EFFECTIVE NEURAL NETWORK TRAINING WITH A DIFFERENT LEARNING RATE FOR EACH WEIGHT

G.D. Magoulas
University of Athens,
Department of Informatics,
U.P. Artificial Intelligence
Research Center–UPAIRC,
GR-15771 Athens, Greece.

V.P. Plagianakos
University of Patras,
Department of Mathematics,
U.P. Artificial Intelligence
Research Center–UPAIRC,
GR-26500 Patras, Greece.

M.N. Vrahatis
University of Patras,
Department of Mathematics,
U.P. Artificial Intelligence
Research Center–UPAIRC,
GR-26500 Patras, Greece.

ABSTRACT

Batch training algorithms with a different learning rate for each weight are investigated. The adaptive learning rate algorithms of this class that apply inexact one-dimensional subminimization are analyzed and their global convergence is studied. Simulations are conducted to evaluate the convergence behavior of two training algorithms of this class and to compare them with several popular training methods.

1. INTRODUCTION

The use of a different learning rate for each weight allows us to find the proper stepsize that compensates for the small magnitude of the gradient in a flat weight direction in order to avoid slow convergence, and dampens a large weight change in a steep weight direction in order to avoid oscillations. Moreover, this approach exploits the parallelism inherent in the evaluation of $E(w)$ and $\nabla E(w)$ by the BP algorithm.

Various batch training methods with a heuristically determined learning rate for each weight have been proposed [4, 5, 10, 11, 13]. These methods follow the iterative scheme:

$$w^{k+1} = w^k - \mathrm{diag}\{\eta_1^k, \ldots, \eta_n^k\}\, \nabla E(w^k), \qquad (1)$$

and try to decrease the error by searching a local minimum with small weight steps. These steps are usually constraint by problem-dependent heuristic learning parameters, in order to ensure subminimization of the error function in each weight direction. However, these attempts to determine a proper learning rate for each weight usually result in a trade-off between the convergence speed and the stability of the training algorithm. Highly problem-dependent heuristic parameters are introduced to alleviate the stability problem. Furthermore, a well known difficulty of this approach is that the use of inappropriate heuristic values for a weight direction misguides the resultant search direction. In such cases, these training algorithms cannot exploit the global information obtained by taking into consideration all the directions and no guarantee is provided that the weight updates will converge to a minimizer of E.

2. ADAPTING A LEARNING RATE FOR EACH WEIGHT BY SUBMINIMIZATION

In function minimization problems, it is well known that all the local minima w^* of a continuously differentiable function E satisfy the necessary conditions

$$\nabla E(w^*) = \Theta^n. \qquad (2)$$

Eq. (2) represents a set of n nonlinear equations which must be solved to obtain w^*. Therefore, one approach to the minimization of the error function E is to seek the solutions of the set of Eq. (2) by including a provision to ensure that the solution found does indeed correspond to a local minimizer.

We propose solving the equation (2) by applying the class of nonlinear Jacobi methods. The main feature of the nonlinear Jacobi process is that it is a parallel algorithm [9], i.e. it applies a parallel update of the variables. So, starting from an arbitrary initial point $w^0 \in \mathcal{D}$, one can subminimize, at the kth iteration, the one-dimensional function:

$$E(w_1^k, \ldots, w_{i-1}^k, w_i, w_{i+1}^k, \ldots, w_n^k), \qquad (3)$$

along the ith direction and obtain the corresponding subminimizer $\hat{w}_i$. This is done in parallel for all $i = 1, \ldots, n$. Obviously for this $\hat{w}_i$

$$\partial_i E(w_1^k, \ldots, w_{i-1}^k, \hat{w}_i, w_{i+1}^k, \ldots, w_n^k) = 0. \qquad (4)$$

This is a one-dimensional subminimization because all other components of the vector w, except the ith, are kept constant. Then the ith weight is updated according to the following equation:

$$w_i^{k+1} = w_i^k + \tau^k(\hat{w}_i - w_i^k), \qquad \tau^k \in (0, 1]. \qquad (5)$$

Depending on the one-dimensional subminimization method [3, 9], used for the subminimization of (3),

0-7803-5682-9/99/$10.00©1999 IEEE.

we can obtain training algorithms. When exact one-dimensional subminimization is applied, and $\tau^k = 1$ for all k, convergence can be established (see [2]). When inexact one-dimensional subminimization is applied, the number of iterations of the subminimization method is related to the requested accuracy in obtaining the subminimizer approximations. The following theorem provides a local convergence result. It shows that there is a neighborhood of a minimizer of the error function and initial weights in this neighborhood for which convergence to the minimizer can be guaranteed and it is applicable to any iterative scheme (5) utilizing inexact one-dimensional subminimization.

Theorem 1 [7]: Let $E: \mathcal{D} \subset \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable in an open neighborhood $\mathcal{S}_0 \subset \mathcal{D}$ of a point $w^* \in \mathcal{D}$ for which $\nabla E(w^*) = \Theta^n$ and the Hessian, $H(w^*)$, is positive definite with the property A^π. Then there exists an open ball with radius r, $\mathcal{S} = \mathcal{S}(w^*, r)$, in $\mathcal{S}_0$ such that any sequence $\{w^k\}_{k=0}^{\infty}$ generated by (5) converges to the point w^* which minimizes E.

Note that a matrix A has the property A^π if A can be permuted by PAP^T into a form that can be partitioned into a block-tridiagonal form [1]. For an algorithm which transforms a symmetric matrix to tridiagonal form, see [15, p.335]. It is worth mentioning that the asymptotic rate of convergence of the algorithms of this class depends on the spectral radius ρ (see [7] for a proof). Due to the nonlinearity of the error function exact minimization steps along each weight direction do not usually help. On the other hand, when the current iterate w^k is close to the minimizer a "better" estimator of the minimizer w^{k+1} can be found without much difficulty. Thus, in practice, one iteration of the one-dimensional subminimization method is used [9].

Below, the global convergence properties are studied. By a globally convergent algorithm we mean an algorithm with the property that for any initial point the sequence of the iterates converges to a local minimizer of the error function (see [3]). A strategy of this kind consists in accepting τ^k along the search direction φ^k if it satisfies the *Wolfe conditions*:

$$E(w^k + \tau^k \varphi^k) - E(w^k) \leq \sigma_1 \tau^k \langle \nabla E(w^k), \varphi^k \rangle, \quad (6)$$

$$\langle \nabla E(w^k + \tau^k \varphi^k), \varphi^k \rangle \geq \sigma_2 \langle \nabla E(w^k), \varphi^k \rangle, \quad (7)$$

where $0 < \sigma_1 < \sigma_2 < 1$ and $\langle \cdot, \cdot \rangle$ stands for the usual inner product in $\mathbb{R}^n$. In practice, the condition (7) is generally not needed, because the use of a backtracking strategy avoids very small steps. Alternatively, see [3], Relation (7) can be replaced by

$$E(w^k + \tau^k \varphi^k) - E(w^k) \geq \sigma_2 \tau^k \langle \nabla E(w^k), \varphi^k \rangle, \quad (8)$$

where $\sigma_2 \in (\sigma_1, 1)$. A simple backtracking strategy to tune the length of the minimization step, so that it satisfies Conditions (6)-(7) at each epoch, is to decrease τ^k by a reduction factor $1/q$, where $q > 1$ [9]. We remark here that the selection of q is not critical for successful learning, however it has an influence on the number of error function evaluations required to obtain an acceptable weight vector. In practice, the condition (7) is simply enforced by placing a lower bound on the acceptable values of τ. This bound has the same theoretical effect as the condition (7) and ensures global convergence [3]. The value $q = 2$ has been found to work without problems in the experiments reported in the next section.

In this framework, an important theorem due to Wolfe [3] states that if E is bounded below, then the sequence $\{w^k\}_{k=0}^{\infty}$ generated by any algorithm that follows a descent direction φ^k whose angle θ_k with $-\nabla E(w^k)$ is such that

$$\cos\theta_k = \frac{\langle -\nabla E(w^k), \varphi^k \rangle}{\|\nabla E(w^k)\| \, \|\varphi^k\|} > 0, \quad (9)$$

and satisfy the Wolfe's conditions, then it holds that $\lim_{k\to\infty} \nabla f(w^k) = 0$ [3]. This is also true when Relation (7) is replaced by Relation (8) [3]. For a relative convergence result, where the sequence $\{w^k\}_{k=0}^{\infty}$ converges q-superlinearly to a minimizer w^*, see [3, p.123]. The following theorem provides a global convergence result for training algorithms with a different learning rate for each weight.

Theorem 2: Suppose that the error function $E: \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. Assume that ∇E is Lipschitz continuous on $\mathbb{R}^n$. Then, given any point $w^0 \in \mathbb{R}^n$, for any sequence $\{w^k\}_{k=0}^{\infty}$, generated by the iterative scheme:

$$w^{k+1} = w^k - \tau^k \, \mathrm{diag}\{\eta_1^k, \eta_2^k, \ldots, \eta_n^k\} \, \nabla E(w^k), \quad (10)$$

where $\tau^k > 0$, η_m^k, $m = 1, 2, \ldots, i-1, i+1, \ldots, n$ are arbitrarily chosen positive real numbers and

$$\eta_i^k = -\frac{\delta}{\partial_i E(w^k)} - \frac{1}{\partial_i E(w^k)} \sum_{\substack{j=1 \\ j \neq i}}^{n} \eta_j^k \, \partial_j E(w^k), \quad (11)$$

where δ is a positive real number and τ^k satisfies the Wolfe's conditions (6)-(7), it holds that $\lim_{k\to\infty} \nabla E(w^k) = 0$.

Proof: Evidently, the error function E is bounded below on $\mathbb{R}^n$. The sequence $\{w^k\}_{k=0}^{\infty}$ follows the direction

$$\varphi^k(w^k) = -\mathrm{diag}\{\eta_1^k, \eta_2^k, \ldots, \eta_n^k\} \, \nabla E(w^k).$$

This is a descent direction, since $\langle \nabla E(w^k), \varphi^k(w^k) \rangle < 0$. In addition, since E is continuously differentiable

and bounded below, then there always exist stepsize τ^k satisfying the Wolfe's conditions (6)-(7). Moreover, the restriction on the angle θ_k is fulfilled since, as it can be easily justified utilizing Relation (9), $\cos\theta_k > 0$. Thus, by the Wolfe's Theorem [3], it holds that $\lim_{k\to\infty} \nabla E(w^k) = 0$. Thus the theorem is proved.

Note that for neural networks with sigmoid activation functions the assumption on continuous differentiability of the error function is redundant.

3. LEARNING RATE STRATEGIES

The problem of minimizing the error function E along the mth direction

$$\min_{\eta_m \geq 0} E(w + \eta_m e_m), \tag{12}$$

for $m = 1, \ldots, i-1, i+1, \ldots, n$ is equivalent to seeking the value of η_m that minimizes the one-dimensional function:

$$\phi_m(\eta) = E(w + \eta_m e_m). \tag{13}$$

Since $E(w) \geq 0$, $\forall w \in \mathbb{R}^n$, then the point w^* with $E(w^*) = 0$ minimizes $E(w)$. By applying one step of the Newton's method to the one-dimensional equation $\phi_m(\eta) = 0$ we obtain:

$$\eta_m^1 = \eta_m^0 - \frac{\phi_m(\eta^0)}{\phi_m'(\eta^0)}. \tag{14}$$

Since $\eta_m^0 = 0$ and $\phi_m'(\eta^0) = \langle \nabla E(w), d_m \rangle$, where $d_m = e_m$, then Eq. (14) can be written as

$$\eta_m = -\frac{E(w^k)}{\partial_m E(w^k)}, \tag{15}$$

while the value of η_i^k can by obtained by (11). The iterative scheme (10) with $\eta_m^k \neq 0$ obtained by Relation (15) constitutes the first proposed training algorithm which we call Alg-1.

Another approach is based on estimates of the Lipschitz constant in each direction. In this case, the direction

$$\varphi^k(w^k) = -\mathrm{diag}\{\eta_1^k, \eta_2^k, \ldots, \eta_n^k\} \nabla E(w^k),$$

is obtained utilizing the inverses of the estimates of the Lipschitz constant K along each weight direction. Specifically, the values of η_m^k, $m = 1, \ldots, i-1, i+1, \ldots, n$ are obtained by:

$$\eta_m^k = \frac{|w_m^k - w_m^{k-1}|}{|\partial_m E(w^k) - \partial_m E(w^{k-1})|}, \tag{16}$$

while η_i^k is given by (11). The scheme (10) with $\eta_m^k \neq 0$ obtained by Relation (16) constitutes the second proposed training algorithm which we call Alg-2.

Relative training algorithms can be easily constructed by utilizing any one-dimensional subminimization method.

4. EXPERIMENTS

In this section, we give comparative results for five batch training algorithms: Back-propagation with constant learning rate (BP); Back-propagation with constant learning rate and constant Momentum [12], named BPM; Adaptive Back-propagation with adaptive momentum (ABP) proposed by Vogl [16]; the Alg-1 and the Alg-2. The FNNs have been implemented in PC-Matlab and 1000 simulations have been run in each test case. The initial weights have been selected following a uniform probability from the interval $(-1, +1)$ and the best available values of the heuristic learning parameters for each problem have been used. The reader has to consider the fact that a gradient evaluation is more costly than an error function evaluation (for example Møller [8] suggests to count a gradient evaluation three times more than an error function evaluation).

In the first experiment a total of 12 Brodatz texture images of size 512×512 is acquired by a scanner at 150dpi [6]. From each texture image 10 subimages of size 128×128 are randomly selected and the co-occurrence feature extraction method is applied. A set of 10 sixteenth-dimensional training patterns are created from each image. A 16-8-12 FNN (224 weights, 20 biases) with sigmoid activations is trained to classify the patterns to 12 texture types. The termination condition is a classification error $CE < 3\%$.

Results regarding the training performance of the algorithms are presented in Table 1, where μ_{GRD} denotes the mean number of gradient evaluations required to obtain convergence, μ_{FE} the mean number of error function evaluations and % denotes the percentage of successful simulations.

Table 1: Results for the texture classification problem.

Algorithm	μ_{GRD}	μ_{FE}	%
BP	15839	15839	96
BPM	12422	12422	94
ABP	560	560	100
Alg-1	791	2185	100
Alg-2	382	591	100

The successfully trained FNNs are tested for their generalization capability using patterns from 20 subimages of the same size randomly selected from each image. The average percentage of classification success is: BP= 90%; BPM= 90%; ABP= 93.5%; Alg-1= 93%; Alg-2= 94.1%.

In general, Alg-2 outperforms all other methods tested. Alg-1 significantly outperforms BP and BPM in the number of gradient and error function evaluations

as well as in the percentage of successful simulations without using any heuristics and initial learning rate. ABP exhibits very good performance, however it requires fine tuning of four heuristic learning parameters.

In the second experiment, a 64-6-10 FNN (444 weights, 16 biases) for recognizing 8×8 pixel machine printed numerals from 0 to 9 [14]. The network is based on neurons of the logistic activation model. Numerals are given in a finite sequence $C = (c_1, c_2, \ldots, c_p)$ of input–output pairs $c_p = (u_p, t_p)$ where u_p are the binary input vectors in $\mathbb{R}^{64}$ determining the 8×8 binary pixel and t_p are binary output vectors in $\mathbb{R}^{10}$, for $p = 0, \ldots, 9$ determining the corresponding numerals. The termination condition for all algorithms tested is an error value $E \leq 10^{-3}$.

Detailed results regarding the training performance of the algorithms are presented in Table 2, where the abbreviations are as in Table 1.

Table 2: Results for the numeric font learning problem.

Algorithm	μ_{GRD}	μ_{FE}	%
BP	14489	14489	66
BPM	10142	10142	54
ABP	1975	1975	91
Alg-1	1361	3708	100
Alg-2	159	581	100

Obviously, the number of gradient evaluations is equal to the number of error function evaluations for the BP, the BPM, and the ABP. Alg-2 has the smallest average number of gradient evaluations which is considered very important in practice. Both Alg-1 and Alg-2 provide a greater possibility of successful training: they exhibit a 100% of success in the 1000 simulation runs. Regarding the performance of the Alg-1, it is better that the BP and the BPM (without needing an initial learning rate) and comparable to the overall performance of the ABP which needs fine tuning five parameters.

5. CONCLUSIONS

In this paper training algorithms that apply one–dimensional subminimization methods to adapt the rate of learning has been investigated. These algorithms use inexact line search strategies and ensure that the error function is sufficiently decreased at every iteration. This approach allows to reduce or even eliminate the influence of user-defined learning parameters. Two algorithms of this class have been compared with several widely used training algorithms and their efficiency has been numerically confirmed by the experiments presented in this paper. The two methods provide stable learning and, therefore, a greater possibility of good performance.

References

[1] O. Axelsson, Iterative Solution Methods, Cambridge University Press, New York, (1996).

[2] M.E. Brewster and R. Kannan, Nonlinear successive over-relaxation, Numer. Math., 44, 309–315, (1984).

[3] J.E. Dennis and R.B. Schnabel Numerical Methods for Unconstrained Optimization and nonlinear equations, Prentice-Hall, Englewood Cliffs, NJ, (1983).

[4] S.E. Fahlman Faster-learning variations on back–propagation: an empirical study. In Proceedings of the 1988 Connectionist Models Summer School, D.S. Touretzky, G.E. Hinton, and T.J. Sejnowski, eds., pp. 38–51, Morgan Kaufmann, San Mateo, CA, (1989).

[5] R.A. Jacobs, Increased rates of convergence through learning rate adaptation, Neural Networks, 1, 295–307, (1988).

[6] G.D. Magoulas, M.N. Vrahatis, and G.S. Androulakis, Effective back–propagation with variable stepsize, Neural Networks, 10, 69–82, (1997).

[7] G.D. Magoulas and M.N. Vrahatis, A model of local convergence analysis of batch-type training algorithms with adaptive learning rates. In Recent Advances in Circuits and Systems, N. Mastorakis, ed., World Scientific, (1998).

[8] M.F. Møller, A scaled conjugate gradient algorithm, for fast supervised learning, Neural Networks, 6, 525–533, (1993).

[9] J.M. Ortega and W.C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, NY, (1970).

[10] M. Pfister and R. Rojas, Speeding-up backpropagation - A comparison of orthogonal techniques, In Proceedings of the Joint Conference on Neural Networks, Nagoya, Japan, 517–523, (1993).

[11] M. Riedmiller and H. Braun, A direct adaptive method for faster backpropagation learning: the Rprop algorithm. In Proceedings of the IEEE International Conference on Neural Networks, San Francisco, CA, 586–591, (1993).

[12] D.E Rumelhart., G.E. Hinton, and R.J. Williams, Learning Internal Representations by Error Propagation, In D.E. Rumelhart and J.L. McClelland, eds., Parallel Distributed Processing: Explorations in the Microstructure of Cognition, 1, pp. 318–362. MIT Press, Cambridge, Massachusetts, (1986).

[13] F. Silva and L. Almeida, Acceleration techniques for the back–propagation algorithm, Lecture Notes in Computer Science, 412, 110–119. Springer-Verlag, Berlin, (1990).

[14] A. Sperduti and A. Starita, Speed up learning and network optimization with extended back-propagation, Neural Networks, 6, 365–383, (1993).

[15] G.W. Stewart, *Introduction to Matrix Computations* Academic Press, New York, (1973).

[16] T.P. Vogl, J.K. Mangis, J.K. Rigler, W.T. Zink, and D.L. Alkon, Accelerating the convergence of the back-propagation method, Biological Cybernetics, 59, 257–263, (1988).