Proceedings of the 1988 Annual Meeting of the International Neural Networks Society, Boston, Massachusetts, September 1988

Learning Algorithms for Neural Networks with Ternary Weights

Tzi-Dar Chiueh and Rodney M. Goodman
Department of Electrical Engineering (116-81)
California Institute of Technology
Pasadena, CA 91125, USA

SUMMARY

There has been much research on the VLSI implementation of neural networks. Most implementations use discrete weights with a few levels, e. g. binary or ternary weights, due to the large number of weights to be fabricated on a chip [1][2]. However, the two major learning algorithms for feedback and feedforward networks, i.e. the Hopfield outer product rule and the backpropagation algorithm and its variants, need weights with many levels. In the Hopfield model, the range of weights is [-M, M], where M is the number of fixed points to be memorized. For the backpropagation algorithm and its variants, the weights need to be continuous, since the algorithms are based on the gradient method (steepest descent), whose performance relies on updating the weights by a small amount at each step. So when one wants to program the weights onto a VLSI chip, the weights need to be quantized to two or three levels, which will very often degrade the performance of the network. To circumvent this difficulty, we examine the possibility of learning algorithms which produce discrete weights (we focus on ternary) that will minimize the errors made by the trained network.

Learning can be considered as the process of minimization of the error function over the domain of weights. In the problem that we are investigating, the domain is $\{-1,0,1\}^S$, where S is the number of weights in the network, and the error function we choose is the same as that of the backpropagation algorithm, i.e. sum of squares of errors. We have tried several possible learning algorithms on a three-layer perceptron and a Hopfield type feedback network. They are:

(A) cyclic descent method,
(B) greedy descent method,
(C) simulated annealing,
(D) genetic algorithm,
(E) combination of (A) and (D), genetic cyclic descent method,
(F) combination of (B) and (D), genetic greedy descent method.

Method A and B start with a set of random weights and iterate; usually they get stuck at a local minimum in the error surface, that is, a place where the error function is not zero. Once the algorithm gets trapped at a local minimum, one needs to start with a different set of random weights. However, for simulated annealing and the genetic algorithm, the iteration process goes on until convergence. Method A and B

are not very efficient in that information gained in previous descent cycles (local optima) is not utilized, and the process starts anew with random weights which have no information whatsoever about the previous computation. Since we know that it is highly improbable that a global optimum is in the neighborhood of a local optimum (this would require that the local optimum is surrounded by a deep shallow valley), it is better to start the iteration process with a set of new weights that is not adjacent to previous local optima. Method E and F work in this fashion. At first N sets of weights are generated randomly and then cyclic descent or greedy descent is applied to those N sets of weights independently. After the iteration process terminates (this implies that a local optimum is reached), a genetic algorithm is used to combine those N sets of suboptimal weights and produce N new sets of weights. The algorithm then repeats itself until a global optimum is reached.

We have applied all six methods to the following problems,

(1) learning the 4-input parity problem on three-layer perceptron with 5, 6, 7, and 8 hidden units.

(2) learning the associative memory problem on a Hopfield memory with 10 neurons.

We have found that the genetic greedy descent method (method F), and the genetic cyclic descent method (method E), are the two best algorithms. Also, as the problem gets more difficult (i.e. fewer hidden units or more memory vectors), they become better and better when compared with the other methods. As for simulated annealing and the genetic algorithm, we found that the convergence process is usually very slow. We also applied the backpropagation method to learning the 4-input parity problem, and found that backpropagation is not only much slower, but that after optimally quantizing the resulting weights to 3 levels, about 15% - 50% of the networks do not work.

In the paper we discuss several methods for learning feedforward and feedback neural networks with ternary weights, and propose several new methods based on the genetic algorithm and conventional optimization methods.

References

[1] L. D. Jackel, "Neural Networks Chips" *Proceedings of Workshop on Neural Network Devices and Applications* , Jet Propulsion Lab., Feb. 1987, pp. 43-52.

[2] A. P. Thakoor, et. al., "Binary Synaptic Connections Based on Memory Switching in a-Si:H", *AIP Conference Proceedings No. 151, Neural Networks for Computing* , Snowbird, Utah, 1986, pp. 426-431.