OPTIMAL CONTROL FOR TRAINING: THE MISSING LINK BETWEEN HIDDEN MARKOV MODELS AND CONNECTIONIST NETWORKS

Athanasios Kehagias

Division of Applied Mathematics
Brown University
Providence, RI 02912

June 12, 1989

Abstract

For every Hidden Markov Model there is a set of "forward" probabilities that need to be computed for both the recognition and the training problem. These probabilities are computed recursively and hence the computation can be performed by a multistage, feedforward network that we will call Hidden Markov Model Net (HMMN). This network has exactly the same architecture as the standard Connectionist Network (CN). Furthermore training an Hidden Markov Model is equivalent to optimizing a function of the HMMN; training a CN is equivalent to optimizing a function of the CN. Due to the multistage architecture, both problems can be seen as Optimal Control problems. By applying standard Optimal Control techniques we discover in both problems that certain backpropagated quantities (backward probabilities for HMMN, backward propagated errors for CN) are of crucial importance to the solution. So HMM's and CN's are similar in architecture and training.

1 Introduction

As Hidden Markov Models (HMM) and Connectionist Networks (CN) are both used for Speech Recognition, they have been often compared, usually in terms of relative performance. Recently there has been some discussion of their similarities [1,2]. In [1] similarities of HMM's and CN's are considered explicitly, but the emphasis is on architecture independent similarities. [1]

The position of the present paper is that HMM's and CN's are indeed very similar, in particular with respect to architecture and training methods. It has been observed by leCun [7] that training of CN's can be considered as an Optimal Control problem. As far as the author knows nobody has yet noticed that the same is true of HMM's; this is the subject of Section 2 in this paper. Furthermore, choice of the optimal control is achieved by the Backward-Forward (BF) algorithm, a steepest ascent method that depends crucially on back propagated quantities. This is exactly analogous to the training of a connectionist network by backpropagation, as pointed out by leCun. It will become clear that these similarities between HMM training and CN training are due to two factors: they both depend on a multistage, feedforward architecture, and their training is a constrained optimization problem. There are no new computational methods proposed in this paper; an effort is made to point out some striking similarities between the two fields and explain these similarities by using the point of

[1] At the time the paper was being completed the author became aware of the work by Kung and Hwang [8,9,10] where similar ideas are presented.

view of Control Theory.

Let us define some terms: By CN is meant the usual network with an input layer (layer 1), an output layer (layer T) and $T-2$ hidden layers. We assume, to make the argument simple, that every layer contains the same number, call it N, of units (but we can easily generalize to the case of varying number of units from layer to layer). The output of unit i at layer t is denoted by $\alpha_t(i)$ and we have

$$\alpha_{t+1}(i) = f_i(\sum_{j=1}^{N} w_{ij}(t)\alpha_t(j)). \tag{1.1}$$

where f_i is a nonlinear function (usually the sigmoid). These are the standard networks defined, e.g. in [9]. The Back Propagation (BP) method is the usual variant with simple descent in the gradient direction.

The HMM's referred to here are the variety described in [8]. Strictly speaking, an HMM is pair of stochastic processes. However, we will be more interested in a recursive computation that is associated with every HMM. As will be explained in Section 2, there are certain probabilities that we want to compute, call them $\alpha_t(i)$, $i = 1, 2, ..., N$, $t = 1, 2, ..., T$. These probabilities obey the following equations:

$$\alpha_{t+1}(i) = \sum_{j=1}^{N} f_{ij}(t)\alpha_t(j), \tag{1.2}$$

It is clear that the computation of (1.2) can be performed by a multistage feedforward network and that (1.2) is very similar to (1.1). We will call this the Hidden Markov Model Network (HMMN). More details are given in Section 2, but it is already clear that (1.1) and (1.2) are very similar.

The rest of the paper explores this similarity further. In Section 2 we will formulate training of HMM as an Optimal Control problem. In Section 3 we will cite leCun's results on the Optimal Control formulation of the CN training problem. In Section 4 we will compare HMM's and CN's in the light of what was learned in Sections 2 and 3.

2 HMM's, the Backward-Forward algorithm and Optimal Control

In this section we review briefly HMM's and their training and point out the connection with optimal control: training of the HMM involves optimal control of an associated network.

The following review of HMM's follows [8]. A HMM is is a pair of stochastic processes: X_t, $t = 1, 2, ..., T$, O_t, $t = 1, 2, ..., T$. X_t (the hidden process) is a Markov process: it takes values in the set $\mathbf{x} = \{x_1, ..., x_N\}$ with certain probabilities:

$$Prob(X_1 = x_i) = c_i, \quad i = 1, 2, ..., N, \tag{2.1}$$

$$Prob(X_{t+1} = x_j \mid X_t = x_i) = a_{ij}, \quad i, j = 1, 2, ..., N, \quad t = 1, 2, ..., T. \tag{2.2}$$

Note that (2.1) and (2.2) are sufficient to compute $Prob(X_t = x_j)$. O_t is the observed process and takes values in $\mathbf{o} = \{o_1, o_2, ..., o_M\}$ in a way that depends on X_t:

$$Prob(O_t = o_k \mid X_t = x_i) = b_{ik} \tag{2.3}$$

We collect the a's, b's and c's in appropriately dimensioned matrices A, B, C and define the triple $\mathbf{M} \doteq (A, B, C)$. $\mathbf{M}$ defines a Hidden Markov Model.

Now, suppose we observe $O_1 = o_{i1}$, $O_2 = o_{i2}$, ..., $O_T = o_{iT}$ (for some integers $i1, i2, ..., iT$). The following probabilities (dependent on the observations) are very useful for the recognition and training problems:

$$\alpha_t(i) \doteq Prob(O_1 = o_{i1}, O_2 = o_{i2}, ..., O_t = o_{it}, X_t = x_i) \tag{2.4}$$

$$\alpha_t{}' = [\alpha_t(i) ... \alpha_t(N)] \tag{2.5}$$

($\alpha_t{}'$ is the transpose of α_t). We will call these quantities a-probabilities or forward probabilities. They are of interest, because

$$Prob(O_1 = o_{i1}, O_2 = o_{i2}, ..., O_T = o_{iT}) = \sum_{i=1}^{N} \alpha_T(i). \tag{2.6}$$

So, for a given observation sequence $O_1 = o_{i1}, ..., O_T = o_{iT}$, $J_0(\mathbf{M}) \doteq \sum_{i=1}^{N} \alpha_T(i)$ is the probability of this particular sequence occurring, having assumed a particular model $\mathbf{M}$ to be true. In recognition, we assume a fixed model and try to find the sequence of highest probability . In training, we fix a sequence $o_{i1}, ..., o_{iT}$ and look for the model $\mathbf{M}$ that maximizes the probability $J_0(\mathbf{M})$. In essence, training is Maximum Likelihood estimation of $\mathbf{M}$.

Now we will formulate the training problem as an Optimal Control Problem. First of all, it is easy to see that the computation of the α-probabilities proceeds in a recursive way. It is proven in [8] that

$$\alpha_1(i) = g_i(O_1)c_i \tag{2.7}$$

$$\alpha_{t+1}(i) = \sum_{j=1}^{N} a_{ji} g_i(O_{t+1}) \alpha_t(j), \quad t = 1, ..., T-1, \tag{2.8}$$

Here $g_i : \mathbf{o} \rightarrow [0,1]$ is a **nonlinear** (decision) function that is defined as follows: $g_i(v) = b_{ik}$ iff $v = o_k$. That is, g_i takes as input an observation and gives as an output the probability that this observation occurred while $X_t = x_i$. We can use matrix notation, construct appropriate matrices $\Phi(t, \mathbf{M})$ and write (2.6) (for a fixed observation sequence), as

$$\alpha_{t+1} = \Phi(t, \mathbf{M}) \alpha_t, \quad t = 1, 2, ..., T-1 \tag{2.9}$$

This is the recursive relation for the α's. Even though t is a time index, we could implement this computation on a T-layers network, so that t could be interpreted as a layer index. Either way of looking at t is equivalent. If we choose the layers interpretation , we have here feedforward layers and there is no skipping between layers. This is very similar to the standard variety of connectionist nets. In fact, we will explicitly consider a net, which we will call the Hidden Markov Model Net (HMMN), which computes (2.7), (2.8) (see Fig.1). This net consists of T layers; the last one is called the output layer. Each layer has N units. The i-th unit of the t-th layer is $\alpha_t(i)$. The i-th unit of the first layer has output according to eq. (2.7) . The i-th unit of next layers has output according to equation (2.8). The input to the net is the observations O_t. Not all input is applied at the first layer: The t-th layer receives input O_t which passes through the **nonlinear** function g_i and multiplies the linear combination of the previous layer output. The similarities with more traditional connectionist networks are obvious; we will comment on them later.

Alternatively, if we choose the time interpretation, we have a Discrete Time Dynamical System, which can be controlled by choice of $\mathbf{M} = (A, B, C)$. The object of the control is to maximize

$$J_0(\mathbf{M}) = \sum_{i=1}^{N} \alpha_T(i). \tag{2.10}$$

We proceed to apply standard methods of Optimal Control. Optimal Control is a theory that deals with the optimization of functions of time evolving systems. See [10] for an introduction to the subject. This is a constrained optimization problem. There are two types of constraints:

1. The controls $\mathbf{M} = (A, B, C)$ are not free quantities. Since their elements are probabilities, we must have the following conditions:

$$\sum_j a_{ij} = 1, \sum_j b_{ij} = 1, \sum_j c_j = 1. \tag{2.11}$$

$$a_{ij} \geq 0, b_{ij} \geq 0, c_j \geq 0. \tag{2.12}$$

We can remove this kind of constraint by making the a's, b's, c's to depend on auxiliary variables q's, r's, s's in the following way:

$$a_{ij} = \frac{q_{ij}^2}{\sum_j q_{ij}^2}$$

$$b_{ij} = \frac{r_{ij}^2}{\sum_j r_{ij}^2}$$

$$c_j = \frac{s_j^2}{\sum_j s_j^2}$$

Collect the q's, r's, s's in appropriately dimensioned matrices Q, R, S. It is easy to prove that any set of Q, R, S values gives admissible A, B, C and, conversely, any admissible value of A, B, C can be produced by appropriate values of Q, R, S. Hence we can also write $\mathbf{M} = (Q, R, S)$. This removes the constraints on the control.

2. The second kind of constraints is that the equations (2.7), (2.8) must be satisfied. Because of the multistage, feedforward nature, Optimal Control concepts prove useful.

Let us put all the factors influencing α_{t+1} in one matrix F. In fact this $F(t) = \Phi(t, \mathbf{M})$ with a fixed $\mathbf{M}$, and eq.(2.13) is the same as eq.(2.9). This matrix depends on O_t, so we can write $F(O_t)$. But since $O_t = o_{it}$ is given, we will write simply $F(t)$. There is also a dependence on Q, R, S, which we will not explicitly denote. The problem is to choose the elements of $F(t)$ under the appropriate constraints; We will have

$$\alpha_{t+1} = F(t)\alpha_t \qquad t = 1, 2, ..., T-1 \tag{2.13}$$

$$\alpha_1 = F(0) \tag{2.14}$$

We will treat these constraints explicitly, by use of the standard technique of Lagrange multipliers . We augment the cost function J_0 with the equations (2.6). Consider the Lagrange multipliers β_t, $t = 1, ..., T$ (see [7] for details):

$$J = J_0 - \sum_{t=1}^{T-1} \beta_{t+1}{}'(\alpha_{t+1} - F(t)\alpha_t) - \beta_1{}'(\alpha_1 - F(0)) \tag{2.15}$$

The introduction of Langrange multipliers allows us to proceed as if the problem is unconstrained At a maximizing point the following conditions must be necessarily satisfied:

$$\frac{\partial J}{\partial \alpha_t} = 0 \tag{2.16}$$

$$\frac{\partial J}{\partial Q} = 0 \tag{2.17a}$$

$$\frac{\partial J}{\partial R} = 0 \tag{2.17b}$$

$$\frac{\partial J}{\partial S} = 0 \tag{2.17c}$$

Let us first look at (2.16). Define

$$H_t \doteq \beta_{t+1}{}'F(t)\alpha_t, \ \ t = 1, ..., T-1$$

This is the Hamiltonian function of the problem. Now (2.16) becomes

$$\beta_t = \frac{\partial H_t}{\partial \alpha_t} \Rightarrow \tag{2.18}$$

$$\beta_t = F'(t)\beta_{t+1} \qquad t = 1, ..., T-1 \tag{2.19}$$

We also have

$$\frac{\partial J}{\partial a_T} = 0 \Rightarrow \beta_T = [1\ 1 ... 1]\,' \tag{2.20}$$

Equations (2.19), (2.20) allow us to solve for β_t backwards. That is, they are back propagating quantities. Equation (2.19) can be written

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} g_j(O_{t+1}) \beta_{t+1}(j) \quad t = 1, ..., T-1 \tag{2.21}$$

Now, eq.(2.21) along with the final condition (2.20) is sufficient to prove that $\beta_t(i) = Prob(O_{t+1} = o_{i,t+1}, ..., O_T = o_{iT} \mid X_t = x_i)$. In [5] this is done in the reverse way. Starting from the above definition of β_t , it is proven that eq.(2.21) has to be satisfied. The reader familiar with the Backward-Forward algorithm will recognize here β_t as the backward probabilities, or β-probabilities.

Another very interesting fact is that α and β have a Hamiltonian structure. The equation (2.8) can be written as

$$\alpha_{t+1} = \frac{\partial H_t}{\partial \beta_{t+1}} \qquad t = 1, ..., T-1 \tag{2.22}$$

(2.18) and (2.22) show clearly the Hamiltonian structure: α and β are conjugate quantities (like position and momentum in classical mechanics). This fact, as far as the author knows, has not been noted before in the speech literature, but of course is no accident: it follows from the Optimal Control formulation.

So far we have not said anything about the actual selection of the control variables (whether these are given in terms of A, B, C or Q, R, S). Equations (2.17a-c) are a necessary condition that has to be satisfied, but obtaining Q, R, S from them is not easy. As a matter of fact, there is no canonical way to explicitly solve the general Optimal Control problem. However, in this particular case there is a way to solve the training problem that completely bypasses (2.17) and the auxiliary variables Q, R, S. The solution is furnished by the Backward-Forward algorithm which consists of the following iteration:

Given $\mathbf{M}(l) = (A(l), B(l), C(l))$,

$$a_{ij}^{l+1} = \frac{\sum_{t=1}^{T-1} \alpha_t^l(i) a_{ij}^l g_j^l(O_{t+1}) \beta_{t+1}^l(j)}{\sum_{t=1}^{T-1} \alpha_t^l(i) \beta_t^l(i)}$$

$$b_{ij}^{l+1} = \frac{\sum_{t:O_t=o_j} \alpha_t^l(i) \beta_{t+1}^l(i)}{\sum_{t=1}^{T-1} \alpha_t^l(i) \beta_t^l(i)}$$

$$c_i^{l+1} = \frac{\alpha_i^l(i) \beta_i^l(i)}{\sum_{i=1}^{N} \alpha_T^l(i)}$$

This is an ascent procedure (not necessarily steepest ascent). As proven in [3]

$$J(\mathbf{M}(l+1)) \geq J(\mathbf{M}(l))$$

and in fact we have guaranteed convergence to a local maximum[2].

The solution of the training problem of HMM by the BF algorithm is completely independent of the Optimal

[2]From the analysis in [3] it is clear that the cross-entropy between models is minimized. This is very interesting in view of the fact that cross entropy is often proposed in connectionist models as an alternative criterion to the squared error.

Control formulation. Why then go through the Optimal Control formulation when we finally use a standard algorithm to solve the problem? The Optimal Control perspective clarifies the meaning of the back propagated quantities that are prominent in the Backward-Forward algorithm and the Hamiltonian structure. It is indeed very satisfying to see that the β's afford two very different and yet equally significant interpretations: as backward probabilities or as Lagrange multpliers. Furthermore, the Lagrange multipliers method is a very general method that is used in many fields; especially there exists an extensive corpus of research in Optimal Control and it is not unreasonable to expect to gain valuable insights to the speech recognition problem by relating it to old and well studied control problems.

In conclusion, in this section we have learned the following:

1. For speech recognition it is necessary to compute certain probabilties (α's) . These probabilities are computed by an HMMN.

2. The HMMN has a multistage feedforward architecture. There is a weighted input form previous layers and nonlinear functions appear at each unit.

3. The training of an HMM involves a constrained optimization of the parameters of HMMN. Due to the multistage nature of HMMN this optimization can be formulated as an Optimal Control problem.

4. In solving the Optimal Control problem certain back propagating quantities are of crucial importance. These quantities can be interpreted in two ways: as backward probabilties or as Lagrange multipliers. They are conjugate (in the Hamiltonian sense) to the forward probabilities.

3 CN, the Back Propagation algorithm and Optimal Control

In this section we will basically repeat the procedure of the previous section for CN's instead of HMMN's. The treatment here is a summary of leCun's treatment in [7]. As far as the author knows, that was the first paper where the Optimal Control interpretation of CN training was proposed.

Consider a CN with $T-2$ hidden layers, an input layer and an output layer. All layers have N units each. The i-th unit consists of a sigmoid function f_i; the output $\alpha_t(i)$ of unit i in layer t is given by:

$$\alpha_{t+1}(i) = f_i(\sum_{j=1}^{N} w_{ij}(t)\alpha_t(j)). \tag{3.1}$$

We can define $\alpha_t{}' = [\alpha_t(1)...\alpha_t(N)]$ and then the equation for layer-to-layer forward propagation is

$$\alpha_{t+1} = F(W(t)\alpha_t) \tag{3.2}$$

The actual values of the α's depends on the initial conditions, that is, the input to the first layer; a set of P input patterns $X_1, ..., X_P$ are given and a set of desired output patterns $Y_1, ..., Y_P$. Each pattern is an N-by-1 vector. When the p-th pattern is input in the net we will have

$$\alpha_1 = X_p, \quad p = 1, 2, ..., P.$$

Now a sequence of α's can be computed from (3.2). Call these $\alpha_1^p, \alpha_2^p, .., \alpha_T^p$. The training problem is to find $W(t), t = 1, 2, .., T$ that minimize the cost function.

$$J_0 = \sum_{p=1}^{P}(Y_p - \alpha_T^p)\,'(Y_p - \alpha_T^p)$$

However there are layer-to-layer constraints to be statisfied. We incorporate these to the augmented cost function by use of Lagrange multipliers β_t^p:

$$J=\sum_{p=1}^{P}(Y_p-\alpha_T^p)'(Y_p-\alpha_T^p)-\sum_{p=1}^{P}\sum_{t=1}^{T-1}\beta_{t+1}^p{}'(\alpha_{t+1}^p-F(W(t)\alpha_t^p))-\sum_{p=1}^{P}\beta_1^p{}'(\alpha_1^p-X_p) \tag{3.3}$$

These computations are implemented on the network of Fig. 2.

The treatment of the problem proceeds exactly as in Section 2. Define the Hamiltonian:

$$H_t^p=\beta_{t+1}^p{}'F(W(t)\alpha_t^p) \tag{3.4}$$

Then

$$J=J_0-\sum_{p=1}^{P}\sum_{t=1}^{T-1}(\beta_{t+1}^p{}'\alpha_{t+1}^p-H_t^p)-\sum_{p=1}^{P}\beta_1^p{}'(a_1-X_p) \tag{3.5}$$

The necessary conditions for optimality are:

$$\frac{\partial J}{\partial \alpha_t^p}=0 \tag{3.6}$$

$$\frac{\partial J}{\partial W(t)}=0 \tag{3.7}$$

From (3.6) it follows that

$$\beta_t^p=\frac{\partial H_t^p}{\partial \alpha_t^p}\quad t=1,...,T-1 \tag{3.8}$$

And the final condition

$$\beta_T^p=2(Y_p-\alpha_T^p)$$

Also , (3.2) implies

$$\alpha_1^p=X_p$$

$$\alpha_{t+1}^p=\frac{\partial H_t^p}{\partial \beta_{t+1}^p}\qquad t=1,...,T-1 \tag{3.9}$$

Equations (3.8) and (3.9) show the Hamiltonian structure of the problem and that the α's and the β's are conjugate quantities. It is also clear that the β's are back propagated quantities. Indeed they are very closely related to the back propagated errors of the Back Propagation (BP) algorithm as will be immediately evident.

As in the previous section, (3.7) is not very useful for determination of $W(t)$. We will instead use the classical back propagation algorithm. That is, we will use the update rule (in the steepest descent direction)

$$W_{l+1}(t)=W_l(t)-\epsilon\cdot\frac{\partial J}{\partial W(t)}$$

It is easy to prove after some manipulation that

$$\frac{\partial J}{\partial W(t)}=\sum_{p=1}^{P}\nabla F\beta_{t+1}^p\alpha_t^p{}'$$

For the details of the proof see leCun [7]. [3] So the rule is:

$$W_{l+1}(t)=W_l(t)-\epsilon\cdot\sum_{p=1}^{P}\nabla F\beta_{t+1}^p\alpha_t^p{}'$$

[3] ∇F is a diagonal matrix: $\nabla F_{ii}=\frac{\partial F_i(z)}{\partial z}$ evaluated at $z=\sum_j w_{ij}\alpha_j$.

This is exactly the Back Propagation algorithm. Once again, starting from an Optimal Control formulation we discover backpropagated quantities that have been of great importance and independently discovered in the traditonal formulation. Let us summarize. In this Section we have learned the following:

1. The standard CN we have considered has a multistage, feedforward structure. There is a weighted linear combination of input from the previous stage, that feeds into a nonlinear function.

2.The training of the CN is a constrained optimization problem. Alternatively, it can be seen as an Optimal Control Problem.

3. In solving by the BP algorithm, certain backpropagated quantities appear, that can be interpreted either as BP errors or Lagrange multipliers. They are conjugate quantities to the network unit outputs.

4 Similarities and Differences of HMM's and CN's

In the preceding two sections we have performed essentially an identical analysis of the HMMN and the CN. We have placed certain restrictions on the type of CN considered but these can be easily removed (see [7] for a discussion of this). At any rate, there are certain striking similarities between the two kinds of networks. They can be summarized as follows:

1. Both networks essentially have the same architecture. There is a nonlinearity in every layer and then linear combinations of the previous layer units are used as input for the next layer.

2. Training is very similar for both types of networks. In both cases we must optimize a certain cost function of the network.

3. The training problem admits in both cases an Optimal Control solution . In the course of this solution, back propagated quantities turn out to be of importance.

In addition there are some differences:

1. The input is utilized in different ways. In the HMMN, one could argue there is no hidden layer. All layers are available to accept input. (However, in both cases the input is some observation, O_t for HMMN, X_t for CN.)

2. The nonlinearity is also used in a different way. It is well known that the addition of nonlinearities in Connectionist Networks, increases their power and scope. This nonlinearity acts on the output of intermediate layers. In the HMMN context, there is no nonlinear action in the layers output. The nonlinearity operates only on the input. It is interesting to speculate on possible improvements if nonlinear functions operated between layers of HMMN's.

In addition there are some general comments to be made. For instance:

1. In network analysis there is an interesting interplay between space and time. There is a certain amount of interchangeability between the two concepts. This is very clearly exemplified in the present situation. We have been looking at networks which can be thought of as evolving either in space (that is from layer to layer, which is traditional Connectionist point of view) or in time (Optimal Control point of view). There are certain gains to be had from each point of view and techniques that are well known in one field might prove useful to the other.

2. The interchanegability of time and space also shows that what is feedforward to one point of view, may be feedback to another point of view. We have been talking in this paper of feedforward architecture; this is clearly true from the space evolution perspective. But in Optimal Control terms, we have been looking at feedback systems: there is only one layer and its output at time t is used as input at time $t + 1$.

3. This raises the question of the possibility of combining the two perspectives. What would the dynamics of a network of many hidden layers be? In such a network the output unit would be fed back to the input unit. In the connectionist context, such networks are called recurrent networks. From the point of view of Optimal Control they are distributed systems (see [10]). It is quite possible that Control Theory may prove useful in investigating such networks. In particular, the prevalent view among connectionists seems to be that hidden layers are mostly useful to shape the response of the output layer; but in the control context, lots of effort goes into controlling every one of the layers of a network, to optimize given criteria.

There is much space for speculation along these lines. However, the point that has been argued here is that there is a basic similarity between HMM's and CN's both in architecture and training. This similarity is most obvious form the Optimal Control Perspective.

5 References

1. H. Bourlard and C.J. Wellekens,"Links between Markov Models and Multilayer Perceptrons", Proceedings of NIPS Conference, Denver , 1988.

2. H. Bourlard and C.J. Wellekens, "Speech Dynamics and Recurrent Neural Networks", Proceedings of ICASSP'89, Glasgow, 1989.

3. A.P. Dempster, N.M. Laird and D.B. Rubin, " Maximum Likelihood from Incomplete Data via the EM algorithm", J. of the Stat. Roy. Soc. B, Vol. 39, No. 1, 1977,pp. 1-38.

4. S.Y.Kung and J.N.Hwang, "A Unified Modelling of Connectionist Neural Networks"

5. S.Y.Kung and J.N.Hwang, "A Unifying Viewpoint of Multilayer Perceptrons and Hidden Markov Models", Proceedings ofIEEE Int. Symposium on Circuits and Systems, Portland, May 1989.

6. S.Y.Kung and J.N.Hwang, "A Unified Systolic Architecture for Artificial Neural Networks", Proceedings of ICASSP'89, Glasgow, May 1989.

7. Y. leCun, "A Theoretical Framework for Back-Propagation", Proc. of the 1988 Connectionist Models Summer School, Pittsburgh, 1988, pp.21-28

8. 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", Bell Sys. Tech. J., Vol. 62, No. 4, April 1983,pp.1035-1074.

9. J.E. McClelland, D.E. Rumelhart, "Parallell Distributed Processing", Cambridge, MIT, 1986.

10. A.P. Sage, "Optimum Systems Control", Englewood Cliffs, Prentice-Hall, 1968.