Iraqi Journal of Statistical Science (20) 2011
The Fourth Scientific Conference of the College of Computer Science & Mathematics
pp [164-173]

Conjugate Gradient Back-propagation with Modified Polack –Rebier updates for training feed forward neural network

Dr. Abbas Al-Bayati* **Dr. Khalil K. Abbo**** **Ibrahem A. Saleh*****

Abstract

Several learning algorithms for feed-forward (FFN) neural networks have been developed, many of these algorithms are based on the gradient descent algorithm well-known in optimization theory which have poor performance in practical applications. In this paper we modify the Polak-Ribier conjugate gradient method to train feed forward neural network. Our modification is based on the secant equation (Quasi-Newton condition). The suggested algorithm is tested on some well known test problems and compared with other algorithms in this field.

المستخلص

طورت عدة خوارزميات التعليم للشبكات العصبية ذوات التغذية الأمامية وتستند كثير من هذه الخوارزميات الى خوارزمية الانحدار السلبي، ومن المعروف في نظرية ألأمثلية انها ليست ذات كفاءة في التطبيقات العملية. في هذا البحث نحاول تطوير طريقة بولاك--ريبيير للمتجهات المترافقة لتعليم الشبكة العصبية ذات التغذية الأمامية، وتطويرنا استند الى معادلة القاطع (شرط شبيه نيوتن). الخوارزمية المقترحة اختبرت لبعض مسائل ألاختبار المعروفة وقورنت مع بعض الخوارزميات المعروفة في هذا المجال.

Introduction

Several learning algorithms for Feed-Forward (FFN) Neural networks have been developed for solving function approximation, pattern recognition, and other well known problems. Many of these algorithms are based on the gradient descent algorithm [9, 15] well known in optimization theory. They usually have a poor convergence rate and

* **Prof. \ College of Computers Sciences and Math.\ University of Mosul**
** **Assit.Prof. \ College of Computers Sciences and Math.\ University of Mosul**
*** **Lecture. \ College of Computers Sciences and Math.\ University of Mosul**

Received:1/10 /2011 Accepted: 21 /12 / 2011

depend on parameters which have to be specified by the users, because no theoretical basis for choosing them exists. The values of these parameters are often crucial for the success of the algorithm. One of these algorithms is the standard Backpropagation (BP) [14].

Although BP is the most common and widely used supervised algorithm, nevertheless because of the user depended parameters, it is usually inefficient on large scale problems.

The neural network training problem can be formulated as a non linear unconstrained optimization problem. So the training process can be realized by minimizing the error function E defined by:

$$E(w,b) = \frac{1}{2}\sum_{i=1}^{n_l}(d_i - x_i^l)^2 \quad (1)$$

Where x^l is a function of w (the weight vector), b (the bias) and d is the target through the equations of the forward pass. This cost function measures the squared error between the desired and actual output vectors. The general purpose of the training is to search an optimal set of connection w in the manner that the error of the network output can be minimized.

The CBP algorithm

Lets diagram the network as

$$x^\circ \xrightarrow{w^1,b^1} x^1 \xrightarrow{w^2,b^2} \ldots\ldots\ldots\ldots \xrightarrow{w^l b^l} x^l$$

Where $x^l \in R^{n_l}$ for all $l = 0,\ldots\ldots\ldots,L$ and w^l is an $n_l * n_{l-1}$ matrix for all $l = 1,\ldots,L$, b^l is the bias for each $l = 1,\ldots,L$ there are $L+1$ layers of neurons, and L hidden layers, we would like to change the weights w and biases b so that the actual output x^l becomes closer to the desired out put d.

The Backpropagation algorithm consists of the following steps.

1- Forward pass. The input vector x° is transformed into the output vector x^l, by evaluating the equation

$$x_i^l(k) = f(u_i^l) = f(\sum_{j=1}^{n_{l-1}} w_{ij}^l x_j^{l-1} + b_i^l)$$

For l=1 to L, and k is index of iteration usually called epoch

2- Error computation .The difference between the desired output d and actual output x^L is computed

$$\delta_i^L(k) = f'(u_i^L)(d_i - x_i^L) \tag{2}$$

3- Backward pass. The error signal at the output units is propagated Backwards through the entire network, by evaluating

$$\delta_j^{l-1}(k) = f'(u_j^{l-1})\sum_{i=1}^{n_l}\delta_i^l w_{ij}^{\ l} \quad , \quad \text{from l=L to 1} \tag{3}$$

4- Learning updates. The weights and biases are updated using the results of the forward and backward passes. Compute the gradient vector $g_k = \nabla E(k) = \dfrac{\partial E}{\partial w_{ij}^l(k)}$ and learning rate α_k and update the weights and biases and set $d_k = -g_k$, $\alpha \in (0,0.5)$

$$w_{ij}^l(k+1) = w_{ij}^l(k) + \alpha\ d_k \tag{4}$$

$$b_i^l(k+1) = b_i^l(k) - \alpha\ \frac{\partial E}{\partial b_i^l} \tag{5}$$

K=k+1; go to step(1).

We see from step(4) the BP algorithm uses the Steepest Descent (SD) search direction with fixed step-size or (learning rate) α in order to perform the minimization of the error function E, the iteration form of the SD algorithm is:

$$\begin{aligned} d_k &= -g_k \\ w_{k+1} &= w_k + \alpha_k d_k \end{aligned} \tag{6}$$

Where k is the current iteration usually called epoch, $w_0 \in R^n$ is given initial weight vector, α_k is learning rate and $d_k = -g_k$ for all k and g the gradient vector of the error function E that is $g = \nabla E(w)$, E represents the batch error measure defined as the sum of squared differences error function over the entire training set.

The inefficiency of SD is due to the fact that the minimization direction and learning rate are chosen poorly; if the step-size does not lead directly to the minimum, steepest descent will zig-zag with many small steps[15]. In the literature there have been proposed many suggestions [3, 4] to define the search direction d_k and most of them use second order information , for example the Quasi-Newton(QN) methods are one of the most exploited methods.

The following iterative method can be seen as general QN procedure to solve the problem(1):

Calculate the search direction d_k by solving the equation

$$d_k = -H_k g_k \tag{7}$$

Then set

$$w_{k+1} = w_k + \alpha_k d_k \tag{8}$$

where, H_k is the secant approximation to the inverse $(\nabla^2 E(w_k))^{-1}$ and α_k is updated by line search. The matrix H_k is usually required to positive definite to ensure a descent direction i.e

$$g_k^T d_k < 0 \tag{9}$$

H_k is updated at every iteration to a new inverse Hessian approximation H_{k+1} for which the general QN equation

$$H_{k+1} y_k = s_k \tag{10}$$

Where $s_k = w_{k+1} - w_k$ and $y_k = g_{k+1} - g_k$, is satisfied [6] . By extending the secant condition (10),Wei in [17] proposed the modified secant condition

$$H_{k+1} \bar{y} = s_k \tag{11}$$

Where

$$\bar{y} = y_k + A_k s_k \tag{12}$$

Where A_k is a simple symmetric and positive matrix.

Despite the theoretical and super linear rate of convergence advantages of Quasi-Newton methods, the main drawback of these methods is the use of the inverse Hessian or an approximation to it and this requires more storage hence not suitable for large-scale problems.

The conjugate gradient method is one choice for solving large scale problems, because it does not need any matrices [11] or [18], In this work we will propose a new conjugate gradient algorithm for training feed forward neural network.

The remainder of this paper is organized as follows, section 2 presents a brief summary of conjugate gradient. In section 3 the new conjugate gradient method presented and finally section 4 deal with our experimental results.

2- Conjugate Gradient Methods(CG)

In Conjugate gradient methods the basic idea for determining the search direction d_k in step(4) Eq.(4) is the linear combination of the negative gradient at the current iteration with the previous search direction namely

$$d_{k+1} = -g_{k+1} + \beta_k d_k \qquad , d_1 = -g_1 \tag{13}$$

In the literature there have been proposed several choices for defining the scalar parameter β_k which gives rise to distinct conjugate gradient methods[16]. The most famous ones were proposed by Fletcher-Reeves (FR) [7] defined as:

$$\beta^{FR} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k} \tag{14}$$

And Polak – Ribiere (PR) [12] written as:

$$\beta^{PR} = \frac{g_{k+1}^T y_k}{g_k^T g_k} \tag{15}$$

If the objective function E is strongly convex quadratic, then in theory the two choices for the update parameter β_k are equivalent with an exact line search i.e. the learning rate α_k computed as:

$$\frac{dE(w_k + \alpha\, d_k)}{d\alpha} = g_{k+1}^T d_k = 0 \tag{16}$$

For non-quadratic cost functions, each choice for β_k leads to different performance. Despite the strong convergence theory that has been developed for the Fletcher-Reeves β^{FR} , this method is susceptible to jamming that is to take small steps without making significant progress to the minimum see [8]. In general, the performance of Polak-Ribiere β^{PR} is better than the performance of β^{FR} method. On the other hand, Powell showed that [13] with an exact line search, PR method could cycle infinitely, without converging to a stationary point. In summary, the convergence of the PR method for general non-linear function is uncertain [8], the reason for this uncertainty is $\beta^{PR} < 0$ in some cases. In each conjugate gradient(CG) iteration, the step size (learning rate) α_k is chosen to yield an approximation minimum for the problem

$$\min_{\alpha \geq 0} E(w_k + \alpha\, d_k) \tag{17}$$

Since $\alpha_k > 0$, the direction d_k should satisfy the descent condition eq.(9) for all k. If there exists a constant c >0 such that

$$g_k^T d_k < -c\|g_k\|^2 \quad , \ c \in (0,1) \tag{18}$$

For all k, then the search direction satisfies the sufficient descent condition. The termination conditions for CG line search are often based on some versions of the Wolfe conditions. The standard Wolfe conditions [2] are

$$E(w_k + \alpha_k d_k) - E(x_k) \le \delta \alpha_k g_k^T d_k \tag{19}$$

$$g_{k+1}^T d_k \ge \sigma\, g_k^T d_k \tag{20}$$

Where d_k is descent direction and $0 < \delta < \sigma < 1$.

3- Back propagation with Modified Polak-Ribiere updates

As mentioned in the previews section the main drawback for PR method is $\beta^{PR} < 0$ in some cases, to overcome this drawback we incorporate the second other information of the objective function E by using equations (11) and (12) that is

$$H_{k+1}\bar{y} = s_k \tag{21}$$

Where H_{k+1} is an approximation to the inverse $\nabla^2 E(w_{k+1})$, and

$$\bar{y}_k = y_k + A_k s_k \tag{22}$$

Where A_k is any symmetric and positive definite, since A_k is symmetric and positive definite we can take A_k as diagonal matrix i.e.

$$A_k = \theta_k I_{n*n} \tag{23}$$

There exist different values for θ_k $(\theta_k > 0)$ for example we may take

$$\theta_k = \frac{s_k^T y_k}{s_k^T s_k}, \qquad [4] \tag{24}$$

$$Or \quad \theta_{kl} = \frac{s_k^T s_k}{s_k^T s_k + \gamma_k s_k^T y_k} \quad , \gamma_k \in [0,1], \qquad [1] \tag{25}$$

Therefor we can modify PR (β^{MPR}) by using equations (11), (18) and (24) or (25)

$$\beta^{MPR} = \frac{g_k^T + \bar{y}_k}{g_k^T g_k} = \frac{g_{k+1}^T y_k + \theta_k g_{k+1}^T s_k}{g_k^T g_k} \tag{26}$$

Then the new search direction is

$$d_{k+1} = -g_{k+1} + \beta_k^{MPR} d_k \tag{27}$$

Where β_k^{MPR} as in equation (26).

Then our modified (MPRBP say) algorithm can be written as:

Modified PR Backpropagation (MPRBP) algorithm

The steps (1), (2) and (3) are same as CBP algorithm and step(4) changes to the following form

Step(4):

1. Initialization: use Nguyen widrow method to initialize the weights and Biases and set k=1, $gaol = err$, $\varepsilon > 0$ and compute $g_k = \nabla E(w_{ij}^l(k)$ And $E(w^l(k))$, $1 = 1,..,L$, $d_k = -g_k$, $\alpha_k = 1/\|g_k\|$
2. If $E(w_k^l) < err$ or $\|g_k\| < \varepsilon$ stop $w_{ij}^l(k)$, $l = 1,..,L$ is optimal else goto 3
3. learning rate computation: compute α_k by line search procedure such that Wolfe conditions (19) and (20) are satisfied and update the weights and biases according to the
   $w_{ij}^l(k+1) = w_{ij}^l(k) + \alpha_k d_k$
   $b_i^l(k+1) = b_i^l(k) + \alpha_k d_k$
4. Direction computation: compute g_{k+1}, β_k^{MPR} and set
   $d = -g_{k+1} + \beta_k^{MPR} d_k$, if Powell restart is satisfied then set $d_{k+1} = -g_{k+1}$
   Else $d_{k+1} = d$; k=k+1 go to 2

3. 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. The performance of the modified Polak-Rebier(MPRBP)has been evaluated and compared with batch versions of BP, constant learning BP (CBP) , Fletcher-Revees (FR) and Polak-Rebier (PR) these algorithms known as (traingd), (traincgf) and (traincgp) respectively in the neural net work toolbox. Toolbox default values for the heuristic parameters of the above algorithms are used unless stated otherwise. The

algorithms were tested using the same initial weights, initialized by the Nguyen-Widrow method [11] and received the same sequence of input patterns. The weights of the 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, mean the mean value of epochs, max the maximum number of epochs, Tav the average of total time and succ. The succeeded simulations out of (100) trials within the error function evaluations limit.

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

3.1 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 [8]. The network architectures for this medical classification problem consists of one hidden layer with 6 neurons and an output layer of one neuron. The termination criterion is set to $E \leq 0.1$ within limit of 1000 epochs, table(1) summarizes the result of all algorithms i e for 100 simulations the minimum epoch for each algorithm is listed in the first column (Min), the maximum epoch for each algorithm is listed in the second column, third column contains (Tav) the average of time for 100 simulations and last columns contains the percentage of succeeds of the algorithms in 100 simulation.

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

Algorithms	Min	Max	Mean	Tav	Succ
CBP	fail	--	--	--	0.0%
FR	31	70	31.870	0.0911	100%
PR	20	56	31.066	0.0701	100%
MPRBP with [4]	18	59	31.850	0.0833	100%
MPRBP with [1]	21	52	28.860	0.0657	100%

3.2 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 architectures 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 0.001. The network is based on hidden neurons of logistic activations with biases and on a linear output neuron with bias. Comparative results are shown in table (2). Figure (1) shows performance of SBP.

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

Algorithms	Min	Max	Mean	Tav	Succ
CBP	Fail	---	---	---	---
FR	66	292	116.60	0.5229	100%
PR	61	208	115.27	0.4874	100%
MPRBP with [4]	66	216	111.13	0.4861	100%
MPRBP with [1]	54	149	104.14	0.4660	100%

REFERENCES

[1] Abbo .K. (2007). "Modifying of Barzilai and Borwein Method for solving large scale unconstrained optimization problem", Iraqi J. of statical science Vol. 11, No.7.

[2] Andrei N.(2010)"New accelerated conjugate gradient algorithms as a modification of Dai- Yuan´s Computational scheme for unconstrained optimization" J. of computational and Applied Mathematics 234.

[3] Apostolopulou M., Sotiropoulos D., Livieris I and Pintelas . (2009) 'A memoryless BFGS neural network training Algorithm " 7th IEE International Conference on Industrial Informatics.

[4] Battiti R. (1992)" First and second order methods for learining : between steepest descent and Newtons method ", Neural computation(4).

[5] Birgin, E. and Martinez, M. (2001), "A spectral conjugate gradient method for unconstrained optimization", Applied Math and Optimization, vol. (43).

[6] Farzin M., Malik H. and Wah J. (2009)," Memoryless Modified

symmetric Ranlc.One Method for Lerge– Scale Unconstrained Optimization" Americ Journal of Applied Sciences G(12).

[7] Fletcher, R. and Reeves, G. (1964), "Function minimization by Gradients". Computer. Journal , Vol. 7.

[8] Hager W. and Zhany H. (2006),"A surrey of non-linear conjugate gradient methods" Pacific J. Optim.2.

[9] Livieris I., Sotiropoulos D. and Pintelas P. (2009),"On Descent spectral CG algorithms for Training Recurrent Neural Network", IEEE Computer Society. 13th Panhellenic Conference on Informatics.

[10] Nguyen D. and Widrow B. (1990), "Improving the learning speed of 2-layer neural network by choosing initial values of the adaptive weights", IEEE First International Jaint Conference on Neural Networks 3.

[11] Nocedal J. and Wright S.(1999),"Numerical Optimization" Springer Verlag, New York.

[12] Polak, E. and Ribiere,G. (1969), " Not sur la convergence de Directions conjugate". Rev. Franaisse Informants. Research operational, 3e Anne., 16.

[13] Powell M.D. (1984)," Nonconvex minimization Calculations and the Conjugate Gradient Method" Numerical Analysis (Dundee. 1983), Leckux Notes in Mathematics Vol. 1066.

[14] Rumelhart D. and Mcclelland J.(1986),"parallel distributed processing explorations in the microstructure of Cogniton. Vol.1: Foundations. MTT press.

[15] Sotiropoulos D., Kostopoulos A. and Grapasa T.(2002)"A spectral version of perry´s Gonjugate Gradient method for Neural Network Training. 4th GRACM Congress on computational Mechanics. Vol(1).Patra.

[16] Wei C. and Wang k. (2010),"Global Convergonce of a new Conjugate gradient method for modified liu-Story formula. J. of East China Normal University No.(1).

[17] Wei Z., Li G. and Qi L. (2006)," New Quasi-Newton methods for unconstrained optimization problems". J.Applied Math. Comput., 175.

[18] Yabe H. and Sakaiwa N. (2005)," A New Non-linear Conjugate gradient method for unconstrained optimization. J. of the Operations Research Society of Japan. vol.48. No.4.