A New Learning Algorithm for Neural Networks with Integer Weights and Quantized Non-linear Activation Functions

Yan Yi[1] and Zhang Hangping[2] and Zhou Bin[3]

Abstract The hardware implementation of neural networks is a fascinating area of research with for reaching applications. However, the real weights and non-linear activation function are not suited for hardware implementation. A new learning algorithm, which trains neural networks with integer weights and excludes derivatives from the training process, is presented in this paper. The performance of this procedure was evaluated by comparing to multi-threshold method and continuous discrete learning method on XOR and function approximation problems, and the simulation results show the new learning method outperforms the other two greatly in convergence and generalization.

1 Introduction

In recent years, Feedforward Neural Networks (FNNs) have been widely used in the areas of pattern recognition, signal processing, time series analysis, and many others. Most of these applications need to be implemented with monolithic integrated circuit or digital signal processor (DSP) in real word. However, the mapping of resultant networks onto fast, compact, and reliable hardware is a difficult task. The problem is that the conventional multilayer FNNs, which have continuous weights, are expensive to store weights and implement calculation in digital hardware. FNNs with integer weights are easier and less expensive to implement in electronics and the storage of the integer weights is much easier to be achieved. The training algorithm in this paper proposes an effective solution for hardware implementation of small-scale FNNs.

There have been some researches focusing on this area. A multiple-threshold method (MTM) has been proposed for generating discrete-weight FNNs

[1] Prof. Yan Yi
Institute of Intelligence &Software, Hangzhou Dianzi University, Hangzhou, China
email: yybjyyj@163.com
[2] Zhang Hangping
email: qingying1226@yahoo.com.cn
[3] Zhou Bin
email: zhoubin02051502@yahoo.com.cn

(CHIEUEH et al., 1988; WOODLAND, 1989). In this simple method, the continuous weights of a fully trained FNN are quantized into discrete valued weights using a nonlinear function (usually a multiple-threshold). The continuous discrete learning method (CDLM) (E.Fiesler et al., 1990) follows a more fruitful strategy. In this method, a trained continuous weight network is quantized. The errors obtained from the discrete network are back-propagated through the continuous network and then trained again. This cycle repeats until the network converges.

Unfortunately, all methods above are unable to discretize the non-linear activation function as a look-up table in training process since they are based on the BP algorithm which needs derivatives of the activation function.

Our main result is a new learning algorithm called optimum descent point learning method (ODPLM), which trains FNNs with integer weights and excluding derivatives of the activation function.

The remainder of this paper is divided into three sections. The first one proposes the new learning algorithm. The second section presents experiments and computer simulation results. The last section presents our conclusion and discussion.

2 Optimum Descent Point Learning Method

ODPLM falls under the category of performing learning, in which the network parameters are adjusted to optimize the performance of the network. The error function *E(X)* is always used to measure the performance of a network quantitatively, and the form of error function is

$$E(X) = \frac{1}{2}\sum_{p=1}^{P} E_p = \frac{1}{2}\sum_{p=1}^{P}\sum_{i=1}^{NL}(d_{pi} - y_{pi})^2 \qquad (1)$$

where X is the matrix of network weights and biases, E_p is the sum of the mean squares of errors associated with the pattern p, d_{pi} is the desired response of an output neuron at the input pattern p, y_{pi} is the response of an output neuron at the input patter p.

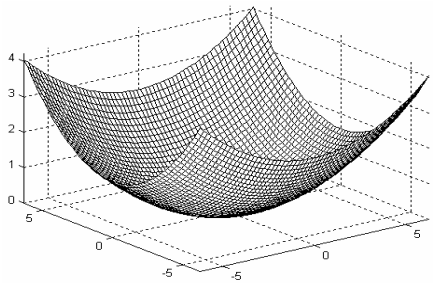

Figure. 1. A performance surface with minimal point (0,0)

The purpose of ODPLM is to search the point with the smallest error function in the parameter

space which has been discretized and confined to integers. The searching process is iterative. We begin from initial guess with integers, X_0, and then search the optimum neighbour which has the smallest error function among all neighbours as the next guess X_{k+1}. A neighbour of X_k is defined as

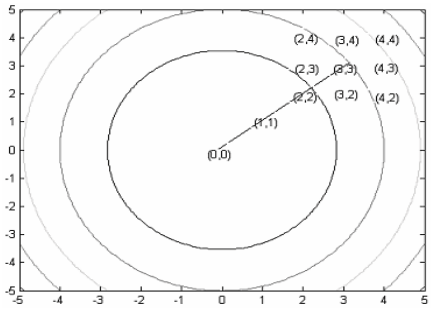

Figure. 2. The contour lines of fig. 1

$$X_k^{\,i} = X_k + p_i \qquad i \in 3^n \qquad (2)$$

where n is the size of matrix X_k, and p_i is a matrix composed of n elements in the set {-1, 0, 1}. All optimum points at each stage construct a path in the discretized space, along which the error function descends steepest.

The size of searching space is 3^n if we want to exhaust all combinatorial possibilities of n elements of p_i. Since the qualities of the final weights are focused on more than the efficiency of a method for off-line training, the exhaustive method is used to search the optimum neighbour for small-scale neural networks, and a further discussion for dealing with large-scale networks is presented in the fourth section.

Figure. 1 shows a performance surface with minimal point (0, 0), and fig. 2 shows the contour lines of fig. 1. When X_k is at the point (3, 3) in fig. 2, points (2, 4), (3, 4), (4, 4), (2, 3), (3, 3), (4, 3), (2, 2), (3, 2), (4, 2) are its neighbours. The point (2, 2) is selected as X_{k+1} since its error function is smallest among eight neighbours. Similarly, the point (1, 1) will be selected as the next optimum point after the point (2, 2), and the process continues until reaches at the minimum point (0, 0).

The activation function can be quantised as a look-up table in training process since OPDLM do not need derivatives. This eliminates the new inaccuracy resulted from the limited size of the look-up table when the fully trained network with integer weights is implemented by hardware.

```
Algorithm 1: <Optimum descent point learning method>
Step1:
      Quantise continuous weights space.
Step2:
      Quantise non-linear activation functions as a look-up
      table.
Step3:
       Intialize the FNNs with integer weights denoted by X_k.
While (e > E_allowed)
    Calculate the error functions of all neighbours of X_k
    with  exhaustive search
    choose the neighbour with the minimal error function as X_k+1
    if(E(X_k+1) > E(X_k))
```

```
        the process has been stuck in local minimum, therefore it
        should be terminated and make a fresh start with a new set
        of initial weights.
        Break
      End if
  End While
```

3. Functionality Tests

The classical learning test problem – the approximation of a sine curve function – has been used for testing the functionality. The reported parameters in the Tables for simulations that have reached solution are: *min* the minimum number of iterations, *mean* the mean number of iterations, *max* the maximum number of iterations, *time* the mean time of successful training processes, *succ*. simulations succeeded out of ten.

Table 1. Software simulation results on the approximation of the function f(x)

	MTM	CDLM	ODPLM
min	3185	8623	6
max	8461	23824	14
mean	6807	11694	11
time	4	20	420
succ.	0%	40%	100%

Let's assume that we want to approximate the following functions:

$$f(x) = e^{-x}\sin(2\pi x),$$

and the training set is obtained by sampling the function at the points x = 0, 0.1, 0.2, …, 0.9, 1. (There are total of 11 input/target pairs.) To approximate this function we will use a 1-4-1 network, where the activation function for the first layer is log-sigmoid and the activation function for the second layer is linear. The allowed error function is 0.01; the range of initial weights is (-10, 10); the learning rate is 0.1.

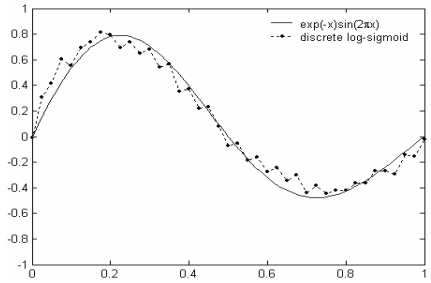

Figure 3 Generalization of the network trained by ODPLM on the approximation of f(x)

The convergence performance of MTM, CDLM, and ODPLM on function approximation problems is shown in table 1. From the tables, we can see that ODPLM outperforms MTM and CDLM greatly in the successful training number out of ten. In addition, the epochs of ODPLM for each process are far less than

the other two. However, ODPLM requires a long computational time for each epoch, so the total running time is longer than the others.

Figure 3 presents the generalization of the best networks trained by ODPLM, in the figure the point-dotted line represents the responses of neural networks with discrete sigmoid function which has been quantised as a 50-size look-up table.

4. Conclusion and Discussion

A new learning algorithm ODPLM is presented in this paper, which trains the FNNs with integer weights and quantized activation functions. In the algorithm, non-linear activation functions have been already quantized in training process, therefore the inaccuracy will not increase in hardware implementation. The simulation results show the new learning algorithm works better than the CDLM and the multi-threshold method in terms of convergence and generalization.

References

1. A.H. Khan and E.L. Hines (1994) Integer-weight neural nets. ELECTRONICS LETTERS, 21st July, vol.30 No.15
2. CHIEUEH, T.D., and GOODMAN, R.M. (1988) Learning algorithms for neural networks with ternary weights. First Annual Meeting of INNS. Boston, MA. Pages 166.
3. E.Fiesler, A.Choudry, H.J.Caulfield. (1990). A weight discretization paradigm for optical neural networks. Proceedings of the International Congress on Optical Science and Engineering. Bellingham, Washington, U.S.A.: The International Society for Optical Engineering Proceedings, volume SPIE-1281, pages 164-173.
4. Martin T.Hagan, Howard B.Demuth(1996) Neural Network Design, PWS Publishing Company
5. MARCHESI.M., BENVENUTO, N., ORLANDI, G., PIAZZI. F., and UNCINI (1990), A. Design of multi-layer neural networks with power-of two weights. IEEE ISCS, New Orleans, pp.2951-2954
6. RUMELHART, D.E., HINTON, G.E., and WILLIAMS, R.J. (1986) Learning internal representation by error backpropagation. Parallel distributed processing: Explorations in the microstructure of cognition. pp. 318-362.
7. T.Lundin, E.Fiesler and P.Moerland (1996) Connectionist Quantization Functions. The Proceedings of the 1996 SIPAR-Workshop on Parallel and Distributed Computing Geneve, Switzerland.
8. V.P.Plagianakos, M.N.Vrahatis. (2000). Training Neural Networks with Threshold Activation Functions and constrained Integer Weights. Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference. Volume 5, pages 161 – 166
9. VON LEHMEN, A., PAEK, E.G., LIAO, P.F., MARRAKCHI,.A, and PATEL, J.s. (1988). Factors influencing learning by backpropagation. Proc.Int. Conf. Neural Networks, San Diego, USA, pp. 335-341