

Enhanced Robustness of Multilayer Perceptron Training

Walter H. Delashmit
Lockheed Martin
Missiles and Fire Control
Dallas, TX 75265

Michael T. Manry
University of Texas at Arlington
Electrical Engineering Department
Arlington, TX 76011

Abstract

Due to the chaotic nature of multilayer perceptron training, training error usually fails to be a monotonically non-increasing function of the number of hidden units. An initialization and training methodology is developed to significantly increase the probability that the training error is monotonically non-increasing. First a structured initialization generates the random weights in a particular order. Second, larger networks are initialized using weights from smaller trained networks. Lastly, the required number of iterations is calculated as a function of network size.

1: Introduction and statement of the problem

When different size (different numbers of hidden units) multilayer perceptrons (MLPs) are initialized, the training error usually fails to be a monotonically non-increasing function of the number of hidden units (N_h) as seen in Figure 1. This occurs because (1) networks of different sizes have different basis functions (i.e., the inputs, the constant “one” used for specifying the thresholds and the outputs of the hidden units) and (2) training is chaotic.

Usually, investigators attack the problem by designing many networks and saving the one with minimum mean square error (MSE). Alternate approaches adjust the initial seeds to control the values of the assigned random numbers. This attempts to avoid the problem by finding a set of seeds with decreasing MSE for increasing number of hidden units.

In this paper, we describe non ad hoc approaches which mitigate chaotic MLP training. In Section 2, we develop a theoretical approach for analyzing the problem. Section 3 presents a solution based on structured weight initialization. This technique is enhanced in Section 4 using structured recursive weight initialization. The technique is further enhanced by improving the net control processing as discussed in Section 5. The final

enhancement performs training using the previous techniques with a variable number of iterations based on network complexity as presented in Section 6. A summary is presented in Section 7. All techniques are illustrated with results from the processing of an actual dataset and the weight initialization techniques are also graphically described.

2: Development of the theory

When a sequence of different size MLPs are trained, larger ones must have smaller training errors to be useful. Alternately, a larger network with a larger training error is not useful. Let $E_i(N_h)$ denote the training error for a network having N_h hidden units which has been initialized using the i th random number seed. Since useless networks occur frequently, investigators often design many networks of a given size in order to obtain monotonically non-increasing $E_{av}(N_h)$ versus N_h curves where $E_{av}(N_h)$ denote the average $E_i(N_h)$ as

$$E_{av}(N_h) = \frac{1}{N_s} \sum_{i=1}^{N_s} E_i(N_h) \quad (1)$$

Monotonicity can be investigated by determining a number, B , such that

$$P(E_{av}(N_h+1) > E_{av}(N_h)) \leq B \quad (2)$$

Let the variance of $E_{av}(N_h)$ be approximated by

$$\sigma_{E_{av}}^2(N_h) = \frac{\text{var}(E_i(N_h))}{N_s} \quad (3)$$

From Figure 2, fitting a multi-order linear regression to the MSE curve, $E_{av}(N_h)$, yields an average MSE curve, $m_{av}(N_h)$. Thus (2) can be expressed by

$$P(E_{av}(N_h+1) - m_{av}(N_h+1) > E_{av}(N_h) - m_{av}(N_h+1)) \leq B \quad (4)$$

For the second term of the left side of the inequality expressed in (4), we can add and subtract $m_{av}(N_h)$ resulting in an equation of the form

$$P(x-y > \epsilon) \quad (5)$$

where

$$\begin{aligned} x &= E_{av}(N_h+1) - m_{av}(N_h+1) \\ y &= E_{av}(N_h) - m_{av}(N_h) \\ \varepsilon &= m_{av}(N_h) - m_{av}(N_h+1) \end{aligned}$$

with x and y being zero mean random variables. Expressing this as

$$P(|x'| > \varepsilon) = P(x' > \varepsilon) + P(-x' > \varepsilon) \approx 2P(x' > \varepsilon) \quad (6)$$

and using the Chebyshev inequality results in

$$P(x' > \varepsilon) = \frac{1}{2} \cdot P(|x'| > \varepsilon) \leq \frac{1}{2} \frac{\sigma_x^2}{\varepsilon^2} \quad (7)$$

where σ_x^2 corresponds to $\sigma_{E_{av}}^2(h)$. From (5) with $E(\cdot)$ denoting the expected value and $\text{var}(\cdot)$ denoting the variance,

$$\text{var}(x-y) = E(x^2) + E(y^2) \quad (8)$$

where $E^2(x-y) = 0$ and $E(xy) = 0$ since x and y are uncorrelated for randomly initialized networks. From (1) - (8), this results in

$$\begin{aligned} &P(E_{av}(N_h+1) > E_{av}(N_h)) \\ &\leq \frac{\text{var}(E_i(N_h+1)) + \text{var}(E_i(N_h))}{2 \cdot N_s \cdot (m_{av}(N_h) - m_{av}(N_h+1))^2} \end{aligned} \quad (9)$$

This shows a functional relationship for the probability of designing useless MLPs (i.e., $P(E_{av}(N_h+1) > E_{av}(N_h))$). This also leads to the need to develop more effective weight initialization techniques to reduce this dependence on the number of networks that need to be designed. Results using this error bound are shown in Figure 3 where the largest value of the bound, B , corresponds to 5 hidden units. This is the value in Figure 1 where the larger network has a larger MSE than the smaller network. Alternately, (9) can be used to determine how many seeds are required to ensure that the larger network will have a desired probability (e.g., 0.02) of having a MSE that is smaller than that of the smaller network. This is expressed by

$$P(E_{av}(N_h) > E_{av}(N_h+1)) \geq 0.98 \quad (10)$$

These results are shown in Figure 4 where the largest number of random seeds is required for 5 hidden units which is consistent with the results shown in Figures 1 and 3.

3: Structured weight initialization

As a first approach, a structured initialization is developed where different size networks have the same

initial basis functions, which are the inputs and hidden units outputs. The j th hidden unit's output, $O_p(j)$, is expressed as

$$O_p(j) = f(\text{net}_p(j)) = \frac{1}{1 + e^{-\text{net}_p(j)}} \quad (11)$$

$$\text{net}_p(j) = \sum_{k=1}^{N+1} w_{hi}(j,k) \cdot x_p(k) \quad 1 \leq j \leq N_h \quad (12)$$

with $w_{hi}(j,k)$ the input-to-hidden unit weights. In this notation, hidden and output unit thresholds are incorporated by augmenting the input vector, assigning the value of one to the input $x_p(N+1)$. The i th output for the p th training pattern, $y_p(i)$, is

$$y_p(i) = \sum_{k=1}^{N+1} w_{oi}(i,k) \cdot x_p(k) + \sum_{j=1}^{N_h} w_{oh}(i,j) \cdot O_p(j) \quad (13)$$

with x_p the input for the p th training pattern, N the number of inputs, N_h the number of hidden units, $w_{oi}(i,k)$ the input-to-output weights and $w_{oh}(i,j)$ the hidden units to output weights. In (13), i varies from 1 to M where M denotes the number of outputs.

Assume that several networks are to be designed each with the same initial random number seed but a different value of N_h . For our approach we first specify all input-to-output weights using the same initial seeds and then all output thresholds are specified. For the networks being developed, the number of inputs and outputs are fixed. Then the weights from the inputs to the first hidden unit are specified and the weights from this hidden unit to the outputs are specified. This continues for subsequent hidden units until all initial weights are specified (Figure 5). This technique reduces some problems associated with inconsistent initialization (Figure 6) by ensuring that the initial basis functions are the same as the MLP complexity increases with additional hidden units.

4: Structured recursive weight initialization

When a series of different size networks is being designed, the final network of a given size can supply most of the initial weights for the next larger network. The additional hidden units in the larger network are initialized as previously indicated (Figure 5). This weight initialization procedure extension is shown in Figure 7 where the weights for the prior set of hidden units are replaced by the final weights that were obtained from the previous training. This is analogous to cascade correlation [1] except we train all hidden units, not just the additional ones. This ensures consistent initial basis functions and makes maximum use of the training results for smaller networks.

5: Enhanced net control processing

After using structured weight initialization, further MLP improvement can be obtained by optimizing net control processing [2]. Net control is used to ensure that the hidden unit activations in the initial network are not saturated. This is implemented by setting the j th hidden unit net function mean and standard deviation to desired values, $m_{hd}(j)$ and $\sigma_{hd}(j)$, respectively. This requires determining the mean $m_h(j)$ and the standard deviation $\sigma_h(j)$ of the j th hidden unit net function. For the j th hidden unit net function ($1 \leq j \leq N_h$) and the i th-augmented input ($1 \leq i \leq N+1$), the initial input-to-hidden unit weights are adjusted as

$$w(j,i) \leftarrow \frac{w(j,i) \cdot \sigma_{hd}(j)}{\sigma_h(j)} \quad (14)$$

and the hidden unit thresholds are adjusted using

$$w(j,N+1) \leftarrow w(j,N+1) + m_{hd}(j) - \frac{m_h(j) \cdot \sigma_{hd}(j)}{\sigma_h(j)} \quad (15)$$

The desired values, $m_{hd}(j)$ and $\sigma_{hd}(j)$ are usually fixed numbers. Alternate implementations specify the means and standard deviations as bounded random values. This produces improved training with a more monotonically non-increasing MSE curve (Figure 8). These results indicate that the best performance is obtained by using a constant desired standard deviation and a random desired mean for the hidden unit net functions. However, since the MSE curve is still not always monotonically non-increasing, additional improvements are required.

6: Specification of required number of iterations

Adding hidden units increases the complexity of the MLP since the number of weights and thresholds increases. If conjugate gradient [3] training is used, the number of iterations should also increase appropriately since in conjugate gradient training the number of iterations should equal the number of unknowns. Techniques for determining how to increase the number of iterations have not previously been proposed. Often a fixed number of iterations is used as the number of hidden units are increased.

The number of iterations required to train the MLP can be determined as a function of N , M , N_h and the number

of iterations for the minimal MLP (one hidden unit). The number of iterations (N_{iter}) can be expressed as

$$N_{iter}(N_h) = f(N_h) N_{iter}(1) \quad (16)$$

$$= \frac{N_h(N+M+1) + M(N+1)}{MN + 2M + N + 1} \cdot N_{iter}(1) \quad (17)$$

where $N_{iter}(1)$ is the selected number of iterations for a MLP with a single hidden unit. This shows that additional iterations are required as the number of hidden units increase.

Using the procedure for structured recursive weight initialization (Figure 7) and the specified number of iterations from (17) the curves of Figure 9 were generated. In this figure we see the performance improvement in obtaining a monotonically non-increasing training error. These results show the training improvement from (1) the initial structured weights followed by (2) recursive structured weights with a fixed number of iterations and then (3) recursive structured weights with the variable number of iterations specified above.

7: Summary

Several techniques have been described which significantly increase the probability that the training error is a monotonically non-increasing function of network size. Structured weight initialization, enhanced net control processing, recursive structured weights initialization and a variable number of iterations significantly reduce the chaotic aspects of MLP training.

8. Acknowledgements

This work was supported in part by the Advanced Technology Program of the State of Texas under grant 003656-0129-2002.

References

1. Fahlman, S. E. and C. Lebiere, "The Cascade Correlation Learning Architecture," *Report CMU-CS-90-100*, August 1991.
2. J. Olvera, "Monomial Activation Functions for the Multilayer Perceptron," *M.S. Thesis*, University of Texas at Arlington, 1992.
3. Chen, H. H., M. T. Manry, H. Chandrasekaran, "A neural network training algorithm utilizing multiple sets of linear equations," *Neurocomputing*, Vol. 25, No. 1-3, pp. 55-72, April 1999.

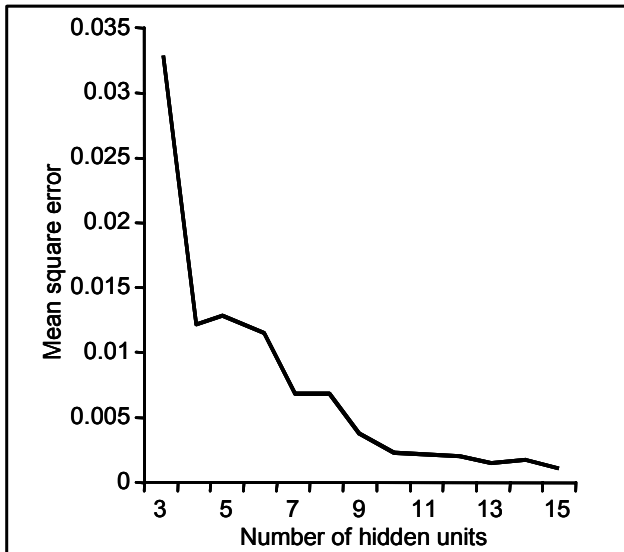


Figure 1. Training error as a function of number of hidden units.

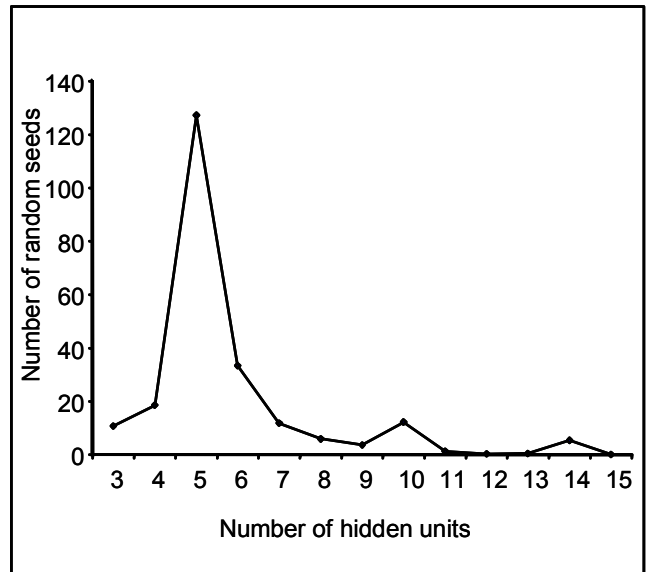


Figure 4. Number of random seeds for monotonically non-increasing error curve.

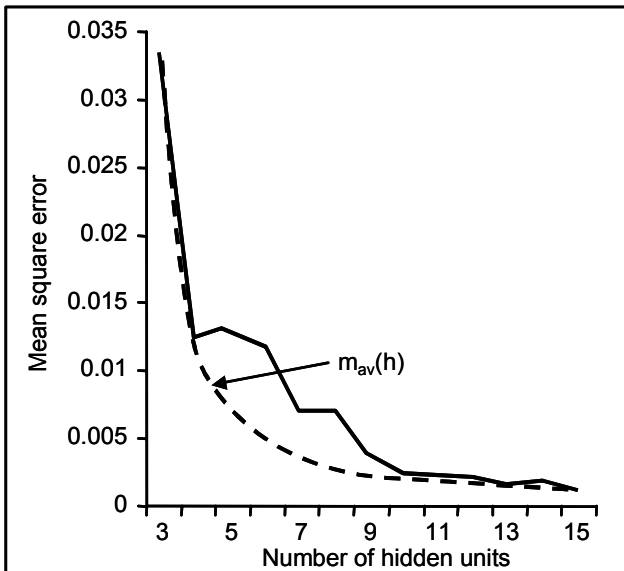


Figure 2. Performance bound technique.

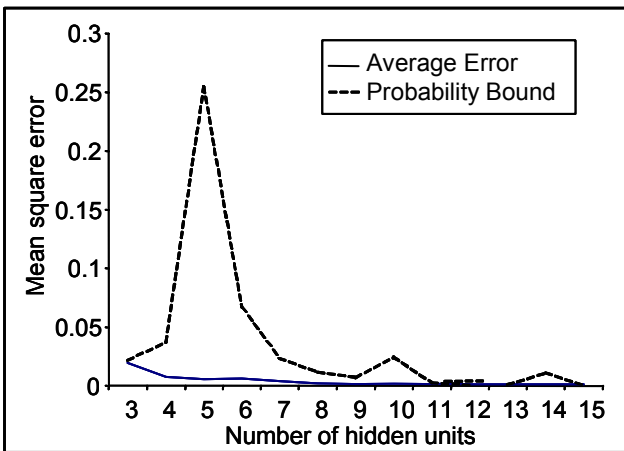


Figure 3. Error bound results

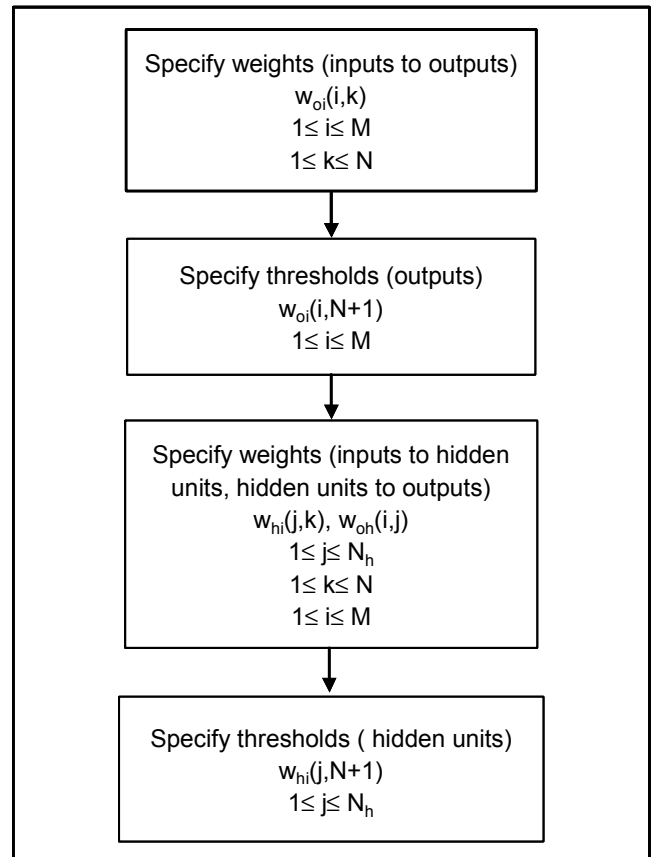


Figure 5. Structured weight initialization technique.

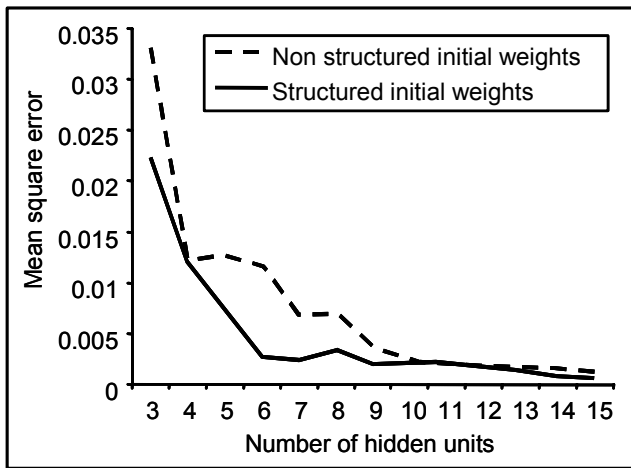


Figure 6. Benefits of using structured weight initialization.

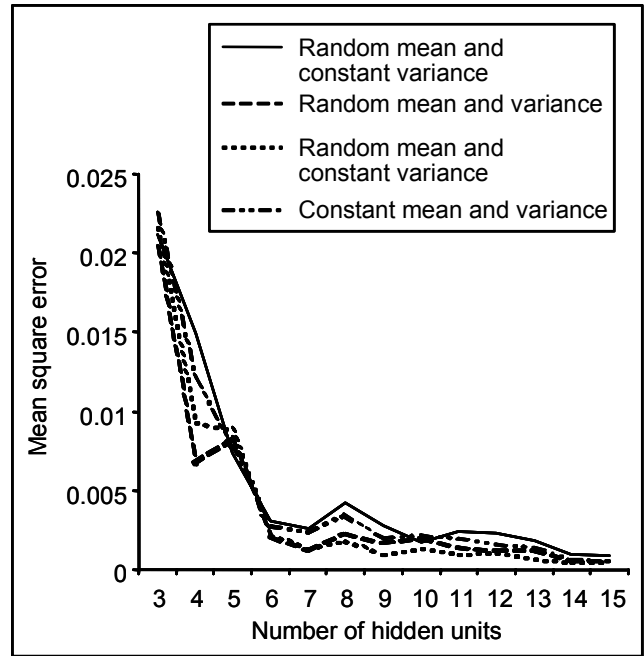


Figure 8. Net control optimization with structured weight initialization.

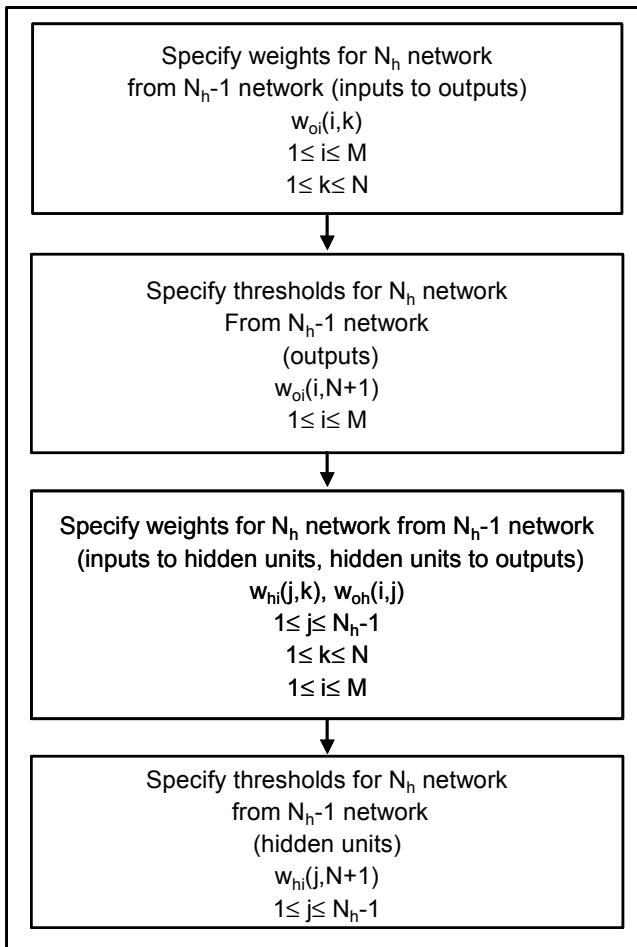


Figure 7. Recursive structured weight initialization.

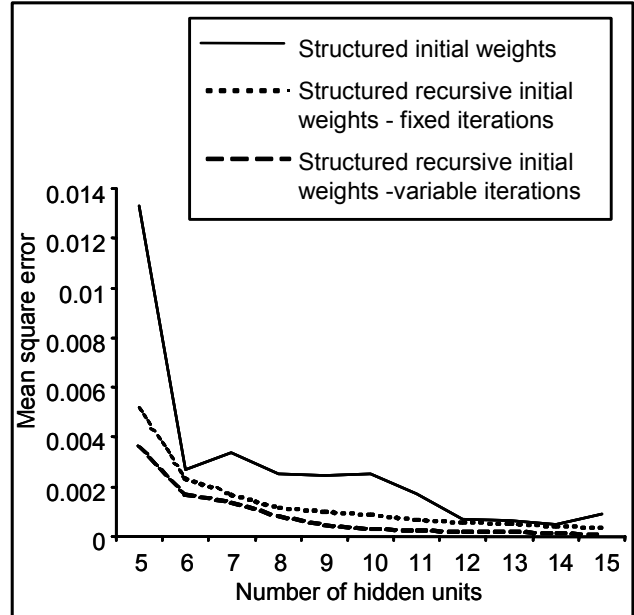


Figure 9. Comparison of three initialization schemes.