The Korean Communications in Statistics **Vol. 14** No. 3, 2007, pp. 601–608

A Method of Determining the Scale Parameter for Robust Supervised Multilayer Perceptrons

Ro Jin Pak[1)]

Abstract

Lee, *et al.* (1999) proposed a unique but universal robust objective function replacing the square objective function for the radial basis function network, and demonstrated some advantages. In this article, the robust objective function in Lee, *et al.* (1999) is adapted for a multilayer perceptron (MLP). The shape of the robust objective function is formed by the scale parameter. Another method of determining a proper value of that parameter is proposed.

Keywords: Multilayer perceptron; robust radial basis function.

1. Introduction

There exist many types of feedforward neural networks in the literature, for example, multilayer perceptron (MLP), a radial basis function (RBF) network (Saha, *et al.*, 1993) *etc.* The MLP model has unknown parameters, often called *weights*, and we seek values for them that make the model fit the training data well. In order to get proper weights, the standard back propagation (BP) algorithm is the most common learning algorithm for an MLP. The basic idea of BP algorithm is based on the least mean squares (LMS) method which is intended to optimize the fit of a model with respect to the training data by minimizing the square of residuals (Hecht-Nielsen, 1990). The use of LMS for data modeling can be dated back to Gauss and Legnedre (Hampel, *et al.*, 1986).

Suppose a data set contains outliers, an MLP obtained by the use of the LMS method often becomes inaccurate. To remedy this problem, the error function is replaced by a function which is symmetric and continuous whose influence function is of bound functions (Liano, 1996).

Let us consider an MLP with one output unit y. The goal of a learning algorithm is to minimize a cost function of the form

$$E = \frac{1}{P}\sum_{p=1}^{P}\rho(r_p),$$

1) Associate Professor, Department of Information & Statistics, Dankook University, Jukjun-Dong, Suji-Gu, Kyunggi-Do 448-160, Korea.
E-mail: rjpak@dankook.ac.kr

where the error function $\rho(\cdot)$ is symmetric and continuous, $r_i = t_i - y_i$ is the residual of pattern i with target t_i and P is the number of training patterns. The LMS method by setting the error function in combining robust estimation (Hampel, *et al.*, 1986) turns out very effective in improving the performance of MLP model (Liano, 1996). Later, Lee, *et al.* (1999) proposed the objective function driven from robust statistics (Chen and Jain, 1994), which is a class of robust object functions with the form given as follows:

$$E_R(r_p) = \sum_{p=1}^{P} [\phi(r_p) - \phi(0)],$$

where $\phi(r_p)$ is a continuous function, $\phi(0)$ is a constant, and P is the total number of inputs. Note that $E_R(r_p)$ becomes the least squares criterion when $\phi(r_p) = r_p^2$ and $\phi(0) = 0$. Lee *et al.* (1999) let the derivative function $\psi(r) = d\phi(r)/dr$ be $\psi(r) = s(r)t(r)$ with the six necessary properties about $s(r)$ and $t(r)$ (details in Lee, *et al.*, 1999, p. 678). Lee, *et al.* (1999) proposed a robust objective function step by step starting with

$$\psi(r) = s(r)t(r) = r \cdot e^{-r^2/2\sigma}.$$

Next, the derivative of $\psi(r)$ is calculated as

$$\begin{aligned} d\psi(r)/dr &= e^{-r^2/2\sigma} - (r^2/\sigma)e^{-r^2/2\sigma} \\ &= (1 - r^2/\sigma)e^{-r^2/2\sigma}. \end{aligned}$$

Setting $d\psi(r)/dr = 0$ yields two extreme points of $\psi(r)$, $\pm\sigma^{1/2}$, one in $r > 0$, and the other in $r < 0$ which can be chosen to be two cutoff points of $\psi(r)$. The confidence interval of the residual is $[-\sigma^{1/2}, \sigma^{1/2}]$. Also

$$\phi(r) = \int re^{-r^2/2\sigma} dr = -\sigma e^{-r^2/2\sigma}.$$

Finally, the corresponding robust objective function is therefore obtained as

$$E_R(r) = \phi(r) - \phi(0) = \sigma(1 - e^{-r^2/2\sigma}). \tag{1.1}$$

The RBF network which is considered as a good candidate for approximation problems because of its faster learning capability compared with other feedforward network. In traditional RBF networks, the Gaussian function and the least squares criterion are selected as the activation function of network and the objective functions, respectively. Lee, *et al.* (1999) demonstrated that the radial basis function (RBF) network with $E_R(r)$ in (1.1) as an objective function has the advantages over the ordinary RBF (Moody and Darken, 1989), which are 1) better capability; 2) faster learning speed; 3) better size of the network; 4) higher robustness to outliers.

In this article, we borrow the objective function $E_R(r)$ for training the traditional MLP and try to find out the way of determining a proper value of σ. The parameter σ

in (1.1) can be replaced by σ^2 for computational convenience from the statistical point of view, then from now on

$$E_R(r) = \phi(r) - \phi(0) = \sigma^2(1 - e^{-r^2/2\sigma^2}). \tag{1.2}$$

Throughout this article, $E_R(r)$ in (1.2) will be used as an objective function.

Researchers have not been very much interested in the value of the scale parameter, the scale parameter is usually predetermined. The main purpose of this article is to show the method of determining a proper value of the scale parameter, which is derived from the data.

2. Updating Sigma : Methods and Simulations

The algorithm described in this section is based on the gradient descent method which provides update rules for the parameter σ. The proposed learning rule about σ can be written as

$$\begin{aligned}\sigma(t+1) &= \sigma(t) + \eta \frac{\partial E_R(r_p)}{\partial \sigma}|_{\sigma=\sigma(t)} \\ &= \sigma(t) + \eta \sum_{p=1}^{P} [2(1 - e^{-x_p^2/2\sigma^2})\sigma - x_p^2(e^{x_p^2/2\sigma^2}\sigma)^{-1}]|_{\sigma=\sigma(t)},\end{aligned} \tag{2.1}$$

where η is a learning rate. In order to see how the parameter σ is updated, an example, so called the T-C problem, is considered. The T-C problem is a fairly simple pattern recognition problem. We wish to train a network to distinguish between the letters T and C, independent of the angle of rotation of these letters. We shall restrict the rotation angles to multiples of 90 degrees, resulting in four possible inputs for each letter, as shown in Figure 2.1. We choose to represent the letter T by an output value of 0.1, and the letter C by an output value of 0.9. For more detail about the T-C problem, refer (Freeman, 1994). The same network scheme in (Freeman, 1994) is employed for simulations: The three-layer back propagation network with nine input nodes, one hidden layer with three nodes, one output node and the sigmoid function for an activation function.

We consider three types of MLP;

1. (ordinary) MLP: MLP with $E_R(r) = r^2/2$.
2. MLP with a fixed sigma: MLP with $E_R(r)$ in (1.2), where σ is fixed. (1.65 is used in this case; 1.65 is the 95^{th} percentile of the standard normal distribution.)
3. MLP with sigma-updates: MLP with $E_R(r)$ in (1.2), where σ is updated as in (2.1).

The network is trained with $\eta = 0.2, 1.0$ and 2.0, respectively, for 2000 iterations (epochs). The training is carried out 500 times with two types of input features: (1) uncontaminated patterns; (2) contaminated patterns with the replacement of the first

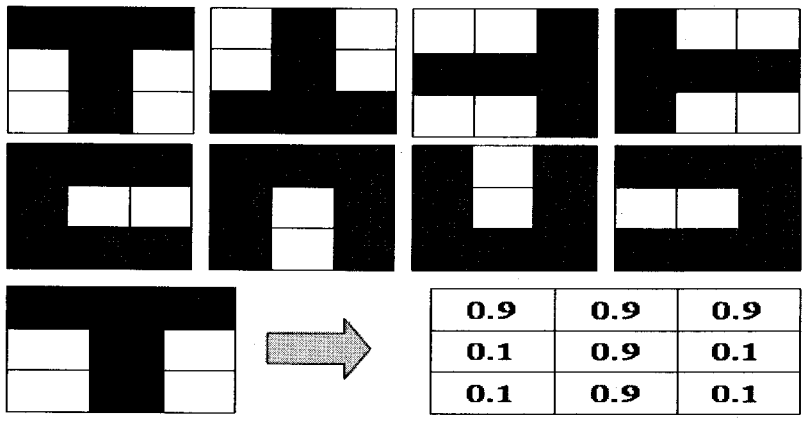

Figure 2.1: This figure shows the data-representation scheme for the T-C problem. Each letter is superimposed on a 3 by 3 grid. Filled grid-squares are represented by the real number 0.9, and empty ones are represented by 0.1. Each input vector consists of nine real numbers.

input pattern (0.9, 0.9, 0.9, 0.1, 0.9, 0.1, 0.1, 0.9, 0.1) by (9, 9, 9, 1, 9, 1, 1, 9, 1). Representative results are displayed in Figure 2.2 and Figure 2.3. The error measure is the average of 500 squares of simulated errors at each iteration from 1 to 2000. Some important discoveries from the simulations are listed as follows:

- Effect of updating the sigma [Figure 2.2]: It is clear that the error measures of the MLP with sigma-updates are more stable than an ordinary MLP and the MLP with fixed sigma being 1.65. In fact, there is no difference in performance between the MLP and the MLP with a fixed sigma.
- Learning rate [Figure 2.3-(a), (b) and (c)]: The learning rate is another key factor for the performance of MLP with sigma-updates. The larger the learning rate, the faster the error measure diminishes.
- Choice of the sigma [Figure 2.3-(d)]: The performance of the MLP with an inapporiately fixed sigma may become very poor. In our case, σ=0.02, which is very small.
- Convergence of the sigma-update [Figure 2.3-(e), (f)]: The sigma is approaching to 1.5515 (1.3716), 1.6589 (1.4731) and 1.7503 (1.6036) for $\eta = 0.2, 1.0$ and 2.0, respectively. The numbers are the medians of 500 sigma-updates at each iteration based on contaminated and uncontaminated patterns (numbers in parentheses are of uncontaminated patterns). In practice, such a limiting value of sigma-updates for a given input patterns would be used for validation and test.

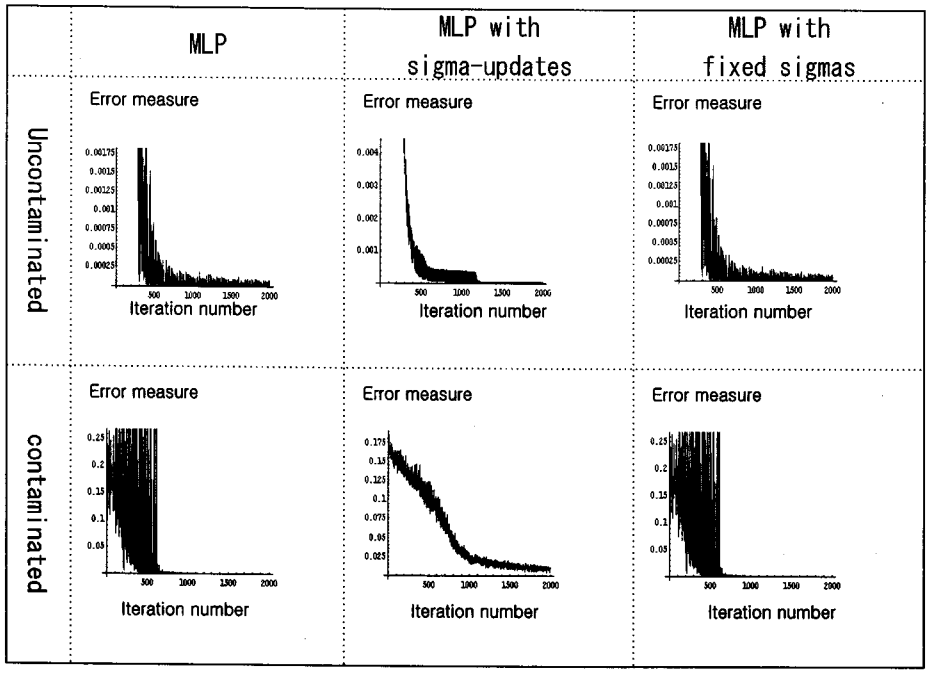

Figure 2.2: Error measure *vs.* Iteration number: $\eta = 2.0$

Another realistic example considered; Iris data with 150 random samples of flowers from the iris species setosa, versicolor, and virginica. From each species there are 50 observations for sepal length, sepal width, petal length, and petal width in cm. A MLP with 5 hidden layers, is trained 2000 times with $\eta = 0.2$, 1.0 and 20. Figure 2.4 shows the traces of the estimates of σ which approaching to about 1.2114, 1.2109 and 1.2103 for $\eta = 2.0$(upper curve), $\eta = 1.0$(middle curve)and $\eta = 0.2$(lower curve), respectively.

3. Comments and Conclusions

Before closing this article, we would like to mention that there is a chance for the sigma to diverge. Finding the limiting value of sigma-updates in (2.1) is to start with an arbitrary element $\sigma(1)$ in σ and to define an iterative sequence by $\sigma(t+1) = g(\sigma)|_{\sigma=\sigma(t)}$, where

$$g(\sigma) = \sigma + \eta \frac{\partial E_R(r_p)}{\partial \sigma}.$$

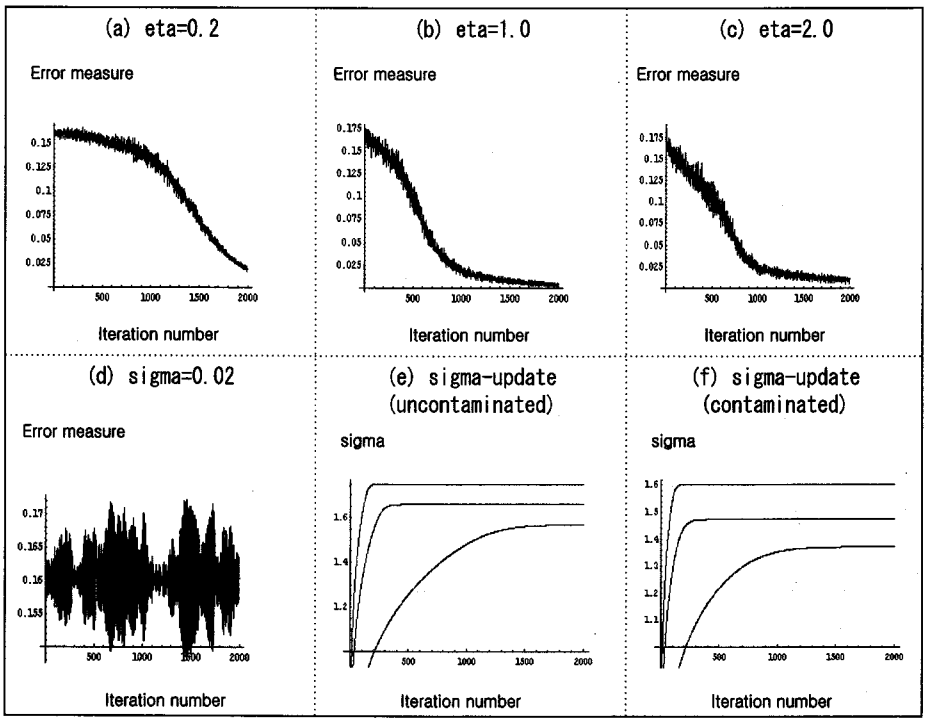

Figure 2.3: Error measure *vs.* Iteration number - (a) $\eta = 0.2$, contaminated, MLP with sigma-update; (b) $\eta = 1.0$, contaminated patterns, MLP with sigma-update; (c) $\eta = 2.0$, contaminated patterns, MLP with sigma-update; (d) $\eta = 1.0$, uncontaminated patterns, MLP with fixed $\sigma = 0.02$: sigma *vs.* Iteration number - (e) uncontaminated patterns, (f) contaminated patterns with $\eta = 2.0$(upper curve), $\eta = 1.0$(middle curve), $\eta = 0.2$(lower curve)

If this sequence converges, then its limit is called a fixed point of $g(\sigma)$. If there exists L, called the Lipschitz constant such as $|g'(\sigma)| \leq L < 1$ on certain interval $l \subset \mathbb{R}$, then there exits a fixed point on that interval l (Andrzej and Dugundji, 2003). In this case

$$g'(\sigma) = 1 + \eta \sum_{p=1}^{P} \{2(1 - e^{-x_p^2/2\sigma^2}) - x_p^2/(\sigma^2 e^{x_p^2/2\sigma^2}) - x_p^4/(\sigma^4 e^{x_p^2/2\sigma^2})\},$$

and a particular $|g'(\sigma)|$ against σ is in Figure 3-1. There exists an intervals near small σ where the $|g'(\sigma)|$ is bounded by the $L < 1$, but for the very large σ the upper bound L is in fact 1 [Figure 3.1-(a)], which means that it is not always possible to find such an interval $l \subset \mathbb{R}$ on which the upper bound $L < 1$. That is, if any sigma-update

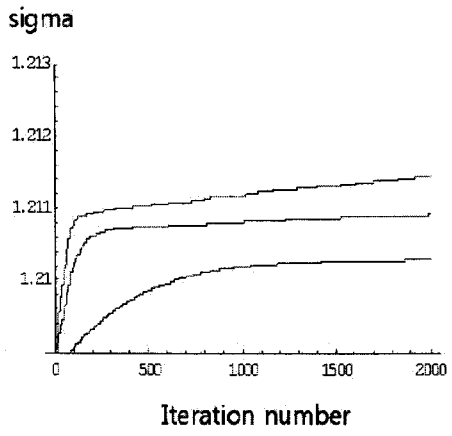

Figure 2.4: Sigma *vs.* Iteration number-Iris data with $\eta = 2.0$(upper curve), $\eta = 1.0$(middle curve), $\eta = 0.2$(lower curve)

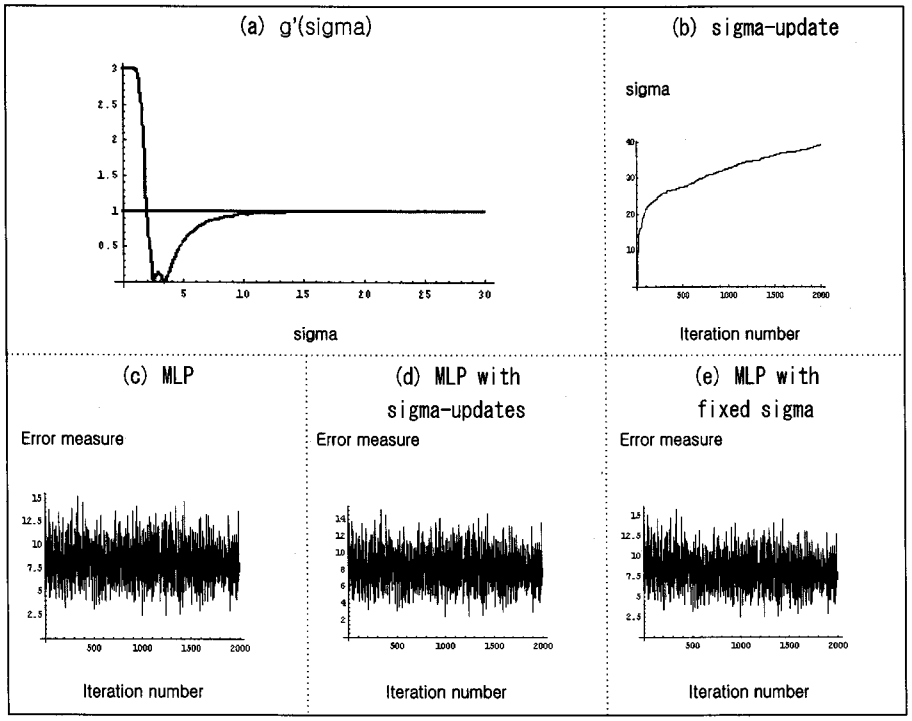

Figure 3.1: A form of $|g'(\sigma)|$ with $\eta = 1.0, P = 1, x = 5$; Learning with contaminated patterns with miscoded target values

happens to be very large in any event, there is a chance for the sigma-updates not to converge to a fixed value. However, the performance of the MLP with sigma-updates won't be worse than that of the ordinary MLP, because $E_R(r) \to r^2/2$ as $\sigma \to \infty$. In order to see the case where sigma-updates are diverging, we run 500 simulations with the contaminated patterns such as 20% of 2000 target values 0.1 and 0.9 are replaced by 1 and 9, respectively. The error measures are plotted against iteration numbers up to 2000 for the ordinary MLP, the MLP with sigma-updates and the MLP with fixed sigma being 1.65 [Figure 3.1-(c), (d) and (e)]. All of the three plots show unstable patterns. Figure 3.1-(b) displays a trace of sigma-updates which keep on increasing.

We propose to update the sigma in the same manner as the weights are updated, and then we would like to conclude that the learning by the proposed method produces more stable result than the ordinary MLP. Though there is still a chance of divergence of the sigma-updates, performance of the proposed learning method is not worse than that by the ordinary learning method based on the least squares criterion.

References

Andrzej, G. and Dugundji, j. (2003). *Fixed Point Theory.* Springer-Verlag, New York.

Chen, D. S. and Jain, R. C. (1994). Robust back propagation learning algorithm for function approximation. *IEEE Transactions on Neural Networks*, **5**, 467–479.

Freeman, J. A. (1994). *Simulating Neural Networks with Mathematica.* Addison-Wiley, New York.

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J. and Stahel, W. A. (1986). *Robust Statistics: The Approach Based on Influence Functions.* Addison-Wiley, New York.

Hecht-Nielsen, R.(1990). *Neurocomputing.* Addison-Wesley, Reading, MA.

Lee, C.-C., Chung, P.-C., Tsai, J.-R., and Chang, C.-I. (1999). Robust radial basis function neural networks. *IEEE Transactions on System, Man and Cybernetics*, **29**, 674–685.

Liano, K. (1996). Robust error measure for supervised neural network learning with outliers. *IEEE Transactions on Neural Networks*, **7**, 246–250.

Moody, J. and Darken, C. (1989). Fast learning in networks for function interpolation. *Neural Computation*, **3**, 281–284.

Saha, A., Wu, C. L. and Tang, D. S. (1993). Approximation, dimension reduction and non-convex optimization using linear superposition of Gaussians. *IEEE Transaction on Computers*, **42**, 1222–1233.

[Received August 2007, Accepted October 2007]