In Proceedings of IJCNN, pp. I926–I931 (IEEE Press, 1992)

LEARNING IN CONNECTIONIST NETWORKS USING THE ALOPEX ALGORITHM

K P Unnikrishnan* **K P Venugopal#**

* Computer Science Department, GM Research Laboratories, Warren, MI 48090.
# Center for Advanced Marine Systems, Florida Atlantic University, Boca Raton, FL 33431.

Abstract

We describe the Alopex algorithm as a universal learning algorithm for neural networks. The algorithm is stochastic and it can be used for learning in networks of any topology, including those with feedback. The neurons could contain any transfer function and the learning could involve minimization of any error measure. The efficacy of the algorithm is investigated by applying it on multilayer perceptrons to solve problems such as XOR, parity and encoder. These results are compared with that ones obtained using backpropagation learning algorithm. The scaling properties of Alopex are studied using the encoder problem of different sizes. Taking the specific case of XOR problem, it is shown that a smoother error surface, with fewer local minima could be obtained by using an information theoretic error measure. An appropriate 'annealing' scheme for the algorithm is described and it is shown that the Alopex can escape out of the local minima.

INTRODUCTION

Computing the gradient of a global performance measure with respect to the network parameters, using only locally available information, is a desirable feature in connectionist networks. In this case, the training algorithm becomes a separate entity and the interconnection strengths can be updated synchronously. This reduces the overhead on a practical hardware implementation. Also this gives additional flexibility to choose an appropriate performance function depending up on the type of the problem. The popular backpropagation(BP) algorithm developed by Rumelhart et al[1], recursively computes the gradient of the error by propagating it backwards through the layers. Because of the deterministic nature of the algorithm, the network may converge in to a local minimum. Also, the algorithm can minimize only the squared error. For specific classes of problems, a smoother error surface (with fewer local minima) can be obtained by choosing an appropriate error function. Hence an algorithm which uses a fixed error function may not be the best one in all the cases.

In this paper, we describe Alopex[2] as a universal learning algorithm for neural networks. It has the following advantages: (i) any error measure can be used; (ii) the neurons may contain any transfer function; (iii) multilayered networks with feedback can be trained; (iv) all connections in the networks are updated synchronously, allowing efficient parallelization; (v) stochastic nature of the algorithm can be used for overcoming local minima; (vi) the algorithm is easily implementable using analog circuitry. The algorithm uses correlation between changes in individual connection strengths and changes in global error measures for modification of connections. We investigate the efficacy of this algorithm for learning in multilayer perceptrons. Taking the specific case of XOR problem, we show that using an information theoretic error measure[7][8], smoother error surfaces and improved classifications are possible. An appropriate 'cooling schedule' for overcoming local minima is described and the simulation results are discussed. Convergence times for XOR, parity and encoder problems are compared with backpropagation, and scaling properties are investigated using the encoder problem.

ALOPEX ALGORITHM

Alopex is a stochastic parallel algorithm which has been used previously, in different forms[4][5][6], in models of visual perception and for solving combinatorial optimization problems[3][4]. Here, the learning process in a neural network is treated as an optimization problem. The objective is to minimize an error measure E, which is a function of the network weights $\{w_{ij}\}$ and training inputs. The version of the algorithm which is used in the present studies, as a learning algorithm in neural networks is as follows:
Consider a neuron i with an interconnection strength w_{ij}, from neuron j. The net input to the neuron i

$$net_i = \sum_j w_{ij} out_j + \theta_i$$

where out_j is the output of neuron j and θ_i is the threshold of neuron i. θ_i is updated like a weight during each iteration. If neuron i contains a sigmoidal transformation, its output is given as,

$$out_i = \frac{1}{1 + exp(-net_i / Q_0)}$$

where Q_0 is the sigmoid gain. Using the Alopex algorithm, during the n^{th} iteration, the weight w_{ij} is updated according to the rule,

$$w_{ij}(n) = w_{ij}(n-1) + \delta_{ij}(n)$$

where $\delta_{ij}(n)$ will have a small positive or negative step of size δ with the following probabilities*:

$$\delta_{ij}(n) = \begin{cases} -\delta \text{ with probability } P_{ij}(n) \\ +\delta \text{ with probability } (1-P_{ij}(n)) \end{cases}$$

The probability $P_{ij}(n)$ is given by the expression:

$$P_{ij}(n) = \frac{1}{1 + exp(-\Delta_{ij}(n)/ T)}$$

where $\Delta_{ij}(n)$ is given by the correlation:

$$\Delta_{ij}(n) = \Delta w_{ij}(n) \,.\, \Delta E(n)$$

$\Delta w_{ij}(n)$ and $\Delta E(n)$ are the changes in the weight w_{ij} and the error measure E over the previous two iterations.
i.e

$$\Delta w_{ij}(n) = [w_{ij}(n-1) - w_{ij}(n-2)] \qquad and$$

$$\Delta E(n) = [E(n-1) - E(n-2)]$$

In the expression for $P_{ij}(n)$, T is a positive 'temperature' that determines the effective randomness in the system. With a non–zero value for T, the algorithm takes biased random walks in the direction of decreasing E. The simulations are started with a large value for the 'temperature' T. Subsequently, the 'temperature' is set equal to the average value of the correlation Δ_{ij} over all weights.

i.e., $$T = |\Delta|$$

---

* During the first two iterations, the weights are randomly perturbed.

This method automatically reduces the 'temperature' when parameters are close to the optima and hence the correlations are small. Towards the end, the step size also can be reduced for precise convergence. The only assumption in the algorithm is that the error measure is a function of the weights and the training set. It makes no assumption about the structure of the network or the actual error measure being used. This allows the use of networks with feedback connections and error measures other than the traditionally used squared error.

SIMULATION RESULTS

Simulations were done to investigate the effectiveness of Alopex as a learning algorithm. The initial set of experiments compare the number of iterations taken by Alopex and backpropagation on three standard learning tasks. Table 1 shows the average number of iterations taken by the two algorithms over 100 trials for learning the XOR, parity and encoder problems. We can see that the average number of iterations taken by Alopex is comparable to those taken by backpropagation. Three different error functions were used to study the performance. It may be pointed out that in Alopex all the weights are updated synchronously and hence with a parallel implementation, the computation time taken per updating would be much less than that of backpropagation.

Table 1: Comparison of BP and Alopex with square error measure

	Backpropagation	Alopex
XOR(2–2–1 network)	3,656	478
Parity(4–4–1 network)	595	353
Encode(4–2–4 network)	2,676	3,092

Alopex has the additional advantage that any error measure could be used for learning. We have used three different error measures. They are:

(i) the traditional squared error measure, $\Sigma(t-o)^2$

(ii) the square root error, $[\Sigma(t-o)^2]^{1/2}$ and

(iii) the information theoretic error measure, $\Sigma\, t.log(t/o)$[7][8]

(t and o denote the target and output vectors, respectively)

Table 2 shows the average number of iterations (over 100 trials) for the three tasks and the number of times the network did not converge after 20,000 trials. For the XOR problem, it can be seen that networks using the information theoretic error measure gets 'stuck' only on 4% of the trials while those using the traditional squared error measure gets 'stuck' on 19% of the trials. This is due to the fact that log error measure creates a smoother error surface with fewer local minima. This is illustrated in Figure 1. Figure 1a and 1b show the error surface around the solution point with respect to the weights w3 & w6 and w4 & w5 (see Fig. 2), for square error. Figure 1c and 1d show corresponding surfaces for log error. We can see that the surface for the log error is much smoother than that for the square one.

Table 2: Comparison of experiments with different error measures

	XOR, 2–2–1 NETWORK		PARITY, 4–4–1 NETWORK		ENCODE, 4–2–4 NETWORK	
Error Measure	Av. no. of iterations	% of times 'stuck'	Av. no. of iterations	% of times 'stuck'	Av. no. of iterations	% of times 'stuck'
Log	1484	4	297	0	2996	0
Square	478	19	353	0	3092	1
Square root	1762	14	332	0	3114	1
B.P(Square)	3656	15	595	0	2676	0

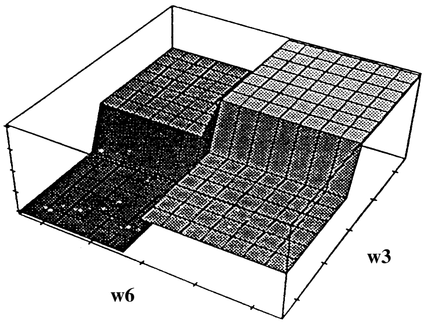

Fig. 1a. Square error surface for XOR network with respect to the weights w3 & w6

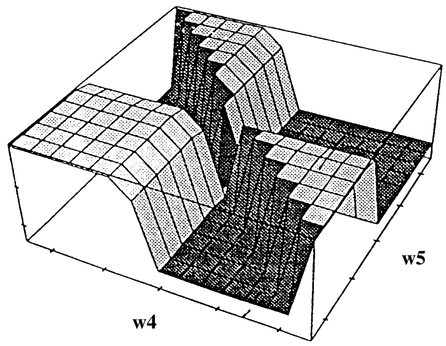

Fig. 1b. Square error surface for XOR network with respect to the weights w4 & w5

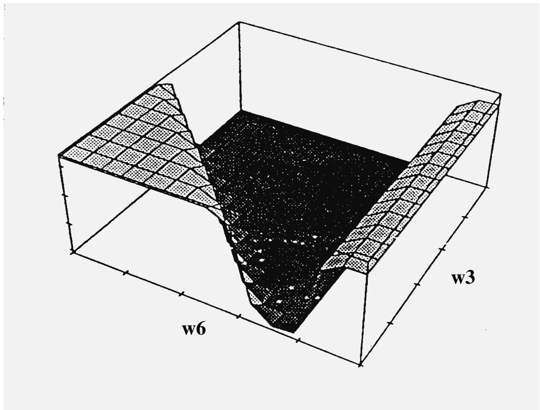

Fig. 1c. Log error surface for XOR network with respect to the weights w3 & w6

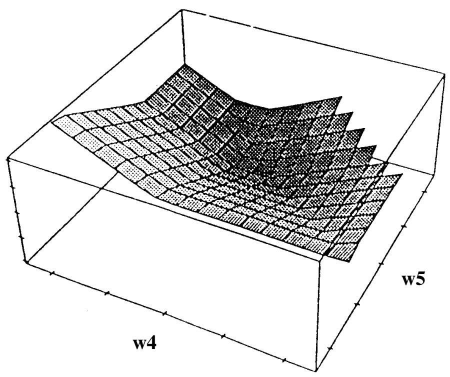

Fig. 1d. Log error surface for XOR network with respect to the weights w4 & w5

Table 3 shows the 'roughness' of the two surfaces at various distances from the solution point. This was done by taking 'rings' at different distances from the solution point and then calculating the average standard deviation of the surface with respect to all the weights. We can see that the error surface for the log error is much smoother (except at the two local minima) compared to the surface for the square error. Figure 3 shows the path taken by the Alopex algorithm to reach the solution point during one of the experiments with log error. It is plotted over the final error surface with respect to the weights w3 & w6. In this case the algorithm has to overcome local minima and the annealing schedule was sufficient for it. It should be noted that the long valley that appears from the starting point to the solution point was not there when the network was at the initial state. The error surface changes during the learning as the weights are also changing.

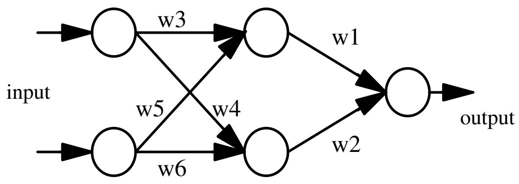

Fig. 2. The 2–2–1 network architecture for XOR problem

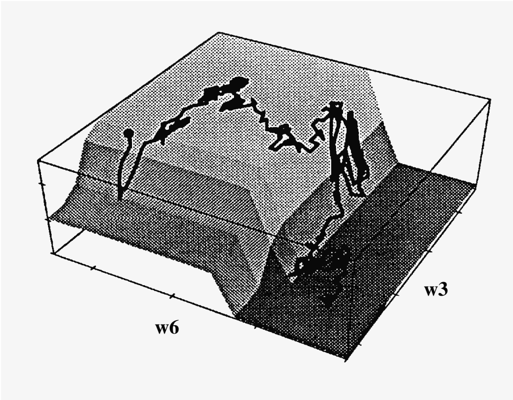

Fig. 3. Path taken by the Alopex algorithm for the XOR problem overcoming local minima

Table 3: Relative roughness of error surfaces

Ring Index	**SQUARE ERROR**	**LOG ERROR**
1	08	00
2	16	00
3	13	00
4	07	00
5	05	17
6	06	00
7	07	17
8	08	00
9	07	00
10	07	00

The scaling properties of the Alopex algorithm were investigated using the encoder problem. Table 4 shows average number of iterations (in 25 trials) taken for 4, 8, 16 and 32 bit encoders using different error functions. For 16 and 32 bit encoders the network did not converge with square and square root errors (the error was observed till 200,000 iterations). In all the trials, the network with square error did not converge in 5 patterns for the 32 bit encoder and in one pattern for the 16 bit encoder, while the log error could converge in all the patterns. The error convergences on a typical trial for log and square errors for encoders of different bit sizes are given in Fig. 4a & 4b. respectively. It could be seen that the error function used affects the network convergence to a large extent.

Table 4: encoder networks with different error measures

Size of input	Average no. of iterations		
	Square error	Square root error	Log error
4 bits	1,734	1,229	1,775
8 bits	6,771	18,662	13,077
16 bits	(no convergence)	(no convergence)	20,280
32 bits	(no convergence)	(no convergence)	75,586

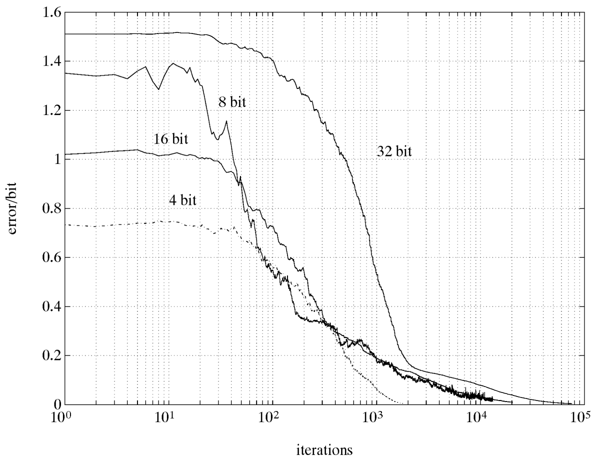

Fig. 4a. Error convergence of the encoder networks with log error

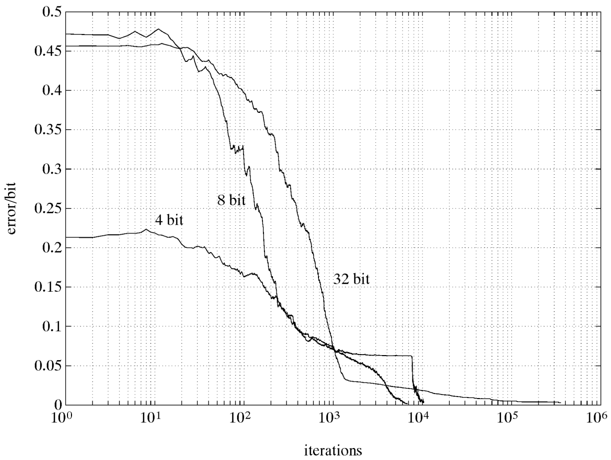

Fig. 4b. Error convergence for the encoder networks with square error

DISCUSSION

We presented a universal learning algorithm for neural networks. The algorithm can be used for learning in networks with any architecture. Preliminary studies where we have used the algorithm for learning in multilayer perceptrons show that the algorithm takes comparable number of iterations as backpropagation. It has a number of advantages over backpropagation learning algorithm such as escaping from local minima, ability to use any error measure and synchronous updating of weights. Also, in this paper we have demonstrated the importance of choosing a proper error function for faster network convergence and better classification.

ACKNOWLEDGEMENT

The authors wish to thank Milind Pandit for helping with some of the simulations.

REFERENCES

[1] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributrd Processing,* vol. 1, MIT Press, 1986.

[2] E. Harth, "Visual perception: a dynamic theory," *Biological Cybernetics***,** vol. 22, 1976, pp. 169–180.

[3] E. Harth, K. P. Unnikrishnan and A. S. Pandya, "The inversion of sensory processing by feedback pathways: a model of visual cognitive functions," *Science*, vol. 237, 1987, pp 187–189.

[4] K. P. Unnikrishnan, A. S. Pandya and E. Harth, "The role of feedback in visual perception,"*Proceedings of IEEE International conference on neural networks*, 1987, pp. 259–267.

[5] K. P. Venugopal and A. S. Pandya, "Alopex algorithm for training multilayer neural networks," *Proceedings of IJCNN '91*, Singapore, Nov. 91.

[6] A. S. Pandya and R. Szabo, "A fast learning algorithm for neural network applications," *Proceedings of IEEE conference on SMC*, Oct. 1991, pp. 1569–1573.

[7] J. J. Hopfield, "Learning algorithms and probability distributions in feedforward and feedback networks," *Proceedings of the Natnl. Acad. of Science. USA,* vol. 84, 1987, pp. 8429–8433.

[8] E. B. Baum and F. Wilczek, "Supervised learning of probability distributions by neural networks," in *Neural information processing systems*, D. Z. Anderson(Ed.), American Institute of Physics, 1988.