^{K}\dots\sum_{k_r=1}^{K}\int_{\tau_1}\dots\int_{\tau_r} g^{j,n,r}(\tau_1,\dots\tau_r)\, x_1(t-\tau_1)\dots x_r(t-\tau_r)d\tau_1\dots d\tau_r \tag{3.27}$$

and the corresponding kernels are given by

$$g^{j,n,r}(\tau_1,\dots\tau_r) = \sum_{i=1}^{M} c_i^j\ \hat{a}_{n,r,i}\ \sum_{l_1=0}^{L}\dots\sum_{l_r=0}^{L}\hat{w}_{k_1,l_1}^i\dots\hat{w}_{k_r,l_r}^i h_{l_1}(\tau_1)\dots h_{l_r}(\tau_r) \tag{3.28}$$

where $\hat{a}_{n,r,i} = \binom{n}{r}(-\Delta_i)^{n-r}$. Strictly speaking, the above formulation is not Volterra series as the functionals are not homogeneous in terms of the input. It can be seen from (3.27) that the n^{th} order functional contains terms of order $r \leq n$. In other words, the n^{th} order functional will include information about the linear, quadratic, and all higher order components of the nonlinear system up to the order n. Nonetheless, the kernels $g^{j,n,n}(\tau_1,\dots\tau_r)$ given by (3.28) can still approximately reveal the n^{th} order component of the nonlinear system.

The above problem can also be effectively circumvented by redefining the input representation $x_k(t)$ in the Volterra synthesis equation (3.14) as a zero mean signal, *i.e.,* $x_k(t) \triangleq x_k(t) - m_k$, where m_k denotes the mean of $x_k(t)$. In this way, mean normalization of the features can be achieved without affecting the functionality of the system. The variance normalization is incorporated into the weight matrix connecting the input layer of the MLP to the hidden layer as (3.24). In the matrix notation, this is equivalent to modifying the weight matrix $\hat{\mathbf{W}} = \mathbf{W}\mathbf{\Sigma}^{-\frac{1}{2}}$, where $\mathbf{\Sigma}$ denotes the diagonal covariance matrix of the features. The Volterra kernels can be subsequently estimated using (3.19)-(3.21).

The Volterra kernels derived correspond to an input $x_k(t)$ which is normalized to zero mean. Furthermore, the interpretation could be in terms of the inputs which are normalized to zero mean and unit variance. In this case, the first order Volterra kernel is given by $\sigma_k g_k(\tau)$, the second order Volterra kernel is given by $\sigma_{k_1}\sigma_{k_2}g_{k_1k_2}(\tau_1,\tau_2)$, and so on.

3.3.3 Calculation of Volterra Kernels : Linear Transformation

Suppose that the input to the system $\mathbf{x}_t = [x_1(t), \ldots x_k(t), \ldots x_K(t)]'$ is transformed into an intermediate representation $\hat{\mathbf{x}}_t = [\hat{x}_1(t), \ldots \hat{x}_f(t), \ldots \hat{x}_F(t)]'$ using a linear transformation matrix $\mathbf{D} = [d_{f,k}]_{F \times K}$ as

$$\hat{x}_f(t) = \sum_{k=1}^{K} d_{f,k}\, x_k(t) \quad f = 1, 2, \ldots F, \quad \text{where} \quad F \leq K$$

or, more compactly, $\hat{\mathbf{x}}_t = \mathbf{D}\mathbf{x}_t$. The MLP is trained using features $\hat{\mathbf{u}}_t = [\hat{u}_{1,1}(t), \ldots \hat{u}_{k,l}(t), \ldots \hat{u}_{K,L}(t)]'$ which are obtained by applying an FIR filter bank of L filters on the intermediate representation $\hat{\mathbf{x}}_t$. In this section, we discuss the calculation of Volterra kernels for the system $(\mathbf{x}_t, \mathbf{y}_t)$ shown in Figure 3.4 (c), where $\mathbf{y}_t$ denotes the linear activation vector at the output of the MLP. An example of this scenario is MFCC feature extraction, where cepstral features $\hat{\mathbf{x}}_t$ are obtained by applying DCT matrix $\mathbf{D}$ on the log-spectral energies $\mathbf{x}_t$ in the mel filter banks. The static and dynamic cepstral features $\hat{\mathbf{u}}_t$ are obtained by applying FIR filters on $\hat{\mathbf{x}}_t$.

Then, the output of the filter bank can be expressed using (3.12) as

$$\begin{aligned} \hat{u}_{f,l} &= \int_{\tau} h_l(\tau) \sum_{k=1}^{K} d_{f,k}\, x_k(t-\tau)\, d\tau \\ &= \sum_{k=1}^{K} d_{f,k}\, u_{k,l} \\ \text{where} \quad u_{k,l} &= \int_{\tau} h_l(\tau)\, x_k(t-\tau)\, d\tau \end{aligned} \tag{3.29}$$

It can be seen from (3.29) that the transformation matrix $\mathbf{D}$ can be incorporated at the output of the FIR filter bank as $\hat{\mathbf{u}}_t = \hat{\mathbf{D}}\, \mathbf{u}_t$, where $\mathbf{u}_t = [u_{1,1}(t), \ldots u_{k,l}(t), \ldots u_{K,L}(t)]'$. The new transformation matrix $\hat{\mathbf{D}}$ of size $F.L \times K.L$ can be obtained from the original transformation matrix $\mathbf{D}$.[5]

If $\boldsymbol{\mu}_u$ denotes the mean of the feature vector, $\boldsymbol{\Sigma}_u$ denotes the diagonal covariance matrix, and $\mathbf{W}$ denotes the weight matrix connecting the input and hidden layer of the MLP, then the linear

[5]The new transformation matrix $\hat{\mathbf{D}} = \mathbf{D}_B\mathbf{P}$, where $\mathbf{D}_B$ is the block diagonal matrix created by repeating matrix $\mathbf{D}$ for L times along the diagonal and $\mathbf{P}$ denotes the permutation matrix required to rearrange the vector $[\hat{u}_{1,1}(t), \ldots \hat{u}_{1,L}(t), \ldots \hat{u}_{f,1}(t) \ldots \hat{u}_{f,l}(t) \ldots \hat{u}_{f,L}(t) \ldots \hat{u}_{F,L}(t)]'$ as $[\hat{u}_{1,1}(t), \ldots \hat{u}_{F,1}(t), \ldots \hat{u}_{1,l}(t) \ldots \hat{u}_{f,l}(t) \ldots \hat{u}_{F,l}(t) \ldots \hat{u}_{F,L}(t)]'$.

activation vector (without bias) at the hidden layer of the MLP is given by

$$\mathbf{s}_t = \mathbf{W}\boldsymbol{\Sigma}_u^{-\frac{1}{2}}(\hat{\mathbf{D}}\mathbf{u}_t - \boldsymbol{\mu}_u)$$

As discussed in the previous section, the mean normalization of the features can also be incorporated at the input level without loss of functionality. In this case, the linear transformation matrix $\hat{\mathbf{D}}$ can be incorporated into the weight matrix connecting the input to the hidden layer as

$$\mathbf{W}' = \mathbf{W}\boldsymbol{\Sigma}_u^{-\frac{1}{2}}\hat{\mathbf{D}} \tag{3.30}$$

3.3.4 Polynomial Expansion of the Activation Function

The most important aspect in the derivation of the Volterra kernels is the approximation of the nonlinear activation function at the hidden layer as a power series in the form (3.17). In practice, the power series is fixed to finite order P, and this decides the order of the Volterra series expansion. In the following discussion, we drop the subscript for time and the index of the hidden node for clarity, and rewrite (3.17) as

$$\phi(b+s) \approx \sum_{n=0}^{P} a_n s^n \tag{3.31}$$

The objective here is to estimate the coefficients $a_0, a_1, a_2, \ldots a_P$ to satisfy the above approximation. This is discussed in the following sections.

Taylor series

Taylor series expansion is a natural choice for approximating nonlinear functions as a power series as discussed in (Wray and Green, 1994). As a consequence of feature normalization, the linear activation value at the hidden node s has a mean equal to zero. The Taylor series expansion for the function $\phi(b+s)$ around the point $s=0$ is given by

$$\phi(s+b) \approx \sum_{n=0}^{P} \frac{\phi^{(n)}(b)}{n!} s^n$$

Sigmoidal functions [6] do not admit a convergent Taylor series for all values of the input. For example, the Taylor series expansion of the hyperbolic tangent function $\tanh(s)$ around the point $s = 0$ is convergent in the region $s \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. In the case of the MLP, it cannot be guaranteed that the input to the sigmoidal function is within the region of convergence. Hence, convergence of Volterra series in the operating region of the MLP cannot be guaranteed. Taylor series can only ensure that the first P derivatives of the power series around the point $s = 0$ are same as that of the actual function.

Mean square estimation

According to the Weierstrass approximation theorem, any continuous function defined on a finite interval can be uniformly approximated to the desired level of accuracy by a polynomial function. This theorem has also been extended to neural networks (Hecht-Nielsen, 1987; Cotter, 1990). As a result, it is sufficient to approximate the nonlinear activation function as a polynomial function in its region of operation using the mean square error criterion (Hakim *et al.*, 1991).

Suppose that the input features to the MLP $\mathbf{u}$ have mean vector of $\boldsymbol{\mu}_u$ and a full covariance matrix $\boldsymbol{\Sigma}_{uu}$, the input to the activation function at the hidden layer can be represented in the vectorial notation as

$$\mathbf{s} = \mathbf{W}\boldsymbol{\Sigma}_u^{-\frac{1}{2}}(\mathbf{u} - \boldsymbol{\mu}_u)$$

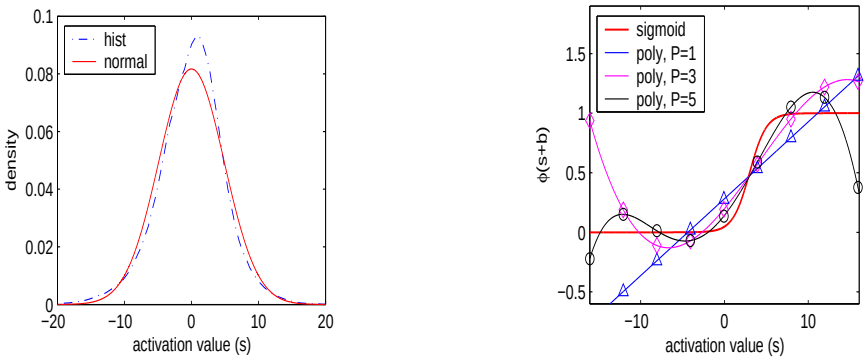
where, $\boldsymbol{\Sigma}_u$ denotes the diagonal matrix containing the feature variances. The covariance matrix of the activation values at the hidden layer is given by

$$\boldsymbol{\Sigma}_{ss} = \mathbf{W}\boldsymbol{\Sigma}_u^{-\frac{1}{2}}\boldsymbol{\Sigma}_{uu}\boldsymbol{\Sigma}_u^{-\frac{1}{2}}\mathbf{W}^T \tag{3.32}$$

As a consequence of feature normalization, the linear activation s (without bias) at the hidden layer have a mean equal to zero. We model the activation values at each hidden node as a unimodal Gaussian random variable with zero mean and the standard deviation obtained from the covariance matrix $\boldsymbol{\Sigma}_{ss}$.

[6] A generic definition of sigmoidal function is given in (Cybenko, 1989) as a continuous monotonic function $\phi(s)$, such that $\lim_{s\to-\infty}\phi(s) = c_1$ and $\lim_{s\to\infty}\phi(s) = c_2$, where c_1 and c_2 are scalar constants. For sigmoid, $c_1 = 0, c_2 = 1$ and hyperbolic tangent $c_1 = -1, c_2 = 1$.

The coefficients of the polynomial function are chosen to optimize the least mean square error between the sigmoidal function $\phi(b+s)$ and its polynomial approximation, assuming that s is normally distributed with zero mean and variance obtained from (3.32). The estimation of the polynomial coefficients is discussed in Appendix A.2.



(a) (b)

Figure 3.6. (a) Histogram of the linear activation to the sigmoid at an hidden node and the corresponding normal density function with variance of 4.9. (b) The sigmoid function $\phi(s+b)$ with bias $b=-3$, and its polynomial approximation for orders P=1, 3, and 5.

Figure 3.6 (a) shows the histogram of the input (excluding the bias) to the sigmoid function at a hidden node, and is obtained on the training data. The same figure also shows the normal density function that is used to model it. Fig 3.6 (b) shows the polynomial fit of order 1, 3, and 5 obtained using the mean square error criterion. With increasing order, the polynomial function gets closer to the sigmoid function. As the hidden bias is incorporated in the polynomial expansion, the estimated coefficients are different for each hidden node.

3.3.5 The Algorithm

In derivation of the Volterra kernels, we assumed for simplicity that the input/output of the system is continuous-time. However, in practice, the system under analysis is a discrete-time system. The discrete-time counterpart of the Volterra synthesis equation (3.14) can be written as

$$y^j(n) = g_0^j + \sum_{k_1=1}^{K} \sum_{m_1} g_{k_1}^j(m_1)\, x_{k_1}(n-m_1) + \sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \sum_{m_1} \sum_{m_2} g_{k_1 k_2}^j(m_1,m_2)\, x_{k_1}(n-m_1)\, x_{k_2}(n-m_2) + \ldots \tag{3.33}$$

The zeroth order discrete-time Volterra kernel is the same as its continuous-time counterpart (3.19). The first order Volterra kernels are expressed in terms of the discrete-time form of the impulse response function of the filter bank as

$$g^j_{k_1}(m_1) = \sum_{i=1}^{M} c_i^j \; a_{1,i} \sum_{l_1=1}^{L} \hat{w}^i_{k_1 l_1} h_{l_1}(m_1) \tag{3.34}$$

The second and higher order Volterra kernels can be expressed in the similar way replacing the continuous time impulse response function with its discrete time version.

$$g^j_{k_1 k_2}(m_1, m_2) = \sum_{i=1}^{M} c_i^j a_{2,i} \sum_{l_1=1}^{L} \sum_{l_2=1}^{L} \hat{w}^i_{k_1 l_1} \hat{w}^i_{k_2 l_2} h_{l_1}(m_1)\, h_{l_2}(m_2) \tag{3.35}$$

The application of Volterra series in the analysis of MLP based acoustic modeling is summarized in the following steps.

1. Identify the subsystem in the combined (feature extraction and MLP classifier) system for analysis. This is decided by two factors (a) if the system under analysis can be rearranged to any of the forms schematized in Figure 3.4 (a, b, c) without affecting its functionality and (b) if the input to the subsystem is easily interpretable. In most cases, the input to the subsystem are $\log$ energies in the auditory filter banks.

2. Fix the order of the power series approximation at the hidden layer P, and estimate its coefficients $a_0, a_1, \ldots a_P$ using the full covariance matrix of the features, the weight matrix $\mathbf{W}$ connecting the input to the hidden layer, and the bias vector $\mathbf{b}_h$ at the hidden layer of the MLP as discussed in Section 3.3.4 and Appendix A.2.

3. Replace the sigmoidal activation function with the power series approximation of different orders and analyze the effect of the approximation on the phoneme classification accuracies.

4. Address feature variance normalization by modifying the weights connecting the input to the hidden layer using (3.24) or (3.30), and the polynomial coefficients estimated in Step-2.

5. Compute the Volterra kernels using (3.34)-(3.35). The kernels are functions of the impulse response function of the FIR filter bank, the statistics (mean and covariance) of the features, and the trained parameters of the MLP.

6. Analyze the Volterra kernels to discover the knowledge learned by the MLP. In this work, we demonstrate this by analyzing the first order Volterra kernels (the linear part) of the system.

3.3.6 Special Cases of the Proposed Framework

Time-delay neural network

Analyzing a time-delay neural network (Waibel *et al.*, 1989) using Volterra series as discussed in (Hakim *et al.*, 1991; Wray and Green, 1994) forms a special case of the framework schematized in Figure 3.5. Suppose that d denotes the maximum delay at the input of the neural network, then in the proposed framework, we consider the FIR filter bank with $L = d + 1$ filters, each having an impulse response function given by

$$h_l(m) = \delta\left(m - l + 1\right), \quad \text{with} \qquad l = 1, 2, \ldots L \quad \text{and} \qquad m = 0, 1, \ldots d$$

Temporal context on the MLP

The time-delay neural network is a causal system, where the output of the MLP depends on the present and past inputs. However, in speech recognition the MLP is often trained with a temporal context spanning both past and future (Bourlard and Morgan, 1994). Suppose that the MLP is trained with a temporal context of $2d + 1$ frames, then in the proposed framework, we consider the FIR filter bank with $L = 2d + 1$ filters with impulse response function given by

$$h_l(m) = \delta\left(m + l - \frac{L+1}{2}\right), \quad \text{with} \quad l = 1, 2 \ldots L \quad \text{and} \qquad m = -d, \ldots 0, \ldots d \tag{3.36}$$

MLP as a standalone system

In scenarios where the MLP is to be analyzed as a standalone system without the FIR filter bank, then $L = 1$ and $h_1(m) = \delta(m)$. In this case, the system is a static nonlinear system *i.e.,* without memory.

3.4 Application of Volterra Series

In this section, we demonstrate the Volterra series representation of the MLP trained using mel filter bank energies (MFBE), multi-resolution relative spectra (MRASTA) features (Hermansky and Fousek, 2005) and the standard MFCC features (Davis and Mermelstein, 1980) with dynamic (delta and delta-delta) coefficients and a temporal context of 90 ms. We empirically demonstrate the convergence of Volterra series for the system and provide some qualitative examples on the spectro-temporal properties learned by the system for certain phonemes. The objective here is to demonstrate the application of the proposed framework. The application of Volterra series for a detailed analysis of an MLP classifier is presented in Chapter 4 in the context of the hierarchical system.

Experiments are performed on the TIMIT database (Fisher *et al.*, 1986). The database, which is hand-labeled using 61 symbols is mapped to a standard set of 39 phonemes (Lee and Hon, 1989) with an additional garbage class. The number of speakers and the size of the train, cross-validation, and test sets and a description on the mapping of phonemes is given in Section 4.3.

The phonetic class-conditional probabilities of phonemes estimated by the MLP are evaluated by performing speaker independent phoneme classification *i.e.,* isolated phoneme recognition experiments. Classification is performed using the hybrid HMM/MLP approach (Bourlard and Morgan, 1994). A phoneme is represented by a three-state left-to-right HMM, thereby enforcing a minimum duration of 30 ms. The emission score in each of the three states of the phoneme is the same, and is derived from the associated output of the MLP.

3.4.1 Volterra Analysis: MFBE-MLP System

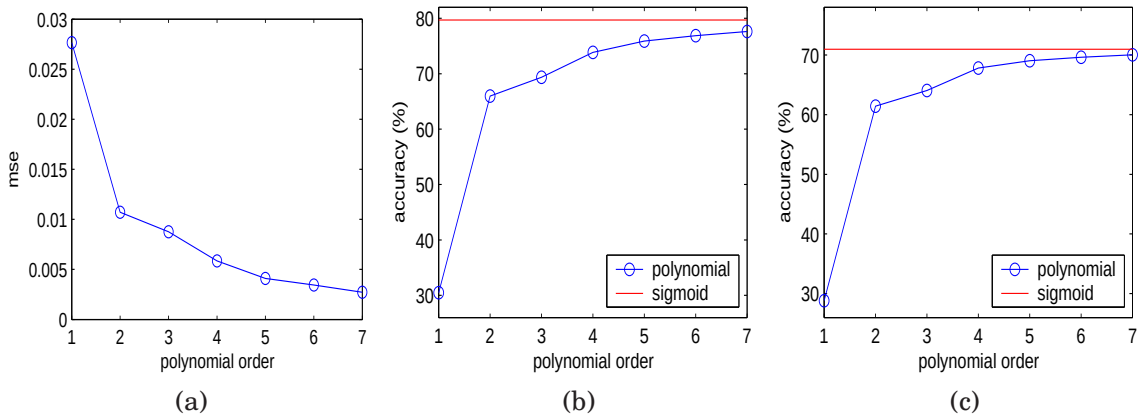
Mel frequency band energies (MFBE) can also be used as features in MLP based acoustic modeling. Application of Volterra series to this system is straightforward as the system is as shown in Figure 3.5. The input to the system under analysis is $\log$ energies in each of the $K = 26$ channels obtained by mel critical band integration. The output of the system is the linear activation values for each of the $N = 40$ phonemes at the output of the MLP.

FIR filter bank

The MFBE energies are presented to the MLP, without any processing, but with a temporal context of around 170 ms.[7] The creation of the temporal context can be achieved using a filter bank comprising of $L = 17$ FIR filters, whose impulse response functions are time-shifted Kronecker delta functions in the form given by (3.36)

Application of Volterra series

The architecture of the MLP under analysis is $442(K.L) \times 1000(M) \times 40(N)$. The sigmoid nonlinear function at the hidden layer of the MLP is approximated as a polynomial function using the mean square error estimation discussed in Section 3.3.4. In Figure 3.7 (a), we plot the mean square error at the hidden layer as a function of the polynomial order. It can be seen that the error decreases monotonically with the polynomial order. As the approximation error at the hidden layer decreases with the polynomial order, there is a corresponding monotonic increase in the phoneme classification accuracy on both training as well as test sets as shown Figures 3.7 (b, c) respectively.



(a) (b) (c)

Figure 3.7. (a) The average mean square error between the sigmoid function and its polynomial approximation at the hidden layer of the MLP trained on MFBE features. (b) phoneme classification accuracy on the train set as a function of the polynomial order used to approximate the sigmoidal nonlinear function. Horizontal lines indicate the accuracy obtained using the sigmoidal function. (c) A similar plot on the test set.

In the calculation of Volterra kernels, the order of the polynomial approximation of the sigmoidal function is first fixed as discussed in Section 3.3.5. Furthermore, as the system is feedforward the Volterra series is finite and therefore necessarily converges. The above plots demonstrate the con-

[7] We take 170 ms to be consistent with the MFCC feature extraction, where a 90 ms context on the concatenated cepstral coefficients (static + dynamic) is equivalent to 170 ms context on the mel filter bank energies.

vergence of the Volterra series to the standard MLP representation as the order of the polynomial that approximates the sigmoidal function tends to infinity.

In the Volterra analysis of a three layered MLP, the only nonlinear function is at the hidden layer, which is typically a sigmoid or hyperbolic tangent. If the power series approximation of the nonlinear function is convergent, then the Volterra series is also convergent (the other conditions being that the filter coefficients and weights of the MLP are bounded, which is always the case). As the power series approximation is done in the operating region of the hidden nonlinearity as shown in Figure 3.6 (a), the Weierstrass theorem ensures the convergence of the series. This is reflected in the minimization of the mean square error as shown in Figure 3.7 (a). The convergence of Volterra series can be seen in the monotonic increase in the phoneme classification accuracy in the same figure. In practice, the accuracy on the test set can diverge if the statistics of the features differ drastically from those of the train set.

Interpretation of Volterra kernels

It can be seen from (3.33) that the first order Volterra kernel $g_k(m)$ is a two-dimensional linear impulse response function. It is a function of both time m and frequency m. It is evident from (3.34) that the time support of the Volterra kernel is same as that of the FIR filters. The frequency axis corresponds to the component of the input representation to the filter bank. Figure 3.8 shows the first order Volterra kernel for phonemes /iy/ (*e.g.,* f**ee**l) and /eh/ (*e.g.,* f**e**ll). In this particular case, the frequency axis corresponds to the center frequency of the 26 mel auditory filters, and the temporal support of the kernels is 170 ms, which is same as the temporal context applied on the mel filter bank energy features.

It can be seen from the plots that for the vowel /iy/, the system has learned to emphasize a lower frequency region when compared to the vowel /eh/. Interpreting this two dimensional kernel on paper can be difficult. To get a clearer picture of the frequency regions learned by the system, in Figure 3.9 we plot the contribution of each critical band in the Volterra kernel for both these vowels by summing up the kernels along the time as $g_k^j = \sum_m g_k^j(m)$.

It can be seen from Figure 3.9 that in the case of phoneme /iy/, the system has learned to emphasize 187-374 Hz frequency band which corresponds to its first formant. On the other hand, for the vowel of /eh/, the system has learned to emphasize slightly higher frequency region of 374-685 Hz,

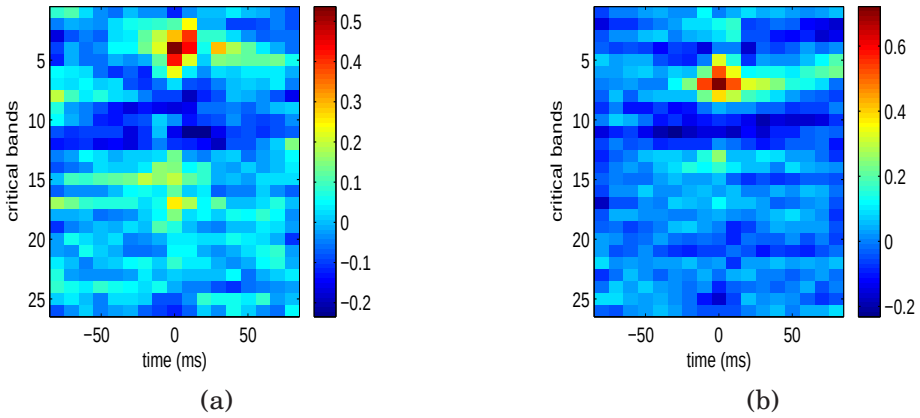


Figure 3.8. (a) Linear Volterra kernel of the trained system for phonemes /iy/ (e.g. f**ee**l). The x-axis corresponds to the time (170 ms) and the y-axis corresponds to the center frequency of the 26 mel filter banks. (b) A similar plot for the phoneme /eh/ (e.g. f**e**ll).

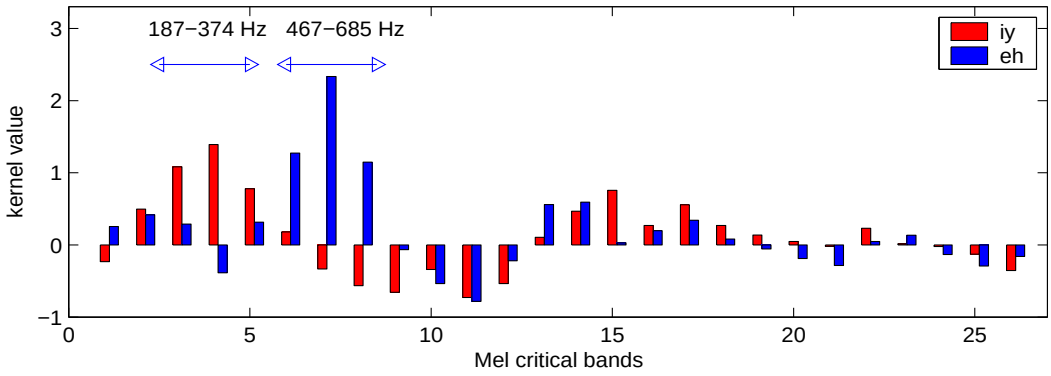


Figure 3.9. Important frequency regions for the vowels /iy/ (*e.g.,* f**ee**l) and /eh/ (*e.g.,* fell) obtained from the linear Volterra kernels for TIMIT.

which contains its first formant. It can also be seen that the difference between first and second formant for the front vowel /iy/ is higher compared to the mid-vowel /eh/. This is consistent with previous studies in acoustic phonetics.

3.4.2 Volterra Analysis : MRASTA-MLP System

MRASTA features (Hermansky and Fousek, 2005) are obtained by filtering the Bark auditory spectrum (log-energies in the Bark critical bands) along the time axis using a bank of multi-resolution band-pass filters. For wideband speech such as TIMIT, $K = 19$ critical bands are typically used. The output of the system under analysis are the linear activation values corresponding to $N = 40$ output phonetic classes.

FIR filter bank

Our implementation of the MRASTA filter bank consists of $L = 14$ FIR filters divided into two sets: (a) seven filters with an impulse response function of the shape of the first order derivative of a Gaussian function, and (b) seven filters with impulse response function of the shape of the second order derivative of a Gaussian function. The impulse response functions are given by

$$\begin{aligned} h_{2l-1}(m) &= f_{1d}(t = 10m,\ \sigma_l) \\ h_{2l}(m) &= f_{2d}(t = 10m,\ \sigma_l), \quad \text{with} \quad l = 1, 2, \ldots 7 \quad \text{and} \quad m = -35, \ldots 35 \end{aligned} \tag{3.37}$$

where $f_{1d}(t, \sigma_l)$ and $f_{2d}(t,\ \sigma_l)$ denote the first and second order derivatives of a Gaussian function with a standard deviation σ_l.[8] The standard deviation of the Gaussian function is varied between 8 ms and 90 ms, and it controls the time-resolution of the filters. The functions are sampled in steps of 10 ms, and span a duration of 700 ms. Figure 2.6 (a and b) show the impulse response functions in the continuous-time domain (before sampling).

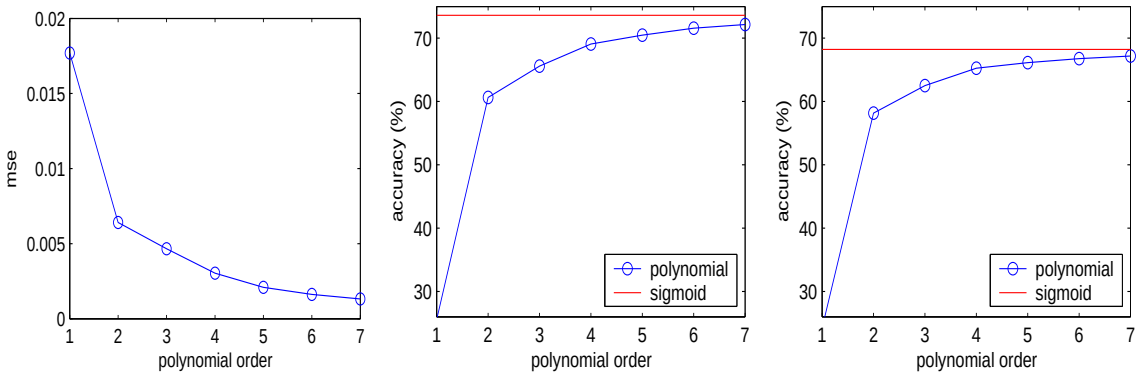
Application of Volterra Series

The architecture of the MLP under analysis is $266(K.L) \times 1000(M) \times 40(N)$. The sigmoid nonlinearity at the hidden layer is approximated as a power series using the mean-square error criterion. Figure 3.10 (a) is a plot of the average mean square error at the hidden layer as a function of the order of the polynomial approximation. Figure 3.10 (b and c) show the monotonic increase in the phoneme classification accuracies with the increase in the order of the polynomial on the train and test sets respectively.

Interpretation of Linear Kernels

The first order Volterra kernel for each of the phonemes $j = 1, 2, \ldots N$ is denoted by $g_k^j(m)$, and is a function of both time m and the auditory filter bank index k. As the input $x_k(m)$ in the Volterra synthesis equation (3.33) is output of the auditory filter bank, k corresponds to a frequency range. For example, $k = 2$ on the Bark frequency scale corresponds to a frequency of 99-200 Hz. It can

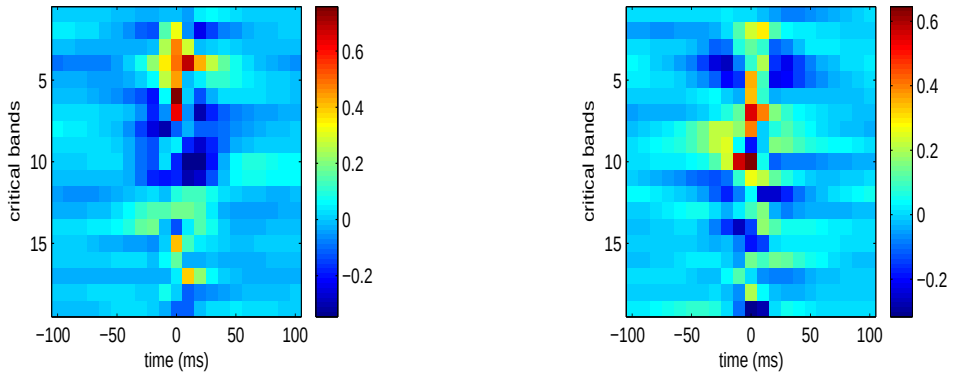
[8] To be precise, $f(t, \sigma) = exp(-t^2/2\sigma^2)$ is the Gaussian function, and if $f^{(1)}(t, \sigma)$ and $f^{(2)}(t, \sigma)$ respectively denote its first and second order derivatives with respect to t, then $f_{1d}(t, \sigma) = \frac{f^{(1)}(t,\sigma)}{\max_t |f^{(1)}(t,\sigma)|}$ and $f_{2d}(t, \sigma) = \frac{f^{(2)}(t,\sigma)}{\max_t |f^{(2)}(t,\sigma)|}$



(a) (b) (c)

Figure 3.10. (a) The average mean square error between the sigmoid function and its polynomial approximation at the hidden layer of the MLP trained on MRASTA features. (b) phoneme classification accuracy on the train set as a function of the polynomial order used to approximate the sigmoidal nonlinear function. Horizontal lines indicate the accuracy obtained using the sigmoidal function. (c) A similar plot on the test set.

be seen from (3.34) that the first order Volterra kernels are linear combinations of the impulse response functions of the FIR filters. As a consequence, the temporal support of the linear Volterra kernels is same as that of the FIR filters. In this case, the temporal support spans 700 ms.



(a) (b)

Figure 3.11. (a) Linear Volterra kernel for the phoneme /iy/ (*e.g.,* d**ee**d) on TIMIT database. (b) A similar plot for the phoneme /ao/ (*e.g.,* d**o**g).

Figure 3.11 (a) is a plot of the first order Volterra kernel for the phoneme /iy/ (*e.g.,* d**ee**d). The kernel corresponds to the spectro-temporal pattern learned by the MLP for the phoneme /iy/ to be discriminated from other phonemes. Figure 3.11 (b) is the first order Volterra kernel for the phoneme /ao/ (d**o**g). The kernels are plotted for a temporal context of 200 ms around the center for clarity as most of the activity is concentrated in that region.

To see the differences between the kernels for the vowels more clearly, in Figure 3.12, we show

the contribution of each critical band in the Volterra kernel by summing up the kernels along the time. It can be seen that for the phoneme /iy/, there is a concentration of energy in the 200-500 Hz range, followed by another in 2111-3500 Hz range. It can be recalled that these are, in fact, the first and second formants for the phoneme /iy/. Moreover, due to the discriminative training, the system has also learned to give negative emphasis in the 692-2111 Hz frequency region. In contrast, for the phoneme /ao/, the system has learned to emphasize the 692-1492 Hz frequency region. Previous studies in acoustic phonetics indicate that for the rounded vowel /ao/, the mean of the first formant is around 600 Hz and the second formant, it is around 900 Hz (Xuedong *et al.*, 2001). It can be seen that the system is not able to differentiate between the two formants and we see an emphasize in the frequency region between 692-1492 Hz.

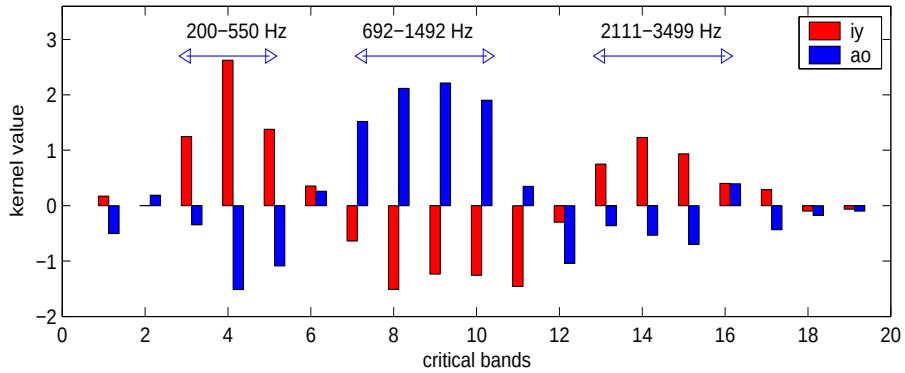


Figure 3.12. Important frequency regions for the /iy/ (*e.g.,* d**ee**d) and /ow/ (*e.g.,* d**o**g) obtained from the linear Volterra kernels for TIMIT.

3.4.3 Volterra Analysis: MFCC-MLP System

In this section, we demonstrate the application of Volterra series to analyze an MLP, which is trained on the standard MFCC features (Davis and Mermelstein, 1980). The input to the system under analysis are the $\log$ energies from each of the $K = 26$ mel auditory filter banks, and the output of the system is the linear activation values of the $N = 40$ output phonetic classes. This system is in the form shown in Figure 3.4 (c). The system for Volterra analysis includes the discrete cosine transformation matrix, followed by the FIR filter bank required to compute the dynamic cepstral coefficients and a temporal context of 90 ms.

FIR filter bank

Figure 2.5 (b and c) are the FIR filters that are typically used in HTK toolkit for computing the delta and delta-delta cepstral parameters. The static cepstral coefficients can be viewed as the output of an FIR filter with a Kronecker delta $\delta(m)$ impulse response function. In other words, a 39 dimensional concatenated feature vector is obtained by filtering the 13 dimensional static cepstral feature vector using 3 FIR filters. The creation of a temporal context of 90 ms can also be achieved using 9 filters with impulse response functions given by (3.36). Both these operations can be implemented using a single filter bank comprising of 27 filters.[9]

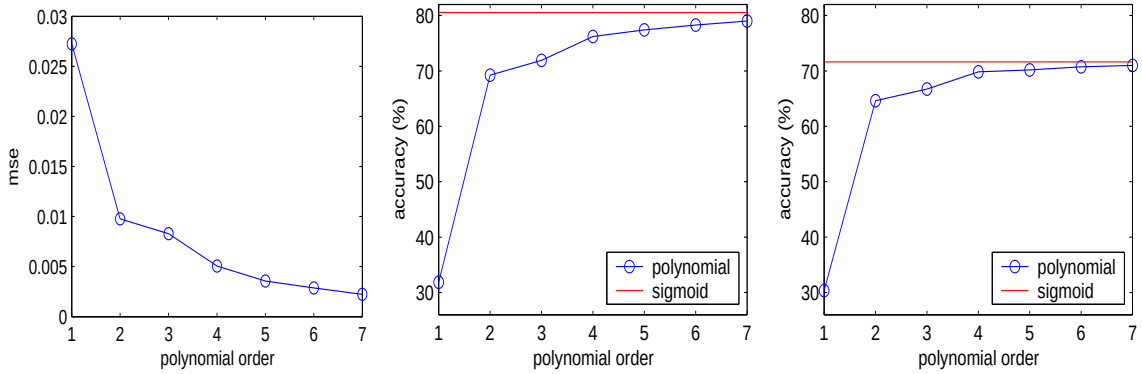


Figure 3.13. (a) The average mean square error between the sigmoid function and its polynomial approximation at the hidden layer of the MLP trained on MFCC features. (b) phoneme classification accuracy on the train set as a function of the polynomial order used to approximate the sigmoidal nonlinear function. Horizontal lines indicate the accuracy obtained using the sigmoidal function. (c) A similar plot on the test set.

The variance normalization and the DCT transformation matrix is incorporated into the weight matrix of the MLP connecting the input layer to the hidden layer as given by (3.30). Figure 3.13 shows the monotonic decrease in the error of polynomial approximation of the sigmoidal function and the monotonic increase in the phoneme classification accuracies with increase in the order of the polynomial. The Volterra kernels are derived in the same way as discussed in previous sections.

Interpretation of Volterra kernels

In Figure 3.14, we plot the first order Volterra kernels for the affricate phonemes /ch/ (e.g. **ch**urch) and /jh/ (e.g. **j**udge). The affricate /ch/ is a combination of two phonemes - the unvoiced stop con-

[9]The impulse response of a cascade of two LTI systems with impulse responses $h_a(t)$ and $h_b(t)$ is given by the convolution $h_{ab}(t) = h_a(t) * h_b(t)$.

sonant /t/ and the unvoiced fricative /sh/. On the other hand, the affricate /jh/ is a combination of voiced stop consonant /d/ and the voiced fricative /zh/ (Xuedong *et al.*, 2001). It can be seen from the plot that for both these phonemes, there is an area of frication between 2000-3500 Hz. This information is important to distinguish these affricates from other phonemes such as vowels. The two affricates are discriminated from each other depending on their low frequency characteristics. It can be seen that for the voiced /jh/, there is a concentration of energy in the low frequency region 125-280 Hz, whereas for the unvoiced affricate /ch/, the emphasis is in the relatively higher frequency region of 778-1089 Hz.

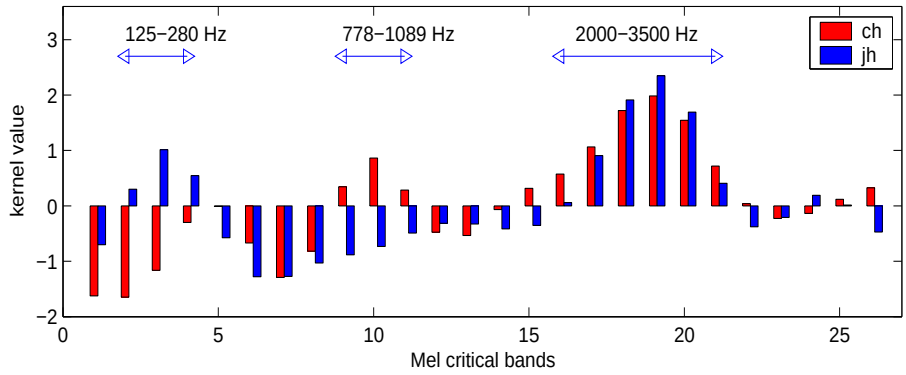


Figure 3.14. A comparison of emphasis/deemphasis of different frequency regions for affricates /ch/ (*e.g.,* **ch**urch) and /jh/ (*e.g.,* **j**udge).

3.5 Wiener Analysis of MRASTA-MLP System

In this section, we discuss the application of Wiener series in the analysis of MLP classifier trained to estimate the phonetic class-conditional probabilities. The objective of this study is twofold. Firstly, it is to re-validate the shape of the kernels obtained in the previous section using Volterra analysis. Secondly, although we showed the calculation of Volterra kernels for a three layered MLP, this method could be useful when analytical calculation of Volterra kernels becomes complicated due to, for example, the presence of more than one hidden layer.

In Section 3.2.4, we showed the Wiener series representation for a single-input single-output system. The Wiener kernels are estimated by presenting white Gaussian noise as stimulus to the system under analysis and then cross-correlating its response to the stimulus as given by (3.10). The extension of Wiener theory to multi-input, multi-output systems is straightforward as dis-

cussed in (Marmarelis and Naka, 1974).

We now discuss the Wiener analysis of the MRASTA-MLP system. As in Section 3.4.2, the input to the system under analysis represent the $\log$ energies in the 19 critical bands. The output of the system represent the linear activation values before the softmax activation function. Let $x_k(n), k = 1, 2, \ldots K; n = 1, 2, \ldots N$ denote the white Gaussian noise generated, where K is the number of critical bands (here 19), and N the number of samples generated. White noise is presented at the input of the MLP as if it were a single utterance obtained from real speech. Let $y_j(n)$ denote the linear output of the MLP corresponding to the phoneme j for the time instant n. The zeroth and first order Wiener kernels are obtained by cross-correlating the stimulus and response as

$$h_0^j = \frac{1}{N} \sum_{n=1}^{N} y^j(n) \tag{3.38}$$

$$h_k^j(m) = \frac{1}{N\sigma_k^2} \sum_{n=1}^{N} \left(y^j(n) - h_0^j \right) x_k(n-m) \tag{3.39}$$

where σ_k^2 is the variance of the noise corresponding to the critical band k. In our experiments, the noise is generated with zero mean and variance corresponding to the $\log$ energies in the critical bands for real speech, and this is estimated on the training data.

The basic idea in correlation based kernel estimation is to present all possible input combinations as the stimulus to the system, and to measure the response of the system. The first order kernel for a phoneme j can be interpreted as the average time-reversed pattern in the input domain that activates the particular output unit.

In Figure 3.15 (a), we compare a time-slice in the first order Volterra and Wiener kernels of the phoneme /iy/ corresponding to the critical band 4 (307-423 Hz) for the MRASTA-MLP system. We chose this particular critical band because from the Volterra analysis (Figures 3.11 (a) and 3.12) it is clear that for the phoneme /iy/, the system has learned to emphasize this frequency region. It can be seen from the figure that the Volterra and Wiener kernels are similar in shape with a correlation coefficient of 0.96. Figure 3.15 (b) is a plot of the kernels for the phoneme /ao/ in the same frequency region.

For further illustration, in Figure 3.16, we plot the time-slice in the first order Volterra and Wiener kernels for the phonemes /iy/ and /ao/ for a frequency region 1035-1247. This is the fre-

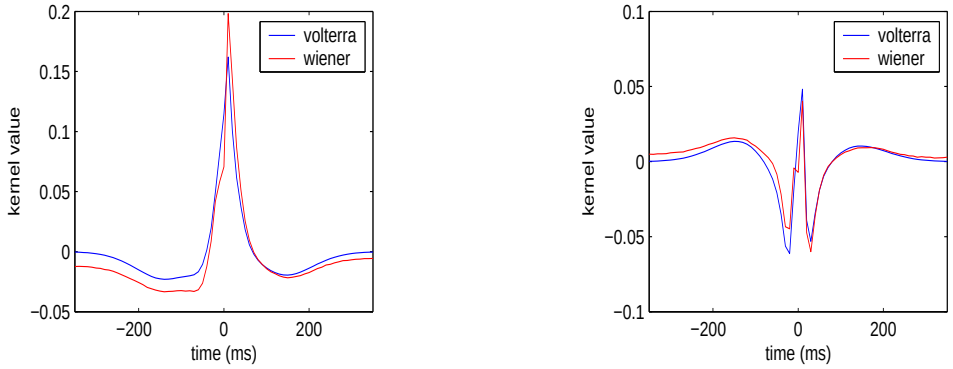


Figure 3.15. (a) Comparison of a time slice of Volterra and Wiener kernels of the phoneme /iy/ corresponding to the critical band 4 with a frequency range of 307-423 Hz. The correlation coefficient between the kernels is 0.96. (b) A similar plot for the phoneme /ao/, where the correlation coefficient between the kernels is 0.95.

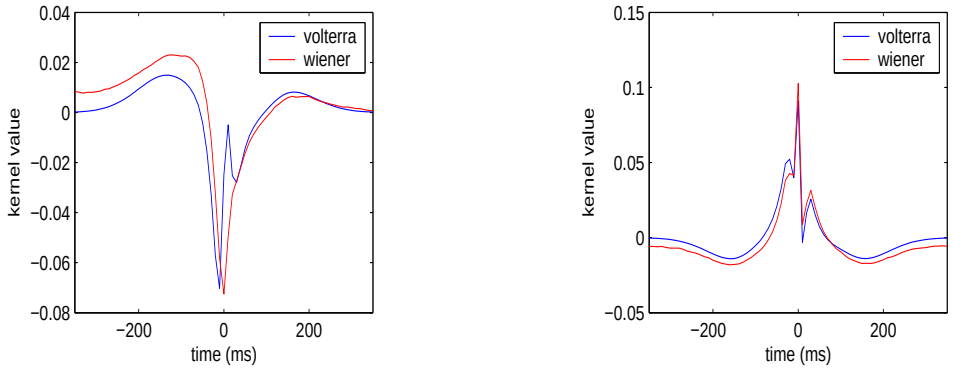


Figure 3.16. (a) Comparison of a time slice of Volterra and Wiener kernels of the phoneme /iy/ corresponding to the critical band 8 with a frequency range of 1035-1247 Hz. The correlation coefficient between the kernels is 0.81. (b) A similar plot for the phoneme /ao/, where the correlation coefficient between the kernels is 0.98.

quency region that the system has learned to emphasize in order to classify the phoneme /ao/ as shown in Figure 3.11 (b) and Figure 3.12.

It can be recalled that the Volterra kernel represents the total linear part of the nonlinear time-invariant system as the functionals are homogeneous. On the other hand, every Wiener functional of order greater than one consists of a varying number of lower order Volterra functionals (3.10), which are called derived Wiener kernels (Franz and Scholkopf, 2003). Hence, the first order Wiener kernel can only approximately reveal the linear part of the nonlinear system. Nevertheless, from Figures 3.15 and 3.16, it can be seen that the kernel shapes are similar.

To obtain further insights, we obtain an objective measure on the similarity between the first

order Volterra kernels, which are identified analytically and the first order Wiener kernels, which are estimated using cross-correlation based methods. In Figure 3.17 (a), we plot the histogram of the correlation coefficient between the Volterra and Wiener kernels. For the MRASTA system, the number of first order kernels is 40, corresponding to the number of output phonetic classes. Each kernel is a function of time (71 samples) and frequency corresponding to the 19 critical bands. We compute the correlation coefficient between the $19.40 = 760$ one-dimensional Volterra and Wiener kernels. In Figure 3.17 (a), we plot the histogram of the correlation coefficients. It can be seen that the majority of the kernels have a correlation coefficient greater than 0.95. However, there are a significant number of kernels with lower correlation coefficient.

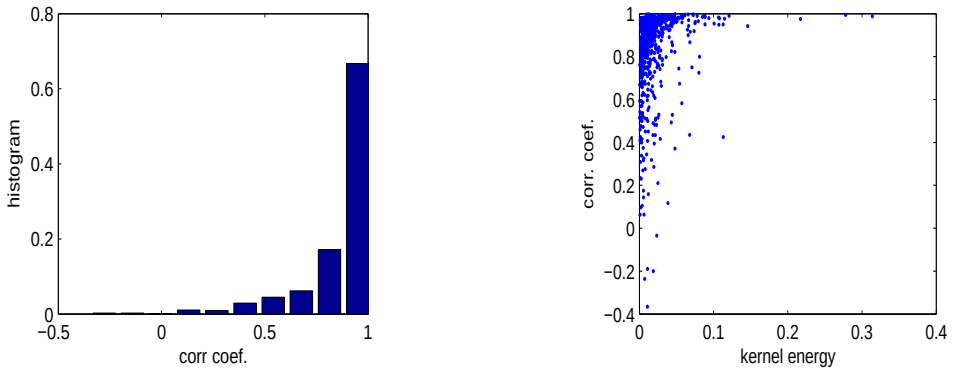
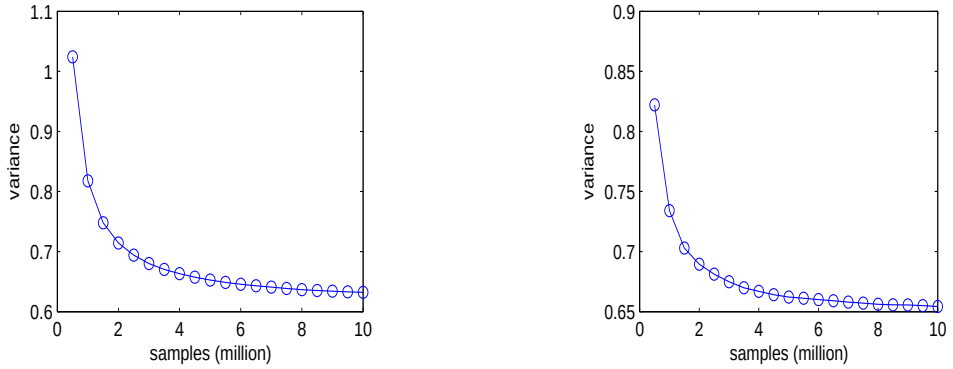


Figure 3.17. (a) The normalized histogram of the correlation coefficient between the Volterra kernel and Weiner kernel. (b) the scatter plot of the correlation coefficient as a function of the energy in the Volterra kernel.

To get a clearer picture of the kernels with low correlation coefficient, in Figure 3.17 (b), we show the scatter plot of the correlation coefficient between Volterra and Wiener kernels and the energy in Volterra kernel. It can be seen that the kernels which have a low correlation coefficient are the ones with lower energies, and low energy kernels are those which do not contribute significantly to the output activation values. In other words, Wiener analysis can be applied to understand the functionality of the system as the kernels that actually contribute to the output are reliably estimated.

It can be seen from (3.38) and (3.39) that Wiener kernels are actually average or mean patterns. To empirically demonstrate convergence, it is sufficient to show that the variance of the estimate decreases with the number of samples used for its estimation. In Figure 3.18, we plot the variance of the Wiener kernels shown in Figure 3.15 for the time instant $t = 0$. It can be seen that the

variance reduces steadily with the number of samples, but it is well known in the literature that the rate of convergence can be very slow.



(a) (b)

Figure 3.18. Variance of the estimates of the Wiener kernels shown as a function of the number of noise samples generated.

First order Volterra kernels

The first order Volterra kernels reveal the linear part of the nonlinear dynamic system under analysis. When analyzing MLPs trained on acoustic features, these kernels represent the average spectro-temporal patterns learned for each of the phonemes. The expression for the first order kernel (3.34) can be rewritten as

$$g_k^j(m) = \sum_{i=1}^{M} c_i^j \; g'_{(i,k)}(m), \text{ where} \tag{3.40}$$

$$g'_{(i,k)}(m) = \hat{a}_{1,i} \sum_{l=1}^{L} \hat{w}_{k,l}^i h_l(m) \tag{3.41}$$

It can be seen that the MLP learns a spectro-temporal pattern $g'_{(i,k)}(m)$ at each hidden node i, which is shared across all phonemes. Here, m and k denotes the time and frequency axis respectively. The first order Volterra kernel for a particular phoneme is a linear combination of spectro-temporal patterns at the hidden layer as given by (3.40). Depending on the phoneme, spectro-temporal patterns at the hidden layer are weighted according to the weight matrix connecting the hidden and output layers of the MLP to obtain the final kernel. This clearly brings out the similarity of the MLP with semicontinuous density modeling discussed in Section 2.4.2.

Best practices in calculation of Volterra Kernels

When identifying the Volterra kernels using data, it is a good practice (for mathematical simplicity as well as reasons of stability) to first identify the constant term in the system response and then model the remaining part of the response as increments with respect to this constant. In this way, the Volterra series starts with order 1 and not 0 (Hasler, 2010). In this chapter, the Volterra kernels are calculated analytically and hence this problem does not arise. However, in the Wiener series formulation, the above approach is followed as the kernels are identified incrementally as given by (3.10). It is also a good practice to consider the input signal as zero mean as this can greatly simplify the calculation of Volterra kernels in certain cases. For example, in the calculation of Volterra kernels where the input features to the MLP are normalized as discussed in Section 3.3.2.

3.6 Summary and Conclusion

The main objective of this work was to provide a framework to apply Volterra series to analyze MLP based phoneme posterior probability estimation. We include a part of the feature extraction (LTI system following the auditory analysis) in the analysis framework so that the Volterra kernels can be interpreted as spectro-temporal patterns. We showed the calculation of the Volterra kernels for the following three systems (a) an FIR filter bank followed by a three layered MLP (b) the features to the MLP are normalized to zero mean and unit variance, and (c) a linear transformation matrix precedes the FIR filter bank. Furthermore, we discussed the approximation of the sigmoidal function at the hidden layer of the MLP as a power series and empirically demonstrated the convergence of Volterra series.

In this chapter, we demonstrated the application of the proposed framework in the analysis of the MLPs trained on mel filter bank energies, MRASTA features and MFCC features. It has been observed (for example, from Figure 3.13) that as the order of the polynomial approximation is increased, the recognition accuracies obtained using truncated Volterra series approaches to the recognition accuracy obtained by the direct evaluation of the MLP. The first three Volterra kernels (constant, linear and quadratic) can reveal most of the information learned by the system. In this respect, we believe the proposed analysis framework can be useful towards better understanding of the trained systems. The detailed analysis of an MLP trained using posterior features is presented in the following chapter.

Chapter 4

MLP Based Hierarchical System

4.1 Introduction

So far we have discussed MLP based acoustic modeling and its application in speech recognition. We briefly summarize the main points here. A well trained MLP classifier estimates the posterior probabilities of phonemes conditioned on the input acoustic features. The estimated phonetic class-conditional probabilities are typically used as local state emission scores or as features in HMM based speech recognition. As phoneme posterior probabilities are also used as local representation of speech in the same way as standard acoustic features, they are commonly referred to as posterior features.

In the posterior feature space, each dimension corresponds to a phoneme. The posterior feature vector at a particular time instant is a point in the posterior feature space, representing the instantaneous soft-decision on the underlying phoneme. It carries useful information such as the probability mass assigned to competing phonemes. The sequence of posterior feature vectors is a trajectory in the posterior feature space, and it can provide additional contextual information such as the evolution of the posterior features within a phoneme (sub-phonemic transition) as well as in its transition to and from neighboring (sub-lexical transition) phonemes.

This research presented in this chapter is based on the premise that the sub-phonemic and sub-lexical contextual information can be exploited in the estimation of more accurate phonetic class-conditional probabilities. To this end, we investigate a hierarchical system, where a second

MLP classifier is trained on the posterior features with a temporal context spanning about 150-230 ms, which corresponds to roughly three phonemes.

The rest of the chapter is organized as follows: In Section 4.2, we describe the MLP based hierarchical system and discuss its similarities/differences with previous works in the literature. In Section 4.3, we describe the experimental setup and the results. In Section 4.4, we discuss the application of Volterra series in the analysis of the second stage of the hierarchical and interpret its linear Volterra kernels in terms of phonetic-temporal patterns. In Section 4.5, we analyze some of the favorable properties of the posterior features, which makes the hierarchical system effective. In Section 4.6, we discuss some of the less explored facets of the hierarchical approach.

4.2 Hierarchical Posterior Estimation

Figure 4.1 is a block schematic of the proposed hierarchical architecture for estimating the phonetic class-conditional probabilities. The first MLP is trained in the conventional way using standard acoustic features. The second MLP is trained using posterior features estimated by the first MLP classifier, taken with a temporal context of around 150-230 ms. The phonetic class-conditional probabilities estimated by the second MLP are used in the same way as those estimated by the conventional single MLP based approach.

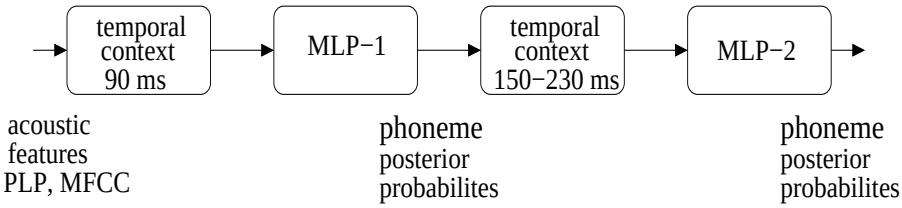


Figure 4.1. Estimation of posterior probabilities of phonemes using a hierarchy of two MLPs. The second MLP is trained using the posterior probabilities of phonemes estimated by the first MLP with a longer temporal context.

In Section 2.5.4, we discussed that the performance of ASR systems using MLP based acoustic modeling can be improved using three broad strategies, namely (a) better features (b) better modeling and (c) finer output classes. With respect to the second classifier in the hierarchical system, the proposed approach can be viewed as using better features.

4.2.1 Motivation

An MLP trained on acoustic features gives a frame-level phoneme classification accuracy of around 60-70%. The errors in classification can be mainly attributed to the limitations in feature extraction and in the modeling. Analysis of the associated phonetic confusion matrices have shown that there exists a definite pattern in classification. For example, if the phoneme /iy/ (*e.g.,* b**ea**t) is misclassified, then it is more likely that vowels such as /ih/ (*e.g.,* b**i**t) or /eh/ (*e.g.,* b**e**t) are assigned a higher probability mass. This information in the distribution of the probability values could be exploited to correct the output of the MLP classifier.

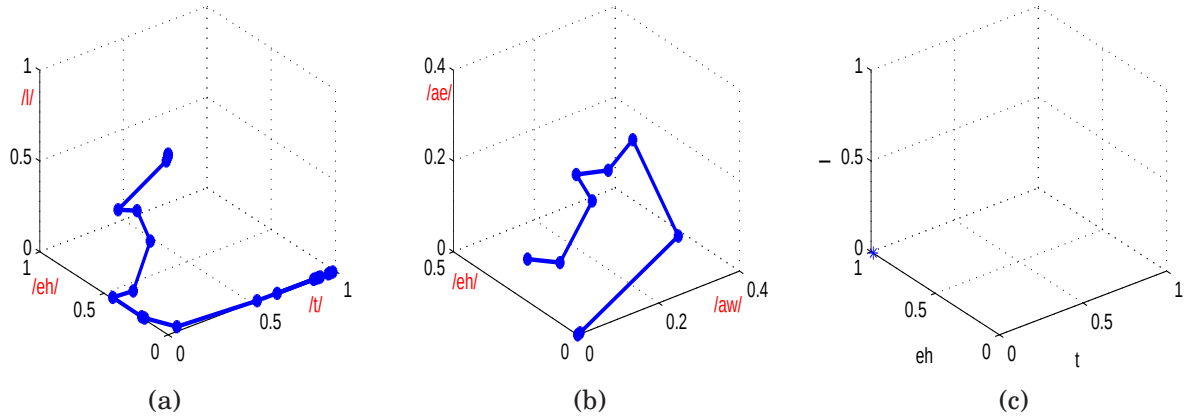


Figure 4.2. (a) A 210 ms trajectory of the posterior features showing the underlying phoneme sequence is /t/ /eh/ /l/, which is a part of the utterance "artificial intelligence". (b) A 90 ms trajectory around the vowel /eh/. (c) Enhanced posterior probability estimate at the center of the vowel /eh/.

In Figure 3.2 (d), we plotted the posterior probabilities of phonemes estimated by the MLP as a function of time for the utterance "*artificial intelligence*" in the TIMIT database. The intensity of the color is proportional to the estimated probabilities. To get a clearer picture, in Figure 4.2 (a), we plot the posterior features in the time interval between 640 ms and 840 ms, where the underlying correct phoneme sequence is /t/ /eh/ /l/. The axes of the three dimensional plot correspond to the phonemes /t/, /eh/ and /l/. The remaining dimensions are not plotted for the sake of clarity. The trajectory starts roughly at the center of the phoneme /t/ and then transits to /eh/, followed by a transition to /l/ and ends at its center.

At the center of phonemes /t/ and /l/, the MLP has assigned a probability mass which is close to one, which indicates a perfect classification. However, in the case of the vowel /eh/, the probability mass assigned is about 0.5, and the remaining mass is assigned to confusing vowels such as /ae/

and /aw/. This can be seen in Figure 4.2 (b), where a 90 ms trajectory around the center of the vowel /eh/ is plotted for the dimensions /eh/, /ae/, and /aw/. This figure clearly shows the sub-phonemic transitions in the vowel /eh/ in a 90 ms interval, while Figure 4.2 (a) shows the sub-lexical transitions in a 210 ms interval.

The posterior features have a simple (or sparse) representation in their feature space as the trajectory will mostly be along the surface of the N dimensional hypercube, where N denotes its dimensionality. Furthermore, it has been shown that the posterior features have lesser nonlinguistic variabilities such as speaker and environmental characteristics (Zhu *et al.*, 2004; Ikbal, 2004). Consequently, contextual information spanning time spans as long as 250 ms can be effectively modeled in the posterior feature space. Figure 4.2 (c) shows the posterior probability estimated by the second MLP at the center of the vowel /eh/, conditioned on the trajectory of posterior features. It can be seen that the enhanced estimate yields a perfect classification.

As the second MLP is trained using posterior features estimated by the first classifier with a certain temporal context, we can expect it to learn the phonetic-temporal patterns, mainly capturing the phonetic confusions at the output of the first classifier. However, as the MLP is a complex classifier with nonlinear activation functions, discovering the phonetic-temporal patterns learnt by the system for each phoneme is not straightforward. Moreover, as the MLP is trained using a discriminative criterion, these patterns cannot be simply derived from the confusion matrix of the first MLP classifier. In addition, confusion matrices do not capture any temporal information. To understand this information, one has to interpret the trained parameters (weights and biases) of the second MLP classifier. In this chapter, we address this issue by representing the second stage of the hierarchical system using Volterra series, thereby decomposing the trained nonlinear dynamic system into its linear, quadratic, and higher order parts. Furthermore, we analyze the linear part of the second MLP and interpret the phonetic-temporal patterns that are learned.

4.2.2 Notations and Formalism

The following notations are used throughout this chapter. $\mathbf{f_t}$ denotes the acoustic feature vector [1] at time t. A temporal context of $2d_1 + 1$ frames on the feature vector $\mathbf{f}_t$ is denoted by $\mathbf{f}_{t-d_1:t+d_1} = [\mathbf{f}'_{t-d_1}, \ldots \mathbf{f}'_t, \ldots \mathbf{f}'_{t+d_1}]'$. The first MLP classifier, denoted by Θ_{mlp1}, estimates the posterior probability

[1] All vectors are column vectors by default. Transpose is denoted by $'$

of each of the K phonetic classes $q_t = k, k = 1, 2, \ldots K$, conditioned on the acoustic features spanning $d_1 \approx 4$ frames around $\mathbf{f}_t$ as

$$x_k(t) = P\left(q_t = k \mid \mathbf{f}_{t-d_1:t+d_1},\ \Theta_{mlp1}\right),\ k = 1, 2, \ldots K \tag{4.1}$$

The estimated posterior probabilities at time t are represented in a vectorial form as $\mathbf{x}_t = [x_1(t), x_2(t), \ldots x_k(t), \ldots x_K(t)]'$, and a temporal context of $2d_2 + 1$ frames on the posterior feature vector is denoted by $\mathbf{x}_{t-d_2:t+d_2}$. The second MLP, denoted by Θ_{mlp2}, estimates the posterior probabilities of phonemes conditioned on a temporal context $d_2 \approx 11$ on the posterior features estimated by the first MLP as

$$z_k(t) = P\left(q_t = k \mid \mathbf{x}_{t-d_2:t+d_2},\ \Theta_{mlp2}\right),\ k = 1, 2, \ldots K \tag{4.2}$$

The output of the second MLP at time t is represented as $\mathbf{z}_t = [z_1(t), z_2(t), \ldots z_k(t), \ldots z_K(t)]'$. In later parts of this section, $\mathbf{f}_{1:T}$ and $\mathbf{x}_{1:T}$ denotes the entire sequence of acoustic and posterior feature vectors respectively, where T denotes the total number of frames in the utterance.

In practice, the input features to the MLP are normalized to zero mean and unit variance. Feature normalization ensures that the operating region on the hidden activation function is in the linear region, leading to a faster convergence of the back propagation training algorithm (LeCun *et al.*, 1998). In the case of the second MLP, as the features are posterior probabilities, mean and variance normalization is equivalent to taking scaled likelihoods as features (refer to Appendix A.3 for the proof). Hence, normalization of posterior features removes the effect of unigram phonetic class priors learned by the first MLP classifier. The priors are, however, again learned by the second MLP classifier.

4.2.3 Background

In this section, we review different approaches in MLP based acoustic modeling, that use hierarchical architectures to better model the temporal information, and contrast them with the hierarchical approach investigated in this chapter. In all the discussed works, the first stage of the hierarchy is an MLP. The second stage of the hierarchy includes classifiers such as MLP, HMM, recurrent neu-

ral network (RNN), or conditional random field (CRF). The reviewed works are categorized into the following groups (*G1* to *G6*), mainly based on the application of temporal context on the posterior features and the type of classifier at the second stage of the hierarchy.

G1: Classifier Combination

Hierarchical architecture of MLPs have been previously studied in the TRAPS (Hermansky and Sharma, 1999) and HATS (Chen *et al.*, 2001) systems. At the first stage of the hierarchical system, separate MLP classifiers are trained for each of the critical bands. Temporal information in the acoustic features is exploited by using the log critical band energies spanning over a period of about one second as input feature. At the second stage, an MLP is used to merge the outputs from the classifiers at the first stage of the hierarchy. In other words, the input to the second MLP classifier are the activations at the output (hidden in case of HATS) layer of the critical band specific MLPs, but without any temporal context. Independent processing of speech in subbands was originally inspired by Allen's interpretation (Allen, 1994) of Fletcher's work (Fletcher, 1995), indicating a similar mechanism in the human auditory system. Similar hierarchical architectures have also been studied in multiband ASR (Bourlard and Dupont, 1996; Tibrewala and Hermansky, 1997).

G2: Feature Combination

Multi-resolution relative spectra (Hermansky and Fousek, 2005) features are obtained by filtering the log critical band energies using a bank of multi-resolution bandpass filters. These features are typically used in Tandem based ASR systems. In more recent studies (Valente and Hermansky, 2008a,b), the multi-resolution filter bank is split into two groups - fast modulation filters (narrow bandwidth) and slow modulation filters (wider bandwidth) - and combined in a hierarchical fashion. At the first stage of the hierarchy, an MLP is trained with features obtained using fast modulation filters. The estimates of posterior probabilities from the first MLP (log + KLT), with a temporal context of 90 ms are appended to the features obtained using slow modulation filters, and used to train the second MLP classifier. ASR studies using this hierarchical system have shown to yield higher recognition accuracies. In this approach, the second MLP acts like a feature combiner.

G3: Hierarchy using HMM

Hierarchical structures have also been investigated in an attempt to integrate additional knowledge such as minimum duration of phonemes and transition probabilities between phonemes (Ketabdar *et al.*, 2006). This knowledge is incorporated into an HMM model Θ_{hmm}. The posterior probabilities of phonemes estimated by the MLP model Θ_{mlp1} are used as emission scores in the HMM states. The new estimates of posterior probabilities are derived from the state occupancy probabilities $P(q_t = k|\mathbf{f}_{1:T}, \Theta_{mlp1}, \Theta_{hmm})$ estimated using the forward-backward algorithm. The new estimates of the posterior probabilities are conditioned on the entire acoustic observation sequence $\mathbf{f}_{1:T}$.

G4: Hierarchy using RNN

Recurrent neural networks (RNN) can also estimate the phonetic class conditional probabilities (Robinson, 1994). In a prior work (Khan *et al.*, 2000), the hierarchical estimation of the phoneme posterior probabilities using RNN was investigated. The first stage of the hierarchical system consists of an MLP trained using the power spectrum of the speech. Its output units represent the articulatory features corresponding to the phonemes. In the second stage, an RNN model Θ_{rnn} is trained on the articulatory features estimated by the MLP. In this case, at time t, the RNN estimates the posterior probabilities of the phonemes $P(q_t = k|\mathbf{x}_{1:t}, \Theta_{rnn})$, conditioned on the present and all the previously observed articulatory feature vectors $\mathbf{x}_{1:t}$.

G5: Hierarchy using CRF

There is a growing interest in CRF based models, especially linear chains (with first order Markovian assumption) for reasons such as discriminative training, relaxed conditional independence assumption, and ability to jointly model features streams with different distributions (Abdel-Haleem, 2006). In more recent works, CRFs have been investigated for hierarchical estimation of phoneme posterior probabilities (Morris and Fosler-Lussier, 2008; Fosler-Lussier and Morris, 2008). At the first stage of the hierarchical system, an MLP estimates the posterior probabilities of phonemes using (4.1). In the second stage, the estimates of the posterior probabilities from the MLP $\mathbf{x}_{1:T}$ are used as features to the CRF model Θ_{crf}. The new estimates of the posterior probabilities of phonemes $P(q_t = k|\mathbf{x}_{1:T}, \Theta_{crf})$ are obtained using a framework similar to HMM based forward-

backward algorithm.

The main difference between the CRF based hierarchical system and HMM based hierarchical system, discussed in *G3*, is in the way the estimates of posterior probabilities from the MLP are used. In the HMM based system, the posterior probabilities of phonemes are used as local acoustic scores in the HMM states, whereas in the CRF based system, they are used as features.

G6: Hierarchy using MLP

In the proposed approach, the MLP at the second stage of the hierarchy yields a new estimate of posterior probabilities, conditioned on a window of the posterior features estimated by the first MLP, and the model Θ_{mlp2} representing the second MLP as $P(q_t = k|\mathbf{x}_{t-d_2:t+d_2}, \Theta_{mlp2})$.

This approach is similar in principle to the RNN based hierarchical approach *G4* and the CRF based hierarchical approach *G5*. The classifiers in the second stage of these systems are trained discriminatively using either posterior features or articulatory features. Apart from the modeling abilities of these classifiers, the main difference between these hierarchical systems is the temporal context on the posterior features. In the RNN based system, the new estimates of posterior probabilities are conditioned on all previously observed posterior feature vectors. In the CRF based approach, it is conditioned on the entire sequence of posterior features. Whereas in our approach, the temporal context on the posterior features is explicitly limited to be around 150-230 ms.

The works described in *G1-G3* are primarily motivated towards exploiting the temporal information in the acoustic features. Whereas in our work as well as *G4* and *G5*, the hierarchical system is motivated towards exploiting temporal information in the posterior features. In this work, the first MLP is trained using standard PLP features. However, it can be trained with any acoustic features, or the first stage can be entirely replaced with more sophisticated MLP based systems described in *G1-G2*. Table 4.1 gives a summary of the discussed approaches highlighting the differences in the temporal context and the nature of the second classifier in the hierarchy.

The proposed hierarchical framework can also be related to the following prior works in the literature

system name	temporal context		C2	C2
	C1 (acoustic)	C2 (posterior)	features	type
G1 (Hermansky and Sharma, 1999; Chen *et al.*, 2001)	long (1s)	nil	P	MLP
G2 (Valente and Hermansky, 2008a,b)	long (1s)	90 ms	A+P_{tr}	MLP
G3 (Ketabdar *et al.*, 2006)	T	nil	-	HMM
G4 (Khan *et al.*, 2000)	any	1:t	P	RNN
G5 (Morris and Fosler-Lussier, 2008)	any	T	P	CRF
G6 (Pinto *et al.*, 2008; Ketabdar and Bourlard, 2008)	any	230 ms	P	MLP

Table 4.1. Summary of the hierarchical systems exploiting temporal information. Notations include: classifier-1 (C1), classifier-2 (C2), acoustic features (A), posterior features (P), posterior features transformed using $\log$ and KLT (P_{tr}), length of the utterance (T).

G7: Bottleneck Features

In bottleneck feature extraction (Grezl *et al.*, 2007), a five layer MLP with a bottleneck constriction at the middle (or compression) layer, is trained to classify phonemes. The linear activation values at the bottleneck layer are used as features in Tandem based speech recognition. The processing from the input to the compression layer can be likened to the first MLP in the hierarchical system, and the processing from the compression layer to the output layer can be likened to the second MLP.

Even though the architectures of both these systems seem to be similar, the motivation for these works and their application in speech recognition are different. In the bottleneck feature extraction, the objective is to obtain lower dimensional features (independent of the phonetic classes), which are more suitable to the ensuing HMM/GMM system. In the proposed hierarchical system, the first MLP transforms the acoustic features to posterior features with lesser undesirable variabilities such as speaker and environment characteristics. Consequently, the second MLP can exploit the temporal information in the posterior features spanning temporal contexts as long as 250 ms. The second MLP gives new estimates of phonetic class conditional probabilities.

G8:Frame-based MPE

The hierarchical system discussed in this work can be related to the frame based minimum phone error (fMPE) system (Povey *et al.*, 2005). In fMPE, a very high dimensional vector of posterior probabilities is obtained from Gaussian mixture models with a temporal context. The high dimensional posterior vector is projected to a lower dimensional feature space, and used as a correction to the input features such as PLP cepstral coefficients. The linear transformation matrix and the acoustic models are jointly trained using the minimum phone error criterion (Povey and Woodland, 2002).

In the MLP based hierarchical system, the high dimensional vector of posterior probabilities is obtained by applying a long temporal context on the posterior features estimated by the MLP. The second MLP acts as a nonlinear transform, and it is trained using a minimum cross-entropy error criterion, which also achieves minimum phone error rate in the asymptotic sense assuming model correctness. Apart from the nonlinear transformation, the major difference between the two is that in fMPE, the transformed posterior vectors are used as a correction to the input features. Whereas, in the hierarchical system, they are used as new features in ASR. Interestingly, fMPE has been shown to be a special case of semi-parametric trajectory modeling and that it captures the trajectories of the acoustic features (Sim and Gales, 2007). In our case, the second MLP learns the trajectories of the posterior features. This is discussed in Section 4.4.

4.3 Experiments and Results

4.3.1 Experimental Setup

The efficacy of the hierarchical system in estimating phoneme posterior probabilities is evaluated by performing speaker independent phoneme recognition experiments on TIMIT as well as CTS databases. We preferred phoneme recognition as it facilitates a detailed analysis of the results. Improvements in word recognition using the hierarchical approach have been previously discussed in (Ketabdar and Bourlard, 2008; Pinto *et al.*, 2009b). In this thesis, ASR experiments are discussed in Chapter 5 and Chapter 6.

The TIMIT database consists of 4.3 hours (including 1.1 hours of NIST complete test set) of read speech, recorded in clean conditions. The 'sa' dialect sentences in the database are not included in the experiments. The database is hand-labeled using 61 phonetic symbols, which include the closures as well as the allophonic variations of certain phonemes. In our experiments, these phonetic symbols are mapped to the standard set of 39 phonemes (Lee and Hon, 1989) with an additional garbage class.[2]

The CTS setup used in the experiments consists of 277.7 hours speech defined as *ctstrain04*, which is a subset of the *h5train03* data set defined at the Cambridge University for training the

[2]Unlike in (Lee and Hon, 1989), the closures are merged with their corresponding bursts (*e.g.,* /bcl/,/b/→/b/). The garbage class handles frames with no labels, and the glottal stop /q/ and its closure /qcl/. The garbage and silence classes are excluded while evaluating the recognition accuracies.

CU-HTK system for RT03 evaluation (Evermann *et al.*, 2004; Woodland *et al.*, 2003). [3] The phonetic transcription of the speech - required for training the MLP as well as computing the accuracy of phoneme recognition - is obtained by Viterbi forced alignment. For this, we used off-the-shelf HMM/GMM acoustic models developed in (Hain *et al.*, 2005) in conjunction with the UNISYN (Fitt, 2000) pronunciation dictionary containing 45 phonemes.

In all the experiments, the acoustic features are the first 13 PLP cepstral coefficients. These coefficients, after speaker specific mean and variance normalization, are appended to their delta and delta-delta derivatives, to obtain a 39 dimensional feature vector for every 10 ms. A three layered MLP with sigmoid nonlinearity at the hidden layer, and softmax nonlinearity at the output layer is used in all the experiments. The parameters of the MLP are optimized using the minimum cross-entropy error criterion. Phoneme recognition is performed using hybrid HMM/MLP approach (Bourlard and Morgan, 1994). The sequence of phonemes is decoded by applying Viterbi algorithm, where each phoneme is represented by a strictly left-to-right, three-state HMM, thereby enforcing a minimum duration of 30 ms. The emission likelihood in each of the three states is the same, and is derived from the associated output of the MLP.

	TIMIT			CTS		
	train	CV	test	train	CV	test
speech (hours)	2.6	0.6	1.1	232.0	36.3	9.4
speakers	375	87	168	4538	726	182

Table 4.2. The number of speakers and the amount of data in the train, cross-validation (CV) and test sets of TIMIT and CTS.

Table 4.2 shows the number of speakers and the amount of data in the training, cross-validation, and test sets of the two databases. On TIMIT, the train and test sets are according to the standard protocol. On CTS, the total data is split into train, CV, and test sets as shown in the table. The parameters of the MLP and the phoneme n-gram models are trained on the train set. The cross-validation set is used to control the learning rate of the MLP. In addition, while decoding, the optimal phoneme insertion penalty (and language model scaling factor, if phoneme n-gram models are used) is optimized on the cross-validation data. All the results reported in this work are on the test set, which is not seen in the entire training process.

[3]The *h5train03* setup consists of around 296 hours of speech from Switchboard-I (Godfrey *et al.*, 1992), Switchboard Cellular, and Callhome English speech corpora, distributed by the Linguistic Data Consortium. For training the AMI RT05 system (Hain *et al.*, 2005), the sentences containing words which do not occur in the dictionary were removed, resulting in 277.7 hours of *ctstrain04* data set.

On the CTS task, training an MLP with 232 hours of speech is computationally expensive.[4] In order to speed up the experiments to obtain various plots, the training data set is split randomly into two equal parts. The first MLP is trained with one half of the training data, and the second MLP is trained with the remaining half. The single MLP based system is, however, trained on the complete training data. On TIMIT, as the amount of training data is small, both the MLPs in the hierarchical system are trained on the same data.

The MLPs are trained using the Quicknet package (Johnson *et al.*, 2000). The phoneme n-gram models are trained using the SRILM toolkit (Stolcke, 2002) and phoneme recognition is performed using the weighted finite state transducer based Juicer decoder (Moore *et al.*, 2006).

4.3.2 Experimental Results

Table 4.3 shows the phoneme recognition accuracies obtained by hierarchical modeling (system S2) in comparison with the standard single MLP modeling (system S1). The single MLP system is trained using PLP features with a 90 ms context. The second MLP in the hierarchical system is trained using the output of the single MLP based system S1, with a temporal context of 230 ms. It can be seen that, by hierarchical modeling we obtain an absolute improvement of 3.5% in recognition accuracy on TIMIT, and 9.3% on CTS. To study the effect of increase in the model capacity on the recognition accuracies, we also compare these results to those obtained by a single MLP based system with the same number of parameters as in the hierarchical system (system S3). In this case, the improvement in the recognition accuracies is 2.5% and 8.3% respectively.

	single MLP baseline (**S1**)	hierarchical two MLPs (**S2**)	single MLP same capacity (**S3**)
TIMIT	68.1	71.6	69.1
CTS	54.3	63.6	55.3

Table 4.3. Phoneme recognition accuracies obtained by using hierarchical posterior estimation as compared to the standard single MLP on TIMIT and CTS databases.

We also perform model selection on the TIMIT database to select the optimal size of the hidden layer of the first MLP. After model selection, the single MLP based system yields a recognition accuracy of 69.2%, which is marginally higher than the one obtained by parameter equalization

[4]Using multi-threaded version of Quicknet (Johnson *et al.*, 2000) (with eight threads and bunch size of 2048), training an MLP of size $351 \times 5000 \times 45$ on 232 hours of speech takes roughly 72 hours to complete 8 epochs on a 2.4 GHz, AMD Opteron processor, with eight cores.

technique. By training a second MLP classifier on the posterior features estimated after model selection, we obtain a recognition accuracy of 72.7%, which corresponds to an absolute improvement of 3.5%. This further demonstrates that the improvement in recognition accuracy is not due to the increase in the modeling complexity.

In Figure 4.3, we compare the phoneme recognition accuracies obtained using the hierarchical approach to those obtained using the single MLP approach for different values of the temporal context. In the case of hierarchical system, the first MLP is always trained using a temporal context of 90 ms on the acoustic features. As the temporal context on the posterior features at the second MLP is increased, the total number of parameters in the MLP is kept constant by appropriately reducing the size of the hidden layer of the second MLP classifier.[5] In the case of single MLP estimator, as the temporal context on the acoustic features is increased, the total number of parameters is kept constant, and equal to those in the hierarchical system (sum of the parameters in both the MLPs).

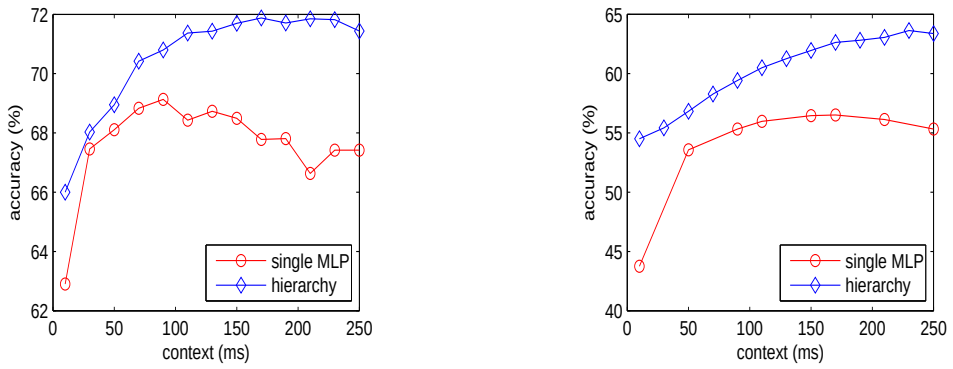


Figure 4.3. (a) Phoneme recognition accuracy on TIMIT using a hierarchical setup as well as single MLP with the same number of parameters. In hierarchical system, the size of the first MLP is $351 \times 1000 \times 40$, and the size of the second MLP for 23 frame context is $920 \times 1083 \times 40$. (b) A similar plot on the CTS, where the size of the first MLP is $351 \times 5000 \times 45$, and the size of the second MLP for 23 frame context is $1035 \times 1334 \times 45$. Any two points in the plot correspond to systems with the same number of parameters, and can be calculated using footnote 5.

It can be seen from the figure that:

1. The hierarchical posterior estimator consistently outperforms the single based MLP posterior estimator with the same number of parameters for all values of context. As the context at the second MLP is increased, even though the number of hidden nodes is decreased, there is a steady increase in the recognition accuracies. Thus it can be concluded that improvement

[5] If F denotes the dimensionality of the features, C denotes the temporal context, and H (and O) denote the size of the hidden (and output) layers, the number of parameters in the MLP is given by $C * F * H + H + H * O + O$.

is due to the topology of two MLPs in tandem, and not merely due to the increase in overall model capacity.

2. In case of CTS, the recognition accuracies begin to saturate at around 230 ms of temporal context at the input of the second MLP. In case of TIMIT, the accuracies begin to saturate after 150 ms, but this could be due to the lack of sufficient training data. In both cases, the effective temporal context of 150-230 ms extends well beyond the typical duration of phonemes (50-70 ms), which suggests that the second MLP is integrating temporal information in the posteriors features corresponding to the neighboring phonemes as well.

3. A long temporal context is more effective when applied on the posterior features rather than on the standard acoustic features. On increasing the temporal context on the acoustic features at the input of the single MLP system, recognition accuracies peak for a context of around 90-110 ms, but are significantly lower compared to the hierarchical system.

From the above discussion it is clear that the hierarchical system is useful as a phoneme posterior estimator, and that a long temporal context is more effective on the posterior features rather than on the acoustic features. Since the second MLP is trained using posterior features, which represents the underlying sequence of phonemes, it is clear that the second MLP is learning the phonetic-temporal patterns.

The following questions, however, remain unanswered: (a) what are the phonetic-temporal patterns learned for each phoneme ? (b) due to the long temporal context extending beyond the typical duration of a phoneme, has the MLP also learned the phonotactics of the language ? and (c) why is the temporal context more effective on the posterior features ? The first two questions can be answered by analyzing the input-output relationship learned by the second MLP classifier using Volterra series. This is discussed in Section 4.4. The effectiveness of temporal context on the posterior features is discussed in Section 4.5.

4.3.3 Second MLP as a Function

The second MLP can be viewed as a vector valued function $\mathbf{f}_{mlp2}(.)$, which takes the estimates of posterior probabilities of phonemes from the first MLP denoted by $\mathbf{x}_{t-d_2:t+d_2}$ as its arguments, and

gives a new estimate of the posterior probabilities of phonemes $\mathbf{z}_t$ as

$$\mathbf{z}_t = \mathbf{f}_{mlp2}(\mathbf{x}_{t-d_2:t+d_2}) \tag{4.3}$$

In the second MLP classifier, let $\mathbf{W}$ denote the weight matrix connecting the input layer to the hidden layer, $\mathbf{C}$ denote the weight matrix connecting the hidden layer to the output, $\mathbf{b}_h$ and $\mathbf{b}_o$ denote the bias vectors at the hidden and output layers respectively, and $\mathbf{f}_{soft}(.)$ and $\mathbf{f}_{sigm}(.)$ denote the vector valued softmax and sigmoid functions at the output and the hidden layers of the MLP respectively. Then, equation (4.3) can be expressed as

$$\mathbf{z}_t = \mathbf{f}_{soft}\left(\mathbf{y}_t\right) \tag{4.4}$$

where the vector $\mathbf{y}_t = [y^1(t), \ldots y^j(t), \ldots y^N(t)]'$ denotes the linear activation vector before the softmax nonlinearity at the output layer of the MLP, and is given by

$$\mathbf{y}_t = \mathbf{b}_o + \mathbf{C}\mathbf{f}_{sigm}\left(\mathbf{b}_h + \mathbf{W}\mathbf{x}_{t-d_2:t+d_2}\right) \tag{4.5}$$

It is difficult to analyze or interpret the input-output relationship $(\mathbf{x}_t, \mathbf{z}_t)$ of the MLP, given by (4.4) and (4.5), due to the presence of nonlinear functions $f_{sigm}(.)$ and $\mathbf{f}_{soft}(.)$. The output nonlinearity can be conveniently dropped from the analysis as parameters of the discriminatively trained MLP $\{\mathbf{W}, \mathbf{b}_h, \mathbf{C}, \mathbf{b}_o\}$ can still be interpreted from the input-output relationship $(\mathbf{x}_t, \mathbf{y}_t)$. This does not affect the interpretability as the output units are still phonemes, and the rank ordering of the estimates are not altered. However, the nonlinearity at the hidden layer can still make the analysis of (4.5) difficult.

In our previous work (Pinto *et al.*, 2008), this problem was circumvented, but not solved, by using a single layer perceptron (SLP) in place of the second MLP in the hierarchical system. The SLP retained the same input-output architecture, training data, and optimization criterion as that of the MLP. The weights of the trained perceptron revealed the linear fit to the observed training data. However, the MLP classifier which was actually used in ASR was not analyzed. In this chapter, we follow a more principled approach and represent the second stage of the hierarchical system (creation of temporal context and the MLP classifier) using Volterra series. For this, we

treat the multi-input $\mathbf{x}_t$, multi-output $\mathbf{y}_t$ system characterized by (4.5) as a nonlinear time-invariant system.

4.4 Application of Volterra Series

In Chapter 3, we discussed the application of Volterra series to cascade of an FIR filter bank and a three-layered MLP. In this section, we compute the Volterra kernels for multi-input $\mathbf{x}_t$, multi-output $\mathbf{y}_t = [y^1(t), \ldots y^j(t), \ldots y^N(t)]'$ system characterized by (4.5). This system can be viewed as N parallel, multi-input, single-output, nonlinear, time-invariant systems, and represented by

$$y_t^j = b_o^j + \mathbf{C}^j \mathbf{f}_{sigm}\left(\mathbf{b}_h + \mathbf{W}\mathbf{x}_{t-d_2:t+d_2}\right), \quad j = 1 \ldots N, \tag{4.6}$$

where, $\mathbf{C}^j$ denotes the weight row vector connecting the hidden layer to the output node j, and b_o^j the bias at the output node j. The system represented by (4.6) can be realized using the framework shown in Figure 3.5, where the temporal context of $2d_2+1$ frames on the posterior features, denoted by $\mathbf{x}_{t-d_2:t+d_2}$, can be created by filtering $\mathbf{x}_t$ using a bank of $L = 2d_2 + 1$ FIR filters. The impulse response of the $2d_2 + 1$ tap FIR filter is given by

$$h_l(n) = \delta\left(n + l - \frac{L+1}{2}\right), \quad \text{with} \quad l = 1, 2 \ldots L \quad \text{and} \quad n = -d_2, \ldots 0, \ldots d_2$$

The Volterra kernels are computed in terms of the above impulse response functions and the weights of the trained MLP using (3.34)-(3.35). In practice, due to feature normalization, $\mathbf{x}_t$ represents posterior features which are normalized to zero mean and unit variance.

In the remaining part of this section, we analyze trained second MLPs in the hierarchical system (see Table 4.3 for results) - one trained on TIMIT ($K = 40, L = 23, M = 1083, N = 40$), and the other trained on CTS ($K = 45, L = 23, M = 1334, N = 45$). Before analyzing the Volterra kernels, the accuracy of first and second order truncated Volterra series is evaluated. For this, we substitute the identified kernels in the synthesis equation (3.33) to obtain the linear activation values of phonemes. Approximate estimates of phoneme posterior probabilities are obtained by applying softmax nonlinearity, and subsequently used in phoneme recognition.

Table 4.4 shows the phoneme recognition accuracies obtained by the first and second order

model	series order	phoneme accuracy	
		TIMIT (%)	CTS (%)
linear	1	68.7	50.1
quadratic	2	70.1	54.9
MLP	∞	71.6	63.6

Table 4.4. Phoneme recognition accuracy obtained by linear and quadratic approximation of the MLP using Volterra series.

Volterra series approximation of the second MLP classifier. It can be seen that on TIMIT, the phoneme recognition accuracy obtained by the first order Volterra approximation is only three percent lower compared to direct evaluation of the MLP function. In other words, the second (quadratic), third (cubic), and higher order parts contribute very little to nonlinear modeling ability of the second MLP. Hence, in this case, the linear Volterra kernels reveal most of the information learned by the nonlinear classifier.

In the case of a more complex CTS task, the second and higher order Volterra kernels contribute significantly (around 13.5%) to the modeling ability of the second nonlinear classifier. Hence, in this case, the linear Volterra kernels can only partially explain the second MLP. The remaining information is complemented by the higher order Volterra kernels. In this work, we restrict the analysis to linear Volterra kernels.

4.4.1 Interpretation of the First Order Volterra Kernels

It is clear from (3.33) that the first order Volterra kernels reveal the linear part of the nonlinear system under analysis. Suppose that the second MLP is trained using a temporal context of 230 ms, then the Volterra kernel for phoneme $j = 1, 2 \dots N$ at the output of the second MLP is given by $g_k^j(m)$, and reveals the contribution of each of the phonemes $k = 1, 2 \dots K$ at the input of the MLP, in a window of $m \in [-11, \dots 0, \dots 11]$, which amounts to 230 ms of context. As the input to the second MLP is in terms of phonemes, the first order Volterra kernels can be interpreted as phonetic-temporal patterns. In our experiments, $N = K$ as both the MLPs in the hierarchical system are trained on the same phoneme set.

The phonetic-temporal patterns observed in the first order Volterra kernels can reveal two important aspects learned by the second MLP classifier: 1) the acoustic confusion among phonemes at the output of the first MLP classifier, and 2) the phonotactics of the language as observed in the

training data. In the remaining part of this section, we discuss these aspects in detail.

Volterra kernels revealing acoustic confusion patterns among phonemes

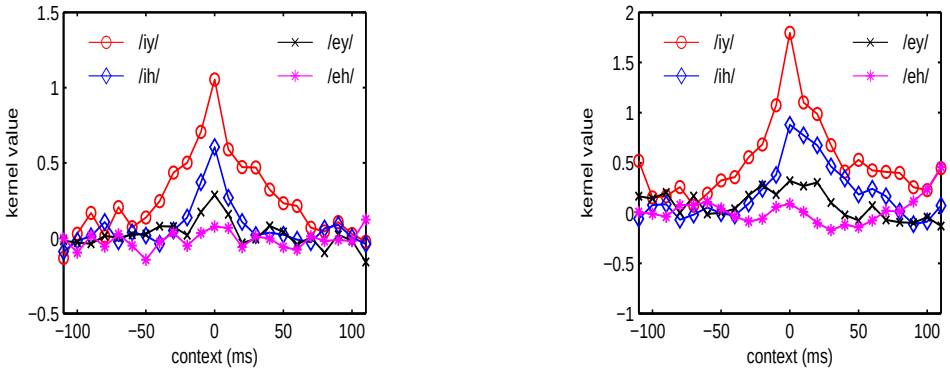


Figure 4.4. (a) First order Volterra kernel of the phoneme /iy/ (*e.g.,* b**ea**t) obtained on TIMIT. (b) A similar plot on CTS database.

Figure 4.4 (a) and (b) are the plots of the first order Volterra kernel of the second MLP classifier for the vowel /iy/ (*e.g.,* b**ea**t) on TIMIT and CTS respectively. The figure shows the impulse response functions corresponding to the top four contributing phonemes at the input of the MLP. The impulse response function corresponding to other phonemes are not plotted in the figure for clarity. The top contributing phonemes are selected based on the energy in their impulse response functions. It is not surprising that the maximum contribution is from the same phoneme /iy/ at the input. There are, however, positive contributions from other confusing vowels such as /ih/, /ey/, and /eh/.

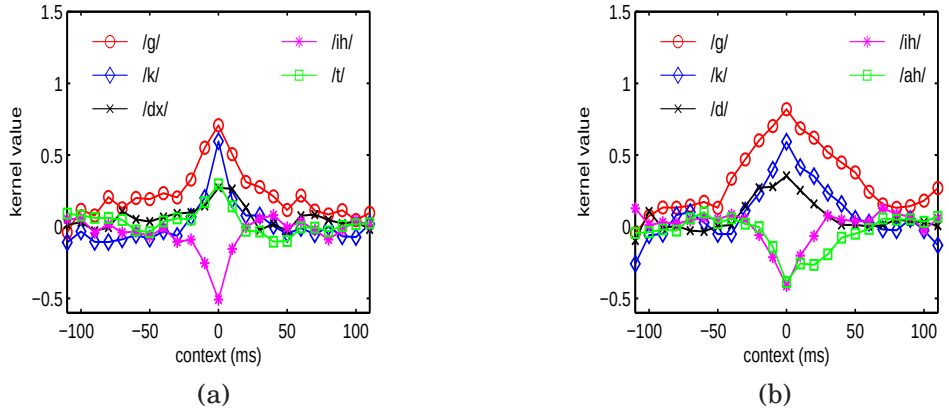


Figure 4.5. First order Volterra kernel of the phoneme /g/ (*e.g.,* **g**oat) obtained on TIMIT. (b) A similar plot on CTS database.

Figure 4.5 (a) and (b) are plots of first order Volterra kernel of the phoneme /g/ (*e.g.,* **g**oat) obtained on TIMIT and CTS databases respectively. It can be seen that the kernels show positive contributions from other confusing consonants such as /k/, /t/, /d/, and /dx/. Moreover, the MLP has also learned to give negative weights to certain vowels such as /ih/ and /ah/. This is due to the discriminative training of the MLP classifier and this information is otherwise not intuitive. It suggests that the consonant /g/ is not likely to be confused with the vowels such as /ih/ or /ah/.

phonemes TIMIT	confusions Volterra	confusion matrix	phonemes CTS	confusions Volterra	confusion matrix
iy	ih, ey, eh	ih	iy	ih, eh, ey	ih, ey
ih	iy, eh, ae	ah	ih	iy, sil, eh	ax, iy
ey	ih, iy, ae	ih, iy	ey	ih, ay, eh	iy, ih
eh	ih, ae, ah	ih, ae, ah	eh	ah, ih, ey	ae, ih, ax, ah
			aa	ah, ay, ow	ah, ay, ao
ah	ih, ao, eh	ih, ao, ow	ah	ay, eh, l	ax, ow
			ax	axr, ah, m	ih, ah
			axr	r, ax, ih	r, ax
uw	ih, iy, w	ih, iy	uw	iy, ih, ow	iy, ax
uh	ih, ah, eh	ih, ah, ow, l, uw	uh	ih, s, ey	ax, ih
ae	ao, ah, aw	eh	ae	eh, ah, ay	eh
ao	ae, ay, ah		ao	aa, l, w	aa, ow
aw	ao, ah, ae	ao, ae	aw	ah, ay, eh	ae, ow, ah, aa, eh, ay
ay	ao, ah, ey	ao	ay	ah, eh, aa	ah
ow	ah, ao, l	l, ah, ao	ow	ah, l, ao	ah, l
oy	ao, ih, ay	ao, ey	oy	r, w, ay	w, l, ao, ow
y	iy, ih, oy	iy, uw, ih	y	iy, ae, ch	iy, sil
w	l, uh, oy	l	w	l, r, ao	
l	ao, ah, ow	ow, ao	l	ah, el, w	ow
			el	l, ow, ao	l, ow, ax
r	er, ae, ao	er	r	axr, iy, w	axr
er	r, ih, ah	r	er	r, axr, ih	r, axr
hh	sil, k, p	sil	hh	s, ae, dh	sil
m	n, p, b	n	m	n, ng, w	n, sil
			em	n, ah, m, en	m, ah, sil, n, ax
n	m, dx, dh	m	n	m, ng, en	d
			en	n, m, ng	n, ax, d, m
ng	n, m, uw	n	ng	n, m, iy	n
p	t, b, k		p	k, t, f	t, sil
t	d, p, k	d, k	t	d, k, m	sil
k	sil, t, p	t	k	sil, p, t	sil, t
b	p, d, m	p	b	p, dh, w	dh
d	t, dx, k	t	d	t, sil, s	t, n, sil
g	k, d, t	k, d	g	k, d, dh	k
dx	d, n, dh	d			
f	p, s, sil		f	s, sil, k	s, sil
th	s, t, f	f, t	th	s, sil, f	s, t, sil
s	z, sh, f	z	s	f, sh, z	sil, z
sh	s, z, jh	s	sh	s, f, ch	s, ch
v	f, b, m		v	sil, f, z	ax
dh	t, th, d	sil	dh	y, b, g	t, d
z	s, sh, th	s	z	s, sil, f	s, sil
			zh	iy, ih, z	z, sh, uw
ch	s, jh, sh	sh, t, jh, s	ch	t, s, k	t, s, sh
jh	s, z, sh	ch, sh	jh	ch, d, y	t, d, ch

Table 4.5. Confusing phonemes at the center of the Volterra kernels (top three) as compared to the phonetic confusion matrix (value > 0.06).

Since both the input and output representations at the second MLP are in terms of phonemes, the first order Volterra kernel can be interpreted as phonetic-temporal confusion patterns. However, unlike the standard phonetic confusion matrix, the first order Volterra kernels reveal the contribution of the input phonemes in a window of certain duration depending on the temporal context used. In Table 4.5, we show the top three contributing phonemes at the center ($m = 0$) of the Volterra kernels for both TIMIT as well as CTS databases. These confusion patterns are compared to the standard confusion matrix, obtained by performing frame-level phoneme classification at the output of the first MLP. Only entries in the confusion matrix with values greater than 0.06 are shown in the table.

It can be seen from the table that the confusions at the center of the Volterra kernels match to a certain extent with standard phonetic confusion matrix derived from the posterior features. However, these confusion entries need not be the same because the Volterra kernels represent the discriminatively trained second MLP classifier, whereas the phonetic confusion matrix is a measure of the phonetic confusion in the posterior features, which are used to train the second MLP.

It is interesting to note that the ability of the second classifier in the hierarchical setup to learn the acoustic confusion among phonemes at the output of the first MLP has also been observed in the CRF based hierarchical system (Fosler-Lussier and Morris, 2008), which is discussed in Section 4.2.3. In this work, we explicitly show using Volterra analysis the phonetic-temporal patterns that are learned.

Volterra kernels revealing the phonotactics of the language

A closer look at the first order Volterra kernels reveals that the MLP has also learned the phonotactics in the training data. In the ensuing discussions, the following notations are used. $P(p1^{+}|p2) = P(p_{n+1} = p1|p_n = p2)$ denotes the probability that phoneme /$p1$/ follows /$p2$/, and is typically used using n-gram statistical language modeling. In contrast, $P(p1^{-}|p2) = P(p_{n-1} = p1|p_n = p2)$ denotes the probability that phoneme /$p1$/ precedes /$p2$/. To estimate this language model, the sequence of phonemes in the training data are reversed, and bigram statistics are estimated.

Figure 4.6 (a) is a plot of the first order Volterra kernel of the phoneme /y/ on TIMIT, showing the contributions of two phonemes /uw/ and /er/ that are most likely to follow /y/. It can be seen that the corresponding kernels have higher value to the left of the origin as compared to the right.

This is because $P(uw^+|y) = 0.52 \gg P(uw^-|y) = 0.04$. As Volterra kernels are impulse response functions, the corresponding matched filters are obtained by time-reversing the kernels about their origin $t = 0$.

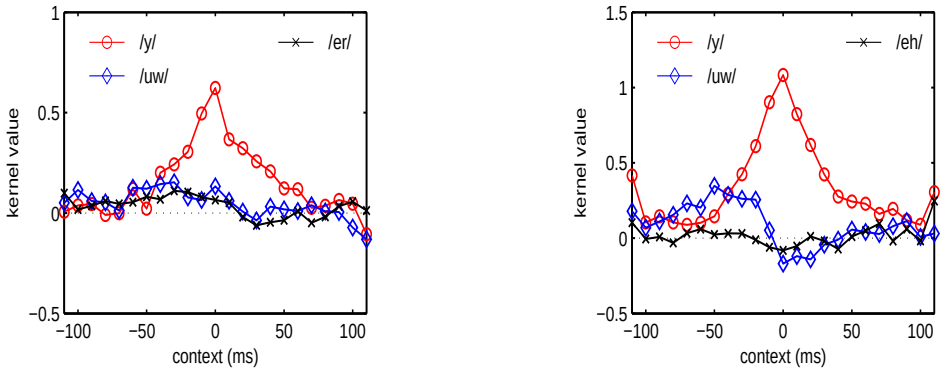


Figure 4.6. (a) Volterra kernel of phoneme /y/ on TIMIT. $P(uw^+|y) = 0.52$, $P(uw^-|y) = 0.04$, $P(er^+|y) = 0.16$, and $P(er^-|y) = 0.03$. (b) Volterra kernel of phoneme /y/ on CTS. $P(uw^+|y) = 0.54$, $P(uw^-|y) = 0.04$, $P(eh^+|y) = 0.30$, and $P(eh^-|y) = 0.001$.

Figure 4.6 (b) is a plot of the Volterra kernel of phoneme /y/ on CTS, showing the impulse response functions of phonemes /uw/ and /eh/, that are most likely to follow /y/. It can be seen that the kernel for /uw/ is consistent with the bigram language model probabilities, but in case of /eh/, there is no such agreement as the kernel is close to zero for all values of the context.

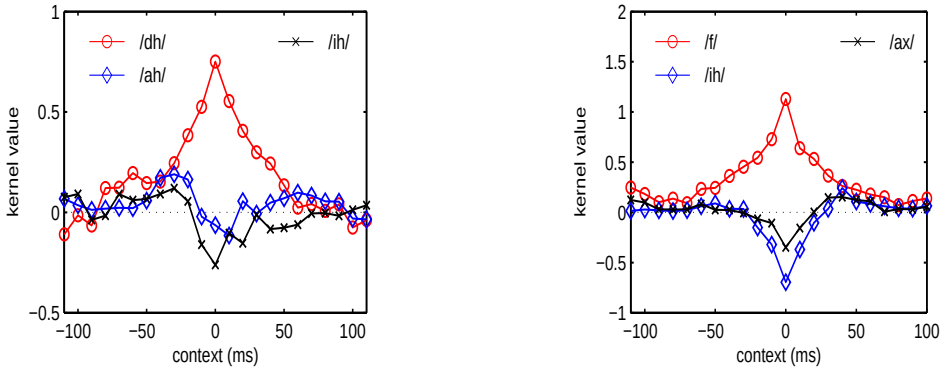


Figure 4.7. (a) Volterra kernel of phoneme /dh/ on TIMIT. $P(ih^+|dh) = 0.34$, $P(ih^-|dh) = 0.04$, $P(ah^+|dh) = 0.29$, and $P(ah^-|dh) = 0.11$ (b) Volterra kernel of phoneme /f/ on CTS. $P(ih^+|f) = 0.07$, $P(ih^-|f) = 0.17$, $P(ax^+|f) = 0.05$, and $P(ax^-|f) = 0.10$.

Figure 4.7 (a) is the plot of the impulse response functions of phonemes /dh/, /ah/, and /ih/ in the first order Volterra kernel of phoneme /dh/ (*e.g.*, **th**is) on TIMIT. It can be seen that the impulse

response functions of phonemes /ih/ and /ah/ have higher weight to the left of origin as compared to the right. This is because the pairs of phonemes /dh//ah/ and /dh//ih/ occur more frequently in the training data than the pairs of phonemes /ah//dh/ and /ih//dh/.

In Figure 4.7 (b), we plot the impulse response functions of phonemes /f/, /ih/, and /ax/ in the first order Volterra kernel of phoneme /f/ (*e.g.,* **f**ar) on CTS. Phonemes /ih/, and /ax/ are the two most likely phonemes to precede /f/ and as a consequence, their impulse response functions have higher values to the right of the origin. Moreover, it can also be seen that at the origin, the impulse response functions of /ih/ and /ah/ have negative weights, which suggests that these vowels are not confusable with consonant /f/. It should be noted that the Volterra kernels reveal the properties of the discriminatively trained MLP. Hence, they need not always be consistent with the bigram probabilities between phonemes (derived from simple counts) in all cases.

The interpretations that can be drawn by analyzing the linear Volterra kernels are summarized below. Let $g_1^1(\tau)$ and $g_2^1(\tau)$ are the impulse response functions (indicating the contributions) of phonemes /p1/ and /p2/ respectively in the Volterra kernel of the phoneme /p1/. The function $g_1^1(\tau)$ will always have a positive peak at the origin $\tau = 0$. Depending on the shape of the function $g_2^1(\tau)$, the interpretations could be as follows: (a) a positive peak at the origin indicates the acoustic confusion between the phonemes, (b) a negative valley at the origin indicates the anti-confusion due to the discriminative training of the MLP, and (c) a peak which is shifted away from the origin reveals the phonotactics implicitly learned by the MLP. Moreover, the Volterra kernels can also reveal the effective temporal duration on the posterior features.

4.4.2 Decoding with Language Models

First order Volterra analysis of the hierarchical system reveals that, apart from the acoustic confusions, the second MLP has also implicitly captured the phonotactics of the language. However, it is not clear if the implicitly learned phonotactics has indeed contributed towards the increase in the recognition accuracies in the hierarchical system. To ascertain this, we performed phoneme recognition by explicitly using phoneme n-gram models.

Figure 4.8 (a) and (b) are plots of the phoneme recognition accuracies on TIMIT and CTS respectively, obtained by decoding with zerogram (loop of phonemes with equal transition probabilities), bigram and trigram phoneme language models. The accuracies are shown for temporal context at

the second MLP ranging from 10ms to 250ms. As the input context is increased, the total number of parameters of the second MLP is kept constant by appropriately modifying the size of the hidden layer. The horizontal dotted lines in the plot indicate the recognition accuracies obtained by a single MLP based system using different language models. It can be seen from the figure that recognition accuracies increase by explicitly using bigram and trigram models. This improvement is observed for all values of the temporal context on the posterior features, but the gain in the accuracies decreases with the increase in context.

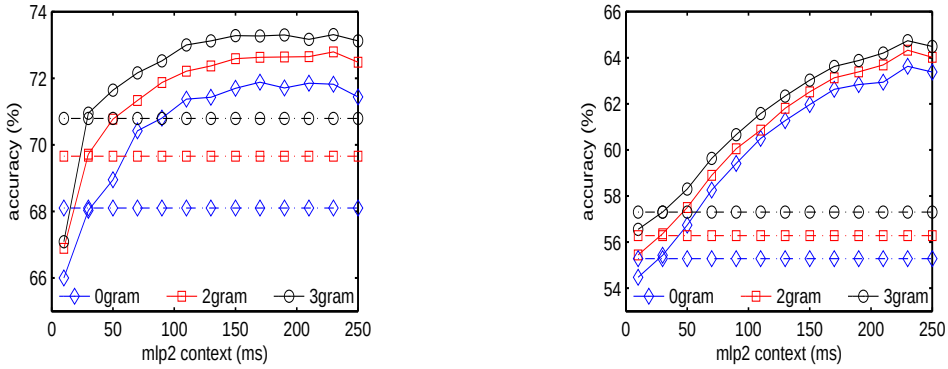


Figure 4.8. (a) Phoneme recognition accuracies on TIMIT using zerogram, bigram, and trigram phoneme language models. The horizontal lines show the accuracy of the first MLP using language models. (b) A similar plot on CTS database.

To illustrate this, in Figure 4.9 we plot the relative gain in the recognition accuracies obtained on CTS by decoding with bigram and trigram language models over no language model, as a function of the temporal context at the input of the second MLP classifier. It can be seen that the gain in accuracy obtained by explicitly using a phoneme n-gram model decreases with the increase in the temporal context. This is because with increase in the temporal context, the second MLP is able to learn the phonotactics more effectively, and gain in accuracy by introducing explicit language models reduces. This further supports the observations from the linear Volterra kernels. However, even with 230 ms context, the MLP has only partially learned the phonotactics and we still obtain 1-2% improvement in accuracies by using language models in decoding.

To summarize briefly, we showed in this section that the second MLP classifier in the hierarchical system learns the phonetic-temporal patterns (acoustic confusions among phonemes and the phonotactics of the language) in the posterior features spanning a temporal context of 150-230 ms.

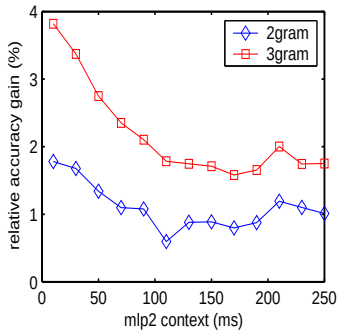


Figure 4.9. Relative gain in recognition accuracy on CTS database obtained by decoding with bigram and trigram language model as compared to no language model for different values of the temporal context at the input of the second MLP.

In the following section, we discuss the important properties of the posterior features that enabled the second MLP to effectively learn these patterns.

4.5 Modeling flexibility of posterior features

In this section, we discuss the useful properties of posterior features such as (a) lesser nonlinguistic variabilities when compared to the acoustic features, (b) sparse distribution, and (c) linear separability in the posterior feature space. We also discuss the consequence of these properties on the complexity of the second MLP classifier and the amount of training data.

4.5.1 Characteristics of Posterior Features

Variability in posterior features

The acoustic features are known to exhibit a high degree of nonlinguistic variabilities such as speaker and environmental (*e.g.,* noise, channel) characteristics. The first MLP classifier can be interpreted as a discriminatively trained nonlinear transformation from the acoustic feature space to the posterior feature space. It has been shown that a well trained (large population of speakers, and different conditions) MLP classifier can achieve invariance to speaker (Zhu *et al.*, 2004) as well as environmental (Ikbal, 2004) characteristics. Moreover, it has also been shown that the effect of coarticulation is less severe on the posterior features when compared to the acoustic features (Ellis *et al.*, 2001; Sivadas and Hermansky, 2002). In other words, the posterior features are

soft-decisions on the underlying sequence of phonemes (*i.e.,* the linguistic message), and have much lesser nonlinguistic variabilities when compared to acoustic features.

Sparseness in the posterior features

The posterior features represent the probabilities of the phonetic classes conditioned on the acoustic features, and hence sum up to one at any given time instant. In addition, they are also sparsely distributed in the posterior feature space as shown in Figure 4.2. To illustrate this objectively, in Table 4.6, we show the average number of components (or phonemes) in the posterior feature vector that capture 90, 95, and 99% of the probability mass value. It can be seen that on TIMIT, on average 3.6 phonemes capture 95% of the probability mass value. The other phonemes share the remaining 5% of the probability mass. On CTS, on average 6.2 phonemes capture 95% of the probability mass value, indicating the more complex nature of the task.

	probability mass value		
	>90%	>95%	>99%
TIMIT (max 40)	2.7	3.6	6.6
CTS (max 45)	4.4	6.2	11.3

Table 4.6. Average number of components (phonemes) in the posterior feature vector that capture 90, 95, and 99% of the probability mass in the posterior probabilities of phonemes estimated by the first MLP.

The sparse distribution of the posterior features has been previously studied in (Zhu *et al.*, 2004), where the authors termed the posterior features as more *regular* compared to the standard acoustic features. It was argued that sparse distribution was one of the favorable properties of posterior features.

Linear separability

The model parameters of the first MLP are optimized to minimize the cross entropy between the estimated posterior probability vectors and the output target vectors, which are typically in the hard-target format. In other words, if K denotes the number of phonemes, the hard target vector $l_{p_i} \in \mathbb{R}^K$ for the phoneme $p_i, \quad i = 1, 2 \ldots K$ is given by $l_{p_i}(k) = \delta(k - i)$. The target vectors are, therefore, at the simplex of the K dimensional space, which makes them linearly separable. Hence, a well trained model attempts to achieve linear separability in the estimated posterior features. The degree to which separability is achieved depends on the complexity of the task.

The properties of posterior features discussed in this section can influence the choice of the second MLP classifier in the following ways:

1. Since the posterior features are trained to be linearly separable and have a sparse distribution, a simpler classifier (in terms of model capacity) may be sufficient at the second stage of the hierarchy. We validate this hypothesis in Section 4.5.2.

2. Since the posterior features have lesser variability, the second MLP could be trained with lesser amount of training data. We test this hypothesis in Section 4.5.3.

4.5.2 Complexity of the Second MLP

In this section, we study the effect of the model capacity (in terms of the number of parameters) of the second MLP in the hierarchical system on the phoneme recognition accuracies. Figure 4.10 is a plot the phoneme recognition accuracies obtained by using the hierarchical approach, as a function of the number of parameters in the second MLP classifier (relative to the number of parameters in the first MLP). The number of parameters is controlled by reducing the size of the hidden layer until it equals the size of the input layer. On both TIMIT as well as CTS, the second MLP is trained using a temporal context of 230 ms. The horizontal dotted lines in the plot indicate the recognition accuracies obtained by using the output of the first MLP classifier.

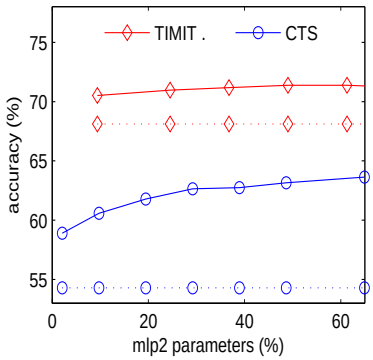


Figure 4.10. Phoneme recognition accuracies as a function of the number of parameters in the second MLP classifier (relative to the number of parameters in the first MLP classifier, which has a size of $351 \times 1000 \times 40$ on TIMIT, and a size of $351 \times 5000 \times 45$ on CTS). In both cases, a temporal context of 230 ms is applied at the input of the second MLP, and the horizontal lines indicate the recognition accuracies obtained by using a single MLP system.

It can be seen from the figure that on both TIMIT as well as CTS, the recognition accuracies drop with the reduction in the number of parameters, and the drop in accuracy is more significant

in the case of CTS. Nonetheless, the hierarchical system still outperforms the single MLP based system on both the tasks. It can be seen from the figure that the second MLP with just 20% of the parameters in the first MLP can still yield significantly higher recognition accuracies over the single MLP based system. As an extreme case, a single layer perceptron (SLP) is used as a second classifier in the hierarchical system. It can be seen from Table 4.7 that even a linear classifier in the second stage of the hierarchy can yield higher recognition accuracies (2.3% and 1.1% respectively on TIMIT and CTS respectively) when compared to the baseline system.

experiment	no hierarchy(%)	MLP hierarchy(%)	SLP hierarchy (%)
TIMIT	68.1	71.6	70.4
CTS	54.3	63.6	55.4

Table 4.7. Phoneme recognition accuracies obtained by hierarchical posterior estimation using a multilayer and single layer perceptron (SLP) classifiers.

It can be recalled from Table 4.4 that, on TIMIT, the phoneme recognition accuracy obtained by first order Volterra series approximation (linear model) was only three percent lower compared to the accuracy obtained by directly evaluating the MLP, indicating the linear separable nature of the posterior features. Therefore, at the second stage of the hierarchy, an MLP classifier with fewer number of parameters (mildly nonlinear) is sufficient. On CTS, however, it can be seen that there is a 13.5% drop in recognition accuracy by approximating the MLP using first order Volterra series, which indicates that on CTS, the posterior features from the first MLP are not as linearly separable as those in TIMIT. This explains the higher drop in recognition accuracies with the reduction in the number of parameters on CTS task.

4.5.3 Size of Training Data

In this section, we study the effect of the amount of data required to train the second MLP in the hierarchical system on the phoneme recognition accuracies. In Figure 4.11, we plot the phoneme recognition accuracies obtained by using the hierarchical approach as a function of the amount of training data used to train the second MLP classifier (relative to the amount of training data used to train the first MLP classifier). The amount of training data is controlled by randomly dropping the sentences in the training set. It can be seen that even with 80% reduction in the training data,

the hierarchical system yields higher recognition accuracies when compared to the baseline system.

In this work, in order to speed up the training time on the CTS task, the training data was split into two halves, and the two MLPs in the hierarchical system were trained on the disjoint data sets. By training the hierarchical system using the above strategy, where the MLPs have sizes $351 \times 5000 \times 45$ and $1035 \times 1334 \times 45$, we obtained a recognition accuracy of 63.6%. However, only a slight improvement in recognition accuracy, about 0.7%, is obtained by training both the MLPs in the hierarchical system on the full 232 hours of data. Moreover, the training strategy for the hierarchical system - same training set or disjoint sets - did not affect the recognition accuracies.

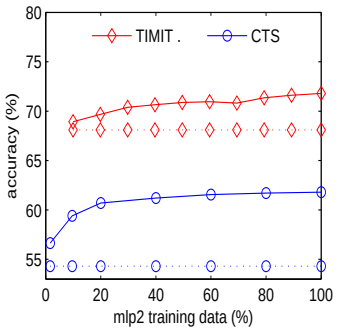


Figure 4.11. Phoneme recognition accuracies as a function of the data used to train the second MLP. 100% data corresponds to 153 minutes on TIMIT, and 116 hours on CTS. An MLP with fewer parameters (200 hidden nodes on TIMIT and 400 on CTS) is used. In both cases, a temporal context of 230 ms is applied at the input of the second MLP. The horizontal lines indicate the accuracies obtained by using a single MLP estimator.

4.6 Discussion

In this section, we discuss some of the interesting aspects of MLP based hierarchical systems.

4.6.1 Choice of Subword Units

In the case of conventional single MLP based system, it has been shown that a more detailed modeling can be achieved by taking the sub-phonemic states, typically three states per phoneme, as the output classes of the MLP. Consequently, higher accuracies in the recognition of phonemes have been observed. A similar trend is also observed in the hierarchical approach, where posterior features representing the sub-phonemic states have been shown to yield higher recognition accuracies. In the case of TIMIT, by using sub-phonemic posterior features, the hierarchical system yields

a phoneme recognition accuracy of 73.4% in comparison with 71.6% obtained using the baseline single MLP based approach (Pinto *et al.*, 2008).

4.6.2 Choice of the First Classifier

In the discussion so far, the second MLP classifier in the hierarchical system is trained using posterior probabilities of phonemes conditioned on acoustic features, which are estimated by an MLP. In general, however, these phonetic class conditional probabilities could be estimated using other statistical models as well. For example, in an earlier work (Pinto and Hermansky, 2008), the posterior probabilities of phonemes or posterior features are estimated using a Gaussian mixture model (GMM). As shown in Table 4.8, the hierarchical system using GMM posterior (and $\log$-likelihood) features with a temporal context of 230 ms yield higher recognition accuracies when compared to the standard HMM/GMM system. Here, single state posterior probabilities are derived by summing up the state posterior probabilities.

classifier	3-state	1-state
HMM-GMM	64.1	62.1
hierarchy, GMM posteriors	68.4	67.1
hierarchy, GMM log-likelihoods	71.0	70.3

Table 4.8. Phoneme recognition accuracy using GMM posteriors and likelihoods as features compared to direct HMM-GMM decoding. A temporal context of 230 ms is applied on the features.

The input to the second classifier could also be a combination of two or more streams of posterior features from different classifiers. For example, in (Pinto and Hermansky, 2008), posterior features estimated using a GMM and an MLP model were jointly used as features to the second MLP with a temporal context of 230 ms. This early integration scheme yielded higher recognition accuracies when compared to the best single stream decoding.

The phonetic class-conditional probabilities estimated in a hierarchical fashion using posterior features from GMM and MLP classifiers can also be combined using late integration schemes such as sum, product, inverse entropy or Dempster Shafer combination rules. Table 4.9 shows the results obtained using single state and three state posterior features for various combination schemes. It can be seen that all late integration combination rules yield significantly higher accuracies in comparison with the single best posterior stream.

	gmm	mlp	sum	prod	I.E.	D.S
1-state	70.3	71.5	73.6	74.0	73.5	73.7
3-state	71.0	73.4	74.2	74.6	74.4	74.6

Table 4.9. Phoneme recognition accuracy using late integration scheme for multi-stream combination. Results shown for sum, product, inverse entropy (IE) and Dempster Shafer (DS) combination as well as individual GMM and MLP streams.

4.6.3 Integrating Articulatory/Phonological Features

Hierarchical integration of articulatory/phonological features have been previously studied in the literature (Khan *et al.*, 2000; Morris and Fosler-Lussier, 2008). In the first stage of the above systems, an MLP is trained to estimate articulatory/phonological features. In the second stage, classifiers such as CRF (Morris and Fosler-Lussier, 2008) and RNN (Khan *et al.*, 2000) are trained to estimate the phonetic class conditional probabilities. Based on the present work, an MLP classifier could be used at the second stage of the hierarchy to estimate the phonetic class conditional probabilities, conditioned on the articulatory/phonological features.

4.6.4 Hierarchical System for Adaptation

A potential application of the MLP based hierarchical system is in adaptation. The first MLP could be trained on a generic task which covers all the phonemes in the target task. The second MLP is trained on the adaptation data corresponding to the specific task. It has already been observed that the second MLP in the hierarchy requires fewer parameters and can be trained using lesser amount of data, making it an ideal case for adaptation, especially in scenarios where the training data is limited. We investigate the application of hierarchical system for task adaptation in the following Chapter 5.

4.7 Summary and Conclusions

We investigated a simple hierarchical architecture for estimating the posterior probabilities of phonemes. The system consisted of two MLP classifiers in tandem. The first MLP is trained on standard PLP features, with a temporal context of 90 ms. The second MLP is trained on the posterior probabilities of phonemes (posterior features) estimated by the first, but with a relatively longer temporal context of around 150-230 ms. Phoneme recognition experiments on TIMIT as well

as CTS databases showed that the hierarchical system is a better estimator of the phonetic class conditional probabilities.

The posterior features are endowed with two important properties. Firstly, they are trained to be linearly separable and have a sparse distribution. Secondly, they have lesser nonlinguistic variabilities such as speaker information, noise characteristics etc in comparison with standard acoustic features. In other words, the posterior features represent the soft-decisions on the underlying sequence of phonemes, and are much simpler to classify. Consequently, the second MLP classifier can effectively learn the contextual information in the temporal trajectories of the posterior features, spanning a temporal context as long as 230 ms.

In order to unearth the phonetic-temporal patterns learned by the second MLP classifier, we applied Volterra series to model the second stage in the hierarchical system, and analyzed its first order Volterra kernels (linear part of the nonlinear system). The analysis of the linear Volterra kernels showed that the second MLP has effectively captured the acoustic confusions among the phonemes at the output of the first classifier, as well as the phonotactics of the language, as observed in the training data.

Furthermore, we demonstrated that a simpler MLP with fewer number of parameters is sufficient at the second stage in the hierarchy, and that it can be trained using lesser amount of training data. We attribute this to the salient properties of the posterior features such as lesser nonlinguistic variabilities, a sparse distribution, and linear separability.

Chapter 5

Task Adaptation

5.1 Introduction

The availability of large amounts of well transcribed (at least at the word level) speech corpora, coupled with faster and cheaper computational infrastructure has heralded an era of complex ASR systems. It is not uncommon these days to train MLP classifiers on thousands of hours of speech data in order to achieve better recognition performance. However, a direct use of these well trained MLP classifiers available off-the-shelf to new tasks or application scenarios may not always yield better performance. This is mainly because of the mismatch between the training and test data conditions of these classifiers. Therefore, adaptation of well trained MLP classifiers available off-the-shelf to new application scenarios is of practical interest.

The basic goal in task adaptation is to estimate a new acoustic model for a given in-domain task using (a) an off-the-shelf MLP classifier, which is well trained on a large amount of out-of-domain data and (b) a limited amount of data available for the in-domain task. Adaptation can be achieved by either modifying the trained parameters of the MLP or augmenting additional structures to it. When a new model is estimated using the limited amount of in-domain data, then its model parameters can have a high variance (or overfitting) if a complex model is used or a high bias if an over simplistic model is used (Li and Bilmes, 2006). On the other hand, if a well trained MLP available off-the-shelf is directly used, it can lead to suboptimal performance due to mismatched conditions. Hence, an ideal solution is to adapt the well trained model using the adaptation data.

An MLP can be viewed as a mapping from the acoustic feature space to the phonetic label space. The decoder maps the phonetic label space to the sequence of words using, among other sources of information, a pronunciation dictionary. Therefore, the mismatch between the training and test conditions of an MLP classifier can arise at two levels.

- Feature mismatch: The acoustic characteristics of the in-domain speech can differ from the out-of-domain data used to train the MLP classifier in several respects. These include the speaking style (conversations versus read speech), the channel characteristics (telephone versus microphone speech), the accents (American versus British English), dialects, etc. For example, consider the case where an MLP trained on conversational telephone speech is used to recognize read speech. Spontaneous speech is significantly different from read speech in both acoustic and linguistic characteristics (Furui, 2003; Nakamura *et al.*, 2007), and this can be a source of mismatch between the training and test conditions.

- Label mismatch: The phoneme set representing the output classes of the MLP may not be consistent with the dictionary available for the new task. Even if the same phoneme set is used, the dictionary used to force align the training transcription to obtain the phonetic labels may not be consistent with the pronunciation dictionary available for the new task.

We investigate the use of an MLP based hierarchical system for task adaptation. A well trained MLP classifier is assumed to be available off-the-shelf, and it is used at the first stage of the hierarchical system. The second MLP is trained on a long temporal context of posterior features using a small amount of adaptation data specific to the target task. We believe the mismatch between the training and test conditions of the first MLP manifests in the form of systematic perturbations in the estimated posterior features, and we expect the second MLP to learn this information on the adaptation data. This study is also motivated by the findings in Chapter 4, were we observed that at the second stage of the hierarchical system, the classifier can be simpler in terms of model parameters and that it can be trained using lesser training data.

In this chapter, we exploit an MLP trained on 232 hours of conversational telephone speech (CTS) data for isolated word recognition on the Phonebook database (Pitrelli *et al.*, 1995), where only 6.7 hours of training data is available. We also study the performance of the hierarchical adaptation system with respect to (a) the temporal context on the posterior features at the input of

the second MLP (b) the complexity of the second MLP in terms of the number of parameters, and (c) the amount of data used for adaptation, to ascertain if the trends are consistent with the findings in the previous chapter, where both the MLPs in the hierarchical system were trained on the same task.

We use the following terminology in the rest of this chapter. Matched condition refers to using only the Phonebook training set in the development of the entire system. Mismatched condition refers to using the phonetic class-conditional probabilities estimated by the CTS MLP directly in decoding. In other words, Phonebook training is not used at all. Adaptation involves either modifying the parameters of the CTS MLP or training an additional structure using the Phonebook data. In all these three cases, the task is to recognize isolated words in the Phonebook test set.

5.2 Background

The techniques developed for speaker adaptation have been shown to be useful in task adaptation as well (Gales *et al.*, 2003; Bocchieri *et al.*, 2004) in the context of HMM/GMM based acoustic modeling. Hence, in this section, we first discuss some of the commonly used speaker and task adaptation techniques in HMM/MLP based acoustic modeling.

Mean and Variance Normalization

In practical applications, the input features to the MLP are normalized to zero mean and unit variance to achieve faster convergence of the back propagation training algorithm (LeCun *et al.*, 1998). Feature normalization can be mathematically expressed as $\hat{\mathbf{x}} = \mathbf{\Sigma}^{-\frac{1}{2}} (\mathbf{x} - \boldsymbol{\mu})$, where $\mathbf{x}$ denotes the acoustic features, $\hat{\mathbf{x}}$ denotes the input to the MLP, $\boldsymbol{\mu}$ denotes the mean of the feature vector, and $\mathbf{\Sigma}$ denotes the diagonal covariance matrix. When testing in matched conditions, the feature mean and variances estimated during the training phase are directly used. In mismatched conditions, however, these statistics are reestimated on the adaptation data, and hence feature normalization can also be viewed as a simple task adaptation technique. This can help in addressing the mismatch in the acoustic feature space up to a constant shift and constant scaling. If the features are cepstral parameters, this is equivalent to global cepstral mean and variance normalization.

Linear Input Networks

Linear input networks have been previously applied for speaker adaptation in MLP based acoustic modeling (Neto *et al.*, 1995; Abrash *et al.*, 1995). In this adaptation scheme, the input $\hat{\hat{\mathbf{x}}}$ to the MLP is obtained by transforming the mean/variance normalized features $\hat{\mathbf{x}}$ as $\hat{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{b}$, where $\mathbf{A}$ is a full transformation matrix and $\mathbf{b}$ denotes a constant shift or bias. The parameters $\{\mathbf{A}, \mathbf{b}\}$ are optimized on the adaptation data using the same optimization criterion as the one that was used to train the MLP. In this process, the parameters of the trained MLP classifier are not updated. In addition to shift and scaling, the linear input network adaption can help in addressing mismatch due to rotation in the cepstral feature space.

This approach can be considered as the HMM/MLP analog of the constrained maximum likelihood linear regression (CMLLR) adaptation technique for HMM/GMM systems. In CMLLR adaptation, linear transformation in the feature space is equivalent to a linear transformation in the model space (Woodland, 1999). That is, a transformation of mean vectors and covariance matrices. In this approach, feature transformation can be viewed as augmenting an additional layer with linear activation function at the front end of the MLP.[1]

Full-Retraining Adaptation

In full-retraining adaptation method, the well trained MLP classifier is treated as initial seed model, and all its parameters are reestimated using the adaptation data. This approach has been successfully applied for speaker adaptation (Neto *et al.*, 1995), where it is commonly referred to as retrained speaker independent adaptation. Retraining all the parameters using a limited amount of adaptation can lead to overfitting problems. This problem is typically handled using an early stopping criterion.

Full retrained adaptation has also been applied for task adaptation. For instance, in the development of the ICSI-SRI system for the transcription of meetings (Stolcke *et al.*, 2005), the MLP trained on 1800 hours of CTS data was adapted for the meeting task. The problem of overtraining was avoided by performing only three additional iterations at a low learning rate. ASR experiments were performed by using the Tandem and hidden activation temporal pattern features

[1] If a temporal context of 9 frames is applied, then the input weight matrix of the MLP is a block diagonal, and is obtained by repeating $\mathbf{A}$ for 9 times along the diagonal. The parameters for all the time context values are tied and updated jointly.

extracted from the adapted MLP. This adaptation scheme was shown to help in achieving better performance.

Semi-Retraining Adaptation

In semi-retraining adaptation method, the weights connecting the input to the hidden layer are kept intact, but the weight matrix connecting the hidden layer to the output layer is retrained using the adaptation data. This approach was proposed recently (Li *et al.*, 2005), where the MLP trained for speaker independent phoneme classification was adapted for the speaker dependent classification using maximum margin criterion. In other words, the hidden-output weights were retrained in the maximum margin separation sense.

In light of the strong relationship established between MLPs and support vector machines (Collobert and Bengio, 2004), retaining the input-to-hidden weights can be interpreted as fixing the kernel and retraining the hidden-to-output weights can be compared to learning the support vectors. This approach is less susceptible to overfitting when compared to full-retraining method, as only a small percentage of the weights are retrained.[2] In a more recent extension to this work, the problem of overfitting is explicitly handled by regularized adaptation, which penalizes the distance between the unadapted and adapted models (Li and Bilmes, 2006).

Hierarchical Adaptation

In Chapter 4, we analyzed the MLP based hierarchical system where both the classifiers were trained on the same task. The hierarchical system was shown to yield higher phoneme recognition accuracies in comparison with the single MLP based conventional approach. The analysis showed that the second MLP effectively learns the contextual information in the posterior feature space. Furthermore, we also showed that a simpler classifier in terms of model complexity is sufficient at the second stage and that it can be trained using a lesser amount of training data.

In this chapter, we investigate if the hierarchical approach can be exploited in task adaptation. An off-the-shelf MLP classifier which is well trained on a large amount of out-of-domain data is used at the first stage of the system. The second MLP is trained on the posterior features estimated

[2]If the architecture of the three layered MLP is $I \times H \times O$, and the number of hidden nodes H is sufficiently large, the percentage of trainable parameters in the MLP is approximately given by $100 * O/(I+1)$. For 9 frames PLP features, $I = 351$ and a typical phoneme set $O = 45$, this percentage is about 13%.

on the in-domain (or adaptation) data. If the mismatch in the training and test conditions of the MLP classifier manifests in systematic and consistent perturbations in the estimated posterior features, then the second classifier can learn the relationship on the adaptation data. In addition, this approach can also benefit by exploiting the contextual information in the sequence of posterior features (captured by a long temporal context). Furthermore, as the posterior features are simpler and carry lesser nonlinguistic variability, we expect the second classifier to be simpler and to be able to be trained using a lesser amount of data.

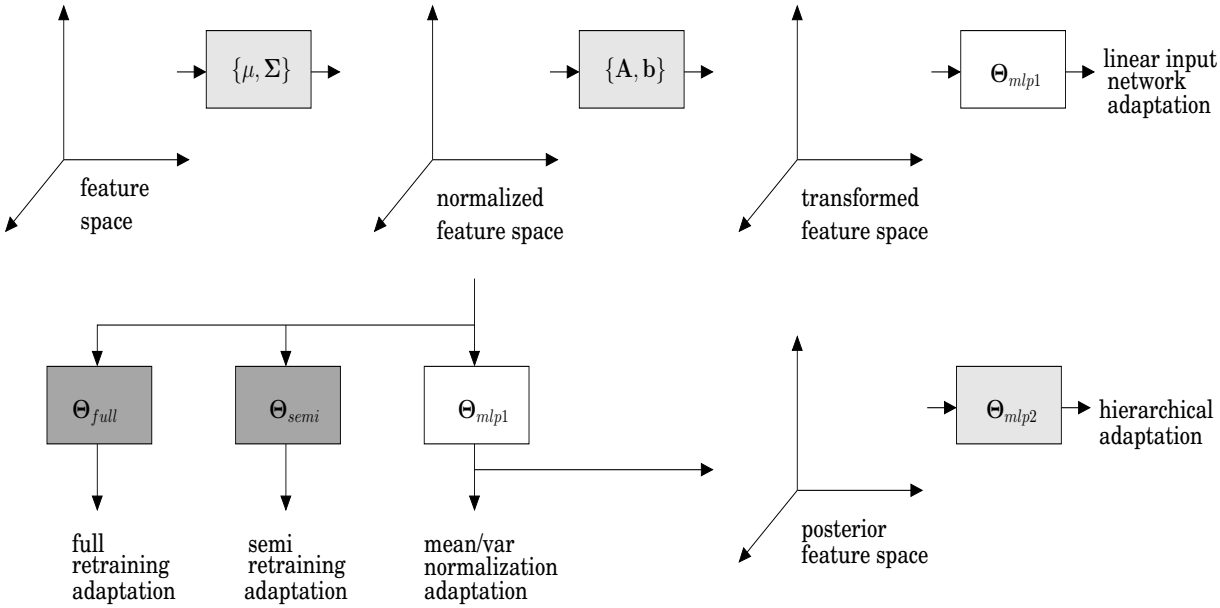


Figure 5.1. The flow diagram of all the adaptation schemes discussed in this section. Trained models (transforms or MLPs) are represented by rectangles. If the model parameters are trained on the adaptation data (in-domain), the rectangles are lightly shaded. If the model parameters are on out-of-domain data, the rectangles are not shaded. If the parameters are trained on both in-domain and out-of-domain data, the rectangles are darkly shaded.

Figure 5.1 is flow diagram illustrating the various adaptation schemes discussed so far. The statistics of the features $\{\mu, \Sigma\}$ are estimated on the adaptation data in the maximum likelihood sense. The parameters of the linear input network $\{\mathbf{A}, \mathbf{b}\}$ are estimated in a discriminative fashion using the cross-entropy criterion that was used to train off-the-shelf MLP classifier Θ_{mlp1}. Both these methods are linear transformations in the acoustic feature space. In the hierarchical adaptation, the second MLP classifier Θ_{mlp2} is trained on the posterior features estimated by the first classifier. This approach can be seen as adaptation via nonlinear transformation in the posterior feature space. The full retrained model Θ_{full} and semi retrained model Θ_{semi} is adapted from the

off-the-shelf MLP Θ_{mlp1} using acoustic features on the in-domain adaptation data.

It can be seen that the semi-retraining adaptation as well as the hierarchical adaptation method can handle the scenario where the phonetic transcription and pronunciation dictionary on the target task is not consistent with the phoneme set of the trained off-the-shelf MLP.

5.3 Experimental Setup

Phonebook Task

Experiments are performed on the Phonebook task, which was designed for speaker independent isolated word recognition over the telephone channel. The test set consists of 6598 utterances from 96 speakers. It is made up of eight subsets, each containing 75 unique words. Isolated word recognition is performed on the test set by following the two protocols as defined in (Dupont *et al.*, 1997).

- **75-lexicon:** Recognition is performed on each of the eight subsets separately by using dictionaries specific to the subset. The average perplexity of the task is 75, which is the size of the dictionary. The word error rate (WER) reported on this protocol is the average across all the eight subsets.

- **600-lexicon:** A common pronunciation dictionary, consisting of 600 words, is used across all the eight subsets in the test sets. This is a harder task when compared to 75-lexicon protocol, with a perplexity of 600.

Phonebook Training Resources

The training set consists of 19421 isolated utterances from 243 speakers, which amounts to about 6.7 hours of speech[3], and the cross-validation set consists of 7920 utterances from 106 speakers. The training and cross-validation sets are as defined in (Dupont *et al.*, 1997). There are no common words shared between the training, validation and test sets of the corpus. The database is distributed with a pronunciation dictionary consisting of 42 phonemes. The phonetic transcription

[3] 32% of the training data is silence.

required for training the MLP is obtained by forced alignment. For this, we use previously trained HMM/GMM based acoustic models and the Phonebook pronunciation dictionary.

Off-the-shelf CTS MLP

For task adaptation studies, we use an off-the-shelf MLP which was trained on 232 hours of conversational telephone speech. This MLP was used for phoneme recognition studies in the Chapter 4. The acoustic features consist of the first 13 PLP cepstral coefficients. After speaker specific mean and variance normalization, dynamic cepstral (delta and delta-delta) coefficients are appended to the base features to obtain a 39 dimensional feature vector for every 10 ms of speech. These features are applied at the input of the MLP with a temporal context of 90 ms. The output classes of the MLP represent the 45 phonemes in the UNISYN pronunciation dictionary (Fitt, 2000). The architecture of the MLP is $351 \times 5000 \times 45$. The phonetic transcription for training the MLP was obtained via forced alignment as discussed in Section 4.3.

Adaptation Scheme

As both Phonebook and CTS contain telephone speech, the channel mismatch can be assumed to be minimal. Nonetheless, there can be a significant difference in the characteristics of the acoustic features. The conversational speech is characterized by higher speaking rate, higher variance of the cepstral parameters and the phenomenon of spectral reduction (Furui, 2003; Nakamura *et al.*, 2007). In addition, Phonebook is a constrained isolated word recognition task, and the prior distribution of phonemes differs from the unconstrained CTS task.

Figure 5.2 (a) shows the use of the off-the-shelf MLP in mismatched conditions. The posterior probabilities of phonemes estimated by the MLP cannot be directly used in the recognition as its output phonetic classes are not consistent with the pronunciation dictionary available on the target task.

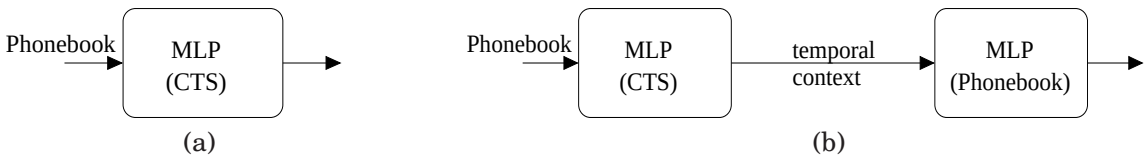


Figure 5.2. The adaptation scheme. (a) Mismatched conditions (b) Mismatched conditions plus adaptation.

Figure 5.2 (b) shows the hierarchical adaptation scheme. In the previous chapter, a second MLP classifier was trained on the estimated posterior features with temporal context of 230ms. This context was found to be optimal in the recognition of phonemes. There are two important changes in the hierarchical system for recognition of words. It was found empirically that log-posterior features yield lower error rates and that a temporal context of around 130-150 ms is optimal (Ketabdar and Bourlard, 2008; Pinto *et al.*, 2009b). We discuss the effect of temporal context on the error rates in Section 5.4.1.

We compare the performance of the hierarchical adaptation technique to two other works in the literature, which in this thesis are referred to as full-retraining adaptation (Stolcke *et al.*, 2005) and semi-retraining adaptation (Li *et al.*, 2005). In full-retraining adaptation, the weights and biases of the CTS MLP are taken as the initial model and additional iterations of the backpropagation training is performed until convergence. The early stopping criterion is applied to prevent overtraining.

In the semi-retraining approach, the input-to-hidden weights and the hidden bias of the CTS MLP are kept intact, but the hidden-to-output weights and the output biases are adapted using the minimum cross-entropy training criterion. The Quicknet software was modified to update only the hidden-to-output parameters of the MLP. If the phonetic transcription on the adaptation data is consistent with the output classes of the CTS MLP, its hidden-to-output weights are taken as the initial model. If a new phonetic transcription is used, then the weights are initialized randomly.

Modeling and Decoding

A three-layered MLP with a sigmoid nonlinear activation function at the hidden layer, and a softmax activation function at the output layer is used throughout the studies. The parameters of the MLP are trained using the minimum cross entropy error criterion. The input features to the MLP are normalized to zero mean and unit variance. In matched conditions, these statistics are estimated on the training data. However, in adaptation studies, where an MLP was trained on conversational speech, the mean and variances are reestimated on Phonebook data.

The HMM/MLP hybrid approach is used for decoding. Each phoneme is modeled by a three-state, strictly left-to-right HMM, thereby enforcing a minimum duration of 30 ms. The (scaled) emission likelihood in each of the three states is the same, and is obtained by normalizing the esti-

mated phonetic class conditional probabilities by the respective class priors. The Viterbi algorithm is applied with a simple loop-of-words language model.

5.4 Results and Analysis

Experiments are performed by using two sets of pronunciation dictionaries: (a) Phonebook dictionary, which is distributed with the Phonebook database and (b) UNISYN dictionary, which was used in the training of the CTS MLP. The results are reported in terms of word error rate (WER).

Decoding with Phonebook dictionary

Table 5.1 shows the WER obtained by HMM/MLP hybrid decoding on the Phonebook test set with 75-lexicon and 600-lexicon decoding protocols using the Phonebook pronunciation dictionary. Due to the differences in the phoneme sets, recognition cannot be performed in mismatched conditions, and the full-retraining adaptation scheme cannot be applied. In the hierarchical adaptation method, we train a second MLP classifier on the log posterior features estimated by the CTS MLP with a temporal context of 130 ms. The table also shows the error rates obtained in matched conditions. That is, by training the system on 6.7 hours of speech from the Phonebook training set. The phonetic class-conditional probabilities are estimated in two ways (a) the baseline single MLP based approach and (b) using a hierarchical system, where the second MLP is trained on log posterior features estimated by the baseline system with a temporal context of 130 ms.

test protocol	mismatched conditions	adaptation			matched conditions	
		full-retraining	semi-retraining	hierarchical	baseline	hierarchical
75-lexicon	-	-	1.0	**0.5**	1.2	0.9
600-lexicon	-	-	3.0	**1.8**	4.0	3.3

Table 5.1. Word error rates on the Phonebook test set in matched conditions and using adaptation techniques. The Phonebook pronunciation dictionary is used in the decoding.

The following are the observations from the results

- Under matched conditions, on both 75-lexicon as well as 600-lexicon protocols, the hierarchical system yields a lower error rate when compared to the baseline single MLP based system. This demonstrates the effectiveness of the hierarchical approach to estimating phonetic class-conditional probabilities in recognition of words.

- Both adaptation schemes yield better performance when compared to matched conditions. This is not surprising as the off-the-shelf MLP is trained on 232 hours of speech, whereas in the matched conditions, only 6.7 hours of speech is available for training. Furthermore, in the proposed adaptation method, there is an additional advantage in using the hierarchical system as contextual information can be exploited (Pinto *et al.*, 2008, 2009a). The hierarchical adaptation approach yields the lowest error rate of 1.8%, which is, to the best of our knowledge, the lowest error rates to be reported on this particular Phonebook task. These results clearly demonstrate how well trained MLP classifiers available off-the-shelf can be exploited for new applications where the training data is limited.

Decoding with UNISYN dictionary

A clearer understanding of the performance of the system in mismatched conditions and different adaptation techniques can be obtained by using a pronunciation dictionary in decoding that is consistent with the output classes of the CTS MLP. The UNISYN American English pronunciation dictionary consists of over 120K entries. Despite this, about 17% (about 500) of the words present in the entire Phonebook (train, validation, and test) corpus were not found in the dictionary. The pronunciation lexicon for these words were hand crafted by carefully studying the main dictionary. The phonetic transcription for training the MLP classifiers is obtained by force aligning the isolated utterances to the sequence of phonemes obtained from the newly created dictionary. Forced alignment is performed using the HMM/MLP hybrid approach, where the posterior probabilities of phonemes estimated by the CTS MLP are used as the local acoustic scores.

test protocol	mismatched conditions	adaptation			matched conditions	
		full-retraining	semi-retraining	hierarchical	baseline	hierarchical
75-lexicon	1.2	0.7	0.6	**0.5**	1.2	1.1
600-lexicon	3.8	2.6	2.2	**2.0**	4.5	3.5

Table 5.2. Word error rates on the Phonebook test set in mismatched, matched, and adaptation conditions. The UNISYN pronunciation dictionary is used in decoding.

Table 5.2 shows the error rates in mismatched, matched, and adaptation conditions obtained using the UNISYN pronunciation dictionary. In mismatched conditions, the means and variances are estimated on the adaptation data and hence, in a strict sense, this is an example of mean/variance adaptation scheme. As observed previously, the hierarchical approach yields lower error rates when

compared to the baseline system in matched conditions. The following are some of the key observations from the table.

- It is interesting to see that on the 600-lexicon protocol, the error rate obtained in mismatched conditions is lower than in matched condition. This indicates that the mismatched condition is compensated to a large extent by the benefits of well trained MLP. Apart from the amount of training data, the CTS MLP is also trained on a large number of phonetic contexts as explained below.

  Although context independent phonetic models are trained, a temporal context of 90 ms enables the MLP to implicitly learn the context dependent information to a certain extent. The Phonebook test set consists of 600 words and each word, on average, consists of about 7 phonemes. There are about 1700 unique phonetic triphone contexts in the test set which are unseen in the Phonebook training set. In contrast, the number of triphone contexts unseen in the CTS training set is only about 300. In other words, the CTS MLP is well trained not only because of the amount of data, but also due to the larger number of phonetic contexts.

- It can be seen that in the case of using UNISYN pronunciation dictionary, the difference between semi-retraining method and the hierarchical adaptation is only about 0.2%. On the other hand, in the case of Phonebook dictionary, the difference was about 1.2%. A possible explanation is that the hidden representation is optimal for the UNISYN phoneme set, and the hidden-to-output weights are trained with the same phoneme set.

- The error rates obtained in matched conditions are higher than the corresponding error rates obtained using the Phonebook dictionary. This indicates that the Phonebook dictionary is slightly better than the UNISYN dictionary in terms of modeling the pronunciations of the spoken words.

In the following sections, we study the performance of the system with respect to (a) the temporal context on the posterior features (b) goodness of the posterior features (c) complexity of the second MLP and (d) size of the adaptation data. We report the results only for the 600-lexicon test protocol as the trends in results are similar on the 75-lexicon protocol. We use the Phonebook pronunciation dictionary as it yields the best results.

5.4.1 Role of Temporal Context

In Fig. 5.3, we plot the word error rate obtained on the 600-lexicon task as a function of temporal context applied on the log posterior features in the hierarchical adaptation technique. The posterior features are estimated using the CTS MLP trained on 232 hours of speech and the second MLP is trained on 6.7 hours of adaptation data. The horizontal dashed line indicates the word error rate obtained by the baseline system in matched conditions, trained on the same amount of data.

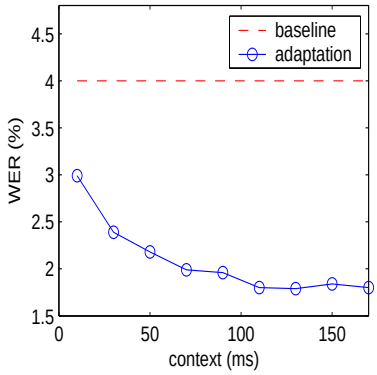


Figure 5.3. Word error rate on the 600-lexicon protocol as a function of the temporal context at the input of the second MLP. The horizontal dashed line indicates the WER obtained from the baseline system in matched conditions.

It can be seen that even without any temporal context, we obtain an absolute reduction of 1% in the error rate over the baseline system in matched conditions. In the hierarchical system without any temporal context, the second MLP can be viewed as a local mapping between the phonemes in the UNISYN dictionary to the Phonebook dictionary. The second MLP could be correcting any systematic perturbations in the estimated posterior probabilities due to the mismatch in the dictionaries.

As the temporal context is increased, the error rate drops further and saturates by around 130-150 ms. With increase in temporal context, the second MLP classifier is also able to capture the contextual information in the posterior features.

5.4.2 Complexity of the second MLP

A well trained MLP classifier attempts to achieve linear separability in the estimated posterior feature space (refer Section 4.5.1 in Chapter 4). The degree to which it actually achieves linear separability depends on the complexity of the task. In addition, the posterior features are sparsely

distributed. As a consequence, the second MLP classifier can be simpler in terms of the number of parameters and this was demonstrated in phoneme recognition studies in Section 4.5.2.

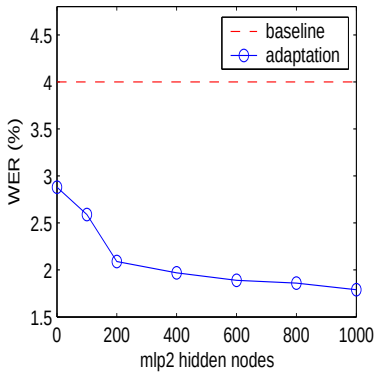


Figure 5.4. The word error rate on the 600-lexicon task as a function of the size of the hidden layer of the MLP. The temporal context on the posterior features is fixed to 130 ms. The WER obtained by using a single layer perceptron is plotted as the number of hidden nodes equals zero.

In Fig. 5.4, we plot the word error rate on the 600-lexicon task as a function of the size of the hidden layer at the second stage of the hierarchical system. The size of the hidden layer controls the amount of nonlinearity that the MLP can model. It can be seen from the plot that the fall in the performance is minimal as the size of the hidden layer is reduced from 1000 to 200 units. As the size is reduced further, the performance drops more sharply. However, the adaptation system still outperforms the baseline system in matched conditions. As an extreme case, a single layer perceptron is used at the second stage of the hierarchy, and this is plotted as the number of hidden nodes equals zero in the figure. As seen in the figure, a simple linear classifier yields an absolute reduction of 1.1% in the error rate over the baseline single MLP system. This observation is consistent with our previous study (Pinto *et al.*, 2009a), where lower phoneme error rates were obtained even when an SLP was used at the second stage of the hierarchy.

5.4.3 Amount of Adaptation Data

In Chapter 4, we discussed that an MLP trained on a large amount of data from a diverse population of speakers and different noise and channel conditions can achieve invariance to speaker as well as environmental conditions. As a consequence of this property, we argued that the second MLP could be trained on a limited amount of training data, and experimental results confirmed this. In this section, we confirm that this argument holds water even in the case of the hierarchical adaptation

scheme.

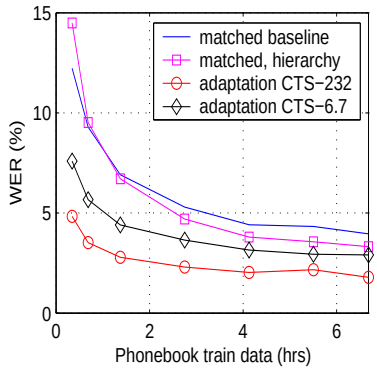


Figure 5.5. The word error rate as a function of the amount of adaptation data (Phonebook) used. A temporal context of 130 ms is considered in the case of the hierarchical system.

In Figure 5.5, we plot the word error rate obtained on the 600-lexicon task as a function of the amount of Phonebook data used for training or adaptation. The hierarchical systems are trained with a temporal context of 130 ms. The plots in the figure correspond to the following four systems.

Matched baseline: The phonetic class-conditional probabilities are estimated by using an MLP, which is trained in matched conditions using PLP features. It can be seen that the performance of the system falls sharply with the reduction of training data. By using only 20 minutes of training data, we obtain a word error rate of 12%.

Matched hierarchical: A second MLP classifier is trained on the log posterior features estimated by the baseline system with a temporal context of 130 ms. The second MLP is trained with the same amount of Phonebook speech that was used to train the first MLP. It can be seen that the hierarchical system consistently yields lower error rates when compared to the baseline system. However, as the training data is further reduced, the hierarchical system ceases to show improvements over the baseline system.

Adaptation CTS-232: In this adaptation system, the posterior features on the Phonebook task are estimated using an MLP which is trained on 232 hours of CTS data. It can be seen that this system yields the lowest error rates. With just 30 minutes of adaptation, the hierarchical system yields an error rate about 4%, which is same as the baseline system trained on 6.7 hours of speech in matched conditions. If the baseline system is trained on 30 minutes, then the error rate is about 9%. This is because of the variability in the acoustic features which need comparatively larger training data.

Adaptation CTS-6.7: In this adaptation system, the first MLP classifier is trained using 6.7 hours of CTS. It can be seen that in this case, 2 hours of adaptation yields the same performance as 30 minutes of adaptation on CTS-232 system. To briefly summarize, if the first MLP in the hierarchical system is trained using a larger amount of data, then smaller amount of adaptation data is sufficient. Furthermore, the difference between the word error rates obtained from CTS-232 system and CTS-6.7 system is larger when the adaptation data is limited, and this gap reduces with the increase in the amount of adaptation data.

5.5 Summary and Conclusions

In this chapter, we discussed the MLP based hierarchical approach for task adaptation in ASR. The second MLP classifier can be viewed as a mapping of a trajectory in the posterior feature space corresponding to CTS phonemes to a point in the posterior feature space corresponding to the Phonebook phonemes. The main conclusions of this chapter can be summarized as follows:

- The hierarchical approach to estimating the phonetic class-conditional probabilities is useful in word recognition in matched conditions. The previous chapter showed its effectiveness in recognition of phonemes. Extensive experiments on large vocabulary continuous speech recognition in Mandarin are presented in the following chapter.

- If the off-the-shelf MLP classifier is trained on a large amount of data, then a lesser amount of adaptation data is sufficient. This is interesting because it allows us to reuse well trained MLP classifiers on new tasks, where the amount of training data is limited.

- The performance of the hierarchical adaptation approach increases with the temporal context on the posterior features, and saturates at about 130 ms. In recognition of phonemes, we observed that a context of 230 ms was optimal.

- The second classifier can be simpler in terms of both the structure and the number of parameters. In fact, even a single layered perceptron yielded lower error rate in comparison with the baseline system in matched conditions. This is consistent with the observations in the previous chapter.

Chapter 6

ASR in Mandarin

6.1 Introduction

In Chapter 4, we showed that the MLP based hierarchical acoustic modeling yields higher phoneme recognition accuracies when compared to the conventional single MLP based approach. In Chapter 5, we demonstrated the effectiveness of this approach in small vocabulary isolated word recognition. In this chapter, we investigate the hierarchical system for large vocabulary continuous speech recognition. For this, we use the Mandarin database developed under the Global Autonomous Language Exploitation (GALE) project.[1] It consists of audio segments acquired from various television programs broadcast in Mandarin. The broadcast segments includes two types of genres, namely broadcast news (BN) and broadcast conversations (BC).

The primary objective of this work is to confirm the usefulness of the MLP based hierarchical system in large vocabulary continuous speech recognition. In addition, the experimental setup also allows us to further evaluate the hierarchical system in the following aspects.

- The hierarchical system is tested on a challenging real-world application scenario as the broadcast programs are from a wide range of domains which include informal and colloquial language. In addition, the experimental setup also allows us to test the hierarchical approach for a new language.

[1] http://www.darpa.mil/ipto/programs/gale/gale.asp

- In Chapter 5, we demonstrated the application of the hierarchical approach for task adaptation. An MLP trained on a large amount of out-of-domain data is used at the first stage of the hierarchical system and the second MLP is trained on the in-domain adaptation data. In this chapter, we investigate the hierarchical system for genre adaptation. The goal is to exploit additional data in one genre (*e.g.,* broadcast news) in the development of the ASR system for the other genre (*e.g.,* broadcast conversations).

- In all the experiments so far, recognition is performed using the hybrid HMM/MLP decoding paradigm, where the MLP is used to estimate the scaled likelihood of feature vectors in the context independent HMM states. This approach is simple yet effective in phoneme recognition or small vocabulary isolated word recognition. For large vocabulary speech recognition, it is advantageous to use state-of-the-art modeling techniques such as context dependent modeling, state tying and speaker adaptation. To this end, the Tandem approach provides an effective solution as the posterior features estimated by the MLP can be processed and used as features input to the HMM/GMM system in the same way as standard acoustic features.

6.2 Hierarchical Tandem System

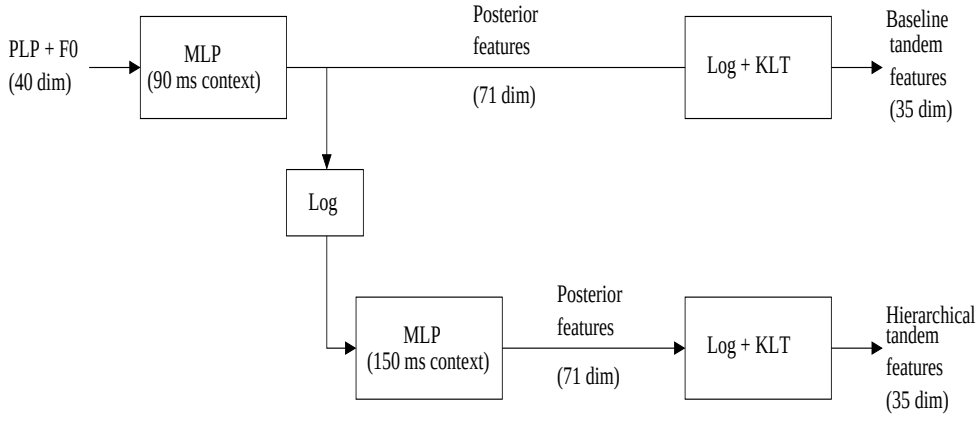


Figure 6.1. (a) The standard Tandem feature extraction technique (b) Hierarchical Tandem feature extraction technique with a temporal context of 150 ms on the posterior features.

Figure 6.1 shows the block schematic of the standard (or baseline) Tandem feature extraction as well as the hierarchical Tandem feature extraction. The input features to the first MLP consists of the first 13 PLP cepstral coefficients appended to their delta and delta-delta parameters, resulting

in a 39 dimensional feature vector. Since Mandarin is a tonal language, appending a smoothed estimate of the log-pitch value to the cepstral features has been found to be useful (Lei *et al.*, 2006). The 40 dimensional combined feature vector is applied at the input of the MLP with a temporal context of 90 ms. The size of the output layer of the MLP is 71, corresponding to the number of phonemes.

In the case of the baseline Tandem system, the output of the MLP is transformed using a logarithm, followed by Karhunen Leove transformation (KLT) and dimensionality reduction to obtain a 35 dimensional feature vector. This dimension is chosen such that at least 95% of the variance in the data is covered. In the hierarchical Tandem system, a second MLP classifier is trained on the log posterior features estimated by the first MLP with a temporal context of 150 ms. This temporal context is based on the findings from the task adaptation study reported in Chapter 5, where it was observed that the word error rates saturate for a context of around 130 ms - 150 ms (refer Figure 5.3). The output of the second MLP is transformed in the same way as the baseline system to obtain the hierarchical Tandem features.

6.3 Experimental Setup

In this section, we describe the experimental setup for the Mandarin ASR system.

Training and Test Data Definition

The training corpus consists of 95 hours of speech, which includes 50 hours of BN and 45 hours BC data. It is a subset of the training set of the 2008 SRI Mandarin speech-to-text system (Lei *et al.*, 2009). More specifically, it is obtained by excluding the TDT4 corpus from the GALE Year 1 training corpus. The snippet level genre classification on the training set was provided by Stanford Research Institute (SRI) using the genre classifier developed at the University of Washington under the GALE project (Marin *et al.*, 2009; Wang *et al.*, 2009). We use the GALE eval06 data as the test set. The genre labels on the test set are provided by the Linguistic Data Consortium. The word and phonetic transcription for the training data was obtained from SRI.

Mandarin ASR System

We use the Mandarin ASR system developed by researchers from SRI, University of Washington, and International Computer Science Institute (ICSI) under the GALE project using the SRI Decipher system (Lei *et al.*, 2006; Hwang *et al.*, 2009; Lei *et al.*, 2009).

We use the experimental setup described in a recent work (Valente *et al.*, 2009). We briefly describe the ASR system here. Speech-silence segmentation and automatic speaker clustering is first performed using Gaussian mixture modeling technique to derive "auto speakers". The vocal tract length normalization factors are estimated for each auto speaker and are used in the estimation of MFCC features (Hwang *et al.*, 2006).

The acoustic modeling is based on the standard HMM/GMM technique. In the training phase, context independent models are first trained for each of the 71 phonemes. Context dependent models are subsequently trained and clustered down to 2000 shared Markov states, which are also known as senones. Each senone is modeled using a mixture of 32 Gaussians using phonetic decision tree based clustering (Hwang *et al.*, 1993). The acoustic model parameters are trained using the simple maximum likelihood criterion. Cross-word triphone modeling and speaker adaptive training is not performed in this study.

A trigram language model, which was estimated using an assortment of text corpora totalling over a billion words (Hwang *et al.*, 2006) was used for this study. The pronunciation dictionary consists of 60K characters, and is transcribed using 70 phonemes. A silence class was added, resulting in a total of 71 output classes. The decoding/testing phase involves two passes:

1. First pass search: A maximum likelihood decoding is performed using a trigram language model and the trained acoustic model to obtain an one best hypothesis for each utterance. The system is referred to as the speaker independent system.

2. By using one best recognition hypothesis, the silence, vowel, and consonant regions are first identified. The constrained maximum likelihood linear regression transformation matrices are then estimated for each auto speaker. The features are subsequently transformed using these matrices.

3. Second pass search: A maximum likelihood decoding is again performed using the transformed features and the same acoustic model that was used in the first pass decoding. As

feature transforms are estimated on a per speaker basis, the second pass system is referred to as the speaker adapted system.

The tunable parameters of the system, namely the language model scaling factor and the Gaussian scaling factor were fixed based on previous study (Valente *et al.*, 2009).[2]

Methodology

The HMM/GMM system is trained using three sets of features:

- mfcc-f0-42: The static feature vector consists of first 13 MFCC coefficients along with an estimate of the log pitch value (f0). The static features are appended to their first and second order temporal derivatives to obtain a 42 dimensional feature vector.

- tandem-35: The phoneme posterior probabilities estimated by the MLP classifier are transformed using logarithm and KLT, followed by dimensionality reduction to obtain a 35 dimensional feature vector. The tandem features are estimated in the conventional way using a single MLP classifier or the hierarchical approach as discussed in Figure 6.1.

- mfcc-f0-tandem-77: Motivated from previous studies (Morgan *et al.*, 2005), we also investigate an augmented feature vector, i.e., concatenation of mfcc-f0-42 and tandem-35 features.

In order to differentiate between Tandem features estimated by standard single MLP approach and hierarchical approach, we refer to the features with a prefix "baseline" and "hierarchical", respectively. For instance, baseline tandem-35 refers to tandem feature estimated by the standard single MLP approach. The standard acoustic features such PLPs and MFCCs were estimated with a frame size and a frame shift of 25.6 ms and 10 ms, respectively. The PLP features were estimated using HTK and the pitch features were obtained from ICSI.

When data from both the genres are used for training the acoustic models, there is clearly an advantage in having a larger amount of data. However, there is also the disadvantage of having mismatched acoustic conditions in the training corpus. To understand this aspect, we investigate these features along the following lines.

[2]For systems using Tandem based features, the language scaling factor was set to 6.5 and the Gaussian scale factor was set to 0.3, whereas for systems using MFCC and pitch features, these factors were set to 7.0 and 0.7 respectively.

- Genre independent system: All the components of the acoustic model, *i.e.,* the MLP classifiers and the HMM/GMM models are trained using data from both BC and BN genres. The recognition results are reported separately for the two genres.

- Genre specific system: Two separate sets of acoustic models are trained for each of the genre. In this case, the training and test genre conditions are matched.

- Genre adaptive system: This system is motivated by our findings in Chapter 5, where the MLP based hierarchical system was used for task adaptation. This system is applicable to only hierarchical Tandem features. The first MLP classifier is trained using the data from both genres. The remaining components of the acoustic model, *i.e.,* the second MLP and the HMM/GMM system are trained on data specific to the target genre.

The size of the hidden layers was chosen such that the total number of parameters is roughly equal to 5% of the training samples. As a result, MLP classifiers in the genre independent system have a higher number of parameters when compared to the genre dependent system. The MLP classifiers were trained at Idiap Research Institute using the Quicknet toolkit. The training and decoding of the ASR system was performed at ICSI, Berkeley using the SRI Decipher system.[3]

6.4 Experimental Results

In the following sections, we discuss the results obtained on the eval06 test set using genre independent, genre specific and genre adaptive systems. The results are reported in terms of character error rate (CER) for speaker independent (SI) and speaker adaptive (SA) systems. The lowest CER obtained is highlighted in boldface.

Genre Independent System

Table 6.1 shows the character error rates on the eval06 dataset obtained using mfcc-f0-42 features, baseline tandem-35 features, and hierarchical tandem-35 features for the genre independent system. The results are reported for the individual genres as well as the entire test set.

[3]We gratefully acknowledge SRI for allowing to use the Decipher ASR system and ICSI for the computational infrastructure. We also thank Wen Wang from SRI and Suman Ravuri from ICSI for helping us with the experimental setup.

Features	BC genre		BN genre		Both genres	
	SI (%)	SA (%)	SI (%)	SA (%)	SI (%)	SA (%)
mfcc-f0-42	33.4	31.0	20.9	19.3	27.0	25.0
baseline tandem-35	34.4	32.7	19.5	17.9	26.8	25.1
hierarchical tandem-35	**31.3**	**29.9**	**18.0**	**16.8**	**24.5**	**23.2**

Table 6.1. Character error rates obtained on the genre independent system using mfcc-f0-42, baseline tandem-35, and hierarchical tandem-35 features.

It can be seen from the table that the hierarchical tandem-35 features yield the lowest CER on both broadcast news as well as broadcast conversations. These results clearly demonstrate the effectiveness of the MLP based hierarchical acoustic modeling in large vocabulary continuous speech recognition. The other main observations from this study are the following:

- The error rates on the BC genre are significantly worse when compared the BN genre. This has been also previously observed in the literature (Wang *et al.*, 2009). Recognition on the BC genre is significantly harder when compared to BN because of two main reasons. Firstly, the conversational speech is spontaneous in nature and characterized by variable speaking rate, spectral reduction, [4] mispronunciations, false starts, repeated words, filled pauses, hesitations, and disfluencies. Secondly, the BC programs span a wide range of domains, which include political, economical, and cultural topics in China and around the world. In addition, the language model, which is estimated from text is more closer to the broadcast news than conversations.

- On broadcast conversations, the mfcc-f0-42 features yield a lower CER when compared to the baseline tandem-35 features. On broadcast news, the opposite trend is observed. On the entire test set, the mfcc-f0-42 and tandem-35 features yield more or less the same performance.

- On the BC genre, the hierarchical yields an absolute decrease of 3.1% on the speaker independent system, whereas on the BN genre, the decrease in CER is about 1.5%. A similar observation was also made in the recognition of phonemes in Table 4.3. On TIMIT, the hierarchical approach resulted in an absolute increase of 3.5% in the phoneme accuracy over the baseline single MLP based system. In the case of CTS, the improvement in the recognition accuracy was about 9.0%.

[4] When compared to read speech, the mean cepstral feature vector of a phoneme in conversational speech is closer to the global mean (Nakamura *et al.*, 2007). In addition, the variance of the cepstral coefficients is higher in spontaneous speech.

- The decrease in CER obtained by using the hierarchical tandem-35 features over the baseline tandem-35 features is slightly higher in the case of speaker independent decoding when compared to the speaker adaptive decoding. It can be seen that on the BC (BN) genre, the decrease in CER is about 3.1% (1.5%) on the speaker independent system, whereas on speaker adapted system, the decrease is about 2.8% (1.1%).

Table 6.2 shows the CER obtained for the genre independent system using the baseline mfcc-f0-tandem-77 and hierarchical mfcc-f0-tandem-77 features.

Features	BC genre		BN genre		Both genres	
	SI (%)	SA (%)	SI (%)	SA (%)	SI (%)	SA (%)
baseline mfcc-f0-tandem-77	29.2	28.0	17.6	16.6	23.3	22.2
hierarchical mfcc-f0-tandem-77	**28.4**	**27.3**	**17.0**	**16.3**	**22.5**	**21.7**

Table 6.2. Character error rates obtained on the genre independent system using baseline mfcc-f0-tandem-77 and hierarchical mfcc-f0-tandem-77 features.

The important observations from the table are as follows:

- A reduction in error rates is observed when mfcc-f0-42 features are augmented with baseline tandem-35 features, and also when mfcc-f0-42 features are augmented with hierarchical tandem-35 features. This shows that the hierarchical tandem-35 and mfcc-f0-42 features bear complimentary information in the same way baseline tandem-35 and mfcc-f0-42 features.

- The improvement in performance obtained by using hierarchical Tandem features over the baseline Tandem features is reduced when these features are augmented with mfcc-f0-42 features. This suggests that the improvement in recognition accuracies obtained by feature concatenation and hierarchical processing are not exactly additive. Nonetheless, the hierarchical mfcc-f0-tandem-77 features yield the lowest error rates in both the BC and BN genres.

Genre Specific and Genre Adaptive Systems

Table 6.3 shows the CER obtained by using mfcc-f0-42, baseline tandem-35, and hierarchical tandem-35 features on the genre specific system and the genre adaptive system.

The important observations from the table are as follows:

- The hierarchical tandem-35 features yield the lowest error rates when compared to mfcc-f0-42 and baseline tandem-35 features. This is consistent with the results in genre independent

Training	Features	BC genre		BN genre	
		SI (%)	SA (%)	SI (%)	SA (%)
genre specific	mfcc-f0-42	33.5	30.9	20.9	19.4
	baseline tandem-35	35.2	33.3	20.3	18.7
	hierarchical tandem-35	31.8	30.5	19.3	17.8
genre adapted	hierarchical tandem-35	**31.0**	**29.5**	**18.8**	**17.5**

Table 6.3. Character error rates obtained for genre specific systems using mfcc-f0-42, baseline tandem-35, and hierarchical tandem-35 features and on genre adaptive system using hierarchical tandem-35 features.

system.

- On mfcc-f0-42 features, the CER of genre specific system (Table 6.3) is more or less same as those obtained for the genre independent system (Table 6.1). However, for both baseline tandem-35 and hierarchical tandem-35 features, the genre specific system yields higher error rates when compared to the genre independent system. This could be attributed to the lesser amount of training data for the genre specific systems.

- On broadcast conversations, the best result is obtained for the genre adaptive system. That is, by training the first MLP with 50 hours of additional data from the BN genre, we obtain an absolute reduction of about 1% in the CER. It can also be noted that the error rates obtained by genre adaptive system is lower in comparison to genre independent system (Table 6.1). Based on these observations it can be argued that on broadcast conversations, the tradeoff between additional data and mismatched conditions is best handled by the genre adaptive system.

- On broadcast news, the genre adaptive system yields lower error rates when compared to the genre specific system. However, these error rates are higher in comparison with the genre independent system. This suggests that on the BN genre it is advantageous to train all the components of the acoustic model using the entire training set.

Table 6.4 shows the CER obtained by using baseline mfcc-f0-tandem-77 and hierarchical mfcc-f0-tandem-77 features on genre specific and genre adapted systems. On broadcast conversations, the genre adapted system yields the best performance, whereas on broadcast news the genre independent system gives lower error rates.

System	Features	BC genre		BN genre	
		SI (%)	SA (%)	SI (%)	SA (%)
genre specific	mfcc-f0-42	33.5	30.9	20.9	19.4
	baseline mfcc-f0-tandem-77	30.3	28.8	18.2	17.2
	hierarchical mfcc-f0-tandem-77	29.2	27.7	17.8	16.7
genre adaptive	hierarchical mfcc-f0-tandem-77	**28.2**	**27.0**	**17.7**	**16.7**

Table 6.4. Character error rates obtained on the genre specific systems using mfcc-f0-42, baseline mfcc-f0-tandem-77, and hierarchical mfcc-f0-tandem-77 features and for genre adaptive system using hierarchical mfcc-f0-tandem-77 features.

6.5 Summary and Conclusions

In this chapter, we compared the Tandem features extracted by conventional single MLP based approach and the MLP based hierarchical approach on the GALE Mandarin ASR task. Studies on genre independent, genre specific and genre adaptive systems showed that:

- The hierarchical approach to estimate Tandem features yields a better ASR system when compared to the conventional single MLP based approach for both standalone Tandem features case as well as when augmented with MFCC features.
- In the case of broadcast conversation genre, training the first MLP classifier of the hierarchical approach with data from both the genres, and the subsequent components of the acoustic model with genre specific data yields a better ASR system.

Chapter 7

Summary and Conclusions

This thesis presented two aspects of multilayer perceptron based (MLP) based acoustic modeling: (a) an MLP based hierarchical acoustic modeling technique and (b) a mathematical framework to analyze the trained parameters of the MLP classifier using Volterra series. In this chapter, we summarize the research carried out and discuss some of the promising future directions.

This thesis was based on the premise that there exists useful contextual information in the sequence of phonetic class-conditional probabilities or posterior features estimated by an MLP classifier. This contextual information manifests in the trajectories of posterior features within a phoneme (sub-phonemic level) and in their transition to and from neighboring phonemes (sub-lexical level). Posterior features carry lesser nonlinguistic information such as speaker and environmental variabilities when compared to acoustic features and they process a sparse distribution. Because of these properties, we hypothesized that contextual information spanning longer temporal contexts can be effectively learned in the posterior feature space.

To this end, we investigated an MLP based hierarchical system to estimate the phonetic class-conditional probabilities. The architecture consisted of two classifiers connected in tandem. The first MLP was trained using standard perceptual linear predictive cepstral features with a temporal context of around 90 ms. The second MLP classifier was trained on the posterior features estimated by the first classier, but with a relatively longer temporal context of 150-230 ms. The posterior probabilities of phonemes estimated by the hierarchical approach are used in the same way as the conventional single MLP based approach.

The effectiveness of the MLP based hierarchical acoustic modeling approach was demonstrated on different ASR applications: (a) continuous phoneme recognition in read speech and conversation telephone speech, (b) small vocabulary isolated word recognition, (c) task adaptation, and (d) large vocabulary continuous speech recognition in Mandarin. On all these applications, the MLP based hierarchical approach for estimating the phonetic class conditional probabilities yielded a better ASR system when compared to the conventional single MLP based approach.

We proposed a generic mathematical framework to represent a cascade of a linear time invariant system and three-layered MLP using Volterra series. In this way, a part of the feature extraction (linear time-invariant system following the auditory analysis) can be included in the analysis of the trained MLP and functionality of the combined system can be interpreted in terms of spectro-temporal patterns. We showed the calculation of the Volterra kernels and demonstrated its applicability in the analysis of MLP classifiers trained on acoustic features such as mel frequency cepstral coefficients.

We analyzed the second MLP classifier in the hierarchical system using Volterra series. Analysis of the linear Volterra kernels showed that it has effectively learned the phonetic temporal patterns at the output of the first classifier as well as the phonotactics of the language as observed in the training data. Furthermore, we showed that the second MLP in the hierarchical system can be simpler in terms of model complexity and that it can be trained using lesser amount of data. This can be attributed to the useful properties of posterior features such as the sparse distribution and lesser nonlinguistic variability.

Some of the promising future directions from this thesis are as follows:

- **Hierarchical approach applied to multilingual ASR:** The first MLP classifier could be trained using an assortment of data sets from various languages which bear some sort of similarity *e.g.,* a group of European languages. The output classes could correspond to the union of the phonemes in these languages. At the second stage of the hierarchical system, an MLP could be trained using the posterior features corresponding to the "global" language with output classes representing the phonemes in the target language. This could possibly help in building ASR systems in a new language where training data is limited or even help in reducing the error rates in a language with sufficient resources.

- **Integrating articulatory features:** The hierarchical approach can also be applied in integrating articulatory features. The first MLP classifier could be trained using acoustic features with the articulatory attributes of phoneme such as place of articulation, manner of articulation etc as the output classes. At the second stage, an MLP is trained on the articulatory features with a longer temporal context to estimate the phonetic class-conditional probabilities. In this case, the second MLP learns the articulatory-temporal patterns for each of the phonemes.

  In the context of multilingual ASR discussed above, using articulatory classes at the output of the first MLP is appealing as articulatory features are less language dependent than the phoneme posterior features.

- **Choice of the classifiers:** In the hierarchical approach discussed in this thesis, the posterior features were estimated by an MLP (first classier) and the contextual information in the posterior features was also captured by an MLP. In Chapter 4, we showed that the posterior features can also be estimated using other classifiers such as a Gaussian mixture model. Some other works in the literature such as (Morris and Fosler-Lussier, 2008) have shown that the second classifier could also be a conditional random field. Although we have not performed extensive experiments, it can be said that the choice of the classifiers here is secondary. The important aspect is transforming the acoustic features into some linguistically meaningful features where the nonlinguistic variability is minimal, and then exploiting the contextual information in the posterior feature space. To this end, better architectures and more powerful modeling techniques could be investigated.

- **Hierarchical system for read and conversational speech:** In both recognition of phonemes (on TIMIT and CTS) as well as recognition of words in Mandarin (BN and BC), we observed that the improvement in performance obtained by the hierarchical approach is higher in the case of conversational speech when compared to read speech. Further investigation needs to be carried in this direction to ascertain if this is indeed the case or just an artifact observed in these datasets.

- **Analysis of MLP classifiers trained on acoustic features:** In this thesis, we did not perform a detailed analysis of the MLP classifiers trained on acoustic features such as mel

frequency cepstral features. The proposed framework can be easily applied in the analysis of an MLP classifier trained to classify vowels. We believe that a careful analysis of the linear and quadratic Volterra kernels along with the phonetic confusion matrix of the truncated Volterra series can reveal significant insights into the functionality of the system.

Appendix A

Appendices

A.1 Derivation of Volterra Kernels

In this section, we show the detailed steps involved in the derivation of the Volterra kernels in Section 3.3.1. Equation (3.18) can be expanded as

$$y^j(t) = b_o^j + \sum_{i=1}^{M} c_i^j \; a_{0,i} \; + \; \sum_{i=1}^{M} c_i^j \; a_{1,i} \; s_i(t) \; + \; \sum_{i=1}^{M} c_i^j \; a_{2,i} \; s_i(t)^2 + \ldots \tag{A.1}$$

By substituting (3.12) in (3.16), and further substituting the resulting equation in (A.1), we obtain

$$y^j(t) = b_o^j + \sum_{i=1}^{M} c_i^j \; a_{0,i} \; + \; \sum_{i=1}^{M} c_i^j \; a_{1,i} \sum_{k_1=1}^{K} \sum_{l_1=1}^{L} w_{k_1 l_1}^i \int_{\tau_1} h_{l_1}(\tau_1) \, x_{k_1}(t-\tau_1) \, d\tau_1 \; +$$
$$\sum_{i=1}^{M} c_i^j a_{2,i} \sum_{k_1=1}^{K} \sum_{l_1=1}^{L} \sum_{k_2=1}^{K} \sum_{l_2=1}^{L} w_{k_1 l_1}^i w_{k_2 l_2}^i \int_{\tau_1} \int_{\tau_2} h_{l_1}(\tau_1) \, h_{l_2}(\tau_2) \, x_{k_1}(t-\tau_1) \, x_{k_2}(t-\tau_2) \, d\tau_1 d\tau_2 + \ldots \tag{A.2}$$

By exchanging summation and integration, and rearranging terms in the above equation, we obtain

$$y^j(t) = \left[b_o^j + \sum_{i=1}^{M} c_i^j \; a_{0,i} \right] + \sum_{k_1=1}^{K} \int_{\tau_1} \left[\sum_{i=1}^{M} c_i^j \; a_{1,i} \sum_{l_1=1}^{L} w_{k_1 l_1}^i h_{l_1}(\tau_1) \right] x_{k_1}(t-\tau_1) \, d\tau_1 +$$
$$\sum_{k_1=1}^{K} \sum_{k_2=1}^{K} \int_{\tau_1} \int_{\tau_2} \left[\sum_{i=1}^{M} c_i^j a_{2,i} \sum_{l_1=1}^{L} \sum_{l_2=1}^{L} w_{k_1 l_1}^i w_{k_2 l_2}^i h_{l_1}(\tau_1) \, h_{l_2}(\tau_2) \right] x_{k_1}(t-\tau_1) \, x_{k_2}(t-\tau_2) \, d\tau_1 d\tau_2 + \ldots \tag{A.3}$$

The Volterra synthesis equation is given by (3.14) as

$$y^j(t) = g_0^j + \sum_{k_1=1}^{K} \int_{\tau_1} g_{k_1}^j(\tau_1)\, x_{k_1}(t-\tau_1)\, d\tau_1 + \sum_{k_1=1}^{K}\sum_{k_2=1}^{K} \int_{\tau_1}\int_{\tau_2} g_{k_1 k_2}^j(\tau_1,\tau_2)\, x_{k_1}(t-\tau_1)\, x_{k_2}(t-\tau_2)\, d\tau_1 d\tau_2 + \ldots, \qquad j = 1,\ldots N \tag{A.4}$$

By comparing (A.2) and (A.4), the first three Volterra kernels are identified as

$$g_0^j = b_o^j + \sum_{i=1}^{M} c_i^j\, a_{0,i} \tag{A.5}$$

$$g_{k_1}^j(\tau_1) = \sum_{i=1}^{M} c_i^j\, a_{1,i} \sum_{l_1=1}^{L} w_{k_1 l_1}^i h_{l_1}(\tau_1) \tag{A.6}$$

$$g_{k_1 k_2}^j(\tau_1,\tau_2) = \sum_{i=1}^{M} c_i^j a_{2,i} \sum_{l_1=1}^{L}\sum_{l_2=1}^{L} w_{k_1 l_1}^i w_{k_2 l_2}^i h_{l_1}(\tau_1)\, h_{l_2}(\tau_2) \tag{A.7}$$

A.2 Mean Square Error Fit

In Section 3.3.4, we discussed that the sigmoidal function can be approximated as a polynomial to the desired level of accuracy in an interval in its operating region. In this section, we show the estimation of the coefficients of the polynomial. Suppose that the sigmoidal function $\phi(s+b)$ is approximated using a polynomial function of order P as

$$\phi(s+b) \approx \sum_{n=0}^{P} a_n s^n \tag{A.8}$$

The mean square error between the sigmoidal function and its polynomial approximation is given by

$$mse(a_0, a_1, \ldots a_N) = E_S\left[\phi(s+b) - \sum_{n=0}^{P} a_n s^n\right]^2$$

Here, the activation values to the sigmoidal activation function $\phi(.+b)$ has a normal distribution with a mean zero and variance σ^2, which can be obtained from (3.32). To minimize the mean squared error, the above equation is differentiated with respect to $a_i \quad i = 1, 2, \ldots P$ and equated to

zero as

$$\sum_{n=0}^{P} a_n E_S\left[s^{n+i}\right] = E_S\left[s^i \phi(s+b)\right] \quad i = 0, 1, \ldots P \tag{A.9}$$

The expectation on the left hand side of the above equation correspond to the central moments of the normal density function. The expectation on the right hand side does not have a closed form solution, and is evaluated using the trapezoidal method of numerical integration. The coefficients of the polynomial $a_1, a_2, \ldots a_P$ can be obtained by solving the system of linear equations (A.9) using, for example, the simple matrix inversion method.

A.3 Normalization of Posterior Features

The expression for the posterior features is given by (4.1). In the following derivation, we drop the subscript for time t and simplify the notations by denoting the event $q_t = k$ by simply q_k. The model for the first MLP is denoted by Θ. Subsequently, (4.1) reduces to $x_k = P(q_k \mid \mathbf{f}, \Theta)$, where q_k denotes the phoneme, $\mathbf{f}$ denotes the input feature vector. The mean of the component k in the posterior feature vector is given by

$$\begin{aligned} m_k &= E_{\mathbf{f}}\left[x_k\right] \\ &= E_{\mathbf{f}}\left[P(q_k \mid \mathbf{f}, \Theta)\right] \\ &= \int p(\mathbf{f}) P(q_k \mid \mathbf{f}, \Theta)\, d\mathbf{f} \\ &= \int p(\mathbf{f})\, \frac{p(\mathbf{f} \mid q_k, \Theta)\; P(q_k \mid \Theta)}{p(\mathbf{f} \mid \Theta)}\, d\mathbf{f} \\ &= P(q_k \mid \Theta) \end{aligned} \tag{A.10}$$

Hence, the sample mean of the posterior features is an estimate of the prior probability of the phonemes q_k. In the above simplification, the property $p(\mathbf{f} \mid \Theta) = p(\mathbf{f})$ is exploited. The mean and

variance of the posterior features are related as

$$\begin{aligned}\sigma_k^2 + m_k^2 &= E_{\mathbf{f}}\left[(x_k)^2\right] \\ &= \int p(\mathbf{f})\ \frac{p(\mathbf{f} \mid q_k, \Theta) P(q_k \mid \Theta)}{p(\mathbf{f} \mid \Theta)}\ x_k\ d\mathbf{f} \\ &= P(q_k \mid \Theta) \int p(\mathbf{f} \mid q_k, \Theta)\ x_k\ d\mathbf{f} \\ &= P(q_k \mid \Theta)\ E_{\mathbf{f}|q_k}\left[x_k\right] \end{aligned} \tag{A.11}$$

The conditional expectation in the above expression can be estimated as the average posterior probability of a phoneme obtained using data belonging to that particular phoneme only. If $\hat{x}_k$ denotes the scaled likelihood of the phoneme q_k, and given by

$$\hat{x}_k = \frac{x_k}{m_k} = \frac{P(q_k \mid \mathbf{f}, \Theta)}{P(q_k \mid \Theta)},$$

(A.11) can be expressed using (A.10) as

$$\frac{\sigma_k^2}{m_k^2} + 1 = E_{\mathbf{f}|q_k}\left[\hat{x}_k\right] \tag{A.12}$$

The posterior feature vector component, normalized to zero mean and unit variance $\hat{\hat{x}}_k$ can be be simplified using (A.12) as

$$\begin{aligned}\hat{\hat{x}}_k &= \frac{x_k - m_k}{\sigma_k} \\ &= \frac{\hat{x}_k - 1}{\left[E_{\mathbf{f}|q_k}\left[\hat{x}_k\right] - 1\right]^{\frac{1}{2}}} \end{aligned} \tag{A.13}$$

From (A.13), it is clear that mean and variance normalization on the posterior features is equivalent to taking scaled likelihoods as features. In other words, by taking scaled likelihoods as features and normalizing them to zero mean and unit variance would yield the same features as in (A.13). The only difference is that in the latter, the prior probabilities are estimated by normalizing the relative frequency of the phonetic labels in the training data. In the above formulation, the priors are estimated using the MLP model. In effect, by normalizing the posterior feature to zero mean and unit variance, the effect of priors in them are removed.

Bibliography

Abdel-Haleem, Y. (2006). *Conditional Random Fields for Continuous Speech Recognition*. Ph.D. thesis, University of Sheffield.

Abrash, V., Franco, H., Sankar, A., and Cohen, M. (1995). Connectionist Speaker Normalization and Adaptation. *Proc. of Eurospeech*, pages 2183–2186.

Allen, E. S. (1941). The Scientific Work of Vito Volterra. *The American Mathematical Monthly*, **48**(8), 516–519.

Allen, J. (1994). How do Humans Process and Recognize Speech? *IEEE Trans. Speech. Audio. Process.*, **2**, 567–577.

Aradilla, G. (2008). *Acoustic Models for Posterior Features in Speech Recognition*. Ph.D. thesis, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland.

Atal, B. S. (1974). Effectiveness of Linear Prediction Characteristics of the Speech Wave for Automatic Speaker Identification and Verification. *Journal of Acoustical Society of America*, **55**(6), 1304–1312.

Bahl, L., Brown, P., de Souza, P., and Mercer, R. (1986). Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **11**(2), 49–52.

Bahl, L. R., Jelinek, F., and Mercer, R. L. (1983). A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, **5**(2), 179–190.

Baker, J. K. (1975). The DRAGON System - An Overview. *IEEE Trans. Acoust. Speech. Signal. Process.*, **23**(1), 24–29.

Baum, L. E. *et al.* (1970). A Maximization Technique Occurring in Statistical Analysis of Probabilistic Functions of Markov Chains. *Annals of Mathematical Statistics*, **41**, 164–171.

Benitez, J., Castro, J., and Requena, I. (1997). Are Artificial Neural Networks Black Boxes? *IEEE Trans. Neural Networks*, **8**(5), 1156–1164.

Bilmes, J. (2006). What HMMs Can Do. *IEICE Transactions in Information and Systems*, **89**(3), 869–891.

Bishop, C. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press.

Bocchieri, E., Riley, M., and Saraclar, M. (2004). Methods for Task Adaptation of Acoustic Models with Limited Transcribed In-Domain Data. In *Proc. of International Conference on Spoken Language Processing*.

Bourlard, H. and Dupont, S. (1996). A New ASR Approach based on Independent Processing and Recombination of Partial Frequency Bands. *Proc. of ICSLP*, pages 422–425.

Bourlard, H. and Morgan, N. (1994). *Connectionist Speech Recognition – A Hybrid Approach*. Kluwer Academic Publishers.

Bourlard, H., Morgan, N., Wooters, C., and Renals, S. (1992). CDNN: A Context Dependent Neural Network for Continuous Speech Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **2**, 349–352.

Boyd, S., Chua, L. O., and Desoer, C. A. (1984). Analytical Foundations of Volterra Series. *IMA Journal of Mathematical Control and Information*, **1**, 243–282.

Bridle, J. S. (1990). Training Stochastic Model Recognition Algorithms as Networks can lead to Maximum Mutual Information Estimation of Parameters. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 211–217. Morgan Kaufmann.

Brown, P. F. (1987). *The Acoustic-Modeling Problem in Automatic Speech Recognition*. Ph.D. thesis, Carnegie Mellon University, USA.

Chen, B., Chang, S., and Sivadas, S. (2001). Learning Discriminative Temporal Patterns in Speech: Development of Novel TRAPS-like Classifiers. *Proc. of ICSLP*, pages 429–432.

Collobert, R. and Bengio, S. (2004). Links Between Perceptrons, MLPs and SVMs. In *International Conference on Machine Learning, ICML*.

Cotter, N. (1990). The Stone-Weierstrass Theorem and its Application to Neural Networks. *IEEE Trans. on Neural Networks*, **1**(4), 290–295.

Cybenko, G. (1989). Approximation by Superpositions of a Sigmoidal Function. *Mathematics of Control, Signals, and Systems*, **2**(4), 303–314.

Davis, S. and Mermelstein, P. (1980). Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentence. *IEEE Trans. Acoust. Speech. Signal. Process.*, **28**(4), 357–366.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum-Likelihood from Incomplete Data via the EM Algorithm. *Journal of Royal Statistical Society ser. B*, **39**, 1–38.

Dupont, S., Bourlard, H., Deroo, O., Fontaine, V., and Boite, J.-M. (1997). Hybrid HMM/ANN Systems for Training Independent Tasks: Experiments on 'PhoneBook' and Related Improvements. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 1767–1770.

Ellis, D., Singh, R., and Sivadas (2001). Tandem Acoustic Modeling in Large-vocabulary Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **1**, 517–520.

Evermann, G. *et al.* (2004). Development of the 2003 CU-HTK Conversational Telephone Speech Transcription System. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 249–252.

Fant, G. (1960). *Acoustic Theory of Speech Production*. Mouton, Haag.

Fisher, W., Doddingtion, G., and Goudie-Marshall, K. (1986). The DARPA Speech Recognition Research Database: Specifications and Status. *Proc. of DARPA Workshop on Speech Recognition*, pages 93–99.

Fitt, S. (2000). Documentation and User Guide to UNISYN Lexicon and Post-lexical Rules. Technical report, Center for Speech Technology Research, University of Edinburgh.

Fletcher, H. (1995). *Speech and Hearing in Communication*. Acoustical Society of America, ASA edition edition.

Fontaine, V., Ris, C., Leich, H., Vantieghem, J., Accaino, S., and Compernolle, D. V. (1996). Comparison Between Two Hybrid HMM/MLP Approaches in Speech Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **6**, 3362–3365.

Fosler-Lussier, E. and Morris, J. (2008). CRANDEM Systems: Conditional Random Field Acoustic Models for Hidden Markov Models. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 4049–4052.

Fousek, P., Lamel, L., and Gauvain, J.-L. (2008). Transcribing Broadcast Data using MLP Features. *Proc. of Interspeech*, pages 1433–1436.

Franco, H., Cohen, M., Morgan, N., Rumelhard, D., and Abrash, V. (1994). Context-dependent Connectionist Probability Estimation in Hybrid Hidden Markov Model-Neural Net Speech Recognition. *Computer Speech and Language*, **8**, 211–222.

Franz, M. O. and Scholkopf, B. (2003). Implicit Wiener Series. part i: Cross-Correlation vs. Regression in Reproducing Kernel Hilbert Spaces. Technical Report TR-114, Max Planck Institute for Biological Cybernetics.

Fritsch, J., Finke, M., and Waibel, A. (1997). Context Dependent Hybrid HME/HMM Speech Recognition Using Polyphone Clustering Decision Trees. In *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 1759–1762.

Furui, S. (1986a). On the Role of Spectral Transition for Speech Perception. *Journal of Acoustical Society of America*, **80**(4), 1016–1025.

Furui, S. (1986b). Speaker-independent Isolated Word Recognition using Dynamic Features of Speech Spectrum. *IEEE Trans. Acoust. Speech. Signal. Process.*, **34**, 52–59.

Furui, S. (2003). Recent Advances in Spontaneous Speech Recognition and Understanding. In *Proc. of ISCA and IEEE Workshop on Spontaneous Speech Processing and Recognition*, pages 1–6.

Gales, M. and Young, S. (2008). The Application of Hidden Markov Models in Speech Recognition. *Foundations and Trends in Signal Processing*, **1**(3), 195–304.

Gales, M. E., Dong, Y., Povey, D., and Woodland, P. (2003). Porting: Switchboard to the Voicemail Task. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 536–539.

Godfrey, J. J., Holliman, E. C., and Mcdaniel, J. (1992). SWITCHBOARD: Telephone Speech Corpus for Research and Development. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **1**, 517–520.

Gold, B. and Morgan, N. (1999). *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*. John Wiley & Sons, Inc.

Grezl, F., Karafiat, M., Kontar, S., and Cernosky, J. (2007). Probabilistic and Bottleneck Features for LVCSR of Meetings. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 757–760.

Hain, T. *et al.* (2005). The Development of AMI System for Transcription of Speech in Meetings. In S. Renals and S. Bengio, editors, *Machine learning for Multimodal Interaction: 2nd International Workshop, Revised Selected Papers*, volume 3869, pages 344–356. Springer-Verlag.

Hakim, N., Kaufman, J., and Meadows, H. (1991). Volterra Characterization of Neural Networks. *IEEE Conf. Signals, Systems and Computers*, **2**, 1128–1132.

Hasler, M. (2010). *Personal Communication*.

Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation*. Prentice Hall.

Hecht-Nielsen, R. (1987). Kolmogorov's Mapping Neural Network Existence. *Proc. IEEE First Annual International Conf. on Neural Networks Theorem*, **3**, 11–14.

Hermansky, H. (1990). Perceptual Linear Predictive (PLP) Analysis of Speech. *Journal of Acoustical Society of America*, **87**(4), 1738–1752.

Hermansky, H. and Fousek, P. (2005). Multi-Resolution RASTA Filtering for Tandem based ASR. *Proc. of Interspeech*, pages 361–364.

Hermansky, H. and Sharma, S. (1999). Temporal Patterns (TRAPs) in ASR of Noisy Speech. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **1**, 289–292.

Hermansky, H., Ellis, D., and Sharma, S. (2000). Tandem Connectionist Feature Extraction for Conventional HMM Systems. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 1635–1638.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, **2**(5), 359–366.

Hwang, M., Lei, X., Wang, W., and Shinozaki, T. (2006). Investigation on Mandarin Broadcast News Speech recognition. *Proc. of International Conference on Spoken Language Processing*.

Hwang, M.-Y., Huang, X., and Alleva, F. (1993). Predicting Unknown Triphones with Senones. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **2**, 311–314.

Hwang, M.-Y., Wang, W., Lie, X., Zheng, J., Cetin, O., and G., P. (2007). Advances in Mandarin Broadcast Speech Recognition. *Proc. of Interspeech*, pages 2613–2617.

Hwang, M.-Y., Peng, G., Ostendorf, M., Wang, W., Faria, A., and Heidel, A. (2009). Building a Highly Accurate Mandarin Speech Recognizer with Language-independent Technologies and Language-Dependent Modules. *To appear in IEEE Trans. on Audio, Speech, and Language Process.*

Ikbal, S. (2004). *Nonlinear Feature Transformations for Noise Robust Speech Recognition*. Ph.D. thesis, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland.

Jelinek, F. (1976). Continuous Speech Recognition by Statistical Methods. *Proc. of the IEEE*, **64**(4), 532–556.

Jelinek, F. (2001). *Statistical Methods for Speech Recognition*. MIT Press.

Johnson, D. *et al.* (2000). The ICSI Quicknet Software Package.

Kaiser, Z., Horvat, B., and Kacic, Z. (2002). Overall Risk Criterion Estimation of Hidden Markov Model Parameters. *Speech Communication*, **38**(3-4), 383–398.

Katz, S. M. (1987). Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Trans. Acoust. Speech. Signal. Process.*, **35**(3), 400–401.

Ketabdar, H. and Bourlard, H. (2008). Hierarchical Integration of Phonetic and Lexical Knowledge in Phone Posterior Estimation. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 4065–4068.

Ketabdar, H., Vepa, J., Bengio, S., and Bourlard, H. (2006). Using More Informative Posterior Probabilities for Speech Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 29–32.

Khan, S. U., Sharma, G., and Rao, P. (2000). Speech Recognition using Neural Networks. *IEEE Conference on Industrial Technology*, **2**, 432–437.

Kittler, J., Hatef, M., Duin, R., and Matas, J. (1998). On Combining Classifiers. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **20**, 226–239.

Klatt, D. H. (1985). The Problem of Variability in Speech Recognition and in Models of Speech Perception. In J. Perkell and D. Klatt, editors, *Variability and Invariance in Speech Processes*. Hillsdale, NJ: Lawrence Erlbaum Assoc.

Klein, D., Depireux, D., Simon, J., and Shamma, S. (2000). Robust Spectrotemporal Reverse Correlation for the Auditory System: Optimizing Stimulus Design. *Journal of Computational Neuroscience*, **9**(1), 85–111.

Korenberg, M. and Hunter, I. (1996). The Identification of Nonlinear Biological Systems: Volterra Kernel Approaches. *Annals of Biomedical Engg.*, **24**(4), 250–268.

LeCun, Y., Bottou, L., Orr, G., and Muller, K.-R. (1998). Efficient BackProp. In G. Orr and K.-R. Muller, editors, *Neural Networks: Tricks of the Trade*, number 1524 in Lecture Notes in Computer Science, pages 9–50. Springer-Verlag.

Lee, K.-F. and Hon, H.-W. (1989). Speaker-Independent Phone Recognition using Hidden Markov Models. *IEEE Trans. Acoust. Speech. Signal. Process.*, **37**(11), 1641–1648.

Lee, L. and Rose, R. (1996). Speaker Normalization using Efficient Frequency Warping Procedures. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **1**, 353–356.

Lee, Y. and Schetzen, M. (1965). Measurement of Wiener Kernels of a Non-linear System by Cross-correlation. *International Journal of Control*, **2**, 237–254.

Lei, X., Siu, M., Ostendorf, M., and Lee, T. (2006). Improved Tone Modeling for Mandarin Broadcast News Speech Recognition. *Proc. of Interspeech*, pages 1237–1240.

Lei, X., Wu, W., Wang, W., Mandal, A., and Stolcke, A. (2009). Development of the 2008 SRI Mandarin Speech-to-text System for Broadcast News and Conversation. In *Proc. of Interspeech*, pages 2099–2103.

Li, X. and Bilmes, J. (2006). Regularized Adaptation of Discriminative Classifiers. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*.

Li, X., Bilmes, J., and Malkin, J. (2005). Maximum Margin Learning and Adaptation of MLP Classifiers. *Proc. of Interspeech*, pages 1789–1792.

LiMin, F. (1994). Rule Generation from Neural Networks. *IEEE Trans. Man. Cybernetics*, **24**(8), 1114–1124.

Makhoul, J. (1975). Linear Prediction: A Tutorial Review. *Proceedings of the IEEE*, **63**(4), 561–580.

Malkin, J., Subramanya, A., and Bilmes, J. (2009). On the Semi-Supervised Training of Multi-Layered Perceptrons. *Proc. of Interspeech*.

Marin, M., Feldman, S., Ostendorf, M., and Gupta, M. (2009). Filtering Web Text to Match Target Genres. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*.

Markel, J. and Gray, A. H. (1976). *Linear Prediction of Speech*. Springer-Verlag, New York.

Marmarelis, P. and Naka, K.-I. (1974). Identification of Multi-Input Biological Systems. *IEEE Trans. Biomedical Engg.*, **21**(2).

Marmarelis, V. (2004). *Nonlinear Dynamic Modeling of Physiological Systems*. Wiley.

Moore, D. *et al.* (2006). Juicer: A Weighted Finite State Transducer Speech Decoder. In S. Renals, S. Bengio, and J. Fiscus, editors, *Machine learning for Multimodal Interaction: 3rd International Workshop, Revised Selected Papers*, volume 4299, pages 285–296. Springer-Verlag.

Morgan, N. *et al.* (2005). Pushing the Envelope - Aside. *IEEE Signal Process. Magazine*, **22**(5), 81–88.

Morris, J. and Fosler-Lussier (2008). Conditional Random Fields for Integrating Local Discriminative Classifiers. *IEEE Trans. Audio. Speech. Language. Process.*, **16**(3), 617–628.

Nadas, A. (1983). A Decision Theoretic Formulation of a Training Problem in Speech Recognition and a Comparison of Training by Unconditional versus Conditional Maximum Likelihood. *IEEE Trans. Acoust. Speech. Signal. Process.*, **31**(4), 814–817.

Nakamura, M., Iwano, K., and Furui, S. (2007). Differences Between Acoustic Characteristics of Spontaneous and Read Speech and their Effects on Speech Recognition Performance. *Computer Speech and Language*, **22**, 171–184.

Neto, J., Almeida, L., Hochberg, M., Martins, C., Nunes, L., Renals, S., and Robinson, T. (1995). Speaker Adaptation for Hybrid HMM-ANN Continuous Speech Recognition System. *Proc. of Eurospeech*, pages 2171–2174.

Ogunfunmi, T. (2007). *Adaptive Nonlinear System Identification: The Volterra and Wiener Model Approaches*. Springer.

Park, J., Diehl, F., Gales, M., Tomalin, M., and Woodland, P. (2009). Training and Adapting MLP Features for Arabic Speech Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 4461–4464.

Pinto, J. and Hermansky, H. (2008). Combining Evidence from a Generative and a Discriminative Model in Phoneme Recognition. *Proc. of Interspeech*, pages 2414–2417.

Pinto, J., Yegnanarayana, B., Hermansky, H., and Magimai.-Doss, M. (2008). Exploiting Contextual Information for Improved Phoneme Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 4449–4452.

Pinto, J., Sivaram, G., Magimai.-Doss, M., Hermansky, H., and Bourlard, H. (2009a). Analysis of MLP based Hierarchical Phoneme Posterior Probability Estimator. *Submitted to IEEE Trans. Audio. Speech. Language. Process.*

Pinto, J., Magimai.-Doss, M., H., and Bourlard, H. (2009b). MLP Based Hierarchical System for Task Adaptation in ASR. *Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.

Pitrelli, J. *et al.* (1995). PhoneBook: A Phonetically-rich Isolated-word Telephone Speech Database. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 101–104.

Povey, D. *et al.* (2005). FMPE: Discriminatively Trained Features for Speech Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **1**, 961–964.

Povey, D. and Woodland, P. (2002). Minimum Phone Error and I-Smoothing for Improved Discriminative Training. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **1**, 105–108.

Rabiner, L. R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proc. of the IEEE*, **77**(2), 257–286.

Richard, M. and Lippmann, R. (1991). Neural Network Classifiers Estimate Bayesian *a posteriori* Probabilities. *Neural Computation*, **3**, 461–483.

Robinson, A. (1994). An Application of Recurrent Nets to Phone Probability Estimation. *IEEE Trans. Neural Networks*, **5**(2), 298–305.

Rudin, W. (1976). *Principles of Mathematical Analysis*. McGraw-Hill, New York.

Schwarz, P., Matejka, P., and Cernocky, J. (2006). Hierarchical Structures of Neural Networks for Phoneme Recognition. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 325–328.

Setiono, R. and Liu, H. (1996). Symbolic Representation of Neural Networks. *IEEE Computer*, **29**(3), 71–77.

Setiono, R., Leow, W.-K., and Zurada, J. M. (2002). Extraction of Rules From Artificial Neural Networks for Nonlinear Regression. *IEEE Trans. Neural Networks*, **13**(3), 564–577.

Sim, K. and Gales, M. (2007). Discriminative Semi-parametric Trajectory Models for Speech Recognition. *Computer Speech and Language*, **21**(4), 669–687.

Sivadas, S. and Hermansky, H. (2002). Hierarchical Tandem Feature Extraction. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, **1**, 809–812.

Stephenson, T., Magimai.-Doss, M., and Bourlard, H. (2004). Speech Recognition with Auxiliary Information. *IEEE Trans. on Speech and Audio Processing*, **4**, 189–203.

Stolcke, A. (2002). SRILM:An Extensible Language Modeling Toolkit. *Proc. of Intl. Conf. on Spoken Language Processing*, **2**, 901–904.

Stolcke, A. *et al.* (2005). Further Progress in Meeting Recognition: The ICSI-SRI Spring 2005 Speech-to-text Evaluation System. *Second Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms*, pages 463–475.

Stolcke, A. *et al.* (2006). Recent Innovations in Speech-to-Text Transcription at SRI-ICSI-UW. *IEEE Trans. Audio. Speech. Language. Process.*, **14**(5), 1729–1744.

Tibrewala, S. and Hermansky, H. (1997). Sub-Band Based Recognition of Noisy Speech. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 1255–1258.

Valente, F. and Hermansky, H. (2008a). Hierarchical and Parallel Processing of Modulation Spectrum for ASR Applications. *Proc. of IEEE Conf. Acoust. Speech. Signal Process. (ICASSP)*, pages 4165–4168.

Valente, F. and Hermansky, H. (2008b). On the Combination of Auditory and Modulation Frequency Channels for ASR Applications. *Proc. of Interspeech*, pages 2242–2245.

Valente, F., Magimai.-Doss, M., Plahl, C., and Ravuri, S. (2009). Hierarchical Processing of the Modulation Spectrum for GALE Mandarin LVCSR System. In *Proc. of Interspeech*, pages 2963–2966.

Viterbi, A. J. (1967). Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Trans. on Information Theory*, **13**(2), 260–269.

Volterra, V. (1930). *Theory of Functionals and of Integro-Differential Equations*. Dover, New York.

Waibel, A., Hanazava, T., Hinton, G., Shikano, K., and Lang, K. (1989). Phoneme Recognition using Time-Delay Neural Networks. *IEEE Trans. Acoust. Speech. Signal. Process.*, **37**(3), 328–339.

Wang, W., Mandal, A., Lei, X., Stolcke, A., and Zheng, J. (2009). Multifactor Adaptation for Mandarin Broadcast News and Conversation Speech Recognition. In *Proc. of Interspeech*, pages 2103–2102.

Wiener, N. (1958). *Nonlinear Problems in Random Theory*. MIT Press.

Woodland, P. (1999). Speaker Adaptation: Techniques and Challenges. *Proc. of IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 85–90.

Woodland, P. C. *et al.* (2003). The CU-HTK English CTS System. *Proc. of the Rich Transcription Workshop*.

Wray, J. and Green, G. (1994). Calculation of the Volterra Kernels of Nonlinear Dynamic Systems using an Artificial Neural Network. *Biological Cybernetics*, **71**(3), 187–195.

Xuedong, H., Acero, A., and Hon, H.-W. (2001). *Spoken Language Processing: A Guide to Theory, Algorithm and System Development*. Prentice Hall PTR.

Young, S., Kershaw, D., Odell, J., Ollason, D., Valtchev, V., and Woodland, P. (2000). *The HTK Book Version 3.0*. Cambridge University Press.

Zhu, Q., Chen, B., Morgan, N., and Stolcke, A. (2004). On Using MLP Features in LVCSR. *Proc. of Interspeech*, pages 921–924.

Zue, V. (1985). The Use of Knowledge in Automatic Speech Recognition. *Proc. of the IEEE*, pages 1602–1615.

Curriculum Vitae

JOEL PINTO

Rue Marconi 19

1920 Martigny

Switzerland

joypinto20@gmail.com

Education

- Doctor of Philosophy (PhD)

  Dept. of Electrical Engineering

  Ecole Polytechnique Fédérele de Lausanne (EPFL), Switzerland

  Nov 2005 - March 2010

- Masters in Engineering (M.E.)

  Dept. of Electrical Engineering

  Indian Institute of Science, Bangalore, India

  Aug 2001 - Jan 2003

- Bachelors in Engineering (B.E.)

  Dept. of Electronics and Communication Engineering

  Manipal Institute of Technology, India

  Aug 1997 - July 2001

Professional Experience

- Research Assistant
  Idiap Research Institute
  Martigny, Switzerland
  Sept. 2005 - present

- Research Consultant
  Hewlett Packard Labs India
  Bangalore, India
  Feb. 2003 - Aug. 2005

Publications

Book Chapters

- C. Stricker, J.-F Wagen, G. Aradilla, H, Bourlard, H. Hermansky, **J. Pinto**, P. Henri, and J. Théraulaz, *Intelligent Multi-modal Interfaces for Mobile Applications in Hostile Environment*, in Human Machine Interaction: Research Results of the MMI Program, Springer-Verlag, 2009.

Journal Papers

- **J. Pinto**, G. Sivaram, M. Magimai.-Doss, H. Hermansky, and H. Bourlard, *Analyzing MLP Based Hierarchical Phoneme Posterior Probability Estimator*, To appear in IEEE Transactions on Audio, Speech, and Language Processing.

- **J. Pinto**, G. Sivaram, M. Magimai.-Doss, H. Hermansky, and H. Bourlard *Volterra Series for Analyzing MLP Based Phoneme Posterior Probability Estimator*, (Manuscript under preparation.)

Conference Proceedings

- **J. Pinto**, M. Magimai.-Doss, and H. Bourlard, *MLP Based Hierarchical System for Task Adaptation in ASR*, The eleventh biannual IEEE workshop on Automatic Speech Recognition and Understanding (ASRU), Merano, Italy, 2009.

- **J. Pinto**, G. Sivaram, H. Hermansky, and M. Magimai.-Doss, *Volterra Series for Analyzing MLP Based Phoneme Posterior Probability Estimator* Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Taipei, Taiwan, 2009.

- **J. Pinto**, I. Szoke, S. Prasanna, and H. Hermansky, *Fast Approximate Spoken Term Detection from Sequence of Phonemes*, Workshop on Searching Conversational Speech at ACM SIGIR, Singapore, 2008.

- **J. Pinto**, B. Yegnanarayana, H. Hermansky and M. Magimai.-Doss, *Exploiting Contextual Information for Improved Phoneme Recognition*, Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Las Vegas, USA, 2008.

- **J. Pinto**, and H. Hermansky, Proceedings of Interspeech, 2008 *Combining Evidence from a Generative and a Discriminative Model in Phoneme Recognition*, Proceedings of Interspeech, Brisbane, Australia, 2008.

- **J. Pinto**, G. Sivaram, and H. Hermansky, *Reverse Correlation for Analyzing MLP Posterior Features in ASR* Proceedings of International Conference Text, Speech, and Dialogue, Czech Republic, 2008.

- **J. Pinto**, A. Lovitt, and H. Hermansky, *Exploiting Phoneme Similarities in Hybrid HMM-ANN Keyword Spotting*, Proceedings of Interspeech, Antwerp, Belgium, 2007.

- **J. Pinto**, and R. Sitaram, *Confidence Measures in Speech Recognition based on Probability Distribution of Likelihoods*, Proceedings of Interspeech, Lisbon, Portugal, 2005.

- S. Singh, S. Manocha, R. Sitaram, K. Bali, and **J. Pinto**, *Building Large Vocabulary Speech Recognition Systems for Indian Languages*, Proceedings of International Conference on Natural Language Processing, Hyderabad, India, 2004.

- **J. Pinto**, R. Muralishankar, and A. Ramakrishnan, *ICA in Speech Recognition using HMMs*, International Conference on Advances in Pattern Recognition, Kolkatta, India, 2003.
- R. Sitaram, K. Anjaneyulu, K. Bali, G. Prasad, and **J. Pinto**, *Local Language Voice Services - Enabling OCMP for Emerging Markets*, Hewlett-Packard Technical Conference, Orlando, USA, 2004.

Technical Reports

- S. Soldo, M. Magimai.-Doss, **J. Pinto**, and H. Bourlard, *On MLP Based Posterior Features for Template Based ASR*, Tech. Rep. No. 37, Idiap Research Institute, 2009.
- **J. Pinto**, H. Bourlard, Z. DeGreve, and H. Hermansky, *Comparing Different Word Lattice Rescoring Approaches Towards Keyword Spotting*, Tech. Rep. No. 32, Idiap Research Institute, 2007.
- A. Lovitt, **J. Pinto** and H. Hermansky, *On Confusions in a Phoneme Recognizer*, Tech. Rep. No. 10, Idiap Research Institute, 2007.
- S. Prasanna, B. Yegnanarayana, **J. Pinto**, and H. Hermansky, *Analysis of Confusion Matrix to Combine Evidence for Phoneme Recognition*, Tech. Rep. No. 27, Idiap Research Institute, 2007.

Research Disclosures

- **J. Pinto**, *Cursor on Handheld Devices with a Camera*, No. 498036, Research Disclosure Journal, Oct 2005.
- S. Badaskar, S. Nemala, S. Singh, **J. Pinto**, and R. Sitaram, *System and Method to Embed Speech (Text to Speech) Information along with Printed Text*, No. 501048, Research Disclosure Journal, Jan 2006.