The Concepts of Hidden Markov Model in Speech Recognition

Technical Report TR99/09

Waleed H. Abdulla and Nikola K. KasabovDepartment of

Knowledge Engineering Lab

Information Science Department

University of Otago

New Zealand

1999

The Concepts of Hidden Markov Model in Speech Recognition

Waleed H. Abdulla and Nikola K. Kasabov

Knowledge Engineering Lab,

Department of Information Science

University of Otago

New Zealand

1. Introduction

Speech recognition field is one of the most challenging fields that have faced the scientists from long time. The complete solution is still far from reach. The efforts are concentrated with huge funds from the companies to different related and supportive approaches to reach the final goal. Then, apply it to the enormous applications who are still waiting for the successful speech recognisers that are free from the constraints of speakers, vocabularies and environment. This task is not an easy one due to the interdisciplinary nature of the problem and as it requires speech perception to be implied in the recogniser (Speech Understanding Systems) which in turn strongly pointing to the use of intelligence within the systems.

The bare techniques of recognisers (without intelligence) are following wide varieties of approaches with different claims of success by each group of authors who put their faith in their favourite way. However, the sole technique that gain the acceptance of the researchers to be the state of the art is Hidden Markov Model (HMM) technique. HMM is agreed to be the most promising one. It might be used successfully with other techniques to improve the performance, such as hybridising the HMM with Artificial Neural Networks (ANN) algorithms. This doesn't mean that the HMM is pure from approximations that are far from reality, such as the successive observation independence, but the results and the potential of this algorithm is reliable. The modifications on HMM take the burden of releasing it from these poorly representative approximations hopping for better results.

In this report we are going to describe the back bone of the HMM technique with the main outlines for successful implementation. The representation and implementation of HMM varies in one way or another but the main idea is the same as well as the results and computation costs, it is a matter of preferences to choose one. Our preference here is the one adopted by Ferguson [1] and Rabiner et al. [2]-[5].

In this report we will describe the Markov Chain, and then investigating a very popular model in speech recognition field (the Left-Right HMM Topology). The mathematical formulation needed to be implemented will be fully explained as they are crucial in building the HMM. The prominent factors in the design will also be discussed. Finally we conclude this report by some experimental results to see the practical outcomes of the implemented model.

2. Markov Chains

The HMM algorithms are basically inspired from the more than 90 years old mathematical model known as Markov Chain. To understand the behaviour of the Markov Chain it is advisable to start with a simple real life example.

Let us consider a simple weather forecast problem and try to emulate a model that can predict tomorrow's weather based on today’s condition. In this example we have three stationary all day weather, which could be sunny (S), cloudy (C), or Rainy (R). From the history of the weather of the town under investigation we have the following table (Table-1) of probabilities of having certain state of tomorrow's weather and being in certain condition today:

		Tomorrow		
		Sunny(S)	Cloudy(C)	Rainy(R)
Today	Sunny(S)	.7	.2	.1
	Cloudy(C)	.05	.8	.15
	Rainy(R)	.15	.25	.6

Table-1 Weather expectation probabilities.

In this case what we are looking for is the weather conditional probability P(Tomorrow/Today). We realise that tomorrow's weather depends on today's condition as well as the previous several days, but we accept the assumption of tomorrow's weather depends only on today's condition as it is in consistency with the first order Markov chain. This assumption is greatly simplifying the problem of formulating the model even in the actual speech recognition case and we will use it when we come to tackle the real problem.

We refer to the weather conditions by state q that are sampled at instant t and the problem is to find the probability of weather condition of tomorrow given today's condition $P(q_{t+1}/q_t)$.

An acceptable approximation for n instants history is :

$$P(q_{t+1}/q_t , q_{t-1} , q_{t-2} , \ldots , q_{t-n}) \approx P(q_{t+1}/q_t).$$

This is the first order Markov chain as the history is considered to be one instant only. The finite state diagram of the weather probabilistic table is shown in Fig(1).

Let us now ask this question: Given today as sunny (S) what is the probability that the next following five days are S , C , C , R and S, having the above model?

The answer resides in the following formula using first order Markov chain:

$$\begin{aligned}P(q_1 = S, q_2=S, q_3=C, q_4=C, q_5=R, q_6=S) &= \\ P(S).P(q_2=S/q_1=S).\ P(q_3=C/q_2=S).\ P(q_4=C/q_3=C).\ P(q_5=R/q_4=C).\ P(q_6=S/q_5=R) & \\ &= 1 \times 0.7 \times 0.2 \times 0.8 \times 0.15 \times 0.15 \\ &= 0.00252\end{aligned}$$

The initial probability P(S) = 1, as it is assumed that today is sunny.

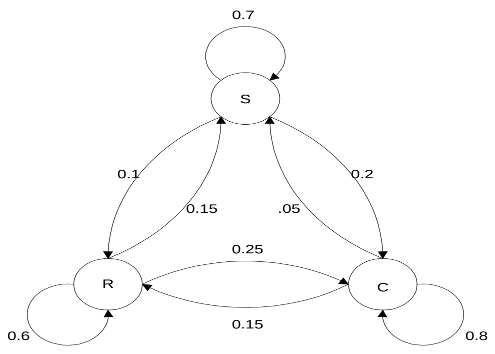


Fig. (1) Finite state representation of weather forecast problem.

3. Hidden Markov Model (HMM)

The particular problem presented in the previous section, the states were observable and they represented the weather conditions (S, C,R). They also represented the observation sequence. This kind of model formulation is very limited due to the need of observable state sequence which is unknown in most problem. The more general case is by considering the state sequence to be hidden (unobservable) and the observations are probabilistic functions of the state. This notion implies the double stochastic process. More precisely, the HMM is a probabilistic pattern matching technique in which the observations are considered to be the output of stochastic process and consists of an underlying Markov chain. It has two components: a finite state Markov chain and a finite set of output probability distribution. The first fruitful investigation of HMM was done by Baum et al. [6]-[8] in the late 60s and early 70s. The technique was applied to the speech recognition field by Baker [9].

To understand the HMM we prefer to start with a simple example inspired from that given by Rabiner et. al. [3]. Assume that we have two persons, one doing an experiment and the other is an outside observer. Let us consider that we have N urns (states) numbered from S_1 to S_N and in each urn there are M coloured balls (observations) distributed in different proportions. Also we have a black bag belongs to each urn, each bag contains 100 counters numbered by three numbers. These numbers are the current urn number S_i and the following two urns numbers S_{i+1} and S_{i+2} in probability proportions of .8, .15, and .05 respectively. The counters of the bag belonging to the urn just before the last are carrying one of two numbers only; S_{N-1} and S_N in probabilities of .9 and .1 respectively. We assume that the starting urn (state) is always urn1 (S_1) and we end up in urnN (S_N). The last urn need no bag as we suggest to stay their when we reach it till the end of the experiment. We start the experiment at time t =1 by drawing a ball from urn1 and register the colour then return it back to the urn. Then draw a counter from the corresponding urn bag. The expected possible numbers on the counters are: 1 (stay in urn1), or 2 (move to the next urn), or 3 (jump to the third urn). We continue with the same procedure of drawing a counter then a ball from the corresponding urn and registering the ball colours till we reach state N and stay there till the end of the experiment at instant T.

The outcome of this experiment is a series of coloured balls (observations) which could be considered as a sequence of events governed by the probability distribution of the balls inside each urn and by the counters existing in each bag. The outside observer has no idea about which urn a ball at any instant has drawn from (hidden states), what he knows is only the observation sequence of the coloured balls(observations).

Several things could be concluded from this experiment :

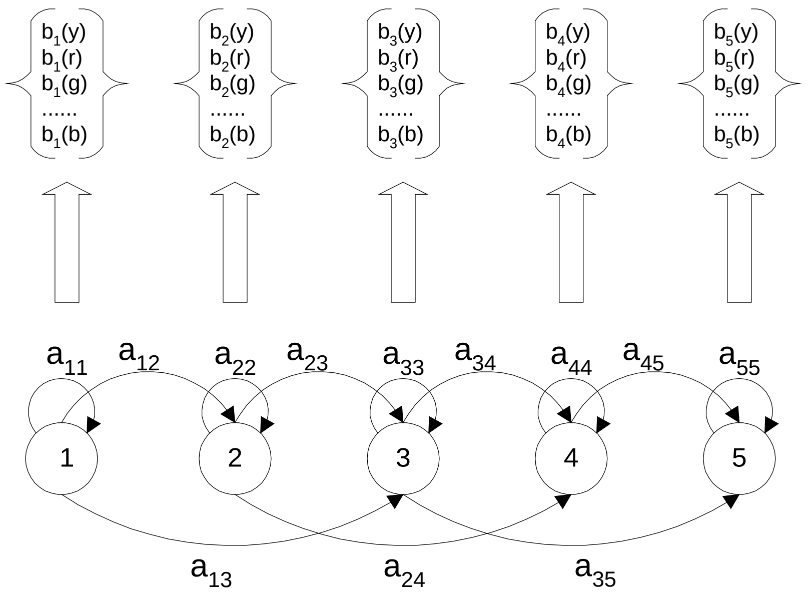
1 – The starting urn is always urn1 (S_1).

2 – The urn which has been left can not be visited again (i.e. moving from left to right direction).

3 – Movements are either by one or two urns to the right.

4 – The last urn visited is always urnN (S_N).

A chain of 5 urns (states) is shown in Fig.(2).



$b_1(y)$ $b_1(r)$ $b_1(g)$ ...... $b_1(b)$
$b_2(y)$ $b_2(r)$ $b_2(g)$ ...... $b_2(b)$
$b_3(y)$ $b_3(r)$ $b_3(g)$ ...... $b_3(b)$
$b_4(y)$ $b_4(r)$ $b_4(g)$ ...... $b_4(b)$
$b_5(y)$ $b_5(r)$ $b_5(g)$ ...... $b_5(b)$
a_{11} a_{12} a_{22} a_{23} a_{33} a_{34} a_{44} a_{45} a_{55}
1 2 3 4 5
a_{13} a_{24} a_{35}

π = { 1 0 0 0 0}

Fig.(2) States chain of the urn experiment using 5 urns. Each numbered circle represents a state and the arrows shows the states' flow during the whole process.

Fig.(2) shows the notations which we intent to use for the rest of the report and they are defined as follows:

a_{ij} represents the probability of state transition (probability of being in state S_j given state S_i)

$a_{ij} = P(q_{t+1}=S_j / q_t=S_i)$ ....(1)

$b_j(w_k)$ is the w_k symbol (ball colour) probability distribution in a state S_j
w is the alphabet and k is the number of symbols in this alphabet.

$\pi = \{1\ 0\ 0\ 0\ 0\ \}$ is the initial state probability distribution.

In this special case of states' chain topology $\pi_i = P(q_1 = S_i) = \begin{cases} 1 & \text{for} \quad i=1 \\ 0 & \text{for} \quad 1<i\leq N \end{cases}$

The model is completely defined by these three sets of parameters a, b, and π and the model of N states and M observations can be referred to by :

$\lambda = (A , B , \pi)$ ....(2)

where $A = \{a_{ij}\}$, $B = \{b_j(w_k)\}$ $1 \leq i , j \leq N$ and $1\leq k \leq M$.

The model that we have been described is a special type of HMM which is normally used in speech recognition. It is called Left-Right HMM as derived from its way of behaviour and its topology (moving from left to right during state transition). The reason for using the L-R topology of HMM is due to its inherent structure which can model the temporal flow of speech signal over time.

It might be not very obvious how the HMM related to the speech signal modelling[10]. This could be envisaged by looking at the speech production mechanism. Speech is produced by the slow movements of the articulatory organ. The speech articulators taking up a sequence of different positions and consequently producing the stream of sounds that form the speech signal. Each articulatory position could be represented by a state of different and varying duration. Accordingly, the transition between different articulatory positions (states) can be represented by $A = \{a_{ij}\}$. The observations in this case are the sounds produced in each position and due to the variations in the evolution of each sound this can be also represented by a probabilistic function $B = \{b_j(w_k)\}$.

The correspondence between the model parameters and what they represent in the speech signal is not unique and could be viewed differently. The important thing is to envisage the physical meanings of the states and observations in each view.

4. HMM Constraints for Speech Recognition Systems

HMM could have different constraints depending on the nature of the problem that wanted to be modelled. The main constraints needed in the implementation of speech recognisers can be summarised in the following assumptions[11]:

1 – First order Markov chain :

In this assumption the probability of transition to a state depends only on the current state

$P(q_{t+1}=S_j/q_t=S_i\ , q_{t-1}=S_k\ , q_{t-2}=S_w\ , \ldots\ , q_{t-n}=S_z\ ) \approx P(q_{t+1}=S_j\ /q_t=S_i)$ ....(3)

2 – Stationary states' transition

This assumption testifies that the states transition are time independent, and accordingly we will have:

$a_{ij} = P(q_{t+1}=S_j\ /\ q_t=S_i)$ for all t ....(4)

3 – Observations independence:

This assumption presumes that the observations come out within certain state depend only on the underlying Markov chain of the states, without considering the effect of the occurrence of the other observations. Although this assumption is a poor one and diviates from reality but it works fine in modelling speech signal.

This assumption implies that:

$P(O_t/O_{t-1}, O_{t-2}, \ldots,O_{t-p}\ , q_t\ , q_{t-1}\ , q_{t-2}\ ,\ldots\ q_{t-p}\ ) = P(O_t/\ q_t\ , q_{t-1}\ , q_{t-2}\ ,\ldots\ q_{t-p})$ ....(5)

where p represents the considered history of the observation sequence.

Then we will have :

$b_j(O_t) = P(O_t/q_t=j)$ ....(6)

4 – Left-Right topology constraint:

$a_{ij} = 0$ for all $j > i+2$ and $j < i$ ....(7)

$$\pi_i = P(q_1 = S_i) = \begin{cases} 1 & \text{for} \quad i=1 \\ 0 & \text{for} \quad 1<i\leq N \end{cases}$$ ....(8)

(i.e. $\pi = \{1\ \ 0\ \ \ldots\ldots\ \ 0\ \}$)

5 – Probability constraints:

Our problem is dealing with probabilities then we have the following extra constraints:

$$\sum_{j=1}^{N} a_{ij} = 1 \qquad \text{..................(9)}$$

$$\sum_{j=1}^{N} \pi_j = 1 \qquad \text{..................(10)}$$

$$\int_{O} b_i(O_t)dO = 1 \qquad \text{..................(11)}$$

If the observations are discrete then the last integration will be a summation.

5. The principal cases of HMM

There are three main cases to be dealt with to formulate a successful HMM. These are:

Case 1: Evaluation

Given:

- a model λ = (A , B , π) ready to be used.
- testing observation sequence O = O_1 , O_2 , O_3 ,.........., O_{T-1} , O_T .

Action:

- compute P(O/λ) ; the probability of the observation sequence given the model.

Case 2: Decoding

Given:

- a model λ = (A , B , π) ready to be used.
- testing or training observation sequence O = O_1 , O_2 , O_3 ,.........., O_{T-1} , O_T .

Action:

- track the optimum state sequence Q = q_1 , q_2 , q_3 ,........., q_{T-1} , q_T that most likely produce the given observations, using the given model.

Case 3: Training

Given :

- a model λ = (A , B , π) ready to be used.
- training observation sequence O^k = O_1^k , O_2^k , O_3^k ,.........., O_{T-1}^k , O_T^k

  where k is the number of examples for training the model.

Action:

- Tune the model parameters to maximise P(O/ λ).

Case 1 is an evaluation procedure as we are seeking to find the probability of producing given observation O by a given model λ. This could be used to find out the best model among many who produces the given observation.

Case 2 is a decoding procedure to detect or unhide the state sequence of a given observation. The observations could be training examples if we want to study the behaviour of each state from different aspects, such as states' duration or spectral characteristics of each state. Some techniques utilise the state duration in their evaluation procedure and in this case the observation will be the test example to detect the states' duration.

Case 3 is the training procedure to optimise the model parameters to obtain the best model that represent certain set of observations belonging to one spoken entity.

The way is paved now to tackle an important goal of our task, namely derivation of the mathematical formulas to the three previous cases.

5–1. Case 1 Formulation

Let us take a simple case then generalise to the complete one. Consider that we have 3 states and 5 observations in a process and we want to find P(O/λ). To explain the whole flow of the process the trellis diagram of Fig.(3) is of big help. The state at each instant is represented by a small circle, and the arrows represent the state transitions.

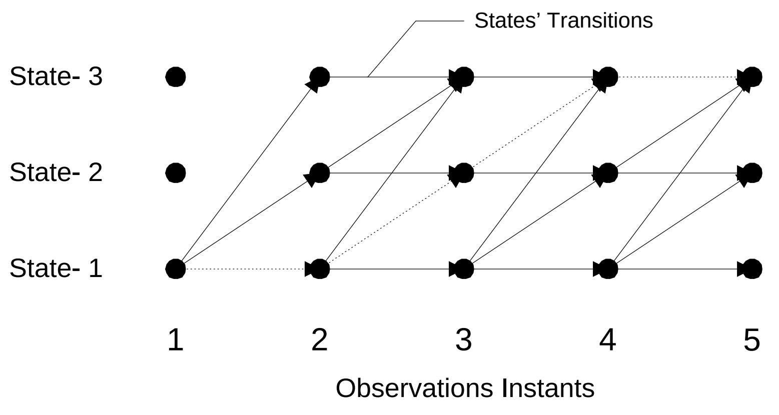


Fig.(3) Trellis Diagram of 3 States, and 5 Instants L-R Model

From Fig(3) we can see all the possibilities that the events might take during the whole process. The dotted lines show one possibility in which $P(O^1/Q^1,\lambda)$ is computed by:

$$\boldsymbol{P}(\boldsymbol{O}^1/\boldsymbol{Q}^1,\lambda)=\prod_{t=1}^{5}\boldsymbol{P}(\boldsymbol{O}_t^1/\boldsymbol{q}_t^1 \qquad \text{..................(12)}$$

$$\boldsymbol{P}(\boldsymbol{O}^1/\boldsymbol{Q}^1,\lambda)=\boldsymbol{b}_{q1}^1.\boldsymbol{b}_{q2}^1.\boldsymbol{b}_{q3}^1.\boldsymbol{b}_{q4}^1.\boldsymbol{b}_{q5}=\boldsymbol{b}_1^1.\boldsymbol{b}_1^1.\boldsymbol{b}_2^1.\boldsymbol{b}_3^1.\boldsymbol{b}_3^1 \qquad \text{..................(12a)}$$

$$\boldsymbol{P}(\boldsymbol{Q}^1/\lambda)=\pi_1\sum_{t=1}^{T-1}\boldsymbol{a}_{q_t q_{t+1}} \qquad \text{..................(13)}$$

$$\boldsymbol{P}(\boldsymbol{Q}^1/\lambda)=\pi_1.\boldsymbol{a}_{q1q2}.\boldsymbol{a}_{q2q3}.\boldsymbol{a}_{q3q4}.\boldsymbol{a}_{q4q5}=\pi_1.\boldsymbol{a}_{11}.\boldsymbol{a}_{12}.\boldsymbol{a}_{23}.\boldsymbol{a}_{33} \qquad \text{..................(13a)}$$

$$\boldsymbol{P}(\boldsymbol{O}^1,\boldsymbol{Q}^1/\lambda)=\boldsymbol{P}(\boldsymbol{O}^1/\boldsymbol{Q}^1,\lambda)\boldsymbol{P}(\boldsymbol{Q}^1/\lambda) \qquad \text{..................(14)}$$

$$\boldsymbol{P}(\boldsymbol{O}^1,\boldsymbol{Q}^1/\lambda)=\boldsymbol{b}_1^1.\boldsymbol{b}_1^1.\boldsymbol{b}_2^1.\boldsymbol{b}_3^1.\boldsymbol{b}_3^1.\pi_1.\boldsymbol{a}_{11}.\boldsymbol{a}_{12}.\boldsymbol{a}_{23}.\boldsymbol{a}_{33} \qquad \text{..................(14a)}$$

This procedure has to be done for all possible states' sequences (paths). The superscripts of O and Q indicate the possibility number. Then the probabilities of all the paths has to be summed to get the overall probability of how likely the model produces the given observation sequence.

$$P(O/\lambda)=\sum_{i=1}^{p}P(O^i,Q^i/\lambda) \qquad \text{...........................(15)}$$

$$P(O/\lambda)=\sum_{i=1}^{p}P(O^i/Q^i,\lambda)P(Q^i/\lambda) \qquad \text{...........................(15a)}$$

where p is the number of possible paths.

The total number of possibilities increases exponentially with the increasing number of states and observation instances. The Left-Right topology is substantially reducing the number of possible paths over the full connection topology (ergodic models in which every state could be reached from any other state at any instant).

Further reduction in the computational cost can be achieved by the Forward-Backward Procedure[12]. This technique greatly reduces the computational cost with simple iterative mathematical formulas. Actually it is a compound procedure composed of forward procedure and backward procedure. In the evaluation case we only need one of them and the forward procedure will be our preference.

5-1.1 Forward Procedure

Initially consider a new forward probability variable $\alpha_t(i)$, at instant *t* and state *i* , has the following formula:

$$\boldsymbol{\alpha_t(i) = P(O_1, O_2, O_3, \ldots\ldots, O_t, q_t = S_i / \lambda)} \qquad \ldots\ldots\ldots\ldots(16)$$

This probability function could be solved for N states and T observations iteratively:

1 – Initialisation

$$\boldsymbol{\alpha_1(i) = \pi_i . b_i(O_1)} \qquad 1 \le \boldsymbol{i} \le \boldsymbol{N} \qquad \ldots\ldots\ldots\ldots\ldots\ldots(17)$$

2 – Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \qquad \begin{matrix} 1 \le t \le T-1 \\ 1 \le j \le N \end{matrix} \qquad \ldots\ldots\ldots\ldots(18)$$

Fig.(4) shows the induction step graphically. It is clear from this figure how state S_j at instant t+1 reached from N possible states at instant t.

3 – Termination

$$P(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i) \qquad \ldots\ldots\ldots\ldots\ldots\ldots(19)$$

This stage is just a sum of all the values of the probability function $\alpha_T(i)$ over all the states at instant T. This sum will represent the probability of the given observations to be driven from the given model. That is how likely the given model produces the given observations. The proof of the termination formula will be given later on.

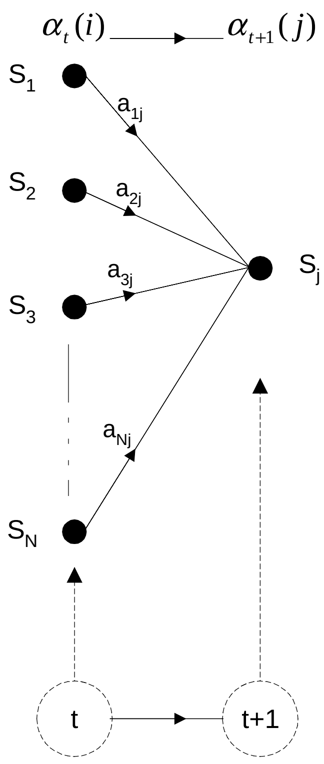


Fig.(4) Forward Probability Function Rpresentation

5-1.2 Backward Procedure

This procedure is similar to the forward procedure but it takes into consideration the state flow as if in backward direction from the last observation entity, instant T, till the first one, instant 1. That means that the access to any state will be from the states that are coming just after that state in time and as shown in Fig.(5).

To formulate this approach let us consider the backward probability function $\beta_t(i)$ which can be defined as:

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, \ldots\ldots\ldots, O_T / q_t = S_i, \lambda). \qquad \ldots\ldots\ldots\ldots(20)$$

In analogy to the forward procedure we can solve for $\beta_t(i)$ in the following two steps:

1 - Initialisation:

$$\beta_T(i) = 1, \qquad 1 \le i \le N. \qquad \ldots\ldots\ldots\ldots(21)$$

These initial values for β's of all states at instant T is arbitrarily selected.

2 – Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij}.b_j(O_{t+1}).\beta_{t+1}(j), \quad t = T-1, T-2, ..., 1 \quad , 1 \le i \le N \quad ....(22)$$

Equation (22) can be well understood with help of Fig.(5). We are still looking from left to right in calculating the partial probability function β (from *t* to *T*)

Fig.(5) shows this behaviour clearly. Even we are still looking from left to right in calculating the partial probability function (from *t* to *T*). However, at each instant we consider that we have β at *t+1* and we need to calculate it at time *t*; as if we are moving backward in time.

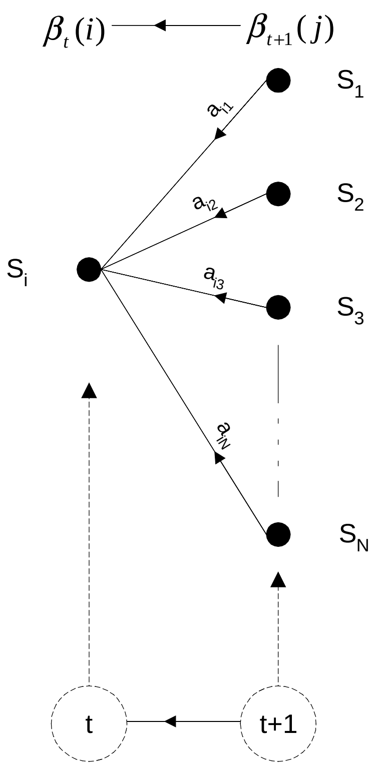


Fig.(5) Backward Probability Function Rpresentation

5-1.3 Computing P(O/λ) from Forward and Backward Probability Functions

The probability function of the model P(O/λ) can be computed from both α and β functions. Fig.(6) demonstrates this computation graphically. At instant t, the event of being in state q_i and moving to state q_j at instant t+1 is calculated by $\alpha_t(i)$ which accounts for the path termination in state q_i. The transition to state q_j is weighted by the product $a_{ij} \, . \, b_j(O_{t+1})$. At instant t+1 the event of observation sequence to the instant T starting from state S_j, while being at state S_i during instant t, is represented by the backward probability function $\beta_{t+1}(j)$.

Then $P(O/\lambda)$ is directly concluded to be :

$$P(O/\lambda) = \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i).a_{ij}.b_j(O_{t+1}).\beta_{t+1}(j) \quad \text{........................(23)}$$

Substitute (22) in (23) to get:

$$\boldsymbol{P(O \mid \lambda) = \sum_{i=1}^{N}\alpha_t(i)\beta_t(i)} \quad \text{......................(23a)}$$

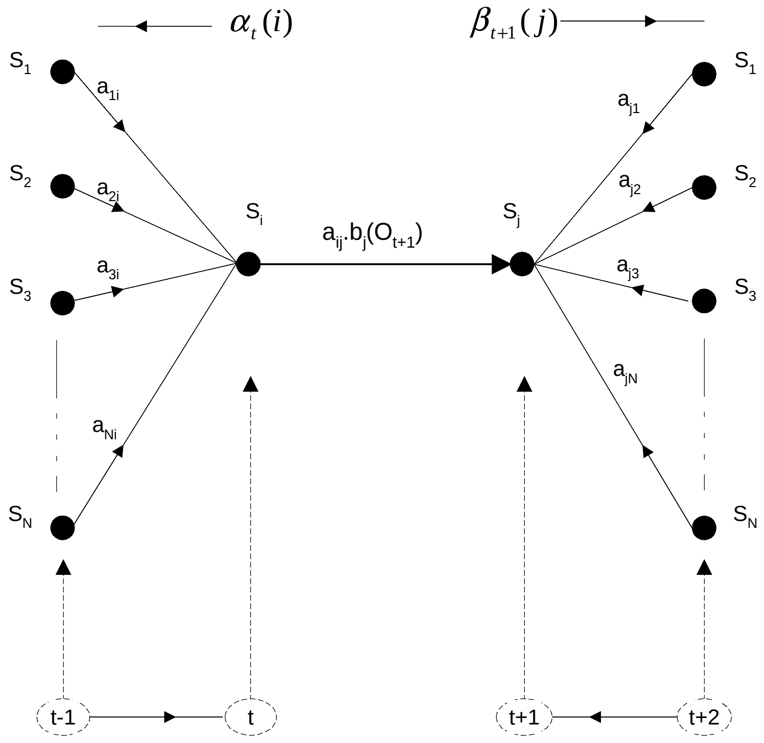


Fig.(6) Forward - Backward Probability Functions to find $P(O/\lambda)$

5-1.4 Proof of Termination Formula in Forward Probability Function

From (18) we have :

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \qquad \begin{array}{l} 1 \le t \le T-1 \\ 1 \le j \le N \end{array} \quad ....(18)$$

Let t = T-1 and substitute it in (18) to get:

$$\alpha_T(j) = \left[\sum_{i=1}^{N} \alpha_{T-1}(i).a_{ij}\right].b_j(O_T) \quad ....(18a)$$

$$\alpha_T(j) = \sum_{i=1}^{N} \alpha_{T-1}(i).a_{ij}.b_j(O_T) \quad ....(18b)$$

From (23) we have:

$$P(O/\lambda) = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i).a_{ij}.b_j(O_{t+1}).\beta_{t+1}(j) \quad ....(23)$$

Let t = T-1 in (23) to get:

$$P(O/\lambda) = \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_{T-1}(i).a_{ij}.b_j(O_T).\beta_T(j) \quad ....(23a)$$

From (21) we have:

$$\beta_T(i) = 1, \qquad 1 \le i \le N.$$

Substitute for $\beta_T(i)$ in (23a) and rearrange the equation to get:

$$P(O/\lambda) = \sum_{j=1}^{N}\left[\sum_{i=1}^{N} \alpha_{T-1}(i).a_{ij}.b_j(O_T)\right] \quad ....(24)$$

The term inside the square brackets is the same as that in (18b), substitute it and you will get the final needed formula:

$$P(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i) \qquad ....(19)$$

5-2. Case 2 Formulation

This case deals with the uncovering the hidden states of the model given the observation sequence and the model. This means that we have to find the optimal state sequence $Q = (q_1, q_2, q_3, \dots, q_{T-1}, q_T)$ associated with the given observation sequence $O = (O_1, O_2, O_3, \dots, O_{T-1}, O_T)$ presented to the model $\lambda = (A, B, \pi)$. The criteria of optimality her is to search for a single best state sequence through modified dynamic programming technique called Viterbi Algorithm[13]. We need to maximise $P(Q/O,\lambda)$ to detect the best state sequence. This could be achieved via maximising the joint probability function $P(Q,O/\lambda)$ using to the Baysian Rule which states that:

$$P(Q/O,\lambda) = \frac{P(Q,O/\lambda)}{P(O/\lambda)} \quad \text{....................(25)}$$

The denominator has nothing to share in maximising $P(Q/O,\lambda)$ as it doesn't include the state sequence factor Q. To go through the Viterbi Algorithm method let us define the probability quantity $\delta_t(i)$ which represents the maximum probability along the best probable state sequence path of a given observation sequence after t instants and being in state i. This quantity can be defined mathematically by:

$$\delta_t(i) = \max_{q_1, q_2, \dots\dots, q_{t-1}} P[q_1 q_2 \dots\dots q_{t-1}, q_t = S_i, O_1 O_2 \dots\dots O_t / \lambda] \quad \text{...........(26)}$$

The best state sequence is backtracked by another function $\psi_t(j)$. The complete algorithm can be described by the following steps:

Step 1: Initialisation

$$\delta_1(i) = \pi_i b_i(O_1) \;, \qquad 1 \le i \le N \qquad \text{.....................(27)}$$

$$\psi_1(i) = 0 \qquad \text{.....................(28)}$$

Step 2: Recursion

$$\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] b_j(O_t) \;, \quad 2 \le t \le T \;, \; 1 \le j \le N \quad \text{...............(29)}$$

$$\psi_t(j) = \arg\max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] \quad , \quad 2 \le t \le T \;, \; 1 \le j \le N \quad \text{................(30)}$$

Step 3: Termination

$$P^* = \max_{1\le i\le N}[\delta_T(i)] \qquad \text{......................}(31)$$

$$q_T^* = \arg\max_{1\le i\le N}[\delta_T(i)] \qquad \text{......................}(32)$$

Step 4: Backtracking

$$q_t^* = \psi_{t+1}(q_{t+1}^*) \ , \quad T-1 \ge t \ge 1$$

It is clear that (29) of Viterbi recursion is similar to (18) of forward induction, except the interchange of summation by maximisation. One thing could be noted here is that Viterbi Algorithm can also be used to calculate the P(O/λ) approximately by considering the use of P^* instead. This is acceptable as it gives comparable results and can be justified through the modified equation (15) to do the summation on the most probable state sequence, which has the major weight among all the possible states' paths.

5-3. Case 3 Formulation

This case is dealing with the training issue which is the most difficult one in all the three cases. The task of this case is to adjust the model parameters, (A ,B,π), according to a certain optimality criteria. There are many techniques to achieve the task of this case and we will describe here the well known Baum-Welch Algorithm, called also Forward –Backward Algorithm. It is an iterative method to reach the local maximas of the probability function P(O/λ). Each time the model parameters are adjusted to get a new model which is proved by Baum et. al. that the new model is either better or reach a critical condition at which the iteration has to be stopped as the local minima has reached. The model is always converge but the global maximisation can not be assured. Fig.(7) shows the non-linear optimisation of this problem and how the global optimality seeking is difficult to locate and greatly depending on the initial point of search.

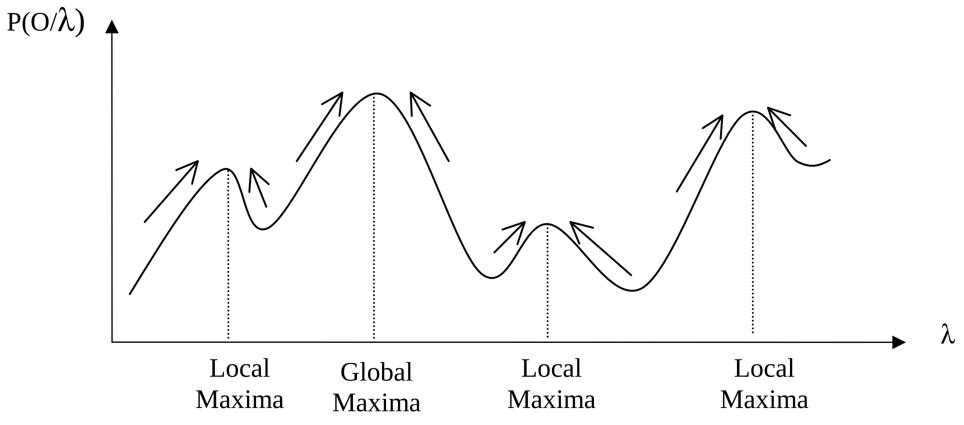


Fig.(7) Optimum Search Possibilities.

To go through the training procedure let us first define a posteriori probability function $\gamma_t(i)$, the probability of being in state i at instant t, given the observation sequence *O* and the model λ. as:

$$\gamma_t(i) = P(q_t = S_i \mid O, \lambda) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(34)$$

$$\gamma_t(i) = \frac{P(O, q_t = S_i \mid \lambda)}{P(O \mid \lambda)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(35)$$

Since

$$P(O, q_t = S_i \mid \lambda) = \alpha_t(i)\beta_t(i) \quad \ldots\ldots\ldots\ldots\ldots\ldots(36)$$

and from (23a)

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_t(i)\beta_t(i) \quad \ldots\ldots\ldots\ldots\ldots\ldots(23a)$$

Then

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \quad \ldots\ldots\ldots\ldots\ldots\ldots(37)$$

Let us define another probability function $\xi_t(i,j)$, the probability of being in state i at instant t and going to state j at instant t+1, given the model λ and the observation sequence *O*.

$\xi_t(i,j)$ can be mathematically defined as:

$$\boldsymbol{\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(38)$$

Multiply both sides of (39) by P(O/λ) to get :

$$\boldsymbol{\xi_t(i,j).P(O \mid \lambda) = P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda).P(O \mid \lambda)} \ldots\ldots\ldots\ldots\ldots(39)$$

From Bayesian Rule

$$\boldsymbol{P(q_t = S_i, q_{t+1} = S_j \mid O, \lambda).P(O \mid \lambda) = P(O, q_t = S_i, q_{t+1} = S_j \mid \lambda)} \ \ldots..(40)$$

The right hand side of (40) can be represented by the forward α and backward β functions , with the help of Fig.(6), as follows:

$$P(O, q_t = S_i, q_{t+1} = S_j \mid \lambda) = \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots(41)$$

Substitute (23a) and (41) in (39) and rearrange to get:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \alpha_t(i) \beta_t(i)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(42)$$

Also, from (23) we can have:

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(43)$$

The relation between $\gamma_t(i)$ and $\xi_t(i,j)$ can be easily deduced from their definitions :

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(44)$$

Now, if $\gamma_t(i)$ is summed over all instants (excluding instant T) we get the expected number of times that state S_i has left, or the number of times this state has been visited over all instants. On the other hand if we sum $\xi_t(i,j)$ over all instants (excluding T) we will get the expected number of transitions that have been made from i to j.

From the behaviour of $\gamma_t(i)$ and $\xi_t(i,j)$ the following re-estimations of the model parameters could be deduced:

$$\hat{\pi}_i = \text{expected number of instants the starting state is } S_i$$

$$= \gamma_1(i) \qquad \text{................(45)}$$

$$\hat{a}_{ij} = \frac{\text{expected number of transitions from } S_i \text{ to } S_j}{\text{expected number of transitions from } S_i}$$

$$= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i) \beta_t(i)} \qquad \text{.............(46)}$$

$$\hat{b}_j(k) = \frac{\text{expected number of instants in state } S_j \text{ and having observation } O_t = w_k}{\text{expected number instances in state } S_j}$$

$$= \frac{\sum_{\substack{t=1 \\ O_t = w_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} = \frac{\sum_{t:O_t = w_k} \alpha_t(j) \beta_t(j)}{\sum_{t=1}^{T} \alpha_t(j) \beta_t(j)} \qquad \text{.............(47)}$$

After the re-estimation of the model parameters we will have another model $\hat{\lambda}$ which is more likely, than model λ, producing observation sequence *O*. This means that

$$P(O/\hat{\lambda}) > P(O/\lambda)$$

This process of re-estimation can be continued till no improvement in $P(O/\lambda)$ is reached, that is we reach local maxima.

6. Discrete Hidden Markov Model (DHMM)

HMM modelling methods applied so far are for process that has discrete observation sequence. These observations could be the outcome indices of Vector Quantization technique (VQ) [14],[15]. VQ is a technique of clustering time series signal, in our case speech signal, into certain number of bins (clusters). Each bin represents the data belong to a certain population with similar (or minimum difference) spectral characteristics. The centre of gravity of each bin is assigned to a certain index and considered as the representative of the cluster population in any process on the signal. The long sequence of speech samples will be represented by stream of indices representing frames of different window lengths. Hence, VQ is considered as a process of redundancy removal, which minimises the number of bits required to identify each frame of speech signal. VQ was initially used successfully with Dynamic Time Warping (DTW) to recognise spoken words, and then proved to be successful with HMM as well. The role of VQ in HMM is to prepare discrete symbols from a finite alphabet. Each speech input will be quantized by the VQ reference bins. Each quantized input will be then considered as an observation. There are many other methods to represent the observations which are not belong to the task of this report, but a very good reference to recommend is [16].

The type of HMM that models speech signals based on VQ technique to produce the observations is called Discrete Hidden Markov Model (DHMM). It is efficient and reliable technique which has comparable results to the more computational DTW technique. In addition the phones, phonemes, and subwords could be modelled easily with DHMM while it is very difficult with DTW as the later needs to detect the segments boundary for comparison. However, VQ is responsible for loosing some information from the speech signal even when we try to increase the codewords. This lose is due to the quantization error (distortion). This distortion can be reduced by increasing the number of codewords in the codbook but cannot be eliminated.

7. Continuous Hidden Markov Model (CHMM)

It is a more sophisticated methodology to develop an improved HMM model of the speech signal. This method, even it needs more memory than DHMM to represent the model parameters but it is not suffering from the distortion problem. On the other hand it needs more deliberate techniques to initialise the model as it might diverge easily with randomly selected initial parameters.

In CHMM the model parameters are also π, A, and B, but they are represented differently. The probability density function (pdf) of certain observations *O* being in a state is considered to be of Gaussian Distribution (other distributions also valid). Let us consider it to be $b_i(O)$ and has the following general form:

$$b_i(O) = \sum_{m=1}^{M} c_{im} \aleph(O; \mu_{im}, U_{im}), \quad 1 \le i \le N \qquad \text{..............................}(48)$$

where:

c_{jm} : is the m-th mixture gain coefficient in state *i*.

$\aleph$: is the pdf distribution which is considered to be Gaussian in our case.

μ_{im} : is the mean of the m-th mixture in state *i*.

U_{im} : is the covariance of the m-th mixture in state *i*.

O : is the observation sequence of the feature vectors of dimension d.

M : is the number of mixtures used.

N : is the number of states.

The following constraints has to be fulfilled to insure the consistency of the model parameters estimation.

$$\sum_{m=1}^{M} c_{im} = 1, \qquad 1 \le i \le N \qquad \text{...................}(49a)$$

$$c_{im} \ge 0 \quad , \quad 1 \le i \le \mathrm{N} \ , \ 1 \le m \le M \qquad \text{...................}(49b)$$

These constraints will lead to proper pdf normalisation, that is

$$\int_{-\infty}^{\infty} b_i(O)\,dO = 1, \text{ for } \quad 1 \le i \le N \qquad \text{....................}(50)$$

The pdf of the observations will be of the form:

$$b_i(O) = \frac{1}{(2\pi)^{d/2}\sqrt{|U_i|}} e^{-\frac{1}{2}(O-\mu_i)' U_i^{-1} (O-\mu_i)} \quad \text{....................(51)}$$

where prime (’) superscript here is referring to the transpose of matrix.
The covariance matrix in (51) could be simplified by using diagonal matrix with elements representing the variance of each mixture. This approximation greatly reduces the computational cost in spite of the necessity to increase the number of mixtures to make it work better.
The reestimation formulas in multimixture continuous density HMM will be as follows:

$$\hat{c}_{im} = \frac{\text{expected instances of being in state i and mixture m}}{\text{expected instances of being in state i}}$$

$$\hat{c}_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m)}{\sum_{t=1}^{T}\sum_{m=1}^{M} \gamma_t(i,m)} \quad \text{....................(52)}$$

$$\hat{\mu}_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m).O_t}{\sum_{t=1}^{T} \gamma_t(i,m)} \quad \text{....................(53)}$$

$$\hat{U}_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m).(O_t - \mu_{im})(O_t - \mu_{im})'}{\sum_{t=1}^{T} \gamma_t(i,m)} \quad \text{....................(54)}$$

where $\gamma_t(i,m)$ is the probability of being in state i with m-th mixture at instant t. It is the same as $\gamma_t(i)$ when m=1.

The following equation represents the modified version of (38) to make it suitable for multimixture case:

$$\gamma_t(i,m) = \left[\frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N}\alpha_t(i)\beta_t(i)}\right]\left[\frac{\mathrm{c}_{\mathrm{im}}\aleph(O_t;\mu_{im},U_{im})}{\sum_{m=1}^{\mathrm{M}}\mathrm{c}_{\mathrm{im}}\aleph(O_t;\mu_{im},U_{im})}\right] \quad \text{........................(55)}$$

For the initial state and the state transition probability distributions they are the same as for DHMM as in (45) and (46).

8. Mixture Density Components Estimation using Maximum Likelihood (ML):

The ML estimation is an optimisation technique that can be used efficiently in estimating the different component of multimixture models. We are not going through the mathematical derivations of the ML but we only describe the method to be used in our task.

Let us first make some definitions:

$b_i(O_t)$: probability of being in state i given observation sequence $O_{t.}$

c_{im} : probability of being in state i with mixture m (gain coefficient).

$b_{im}(O_t)$: probability of being in state i with mixture m and given O_t.

$\Phi(w_{im}|O_t)$: probability function of being in a mixture class w_{im} given O_t in state i.

T_i : is the toatal number of observations in state i .

T_{im} : is the number of observations in state i with mixture m.

N : number of states.

M : number of mixtures in each state.

Now we are ready to implement the algorithm through applying the following steps:

1 – Take several versions of observations of certain word say digit zero spoken several times by many speakers.

2 – Apply Veterbi algorithm to detect the states of each version of the spoken word.

3 – Put the whole observations belonging to each state from all the versions of the spoken word into separate cells. Now we have N cells and each one represents the population of certain state derived from several observation sequences of the same word.

4 – Apply vector quantization technique to split the population of each cell into M mixtures and getting w_M classes within each state.

5 – Using the well known statistical methods to find the mean μ_{im} and the covariance U_{im} of each class. The gain factor c_{im} can be calculated by:

$$c_{im} = \frac{\text{number of observations benig in state i and mixture m}}{\text{total number of observations in state i}} \quad \ldots\ldots\ldots\ldots.(56)$$

6 – Calculate $\Phi(w_{im}|O_t)$ from the following formula:

$$\Phi(w_{im} \mid O_t) = c_{im}.\frac{b_{im}(O_t)}{b_i(O_t)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots.(57)$$

7 – Find the next estimate of $\hat{c}_{im}, \overset{\wedge}{\mu}_{im}$, and $\overset{\wedge}{\mathrm{U}}_{\mathrm{im}}$ from the formulas given by ML :

$$\hat{c}_{im} = \frac{1}{T_i}\sum_{t=1}^{T_i}\Phi(w_{im} \mid O_t) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.(58)$$

$$\hat{\mu}_{im} = \frac{1}{T_{im}}\sum_{t=1}^{T_i}\Phi(w_{im} \mid O_t).O_t \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.(59)$$

$$\hat{U}_{im} = \frac{1}{T_{im}}\sum_{t=1}^{T_i}\Phi(w_{im} \mid O_t).(O_t - \hat{\mu}_{im})(O_t - \hat{\mu}_{im})' \quad \ldots\ldots\ldots.(60)$$

$$\hat{b}_{im}(O_t) = \sum_{m=1}^{M}\hat{c}_{im}\aleph(O;\hat{\mu}_{im},\hat{U}_{im}), \quad 1 \le i \le N \quad \ldots\ldots\ldots\ldots.(61)$$

$$\hat{b}_i(O_t) = \sum_{m=1}^{M}\hat{c}_{im}\hat{b}_{im}(O_t) \quad \ldots\ldots\ldots\ldots.(62)$$

8 – Compute the next estimate of using the formula:

$$\hat{\Phi}(w_{im} \mid O_t) = \frac{\hat{c}_{im}\,\hat{b}_{im}(O_t)}{\sum_{n=1}^{M} \hat{c}_{in}\,\hat{b}_{in}(O_t)} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(63)$$

9 – IF $\quad |\,\Phi(w_{im} \mid O_t) - \hat{\Phi}(w_{im} \mid O_t)\,| \le \varepsilon$ THEN END

ELSE   Make the new value of $\Phi(w_{im}|O_t)$ equal the newly predicted one.

$$\Phi(w_{im} \mid O_t) = \hat{\Phi}(w_{im} \mid O_t) \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(64)$$

GO TO STEP 7.

where ε is a very small threshold value.

9. Implementation Factors

There are several factors that may have in one way or another effects on the implemented model. We are going to describe the more important factors and how to reduce their effects.

9-1. Scaling Factor:

The scaling factor is a major issue in implementing the HMM because of the underflow that may easily occur when calculating the probability function P(O/λ). This is due to the long sequence of multiplications of less than one values probability functions. For instance in using the forward procedure to calculate $\alpha_i(t)$ in (18) we can see easily how many multiplication of probability functions we have to make to calculate any spoken entities.

The straight forward technique of scaling is started by defining the scaling coefficient *c(t)*[2]:

$$c(t) = \frac{1}{\sum_{i=1}^{N} \alpha_t(i)} \qquad \ldots\ldots\ldots\ldots\ldots(65)$$

Now let us compute $\alpha_i(t)$ from (18) and then multiply it by *c(t).* This will lead to :

$$\sum_{i=1}^{N} c(t)\alpha_t(i) = 1 \quad , \quad 1 \le t \le T \qquad \text{.......... ....}(66)$$

Same thing can be done with *$\beta_t(i)$* to form the product *$c(t)\ \beta_t(i)$*. The re-estimation formula of (46) can be rewritten again to include the scaling to become:

$$\hat{a}(i,j) = \frac{\sum_{t=1}^{T-1} C_t \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) D_{t+1}}{\sum_{t=1}^{T-1}\sum_{r=1}^{N} C_t \alpha_t(i) a_{ir} b_r(O_{t+1}) \beta_{t+1}(r) D_{t+1}} \qquad \text{........................}(67)$$

where

$$C_t = \prod_{\tau=1}^{t} c_\tau \qquad \text{.........................}(68)$$

and

$$D_t = \prod_{\tau=t}^{T} c_\tau \qquad \text{.........................}(68a)$$

The numerator and the denominator of (67) consist of the product $C_t D_{t+1} = \prod_{\tau=1}^{T} c_\tau$, which can be factored out and retain the original equation of (46). This scaling technique can also be applied successfully to (47).

The scaling coefficients can be used to find *log P*(O/λ) by the following method:

Consider that we have c_t for t=1,2,3,.......,T and we obtained C_T from (68), then from (65) we will get:

$$\boldsymbol{C_T} = \prod_{t=1}^{T} \boldsymbol{c_t} = \left[\sum_{i=1}^{N} \alpha_T(\boldsymbol{i})\right]^{-1} \qquad \text{......................}(69)$$

Using (19) we will have:

$$\boldsymbol{C_T} = \prod_{t=1}^{T} \boldsymbol{c_t} = \frac{1}{\boldsymbol{P}(\boldsymbol{O} \mid \lambda)} \qquad \text{......................}(70)$$

Take the log of the last two terms:

$$\log\left(\prod_{t=1}^{T} \boldsymbol{c_t}\right) = -\log[\boldsymbol{P}(\boldsymbol{O} \mid \lambda] \qquad \text{......................}(71)$$

By using log properties we can obtain:

$$\log[\boldsymbol{P}(\boldsymbol{O} \mid \lambda] = -\sum_{t=1}^{T} \log(\boldsymbol{c_t}) \qquad \text{......................}(72)$$

Equation (72) shows that log*(P(O|λ)* can be computed but not *P(O|λ* as the later will be out of the dynamic range of the computer.

Viterbi Algorithm also shows itself her again to be successful technique in calculating log*(P(O|λ)* even without bothering about scaling problem.

To follow Viterbi Algorithm let us assume that:

$$\phi_t(i) = \log[\pi_i b_i(O_1)] \qquad \text{......................}(73)$$

Take the log of both sides of (29) and use (72) to get:

$$\phi_t(i) = \max_{1 \le i \le N}[\phi_{t-1}(i) + \log(a_{ij})] + \log[b_j(O_t)] \qquad \text{....................}(74)$$

Now $\log[P(O \mid \lambda)] = \max_{1 \le i \le N}[\phi_T(i)]$.

9-2. Multiple Observation Sequence factor [2]

The main disadvantage of Left-Right topology of HMM is that the observations can not be concatenated into one string and submitted to the model for training. This is due to the one direction left-right move and once a state left we can not go back to it. Accordingly the model will stuck in the last state after passing the first observation sequence and no modelling possible for the other sequences.

The model has to be modified to accept multiple sequence submission to allow the model to be trained by many versions of the same spoken entity.

Let us define the set of observations of the k multiples of observations of a spoken entity by:

$$O = [O^1, O^2, O^3, \ldots\ldots\ldots\ldots O^k]$$
$$where$$
$$O^k = [O_1^k, O_2^k, O_3^k, \ldots\ldots\ldots\ldots O_{T_k}^k]$$

The goal of HMM is to maximise P(O/λ) by adjusting the parameters of λ. In multiple observations P(O/λ) is defined by:

$$P(O \mid \lambda) = \prod_{k=1}^{K} P(O^k \mid \lambda) \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(75)$$

in more abstract way

$$P(O \mid \lambda) = \prod_{k=1}^{K} P_k \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(75a)$$

The multiple observation sequence implication can be done by normalising the numerators and denominators (46) and (47) by P_k to get:

$$\hat{a}_{ij} = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) a_{ij} b_j(O_{t+1}^k) \beta_{t+1}^k(j)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) \beta_t^k(i)} \qquad \ldots\ldots\ldots\ldots\ldots\ldots(76)$$

$$\hat{b}_j(\eta) = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{\substack{t=1 \\ O_t = w_\eta}}^{T_k - 1} \alpha_t^k(i) \beta_t^k(i)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k - 1} \alpha_t^k(i) \beta_t^k(i)} \qquad \ldots\ldots\ldots\ldots\ldots\ldots(77)$$

The same procedure of normalisation could be used in the case of continuous density distribution to find the parameters of the model.

9-3. Initial Model Parameters Estimate factor:

When we initially try to build an HMM model we normally have nothing but streams of observations. If we are fortunate then we have parameters from old models, which is not normally the case. To put the initial model parameters, we have to be careful as one might easily slip into divergence with bad model initialisation. The problem with discrete observations HMM is less effective as we can initialise the model parameters with random values, but taking into consideration the constraints in (9),(10), and (11). In continuous density HMM (CHMM) the problem is more serious and the parameters should be judiciously selected to get rid of the divergence fate. Let us take the problem in Left-Right HMM Topology and suggest a safe way to follow.

The parameters that constitute any model λ are π, A, and B. For π it is straight forward and known to be always $\pi = [1\ 0\ 0\ 0\ 0 \ldots\ldots.. 0]$, of course this is with Left-Right topology models. For the states' transition parameters A=[a_{ij}] the choice is also flexible and if we have the topology of Fig(2) then A will be the following matrix for seven states model:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & 0 & 0 & 0 \\ 0 & a_{22} & a_{23} & a_{24} & 0 & 0 & 0 \\ 0 & 0 & a_{33} & a_{34} & a_{35} & 0 & 0 \\ 0 & 0 & 0 & a_{44} & a_{45} & a_{46} & 0 \\ 0 & 0 & 0 & 0 & a_{55} & a_{56} & a_{57} \\ 0 & 0 & 0 & 0 & 0 & a_{66} & a_{67} \\ 0 & 0 & 0 & 0 & 0 & 0 & a_{77} \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots(78)$$

The values of a_{ij} can be selected as :

a_{ii} = 0.94 , $a_{i,i+1}$ = 0.04, and $a_{i,i+2}$ = 0.02 for $1 \le i \le 5$

a_{ii} = 0.97 , $a_{i,i+1}$ = 0.03 for $i = 6$

a_{ii} = 1 for $i = 7$

These values deduced from the fact that the observations tends to stay in their current state and have less tendency to move to the next state and more less tendency to jump the next state. After optimisation we can see that the observations wanted to stay in their

current state is true, this imply that $a_{ii} > a_{i,i+1}$ and $a_{ii} > a_{i,i+2}$. However, the observations might prefer to stay in the next state or jump it, i.e. $a_{i,i+1} > a_{i,i+2}$ or $a_{i,i+1} < a_{i,i+2}$.
A more precise way is by using initial uniform segmentation of each utterance into the proposed number of states and apply the following algorithm[11]:

The suitable a_{ij} (in programming we use the notation a(i,j)) initialisation is found to be dependant on states' duration. In this case the average duration D of each state is estimated by :

$$D = \frac{1}{NxK} \sum_{n=1}^{K} | x_n | \text{ ....(79)}$$

where K is the number of versions of an utterances in the training set.
N is the number of states
$|X_n|$ is the length of the n_th utterance in the training observations.

Then, the transition matrix elements a(i,j) are estimated by:

$$\begin{array}{ll}
For & 1 \le i \le N-2 \\
a(i,i) = \frac{D-1}{D} & \text{..........................}(80) \\
a(i,i+1) = .8 \times (1 - a(i,i)) & \text{.........................}(80a) \\
a(i,i+2) = 1 - a(i,i) - a(i,i+1) & \text{.........................}(80b) \\
For & i = N-1 \\
a(i,i) = \frac{D-1}{D} & \text{.........................}(80c) \\
a(i,i+1) = 1 - a(i,i) & \text{.........................}(80d) \\
For & i = N \\
a(i,i) = 1 & \text{.........................}(80e)
\end{array}$$

The formula is inferred from the fact that if there are K elements of duration D in each state then there will be only one transition to the next state.
What is left now is the most problematic parameters $B = \{b_i(O_t)\}$, they have to be very well initialised. In our case we suggest the following steps:

1 - Uniformly segment the utterances of each spoken entity by N states.

2 - Take the mean and the covariance of each segment.

3 – Consider the observations follow Gaussian density probability distribution.

After the previous suggestions for initialising the model parameters we can apply Viterbi Algorithm to extract the optimum parameters.

The technique described in this section is for unimodal (single mixture) distribution. To extend it to multimodal (multimixture) distribution the followings are suggested:

1 – Apply the same procedures for π, A, B used in unimodal distribution.

2 – After uncovering the real state sequence from Viterbi Algorithm, aggregate the observations, of all the versions of the spoken entity, belonging to each state in separate cells.

3 – Use Vector quantization technique to cluster each cell into several mixtures.

4 – Optimise the clustering by any known statistical technique; such as maximum likelihood and expectation maximisation.

5 – Find the mean and the covariance of each cluster (mixture)

The model is now complete.

9-4. Number of States Factor

One thing left which has to be decided from the initial instant of designing the model. It is the optimum number of states needed to model the problem. There is no straight forward answer to this requirement. The number of states is decided empirically depending on the nature of the problem. Some times previous experience about the problem is necessary or one has to suggest different number of states then select the one who gives the best results. Also, if we could define the physical meaning of the states we can limit the number of states. In isolated words recogniser the number of states are suggested to be between 4 and 12. This is justified by assuming that the states are representing the phonemes or the phones of the utterances. In phonemes modelling the number of states are mostly assumed to be 3, as the phonemes could be segmented into initial, stable, and final states.

9-5. State Duration Incorporation:

The basic HMM does not take into consideration the state duration factor in its modelling procedure. This is considered as major weakness in the model since the duration carry important information about the temporal structure of the speech signal. Our duty now is to find some useful way to include the duration within the conventional model. Let us first ask this question:

What is the probability of being in a state for τ instants?

The answer resides in finding the probability density function $p_i(\tau)$ which has the definition of :

$$p_i(\tau) = P(q_1=S_i, q_2=S_i, q_3=S_i, \ldots\ldots\ldots\ldots, q_\tau=S_i, q_{\tau+1}=S_j, \ldots\ldots) \quad \ldots\ldots\ldots\ldots(81)$$

$$=\pi_i\,(a_{ii})^{\tau-1}(1-a_{ii}) \quad \ldots\ldots\ldots\ldots(81a)$$

Now we can calculate the expected duration in state i by the following equation:

$$\bar{\tau}_i = \sum_{\tau=1}^{\infty} \tau p_i(\tau) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(82)$$

Using (81a) and considering $\pi_i = 1$ we get :

$$\bar{\tau}_i = \sum_{\tau=1}^{\infty} \tau(a_{ii})^{\tau-1}(1-a_{ii})$$

$$=(1-a_{ii})\sum_{\tau=1}^{\infty} \tau(a_{ii})^{\tau-1}$$

$$=(1-a_{ii})\frac{\partial}{\partial a_{ii}}\sum_{\tau=1}^{\infty}(a_{ii})^{\tau}$$

$$=(1-a_{ii})\frac{\partial}{\partial a_{ii}}\left(\frac{a_{ii}}{1-a_{ii}}\right)$$

$$=\frac{1}{1-a_{ii}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(83)$$

Now, if we return to our first example about weather forecast and ask the question:

What is the average consecutive sunny, cloudy, and rainy days?

The answer is by applying (83) using the values of a_{ii} from Table-1 to get :

$$Sunny\ days = \frac{1}{1-a_{11}} = \frac{1}{1-0.7} \approx 3$$
$$Cloudy\ days = \frac{1}{1-a_{22}} = \frac{1}{1-0.8} \approx 5$$
$$Rainy\ days = \frac{1}{1-a_{33}} = \frac{1}{1-0.6} \approx 3$$

Unfortunately this duration distribution is meaningless when we try to apply it to speech recognition problems. Therefore, another way to incorporate the duration has to be considered. One option is to include the state duration in the model formulas, this needs reformulating the whole model parameters[4]. The model works perfectly in this case but the problem now is with the vast increase in computational cost that makes the use of this new model impractical.

The other option is to use heuristic technique to include the duration to obtain comparable performance as the correct theoretical duration inclusion with very low computational and storage costs. The state duration probability function $p_j(\tau)$ is estimated during the model training case and may be defined as:

$p_j(\tau)$: is the probability of being in state j for τ duration.

The duration probability density function is considered to be Gaussian with 3 to 5 mixtures.

During recognition the state duration are calculated from the backtracking procedure in Viterbi Algorithm. Then, the log likelihood value is incremented by the log of the duration probability value as below:

$$\log[\hat{P}(q,O \mid \lambda)] = \log[P(q,O \mid \lambda)] + \eta \sum_{j=1}^{N} \log[p_j(\tau_j)] \qquad \text{......................}(84)$$

where η is a scaling factor.

τ_j is the normalised duration of being in state j, as detected by Viterbi Algorithm.

9-6. Data Representation Factor:

The training and testing speech data are taken from the:

http://Kel.otago.ac.nz/hyspeech/corpus

The initial sets of data are digits 0-9 spoken by 21 speakers (11 males and 10 females) and each digit is spoken three times by each speaker. Among those words 42 uttered digits used for training and 15 for testing. The speech data in Otago Speech Corpus are sampled at 22050 Hz with short silence before and after each utterance.

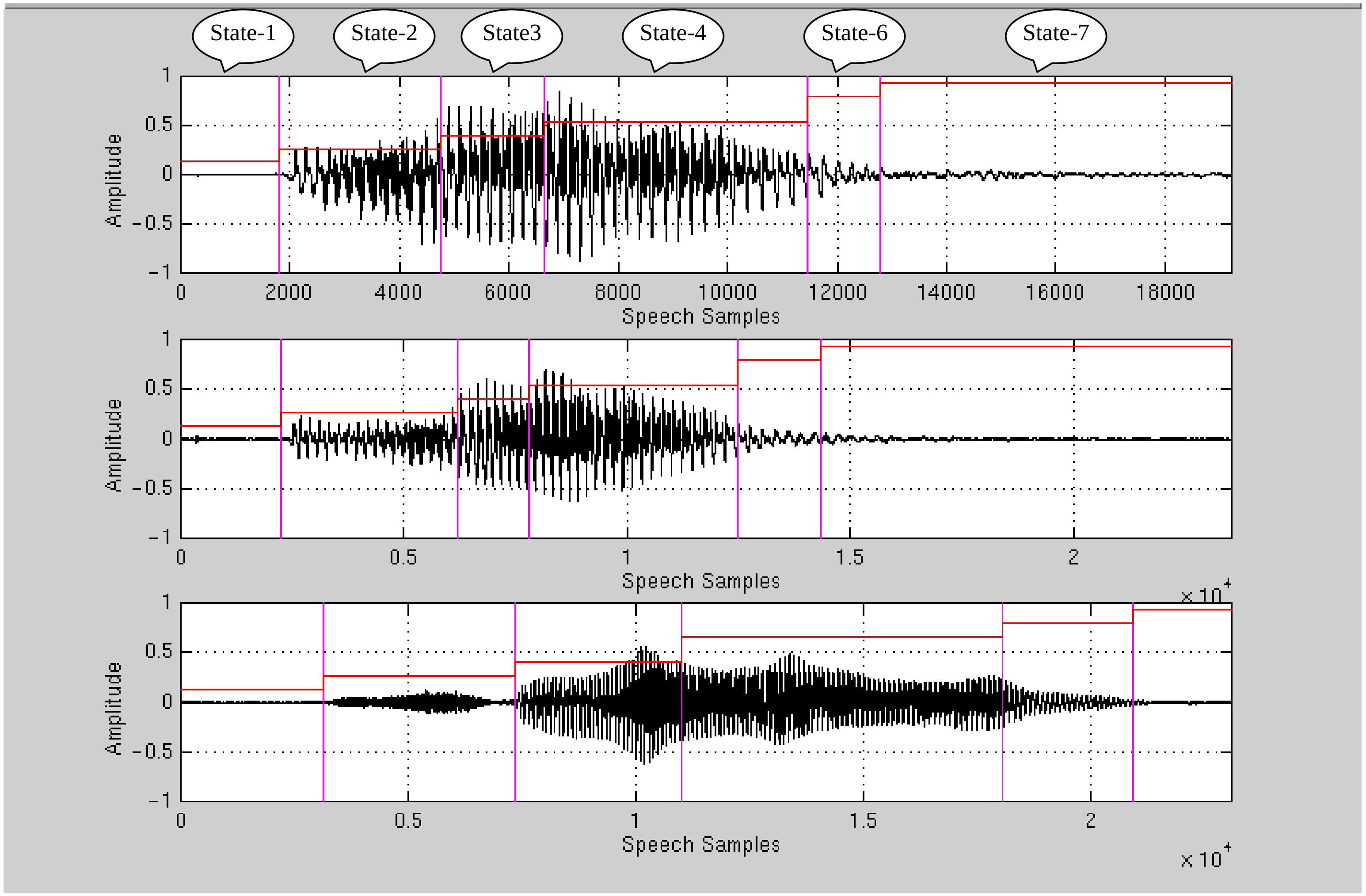
The next step is to transform the time signal into Mel scale coefficients. The number of coefficient are selected to be 26 (12 mels and 12 delta mels with one power and its delta). Also experiments have been done on 13 coefficients without considering the dynamic behaviour of the signal. The Mel scale coefficients as extracted features are selected because they imitate into some extent the feature selection in human ears. The Mel scale method considers the spectrum are linearly distributed below 1000Hz and logarithmically above that. This makes the filter banks moving on linear centres below 1000Hz (i.e. 100, 200, 300, ..., 1000) and on logarithmic centres over that (i.e. 1149, 1320,1516,....). The very good characteristics of the Mel scale coefficients is that they allow the use of Euclidean distance measure in finding the distance between two examples. This is greatly reduces the computational cost of procedures that depend on distance measure like those in VQ.

10. Results and Conclusions

In this section we are going to show some experimental results and discuss some useful conclusions.

10-1. The first experiment dealt with the segmentation of spoken words into states. Fig.(8) shows different versions of the spoken word zero by three different speakers. We can see clearly how the time signal varied even for the same word. The states are found by Viterbi Algorithm and assigned clearly to their corresponding segments. Also we can see that the observations are not always passing through all the states that the model has been designed on. In this case state 5 was jumped by digit zero observations when they were submitted to digit zero model.

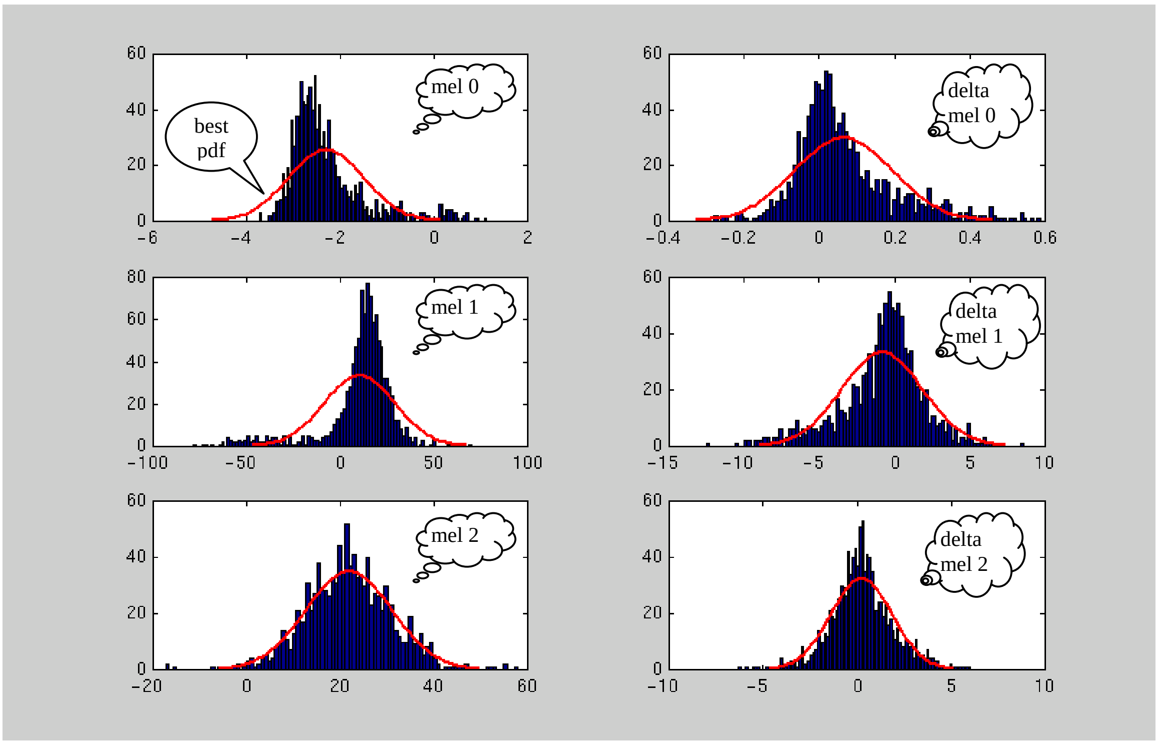
Fig.(8) States' Assignment of Digit ZERO Presented to the ZERO State Model.
The analysis uses 13 Mel scales coefficients without taking the dynamic coefficients into consideration. There are 6 states detected in all the three versions of the spoken digit ZERO i.e 1 , 2 , 3 , 4 , 6 , and 7.



10-2. The second experiment dealt with Mel scales coefficients distribution.

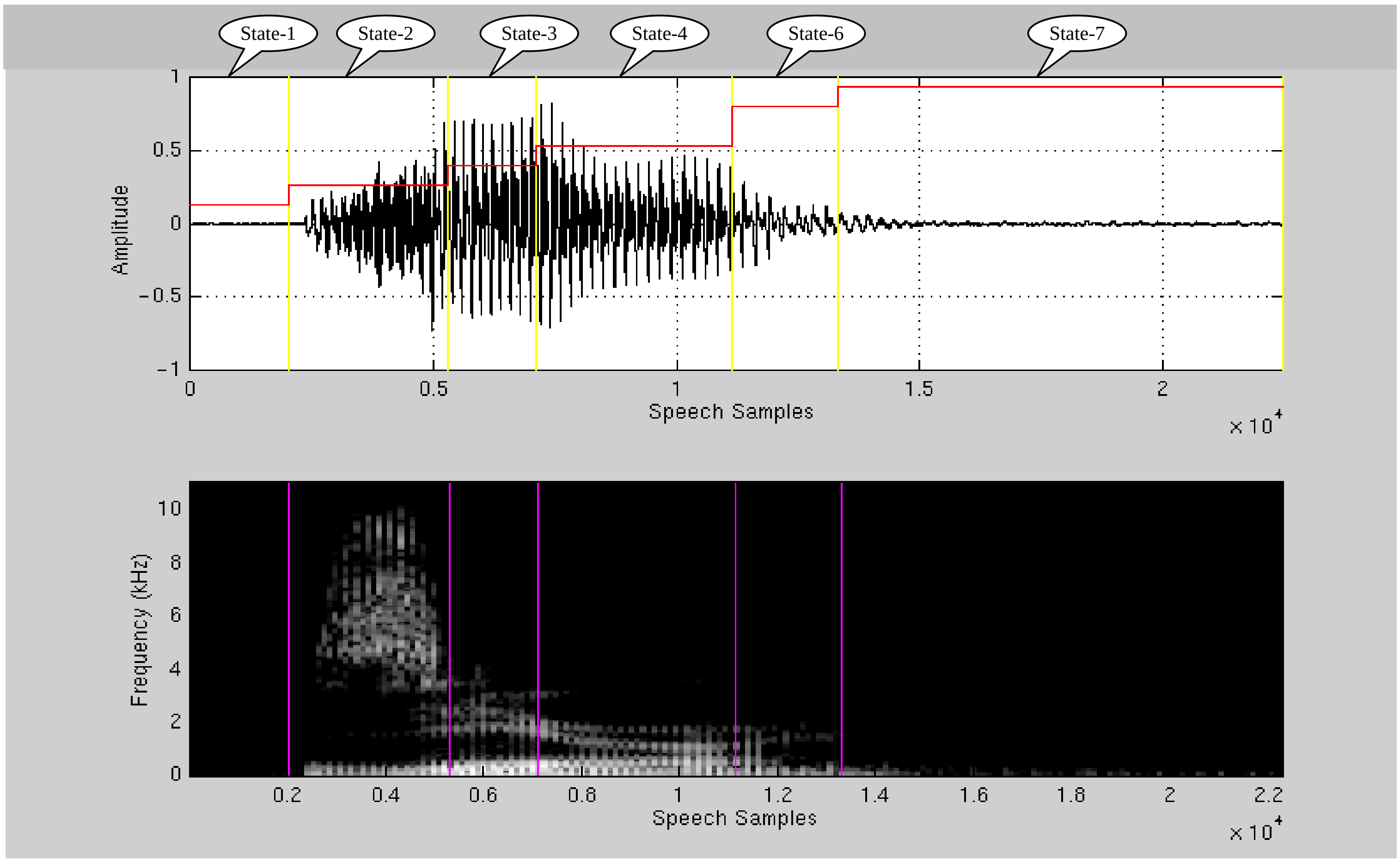
Fig.(9) Shows some distributions of Mel scale coefficients and their deltas of state one in spoken digit zero. The power of the signal and its delta are represented by mel 0 and delta mel 0. The Mel coefficients capture the stable signal characteristics, while the deltas capture the dynamic characteristics. Also, we can see from this figure the best fit probability distribution function (pdf) for each coefficient. It is clear that some coefficients like mel 0 , mel 1 and their deltas are far from being represented by single pdf. This consolidates the need of multimodal (multimixture) representation of the coefficients. In our model we approximate the Mel scale coefficients distribution by 5 to 9 mixtures.

Fig.(9) Mel Scale Coefficients Distribution
The histogram and the best fit normal pdf of mel0,mel1, and mel2 with their deltas.



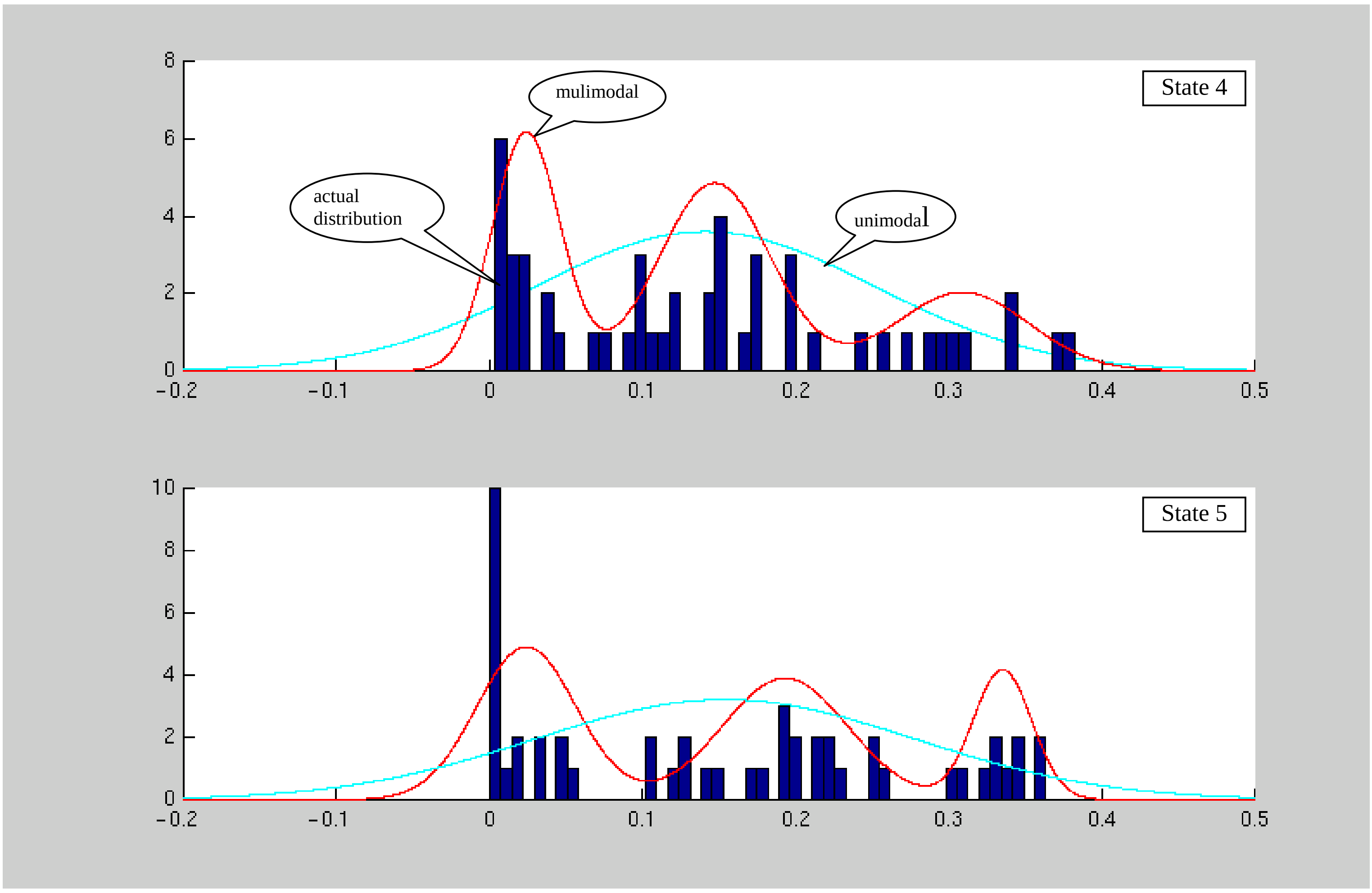
10-3. The third experiment was carried out to show the correspondence between the speech signal, states, and the spectra. This relation gave us more understanding and confidence on the behaviour of the observation vectors within each state. Fig.(10) shows this clearly and we can notice the difference in spectral behaviour of different states.

Fig.(10) Shows the correspondence between the time signal samples, states, and spectrum of spoken digit zero.



10-4. The fourth experiment was about the representation of the state duration distribution using multimodal (multimixture) probability distribution. Fig.(11) shows the normalised duration probability distribution for unimodal and multimodal (with 3 mixtures) representation. The multimodal pdf shows superiority in representing the distribution.

Fig.(11) Multimodal Representation of States' Duration.
The unimodal is poorly representing the states duration, while the multimodal is smoothly follow up the duration distribution even with only three mixtures used.



11. References :

[1] J.D. Ferguson, “Hidden Markov Analysis: An Introduction“, in Hidden Markov Models for Speech, Institute of Defence Analyses, Princeton, NJ, 1980.

[2] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, “An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition”, The Bell System Technical Journal, vol. 62, no.4, pp.1035-1073, Apr. 1983.

[3] L. R. Rabiner, B. H. Juang, “An Introduction to Hidden Markov Models”, IEEE ASSP Magazine, pp. 4 – 16, Jan. 1986.

[4] L. R. Rabiner, “ A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”, vol. 77, no. 2, pp. 257-286, 1989

[5] L. R. Rabiner, B. H. Juang, “Fundamentals of Speech Recognition”, Prentice Hall, Englewood Cliffs, New Jersey, 1993.

[6] L.E. Baum and T. Petrie, “ Statistical Inference for Probabilistic Functions of Finite State Markov Chains”, Ann. Math. Stat., vol. 37 ,pp 1554-1563, 1966.

[7] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, “ A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains”, Ann. Math. Stat., vol. 41, no.1, pp 164-171, 1970.

[8] L. E. Baum, “An Inequality and Associated Miximization Technique in Statistical Estimation for Probabilistic Functions of Markov Processes”, Proc. Symp. On Inequalities, vol. 3, pp 1-7, Academic Press, New York and London 1972.

[9] J. K. Baker, “ The Dragon System – An Overview”, IEEE Trans. Acoustic, Speech, and Signal Processing, vol. ASSP-23, no. 1, pp. 24-29, Feb. 1975.

[10] F. J. Owens , “ Signal Processing of Speech “, Macmillan Press Ltd., London, 1993.

[11] E. Harborg, “Hidden Markov Models Applied to Automatic Speech Recognition”, PhD Thesis, Norwegian Institute of Technology, Trondheim, Aug. 1990.

[12] L. E. Baum and J. A. Egon, “An inquality with applications to statistical Estimation for Probabilistic Functions of Markov Process and to a Model for Ecology”, Bull. Amer. Meteorol. Soc., vol.73, pp.360-363, 1967.

[13] G. D. Forney, “The Viterbi Algorithm”, Proc. IEEE, vol. 61, pp.268-278, Mar, 1973.

[14] Y. Linde, , A. Buzo, R. M. Gray “An ALgorithm for Vector Quantizer Design”,IEEE Trans. on Comm. vol. COM-28, no.1, pp. 84-95, 1980.

[15] R. M. Gray , “ Vector Quantization”, IEEE ASSP Magazine, vol. 1, no. 2, pp. 4 – 29 , 1984.

[16] F. Jelinek ,”Statistical Methods for Speech Recognition”, The MIT Press, 1998.