ACOUSTIC MODELING FOR SPEECH RECOGNITION

A Project Report

submitted by

G SRIRAM

in partial fulfillment of the requirements for the award of

DUAL DEGREE

Department of Electrical Engineering

Indian Institute of Technology Madras

June 2014

THESIS CERTIFICATE

This is to certify that the thesis titled Acoustic Modeling for Speech Recognition, submitted by G Sriram (EE09B079), to the Indian Institute of Technology, Madras for the award of Dual Degree, is a bona fide record of the research work done by him under my supervision. The contents of the thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. S.Umesh
Research Guide
Professor
Department of Electrical Engineering
IIT Madras-600036

Place : Chennai
Date : 13-6-2014

ACKNOWLEGEMENTS

I would like to thank my project guide, Prof. S.Umesh, for giving me this opportunity to work under him. This project has given me invaluable knowledge and experience which will help me in my future assignments. I would also like to thank him for giving me a wonderful lab atmosphere to work in. I would like to thank my lab mates for helping me during the course of the project and making my working experience memorable.

I would like to thank the Department of Electrical Engineering and IIT Madras for providing me education on par with the best in the world. I would like to thank my parents and my friends for their love and encouragement throughout the course of my studies and my entire life.

ABSTRACT

In this thesis, the performances of the Continuous Density Hidden Markov Model (CDHMM), Subspace Gaussian Mixture Model (SGMM) and the recently introduced Transform-based Phone CAT model for speech recognition is investigated.

The Transform-based Phone CAT technique is inspired from the Transform-based Cluster Adaptive Training (CAT) technique used for rapid speaker adaptation of Gaussian Mixture Models (GMMs). Analogous to the CAT, a compact canonical model is adapted through piecewise linear transformations to a set of cluster models representing the phones. The parameters of the distributions in the tied context-dependent phone states are modelled as weighted linear interpolation of the phone cluster models.

Approaches to optimize the Transform-based Phone CAT technique with respect to time are discussed. Transform-based Phone CAT technique's working is analysed to find the sections where improvements can be made to quicken the training process without affecting the performance of the system.

ABBREVIATIONS

ASR	Automatic Speech Recognition
CAT	Cluster Adaptive Training
CDHMM	Continuous Density Hidden Markov Model
CMN	Cepstral Mean Normalization
CMS	Cepstral Mean Subtraction
EM	Expectation Maximization
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
LDA	Linear Discriminative Analysis
MFCC	Mel-frequency Cepstral Coefficients
MLLR	Maximum Likelihood Linear Regression
MLLT	Maximum Likelihood Linear Transform
p.d.f.	Probability Density Function
SAT	Speaker Adaptation Training
SGMM	Subspace Gaussian Mixture Model
WER	Word Error Rate
UBM	Universal Background Model

CONTENTS

List of Tables

List of Figures

Chapter 1

Introduction to ASR

Automatic speech recognition (ASR) can be defined as the independent, computer-driven transcription of spoken language into readable text in real time (Stuckless, 1994). In a nutshell, ASR is technology that allows a computer to identify the words that a person speaks into a microphone or telephone and convert it to written text. Having a machine to understand fluently spoken speech has driven speech research for more than 50 years.

The ultimate goal of ASR research is to allow a computer to recognize in real-time, with 100 % accuracy all words that are intelligibly spoken by any person, independent of vocabulary size, noise, speaker characterisitics or accent. Today, if the system is trained to learn an individual speaker's voice then much larger vocabularies are possible and accuracy can be greater than 90%.

Commercially available ASR systems usually require only a short period of speaker training and may successfully capture continuous speech with a large vocabulary at normal pace with highaccuracy.Most commercial companies claim that recognition software can achieve between 98 to 99% accuracy if operated under optimal conditions. ` Optimal conditions' usually assume that users: have speech characteristics which match the training data, can achieve proper speaker adaptation, and work in a clean noise environment (e.g. quiet space).

It is a difficult problem because of the different kinds of variability in speech due to changes in speaker and environment. Statistical parametric models like HMM are generally used to model the production of speech sounds. The performance of the speech recognition systems entirely depends on the how good the modeling is and how well the parameters of the model can be estimated using the available training data.

In conventional CDHMM systems that are typically used in speech recognition

applications, the p.d.f. of each HMM state is a Gaussian Mixture Model (GMM). A lot of parameters (means, variances and weights) are required to define these GMMs, thus emanding a large amount of training data.

A relatively new acoustic modeling technique, known as SGMM, was introduced in Povey (2009), which takes advantage of the high correlation between the state's distributions to generate the GMM parameters indirectly using only a small number of state- specific parameters. The state GMM parameters are constrained to lie in a low dimensional subspace of the total parameter space. The parameters that are used to define this subspace are shared among all the states and thus can be estimated robustly using limited amount of data and even out-of-domain data. This has been verified through several multilingual experiments (Burget et al. (2010), Mohan et al. (2012)).

The model is based on the same principle as the SGMM to use a compact subspace model that can generate the GMMs using few state-specific parameters. It models the phone models in a way analogous to the speaker models in CAT. It consists of a compact canonical model that is adapted through piece-wise MLLR transforms to the phone models. The tied context- dependent phone state models are expressed as a linear combination of these phone models.

Chapter 4 describes the basic theory on which the transform-based Phone CAT model is based on. Sections 3.1 and 3.2 outline the basic HMM-based speech recognition procedure and the conventional HMM- GMM system. Sections 3.3 describes the SGMM.

Sections 4.2 to 4.5 describe in depth the modeling technique and the estimation procedure of the transform-based Phone CAT models. Chapter 5 gives the conclusions.

Chapter 2

Introduction to speech recognition

In real world, the observations are generally in the form of signals. The signals can be discrete in nature or continuous in nature (like speech signals, music). Most of the real world signals are generated continuously in time. But in all practical applications, we can extract only a finite number of samples of the signal and they need to be quantized to take only a finite number of values.

The statistics of signals (such as speech) vary over time and hence are non-stationary. But they can be assumed to be stationary over a short observation window (25ms) and fall into a category of pseudo-stationary signals. This allows us to model the signals with efficient parametric models.

The models used to characterize signals can be broadly classified into deterministic and statistical models. Deterministic models require some properties of the observed signal to be exploited, like the signal's frequency is within this set or the signal resembles a sine wave etc. Here, modeling is basically finding out parameters like frequency, amplitude of the signal etc.

Signals like speech can be modeled as the outcome of a random process and the parameters of this process can be estimated accurately. For temporal pattern recognition applications like speech, stochastic models known as Hidden Markov Models (HMM) are widely used. The outputs of the states are observed but the states which produced these outputs are not observable. The output is conventionally modeled to be generated from a Gaussian Mixture Model (GMM). This is referred to as the HMM-GMM system.

Section 3.1 gives a brief introduction to the HMM- based speech recognition system. Section 3.2 describes the conventional HMM-GMM system. Section 2.2 describes the Subspace Gaussian Mixture Model (SGMM) based system.

Chapter 3

Statistical Models for Speech recognition

3.1 HMM-based Speech Recognition

3.1.1 Introduction:

The pseudo-stationary property of speech signals allows the speech signal to be divided into 25ms observation windows. The statistical properties of the signal can be assumed to be constant over this window. The data in this window is converted into discrete parameter vectors. This process of conversion of continuous speech signal into a sequence of discrete vectors is known as Feature Extraction. These vectors are also known as feature vectors or observation vectors.

One of the most widely used features is the Mel Frequency Cepstral Coefficients (MFCC).The objective of the speech recognition system is to convert this sequence of observations into a sequence of symbols (or words) that can be understood by a machine. The observation sequence can be modeled as to be generated by a sequence of states as defined by a HMM.

A typical acoustic modelling uses a 3-state left-to-right HMM topology (Fig. 2.1) to model the features generated by a single phonetic unit.A first order Hidden Markov process is assumed meaning that the transition into a particular state depends only on the previous state and that the observation depends only on the current state.

3.1.2 Characterization

An HMM is characterized by the following:

- N , the number of states in the model. The individual states in the model are given by $S = \{ S_1 , S_2 ..., S_N \}$. The state at frame t is denoted as q_t .
  For the model of a basic phonetic unit such as a phoneme, we typically use $N = 3$.

- A, the state transition probability distribution. $A = \{a_{ij}\}$ where

$$a_{ij} = P\left[q_t = S_j \,\middle|\, q_{t-1} = S_i\right] \ , 1 \le i, j \ \le N \qquad (3.1)$$

For speech systems, we use a left-to-right topology, which implies that $a_{ij} = 0$ for $j < i$.

- π, the initial state distribution. $\pi = \{\pi_i\}$ where

$$\pi_i = P[q_0 = S_i \,|] \ , 1 \le i \ \le N \qquad (3.2)$$

The model of a basic phonetic unit such as phoneme has $\pi_i = 0$ for $i \neq 1$.

- The observation probability distribution in state j. In the case of a discrete HMM with output vectors v_1 , v_2 , ..., v_k , the probability of observing v_k in the state j is given by

$$b_j(k) = P[v_k at\ t \,|q_t = S_j] \ , 1 \le i \ \le N, 1 \le k \le M \qquad (3.3)$$

The observation vector, $x(t)$, can assumed to be generated from a continuous distribution. The probability density function (p.d.f.) can be modeled as a mixture of Gaussians or a GMM:

$$b_j\big(x(t)\big) = P[x(t)|q_t = S_j, \mu_{ji}, \Sigma_{ji}] = \sum_{i=1}^{I} w_{ji} N\big(x; \mu_{ji}; \Sigma_{ji}\big) \ , 1 \le j \ \le N \quad (3.4)$$

where I is the number of Gaussians in the GMM; μ_{ji}, Σ_{ji} are the means and the covariance matrix of the Gaussian component i of state j; w_{ji} is the Gaussian weight with the constraint : $\sum_{i=1}^{I} w_{ji} = 1$

For convenience, the compact notation λ=(A,B,π) is used to denote the complete parameter set of the model.

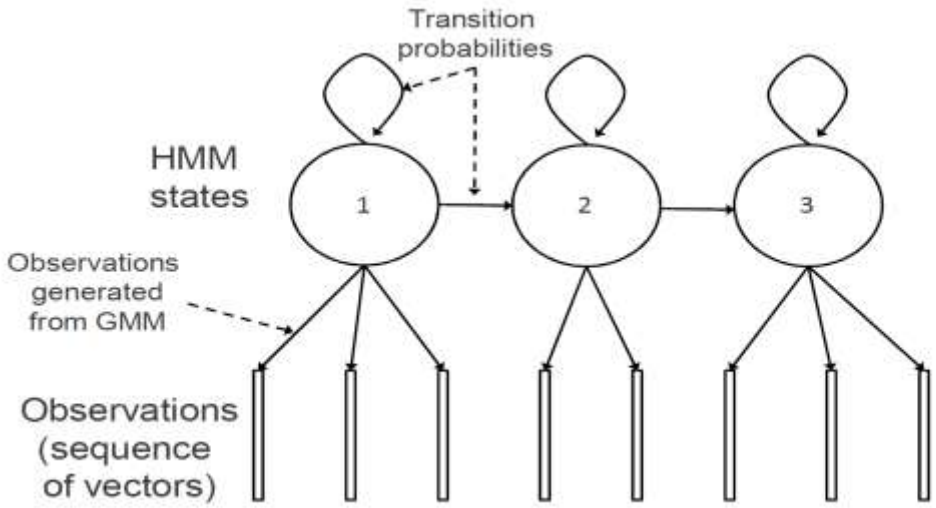


Figure 3.1 Typical left-to-right HMM with 3 emitting states

3.1.3 The three problems associated with HMMs and their solutions

Problem 1

Given the observation sequence $x = x(1)x(2)x(3)....x(T)$ and model $\lambda=(A,B,\pi)$, how do we compute $P(x|\lambda)$, the probability of the observation sequence given the model?

This is calculated by considering every possible state sequence of length T.

Consider one sequence $Q=q_1,q_2,.....q_T$ where q_1 is the initial state.

$$P(x|Q,\lambda) = \prod_{t=1}^{T} P(x(t)|q_t,\lambda) = b_{q_1}\big(x(1)\big) * b_{q_2}\big(x(1)\big) .. b_{q_T}\big(x(T)\big) \quad (3.5)$$

$$P(Q|\lambda) = \pi_{q_1} a_{q_1 q_2} a_{q_2 q_3} \dots a_{q_{T-1} q_T} \quad (3.6)$$

$$P(x|\lambda) = \sum_{over\ all\ Q} P(x|Q,\lambda)P(Q|\lambda) \quad (3.7)$$

This is computationally very inefficient. The **forward-backward algorithm** is an alternate and an efficient method for solving this problem.

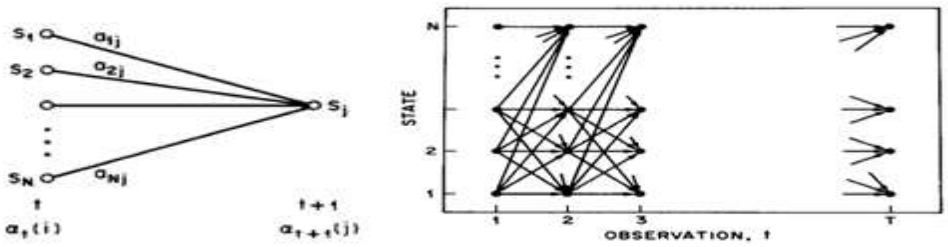
Consider

$$\alpha_t(i) = P[x(1)x(2)\dots x(t), q_t = S_i|\lambda] \ , 1 \le i \ \le N \quad (3.8)$$

which is the probability of observing the partial observation sequence *(x(1) till x(t))* and state S_i at time *t*, given the model.

$\alpha_t(i)$ and thus $P(x|\lambda)$ can be computed by the solving following set of equations:

- Initialization : $\alpha_1(i) = \pi_1 b_i\big(x(1)\big)$ (3.9)
- Induction : $\alpha_{t+1}(i) = [\sum_{i=1}^{N} \alpha_t(i)a_{ij}]\, b_j x(t+1)$ (3.10)
- Termination : $P(x|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$ (3.11)



Figures 3.2 and 3.3 illustrate the steps for the computations of $\alpha_t(i)$ and $P(x|\lambda)$

Problem 2

Given the observation sequence *x = x(1)x(2)x(3)...x(T)* and model parameters λ, how do we find the state sequence $Q = q_1\, q_2 ... q_T$ that best explains the observation sequence?

<u>This can be solved in two ways:</u>

1. Finding the optimal state in which the process is at time t individually. We do this for all "t" to get the sequence.

Consider $\gamma_t(i) = P(q_t = S_i|x, \lambda)$ the probability of being in state S_i at time t, given the observation and the model. This can be expressed as

$$\gamma_t(i) = \frac{\alpha_t^i \beta_t^i}{\sum_{i=1}^{N} \alpha_t^i \beta_t^i} \qquad (3.12)$$

where $\alpha_t(i)$ is the forward variable and β_t^i is the backward variable.

So, the state sequence is $\hat{q}_1 \ldots \hat{q}_T$ where

$$\hat{q}_t = \arg max_i \left[\gamma_t^i\right] \qquad (3.13)$$

2. Viterbi method:

We need to find $\arg max_Q P(Q|x,\lambda)$

This can be computed as

$$\arg max_Q P(Q|x,\lambda)$$
$$= \arg max_Q \frac{P(Q|\lambda) * P(x|Q,\lambda)}{P(x|\lambda)}$$
$$= \arg max_Q P(Q|\lambda) * P(x|Q,\lambda) \qquad (3.14)$$

This equation (3.12) is the Viterbi search algorithm. This is the problem that is solved when attempting to recognize the word sequence that can best recognize the speech frames.

Problem 3

How do we adjust the model parameters (A, B, π) so that P(x|λ) is maximized or in other words, to explain the observations?

This is done using Baum-Welch algorithm, which is a specific case of the Expectation Maximization (EM) algorithm.

The re-estimation formulae can be derived directly by maximizing the Baum's auxiliary function over $\bar{\lambda}$:

$$Q(\lambda,\bar{\lambda}) = \sum_Q P(Q|x,\lambda) \log\left[P\left(Q,x|\bar{\lambda}\right)\right] \qquad (3.15)$$

The set of re-estimation formulae for A, B, π:

$$\overline{\pi_i} = number\ of\ times\ in\ state\ S_i at\ time\ t = 1 = \gamma_1(i) \qquad (3.16)$$

$$\overline{a_{ij}} = \frac{expected\ number\ of\ transitions\ from\ S_i to\ S_j}{expected\ number\ of\ transitions\ from\ S_i} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \qquad (3.17)$$

$$\overline{b_j(k)} = \frac{expected\ number\ of\ times\ in\ S_j\ observing\ v_k}{expected\ number\ of\ times\ in\ S_j} = \frac{\sum_{t=1\ s.t.x(t)=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \qquad (3.18)$$

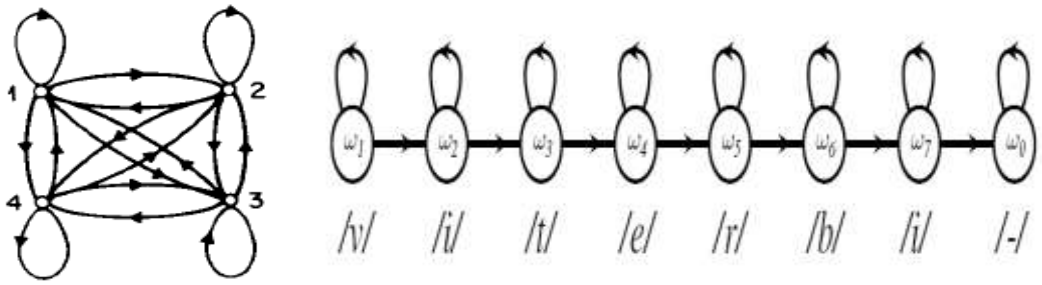
where $\xi_t(i,j) = P\left[q_t = S_i\ , q_{t+1} = S_j|x,\lambda\right]$ is the probability of being in state i at time

t and state j at time t+1, given the model and observation.

These problems and their respective solutions are discussed in detail in "A tutorial on Hidden Markov Models and Selected Applications in Speech recognition", Lawrence R.Rabiner.

3.1.4 The common HMM types

- Ergodic : Every state of the model can be reached from every other state.
- Bakis: As time increases, states proceed from left to right.



Figures 3.4 and 3.5 are the common HMMs used, the ergodic and the left-to-right model respectively.

3.2 CDHMM system

CDHMM system also known as, HMM-GMM system, is the conventionally used system for speech recognition. It models each context-dependent phone with a generative model based on a left-to-right three state HMM topology. The total number of context-dependent phonetic states after tree-based clustering is of the order of a few thousands. Each state is denoted by the index j with $1 \leq j \leq J$. The observation vector is assumed to be generated within each HMM state j from a GMM:

$$P(x|j) = \sum_{i=1}^{M_j} w_{ji}\, N\left(x; \mu_{ji}; \Sigma_{ji}\right) \qquad (3.19)$$

where x is the observation vector, $w_{ji}, \mu_{ji} and\, \Sigma_{ji}$ are the prior, mean and covariance matrix of the i^{th} Gaussian component and M_i is the number of Gaussians in the j^{th} state.

3.3 Subspace Gaussian Mixture Model (SGMM)

SGMM is similar to the GMM-based system, but the model parameters for each state are specified by a single state vector v_j. Thus μ_{ji} lies in a state-independent subspace defined by the columns of M_i. The covariance is shared across all states, so that we have a state-independent Σ_i.

The basic model can be expressed as:

$$P(x|j) = \sum_{i=1}^{I} w_{ji}\, N(x; \mu_{ji}; \Sigma_i) \qquad (3.20)$$

$$\mu_{ji} = M_i v_j \qquad (3.21)$$

$$w_{ji} = \frac{\exp\,(w_i^T v_j)}{\sum_{i'=1}^{I} \exp\,(w_{i'}^T v_j)} \qquad (3.22)$$

where v_j is the state projection vector, x is the feature vector, M_i and w_i define the subspaces in which the means and the unnormalized log weights respectively lie and Σ_i is the shared covariance. j is the index of the context-dependent state ($1 \leq j \leq J$) with J in the order of a few thousands. i is the Gaussian index in the GMM of I mixtures (usually $200 < I < 2000$). v_j is the only state specific parameter. M_i w_i and Σ_i are "shared" parameters.

Using SGMM results in reduction in the number of state specific parameters and increase in the number of global parameters. Since the global parameters do not depend on a specific phone, there is a lot of data available to train the parameters. It is possible to train these parameters using out-of-domain data even from other languages as shown in Povey et al. (2011a).

3.3.1 Training Procedure

The training of the SGMM system begins with the traditional HMM-GMM system. First, a large GMM consisting of all the gaussians in the HMM-GMM system is built. This is typically in the order of tens of thousands. The gaussians are repeatedly merged to get a desired number of gaussians with diagonal covariances. These gaussians are trained with EM algorithm for full covariance re-estimation. The resulting model is called a Universal Background Model (UBM). The UBM can be viewed as a compact model representing all kinds of speech from all speakers. This UBM is used to initialize the SGMM model. This

is done in such a way that the initial p.d.f. of all states is equal to the UBM. The HMM-GMM system provides the Viterbi alignments for the initial SGMM parameter re-estimation iterations. Once the SGMM parameters are estimated by EM algorithm to a sufficient extent, the SGMM training can be continued with self-alignment (alignments from the SGMM itself).

3.3.2 Phones

The basic linguistic unit that we model is the phoneme or phones. There are around 40 phones in English language. Using only these gives a very simplistic model. For large vocabulary recognition, we need to consider the phones uttered before and after the phone in consideration which are called the left and the right context of the phone. This model with the left and right contexts is called a triphone model.

There are as many as 40^3 triphones possible, but many of them are not used in the training data. The GMMs used to model the triphones have many parameters to be estimated. We require a large amount of data to get a good estimate of the parameters. So, we "tie" similar triphones using a decision-tree based top-down clustering approach. The decision tree based clustering has been described in detail in Young et al. (1994). At the end of such a clustering process, we get a few thousand triphone models.

Chapter 4

Transform-based Phone CAT

Phone CAT (Srinivas et al. (2013)) is an acoustic modeling technique inspired from the Cluster Adaptive Training (CAT) (Gales (2000)) for rapid speaker adaptation. While the CAT adapts a speaker independent model to different clusters of speakers, the Phone CAT adapts a Universal Background Model (UBM) to a set of clusters representing the phones (monophones). The context-dependent phone (triphones) states are modeled as linear weighted interpolations of the phone cluster models, just as in the case of CAT where the model means for a speaker are obtained as a linear weighted interpolation of the cluster means corresponding to different speakers. The context information of the phone is captured in the form of a linear interpolation weight vector. This technique has many similarities to the SGMM (Povey et al. (2011a)), described in Section 2.4.

This technique exploits the correlations in the acoustic space between the distributions of the context dependent phone states and gives a very compact representation using a UBM and several MLLR transforms.Section 4.1 briefly describes the model-based Phone CAT technique. Section 4.2 introduces the Transform-based Phone CAT model. Sections 4.3 and 4.4 describe in detail the initialization of the model and the training procedure.

4.1 Model-based Phone CAT

The model-based Phone CAT consists of a set of P clusters corresponding to the P monophone models. Each cluster p has a cluster-specific mean $\mu_i^{(p)}$ for each Gaussian component $1 \leq i \leq I$ Each state j corresponding to a context-dependent HMM state is expressed as linear combination of the P cluster means with the interpolation weights v_j , which is called as the state vector. Thus the mean of the i^{th} Gaussian of the j^{th} state is modeled as follows:

$$\mu_{ji} = M_i v_j \qquad (4.1)$$

$v_j = [\, v_j^1 \quad v_j^2 \quad \ldots \quad v_j^P \,]$ is the state vector, and $M_i = [\, \mu_i^1 \quad \mu_i^2 \quad \ldots \quad \mu_i^P \,]$ is the matrix obtained by stacking the i^{th} mean of all the P phone clusters, where μ_i^p is the mean of the

i^{th} Gaussian of the p^{th} cluster.

The Model-based Phone CAT has 2 distinct model sets. At the lower level, there is a set of P monophone models. The monophone models cannot model the context. So, at the higher level, there are J triphone model states. The Model-based Phone CAT assumes that each of these tied states has a strong relation to the P monophone models; that it lies in a subspace spanned by the monophone models. (4.1) represents this relation. The monophone means $\mu_i^1 \ \mu_i^2 \ \dots \ \mu_i^P$ form the basis vectors of this subspace. During the training process, both the basis vectors and the interpolation weights are re-estimated; with the model in effect learning a better subspace.

4.2 Transform-based Phone CAT

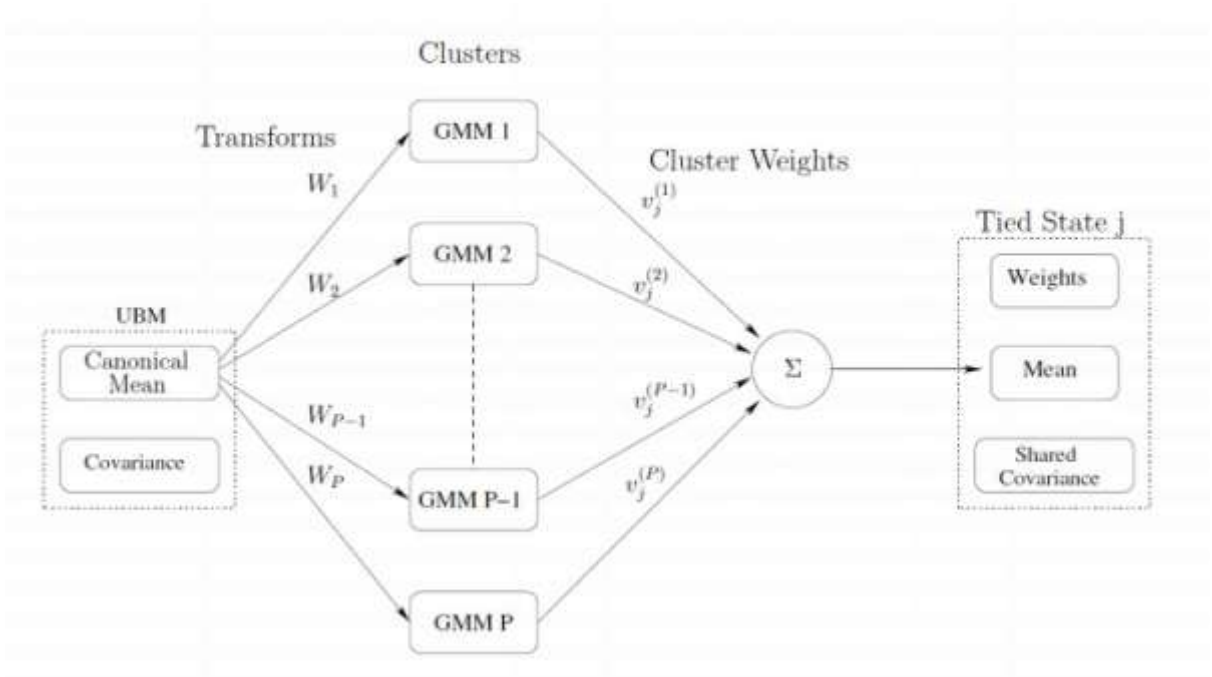


Figure 4.1 Transform-based Phone-CAT model

In the transform-based Phone CAT, the means of the P clusters, corresponding to the P monophones (it is a typical mapping, but may not necessarily correspond to P monophones), are not specified directly, but as linear transformations of the means of a canonical model. In the basic model, there is an MLLR transform, W_p associated with each cluster p. The cluster-specific mean μ_i^p for Gaussian component i is specified as:

$$\mu_i^p = W_p \xi_i = W_p [\mu_i \ 1]^T \qquad (4.2)$$

where ξ_i is the extended canonical model mean with μ_i being the canonical mean of the i^{th} Gaussian. The mean for the i^{th}Gaussian of the context-dependent state j is expressed as a weighted linear interpolation of the cluster-specific means given in (4.2)

$$\mu_{ji} = [\ \mu_i^1 \quad \mu_i^2 \quad \dots \quad \mu_i^P\ ]v_j = \sum_{p=1}^{P} \mu_i^p v_j^p$$

$$= (\sum_{p=1}^{P} v_j^p W_p)\xi_i \qquad (4.3)$$

$v_j = [\ v_j^1 \quad v_j^2 \quad \dots \quad v_j^P\ ]$ is the state vector

The Transform-based Phone CAT model has 3 distinct model sets. At the lowest level, there is a compact canonical model representing the average variability of all the speech data. At the intermediate level, there is a set of P clusters representing the P phone models. These P models are linear transformations, represented by (4.2) , of the canonical model. At the highest level, there is a set of J tied states, whose models are obtained as linear interpolation of the P models in the clusters.

The transform-based Phone CAT model has a GMM as the generative model in each context-dependent state. But the means are not specified directly, but with a mapping from the P dimensional state vector v_j . The covariance matrix Σ_i is diagonal and shared across all the context-dependent states. The weights are expressed through a subspace model similar to the SGMM (2.13). The model can be expressed as:

$$P(x|j) = \sum_{i=1}^{I} w_{ji}\, N\left(x; \mu_{ji}; \Sigma_i\right) \qquad (4.4)$$

$$w_{ji} = \frac{\exp\,(w_i^T v_j)}{\sum_{i'=1}^{I} \exp\,(w_{i'}^T v_j)} \qquad (4.5)$$

where x is the feature vector, $1 \leq j \leq$ J is the state index of the context-dependent state, w_i is the weight projection vector, $\mu_{ji}$is obtained as in (4.3) and I is the number of Gaussian components in the GMM. The number of Gaussians I is typically 400 to 4000. In the SGMM, typically a 400 mixture full-covariance matrix is used. Here, since the number of global parameters is lower, the number of mixtures can be higher. If the weights are modelled directly as rather than using (4.5), the number of parameters in the model will be dominated by the weights, which is undesirable. $\Sigma_i, w_i, W_p, \mu_i$ are global

parameters and v_j is the state specific parameter of the model.

4.3 Training procedure

The model training starts with a traditional HMM-GMM system. This provides the phonetic context information (the decision trees), a set of Gaussian mixtures to build a UBM as the canonical model and the Viterbi state alignments for the initial training iterations. The model is initialized using these and trained for a few iterations using the alignments obtained from the HMM-GMM system. In the next phase of training, the alignments are obtained from the transform-based Phone CAT system itself.

The state vector parameters $\Lambda = \{v_j\}$; $1 \leq j \leq J$, canonical model parameters $M = \{\{\mu_1, \dots, \mu_I\}, \{\Sigma_1, \dots, \Sigma_I\}\}$ and the subspace parameters $S = \{\{w, \dots, w\}, \{W_1, \dots, W_I\}\}$

The training scheme followed is:

1. Re-estimate the state vector parameters Λ using {M,S} and the pre-update value of Λ
2. Re-estimate the subspace parameters S given {*,M} and the pre-update value of S.
3. Re-estimate the canonical model parameters M given {S,*} and the pre-update value of M..
4. Go to 2 until convergence.
5. Go to 1 until convergence.

The pre-update values are used to calculate the Gaussian posteriors. These values are usually accumulated in the form of statistics. The structure of the model allows efficient pruning of the gaussians that are used for likelihood computation in each frame: only the top few gaussians in the UBM that give the highest likelihood for the frame are selected and used. The statistics accumulated and the update equations are described in Section 4.5.

4.4 Model initialization

First the UBM is trained and it is then used to initialize the transform-based Phone CAT model. The UBM is initialized by a bottom-up-clustering algorithm as in the case of SGMM (Povey et al. (2011a)). The set of diagonal Gaussians in all the states of the HMM-GMM system is clustered to create a mixture of diagonal Gaussians. This mixture

of Gaussians is further trained by EM algorithm using all the speech data to get the final UBM.

The transform-based Phone CAT model is initialized such that the GMM in each state is identical to the UBM. The MLLR transforms are all set to identity matrices with 0 bias so that all the cluster-specific means are initially identical to the UBM means. The state vectors v_j is assigned a vector giving a weight 1 to only one cluster depending on a mapping function C and 0 to every other cluster. Therefore the initialization is:

$$W_p = [I_{D\ X\ D} 0_{D\ X\ 1}], 1 \leq p \leq P \quad (4.6)$$

$$\mu_i = \mu_i^{UBM}, 1 \leq i \leq I \quad (4.7)$$

$$\Sigma_i = \Sigma_i^{UBM}, 1 \leq i \leq I \quad (4.8)$$

$$v_j = e_k, 1 \leq j \leq J, k = C(j) \quad (4.9)$$

$$w_i = 0, 1 \leq i \leq I \quad (4.10)$$

Where $I_{D\ X\ D}$is a D X D identity matrix, $0_{D\ X\ 1}$ $is\ a\ vector\ of\ zeros,$
e_k is a P dimensional unit vector with the k[th] dimension being 1 and other dimensions 0 and C:{1,...,J} → {1,...,P} is a mapping from state j to cluster p. If the context-dependent phone has 3 states, the context-dependent states corresponding to each of the 3 states can be mapped to different clusters. If every context-dependent phone has 3 states, then with this mapping the model will end up having P = 3K clusters, where K is the number of phones.

4.5 Training of the model

This section describes the accumulation and the update stages of the training of the model.

4.5.1 Expectation Maximization (EM) algorithm

The auxiliary function to be optimized is:

$$Q = \sum_{j,i,t} \gamma_{ji}(t)[\log(w_{ji}) - \frac{1}{2}|\Sigma_i| - \frac{1}{2}(x(t) - \mu_{ji})^T \Sigma_i^{-1}(x(t) - \mu_{ji})] \quad (4.11)$$

where $\gamma_{ji}(t)$ = P (j, i|t) is the posterior probability of the j[th] state, i[th] Gaussian component at time t,

x (t) is the feature vector at time t

w_{ji} and μ_{ji} are expressed according to (4.3) and (4.5).

The update equations for each of the parameters $\Sigma_i, w_i, W_p, \mu_i, v_j$ are obtained by optimizing Q with respect to the parameter keeping the other parameters fixed. The update equations along with the required accumulations are described in the subsequent sections

4.5.2 Cluster transform(W_p) re-estimation:

4.5.2.1 Diagonal covariance:

Using Gales(2000) method of re-estimating an entire row of a cluster transform W_p,

$$W_p^{(k)} = k_p^{(k)}[G_p^{(k)}]^{-1} \tag{4.12}$$

where

$$k_p^{(k)} = \sum_{i=1}^{I} \frac{1}{(\sigma_{kk}^i)^2} [\{k_{pk}^{(i)} - \sum_{l \neq p}^{P} g_{lp}^{(i)} W_l^{(k)} \xi_i\} \xi_i^T] \tag{4.13}$$

$$G_p^{(k)} = \sum_{i=1}^{I} \frac{g_{pp}^{(i)}}{(\sigma_{kk}^i)^2} \xi_i \, \xi_i^T \tag{4.14}$$

where

$$G^{(i)} = [g_{pq}^{(i)}]_{1 \leq p,q \leq P} = \sum_{j,t} \gamma_{ji}(t) v_j v_j^T \tag{4.15}$$

$$K^{(i)} = [k_{pk}^{(i)}]_{1 \leq p \leq P, 1 \leq k \leq D} = \sum_{j,t} \gamma_{ji}(t) v_j x(t)^T \tag{4.16}$$

4.5.2.2 Full covariance:

The update equations are quite complex and computationally very expensive, making it practically infeasible. The equation (4.12) is valid only for diagonal covariance. The re-estimation is done using a second order gradient descent approach.

In each iteration, the gradient of the auxiliary function Q is computed w.r.t W_p

$$L_p = \frac{\partial Q}{\partial W_p} = \sum_{j,i,t} \gamma_{ji}(t)\Sigma_i^{-1}(x(t) - \sum_{p=1}^{P}\left(v_j^p W_p\right)\xi_i)\xi_i^T v_j^p] \quad (4.17)$$

In each iteration, the second order gradient of the auxiliary function Q is computed w.r.t $W_p$assuming $\Sigma_i^{-1}$is diagonal for simplifying computations:

$$\frac{\partial^2 Q}{\partial W_p^{(k)2}} = \sum_{i=1}^{I} \gamma_{ji}(t)\frac{v_j^{(p)2}}{(\sigma_{kk}^i)^2}\xi_i\,\xi_i^T \quad (4.18)$$

The equation for the 2nd order gradient descent estimation of W_p is:

$$\widehat{W}_p^{(k)} = W_p^{(k)} + \alpha\left[\frac{\partial^2 Q}{\partial W_p^{(k)2}}\right]^{-1}\frac{\partial Q}{\partial W_p} \quad (4.19)$$

where $W_p^{(k)}$is the kth row of W_p, α is a learning rate.

The change in the auxiliary function Q is computed:

$$\Delta Q = \Delta\sum_{j,i,t}[\gamma_{ji}(t)x(t)^T\Sigma^{-1}\mu_{ji} - 0.5\gamma_{ji}\mu_{ji}^T\Sigma^{-1}\mu_{ji}] \quad (4.20)$$

If ΔQ is positive, we move on to the next iteration. Else, the learning rate is halved and $W_p^{(k)}$ is reset to its original value and $\widehat{W}_p^{(k)}$ is computed again. The auxiliary function is tested again for increase and the process is repeated until an increase is observed or until a limiting learning rate value is reached.

In the next iteration, the gradients and second gradients are computed again using the new updated value of $W_p^{(k)}$ After completing the estimation of one MLLR transform $W_p^{(k)}$, the next transform $W_{p+1}^{(k)}$ is estimated with (4.17) and (4.18) using the update value for $W_p^{(k)}$.

4.5.3 Estimation of State Vectors

Making several approximations, as in Povey (2009), a closed-form expression for the

update of v_j obtained is:

$$v_j = G_j^{-1} k_j \tag{4.21}$$

where the accumulates G_j^{-1} and k_j are given by

$$k_j = y_j + \sum_{i=1}^{I} w_i(\gamma_{ji} - \gamma_j w_{ji} + \max(\gamma_{ji}, \gamma_j w_{ji}) w_i^T v_j) \tag{4.22}$$

$$G_j = \sum_{i=1}^{I} H_i \gamma_{ji} + \max(\gamma_{ji}, \gamma_j w_{ji})\, w_i^T v_j \tag{4.23}$$

Where $H_i = M_i^T \Sigma_i^{-1} M_i$ (4.24)

$$y_j = \sum_{j,t} \gamma_{ji}(t) M_i^T \Sigma_i^{-1} x(t) \tag{4.25}$$

$$\gamma_{ji} = \sum_t \gamma_{ji}(t) \tag{4.26}$$

$$\gamma_j = \sum_i \gamma_{ji} \tag{4.27}$$

$$M_i = [W_1 \xi_i \ldots W_P \xi_i] \tag{4.28}$$

4.5.4 Estimation of Canonical model parameters

The update equations for the mean and covariance of the i[th] Gaussian component are:

$$\mu_i = \left[\sum_{p=1}^{P}\sum_{q=1}^{P} g_{pq}^{(i)} A_p^T \Sigma_i \mathrm{A}_q\right]^{-1} \left[\sum_{p=1}^{P} A_p^T \Sigma_i^{-1} \left(k_p^{(i)T} - \sum_{q=1}^{P} g_{pq}^{(i)} b_q\right)\right] \tag{4.29}$$

$$\Sigma_i = diag\left\{\frac{L^{(i)} - 2\sum_{p=1}^{P} k_p^{(i)} M_i^{(p)T} + \sum_{p=1}^{P}\sum_{q=1}^{P} g_{pq}^{(i)} M_i^{(p)} M_i^{(q)T}}{\sum_j \gamma_{ji}}\right\} \tag{4.30}$$

where A_p and b_p are the first D columns and the (D + 1)th column of $W_p = [\mathrm{A}_p \;\; \mathrm{b}_p]$

$$M_i^{(p)} = W_p \xi_i$$

$k_p^{(i)}$ is the p[th] row of statistics (4.16) ,

$$L^{(i)} = \sum_{j,t} \gamma_{ji}(t) x(t) x(t)^T \qquad (4.31)$$

4.5.5 Estimation of weight projections

It is an iterative process with the following being computed every iteration:

$$w_i^{(n)} = w_i^{(n-1)} + F_i^{(n)-1} g_i^{(n)} \qquad (4.32)$$

$$F_i^{(n)} = \sum_{j=1}^{J} \max\left(\gamma_{ji}, \gamma_j w_{ji}^{(n-1)}\right) v_j v_j^T \qquad (4.33)$$

$$g_i^{(n)} = \sum_{j} \left(\gamma_{ji} - \gamma_j w_{ji}^{(n-1)}\right) v_j \qquad (4.34)$$

where $.^{(n)}$ represents the value at the nth iteration. This is done similar to the way it is mentioned in SGMM (Dan Povey et al. (2011a))

Chapter 5

5.1 Approaches to optimize the Transform-based Phone-CAT algorithm

Time taken by the different sections of the training phase:

Timit Database is used for the following experiments.

Totally, there are 20 training passes and during each pass, cluster transforms are updated at the most 3 times (Full covariance matrices are used).

During every training pass, training of the below-mentioned parameters approximately takes the following time :

Part of the training process	**Time (in seconds)**
state vectors (v)	15
Cluster transforms (W)	150
Weight vector (w)	20
Canonical parameters (μ,Σ)	30
Computing posteriors and statistics for the next training pass	150

Table 5.1 showing the time consumed by the different parts of the training process of phone-CAT algorithm

It is observed that the rate determining portion of the algorithm is the re-estimation of the cluster transforms.

There are two main areas where this time is being spent:

1. The Gradient-Ascent method, which is used in the estimation of W_p, runs 3 * 40 (3 times for every cluster per re-estimation) * 20 (re-estimations) = 2400 times.
2. Computation of $\left[\frac{\partial^2 Q}{\partial W_p^{(k)2}}\right]^{-1} \frac{\partial Q}{\partial W_p}$

Approach 1:

The 2nd order gradient descent being used has a lot of computations (calculating inverse of the second differential of the Q function) though it converges faster. Instead, the learning rate can be updated after every iteration. The basic idea is that once the direction is found out, if there is a small increase, move faster (increase the step size optimally) and if there is a decrease by even a small quantity, reset to the pre-update value and reduce the learning rate.

When there is a decrease in Q, the algorithm is already resetting to the pre-update value and halving the learning rate. We need to ensure that we don't cross the point of optimum when we increase the learning rate. A suitable increase in the learning rate will get the cluster transform closer to the optimum value as we are already limiting the number of iterations of gradient descent algorithm to 3. This will either give an improved performance of the system (if # cluster transform iter = 3) or will make it approach the optimal point faster (if the # cluster transform iterations isn't limited to a value).

Approach 2:

Re-estimation equation for μ_i can be simplified in a way that we compute a particular D x D matrix and use it twice in every re-estimation process. This is different to the method in use where sufficient statistics are accumulated before every re-estimation.

This is how μ_i can be simplified:

$$\mu_i = [\sum_{j,t} \gamma_{ji}(t) \sum_{p} A_p^T v_j^{(p)} \Sigma_i^{-1} \sum_{q} A_q v_j^{(q)}]^{-1} [\sum_{j,t} \gamma_{ji}(t) \sum_{p} A_p^T v_j^{(p)} \Sigma_i^{-1} \left(x(t) - \sum_{q} b_q v_j^{(q)} \right)]$$

where

$$\sum_{j,t} \gamma_{ji}(t) \sum_{p} A_p^T v_j^{(p)} \Sigma_i^{-1}$$

is used twice but μ_i is computed differently in the present algorithm as seen before. But the disadvantage of this approach is that this simplification is specific to the parameter in

hand μ_i. Other parameter's re-estimation equations do not have terms from this equation repeating.

5.2 Investigating the performances of the various statistical models for speech Recognition

Timit corpus of read speech contains broadband recordings of 630 speakers of eight major dialects of American english, each reading ten phonetically rich sentences.

Experiment	Word Error Rate %
CDHMM monophone (15000 Gaussians)	**34.65** [2500/7215, 133 ins, 945 del, 1422 sub]
CDHMM Triphone (15000 Gaussians, 2500 Tied states)	**30.48** [2199/7215, 264 ins, 650 del, 1285 sub]
CDHMM LDA + MLLT (15000 Gaussians, 2500 Tied states)	**27.80** [2006/7215, 296 ins, 534 del, 1176 sub]
CDHMM LDA + MLLT + SAT (15000 Gaussians, 2500 Tied states)	**25.43** [1835/7215, 271 ins, 502 del, 1062 sub]
SGMM (**Triphone**, 700 Tied states, 400 mixture UBM)	**26.60** [1919/7215, 232 ins, 539 del, 1098 sub]
SGMM (**LDA + MLLT**, 700 Tied states, 400 mixture UBM)	**26.03** [1878/7215, 190 ins, 587 del, 1101 sub]
SGMM (**LDA + MLLT + SAT**, 700 Tied states, 400 mixture UBM)	**23.56** [1700/7215, 211 ins, 525 del, 964 sub]
Transform based Phone-CAT (**Triphone**, 2500 Tied-states, 400 mixture UBM, 39 clusters, Full covariance)	**26.99** [1947/7215, 202 ins, 617 del, 1128 sub]
Transform based Phone-CAT (**LDA + MLLT**, 2500 Tied-states, 400 mixture UBM, 39 clusters, Full covariance)	**26.07** [1881/7215, 225 ins, 568 del, 1088 sub]
Transform based Phone-CAT (**LDA + MLLT + SAT**, 2500 Tied-states, 400 mixture UBM, 39 clusters, Full covariance)	**23.94** [1727/7215, 201 ins, 529 del, 997 sub]

Table 5.2 showing the performances of the various statistical models in speech recognition

Baseline CDHMM is the conventional speech recognition system. It is used to initialize the SGMM and Transform-based Phone-CAT systems.

Observations:

- CDHMM system monophone model as expected gives a poor WER but with triphone models built and with LDA, MLLT and SAT, WER improves considerably.
- SGMM and Phone CAT systems also show improvement with the triphone models.
- Though SGMM full covariance outperforms the Transform-based phoneCAT full covariance result, the number of global parameters used in phoneCAT is lesser compared to that in SGMM.

CONCLUSIONS AND FUTURE WORK

- We conducted experiments to compare and investigate the performances of the various speech recognition systems, namely, the CDHMM, the SGMM and the Transform-based Phone-CAT systems.

- We observed that the rate-determining step of the phone CAT algorithm is the re-estimation process of the cluster transform(W_p) (2^{nd} order gradient descent along with the $\left[\frac{\partial^2 Q}{\partial W_p^{(k)2}}\right]^{-1} \frac{\partial Q}{\partial W_p}$ computation.

- Can look at a computationally efficient, feasible and a less complex alternative for the cluster transform estimation.

Bibliography

1. **Vimal Manohar, Bhargav Srinivas Ch., Umesh S**, "Acoustic modeling using Transform-based Phone-Cluster Adaptive Training" In Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop

2. **Povey, D., A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, et al.**, The kaldi speech recognition toolkit. In IEEE 2011 Workshop on Automatic Speech Recognition and Understanding. 2011b

3. **Rabiner, L. R. (1989)**. A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2), 257–286.

4. **Povey, D. (2009)**. A tutorial-style introduction to subspace gaussian mixture models for speech recognition. Technical Report MSR-TR-2009-111, Microsoft Research.

5. **Povey, D., L. Burget, M. Agarwal, P. Akyazi, K. Feng, A. Ghoshal, O. Glembek, N. K. Goel, M. Karafiát, A. Rastrow, et al.**, Subspace gaussian mixture models for speech recognition.In Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on. IEEE, 2010.49

6. **Povey, D. and G. Saon (2006)**. Feature and model space speaker adaptation with full covariance gaussians. Interspeech, paper.

7. **Garofolo, John, et al. (1993)** TIMIT Acoustic-Phonetic Continuous speech corpus, Linguistic Data Consortium

8. **Srinivas, B., N. M. Joy, R. R. Bilgi, and S. Umesh,** Subspace modeling technique using monophones for speech recognition. In Communications (NCC), 2013 National Conference on. 2013.

9. **Povey, D., L. Burget, M. Agarwal, P. Akyazi, F. Kai, A. Ghoshal, O. Glembek, N. Goel, M. Karafiát, A. Rastrow, R. C. Rose, P. Schwarz, and S. Thomas (2011a)**. The subspace gaussian mixture model - a structured model for speech recognition. Computer Speech & Language, 25(2), 404 – 439. ISSN 0885-2308.

.