A STUDY OF ACCELERATION TECHNIQUES IN TRAINING NEURAL NETWORKS – LOCAL ADAPTIVE TECHNIQUES

Norpah Mahat
Faculty of Computer and Mathematical Sciences
Universiti Teknologi MARA (UiTM) Cawangan Pulau Pinang

ABSTRACT

Till today, it has been a great challenge in optimizing the training time in neural networks. This paper presents Local Adaptive Techniques and Dynamic Adaptation Methods as acceleration techniques for neural networks. The first technique is based on weight-specific information, such as the temporal behavior of the partial derivative of the current weight. The second technique is dynamically adapts the momentum factor, α, and learning rate, η, with respect to the iteration number or gradient. Some of the most popular learning algorithms are described and discussed. Simulations on a real world application problem are conducted to evaluate and compare the performance of a local adaptive strategies with various popular training algorithms include global adaptive strategies. These techniques have been compared and measured in terms of gradient and error function evaluations, and percentage of success.

Keywords: *Neural Networks, Local Adaptive Techniques, Dynamic Adaptation Methods*

Introduction

The Backpropagation(BP) algorithm is the most used type and standard algorithm for training multilayerd feedforward artificial neural networks. However, in some cases, the BP takes a long time to adapt the weights between the units in the network to minimize the mean squared errors between the desired outputs and the actual network outputs. But some

ISSN 1675-7939

problem can be solved by using BP. For example the research from Baboo S.S. & Shereef I.K. (2010) indicate that the BP based weather forcast have shown an improvement in real time processing of weather data.

The issue of changing the learning rates, η, dynamically during training has been widely investigated and several techniques for learning rate adaptation have been proposed so far in acceleration strategies of BP neural networks. The use of these strategies aim at finding the proper learning rate that substitute for a small magnitude of the gradient in a flat region and decrease a large weight changes in a highly deep region. The algorithms for these techniques employ heuristics strategies to adapt the learning rate at each iteration and require fine tuning for other learning parameters that help to ensure a minimization of the error function along each weight direction.

There has been much research based on learning rate parameter. Recently, Otair.M.A & Salameh.W.A. (2005) proposed a very small value for learning rate with Optical Backpropagation makes the adapted final weights very closed become to the final weights that introduced from backpropagation. So, it can escape from local minimum. Another, a differential adaptive learning rate method (DALRM) have been proposed by Iranmanesh.S & Mahdavi M.A. (2009). This method escape from local minima by using a large learning rate at first and then gradually reduce the learning rate. This way reduced the network error in a short time and the network's learning speed has highly increased. Omaima (2010) designed the Backpropagation Neural Networks (BPNN) image compression system and the performance of the system can be increased by modifying the network itself, learning parameters and weights.

These adaptive learning strategies can be divided into two categories, Global and Local Adaptive Techniques (Riedmiller, M. & Braun, H., 1993). This paper concentrates on Local Adaptive Techniques, namely, Learning Rate Adaptation by Sign Changes, SuperSAB, Delta-Bar-Delta Rule, Quickprop and Rprop. Another technique is known as Dynamic Adaptation Methods (Zainuddin Z. & Evans D.J., 1997), which dynamically adapts the momentum factor and learning rate. These techniques have been compared and measured in terms of gradient and error function evaluations, and percentage of success.

Backpropagation Algorithm

The error signal, $e_j(n)$ at the output of neuron j at iteration n is defined by

$$e_j(n) = d_j(n) - y_j(n) \tag{1}$$

where $d_j(n)$ refers to the desired response for neuron j and is used to compute $e_j(n)$ and $y_j(n)$ refers to the function signal appearing at the output of neuron j at iteration n.

The objective of back-propagation algorithm is to minimize $e_j(n)$ so that the desired response will be close to the actual response.

We define the instantaneous value of the error energy for neuron j as $\frac{1}{2}e^2{}_j(n)$. Correspondingly, the instantaneous value $\xi(n)$ of the total error energy is obtained by summing $\frac{1}{2}e^2{}_j(n)$ over all neurons in the output layer. We may thus write

$$\xi(n) = \frac{1}{2}\sum_{j\in C} e_j^2(n) \tag{2}$$

where the set C includes all the neurons in the output layer of the network. Let N denote the total number of patterns (examples) contained in the training set. The average squared error energy is obtained by summing $\xi(n)$ over all n and then normalizing with respect to the set size N, as shown by

$$\xi_{av} = \frac{1}{N}\sum_{n=1}^{N} \xi(n) \tag{3}$$

The objective of the learning process is to adjust the free parameters (i.e. synaptic weights and bias levels) of the network to minimize ξ_{av}. To do this minimization, the weights are updated on a pattern-by pattern basis until one epoch, that is, one complete presentation of the entire training set. The arithmetic average of these individual weight changes over the training set is therefore an estimate of the true change as would result from modifying the weights based on minimizing the cost function ξ_{av} over the entire training set. In its most basic form, it is a simple gradient optimization procedure:

$$w_{ji}(n+1) = w_{ji}(n) - \eta\partial\xi/\partial w_{ji} \tag{4}$$

In the batch mode variant the descent is based on the gradient $\Delta\xi$ for the total training set:

$$\Delta w_{ji}(n) = -\eta * \frac{\partial \xi}{\partial w_{ji}} + \alpha * \Delta w_{ji}(n-1) \quad (5)$$

where ξ is the cost function being minimized, w_{ji} is a generic weight in the network, α is a momentum factor and η is the learning rate or step size parameter.

Local Adaptive Techniques

Many techniques have been proposed to date to deal with problems of gradient descent. These techniques can be roughly be divided into two categories, Global and Local adaptive techniques. Global techniques are algorithms that use global knowledge of the state of the entire network, such as the direction of the overall weight-update vector. For example, a class of global algorithms are Steepest Descent and Conjugate Gradient (CG) methods. The CG methods include Fletcher Reeves, Powell Beale and Polak Ribiere method.

Local adaptation strategies are based on weight specific information only, which means that they use an independent learning rate, η, for every adjustable parameter (every connection). Therefore they are able to find an optimal learning rate for every weight. Some of the local adaptive techniques will discussed below.

Sign Changes

Learning rate adaptation by sign changes (Silva and Almeida, 1990) will adapt the step size using a separate learning rate, η_{ji} for each connection. The adaptation is done by observing the signs of the last two gradients. As long as no change in sign is detected, the corresponding learning rate is increased. If the sign changes, the learning rate is decreased.

If, in two successive iterations, the updates of x (or equivalently, the gradient values) have opposite signs, that means that we have "jumped over a minimum" and that the step size is too large. On the other hand, if two successive signs are equal, it appears that we could have moved somewhat faster, while still not passing beyond the minimum. The basic heuristic for adaptation is then simply to decrease the step size of two successive updates that have opposite signs, and to increase it if they

have the same sign. It was proposed to use a linear step size increase, and an exponential step size decrease. The step size update is according to the following:

$$\eta_{ji}(n)=\eta_{ji}(n-1)*u, \qquad \text{if } \frac{\partial\xi}{\partial w_{ji}}(n)*\frac{\partial\xi}{\partial w_{ji}}(n-1)\geq 0 \tag{6}$$

$$\eta_{ji}(n)=\eta_{ji}(n-1)*d, \qquad \text{else.}$$

Then, update the weight according to equation (5).

The choice of proper parameters u and d is easy as long as $u=1/d$ holds. From the simulation that have been tested, the recommended values are 1.1-1.3 for u or 0.7-0.9 for d. They also use a backtracking strategy which restarts an update step if the total error increases. For this restart all learning rates are halved.

Delta Bar Delta Rule

Delta-Bar-Delta algorithm (Jacobs, 1988) controls the learning rates, η, by observing the sign changes of an exponential averaged gradient. Increase the learning rates by adding a constant value instead of multiplying it. Hence,

1. Choose some small initial value for every $\eta_{ji}(0)$.
2. Adapt the learning rates:

$$\eta_{ji}(n)=\eta_{ji}(n-1)+u, \qquad \text{if } \frac{\partial\xi}{\partial w_{ji}}(n)*\frac{\partial\xi}{\partial w_{ji}}(n-1)\geq 0 \tag{7}$$

$$\eta_{ji}(n)=\eta_{ji}(n-1)*d, \qquad \text{if } \frac{\partial\xi}{\partial w_{ji}}(n)*\frac{\partial\xi}{\partial w_{ji}}(n-1)\leq 0$$

$$\eta_{ji}(n)=\eta_{ji}(n-1), \qquad \text{else.}$$

In particular it is difficult to find a proper u. Small values may result in slow adaptations while big ones endanger the learning process. Very different values are recommended for u (5.0,0.095,0.085,0.035) and d (0.9,0.85,0.666).

SuperSAB

Super SAB (Tollenaere, 1990) is also based on the idea of sign independent learning rate adaptation. The basic change is to increase the learning rate, η, exponentially instead of linearly as with Delta-Bar-Delta method. This

is done to take the wide range of temporarily suitable learning rates, η, into account. By using a proper upper limit η_{max}, the algorithm behaves perfectly all over the training period.

Recommended values for u is 1.05 and d is 0.5 and recommended values for η_{max} is between 0 and 1. Hence,

$$\eta_{ji}(n)=\eta_{ji}(n-1)*u, \quad \text{if } \frac{\partial \xi}{\partial w_{ji}}(n)*\frac{\partial \xi}{\partial w_{ji}}(n-1)\geq 0 \wedge \eta_{ji}(n-1)\leq \eta_{\max} \tag{8}$$

$$\eta_{ji}(n)=\eta_{ji}(n-1)*d, \quad \text{if } \frac{\partial \xi}{\partial w_{ji}}(n)*\frac{\partial \xi}{\partial w_{ji}}(n-1)\leq 0$$

$$\eta_{ji}(n)=\eta_{ji}(n-1), \quad \text{else.}$$

Quickprop

This is an optimization of back-propagation based on Newton's method (Fahlman, 1988). It is applicable when, between two steps, the gradient has decreased in magnitude and has changed sign. Then a parabolic estimate of the MSE is used to determine the weights for the next step. Quickprop computes the derivatives in the direction of each weight. After computing the first gradient with regular back-propagation, a direct step of the error minimum is attempted by

$$\Delta x(t)=\frac{f'(x(t))}{f'(x(t-1))-f'(x(t))}\Delta x(t-1) \tag{9}$$

Rprop

Rprop (Riedmiller and Braun, 1993) uses an adaptive version of the "Manhatan-Learning" rule and is a local adaptive learning scheme. The basic principle of Rprop is to eliminate the harmful influence of the size of the partial derivative on the weight step not influenced by the magnitude of the gradient. Only the sign of the derivative is used to find the proper update direction.

Rprop uses independent update step size Δ_{ji} for every connection. Furthermore, these step sizes are adapted with respect to the sign of the actual and the last derivative. The step sizes are bounded by upper and lower limits in order to avoid oscillation and arithmetic underflow of floating point values. Finally, local backtracking is applied to those connections where sign changes of the derivative are detected. Hence,

1. Choose some small initial value for every update step size $\Delta_{ji}(0)$.
2. Adapt the step sizes:

$$\Delta_{ji}(n)=\Delta_{ji}(n-1)*u, \quad \text{if } \frac{\partial\xi}{\partial w_{ji}}(n)*\frac{\partial\xi}{\partial w_{ji}}(n-1)\geq 0 \tag{10}$$

$$\Delta_{ji}(n)=\Delta_{ji}(n-1)*d, \quad \text{if } \frac{\partial\xi}{\partial w_{ji}}(n)*\frac{\partial\xi}{\partial w_{ji}}(n-1)<0$$

$$\Delta_{ji}(n)=\Delta_{\max}, \quad \Delta_{ji}(n)\geq\Delta_{\max}$$

$$\Delta_{ji}(n)=\Delta_{\min}, \quad \Delta_{ji}(n)\leq\Delta_{\min}$$

3. Update the connection:

$$\Delta w_{ji}(n)=-\Delta_{ji}(n) \quad \text{if } \frac{\partial\xi}{\partial w_{ji}}(n)>0 \tag{11}$$

$$\Delta w_{ji}(n)=+\Delta_{ji}(n) \quad \text{if } \frac{\partial\xi}{\partial w_{ji}}(n)<0$$

$$\Delta w_{ji}(n)=0, \quad \text{else}$$

Recommended values for the parameters are: Δ_{max} = 50.0, Δ_{min} = 0.000001 and u = 1.2.

Dynamic Adaptation Methods

These techniques can be classified into the local adaptive techniques category since an optimal learning rate or momentum factor is assigned to each individual weight at different iterations. These techniques have been mathematically derived and proven to be effective and superior in terms of convergence when tested and compared with the batch BP (Evans & Zainuddin, 1997; Zainuddin & Evans, 1997; Zainuddin & Sathasivam, 2001).

Dynamic Momentum Factor (DMF)

DMF is an adjustment applied to the momentum factor at iteration n. Let $\Delta\alpha_{ji}(n, 0)$ denote the positive adjustment applied at iteration n to the momentum constant at iteration 0, $\alpha_{ji}(0, 0)$ We define $\Delta\alpha_{ji}(n, 0)$ as

$$\Delta\alpha_{ji}(n, 0) = \gamma_a^b + \alpha_{ji}(0, 0) \tag{12}$$

for all $n \in [a, b]$ where $0 \leq \gamma_a^b \leq 1 - \alpha_{ji}(0, 0)$ and $\gamma_a^b > \gamma_c^d$ for $a > c$ and $b > d$.

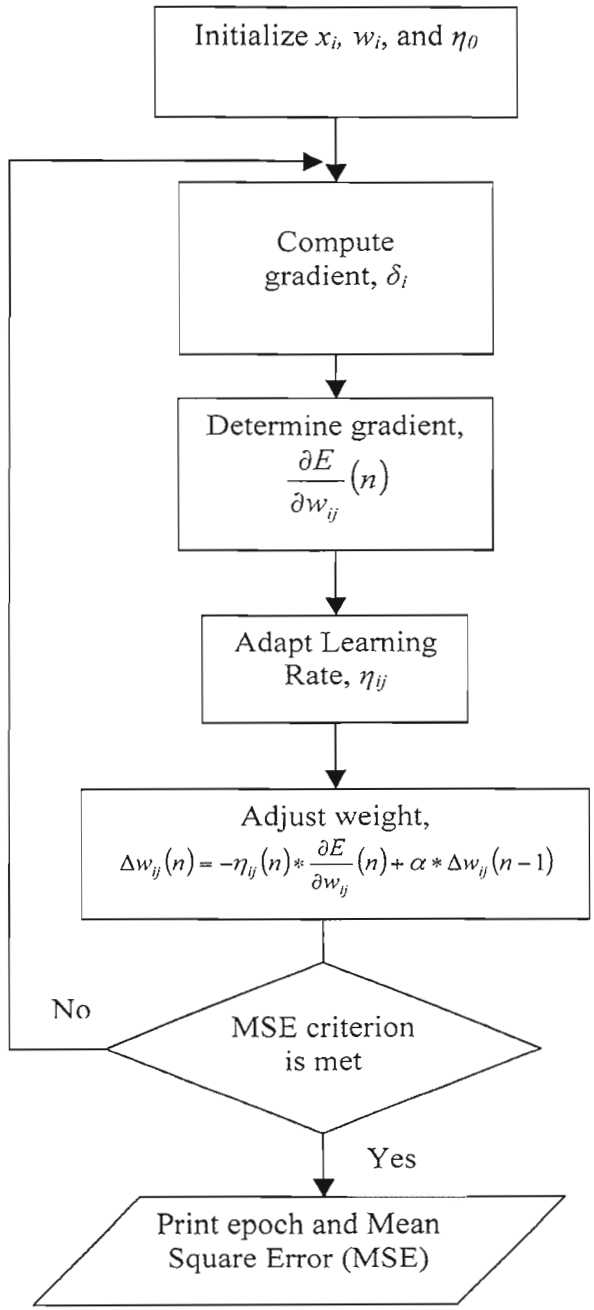

Figure 1: Flow Chart for the Local Adaptive Techniques

Dynamic Learning Rate 1 (DLR 1)

DLR method 1 is an adjustment applied at iteration n to the learning rate parameter, based on the gradient of each weight. Subsequently, it varies with every iteration.

Let $\Delta\eta_{ji}(n)$ denote the adjustment applied at iteration n to the learning rate parameter at iteration 0, $\eta_{ji}(0)$, We define as $\Delta\eta_{ji}(n)$

$$\Delta\eta_{ji}(n) = \lambda_{\delta b}^{\delta a} + \eta_{ji}(0) \tag{13}$$

for all $\delta = \left| \frac{\partial\xi(n)}{\partial w_{ji}(n)} \right| \in (\delta a, \delta b)$ and $\lambda_{\delta b}^{\delta a} < \lambda_{\delta d}^{\delta c}$ for $\delta a > \delta c$ and $\delta b > \delta d$

Dynamic Learning Rate 2 (DLR 2)

DLR method 2 is a positive adjustment applied at iteration n to the learning rate parameter. Let $\Delta n_{ji}(n, 0)$ denote the positive adjustment applied at iteration n to the learning rate parameter at iteration 0, $n_{ji}(0, 0)$. We define $\Delta n_{ji}(n, 0)$ as

$$\Delta\eta_{ji}(n, 0) = \chi_a^b + \eta_{ji}(0, 0) \tag{14}$$

for all $n \in [a, b]$ where $\chi_a^b \geq 0$ and $\chi_a^b > \chi_c^d$ for $a > c$ and $b > d$.

The initial value of η, $\eta_{ji}(0, 0)$ can be any small value in the interval [0,1].

Simulation Problem

Computer simulations for the local adaptive techniques, global adaptive techniques and dynamic adaptation methods are presented. We will compare the performance of all the techniques with the Backpropagation (BP). The methods include: (1) Backpropagation (BP) (2) Sign Changes (SC) (3) Delta-Bar-Delta Rule (DB) (4) SuperSAB (SAB) (5) Quickprop (QP) (6) Rprop (RP) (7) Dynamic Momentum Factor (DMF) (8) Dynamic Learning Rate 1 (DLR1) (9) Dynamic Learning Rate 2 (DLR2) (10) Powell-Beale (CGB) (11) Fletcher-Reeves (CGF), and (12) Polak-Ribiere (CGP) . Here we compare the performance of these techniques in the diabetes diagnosis and diagnosis of breast cancer problems.

Table 1 and Table 2 show the percentage improvement of the algorithms compared to BP algorithm in terms of gradient evaluation

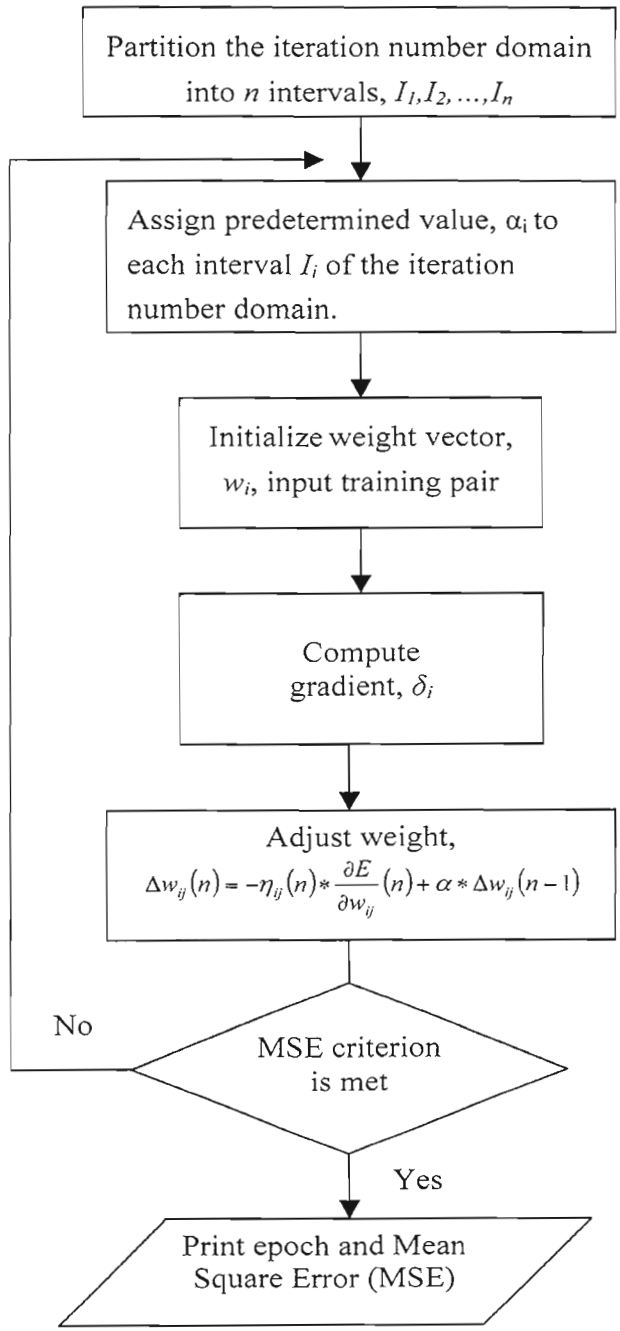

Figure 2: Flow Chart for the Dynamic Adaptation Method

and error function evaluation. The goal of this training for evaluating gradient of an error function. Error function is the sums of errors associated with each training data point. Each gradient evaluation and error function evaluation consists of μ, SD and Min/Max which respectively denote mean number of gradient or error function evaluation, standard deviation and minimum/maximum number of gradient or error function evaluation. The mean number of gradient or error function evaluations required to obtain the convergence. The lower number of mean show that the short time were taken in learning the networks. % denotes the percentage of success for each techniques (the value of mean error function evaluation) when compared with BP, means how long the techniques takes time to converge to actual output or the training speed (the number of iteration) of the techniques in learning the network.

Diabetes Diagnosis

This benchmark problem is a pattern recognition problem. The target of the network is to decide if an individual has diabetes, based on personal data (age, number of times pregnant) and results of medical examinations (blood pressure, body mass index, result of glucose tolerance test). The network used for this problem is an 8-6-2 network with tansig neurons in all layers. The architecture of the network was obtained from Arulampalam & Bouzerdoum (2003). The number of hidden nodes was determined by the "rules of thumb" (Blum, 1992).

The training set has 768 vector pairs and the testing set consists of 100 vector pairs. This data was transformed to normalize form by using mean standard deviation (*prestd*) function. Comparative tests were performed on the performance of Local Adaptive Techniques, BP and several other methods. The training was continued until 20,000 epochs and MSE reached at 0.035. The diabetes network is very large, so that the network needs more time to learn and adapt to the data sets. The following table summarizes the comparative results of training this network with various algorithms. The comparative results are illustrated in Table 1 and Figure 3.

Discussion of Results

From the Table 1, in terms of performance rate, all techniques were capable to improve the learning speeds; over 89% of the cases compared

Table 1: The Simulation Results for the Diabetes Diagnosis

Alg	Gradient Evaluation			Error Function Evaluation			Performance Rate(%)
	μ	SD	Min/Max	μ	SD	Min/Max	
BP	$9.5*10^5$	22360	$9*10^5/1*10^6$	$9.5*10^5$	22360	$9*10^5/1*10^6$	–
SC	96557	4497.4	86501/106614	96557	4497.4	86501/106614	89.84
DB	41657	940	39555/43759	41657	940	39555/43759	95.62
SAB	31665	633.25	30249/33081	31665	633.25	30249/33081	96.67
QP	29817	606.2	28462/31173	30516	293.6	29860/31173	96.86
RP	49857	63.73	49715/50000	49857	63.73	49715/50000	94.75
DMF	30284	586.5	28973/31596	30284	586.5	28973/31596	96.81
DLR1	16800	261.8	16215/17386	16800	261.8	16215/17386	98.23
DLR2	31097	695.6	29542/32653	31097	695.6	29542/32653	96.73
CGB	2299	541.1	1089/3509	3038	654.8	1574/4502	99.72
CGF	12396.5	1473.3	9102/15691	14008	1735.2	10128/17889	97.22
CGP	2790	245.1	2242/3338	3332.5	262.3	2746/3919	99.68

to BP. The average performance of error or gradient function evaluation for BP algorithm is still inferior when compared to the adaptive techniques performance. Regarding the training phase, the Global Adaptive Method, CGB algorithm exhibits the best performance of average number of error or gradient function evaluations and percentage of success. It is followed by the CGP and CGF algorithms, which also give good results. According to Dai and Yuan (1998), the formula of a restart direction by the CGB algorithm seems more reasonable to generate a higher learning speed.

Among the Local Adaptive Techniques, the Quickprop algorithm gives a better performance than the BP method. A little surprising for the Rprop technique is it cannot converge very well compared to the Diagnosis of Breast Cancer problem. This is because of the choice of learning rate(η) for Rprop techniques are not appropriate, so that the solution gets stuck a local minima values. The standard deviation for the Rprop training is 63.73, which shows a reasonable variation when compared to the other methods.

The Sign Changes algorithm has the ability to handle the large network and appears fast enough for this application, but still slower training when compared with other adaptive methods. This is because the adaptation of the Sign Changes algorithm is done by observing the signs of the last two gradients and not the magnitude of the gradients. As long as no change in sign is detected, the corresponding learning rate is increased. A possibility of this algorithm to get entrapment in neighborhoods of undesired local minima is high. There were some cases

of the solution getting stuck at local minima values for all techniques, which means that some of the patterns are not correctly classified.

Among the Dynamic Adaptation Methods, DLR1 exhibits the best performance of average number of error or gradient function evaluations and percentage of success. The other techniques measured by the mean number of error and gradient function evaluations also proved to accelerate the convergence rate considerably.

The fast and robust convergence of adaptive learning algorithm, and the failure of pure gradient descent, demonstrates the ability of the advanced techniques to exploit their adaptability to solve very complex learning tasks in situations where a suitable solution in weight space is difficult to find. It also shows that learning rate adaptation according to gradient is more applicable to get a good convergence. In the generalization aspect, all techniques show a good capability, which means the variation in the network outputs is well explained by the corresponding targets.

Diagnosis of Breast Cancer

The Diagnosis of Breast Cancer problem is chosen to compare and evaluate the Local Adaptive Techniques with Back-propagation method. The diagnosis predicts two fields, benign or malignant. The data sets are linearly separable using all 30 input features. Ten real valued features are computed for each cell nucleus: radius (mean of distances from center to points on the perimeter), texture (standard deviation of gray-scale values), perimeter, area, smoothness (local variation in radius lengths), compactness, concavity, concave points, symmetry and fractal dimension. The mean, standard error, and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features.

A 30-6-4-2 network was used where it consists of 30 input nodes, 2 hidden layer and 2 output nodes. The architecture of the network was obtained from Mey and Chai (1999). The number of hidden nodes was determined by the "rules of thumb" (Blum, 1992), which was described in Chapter 3. The first hidden layer consists of 6 nodes and the second hidden layer consists of 4 nodes.

The breast cancer training set consists of 200 vector pairs while the testing set consists of 50 vector pairs. For all simulation, the training has been continued until the MSE $\leq 1*10^{-3}$ within 100,000 epochs. The comparative results are illustrated in Table 2 and Figure 3.

Table 2: The simulation results for the Diagnosis of Breast Cancer

Alg	Gradient Evaluation			Error Function Evaluation			Performance Rate(%)
	μ	SD	Min/Max	μ	SD	Min/Max	
BP	$3.5*10^5$	22360	$3*10^5/4*10^5$	$3.5*10^5$	22360	$3*10^5/4*10^5$	–
SC	1144.5	208.6	678/1611	1144.5	208.6	678/1611	99.67
DB	1109	148.5	777/1441	1109	148.5	777/1441	99.68
SAB	1083	157.9	730/1436	1083	157.9	730/1436	99.69
Qprop	876	94.4	665/1087	876	94.4	665/1087	99.75
Rprop	54.5	1.1	52/57	54.5	1.1	52/57	99.98
DMF	811	69.3	656/966	811	69.3	656/966	99.76
DLR 1	807.5	83.4	621/994	807.5	83.4	621/994	99.77
DLR 2	937.5	28.8	873/1002	937.5	28.8	873/1002	99.73
CGB	115	20.1	70/160	191	32.6	118/264	99.96
CGF	1113.5	19.5	1070/1157	1112.5	49.9	1001/1224	99.68
CGP	84.5	10.1	62/107	123	13.0	94/152	99.97

Discussion of Results

It can be observed in Table 2 that the average performance of the BP method is inferior to the performance of the adaptive techniques. However, it is worth noticing where the Rprop algorithm exhibits the best performance in terms of the average number of gradient and error function evaluations required and it had the highest percentage of success. The mean error and gradient function evaluation is 54.5, needs fewer than all the other methods. The standard deviation is 1.1, which means the Rprop has a reasonable variation than others. The standard deviations are included only to give us a crude idea of the variation in the values; the distribution of learning times seldom looks like a normal distribution, often exhibiting multiple humps, for example. The percentage of success for the Rprop over BP is 99.98%, shows the best performance than the other techniques. The Rprop algorithm uses independent "update step size" for every connection which can avoid the phenomena of being trapped in local minima values. This property generates a higher learning speed.

This is followed by global adaptive methods CGP and CGB algorithms. For DMF, DLR 1 and DLR 2, the results are quite similar to the Quickprop algorithm. However, for the DLR1, the learning rate adaptation according to gradient descent shows a better convergence. For Sign Changes, Delta Bar Delta, SuperSAB and CGF, the average of function evaluations is 1100, which indicates better result than the

BP method. There were no cases of the solution getting stuck at local minima values for all methods except the BP algorithm implying that the choices of weight are appropriate.

As can be seen, all techniques were capable of achieving perfect results, close to 100% in each case when compared with BP. The BP algorithm is very slow because it is difficult to choose an appropriate learning rate. If we increase the step size, the algorithm will become unstable when it reaches steeper portions of the performance surface. In the local adaptive techniques, the weight update (step size) is adjusted at each iteration and will produce generally faster convergence.

In the generalization aspect, the Sign Changes technique had proven its capability (R = 1) and other techniques are closer to 1. The variation in the network outputs is explained very well by the corresponding targets.

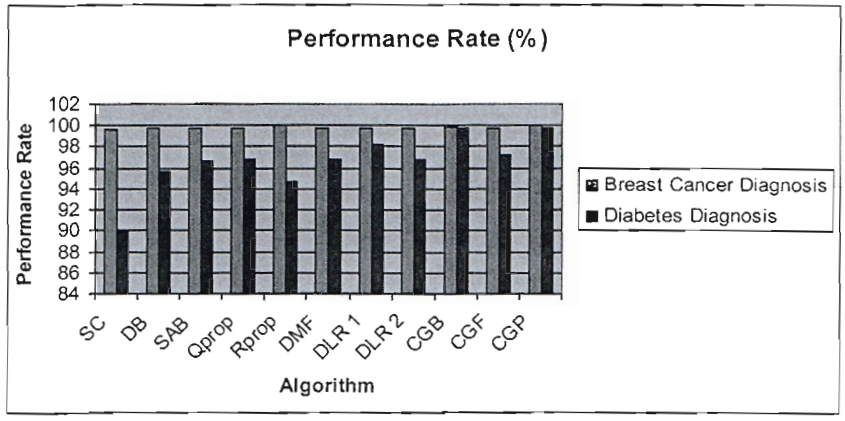

Figure 3: Trends in Algorithm Performance

Conclusion

This paper introduced a new algorithm of adaptive techniques, which has been proposed for the training of multilayer neural networks, and it enhanced the version of the Backpropagation BP algorithm. The study shows the adaptive techniques which is changing the learning rate dynamically during training is beneficial in speeding up the learning process. In conclusion, the method proposed in this study provides an accurate and convenient techniques for the diabetes diagnosis and the diagnosis of breast cancer. For a future work, the study on the selection of learning rates should be extended so that neural networks learning process can be expedited.

References

Arulampalam, G., & Bouzerdoum, A. (2003). A generalized feedforward neural network architecture for classification and regression. In *Neural Network 16 2003 Special Issue,* 2003, Edith Cowan University, Australia, 16(5-6), 561-568.

Baboo. S.S., & Shereef, I.K. (2010). An Efficient Weather Forecasting System using Artificial Neural Networks. International Journal of Environmental Science and Development, Vol. 1(4), 321-326, October 2010.

Blum, A. (1992). *Neural Networks in C++*. New York: John Wiley & sons.

Dai, Y.H., & Yuan, Y. (1998). Convergence properties of Beale-Powell Restart Algorithm. *Chinese Academy of Sciences,* China.

Evans, D.J. & Zainuddin, Z. (1997). Acceleration of the backpropagation through dynamic adaptation of the momentum. In *Neural, Parallel & Scientific Computations*, **5**(3), 297-308. (see also Internal Report No.1028, PARC, Loughborough University of Tech., U.K., 1996).

Fahlman, S.E. (1988). An empirical study of learning speed in backpropagation networks, Technical Report, CMU-CS-88-162.

Fahlman, S.E. & Lebierre, C. (1990). The Cascade-Correlation learning architecture. In *Advances in Neural Information Processing Systems 2* (Toureyzky, ed.), Morgan-Kaufman.

Iranmanesh, S. & Mahdavi, M.A. (2009). A Differential Adaptive Learning Rate Method for Back-Propagation Neural Networks. World Academy of Science, *Engineering and Technology* 50, 285-288.

Jacobs R. A. (1988). Increased rates of convergence through learning rate adaptation. In *Neural Networks*, 1(4), 295-308.

Magoulas, G.D., Vrahatis, M.N. & Androulakis, G.S. (1999). Improving the convergence of the backpropagation algorithm using learning rate adaptation methods. In *Neural Computation,* 11(7), 1769-1796.

Mey, K.C., & Chai, N.C. (1999). Pemecutan Pembelajaran Rangkaian Neural Dan Pembaikan Keupayaan Pengumuman Melalui Prosedur Reputan Pemberat, Universiti Sains Malaysia.

Omaima, N.A. aL-Allaf (2010). Improving the Performance of Backpropagation Neural Network Algorithm for Image Compression/ Decompression System. *Journal of Computer Science* 6(11), 1347-1354.

Otair., M. A. & Salameh.W.A. (2005). Speeding Up Back-Propagation Neural Netoworks. Proceedings of the 2005 Informing Science and IT Education Joint Conference, p.167-173.

Riedmiller, M. & Braun, H. (1993). A direct adaptive method for faster backpropagation learning. The RPROP algorithm. In *Proceedings of the IEEE International Conference on Neural Networks* (ICNN) (Ruspini, H. ed), p. 586-591. San Francisco.

Schiffmann, M. & Joost, R. Werner (1994). Optimization of the Backpropagation Algorithm for Training Multilayer Perceptrons, University of Koblenz.

Silva F.M. & Almeida L.B. (1990). Acceleration techniques for the backpropagation algorithm. *Neural Networks EURASIP Workshop*, Sesim.

Tollenaere, T. (1990). SuperSab:fast adaptive backpropagation with good scaling properties. *Neural Networks*, 3(5), 561-573.

Zainuddin, Z. & Evans, D.J. (1997). Acceleration of the backpropagation through dynamic adaptation of the learning rate, *Int. Journal of Computer Mathematics*, 334, 1-17. (see also Internal report No. 1029, PARC, Loughborough University of Tech., U.K. 1996).

Zainuddin, Z. & Sathasivam S. (2001). Modeling nonlinear relationships in ecology and biology using neural networks, *Proceedings of the National Workshop in Ecological and environmental modeling (ECOMOD)*, Sept. 3-4, 2001.