Experimentation on Learning Automata Based Methods for Adaptation of BP Parameters

H. Beigy M. R. Meybodi

Computer Engineering Department
Amirkabir University of Technology
Tehran Iran
email: beigy@ce.aku.ac.ir

Abstract

The backpropagation learning algorithm has number of parameters such learning rate (η), momentum factor (α) and steepness parameter (λ). whose values are not known in advance, and must be determined by trail and error. The appropriate selection of these parameters have large effect on the convergence of the algorithm. Many techniques that adaptively adjust these parameters have been developed to increase speed of convergence. A class of algorithms which are developed recently uses learning automata (LA) for adjusting the parameters η, α, and λ based on the observation of random response of the neural networks. In earlier papers the effectiveness of LA based algorithms using problems such as encoding problem, symmetry problem, parity problem, XOR problem, etc., were examined. In this note we test the LA based methods on more realistic problems including classification of sonar signals, vowel recognition, printed farsi digit recognition, and printed farsi character recognition. It is demonstrated through simulation that LA based schemes comparing to other schemes such as SAB ,Super SAB, and ASBP method have higher performance. The result of simulations approves of the claim made in other articles that learning automata is a good tool for designing parameter adaptation methods for neural networks.

Keywords: Neural Network, Backpropagation, Learning Automata, Learning Rate, Steepness Parameter, Momentum Factor

1. Introduction

Despite the many successful applications of backpropagation (BP) learning algorithm, it has many drawbacks. For complex problems it may require a long time to train the networks, and it may not train at all. Long training time can be the result of the non-optimum values for the parameters of the training algorithm. It is not easy to choose appropriate values for these parameters for a particular problem. Thus several researches have suggested algorithms for automatically adjusting the parameters of training algorithm as training proceeds, such as algorithms proposed by Arabshahi et al. [2], Kandil et al. [3], Parlos et al. [4], Cater [5], Franzini [6], Vosl et al. [7], Tesnuro and Janssens [8], Deros and Orban [22], Darken and Moody [9], Solmon [9], Tolleraere [23], Fallside and Chan [9], Jacobs [20], Sperduti and Starita [11] and Riedmiller and Heinrich [10] to mention a few. Several learning automata (LA) based procedures have been recently developed [12-17][32-35]. In these methods variable structure learning automata (VSLA) or fixed structure learning automata (FSLA) have been used to find the appropriate values of parameters for the BP training algorithm. In these schemes either a separate learning automata is associated to each layer or each neuron of the network or a single automata is associated to the whole network to adapt the appropriate parameters. It is shown that the learning rate adapted in such a way increases the rate of convergence of the network by a large amount.

In this report we study the performance of LA based schemes on more realistic problems: classification of sonar signals, vowel recognition, printed farsi digit recognition, printed farsi character recognition. It is demonstrated through simulation that LA based schemes comparing to the other known schemes such as SAB [20], SuperSAB[20], adaptive steepness method (ASBP) [11], have higher performance and results in faster convergence.

The rest of the paper is organized as follows. Section 2 briefly presents the basic backpropagation algorithm and learning automata. Existing LA based adaptation schemes for BP parameters are described in section 3. Section 4 at first presents four new problems and then examine the LA based schemes on these problems. Section 5 discusses the time and space complexity of LA based schemes. The last section is conclusion.

2. Backpropagation Algorithm and Learning Automata

In this section, in all brevity, we discuss the fundamentals of backpropagation learning algorithm and learning automata.

Back-propagation Algorithm: Error back-propagation training algorithm (BP) which is an iterative gradient descent algorithm is a simple way to train multilayer feedforward neural networks [1][5]. The BP algorithm is based on the gradient descent rule :

$$\Delta w_{jk}(n) = -\alpha \frac{\partial E}{\partial w_{jk}} + \mu \Delta w_{jk}(n-1) \tag{1}$$

where w_{jk} is the weight on the connection outgoing from the unit j and entering the unit k, α, μ , and n are learning rate, momentum factor, and time index, respectively. In the BP framework α and μ are constant and E is defined as:

$$E(n) = \frac{1}{2} \sum_{p=1}^{\#\text{patterns}} \sum_{j=1}^{\#\text{outputs}} [T_{p,j} - O_{p,j}]^2 \tag{2}$$

Where $T_{p,j}$ and $O_{p,j}$ are desired and actual outputs for pattern p at output node j and the index p varies on the training set. In the BP algorithm framework, each computational unit computes the same activation function. The computation of the sensitivity for each neuron requires the derivative of activation function, therefore this function must be continuos. The activation function is normally a sigmoid function chosen between the two following functions:

$$f(x) = \frac{1}{1 + e^{-\lambda x}} \tag{3}$$

$$f(x) = \frac{1 - e^{-\lambda x}}{1 + e^{-\lambda x}} \tag{4}$$

The steepness parameter λ determines the active region (region in which the derivative of sigmoid function is not very small) of activation function. As the steepness parameter decreases from positive infinity to zero, the sigmoid function changes from a unit step function to constant value 0.5. For large values of the steepness parameter λ, the derivative is very large and the active region of sigmoid function is very small. In this region, large derivative forces the algorithm to oscillate. Small active region means that the weights are updated rarely. For small value of steepness parameter λ, the active region is very large, but the derivative is very small and speed of convergence is very low. The steepness parameter λ is often set to a constant value and not changed by the learning algorithm. We gain much flexibility, if we move the net inputs of the sigmoidal functions near to their active regions, where the associated gradient are not very close to zero. This enables the BP algorithm to avoid some points in the network parameters space where the BP algorithm would effectively stop, even though it is not close to a local minima point. This will cause the gradient of the error function to be small if the sigmoidal is shifted far outside the active region of the input to the function. Therefore, we want to center each sigmoid to be inside the active region of the sigmoidal function.

The momentum term in weight adaptation equation (1) causes large change in the weight if the changes are currently large, and will decrease as the changes become less. This means that the network is less likely to get stuck in local minima early on, since the momentum term will push the changes over local downward trend. Momentum is of great assistance in speeding up convergence along shallow gradients, allowing the path, the network takes toward the solution to pickup speed in the downhill direction. The error surface may consist of long gradually sloping ravines which finish at a minima point. Convergence along these ravines is slow, and usually the algorithm oscillates across the ravine valley as it moves towards a solution. This is difficult to speed up without increasing the chance of overshooting the minima, but the addition of the momentum term is fairly successful. This difficulty could be removed if we select the momentum factor to be small at the near of minima and to be large far from minima. The proper choice of μ and λ have a tremendous effect in the performance of BP learning algorithm. Improper choices of these parameters may result slow convergence, paralysis and continuos instability.

***Learning Automata*:** Learning automata can be classified into two main families, fixed and variable structure learning automata. Examples of the FSLA type which we use in this paper are the Tsetline, Krinsky, and Krylov automata.

Fixed structure Learning Automata: A fixed structure automata is quintuple $\langle\alpha, \phi, \beta , F, G\rangle$ where:
1) $\alpha = (\alpha_1, . . . , \alpha_R)$ is the set of actions that it must choose from.
2) $\Phi = (\Phi_1, . . . , \Phi_s)$ is the set of states.
3) $\beta = \{0, 1\}$ is the set of inputs where “1” represents a penalty and “0” a reward.
4) F: $\Phi \times\beta\rightarrow\Phi$ is a map called the transition map. It defines the transition of the state of the automata on receiving an input, F may be stochastic.
5) G: $\Phi\rightarrow \alpha$ is the output map and determines the action taken by the automata if it is in state ϕ_j.

The selected action serves as the input to the environment which in turn emits a stochastic response $\beta(n)$ at the time “n”. $\beta(n)$ is an element of $\beta=\{0, 1\}$ and is the feedback response of the environment to the automata. The environment penalize (i.e. $\beta(n) = 1$) the automata with the penalty probability c_i, which is the action dependent. On the basis of the response $\beta(n)$, the state of the automata is $\phi(n)$ is updated and a new action chosen at (n+1). Note that the $\{c_i\}$ are unknown initially and it is desired that as a result of the interaction between the automata and the environment arrives at the action which presents it with the minimum penalty response in an expected sense.

Variable structure learning automata: Variable structure learning automata is represented by sextuple $\langle\beta , \phi , \alpha , P, G, T\rangle$, where β a set of inputs actions, ϕ is a set of internal states, α a set of outputs, P denotes the state probability vector governing the choice of the state at each stage k, G is the output mapping, and T is learning algorithm. The learning algorithm is a recurrence relation and is used to modify the state probability vector.

It is evident that the crucial factor affecting the performance of the variable structure learning automata, is learning algorithm for updating the action probabilities. Various learning algorithms have been reported in the literature [18]. Let α_i be the action chosen at time k as a sample realization from distribution p(k). The linear reward-inaction algorithm (L_{R-I}) is one of the earliest schemes. In an L_{R-I} scheme the recurrence equation for updating p is defined as

$$p_j(k) = \begin{cases} p_j(k) + \theta\left(1 - p_j(k)\right) & \text{if } i = j \\ p_j(k) + \theta \; p_j(k) & \text{if } i \neq j \end{cases} \qquad (5)$$

if β is zero and P is unchanged if β is one. The parameter θ is called step length, it determines the amount of increases (decreases) of the action probabilities. For more information on learning automata refer to [18][26][27][28][33].

Often the mean-square error surfaces for backpropagation algorithm are multi-modal. The learning automata is known to have well established mathematical foundation and global optimization capability. This latter capability of learning automata can be used fruitfully to search a multi-modal mean-square error surface. Learning automata can be used to find the appropriate value for different parameters of BP learning algorithm including learning rate, steepness parameter, and momentum factor. In the next section LA based BP parameter adaptation schemes reported in the literature are briefly described.

3. LA Based Schemes For Adaptation of BP Parameters

In this section, we briefly describe LA based schemes [12-17] for adaptation of BP parameters. In all of the existing schemes, one or more automaton have been associated to the network. The learning automata based on the observation of the random response of the neural network, adapt one or more of BP parameters. The interconnection of learning automata and neural network is shown in figure 1. Note that the neural network is the environment for the learning automata. The learning automata according to the amount of the error received from neural network adjusts the parameters of the BP algorithm. The actions of the automata correspond to the values of the parameters being calculated and input to the automata is some function of the error in the output of neural network.

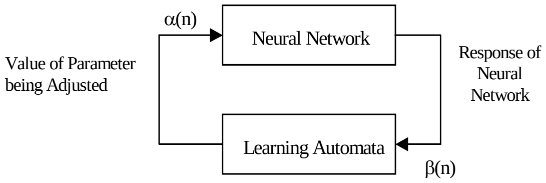

Figure 1: The interconnection of learning automata and neural network

A function of error between the desired and actual outputs of network is considered as the response of environment. A window on the past values of the errors are swiped and the average value of the error in this window computed. If the difference of the average value in the two last steps is less than the predefined threshold value, the response of the environment is favorable and if this difference of average value is greater than the threshold value.

Existing LA based procedures for adaptation of BP parameters can be classified into four groups which we call them group ***A, B, C,*** and ***D***. In group ***A*** schemes, one automaton is used for the whole network [12,14] whereas. in group ***B*** schemes, separate automata one for each layer (hidden and output) are used [13, 15, 16, 17]. Each group A and ***B*** depending on the type of automata used (fixed or variable structure) can be classified into two subgroups. The parameter adapted by group ***A*** schemes will be used by all the links or neurons of the networks and therefore these schemes fall in the category of global parameter adaptation method, whereas group ***B*** schemes by adapting the parameter for each layer independently may be referred to as quasi-global parameter adaptation methods.

In a class ***C*** scheme one automata is associated to each link of the network to adjust the parameter for that link and in a class ***D*** scheme one automata is associated to each neuron of the network to adjust the parameter for that neuron. Group ***C*** and ***D*** schemes may be referred to as the local parameter adaptation methods. In [34] class ***C*** schemes are used for adaptation of learning rate and class ***D*** schemes are used for adaptation of steepness parameter. In class ***C*** and ***D*** schemes, the automata receives favorable response from the environment if the algebraic sign of derivative in two consecutive iterations is the same and receives unfavorable response if the algebraic sign of the derivative in two consecutive iterations alternates. For the sake of convenience in presentation, we use the following naming conventions to refer to different LA based schemes in classes ***A***, ***B***, ***C***, and ***D***. Without loss of generality, we assume that in class ***A*** and class ***B***, the neural network has one hidden layer.

Automata-AX(γ): A scheme in class ***A*** for adjusting parameter γ which uses X structure learning automata.

Automata.**Automata$_1$-Automata$_2$-BX(γ):** A scheme in class ***B*** which uses X structure learning automata for hidden layer and X structure learning automata **Automata$_2$** for output layer.

Automata-CX(γ): A scheme in class ***C*** for adjusting parameter γ which uses X structure learning automata **Automata** .

Automata-DX(γ): A scheme in class D for adjusting parameter γ which uses X structure learning automata **Automata** .

The rate of convergence can be improved if both learning rate and steepness parameter are adapted simultaneously. Simultaneous use of class ***C*** and class ***D*** schemes for adaptation of learning rate and steepness parameters is also reported in [34]. In [34] a class ***C*** scheme is used for adaptation of learning rate and a class ***D*** scheme is used for adaptation of steepness parameter. A learning automata based scheme that simultaneously adapts learning rate and steepness parameter is denoted by **Automata1-Automata2-CDF(μ,λ),** if FSLA is used and **Automata1-Automata2-CDV(μ, λ),** if VSLA is used.

A simple method of increasing the learning rate and stability of training algorithm is to modify the standard BP by including the momentum factor [1] as given in equation (1). A learning automata based scheme which simultaneously adapt the learning rate and momentum factor is denoted by **Automata1-Automata2-CF**(η, α).

The letters **F** and **V** in above names denote FSLA and VSLA, respectively. X denotes either fixed or variable. For all the LA based schemes reported in the literature, it is shown through simulation that the use of LA for adaptation of BP learning algorithm parameters increases the rate of convergence by a large amount.

Figures 2 through 5 borrowed from[34][35][17], compare the effectiveness of different LA based schemes in different classes for adaptation of BP parameters. Shown in these figures are typical error curves for different LA based methods. As it is shown LA based schemes results in dramatically faster convergence, and has significantly smaller tail than standard BP and some other non-LA based schemes. In figure 2 the effectiveness of different schemes from class A are compared [17]. Figure 3 compares different schemes in class B and one scheme from class A. For this simulation Tsetline automata is associated to the hidden layer and the effect of association of different learning automata to the output layer are shown for digit problem. Figure 4 compares the performance of different schemes from class A with J-CF (η) scheme from class C for digit problem[34]. Figure 5 compares the performance of ASBP method with J-DF(λ) scheme from class D and standard BP for parity problem[35]. To the authors knowledge, ASBP method is the only method for adaptation of steepness parameter. For more simulations and an extensive discussion about different learning automata based schemes refer to [12-17][34-36].

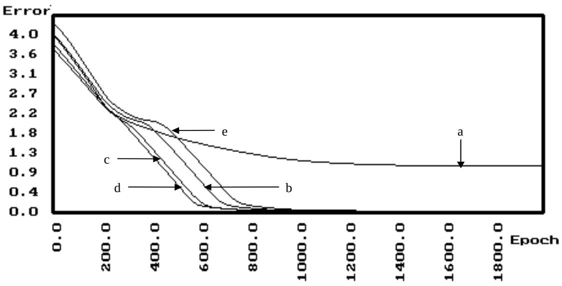

Figure 2

a: Standard BP b: Tsetline-AF (η)

c:Krinsky-AF(η) d:Krylov-AF (η) e:L_{R-P}-AV(η)

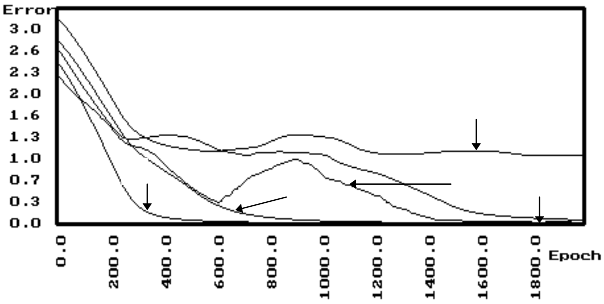

Figure 3

a) Standard BP b) Tsetline-AF

c) Tsetline-Krinsky-BF d) Tsetline-Tsetline-BF

e) Tsetline-TsetlineG-BF

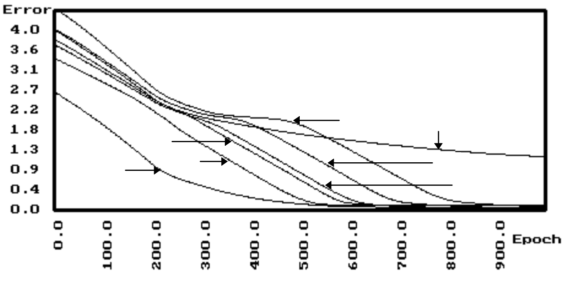

Figure 4

a: standard BP b: Tsetline-AF (η)

c: Krinsky-AF (η) d: Krylov-AF (η)

e: L_{R-P}-AV (η) f: VLR g: J-CF (η)

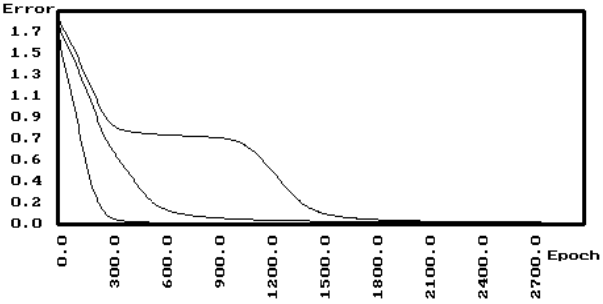

Figure 5

a: Standard BP b: ASBP c: J-DF(λ)

4. Experimentation

In this section we first introduce several new test problems and then present the results of simulations on these problems.

Classification of Sonar Signals: The task is to train a neural network to discriminate between sonar signal bounced off a metal cylinder and those bounced off a roughly cylindrical rock [29]. The training set consists of 104-member 60-dimensional patterns. The network that must learn to distinguish mine from rock, has 60 input units, 24 hidden units, and 2 outputs, one indicating a cylinder and the other a rock.

Vowel Recognition: In this problem, we have eleven steady state vowels of British English. These vowels are from fifteen different speakers. Each vowel is represented by a set of LPC derived log area ratios. The training set consists of 110 patterns. The network architecture used for recognition of vowels, have 10 input units, 22 hidden units, and 11 output units [32].

Printed Farsi Character Recognition: The Farsi alphabet consists of 32 basic characters shown in figure 6. These characters differ from other systems of characters in their structure and in the way they connect to form words. The same character may take different shapes according to its position in the word. For example, the character “GHEYN” has four different shapes according to its appearance at the head, the middle, the tail, or the isolated. This feature increases the number of Farsi characters more than 90 different shapes, including all forms of 32 basic characters, numeric characters, and punctuation characters[37]. A page of 32 isolated Farsi characters are scanned with resolution of 300dpi. Each character has 5 samples. The momentum constants M_1 through M_7 are used as inputs to the neural network. The network architecture used for this problem consists of 7 inputs which are connected to 31 hidden units, which are connected to 32 output units.

آ ALEF	ب BE	پ PE	ت TE	ث SE	ج JIM	چ TCHE	ح HE
خ KHE	د DAL	ذ ZAL	ر RE	ز ZE	ژ JE	س SIN	ش SHIN
ص SAD	ض ZAD	ط TA	ظ ZA	ع EYN	غ GHEYN	ف FE	ق GHAF
ک KAF	گ GAF	ل LAM	م MIM	ن NON	و WAW	ه HE	ی YE

Figure 6

Printed Farsi Digit Recognition: The ten printed Farsi digits are shown in figure 7. There is a page of 170 printed farsi digits, 17 samples for every digit[37]. 160 samples are used to train the network and the remaining samples are used for testing purpose. This page is digitized with resolution of 300dpi. The momentum constants M1 through M7 are extracted from digitized images and submitted as inputs to the neural network. The network that must learn to classify these digits has 7 input nodes, 30 hidden units, and 10 output units, one for each digit. Typical simulations for these four problems for different LA based parameter adaptation schemes are shown in figures 8 through 16. Figures 8 through 10 show the results of simulation for sonar signal recognition when different LA based schemes are used. As it is seen the best performance is exhibited by J-J-CDF(η,λ) scheme which outperforms some of known method such as ASBP method[11], SAB and Super SAB. The next set of figures (11 through 16) belongs to Vowel recognition problem, farsi digit problem, and farsi character recognition problem. For these problems the best result is obtained for J-J-CDF(η,λ) scheme which again outperforms the above mentioned known problems. Note that simulation results indicate that simultaneous adaptation of η and λ produces higher rate of convergence.

۰ digit 0	۱ digit 1	۲ digit 2	۳ digit 3	۴ digit 4	۵ digit 5	۶ digit 6	۷ digit 7	۸ digit 8	۹ digit 9

Figure 7

For all the simulations, we have taken the momentum factor (α) to be zero. For all simulations, parameters of different schemes are chosen in such away that best performances will be obtained. The plot for each simulation is averaged over 200 runs.

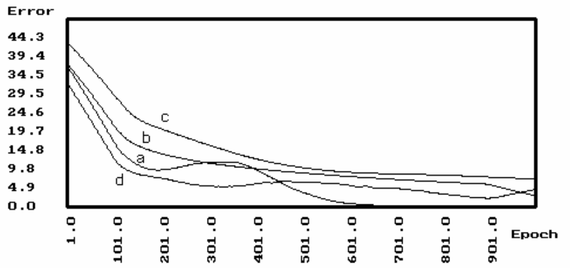

Figure 8: Sonar Signal Recognition
a: Standard BP b: SAB
c: Super SAB d: J-CF (η)

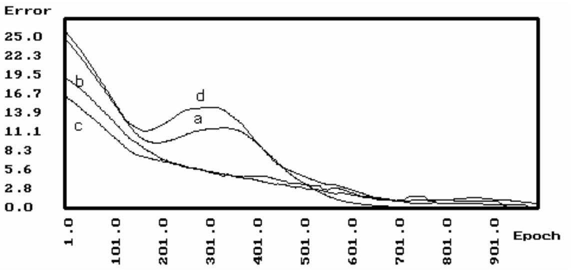

Figure 9: Sonar Signal Recognition
a: J-CF (η) b: Krinsky-CF (η)
c: Tsetline-CF (η) d: Krylov-CF (η)

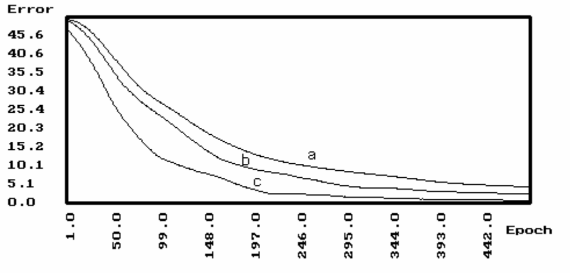

Figure 10: Sonar Signal Recognition
a: ASBP b: J-DF(λ) c: J-J-CDF(η,λ)

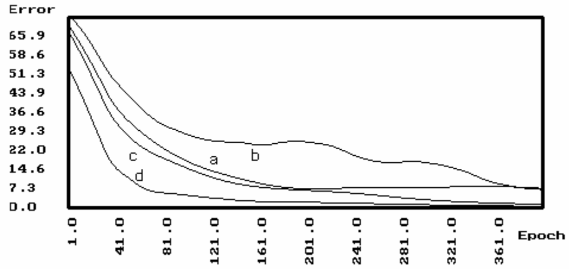

Figure 11: Vowel Recognition
a: Standard BP b: SAB
c: Super SAB d: J-CF (η)

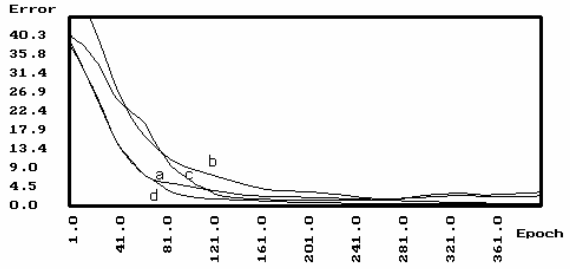

Figure 12: Vowel Recognition
a: J-CF (η) b: Krinsky-CF (η)
c: Tsetline -CF (η) d: Krylov-CF (η)

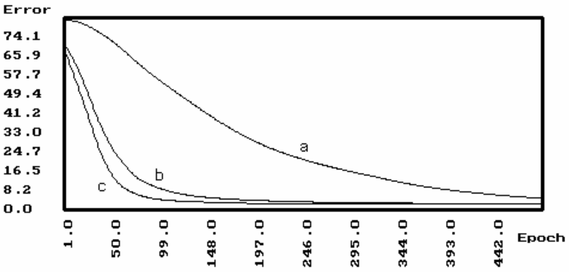

Figure 13: Vowel Recognition
a: ASBP b: J-DF(λ) c: J-J-DF(η,λ)

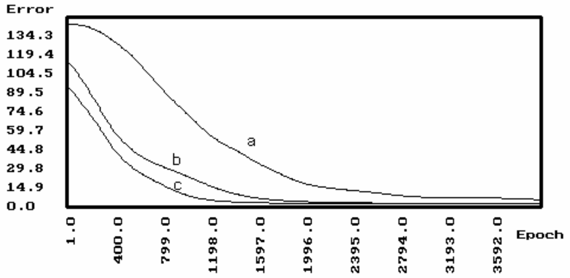

Figure 14: Farsi Digit Recognition
a: ASBP b: J-DF(λ) c: J-J-CDF(η,λ)

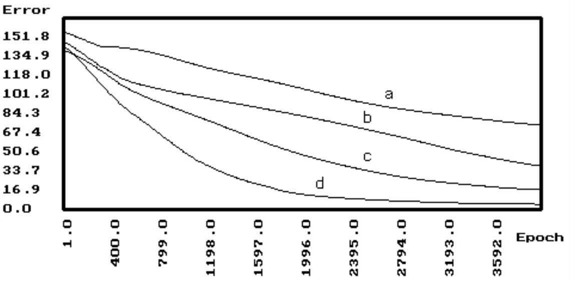

Figure 15: Farsi Digit Recognition
a: Standard BP b: SAB
c: Super SAB d: J-CF (η)

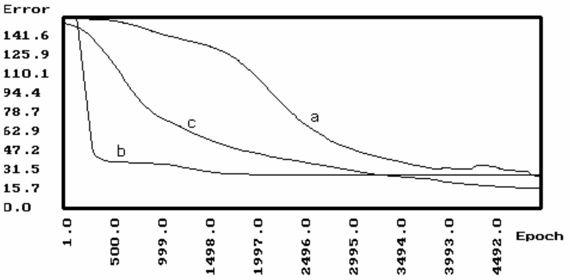

Figure 16: Farsi Character Recognition
a: ASBP b: J-DF(λ) c: J-J-CDF(η,λ)

5. Time and space complexity of LA-based adaptation schemes

In this section we discuss the time and storage overhead imposed on BP algorithm when LA based adaptation schemes are used. For implementation of FSLA three memory locations are needed in order to keep track of the state, number of actions, and memory depth of the automata. Changing state and also realizing the action associated with the state of the automata at each epoch requires few comparison, integer subtraction and addition. Therefore the storage and time overhead imposed by each FSLA is θ(1). This leads to θ(M) storage and θ(M) time overhead for ***CF*** type schemes and θ(N) storage and θ(N) time overhead for ***DF*** type schemes, where M and N are number of weights and number of neurons in the network, respectively. For these schemes in addition to storage needed to implement FSLA, storage for previous sign of gradient, adapted parameter, and reinforcement signal β for each neuron are needed. When VSLA with K actions used, the storage needed by ***CV*** and ***DV*** type schemes are θ(KM) and θ(KN), respectively. This is because of storage required by each automata to store its action probability vector P. Due to updating the action probability vector P by the automata at every epoch, the time overhead for ***CV*** and ***DV*** type schemes are θ(KM) and θ(KN), respectively. Class ***A*** and ***B*** schemes have overhead of θ(1) for both time and storage if FSLA is used and overhead of θ(K) for both time and space if VSLA is used. Table 3 summarizes the storage and time overhead imposed by different schemes.

Table 3 : The time and space overhead of proposed schemes

Algorithm	Storage Overhead	Time Overhead
AV	θ(1)	θ(1)
AF	θ(1)	θ(1)
BV	θ(1)	θ(1)
BF	θ(1)	θ(1)
CV	θ(KM)	θ(KM)
CF	θ(M)	θ(M)
DV	θ(KN)	θ(KN)
DF	θ(N)	θ(N)
SAB	θ(M)	θ(M)
SuperSAB	θ(M)	θ(M)

In order to justify the overheads imposed by proposed schemes the ratio of execution time of SAB, SuperSAB, and J-CF schemes to the execution time of standard BP algorithm for Farsi digit recognition problem are measured and given in table 4.

Table 4 : Ratio of execution time of SAB, SuperSAB, and J-CF to standard BP

Algorithm	SAB	SuperSAB	J-CF
Ratio of Execution Time	1.080	1.112	1.153

6. Conclusion

In this report we have studied the performance of LA based schemes on four different problems: classification of sonar signals, vowel recognition, printed Farsi digit recognition, and printed farsi character recognition. The result of studies in this note and previous works on LA based schemes indicates the fact that learning automata is a good tool for designing procedures for adaptation of neural networks parameters.

References

[1] Rumelhart, D. E., Hinton, G. E., and Williams, R. J.(1986). Learning Internal Representations by Error Propagation. In Parallel distributed processing, Cambridge, MA: MIT Press.

[2] Arabshahi, P. , Choi, J. J., Marks, R. J. , and Caudell, T. P. (1992). Fuzzy Control of Back propagation. Proc. of IEEE Int. Conf. on Fuzzy Systems, pp. 967-972.

[3] Kandil, N., Khorasani, K. , Patel, R. V., and Sood, V. K. (1996). Optimum Learning Rate for Back propagation Neural Networks. In P. K. Simpson (Eds.), Neural Networks Theory, Technology, and Applications, pp. 249-251.

[4] Parlos, A. G., Fernadez, B., Atya, A. F., Muthusami, J., and Tsai, W. K. (1994). An Accelerated Learning Algorithm for Multi-Layer Preceptron Networks. IEEE Trans. on Neural Networks, 5, pp. 493-497.

[5] Cater, J. P. (1987). Successfully Using Peak Learning Rates of 10 (and Greater) in BP Networks with the Heuristic Learning Algorithm. IEEE Proc. of First Int. Conf. on Neural Networks, Vol. II, pp. 645-651.

[6] Franzini, M. A. (1987). Speech Recognition with Backpropagation. IEEE Proc. of Ninth Annual Conf. on Engineering in Medicine and Biology, pp. 1702-1703.

[7] Vosl, T. P., Mangis, J. K., Rigler, A. K., Zink, W. T., and Alkon, D. L. (1987). Accelerating the Convergence of Backpropagation Method. Biological Cybernetics, pp. 257-263.

[8] Tesauro, G. and Janssens, B. (1988). Scaling Relationships in Backpropagation Learning. Complex Systems, pp. 39-44.

[9] Sarkar, D.(1995). Methods to Speedup Error Backpropagation Learning Algorithm. ACM Computing Surveys, **27**, pp. 519-542.

[10]Riedmiller, M. and Heinrich, B. (1992). A Direct Method for Faster Backpropagation Algorithm. Neural Networks, **5**, pp. 465-471.

[11]Sperduti, A. and Starita, A. (1995). Speed Up Learning and Network Optimization with Extended BP. Neural Networks, **6**, pp. 365-383.

[12]Menhaj, M. B. and Meybodi, M. R. (1995). A Novel Learning Scheme for Feedforward Neural Nets. Proc. of ICEE-95 Conf., University of Science and Technology, Tehran, Iran.

[13]Menhaj, M. B. and Meybodi, M. R. (1996). Flexible Sigmodal Type Functions for Neural Nets Using Game of Automata. Proc. of CSICC-96 Conference, Amirkabir University of Technology, Tehran, Iran , pp. 221-232.

[14]Menhaj, M. B. and Meybodi, M. R. (1996). Application of Learning Automata to Neural Networks. Proc. of CSICC-96 Conference, Amirkabir University of Technology, Tehran, Iran , pp. 209-220.

[15]Menhaj, M. B. and Meybodi, M. R. (1997). Using Learning Automata in Backpropagation Algorithm with Momentum. Technical Report, CE Dept., Amirkabir University of Technology, Tehran, Iran.

[16]Beigy, H. and Meybodi, M. R. (1999). Adaptation of Momentum Factor and Steepness parameter in Backpropagation Algorithm Using Fixed Structure Learning Automata. Proc. of CSICC-99 Conference, Sharif University of Technology, Tehran, Iran , pp. 117-124.

[17]Beigy, H., Meybodi, M. R., and Menhaj, M. B. (1998). Adaptation of Learning Rate in Backpropagation Algorithm Using Fixed Structure Learning Automata. Proc. of ICEE-95 Conference, K. N. Tosi University of Technology, Tehran, Iran, pp. 117-123..

[18]Narendra, K. S. and Thathachar, M. A. L. (1989). Learning Automata: An Introduction**.** Prentice-Hall, Englewood cliffs.

[19]Menhaj, M. B. and Hagen, M. H. (1995). Rapid Learning Using Modified BP Algorithms for Multi-Layer Feedforward Neural Nets. Proc. of ICEE-95 Conference, University of Science and Technology, Tehran, Iran.

[20]Jacobs, R. A. (1988). Increased Rates of Convergence Through Learning Rate Adaptation. Neural Networks, **1**, pp. 295-307.

[21]Scales, L. E. (1985). Introduction to Non-linear Optimization. New York, Springer-verlag.

[22] Devos, M. R. and Orban, G. A. (1988). Self Learning BP Proc. of NeuroNimes.

[23]Tollenaere, T. (1990). SuperSAB: Fast Adaptive Backpropagation with Good Scaling Properties. Neural Networks, 3.

[24] Hsin, H., Li, C. C., Sun, M., and Sclabani, R. J. (1995). An Adaptive Training algorithm for BP Neural Network. IEEE Trans. on System, Man and Cybern. 25, pp. 512-514.

[25]Schaffer, J. D., Whitley, D., and Eshelman, L. J. (1992). Combination of Genetic Algorithms and Neural Networks: A Survey of the State of the Art. Proc. of Int. Workshops on Combination of Genetic Algorithms and Neural Networks, COGANN-92, pp. 1-37.

[26]Meybodi, M. R. and S. Lakshmivarhan, S. (1982). Optimality of a General Class of Learning Algorithm", Information Science, 28, pp. 1-20.

[27]Meybodi, M. R. and S. Lakshmivarhan, S. (1984). On a Class of Learning Algorithms which have a Symmetric Behavior Under Success and Failure. Springer Verlag Lecture Notes in Statistics, pp. 145-155.

[28]Meybodi, M. R. (1987). Results on Strongly Absoulutely Expedient Learning Automata. Proc. of OU Inference Conf. 86, D. R. Mootes and R. Butrick (Eds.), Athens, Ohio: Ohio University Press, pp. 197-204.

[29]Gorman, R. P. and Sejnowski, T. J. (1988). Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets. Neural Networks, **1**, pp. 75-89.

[30]Gori, M. and Tesi, A. (1992). On the Problem of Local Minima in Backpropagation. IEEE Trans. on Pattern Analysis and Machine Intelligence, 14, pp. 76-86.

[31]Frasconi, P., Gori, M., and Tesi, A.(1992). Success and Failures of Backpropagation: A Theoretical Investigation. Technical Report, Dipartimento di Sistemi e Information, Universita di Firenze, Firenze, Italy.

[32]Deterding, D. H. (1989). Speaker Normalisation for Automatic Speech Recognition. Ph.D. Thesis, University of Cambridge, UK.

[33]Lakshmivarahan,S.(1981).Learning Algorithms: Theory and Applications. New York, Springer-verlag.

[34]Meybodi, M. R. and Beigy, H.(1998), New Class of Learning Automata Based Schemes for Adaptation of Bachpropagation Algorithm Parameters, Proc. of EUFIT-98, Achen, Germany, pp.339-344. (Extended Vesion Of this article has been appeared in proceedings of 7th Iranian Conference on Electrical Engineering, Iran Telecomunication Research Center, May 1999, pp. 1-16)

[35]Beigy, H. and Meybodi, M. R.(2000), Adaptation of Momentum Factor and Steepness Parameter in Backpropagation Algorithm Using Fixed Structure Learning Automata, International Journal of Science and technology, to appear.

[36]Arabshahi, P. et al (1996), Fuzzy Parameter Adaptation in Optimazation: Some Neural Net Training Examples, IEEE Computational Science and Engineering, pp. 57-65.

[37] Dastpak, V. (1992). Automatic Recognition of Farsi Printed Letters. Ms. Thesis, Computer Eng. Dept. Amirkabir University of Technology, Tehran, Iran (In Persian).