On Weight Initialization in Multilayer Perceptron Networks

Mikko Lehtokangas
Tampere University of Technology
Signal Processing Laboratory
P.O.Box 553, FIN-33101 Tampere, Finland
Phone: +358-3-3653881, Fax: +358-3-3653095
email: mikkol@cs.tut.fi

ABSTRACT: In recent years many studies have demonstrated that the used weight initialization strategy has significant impact in the training process of multilayer perceptron networks. However, even nowadays a common approach is to use small random values for weight initialization. In many difficult classification problems this simple strategy is not adequate. This has led to another widely used strategy, in which random initial values are used with the constraint that the initial weight distribution should be such that sigmoidal units operate within the active region in some well defined manner. There are numerous variations of this approach differing in the manner on how the active region is used. Another computationally simple approach is to utilize the concept of candidate weight initialization. In addition, there are numerous other methods, but they often involve extensive statistical and/or geometrical analysis of the data. Even though a lot of work have been done in this area, very little work have been done on comparing different approaches. In this study we compare three different weight initialization approaches. The first is the basic initialization with small random values. In the second we utilize the constraint that initially the sigmoidal units should operate in the active region in a well defined manner. Thirdly we consider the candidate weight initialization approach. Empirical simulations demonstrate that the candidate weight initialization approach is the most promising method for difficult classification problems while the basic initialization with small random values is the worst (as expected).

KEYWORDS: weight initialization, multilayer perceptron network, classification

INTRODUCTION

Multilayer perceptron networks are powerful models for solving nonlinear mapping problems, Haykin (1999). Their training is usually done with gradient descent based optimization routines, Fletcher (1990). The training can be viewed as a nonlinear optimization problem in which the goal is to find a set of network weights that minimize a cost function. The cost function which is usually a function of the network mapping errors describes a surface in the weight space, often referred to as the error surface. Training algorithms can be viewed as methods for searching minimum of this surface. The complexity of the search is governed by the nature of the surface. For example, error surfaces for multilayer perceptron networks can have many flat regions where learning is slow, and long narrow "canyons" that are flat in one direction and steep in the other directions. Thus, in realistic cases, the large number of very flat and very steep parts of the surface make it difficult to search the surface efficiently using gradient based training routines. In addition, the cost function is characterized by a large number of local minima with values in the vicinity of the best or global minimum.

Because of the complexity of the search space, the main drawbacks of gradient techniques are that they are slow and unreliable in convergence. Major reasons for poor training performance are the problem of determining optimal steps (i.e. size and direction in the weight space in consecutive iterations), the problem of network size, and weight initialization. The improved training algorithms (more optimal steps) and optimal network size do not alone guarantee adequate convergence because of the initialization problem. When the initial weight values are poor the training speed is bound to get slower even if improved training algorithms and optimal network size are used. In the worst case the network may converge to a poor local optimum.

Many recent studies have shown that a proper weight initialization can facilitate the training process significantly, Denoeux and Lengelle (1993), Drago and Ridella (1992), Wessels and Barnard (1992). The conventionally employed rule for weight initialization is to use small random values. The motivation for this is that large absolute values of weights can cause network nodes to be highly active or inactive (saturated) for all training samples, and thus insensitive to the training process. However, too small values can also slow down the training. The randomness is introduced for

preventing network nodes from adopting similar functions (this is also called symmetry breaking). This conventional random initialization can be adequate in many simple problems. However, in many difficult classification problems this simple strategy is not adequate. This has led to another widely used strategy, in which random initial values are used with the constraint that the initial weight distribution should be such that sigmoidal units operate within the active region in some well defined manner, Nquyen and Widrow (1990). There are numerous variations of this approach differing in the manner on how the active region is used, Thimm and Fiesler (1997). Another computationally simple approach is to utilize the concept of candidate weight initialization, Lehtokangas et al. (1998). In addition, there are numerous other methods, but they often involve extensive statistical and/or geometrical analysis of the data. Even though a lot of work have been done in this area, very little work have been done on comparing different approaches. In this study we compare three different weight initialization approaches. The first is the basic initialization with small random values. In the second we utilize the constraint that initially the sigmoidal units should operate in the active region in a well defined manner. Thirdly we consider the candidate weight initialization approach. Next the two latter approaches are described with more detail to facilitate the comparison.

REFERENCE PATTERN INITIALIZATION

We have recently proposed a so called reference pattern (RP) weight initialization scheme, Lehtokangas and Saarinen (1998). It belongs to the category where the constraint that initially the sigmoidal units should operate in the active region in a well defined manner, is utilized. Namely, with RP-initialization the aim is to set the initial weight values to be such that inputs to network nodes are within the active region (in active region the derivative of the hidden unit activation function is non-zero; otherwise the hidden unit is saturated). In addition the discriminant functions formed by the hidden nodes are evenly distributed to the input space according to the reference patterns. For the reader's convenience, with one hidden layer and hyperbolic tangent activation functions, the RP-initialization can be described by the following steps:

step 1. Let the number of hidden units to be initialized be q. Create q reference input patterns from the training data for example by clustering. Create also the corresponding desired reference output patterns. Different methods for obtaining the reference patterns have been discussed in Lehtokangas and Saarinen (1998).

step 2. Initialize the weights w_{ij} (these are the connections between input and hidden layers) by finding the least mean square error optimal solution for the following modelling problem. Let us define a linear model as

$$\boldsymbol{h}_j = \boldsymbol{W}\boldsymbol{c}_j , \qquad j = 1, \ldots, q \tag{1}$$

in which $\boldsymbol{h}_j$ is a qx1 vector containing the outputs of the linear model, $\boldsymbol{W}$ is a qx(p+1) matrix containing the weights w_{ij} to be initialized (including the biases), and $\boldsymbol{c}_j$ is a (p+1)x1 vector consisting of the j:th reference input pattern created in step 1 and a unit input to bias weights (the first element). The desired model output $\boldsymbol{d}_j$ is a qx1 vector in which the j:th element has the value 1 and all the other elements have the value -1. The desired output values have been selected so that they are well within the active region of the activation function which was the hyperbolic tangent function in this work.

step 3. After $\boldsymbol{W}$ has been initialized, feed the reference input patterns through the hidden layer. Then initialize the weights v_{jk} (these are the connections between hidden and output layers) by finding the least mean square error optimal solution for another linear modelling problem defined as

$$\boldsymbol{o}_j = \boldsymbol{V}\boldsymbol{a}_j , \qquad j = 1, \ldots, q \tag{2}$$

in which $\boldsymbol{o}_j$ is a rx1 vector containing the outputs of the linear model, $\boldsymbol{V}$ is a rx(q+1) matrix containing the weights v_{jk} to be initialized (including the biases), and $\boldsymbol{a}_j$ is a (q+1)x1 vector consisting of hidden layer outputs for the j:th reference input pattern created in step 1 and a unit input to bias weights (the first element). The desired model output $\boldsymbol{d}_j$ is now a rx1 vector which is the j:th reference output pattern created in step 1.

This concludes the initialization procedure. More detailed description can be found in Lehtokangas and Saarinen (1998). As can be seen from the above, the proposed RP-initialization scheme involves only creating reference patterns and solving two relatively small linear optimization problems given in equations (1) and (2). Considering the reference pattern creation, it should be emphasized that the only restriction here is that the selected reference patterns should be representative of the training data. By this we mean that reference patterns should only indicate where the training data lies in the input space. The rationale of the RP-initialization approach can be explained as follows. For each hidden unit we attempt to assign a representative reference pattern. That is, when a reference pattern is fed to the network we desire that the input to the corresponding hidden unit would be one and minus one for all the other units (in case of hyperbolic tangent

activation function). First of all, this causes that the inputs to the hidden units are within the active region for all the reference patterns. Because the reference patterns are supposed to be representative of the training data, it follows that the inputs to the hidden units are within the active region also for all or most of the training data. Secondly, each hidden unit attempts to become sensitive to one of the reference patterns. Therefore if the reference patterns are representative (the input space area where training data is located is evenly covered) then the weights become initialized so that the discriminant functions formed by hidden nodes are evenly distributed to the relevant region of input space where the training data lies. This could be called as deterministic symmetry breaking.

MAXIMUM COVARIANCE INITIALIZATION

The maximum covariance (MC) initialization method, Lehtokangas et al. (1996), is an efficient example of candidate weight initialization approach. There the underlying idea is based on the stepwise regression which is an old statistical technique for selecting the best regression equation, Draper and Smith (1981). First a large number of candidate hidden units (or candidate regressors) are created by initializing their weight with random values. Then the desired number of hidden units is selected among the candidates by the use of the MC criterion. A practical MC-initialization procedure can be described by the following steps:

step 1. Create Q candidate hidden units ($Q>>q$; q denotes the desired number of hidden units) by initializing the weights feeding them with random values.

step 2. Do not connect the candidate units to the output unit yet. The only connection feeding the output unit at this time is the bias weight. Set the value of the bias weight to be such that the network output is the mean of the desired output sequence.

step 3. Calculate the covariance for each of the candidate unit from the equation

$$C_k = \frac{1}{n}\sum_{l=1}^{n} (y_{k,l} - \bar{y}_k)\,(\varepsilon_l - \bar{\varepsilon}) \ , \tag{3}$$

k=1,...,Q. In above $y_{k,l}$ is the output of the k:th candidate hidden unit for the l:th example. The parameter $\bar{y}_k$ is the mean of the k:th hidden unit outputs, ε_l is the output error at the network output and $\bar{\varepsilon}$ is the mean of the output errors.

step 4. Find the maximum absolute covariance $|C_k|$ and connect the corresponding hidden unit to the output unit. Decrement the number of candidate hidden units Q by one.

step 5. Optimize the currently existing weights that feed the output unit with linear regression. Note that the number of these weights is increased by one every time a new candidate unit is connected to the output unit, and due to the optimization the output error changes each time.

step 6. If q candidate units have been connected to the output unit then quit the initialization phase. Otherwise go back to step 3.

The above maximum covariance method can be seen as a deterministic method for choosing the best initializations (according to the covariance criterion) from a large set of random initializations. Compared to the reference pattern initialization in the previous section, the maximum covariance initialization obviously requires more computation since the value of Q can be very large, and linear regression (step 5.) has to be performed q times. In certain type of problems this difference in computational cost can become critical feature when considering the usefulness of the methods.

EXPERIMENTS

In this section the performances of the basic random, reference pattern and maximum covariance initialization methods are empirically investigated in the context of multilayer perceptron network training. We have used two benchmark problems. The first problem is the two spirals problem, Fahlman and Lebiere (1990), where the goal is to separate two interlocked spirals. This problem has been found relatively hard to solve with multilayer perceptron networks. The second problem is the channel equalization problem described in Kantsila et al. (to appear). There the goal is to equalize a burst of bits transmitted through a fixed communication channel having 20db signal-to-noise ratio. The optimization procedure we used for actual training was the RPROP algorithm, Riedmiller and Braun (1993). Each of the simulations were repeated ten times.

According to some experimentation, in the two spirals problem the number of hidden units was set to 40 (161 network parameters). In terms of number of parameters, this is similar sized network as used in previous studies, Fahlman and

Lebiere (1990). In the equalization problem the number of hidden units was set to 7. The number of candidates used in the MC-initialization was set to be ten times the number of hidden units. With these settings the learning curves after different initializations are depicted in Figs. 1-3. Obviously, with basic random initialization there are considerable convergence difficulties. The RP-initialization provides significant improvements, but the MC-initialization is in its own class. Considering computational costs, in the two spirals problem the RP- and MC-initialization correspond to 14 and 150 epochs of RPROP training, respectively. Hence MC-initialization requires over ten times more computation, but 150 epochs is still quite low value considering the actual learning curves in Figs. 1a, 2a and 3a. Similarly, in the equalization problem the RP- and MC-initialization correspond to 7 and 17 epochs of RPROP training, respectively. As a result, the MC-initialization method seem to be the most promising one. However, one should keep in mind that in larger problems the MC-initialization may become computationally infeasible, whereas RP-initialization could still provide viable alternative due to its simplicity. In any case, the results once more demonstrate that in difficult problems one should use more advanced initialization techniques than just the usual initialization with small random values.

CONCLUSIONS

Weight initialization in the context of multilayer perceptron network training was considered. First we described the problem of weight initialization. Then two weight initialization approaches were briefly described. These were the reference pattern and maximum covariance initialization. After that, we empirically investigated the performances of these approaches. Comparison with the usual initialization with small random values was also included. The obtained results demonstrated that in difficult problems advanced initialization methods can considerable improve the convergence. Especially the maximum covariance initialization was found to provide the greatest potential for the initialization problem. However, in large scale problems the reference pattern approach may provide to be better alternative due to its computational simplicity.

ACKNOWLEDGEMENTS

This work has been supported by the Academy of Finland.

REFERENCES

Denoeux T. and Lengelle R., 1993, "Initializing back propagation networks with prototypes," Neural Networks, vol. 6, pp. 351-363.

Drago G. and Ridella S., 1992, "Statistically controlled activation weight initialization (SCAWI)," IEEE Transactions on Neural Networks, vol. 3, no. 4, pp. 627-631.

Draper N. and Smith H., 1981, Applied regression analysis, 2nd ed., John Wiley & Sons Inc.

Fahlman S. and Lebiere C., 1990, "The cascade-correlation learning architecture," In D. Touretzky (Ed.), Advances in Neural Information Processing Systems 2, San Mateo, CA, Morgan Kaufman, pp. 524-532.

Fletcher R., 1990, Practical methods of optimization, 2nd ed., Wiley, Chichester.

Haykin S., 1999, Neural networks: a comprehensive foundation, 2nd ed., Prentice Hall, New Jersey.

Kantsila A., Lehtokangas M. and Saarinen J., to appear, “Burst adaptive equalization of binary data,” Journal of Intelligent Systems.

Lehtokangas M., Korpisaari P. and Kaski K., 1996, "Maximum covariance method for weight initialization of multilayer perceptron network," Proceedings of the European Symposium on Artificial Neural Networks, ESANN'96, pp. 243-248.

Lehtokangas M. and Saarinen J., 1998, “Weight initialization with reference patterns,” Neurocomputing, vol. 20, no. 1-3, pp. 265-278.

Lehtokangas M., Salmela P., Saarinen J. and Kaski K., 1998, “Weight initialization techniques,” In C. Leondes (Ed.), Algorithms and Architectures, Series on Neural Network Systems Techniques and Applications, Academic Press, San Diego.

Nquyen D. and Widrow B., 1990, "Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights," Proceedings of International Joint Conference of Neural Networks, vol. 3, pp. 21-26.

Riedmiller M. and Braun H., 1993, “A direct adaptive method for faster backpropagation learning: the RPROP algorithm,” *Proceedings of IEEE International Conference on Neural Networks*, pp. 586-591.

Thimm G. and Fiesler E., 1997, “High order and multilayer perceptron initialization,” IEEE Transactions on Neural Networks, vol. 8, no. 2.

Wessels L. and Barnard E., 1992, "Avoiding false local minima by proper initialization of connections," IEEE Transactions on Neural Networks, vol. 3, no. 6, pp. 899-905.

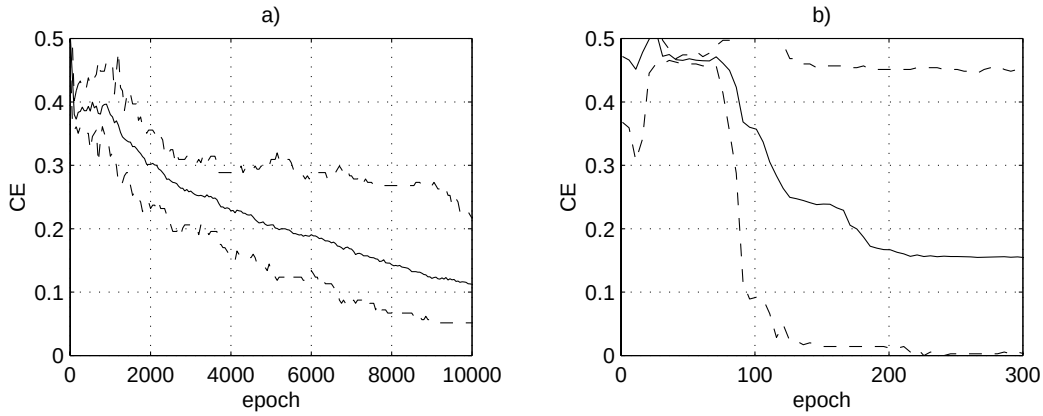

Figure 1: Classification error as a function of training epochs with the initialization with small random values; a) two spirals problem, and b) equalization problem (for independent test set). The solid line represents the average curve, while the dashed lines represent the minimum and maximum curves, respectively.

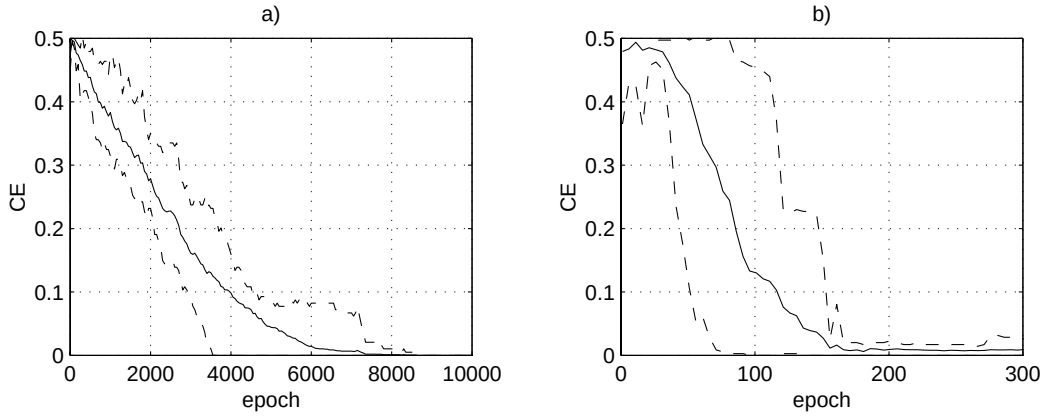

Figure 2: Classification error as a function of training epochs with the RP-initialization; a) two spirals problem, and b) equalization problem (for independent test set). The solid line represents the average curve, while the dashed lines represent the minimum and maximum curves, respectively.

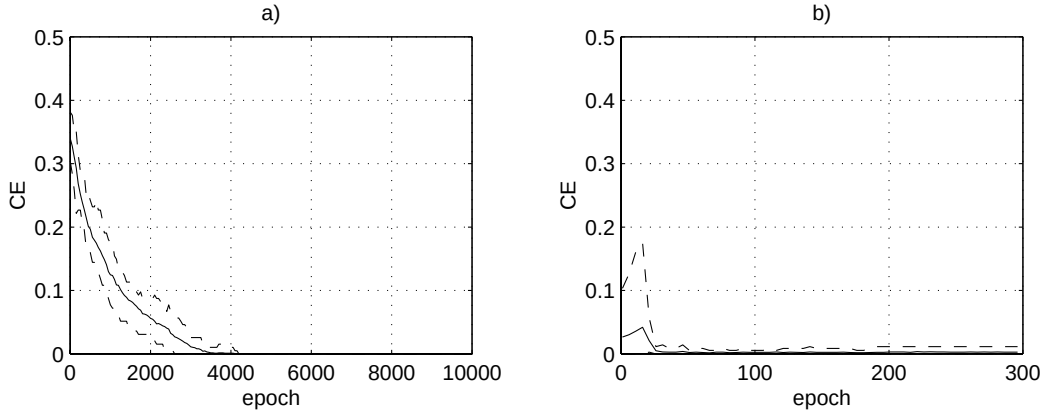

Figure 3: Classification error as a function of training epochs with the MC-initialization; a) two spirals problem, and b) equalization problem (for independent test set). The solid line represents the average curve, while the dashed lines represent the minimum and maximum curves, respectively.