LEARNING PROCESS IN AN ASYMMETRIC THRESHOLD NETWORK.

YANN LE CUN

Ecole Supérieure d'Ingénieurs en Electrotechnique et Electronique.
89, rue Falguière 75015 PARIS
and
Laboratoire de dynamique des réseaux. 1, rue Descartes 75005 PARIS.

1 - INTRODUCTION

Threshold functions and related operators are widely used as basic elements of adaptive and associative networks [Nakano 72, Amari 72, Hopfield 82]. There exist numerous learning rules for finding a set of weights to achieve a particular correspondence between input-output pairs. But early works in the field have shown that the number of threshold functions (or linearly separable functions) in N binary variables is small compared to the number of all possible boolean mappings in N variables, especially if N is large. This problem is one of the main limitations of most neural networks models where the state is fully specified by the environment during learning: they can only learn linearly separable functions of their inputs. Moreover, a learning procedure which requires the outside world to specify the state of every neuron during the learning session can hardly be considered as a general learning rule because in real-world conditions, only a partial information on the "ideal" network state for each task is available from the environment. It is possible to use a set of so-called "hidden units" [Hinton,Sejnowski,Ackley. 84], without direct interaction with the environment, which can compute intermediate predicates. Unfortunately, the global response depends on the output of a particular hidden unit in a highly non-linear way, moreover the nature of this dependence is influenced by the states of the other cells. Thus, it is difficult to decide whether the output of a hidden unit is wrong for a particular input, and, consequently, how to modify its weights. This last problem has been referred to as the "credit assignment problem" (CAP) in [Hinton & al. 84]. Attempts to find a learning rule taking into account hidden units and generating high order predicates failed until recently, which could explain for the decrease of interest in this field for the past 15 years [Minsky & Papert 68].

In this paper, we consider learning and associative memorization as dynamic processes and show how to describe the evolution of the weights through an "energy" function. This method will be used to solve the CAP and applied to a model of hierarchical associative memory called HLM (Hierarchical Learning Machine).

NATO ASI Series, Vol. F20
Disordered Systems and Biological Organization
Edited by E. Bienenstock et al.
© Springer-Verlag Berlin Heidelberg 1986

2- LEARNING AS ITERATIVE MINIMIZATION

In the following, we consider a neural network with state vector $\mathbf{X}$ in $\{-1,+1\}^N$. The time evolution of the network is described by:

$$\mathbf{X}(t+1) = \mathbf{F}[\mathbf{W}(t).\mathbf{X}(t)] \qquad (1)$$

where $\mathbf{W}(t)$ is the weight matrix at time t and $\mathbf{F}$ is the mapping whose i^{th} coordinate has value +1 if the i^{th} coordinate of its argument is positive, -1 otherwise (threshold function).

Learning modifies the weights and can be viewed as minimizing a given cost function or criterion. The simplest way to minimize a function is to use a gradient descent method: given C(t) where t is a discrete time index and whose time average <C> is taken as the criterion, the recursive formula:

$$\mathbf{W}(t) = \mathbf{W}(t-1) - \mathbf{K}(t).\ \mathrm{grad}_{\mathbf{W}}\,[C(t)]. \qquad (2)$$

minimizes <C> with respect to $\mathbf{W}$, with a noise level defined by $\mathbf{K}$ (where $\mathbf{W}(t)$ is the weight matrix at time t, and $\mathbf{K}$ is a diagonal positive matrix).

For example, we can choose for C the Hopfield's energy [Hopfield 82]:

$$C(t) = -1/2\ \mathbf{Y}(t)^T.\mathbf{W}(t).\mathbf{Y}(t) \qquad (3)$$

where the patterns to be memorized, $\mathbf{Y}(t)$, are presented sequentially and $\mathbf{Y}(t)^T$ is $\mathbf{Y}(t)$ transpose. In this case, (2) gives:

$$W_{ij}(t) = W_{ij}(t-1) + K_i(t)Y_i(t)Y_j(t) \qquad (4)$$

which is the classical Hebb's law. Running this procedure will "dig holes" in the energy landscape around states to be memorized. Notice that this process diverges if the patterns are presented indefinitely.

Another possible criterion is:

$$C(t) = [\mathbf{Y}(t) - \mathbf{W}(t).\mathbf{Y}(t)]^T[\mathbf{Y}(t) - \mathbf{W}(t).\mathbf{Y}(t)] \qquad (5)$$

in case of auto-association ($\mathbf{Y}(t)$ associated to itself) or

$$C(t) = [\mathbf{Y}(t) - \mathbf{W}(t-1).\mathbf{X}(t-1)]^T[\mathbf{Y}(t) - \mathbf{W}(t-1).\mathbf{X}(t-1)] \qquad (6)$$

in case of hetero-association ($\mathbf{Y}(t)$ associated to $\mathbf{X}(t-1)$).

Equation (2) then gives:

$$W_{ij}(t) = W_{ij}(t-1) + K_i(t)[Y_i(t)-A_i(t)]X_j(t-1) \qquad (7)$$

where $\mathbf{A}$ is the vector defined by: $\mathbf{A}(t) = \mathbf{W}(t-1).\mathbf{X}(t-1)$ (ie A_i is the total input to cell i).

Equation (7) is known as the Widrow-Hoff rule [Widrow & Hoff. 60] (or least mean square algorithm) and is strongly related to the pseudo-inverse method [Duda & Hart. 73, Kohonen 74, 84].

Under some conditions on $\mathbf{K}$, this method provides an exact solution when the $\mathbf{X}$'s are linearly independent. Otherwise, the process is still convergent, but may find only sub-optimal solutions. Nevertheless, a "good" solution is found in both the separable and the non-separable cases [Duda & Hart. 73].

A similar method can be applied to the Boltzmann Machine [Hinton & al. 84] if we take:

$$C(t) = \mathbf{X}(t)^T.\mathbf{W}(t).\mathbf{X}(t)\rangle - \langle\mathbf{Y}(t)^T.\mathbf{W}(t).\mathbf{Y}(t) \qquad (8)$$

where $\mathbf{X}$ is the network state when input cells only are clamped and $\mathbf{Y}$ is the network state when both input and output cells are clamped. (2) now gives:

$$W_{ij}(t) = W_{ij}(t-1) + K_i(t)[Y_i(t)Y_j(t) - X_i(t)X_j(t)] \qquad (9)$$

This is a deterministic version of the Boltzmann Machine learning rule. This algorithm solves the "credit assignment problem", because the **Y**'s have a dynamics which is not fully specified by the external environment.

3- THE HIERARCHICAL LEARNING MACHINE

In the learning rules described above, the **Y**'s played the role of desired states. In classical neural network models, without hidden units, the desired states are fully specified externally, while in the Boltzmann Machine and the HLM, they are partly computed by the network itself. Solving the CAP requires being able to change the output of each hidden unit so as to satisfy the global criterion. This can be done by computing each local criterion -attached to hidden units- while remaining consistent with all other local criteria.

In the following, we consider "smooth" neural networks, with real valued cell outputs defined by $X_i(t+1) = f[\sum_j W_{ij}(t)X_j(t)]$, where f is an odd function, with strictly positive first derivative. Let us denote $A_i(t)=\sum_j W_{ij}(t)X_j(t)$ and C a global criterion, depending on **A**, and defined by:

$$C(t) = \sum_{j=1}^{N} C_j(t) \qquad (10)$$

where C_j is the criterion attached to the j-th cell and depends on A_j.

Assume that cell k is a hidden unit and let C_k be its local criterion. An optimal definition for C_k can be chosen such that minimizing C_k with respect to the $W_{k\ell}$ (ℓ=1,...,N) amounts to minimizing $\sum_{j\neq k} C_j$ (assuming that all C_j , $j\neq k$ are known). This condition can be expressed by:

$$\frac{\partial C_k}{\partial W_{k\ell}} = \sum_{j\neq k} \frac{\partial C_j}{\partial W_{k\ell}} \qquad \ell=1,\dots,N \qquad (11)$$

It is possible to use a weaker constraint:

$$\mathrm{SIGN}\left[\frac{\partial C_k}{\partial W_{k\ell}}\right] = \mathrm{SIGN}\left[\sum_{j\neq k} \frac{\partial C_j}{\partial W_{k\ell}}\right] \qquad \ell=1,\dots,N \qquad (12)$$

where SIGN is the threshold function . This condition ensures that C_k and $\sum_{j\neq k}C_j$ vary in the same direction. We can compute each term:

$$\frac{\partial C_j}{\partial W_{k\ell}} = \frac{\partial C_j}{\partial A_j}.\frac{\partial A_j}{\partial X_k}.\frac{\partial X_k}{\partial A_k}.\frac{\partial A_k}{\partial W_{k\ell}} = \frac{\partial C_j}{\partial A_j} W_{jk} \frac{\partial A_k}{\partial W_{k\ell}}.\frac{\partial X_k}{\partial A_k} \qquad (13)$$

$$\frac{\partial C_k}{\partial W_{k\ell}} = \frac{\partial C_k}{\partial A_k}.\frac{\partial A_k}{\partial W_{k\ell}} \qquad (14)$$

Condition (12) implies:

$$\mathrm{SIGN}\left[\frac{\partial C_k}{\partial A_k}\right] = \mathrm{SIGN}\left[\sum_{j\neq k} W_{jk} \frac{\partial C_j}{\partial A_j} \frac{\partial X_k}{\partial A_k}\right] \qquad (15)$$

The last term in the bracketed sum is always positive, hence:

$$\mathrm{SIGN}\left[\frac{\partial C_k}{\partial A_k}\right] = \mathrm{SIGN}\left[\sum_{j\neq k} W_{jk} \frac{\partial C_j}{\partial A_j}\right] \qquad (16)$$

If we take for C its simplest form: $C_i = - Y_i\ A_i$ (17)

we obtain: $\mathrm{SIGN}[\ Y_k\ ] = \mathrm{SIGN}[\ \sum_{j\neq k} W_{jk}\ Y_j\ ]$ (18)

We can choose any absolute value for Y_k, in particular:

$$Y_k = \mathrm{SIGN}[\ \sum_{j\neq k} W_{jk}\ Y_j\ ] \qquad (19)$$

Using vector notation, (19) becomes: $\mathbf{Y} = \mathbf{F}(\mathbf{W}^T\ \mathbf{Y})$ (20)

and we can compute a "desired response" for each hidden cell by back propagating this cost function gradient [Hinton 85].

Minimizing the error between **Y** and **X** can be done with the Widrow Hoff rule (7). We obtain a system of recursive equations describing the complete behavior of HLM:

$$\begin{aligned} &\mathbf{X}(t+1) = \mathbf{F}[\ \mathbf{W}(t)\ .\ \mathbf{X}(t)\ ] \\ &\mathbf{Y}(t+1) = \mathbf{F}[\ \mathbf{W}(t)^T\ .\ \mathbf{Y}(t)\ ] \\ &W_{ij}(t+1) = W_{ij}(t) + K_i(t)[Y_i(t+1)-A_i(t+1)]X_j(t) \qquad (21) \\ &A_i(t+1) = \sum_k W_{ik}(t) X_k(t) \end{aligned}$$

assuming that the X's and Y's of some cells can be clamped externally.

4- SIMULATIONS

The learning rule we obtain is totally local in space and time. The global criterion <C> is convex provided there is no hidden cell. This implies that several solutions to a particular task can co-exist, and that the weights can be trapped into a local minimum of <C>. This means too that the weight configuration found out by learning depends on the initial weight configuration. Moreover, it depends on the whole network history: what has been learned before and how.

Two parameters have a great influence on the weight dynamics. The first one is the matrix **K** which defines the step size towards the criterion minimum at each iteration. A trade-off must be found between the speed of convergence (proportional to the K_i's) and the accuracy and stability of the stable point (achieved with small K_i's). It must be noticed that large values for K_i's facilitate escaping from local minima because large energy barriers can be crossed over. The second parameter is the nature of the learning pattern sequence. By definition, the <C> landscape is statistically defined and can be completely modified by changing the relative occurrence frequency of each pattern. Increasing the occurrence frequency of poorly learned patterns increases the energy of the nearest local minimum, and therefore, lowers the heights of the surrounding barriers. This can define what a "good pedagogy" must be, and can be interpreted as a "smart noise".

A particular structure has been chosen for testing the algorithm. It has some restrictions to facilitate the study of the weight dynamics. First, the cells are threshold automata (i.e. function f is approximated by a threshold function); second, all the interactions are local in a 3-D space; third, there is no loop in the interaction graph, which means that the matrix **W** is isomorphic to a triangular matrix. The network has a hierarchical architecture with several layers. Each layer is composed of 64 cells (8 by 8 plane) with a toric topology to avoid boundary effects. The first layer is the input plane (retina). Each cell in a layer receives signals only from 9, 25, or 49 cells in the previous layer. There is no interaction between cells in a given layer. The last layer (output plane) has a variable number of cells which "see" the entire previous layer (see fig 1). This structure allows hierarchical information processing, the abstraction level of the representation increases as the information is processed by the successive layers. The locally connected loop-free structure causes the **X** and **Y** dynamics to have trivial fixed points, and makes the study of the weight dynamics easier. It must be noticed that in such a struture, since there is no direct path between input and output cells, the information *has* to be processed by the hidden units.

A learning iteration is composed of three phases:

- Present a pattern in the input plane (clamp the X_i's of the first layer cells) and compute the stable **X** state.
- Present the associated desired response (clamp the Y_i's of the output cells) and compute the stable **Y** state (using back-propagation).
- Modify the weights using Widrow-Hoff rule with **X** and **Y.**

For the simulations, the matrix **K** is chosen as **K**= k.**I** , where **I** is the unit matrix. k is modified manually during the learning phase.

The most important thing we have to test by simulation is whether the network is able to generalize or not. The generalization is the ability to produce a correct response for a non learned input pattern. Without generalization, learning is only memorizing. We have chosen for that test a set of alphabetic characters presented in several positions on the retina (See fig 2). There are 5 instances of the first 6 characters of the alphabet, each one presented in 4 different positions (total of 120 patterns). There are 6 classes, each of them associated with the activation of one of the 6 output cells. A correct classification (with 100% recognition rate) is obtained with a 6 layer network and connectivity 25. This task is somewhat complex because very different patterns (in the Hamming sense) must be put into the same class, while some close patterns must be separated.

Fig. 3 shows the classification achieved by a 6 layer network with connectivity 49 when 5 to 7% of the pixels have been randomly inverted during learning and recognition phases.

Fig. 4 shows classification made by the previous network on a set of non-learned input patterns. With this kind of data, the observer has a "semantic criterion" (the only one possible) to test the quality of the generalization. Of course, Hamming distance could be used to measure the spatial basins of attraction, but this would have little semantic relevance in this context.

In these examples, the generated boolean function is not linearly separable and requires using hidden units. The learning phase is somewhat long (a few thousands iterations), and needs a sophisticated pedagogy. All the learning set must not be presented at once (new patterns may be added when previous ones are learned). The value for k must be carefully chosen, decreased when learning is successfull, increased when trapped in a local minimum. The patterns must be presented in a random order.

Note that by presenting a pattern until it is recognized instead of changing it at each iteration, the learning time is reduced by a significant amount.

Other simulations have been performed in a more autonomous learning mode that modelizes Pavlovian conditioning [Le Cun 85]. In this last experiment, the network response is taken as the desired output, i.e. the desired response is self-generated.

5 - CONCLUSION

Numerous simulations must be made to evaluate the algorithm performances, for instance by using a structure with loops, and real valued outputs.

Nevertheless, this demonstrates the possibility of learning without specifying the states of all neurons.

The behaviour of HLM shows strong analogies with animal learning, especially when considering the effects of a pedagogy on the results.

REFERENCES

[AMARI S.I.] : "Learning patterns and patterns sequences by self-organizing net of threshold elements". IEEE Trans. Com. Vol C-21, No 11, Nov 72.

[DUDA R.O., HART P.E.] : "Pattern classification and scene analysis". Wiley 73.

[HOPFIELD J.J.] : "Neural networks and physical systems with emergent collective computational abilities". P. Nat. Ac. Sci. USA, Nov 82.

[HINTON G., SEJNOWSKI T., ACKLEY D.] : "Boltzmann Machines, constraint satisfaction networks that learn". CMU Tech. Rep. CS-84-119, May 84.

[HINTON G.] , Private communication 1985.

[KOHONEN T., RUOHONEN M.] : "Representation of associated data by matrix operators". IEEE Trans. Computers, July 1973.

[KOHONEN T.] : "An adaptive associative memory principle". IEEE T. Comp, Apr 74

[KOHONEN T.] : "Self-organization and associative memories". Springer 1984.

[LE CUN Y.] : "A learning scheme for asymmetric threshold network". Proc. of COGNITIVA 85, Paris, June 1985 (in french).

[MINSKY M., PAPERT S.] : "Perceptron". M.I.T. Press, 1968.

[NAKANO K.] : "Associatron, a model of associative memory". IEEE Trans Syst. Man Cyb., Vol SMC -2, No 3, July 1972.

[WIDROW B., HOFF H.E.] : "Adaptive switching circuits". 1960 IRE WESCON Conv. Record, Part 4, 96-104, Aug 1960.

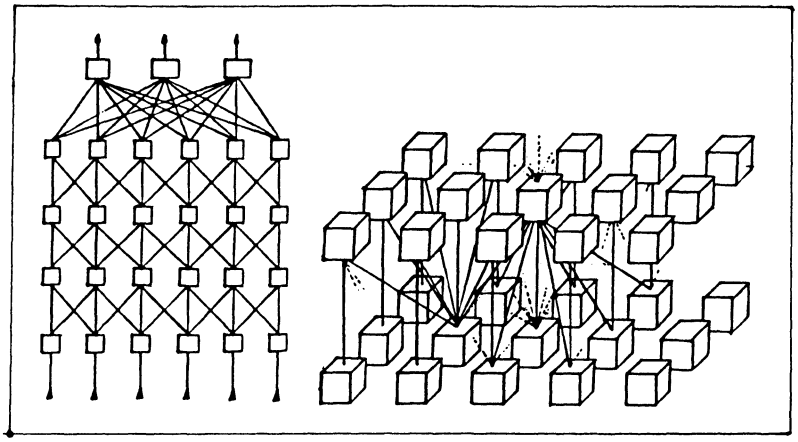

FIG. 1 : The network structure chosen for the simulations is Three-dimensional and composed of several 8 by 8 layers. Each cell in a layer is connected to 9, 25 or 49 cells in the previous layer. The last layer is fully connected to the previous one.

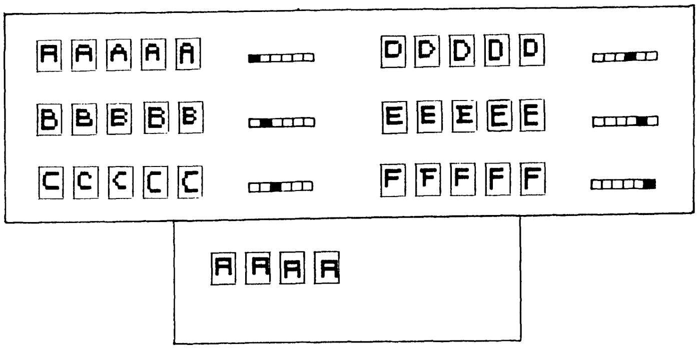

FIG. 2 : Five examples of the first six alphabetic characters are presented in four different positions. Each class is coded by the activation of one of the output cells.

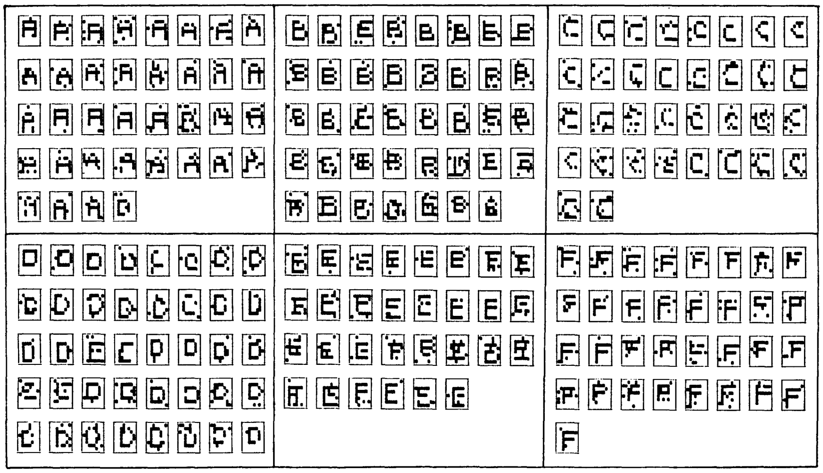

FIG. 3 : The figure shows the classification produced by a six layer network with 7 by 7 receptive fields. 5 to 7% of the pixels were randomly inverted during learning and recognition.

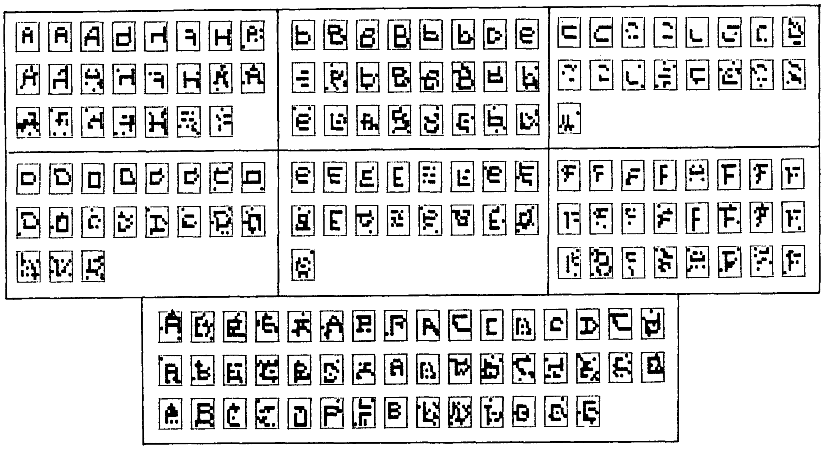

FIG. 4 : A generalization test set. Classification produced by the same network as in fig. 3 on distorted noisy patterns. The patterns for which none of the six desired responses has been produced are put together at the bottom of the figure.