Multilayer Perceptron Trained with Numerical Gradient

Włodzisław Duch
Department of Informatics
Nicholaus Copernicus University Toruń, Poland
www.phys.uni.torun.pl/~duch
and School of Computer Engineering
Nanyang Technological University, Singapore

Mirosław Kordos
Institute of Computer Science
The Silesian University of Technology
Gliwice, Poland

***Abstract*—An application of numerical gradient (NG) to training of MLP networks is presented. Several versions of the algorithm and the influence of various parameters on the training process are discussed. Optimization of network parameters based on global search with numerical gradient is presented. Examples of two-dimensional projection of the error surface are shown and the influence of various numerical gradient parameters on the error surface is presented. The speed and accuracy of this method is compared with the search-based MLP training algorithm**.

***Index Terms*—Neural Networks, Numerical Gradient**

I. Introduction

Numerical Gradient (NG) in its basic form is a local gradient-based search method. In contradiction to training algorithms, which use analytical gradient [1], it does not require the knowledge of transfer functions and connection structures among neurons. Moreover transfer functions do not have to be differentiable. Another advantage of NG is the possibility of an easy visualization of the error surface and examining the influence of various parameters on it.

A good initialization of network parameters followed by a gradient optimization may be the quickest method to find an optimal neural network solution [2]. Good initialization can be achieved by a global version of NG, exploring parameter changes that lead to better, but sometimes also to worse solutions, using 'educated guesses' to find better start for the local search than multistart with random weight values.

Three variants of local training algorithms are described in the next section, using discrete, continuous and mixed NG. A version of NG that performs global search first, and then local search around the most promising points, is described in the third section. Influence of the training parameters on the error surface is considered in the fourth section, followed by the comparison with search-based training.

II. Training Algorithm

The weights and biases of the hidden and output layer neurons should be optimized. The supervised learning starts with random initial weights. The initial weights cannot be all equal (e.g. all zero), because all weights of hidden neuron will than take the same value. The error function (e.g. MSE) is calculated either on the whole training set, or on a randomly chosen part of it.

The training algorithm consists of two stages: finding the gradient direction and finding the minimal error along this direction. Instead of analytical formulas used in the back-propagation gradient-based methods, all network parameters are perturbed to find the gradient direction.

A. Finding the Gradient Direction

To find the gradient direction for each network parameter w a constant value Δw is added to it and an error $E[w+\Delta w]$ calculated. If it decreases, then the component of the gradient direction $\mathbf{V}[w]$ is calculated as:

$$\mathbf{V}[w] = \eta\ [(E[w]-E[w+\Delta w]) / E[w]]^n$$

The same value Δw is subtracted from the weight w and $E[w-\Delta w]$ calculated. If it decreases more then it decreased previously, then the gradient component should be:

$$\mathbf{V}[w] = -\ \eta\ [(E[w]-E[w-\Delta w]) / E[w]]^n$$

If changing the weight does not change the error than $\mathbf{V}[w] = 0$. The values $\mathbf{V}[w]$ are proportional to the error decrease as a result of the perturbation with Δw. The proportions are adjusted by the exponent n ($n > 0$). Usually $n = 1$ gives the best results, but in some cases larger exponents may be beneficent.

B. Finding the Minimum along the Gradient Direction

Making one step Δw along the gradient direction $\mathbf{V}$ moves from the starting point $D = \Delta w\|\mathbf{V}\|$ units, where

$$\|\mathbf{V}\| = \left(\sum_w \mathbf{V}[w]^2 \right)^{1/2}$$

Since the starting point of the next iteration is in the minimum of the previous iteration, D is also the distance between two successive starting points. The bigger $\|\boldsymbol{V}\|$ is the faster an error changes when moving along the gradient direction (the error surface is steeper). Some examples of the influence of Δw on the training and the error surface are given later.

In the Discrete NG the update is made on one unit Δw in the direction of V. The minimum region is found only roughly, but quickly. If Δw is not too big, the method is quite efficient. If Δw is very big, then it can be used as a global search. Δw can also decrease, changing slowly from global to local search.

In the Continuous and Mixed NG better approximations are used to find the minimum more precisely (e.g. parabolic approximation). Sample plots of the error surface along the gradient direction are presented in section 4.

In our experiments the number of training cycles was linearly related to the precision of finding the minimum along the gradient direction. For example, increasing the accuracy of finding the minimum by 10% decreases the number of cycles needed to train the network also by 10%.

C. *Three versions of NG.*

Discrete NG. At the beginning Δw = 1/slope. After calculating all $\mathbf{V}[w]$ each weight is changed by $\Delta w \cdot \mathbf{V}[w]$, moving to the next iteration. The minimum along the gradient direction is not searched for. This is repeated until the error decreases. Then $\Delta w = \Delta w/2$ is taken, and the process continues.

Continuous NG. In this case the gradient (or derivative) definition is used:

$$\frac{\partial E}{\partial w} = \lim_{\Delta w \to 0} \frac{E(w + \Delta w) - E(w)}{\Delta w}$$

Since $\Delta w \rightarrow 0$, a small value of Δw = 0.001/slope is taken and is constant during the training. After finding the gradient direction the minimum along this direction is searched for. From this minimum the algorithm is repeated.

Mixed NG. In the first iteration Δw=0.1/slope. In all next iterations Δw is half of the distance D from the previous iteration starting point to the previous iteration minimum. This is a modification of continuous NG made to speed up the training.

III. Local and Global Search

Gradient methods perform local search. We have expanded NG search method to allow also for a global search. First the weight space is searched through using a discrete NG with a large step Δw and a large step along the gradient line (large $\|\mathbf{V}\|$). The erratic error behavior during such search is shown in Fig. 1, using small (106 samples) medical data sets (acute appendicitis data, [4]).

In the next step a local NG search with small Δw is used around the most promising points, to localize precisely the minimum along the gradient direction.

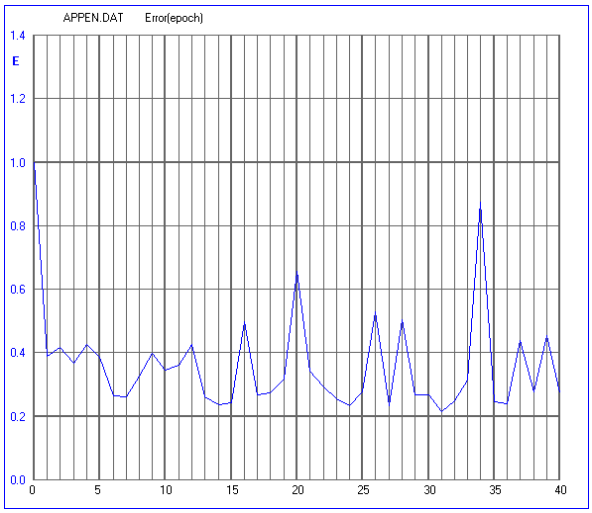

Fig. 1. An example of the MSE error changes during the first 40 training cycles of the global search (appendicitis data)

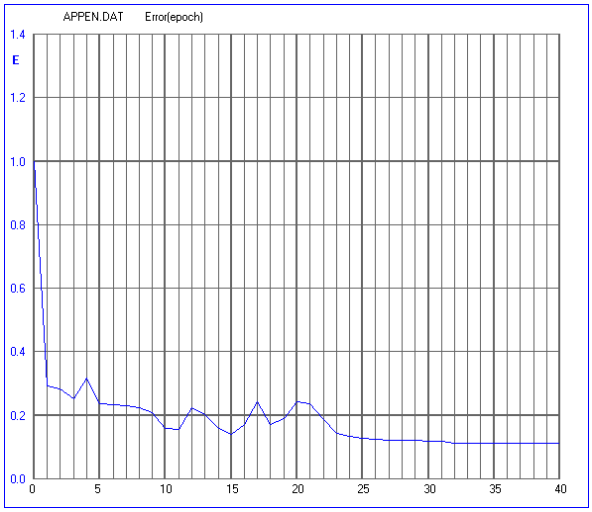

Fig. 2. MSE error changes in deterministic simulated annealing search (appendicitis data)

Another possibility is to decrease Δw and $\|\mathbf{V}\|$ gradually. This methods starts from a global search, and gradually transits to a local search. It resembles simulated annealing [3], but is fully deterministic. No random numbers are used except for generating initial weights. An example of error changes during this search is shown in Fig. 2 for the appendicitis data [4]. Convergence is much smoother comparing with Fig. 1, and a solution with lower MSE has been found than using the previous method of local search with random start.

IV. Training Parameters and Error Surface

The convergence speed depends on four factors: distribution of the initial weights, Δw and on the type of NG method and additional modifications of the method.

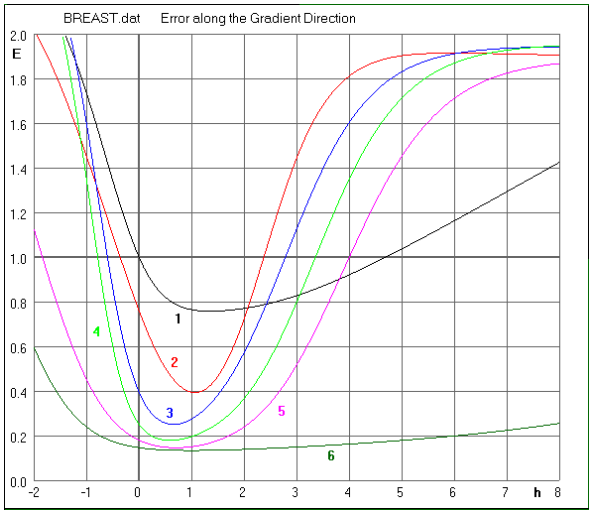

Fig. 3. Δw < optimal Δw in continuous or mixed NG; errors along the gradient direction for 6 training cycles. Horizontal axis (h): distance in the gradient direction from the starting point (h=0) for each training cycle. Vertical axis (E): relative MSE (MSE=1 when the training starts) (breast cancer data)

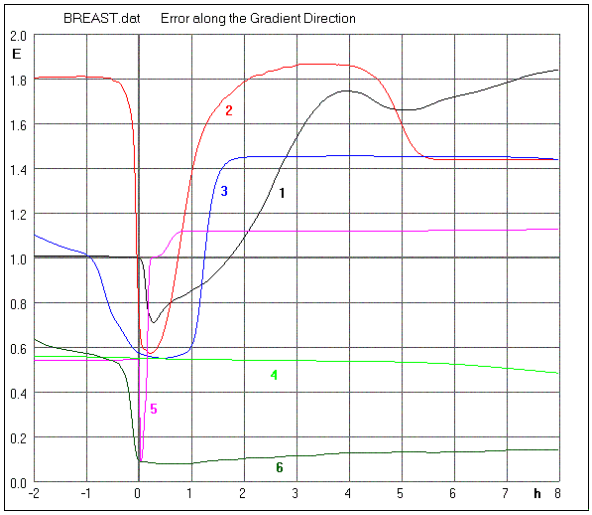

Fig. 4. Global search with big Δw, initialization in broad range. If global search is not required then local search with optimal Δw usually finds the solution quicker. The error surface with global search is much more irregular (breast cancer data)

While using local discrete NG, for small Δw (about 0.1 for breast cancer data, [4], another small two-class medical data, with 286 samples) smooth error surface is obtained, but the convergence on the training set is slow. For larger Δw a more irregular error surface is found with a tendency to have multiple local minima along the gradient direction. If many local minima exist along the line then simple one-dimensional minimization methods based on parabolic approximation may not be sufficient. With too big Δw (more then 1.5 for breast cancer data) the training is no longer convergent. There exists an optimal Δw (about 0.4 for breast cancer data), but finding it may require some trials. In a continuous or mixed NG the shape of an error surface is very similar for 0.001< Δw <optimal Δw. The calculated gradient direction depends on Δw. The training is quicker if Δw = optimal Δw, as used in the mixed NG or is close to optimal Δw but smaller. The error in h=0 (starting point) for each curve is the same as the minimum of the previous curve. This is shown in Fig. 3. The curves in Fig. 3 are similar to those discrete NG with optimal Δw (not shown here, because of limited space), and a well-chosen Δw in a discrete NG can replace finding the minimum along the gradient direction. The mixed NG tries to find Δw close to the optimal value.

V. Method Improvements

We have tested and successfully applied several methods to speed up the training.

1. Dividing the training set into several randomly chosen parts and performing training on a different part in each cycle. This can speed up the training several times for small and tens of times for large datasets.
2. Using the following form of momentum:

$$\Delta w = (1+momentum)*V[w]+momentum*previousV[w]$$

The optimal momentum can range from 0.5 for small to 0.1 for large datasets. Using momentum can stabilize and speed up the training a few times.

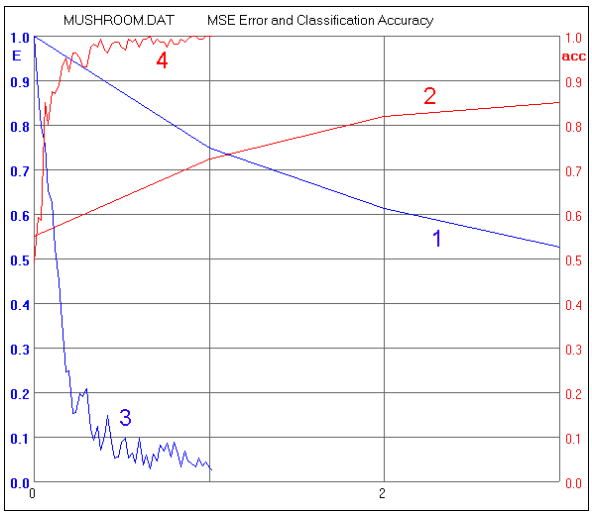

Fig. 5. Comparison of MSE error and classification accuracy for the same dataset. Horizontal axis: calculation time.
1,2 – error and accuracy for standard mixed NG
3,4 – error and accuracy for improved mixed NG using momentum and dividing the training set into 10 randomly parts and choosing a different part each training cycle

3. Boundary vectors. Afret some initial cycles only vectors, which produce greater error and therefore are supposed to be situated closer to class boundaries are selected for further training. This method is especially useful for large, and not too noisy data, where only a small subset of vectors is selected for further training. In such a case the speed-up may be quite high.

4. Freezing weights. Weights that change less than min_dW two or three consecutive cycles are considered to be either irrelevant or to have already reached their optimal values. These weights are not examined in the next cycles. If after the training they are equal to their random values before the training, then such weights are set to zero. The speed up is small for datasets with few features and may reach few times for datasets with many features, since in this case many of them can be irrelevant for classification.

VI. Experimental Results and Comparison with SMLP Algorithm

The search-based MLP training (SMLP) used in our previous paper [5] can also be applied to a typical MLP network. The search method changes one or two weights at a time by $+\Delta w$ or $-\Delta w$. If the error decreases then new weight value is accepted and a step in its direction is immediately made. Each weight is changed by the same Δw. Global search can be realized by Beam Search or multistart techniques.

In NG the step is made only once per training cycle, when all the changes of single weight are already calculated. Each weight is changed by a different value Δw proportional to the error change in its direction. If we use the discrete version of NG and allow the algorithm to go to points with not only lower, but also higher error values then previously, then two version of a global search can be used: global and then local search starting from the most promising points, and deterministic NG simulated annealing.

Tab.1. Experimental results. The classification accuracy in 10-fold crossvalidation (in brackets accuracy on training set)

dataset (number of vectors)	standard NG		improved NG	
	epochs needed	accuracy	epochs needed	accuracy
mushroom (8124)	24724	0.996	309	0.996
ionosphere (351)	8638	0.93(0.98)	203	0.91(0.92)
sonar (208)	14230	0.80(0.995)	554	0.80(0.99)
iris (150)	1990	0.96(0.98)	258	0.96(0.98)
breast (699)	425	0.97(0.98)	70	0.97(0.98)

By "epoch" we mean calculating an error once on all vectors, by "training cycle" running the algorithm once.

Our experiments on several data sets from UCI repository showed that for small databases the search method and NG require similar computational effort and produce similar results. For big and noisy datasets the search method is quicker and finds solutions with better accuracy on the training set, but the solutions found with NG have better generalization and therefore have higher accuracy on test sets. If the data is complex and require global minimizations then NG global search method is often quicker than multistart or beam search with the search method. Both methods can use discrete values and do not require either the knowledge of connection structure or differentiable transfer functions.

Table 1 contains some results for NG using 10-fold cross-validation with mixed NG and double parabolic approximation for finding minimum along the gradient direction. The improved NG uses the best combinations found of speed improvement methods.

VII. Conclusions

Numerical calculation of gradients is especially suitable for hardware implementations. Weight perturbation method has been used by Koosh [6] in VLSI implementation of neural models to learn logical functions AND and XOR.

In contrast to training algorithms, which use analytical gradients, NG does not require the knowledge of transfer functions and can be applied to optimization of any black-box parameters, as long as responses to perturbations of these parameters are calculated. This makes it a very universal approach with a possible wide range of applications. It is also very easy to implement. Its shortage, common to all gradient methods, is the possibility of finding only a local minimum. This can be overcome with the global versions of NG: global and then local NG search and deterministic NG simulated annealing. Many variants of search methods remain to be explored, but the initial results reported here are promising. Detailed comparison with analytical gradient-based training methods will be done soon.

VIII. References

[1] C. Bishop, *Neural networks for pattern recognition,* Clarendon Press, Oxford 1999.

[2] W. Duch, J. Korczak, *Optimization and global minimization methods suitable for neural networks*, Dept. of Informatics, NCU technical report 1998

[3] S. Kirkpatric, C.D. Gellat, Jr. and M.P. Vecchi, *Optimization by simulated annealing*, Science **220**: 671-680, 1983

[4] S.M. Weiss, I. Kapouleas, *An empirical comparison of pattern recognition, neural nets and machine learning classification methods*, in: J.W. Shavlik and T.G. Dietterich, Readings in Machine Learning, Morgan Kauffman Publ, CA 1990

[5] M. Kordos, W. Duch, *Search-based Training for Logical Rule Extraction by Multilayer Perceptron*. ICONIP/ICANN 2003 (this volume)

[6] V.F. Koosh, Analog computation and learning in VLSI. PhD Thesis, Caltech, Pasadena, CA, 2001