Iraqi Journal of Statistical Sciences (26) 2014
PP[1-11]

Learning rate for the back propagation algorithm based on modified scant equation

Dr.Khalil K. Abbo* Marwa S. Jaborry**

Abstract

The classical Back propagation method (CBP) is the simplest algorithm for training feed-forward neural networks. It uses the steepest descent direction with fixed learning rate α to minimize the error function E, since α is fixed for each iteration this causes slow convergence for the CBP algorithm. In this paper we suggested a new formula for computing learning rate α_k, using modified secant equation to accelerate the convergence of the CBP algorithm. Simulation results are presented and compared with other training algorithms.

عامل تعلم جديد لخوارزمية الانتشار العكسي مستند الى معادلة القاطع المطورة

الملخص

تعد خوارزمية الانتشار العكسي القياسية ابسط خوارزمية لتعليم الشبكات العصبية ذوات التغذية الامامية، يستخدم اتجاه الانحدار السلبي مع عامل تعليم ثابت في كل تكرار لتصغير دالة الخطأ. لأن عامل التعليم ثابت في كل تكرار وهذا يسبب بطء تقارب الخوارزمية. في هذا البحث اقترحنا صيغة جديدة لحساب عامل التعليم باستخدام معادلة القاطع المطورة لتعجيل تقارب خوارزمية الانتشار العكسي القياسية.وقد عرضت نتائج المحاكاة وقورنت مع خوارزميات تعليم اخر.

*Assistant Professor / Department of Math / College of Computer Science and Mathematics / University of Mosul.
** Researcher/ Department of Math / College of Computer Science and Mathematics / University of Mosul.

Received Date 10/6/2013 Accept Date 16/9/2013

1.Introduction

Neural networks are composed of simple elements operating in parallel. These elements are inspired by biological neurons systems. As in nature, the network function is determined largely by the connections between elements. We can train a neural network to perform a particular function by adjusting the values of the connections(weights), between elements, commonly neural networks are adjusted, or trained so that a particular input leads to as specific target output. The network is adjusted,

based on a comparison of the output and the target, until the network output matches the target. Typically many such input/target pairs are used in this supervised learning to train a network. Batch training of network proceeds by making weight and bias changes based on an entire set (batch) of input vectors [6].

The batch training of the Multi-layer Feed-forward Neural network (MFFN) can be formulated as a non-linear unconstrained minimization problem [8, 9] .Namely

$$\min\ E(w_k),\ w_k \in R^n\ . \qquad (1)$$

where E is the batch error measure defined as the sum of squared differences Error functions over the entire training set , defined by

$$E(w)=\frac{1}{2}\sum_{p=1}^{P}\sum_{j=1}^{N_M}(o_{j,p}^{M}-t_{j,p})^2 \qquad (2)$$

where $(o_{j,p}^{M}-t_{j,p})^2$ is the squared differences between the actual j-th output layer neuron for pattern P and the target output value. The scalar P is an index over input-output pairs, the general purpose of the training is to search an optimal set of connection weights in the manner that the error of the network output can be minimized.

The most popular training algorithm is the Classical Batch Back Propagation (CBP) introduced by Rumelhart, Hinton and Williams[12]. Although the CBP

algorithm is a simple learning algorithm for training Multi-layer Feed-Forward MFF networks, unfortunately it is not based on a sound theoretical basis and is very inefficient and unreliable. One iteration of the CBP algorithm can be written

$$w_{k+1} = w_k - \alpha_k g_k \quad (3)$$

Where w_k is the vector of current weights and biases, $g_k = \nabla E(w_k)$ and α_k is the learning rate, with CBP the learning rate is held constant throughout training. The performance of the algorithm is very sensitive to the proper setting of the learning rate [5].

In order to overcome to the drawbacks of the CBP algorithm many gradient based training algorithms have been proposed in the literature [1, 2, 5, 7, 13] .

2. Some Modifications on CBP.

A surprising result was given by Brazilian and Brownie [3], which gives formula for the learning rate α_k and leads to super linear convergence. The main idea of BB method is to use the information in the previous iteration to decide the step size (learning rate) in the current iteration. The iteration (3) is viewed as

$$w_{k+1} = w_k - D_k g_k \quad (4)$$

where $D_{k+1} = \alpha_k I$. In order to force matrix D_{k+1} having certain quasi-Newton (QN) property, is reasonable to require either

$$\min \left\| s_k - D_{k+1} y_k \right\|_2 \quad (5)$$

Or

$$\min \left\| D_{k+1}^{-1} s_k - y_k \right\|_2 \quad (6)$$

where $s_k = w_{k+1} - w_k$ and $y_k = g_{k+1} - g_k$. Solving equation (5) or (6) for α_k they obtained

$$\alpha_k^{BB1} = \frac{s_k^T y_k}{y_k^T y_k} \quad (7)$$

or

$$\alpha_k^{BB2} = \frac{s_k^T s_k}{s_k^T y_k} \qquad (8)$$

respectively. Note that we abbreviate the method defined in equation(3) with learning rate defined in equations (7) and (8) as BB1 and BB2 methods, respectively.

An alternative approach is based on the work of Plagianakos et al [11]. Following this approach, equation (3) is reformulated to the following Scheme:

$$w_{k+1} = w_k - Bg_k \qquad (9)$$

where $B = diag[\lambda_1, \lambda_2, ..., \lambda_n]$ and $\lambda_i, i = 1, ..., n$ are eigen values for the $\nabla^2 E(w_k)$, or approximations to the Eigen-values for $\nabla^2 E(w_k)$. A well known difficulty to this approach is that the computation of the Eigen values or estimating them is not asimple task, hence the schema defined in equation(9) is not practical .

3. A New Efficient Monotone Learning rate

Due to the unexpected theoretical properties and the striking numerical performance of the BB1 and BB2 methods, it inspired lots of researches on the gradient methods [4]. We believe that the main drawback of the BB methods happen when $g_{k+1} \cong g_k$ which leads to $s_k^T y_k \cong 0$ or $y_k^T y_k \cong 0$, hence the algorithm becomes undefined, to overcome this difficulty we will introduce a new formula to compute α_k, our idea is based on using modified secant equation as follows:

consider the matrix B defined by

$$B = diag[\lambda_k^1, \lambda_k^2, ..., \lambda_k^n] \qquad (10)$$

where $\lambda_k^{(i)} \geq 0$ (i=1,…,n) are the eigen-values for the $\nabla^2 E(w_k)$ at iteration k as we know computing $\lambda_k^{(i)}$ is not a simple task, hence we can assume that

$$\alpha_k = \frac{\sum_{i=1}^{n} \lambda_k^i}{n} \qquad (11)$$

furthermore, let

$$M_k = \alpha_k I_{n*n} \tag{12}$$

since $\lambda_k^i \geq 0$ for $\forall\ i$ and from equations (11) and (12), M_k is diagonal, symmetric and positive definite, therefore M_k satisfies the following modified secant equation

$$M_k z_k = s_k \tag{13}$$

There are different values for z_k [14], in this paper we consider the following

$$z_k = y_k + \gamma \frac{\theta_k}{s_k^T u_k} u_k \tag{14}$$

where γ is apositive scalar, $u_k \in R^n$, $s.t\ \mathrm{s_k^T} u_k \neq 0$ and

$$\theta_k = 6(E_k - E_{k-1}) + 3(g_k + g_{k-1})^T s_k \tag{15}$$

To compute the value of α_k in equation (12), we minimize the following Quadratic equation

$$\min q(\alpha),\ \ \alpha \in R \tag{16}$$

$$q(\alpha) = \frac{1}{2}\left\| s_k - M_k z_k \right\|^2 = \frac{1}{2}\left\| s_k - \alpha_k z_k \right\|^2 \tag{17}$$

By computing the derivative of $q(\alpha)$ and letting it to zero we obtain

$$\frac{dq(\alpha)}{d(\alpha)} = -s_k^T z_k + \alpha\, z_k^T z_k = 0 \tag{18}$$

or

$$\alpha_k = \frac{s_k^T z_k}{z_k^T z_k} \tag{19}$$

Using equations (14) and (15) in equation (19) we get

$$\alpha_k = \frac{s_k^T y_k + \gamma\, \theta_k}{y_k^T y_k + 2\gamma \frac{\theta_k}{s_k^T u_k} y_k^T u_k + \gamma^2 \frac{\theta_k^2}{(s_k^T u_k)^2} u_k^T u_k} \tag{20}$$

there are different choices for the vector u in this work, we use $u_k = s_k$, then equation (20) becomes

$$\alpha_k = \frac{s_k^T y_k + \gamma\,\theta_k}{y_k^T y_k + 2\gamma \frac{\theta_k}{s_k^T s_k} s_k^T y_k + \gamma^2 \frac{\theta_k^2}{s_k^T s_k}} \qquad (21)$$

with this choice of α_k we will use the backtracking strategy to ensure that the learning rare α_k satisfies the following Wolfe condition:

$$E(w_k + \alpha_k d_k) \le E(w_k) + \rho \alpha_k g_k^T d_k \qquad (22)$$

$$g_{k+1}^T d_k) \ge \sigma\, g_k^T d_k \qquad (23)$$

where $0 < \rho < \sigma < 1$. At this point we will summarize the new training algorithm, we abbreviate this new algorithm as MSBP

Algorithm MSBP for training FFNN

Step(1): Initiate $k = 1, w_1 \in \mathrm{R}^n, 0 < \rho < \sigma < 1,\ \varepsilon_1, \varepsilon_2 > 0$ and $\gamma \in (0,1)$

Step(2): If $\|g_k\| < \varepsilon_1$ *or* $E(w_k) < \varepsilon_2$ stop else go to step(3).

Step(3): Compute the search direction using steepest descent direction i. e

$$d_k = -g_k$$

Step(4): Compute learning rate α_k, if k=1 then $\alpha_k = \frac{1}{\|g_k\|}$ else

use equation (21) with backtracking strategy to ensure the Wolfe conditions (22) and(23) hold.

Step(5): Update the weigh vector according to the following relation

$$w_{k+1} = w_k + \alpha_k d_k$$

Step(6): Set k=k+1, go to Step(2).

Note:

Below are some observations on which the convergence of MSBP Algorithm rests:

then second Wolfe condition (23) $s_k^T y_k \neq 0$ **1**. In equation (21) if

Ensures $s_k^T y_k > 0$. Since $0 < \sigma < 1$ and $d_k = -g_k$ therefore

$$s_k^T y_k = s_k^T g_{k+1} - s_k^T g_k \geq \sigma\, s_k^T g_k - s_k^T g_k = \frac{-(1-\sigma)}{\alpha_k} g_k^T d_k > 0$$

2.We see from equation (21) that $0 < \alpha_k < 1,\ \forall k$

4. Experiments and Results:

A computer simulation has been developed to study the performance of the learning algorithms. The simulations have been carried out using MATLAB(7.6) the performance of the MSBP has been evaluated and compared with batch versions of the Classical Back Propagation (CBP). The algorithms were tested using the initial weights, initialized by the Nguyen – widrow method [10] and received the same sequence of input patterns . The weights of network are updated only after the entire set of patterns to be learned has been presented .

For each of the test problems, a table summarizing the performance of the algorithms for simulations that reached solution is presented . The reported parameters are min the minimum number of epochs for 50 simulation , mean the mean value of epochs for 50 simulation, Max the maximum number of epochs for 50 simulation, Tav the average of total time for 50 simulation and Succ, the succeeded simulations out of (50) trails within error function evaluations limit.

If an algorithm fails to converge within the above limit considered that it fails to train the FFNN, but its epochs are not included in the statical analysis of the algorithm, one gradient and one error function evaluations are necessary at each epoch.

4.1 Problem (1): (Spect Heart Problem)

This data set contains data instances derived from Cardiac Single Proton Emission Computed Tomography (SPECT) images from the university of Colorado [9]. The network architectures for this medical classification problem consists of one hidden layer with 3 neurons and an output layer of one neuron. The termination criterion is set to $\varepsilon_2 < 0.01$ within the limit of 2000 epochs,

table(1) summarizes the resuls of all algorithms i.e for 50 simulations the minimum epochs for each algorithm are listed in the first column (Min), the maximum epoch for each algorithm are listed in the second column, third column contains (Mean) the mean value of epochs and (Tav) is the average of time for 50 simulations and last column contains the percentage of succeeds of the algorithms in 50 simulations.

Table(1): Results of simulations for the Heart problem

Algorithms	Min	Max	Mean	Tav	Succ
CBP	-----	-----	-----	------	2 %
BB1	21	68	33.1	0.75	100 %
BB2	45	113	79.74	1.1852	100 %
MSBP	20	59	32.	0.734	100 %

Form table (1), we note that the algorithm MSBP is the best algorithm with respect to the epochs number and the time.

4.2 Problem (2): Continuous Function Approximation:

The second test problem we consider is the approximation of the continuous trigonometric function: $f(x) = \sin(x) * \cos(3x)$. The network architecture for this problem is 1-15-1 FNN (thirty weights, sixteen biases) is trained to approximate the function f(x), where $x \in [-\pi, \pi]$ and the network is trained until the sum of the squares of the errors becomes less than the error goal 0.002, comparative results are shown in table (2).

Table(2): Results of simulations for the function approximation problem

Algorithms	Min	Max	Mean	Tav	Succ
CBP	fail	--	--	--	0.0%
BB1	92	382	184.7	2.2076	100%
BB2	973	1912	---	---	85%
MSBP	96	364	182.02	2.14	100%

Form table (2), we conclude that the algorithm MSBP is the beast algorithm with respect to the succeeded simulations, number of epochs and the time.

4.3 Problem (3):(XOR Problem)

The last problem we have been encountered with is the XOR Boolean function problem, which is considered as a classical problem for the FFNN training . The XOR function maps two binary inputs to a single binary output. As it is well known this function is not linearly separable. The network architectures for this binary classification problem consists of one hidden layer with 3 neurons and an output layer of one neuron. The termination criterion is set to $\varepsilon_2 \leq 0.002$ within the limit of 1000 epochs, and table(3) summarizes the result of all algorithms i.e for 50 simulations the minimum epochs for each algorithm are listed in the first column (Min), the maximum epochs for each algorithm are listed in the second column, third column contains (Mean) the mean value of epochs and (Tav) is the average of time for 50 simulations and last columns contain the percentage of succeeds of the algorithms in 50 simulations

Table(3): Results of simulations for the XOR function

Algorithms	Min	Max	Mean	Tav	Succ
CBP	Fail	--	--	--	0.0%
BB1	7	43	23.9	0.4904	100%
BB2	19	2674	134.9	1.618	72%
MSBP	17	32	24.45	0.5151	100%

Form table (3), we conclude that the algorithm BB1 is the beast algorithm with respect to the succeeded simulations, number of epochs and the time.

REFFRNCES

[1] Abbo K. and Hind M.(2012) 'Improving the learning rate of the Backpropagation Algorithm Aitken process'. Iraqi Journal of the statistical sciences, accepted (to appear).

[2] Abbo K. and Zena T.(2012) 'Minimization algorithm for training feed-forward neural etworks'. J. of Education and Sci. (to appear).

[3] Brazilia J and Brownie M.(1988)'Tow point step- size gradient methods ' IMA. Journal of Numerical Analysis, 8.

[4] Fletcher R (2001)'On the Barzilai-Borwein method' Research Report, University of Dunde, UK.

[5] Gong L., Liu G., Li Y. and Yuan F. (2012) 'Training Feed- forward Neural Networks Using the gradient descent method with optimal Step size, J. of Computational Information Systems 8:4

[6] Hertz J., Krogh A .and Palmer R .(1991) 'Introduction to the theory of Neural computation'. Addison-Wesley ,Reading , MA .

[7] Jacobs R (1988) 'Increased rates of convergence through learning rate adaptation' .Neural Networks , vol. 1, no.4.

[8] Kostopoulos A. Sotiropoulos D. and Grapsa T. (2004). "A new efficient learning rate for Perry's spectral conjugate gradient Training method",1st International Conference ' From Scientific Computing to Computational Engineering'. 1st IC-SCCE. Greece.

[9] Livieris I. and Pintelas R. (2011). "An advanced conjugate gradient training algorithm based on a modified secant equation", Technical Report NO. TR11-03. University of PatrasDepartment of Mathematics, Patras, Greece.

[10] Nguyen D. and Widrow B. (1990). "Improving the learning speed of 2-layer neural network by choosing initial values of the adaptive weights", Biological Cybernetics, 59:

[11] Plagianakos V., Magoulas G., and Vrahatis M. (2002) ' Determing non-monotone strategies for effective training of multi-layer perceptrons'. IEEE Transactions on Neural Networks, 13(6).

[12] Rumelhart D., Hinton G. and Williams R (1986) 'Learning representations by back-propagation errors' Nature,32

[13] Sotirpoulos D., Kotsiopoulos A and Grapsa T.(2004) 'training neural networks using two point step-size gradient methods'. International conference of numerical Analysis andApplied Mathematics. Patras, Greece.

[14] Zhang H. and Hager W. (2006) A survey of nonlinear conjugate gradient methods. Pacific journal of Optimization, 2.