Generalised scheme for optimal learning in recurrent neural networks

K. Shanmukh
Y.V. Venkatesh

Indexing terms: Neural network, Perceptron, Hopfield network, Bidirectional associative memory, Maximal stability, Optimal learning, AdaTron

Abstract: A new learning scheme is proposed for neural network architectures like Hopfield network and bidirectional associative memory. This scheme, which replaces the commonly used learning rules, follows from the proof of the result that learning in these connectivity architectures is equivalent to learning in the 2-state perceptron. Consequently, optimal learning algorithms for the perceptron can be directly applied to learning in these connectivity architectures. Similar results are established for learning in the multistate perceptron, thereby leading to an optimal learning algorithm. Experimental results are provided to show the superiority of the proposed method.

Introduction

A major goal of present day research on artificial neural networks **(ANN)** is to design the interconnections among a large number of (primitive) neurons, in such a way as to realise appropriate pattern recognition characteristics.

The **ANNs** range from perceptrons (with no feedback) [1] to cellular networks (with local feedback) to Hopfield networks (with global feedback) [2]. When the **ANN** is to be used for pattern analysis, the stability of the network as a dynamic system is of paramount importance. If the network, starting from an arbitrary point in state-space, evolves in time and reaches a stable equilibrium point, then the network is interpreted as having recognised the pattern corresponding to the stable state. The distinctive characteristic of **ANNs** is that this type of convergence to the stable state may take place, even though the input pattern is an incomplete or distorted version of the correct pattern. The **ANN** may have many points of stable equilibria, corresponding to the many patterns which it can recognize. The region around an equilibrium point, consisting of all the points starting from which the trajectory of the dynamical system will converge to the equilibrium point, is called the basin of attraction for the equilibrium point. The **ANN** is to be so designed as to have the basin of attraction as large as possible for each equilibrium point.

The synthesis of the **ANNs** for pattern analysis involves designing the dynamical system, in such a way as to realise a separate equilibrium point for each of the patterns to be recognised. This implies a learning stage, in which the training patterns are used to fix the connection strengths.

In this paper, we develop an optimal learning scheme for associative memory architectures like the Hopfield network and bidirectional associative memory. We generalise this for the multistate perceptron. This is an indication of the special feature of the proposed technique, which can be used for learning in any neural network architecture without hidden neurons, as long as the problem of learning can be converted to the problem of satisfying **a** set of linear inequalities of a particular type.* In order to avoid terminological confusion, we employ, throughout the paper, the terms 'Hopfield network' and **'BAM'** to refer to connectivity architectures and not to the learning algorithms.

2 Perceptron and the perceptron convergence theorem

A two-state perceptron is a single neuron which takes an n-dimensional vector as input, and gives a binary output, see Fig. **1.** The input–output relation is nonlinear, and is given by

$$y = \mathrm{sgn}\,(W^T X - t)$$

where X and W are the input and weight vectors, respectively, t a scalar called the threshold, y the output, T the transpose operation, and the function sgn (h) is -1 for $h < 0$ and $+1$ for $h \geqslant 0$. It is known that the Perceptron can be used to solve the linearly separable two-class classification problems. In this case, the set of the patterns belonging to the two classes is the 'training set', and

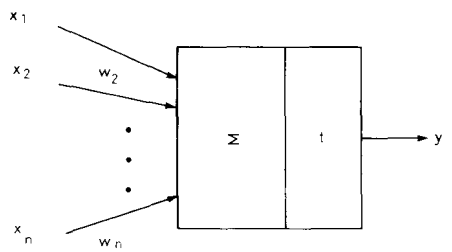

Fig. 1 *Two-state perceptron*

* **At the time of the present revision of the paper, the authors came across the latest reference [14] in which a similar criterion for optimality has been used. However, there is no proof of convergence in Reference 14, and the framework used for establishing the domain of attraction is different.**

'learning' is the process of finding W and t such that the perceptron classifies the patterns correctly.

Formally, let $\{Xi\}$ $s = 1, 2; k = 1, 2, \ldots, N(s)$ be the set of patterns. Here s stands for the class number, and k is the pattern number within the sth class. Let the output, y^s, be -1 for $s = 1$ and $+1$ for $s = 2$. Learing amounts to finding W and t such that

$$(W^T X_k^s - t)y^s > 0 \quad \text{for } s = 1, 2 \forall k \tag{1}$$

The following well-known perceptron convergence theorem [1] gives an iterative learning algorithm that converges to a solution whenever there exists one. Without loss of generality, we neglect the threshold term, t, in the following discussion.

Perceptron-convergence theorem: Starting from the initialisation weight vector, $W(0) = 0$, each of the patterns, X_i^s, is repeatedly presented to the algorithm. If X_i^s is the pattern presented at step t, then the vector $W(t-1)$ of the preceding step is modified according to the rule

$$W(t) = W(t-1) + \varepsilon X_i^s y^s$$

$$\text{where } \varepsilon = \begin{cases} 1 & \text{if } W^T(t-1)X_i^s y^s < 0 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

If a solution W^* exists satisfying eqn. I, then this algorithm converges in a finite number of steps to a solution of eqn. 1.

The proof of this theorem (see Reference 1) involves showing that the Schwartz inequality,

$$\frac{W^{*T} W(t)}{\|W^*\| \, \|W(t)\|} \leqslant 1 \tag{3}$$

will be violated, if the modification of the weight vector does not stop after a finite number of iterations.

In the above algorithm, even if the modification of weights is done after a batch of patterns is presented, the method still converges.

2.1 Learning with constrained weights

It has been observed that, in the above algorithm, if an element of the weight vector, w_i, always gets the input of the same sign, and if the sign of w_i is opposite to that of the input, then that weight w_i can be kept constant while updating the vector W. Still the algorithm converges in a finite number of steps. The proof again uses the Schwartz inequality to show the number of corrections is bounded above.

Now consider the problem of finding a weight vector W such that

$$W^T X_i > t_i \quad t_i > 0 \; \forall i$$

In this case, too, the algorithm converges to a solution whenever there exists one. The proof of convergence follows from the above result, and the following representation of the problem:

$$W^T X_i > t_i \quad \forall i \Rightarrow (w_1 \quad w_2 \quad \cdots \quad w_n) \begin{pmatrix} X_i(1) \\ X_i(2) \\ \vdots \\ X_i(n) \end{pmatrix} > t_i \quad \forall i$$

$$\Rightarrow (w_1 \quad w_2 \quad \cdots \quad w_n \quad t_i) \begin{pmatrix} X_i(1) \\ X_i(2) \\ \vdots \\ X_i(n) \\ -1 \end{pmatrix} > 0 \; \forall i$$

In this formulation, t_i and -1 are of opposite signs, as a consequence of which we can keep t_i fixed without affecting the convergence.

Based on the above discussion, we observe that the PLA, in general, can solve a system of linear inequalities of the following type:

$$W^T X_i > t_i \quad \text{where } t_i\text{s are positive} \tag{4}$$

Viewing the PLA in this way is useful in many ways. For instance, any linear constraint on W of the type 4 can be allowed in the PLA, since it can be converted to an equivalent pattern. and added to the pattern set.

As an example, consider the problem of learning in a perceptron with sign-constrained weights [3]. If a weight w_i is constrained to be positive, then that constraint can be converted to the pattern $X = (0\ 0 \cdots 1 \cdots 0\ 0)^T$ so that

$$W^T X > 0 \Rightarrow w_i > 0$$

So the algorithm converges even when the signs of the weights are constrained, if there exists a solution with the given signs. In this way, we obtain the algorithm given in Reference 3 as a special case.

2.2 Perceptron of maximal stability

The perceptron generalises well to the classification of patterns which are not in the training set, if the inequalities 1 are satisfied for a positive constant, c, on the right-hand side instead of zero. An optimal learning algorithm is the one which results in the choice of W and t such that c is maximum, with respect to the normalised W. With such W and t, even if the patterns are corrupted to some extent, one can expect the classification to be correct because of higher tolerance in c. The perceptron with such W and t is the perceptron of maximal stability [4]. The determination of such W and t leads to the following optimisation problem for the pattern set $\{Xi\}$:

maximise c

subject to $(W^T X_k^s - t)y^s > c\|W\|$

Gardner *et al.* [5] show that the PLA can be used, for finding the weights, W, that satisfy the above inequalities for a fixed c. In Reference 6 an iterative algorithm, called AdaTron, is proposed to solve the above optimisation problem. In fact, AdaTron is a combination of the Adaline rule [7] and the PLA [1].

3 Optimal learning in various neural-network models

In this Section, we use the above procedure of converting the constraints on W into patterns, in order to develop optimal learning algorithms for the neural network models such as BAM, the Hopfield network and the multistate perceptron.

3.1 Learning in bidirectional associative memory and the Hopfield network

For clarity, we deal with the BAM and the Hopfield network separately. As is well known, the BAM is a two-layer feedback neural network, which can be used to store associations between patterns (see Fig. 2). The neurons in one layer are connected to the neurons in the other layer, but there are no connections within a layer. Let N and P be the number of neurons in the first and second layers, respectively. Let W_{ij} be the connection weight between neuron i in the first layer and neuron j in

useful for multi-class pattern classification. Here the output y can take any value from the set $\{0, 1, 2, \ldots, Q-1\}$. The perceptron now has $Q-1$ thresholds $(t_1 < t_2 < \ldots < t_{Q-1})$ to separate the neighbouring classes, see Fig. 6. The input–output relation is given by

$$\begin{aligned} y &= 0 && \text{if } W^T X_k^s < t_1 \\ &= s && \text{if } t_s \leqslant W^T X_k^s < t_{s+1} \\ &= Q-1 && \text{if } t_{Q-1} \leqslant W^T X_k^s \end{aligned}$$

The problem of learning is to find W and thresholds, t_1, $t_2, \ldots,$ such that the input patterns are mapped to their

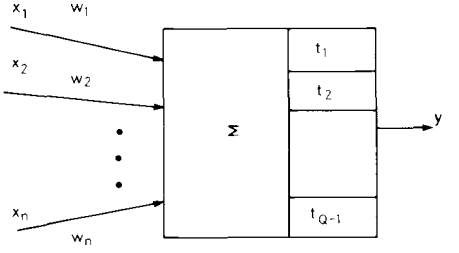

Fig. 6 *Multistate perceptron with Q states*

corresponding classes. This is equivalent to finding W and thresholds satisfying the following inequalities:

$$\begin{aligned} & W^T X_k^s < t_1 && \text{for } s = 0 \\ & t_s \leqslant W^T X_k^s < t_{s+1} && \text{for } s = 1, 2, \ldots, Q-2 \\ & t_s \leqslant W^T X_k^s && \text{for } s = Q-1 \end{aligned} \tag{9}$$

We define the gap between two neighbouring classes, s and $s+1$, as

$$g^s(W) = \min_{p,q} \frac{W^T(X_p^{s+1} - X_q^s)}{\|W\|}$$

We define $G(W)$ as the minimum gap between any two classes, given by

$$G(W) = \min g^s(W)$$

The optimal multistate perceptron is then defined as the one which maximises $G(W)$. This is equivalent to maximising c in the following inequalities:

$$\begin{aligned} & t_1 - W^T X_k^s > c\|W\| && \text{for } s = 0 \\ & t_{s+1} - W^T X_k^s > c\|W\| && \text{for } s = 1, 2, \ldots, Q-2 \\ & W^T X_k^s - t_s > c\|W\| && \text{for } s = 1, 2, \ldots, Q-2 \\ & W^T X_k^s - t_s > c\|W\| && \text{for } s = Q-1 \end{aligned}$$

In Reference **4,** an iterative algorithm is given to find W and ts that satisfy expr. 9, but the problem of finding the multistate perceptron of maximal stability is posed as an open question. Here we give a solution to this problem. First, we derive a learning algorithm for the multistate perceptron, by showing that learning in this case is equivalent to learning in the 2-state perceptron.

Consider the learning problem in a three-state perceptron. We need to find W, t, and t, such that

$$\begin{aligned} & W^T X_k^0 < t_1 \\ & t_1 \leqslant W^T X_k^1 < t_2 \\ & t_2 \leqslant W^T X_k^2 \end{aligned}$$

This is equivalent to finding the weight vector for the two-state perceptron, with the following input–output pattern pairs:

$$[X_k^0, -1, 0] \rightarrow -1 \quad [X_k^1, -1, 0] \rightarrow +1$$
$$[X_k^1, 0, -1] \rightarrow -1 \quad [X_k^2, 0, -1] \rightarrow +1$$

It can be verified that the weight vector which gives this mapping satisfies the constraints of the three-state perceptron. The last two elements in the weight vector of the two-state perceptron correspond to the thresholds of the three-state perceptron. Thus, by applying the learning algorithm of the two-state perceptron to the transformed input–output pairs, we obtain weights and the corresponding thresholds which solve the three-state perceptron learning problem. Therefore the convergence of the algorithm follows directly from the perceptron convergence theorem.

In order to get the weights and thresolds for a multistate perceptron, we generate transformed input–output pairs by appending additional element to the input patterns, see Fig. 7. The number of these additional elements

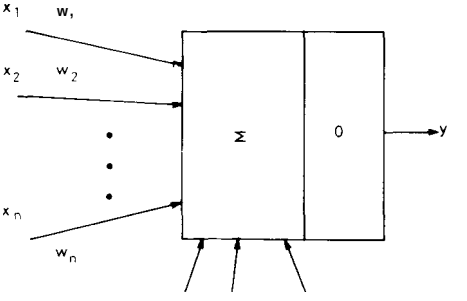

Fig. 7 *Perceptron equivalent to the multistate perceptron of Fig. 6*

is equal to the number of thresholds in the perceptron. The output is $+1$ or -1, depending on which threshold is being considered, and on which side of that threshold the input pattern is supposed to lie. Thus for patterns in classes 0 and $Q-1$, we need to check with only one threshold, and, correspondingly, we will generate only one new input–output pair for every input–output pair in the original pattern set. And for patterns in other classes, we obtain two new input–output pairs for every input–output pair in the original pattern set, as they have to be compared with two thresholds. The transformed input–output pairs will be of the following form:

	0	1	⋯	s	$s+1$	⋯	$Q-1$		
$[X_k^s$	-1	0	⋯	0	0	⋯	0]	$-$	$-\ s \neq 0$
$[X_k^s$	0	0	⋯	-1	0	⋯	0]	$\rightarrow +1$	$s = 1, 2, \ldots, Q-2$
$[X_k^s$	0	0	⋯	0	-1	⋯	0]	$\rightarrow -1$	$s = 1, 2, \ldots, Q-2$
$[X_k^s$	0	0	⋯	0	0	⋯	-1]	$\rightarrow +1$	$s = Q-1$

Now, the problem of obtaining the optimal multistate perceptron is equivalent to obtaining the maximally stable two-state perceptron for the transformed input–output pairs. This maximises the minimum gap between any threshold and the patterns of its neighbouring class. This is exactly what we require for a maximally stable multistate perceptron. **So** the AdaTron algorithm of the two-state perceptron can be used to find the multistate perceptron with maximal stability characteristics.

4 Experimental results

In the first part of our experiments, we consider the problem of optimal learning in BAM and the Hopfield network. We consider a BAM with $N = P = 15$, and a Hopfield network with 30 neurons. We compare the proposed method with the weighted learning method of Reference 10, using a set of test patterns. The number of patterns stored, M, is varied from 2 to 7. For each value of M, 1000 experiments are performed. In each experiment, patterns are generated randomly, and the two algorithms are applied. If both the methods are successful in storing the patterns, then the network is tested for pattern-retrieval performance. For this purpose, 300 patterns are generated randomly, and 100 of them are corrupted by flipping 10% of the bits, another 100 patterns are corrupted by flipping 15% of the bits, and 20% of the bits are flipped in the remaining 100 patterns. The network is initialised with the corrupted pattern, and allowed to run. Then the results are checked to verify whether the network has converged to the correct pattern.

Results of the comparisons for BAM are summarised in Tables 1–3: Table 1 gives the percentage of experiments in which the two methods were successful in storing the patterns. Table 2 contains $(c/\|W\|_i)$ (where $\|W\|_i$ is the Euclidean norm of the vector of weights connecting neuron i to other neurons; and c, as obtained by the algorithm, is the largest positive number that satisfies expr. 7 with respect to a normalised weight matrix), which is a measure of the size of the basin of attraction. This parameter is averaged over all neurons, and over all experiments in which both the methods were successful in storing the patterns. Table 3 gives the percentage of successful recalls, i.e. runs in which the network converged to the correct pattern in the pattern retrieval phase. Tables 4–6 give similar results for the Hopfield network.

Table 1: Storage of patterns in BAM

M	2	3	4	5	6	7
Weighted Hebb rule	100	98.8	89.0	57.9	20.8	3.3
Proposed method	100	99.9	99.9	99.7	99.6	99.2

Table 2: Measure of basin of attraction in BAM

M	2	3	4	5	6	7
Weighted Hebb rule	2.64	1.61	1.06	0.76	0.58	0.49
Proposed method	2.69	1.88	1.43	1.19	1.03	0.94

Table 3: Percentage of successful recalls in BAM

M	2	3	4	5	6	7
Weighted Hebb rule	99	91	80	70	60	51
Proposed method	99	93	85	78	67	64

In the second part of our experiments, we applied the proposed method for learning in the multistate perceptron. For simplicity, we considered a 3-state perceptron, and generated 3 sets of random patterns from the three square regions shown in Fig. 8. Our results are compared with those obtained from the learning algorithm given in Reference 4.

Table 4: Storage of patterns in the Hopfield network

M	2	3	4	5	6	7
Weighted Hebb rule	99	97	90	77	51	22
Proposed method	100	100	100	100	100	100

Table 5: Measure of basin of attraction in the Hopfield network

M	2	3	4	5	6	7
Weighted Hebb rule	3.54	2.35	1.68	1.24	0.99	0.82
Proposed method	3.78	3.03	2.57	2.25	2.01	1.81

Table 6: Percentage of successful recalls in the Hopfield network

M	2	3	4	5	6	7
Weighted Hebb rule	99	95	89	79	71	64
Proposed method	99	99	98	96	94	91

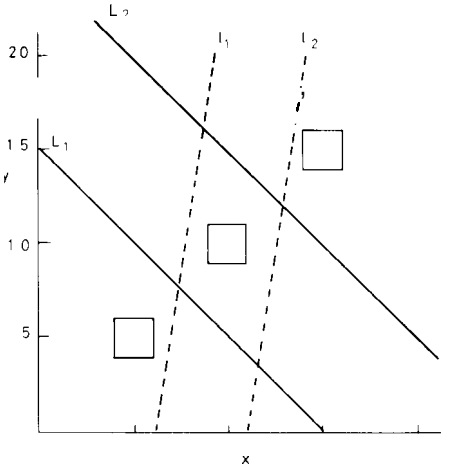

Fig. 8 *Three-state perceptron*

In Fig. 8, the dotted lines l_1 and l_2 show the decision boundaries obtained by the algorithm in Reference 4. The solid lines L_1 and L_2 in the Figure are the ones obtained by our method. Clearly, L_1 and L_2 are better decision boundaries than l_1 and l_2. From these results, the superiority of the proposed method is evident.

5 Conclusions

In this paper, we have described a method of exploiting the learning algorithms of a two-state perceptron, to obtain optimal weight matrices for general neural architectures like the bidirectional associative memory, Hopfield network and the multistate perceptron. More details of the proposed method can be found in References 12 and 13. This procedure is applicable even when the connectivity is partial, as in the case of cellular neural networks. It can also be applied to learning in the Hopfield network/BAM with multistate neurons.

6 References

1 MINSKY, M.L., and PAPERT, S. 'Perceptrons' (Cambridge MA MIT Press, 1969)

2 HOPFIELD, J.J.: 'Neural networks and physical systems with emergent collective computational abilities', *Proc. Natl. Acad. Sci.*, 1982, **79**, pp. 2554–2558

3 AMIT, D.J., WONG, K.Y.M., and CAMPBELL, C.: 'Perceptron learning with sign-constrained weights'. *J. Phys. A. Math. Gen.*, 1989.21, pp. 2039–2045

4 ELIZALDE, E., and GOMEZ, S.: 'Multistate perceptrons: learning rule and perceptron of maximal stability', *J. Phys. A, Math. Gen.*, 1992.25, pp. 5039–5045

5 GARDNER, E.: 'The space of interactions in neural network models', *J. Phys. A, Math. Gen.*, 1988, 21, pp. 257–270

6 ANLAUF, J.K., and BIEHL, M.: 'The AdaTron: an adaptive perceptron algorithm', *Europhys. Left.*, 1989, **10**, (7), pp. 687–692

7 WIDROW, G., and HOFF, M.E.: 'Adaptive switching circuits'. IRE, Western Electronic Show and Convention, Convention Record, Part 4, pp. 96–104
8 KRAUTH, W., and MEZARD, M.: 'Learning algorithms with optimal stability in neural networks', *J. Phys. A, Math. Gen.*, 1987, 20, p. L745
9 KOSKO, B.: 'Adaptive bidirectional associative memories', *Appl. Opt.*, 1987.26, (23), pp. 4947–4960
10 WANG, T., ZHUANG, X., and XING, X.: 'Weighted learning of bidirectional associative memories by global minimization', IEEE *Trans. Neural Netw.*, 1993.3, (6), pp. 1010–1018
11 TOU, J.T., and GONZALEZ, R.C.: 'Pattern recognition principles' (Addison-Wesley, 1974)
12 SHANMUKH, K., and VENKATESH, Y.V.: 'On an optimal learning scheme for bi-directional associative memories'. Int. Joint Conf. on Neural Networks, Nagoya, 1993
13 SHANMUKH, K., and VENKATESH, Y.V.: 'Multistate perceptron with optimal stability'. ANNNSI-93, Bombay, 1993
14 WANG, T., ZHUANG, X., and XING, X.: 'Designing bidirectional associative memories with optimal stability', IEEE *Trans. Sysr., Man Cybern.*, 1994, SMC-24, (5), pp. 778–790