Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 38 (2012) 448 – 455

Procedia Engineering

www.elsevier.com/locate/procedia

International Conference on Modeling, Optimization and Computing - (ICMOC-2012)

A Comparative Study of Back Propagation and Simulated Annealing Algorithms for Neural Net Classifier Optimization

Subhendu Sekhar Behera[a], Subhagata Chattopadhyay[b¥]

[a]*B.Tech,Dept. of Computer Science & Engineering,National Institute of Science & Technology,Berhampur-761008,India*
[b¥]*Professor,Dept. of Computer Science & Engneering,National Institute of Science & Technologys,Berhampur-761008,India*

[a]ssbambitious@gmail.com;[b¥]subhagatachatterjee@yahoo.com (¥corresponding author)

Abstract

Optimizing the convergence of a Neural Net Classifier (NNC) is an important task to increase the speed and accuracy in the decision-making process. Learning algorithms are used to facilitate such optimization process. Simulated Annealing (SA) and Back propagation (BP) are two popular optimization algorithms. The objective of the study is to compare the optimization performances of SA and BP on a feed-forward NNC that relies more on BP. For the study, five standard datasets, such as WINE, IRIS, DIABETES, TEACHING ASSISTANT EVALUATION (TAE), and GLASS are considered. Experimental results reveal that during training and testing SA has outperformed BP (except for WINE data and that is during testing only). The generalised notion is that BP is one of the best optimisers for NNC. Hence, the experimental results are interesting. The authors assume that it is due to the fact that SA is a much randomized approach to find the global solution, while BP is a gradient-based search and thus has an inclination to be trapped in the local minima. Another important reason could be that BP relies on its best learning (occurred during its training) while classifying test cases, which might not be appropriate for a new dataset. SA, on the other hand, initiates a new search with the given dataset and because of its randomized search space, is able to converge into the global minima. The paper therefore argues that for optimising in a classical NNC, SA is more efficient than BP because of its random and flexible search space.

Keywords: ANN;Back Propagation;Simulated Annealing; Classification; Comparison

1. Introduction

Artificial Neural Network (ANN) is a mathematical representation of the networks of neurons in the brain and shares similar functionalities, such as receiving inputs, processing it and then generates output [1]. It resembles a connected graph of nodes, which are intertwined by the weighted links similar to the biological neurons. There are various types of ANNs, for example, Multi-layered Perceptron, Feed-forward

doi:10.1016/j.proeng.2012.06.055

network, Kohonen's network, Adaptive resonance network etc. The first two nets behave as classifiers i.e., these can learn from examples and the learning can be directly supervised, while the other nets learn from observations and then update the network weights; thereby follow unsupervised learning method, seen in cases of clustering. In this paper a feed-forward network (FFN) has been developed for classification purpose. FFN allows the signal/information to flow from input to output layer in feed-forward direction through hidden layer(s) [2]. All FFNs, as mentioned, can be trained in a supervised way so that it can learn the feature patterns available within the data. To obtain the desired accuracy in class predictions, proper training is mandatory. While training, the aim is to make the network learning the features at the best, which is reflected by minimizing the squared error (i.e., the squared difference between the calculated and the desired output).

There are several algorithms to optimize such training process. Back Propagation (BP) is one of the most popular NN training algorithms for supervised learning. The weights are corrected and updated with a generalized delta rule to minimize the prediction error through iterations. The weight correction methodology comprises of back-propagating the errors from output layer to hidden layer, thus finding the optimal set of weights [3].

Simulated Annealing (SA) is a probabilistic meta-algorithm for global optimization [4]. It is an analogy to the physical process where a solid is slowly being cooled until its structure is in a 'frozen' state, which happens at a minimum energy configuration [5]. Alike BP algorithm, in SA, the weights have to go through a number of configurations in the process until it reach the Global minimum [6]. There are also several other optimization techniques such as evolutionary algorithms e.g. Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Genetic Programming (GP) and so on; however, these are beyond scope of this paper.

Recent applications of BP with ANN include: future trade prediction [7], energy saving in wireless sensor networks [8], Analysis of LHM Elemental basic structure [9], weather forecasting [10,11], identification of seismic arrival types [12], Healthcare, especially mental health [13, 14] and in many other fields. On the other hand, significant contributions of SA include: Optimization of Chromatographic Separation [15], Water Distribution System [16], Structural Optimization [17], and so on. Detail description of these studies is beyond scope of this paper due to space constraint.

The key objective of this work is to test the performance of SA and compare with BP in the feed forward NNC architecture for pattern recognition, as SA has not been (i) tested widely on such architecture and (ii) compared with BP, which is the most popular optimization technique for NNC optimization.

2. Methodology

In this section the data, tools, and techniques used in the study are briefly described.

2.1. Data: Three standard datasets, such as WINE (178 x 13; Class: 1-3), IRIS (150 x 4; Class: 1-3), TEACHING ASSISTANT EVALUATION (TAE: 149 x 5; Class: 1-3), GLASS (214 x 10; Class: 1-6) and DIABETES (150 x 8; Class: 1-2), which are collected from UCI Machine Learning Repository [18].

2.2. Tool: Dev C++ has been used for developing and implementing the NNC, BP and SA algorithms. *MATLAB* 9 has been used for plotting the results as graphs.

2.3. Techniques:

2.3.1. Feed-Forward NNC: It is developed using the structure feature in C language. For each dataset used, we developed different network. The number of hidden layers of each network is limited to *one* for faster and simpler calculation. The number of neurons in the hidden layer is half of the neurons in input layer [19]. Table1 describes the structure of each network, developed.

Table 1. Description of the networks developed

Network description	Wine	Iris	Diabetes	TAE	Glass
no. of layers	3	3	3	3	3
no. of hidden layers	1	1	1	1	1
no. of neurons in input layer	13	4	8	5	9
no. of neurons in ouput layer	1	1	1	1	1
no. of neuons in hidden layer	6	2	4	3	5

The log sigmoid function in equation 1 is used as the transfer function associated with the neurons in hidden and output layers to obtain the normalized [0, 1] nodal outputs.

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

2.3.2. Normalization: As we use log sigmoid as the transfer function, we normalize the input values [0,1]. It will reduce calculation complexities. The class values for each dataset are also normalized in the range of 0 to 1 for homogenization of the data.

2.3.3. BP algorithm : The algorithm works in two phases. First, a training input pattern is presented to the input layer, which is forward-propagated to the output layer through hidden layer to generate the network output. Mean Square Error (*MSE*) is then calculated by comparing the calculated output (*CO*) and the target output (*TO*) for each input (see equation 2, where '*N*' denotes the number of total instances).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(TO - CO)^2 \tag{2}$$

In the next phase, with the MSE, the network signal/information back-propagates from the output layer to the input layer and the respective connector weights are updated using a 'generalized Δ rule' that is comprised of learning rate (λ) and momentum constant (α) [3]. Equation 3 and 4 express the Δ rule of weight updation. In these equations, the notation '*w*' denotes the weights between the connectors '*i*' and '*j*' and '*t*' is the state of iteration.

$$\Delta w_{i,j}^{t} = -\lambda.\frac{\partial MSE}{\partial w_{i,j}^{t}} + \alpha.w_{i,j}^{t} \tag{3}$$

$$w_{i,j}^{t+1} = \Delta w_{i,j}^{t} + w_{i,j}^{t} \tag{4}$$

For obtaining the best λ (which produce minimum MSE) rigorous parametric study has been performed in this study [20]. The λ at which the error is the least is chosen for the comparison with the SA algorithm. In this study, the momentum constant (α) is set equal to 0.9 for all cases to speed up the learning process. The epoch size is set as 100.

2.3.4. SA algorithm: Temperature (T) is a very important parameter of this technique which is the analog of the T in physical systems. Starting at a high T the algorithm reaches the lowest T with gradual decrement with attaining of a "thermal equilibrium" state at each T [21]. At each T, the weights are randomized. New set of weights are accepted as the newly optimized set if the MSE with this set is lower than the previous set or with a probability that the current set of weights will lead to the global minimum. As considered in BP, cost function and transfer functions are used. In this study it is assumed if the

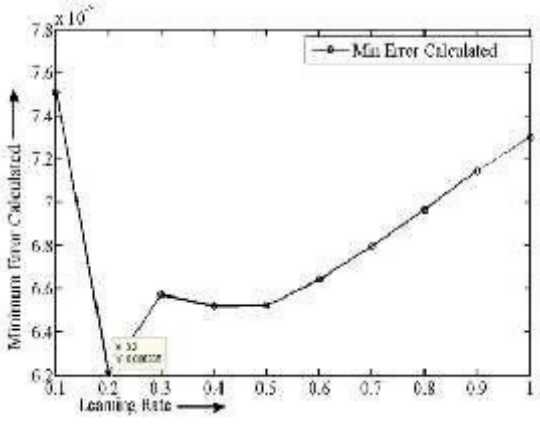
number of changes in the weight set is more than 10 or the number of iterations is more than 15000 then the equilibrium state at a particular T is said to be attained. The initial T is chosen as 10^{o} C and final T is 1^{o} C randomly. Also the T is lowered with by a factor of 0.95 arbitrarily, because it is difficult to find the exact values of initial, final and also the lowering of T. After a careful study, the search range is set from -2 to +2. The implementation algorithm is as follows (E(s)) is the objective function.

```
create initial solution s
set initial temperature T
while not terminate do
    repeat k times
          s′=perturb s
           ΔE=E(s′)-E(s)
          If (ΔE<=0) then
                s=s′
          else if (rnd(0,1) < exp(-ΔE/T)) then
                s=s′
          end if
    end repeat
    decrease T
end do
```

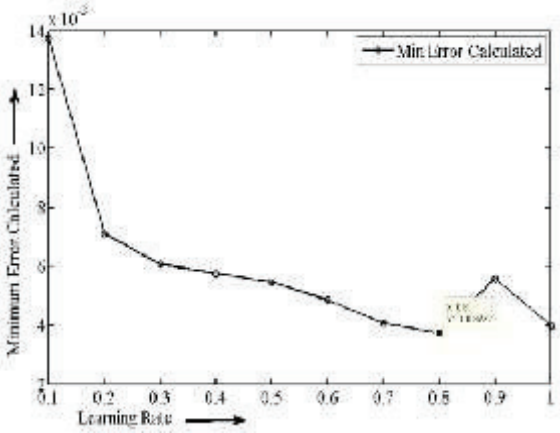
3. Results and discussions

3.1. Tuning the learning rate (λ):

Training and testing of the network with the BP algorithm is performed for the given epoch size for all datasets with all possible λ values (see fig.1). The λ with minimum of MSE for test cases is chosen for the comparison with SA algorithm. From fig.1 it may be noted that for (a) IRIS data, the MSE is 0.0062 at λ = 0.2, (b) for WINE data minimum MSE is 0.0037 at λ = 0.8; (c) for DIABETES data minimum MSE is 0.0980 at λ = 0.7; (d) for TAE, minimum MSE is 0.0558 at λ = 0.2; and finally (e) for GLASS data, the values are 0.0363 at λ = 0.1, respectively. These optimum values of λ (i.e., λ*) are used for the comparison between BP and SA. In case of BP epoch-wise MSE are computed; while for SA the steps of cooling has been considered to measure the MSE.



(a) **IRIS**



(b) **WINE**

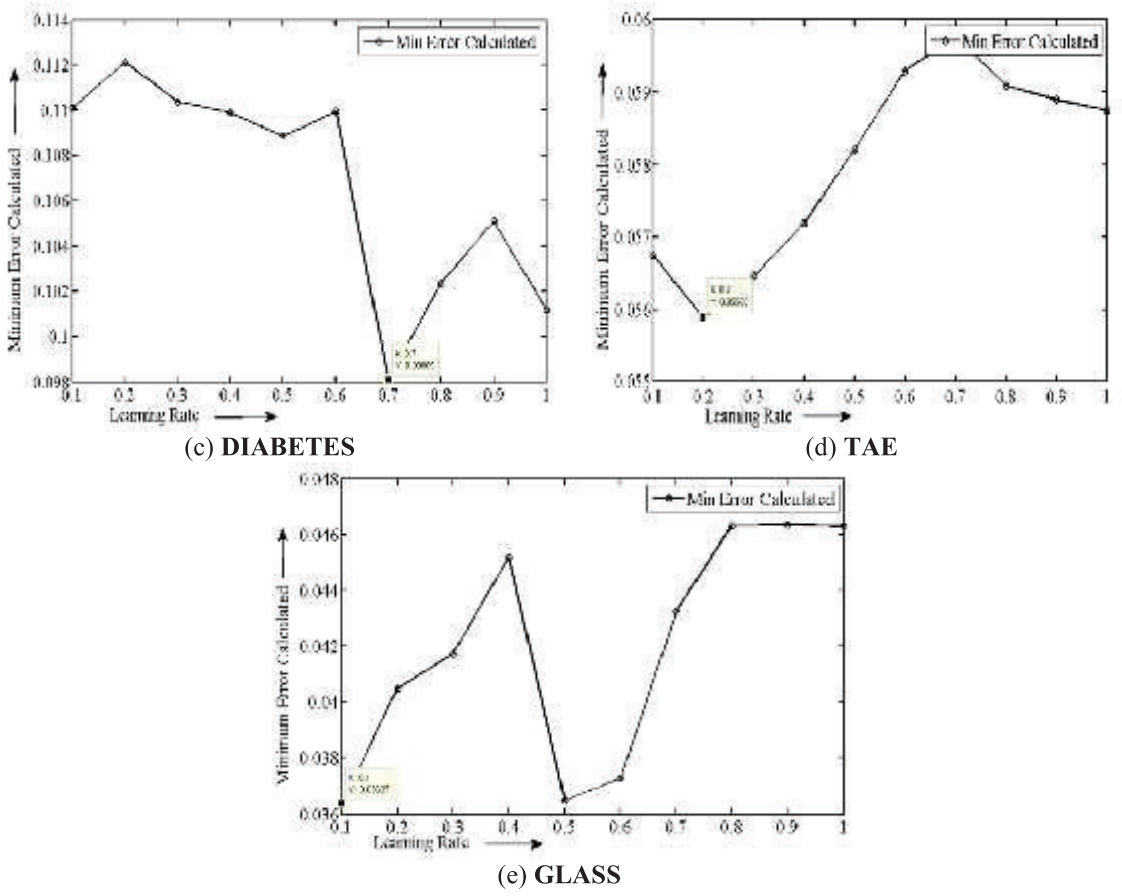


Fig. 1. MSE vs. λ parametric studies for the datasets (a) WINE; (b) IRIS; (c) DIABETES; (d) TAE; and (e) GLASS.

3.2 Comparison of BP and SA epoch-wise:

As mentioned, the range of parameter 'Temperature' (T) has been chosen from 10^{o} C to 1^{o} C. The 'equilibrium state' for it comprises of either 10 changes made in set of weights or 15000 iterations, whereas the epoch size in case of BP is set 100. Therefore, it is obvious that more time for training is given to SA. Fig. 2 shows the results of training and test cases for all five data. Table 2a and 2b summarise the results during training and testing, respectively.

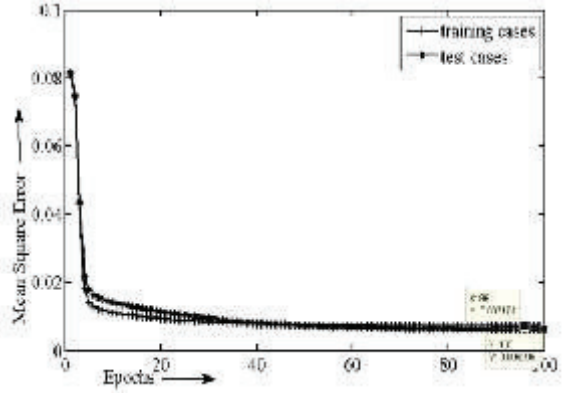
Table 2a. Mapping of perormances of BP and SA (During training)

Data set	**BP (λ*)**		**SA (-2,2)**	
	Epoch	MSE	T	MSE
1.IRIS	96	0.0071	1.577	*0.0048*
2.WINE	44	0.0017	1.498	*0.0001*
3.DIABETES	100	0.0748	1.046	*0.0672*
4.TAE	100	0.0634	1.046	*0.0541*
5.GLASS	100	0.0125	1.046	*0.0041*

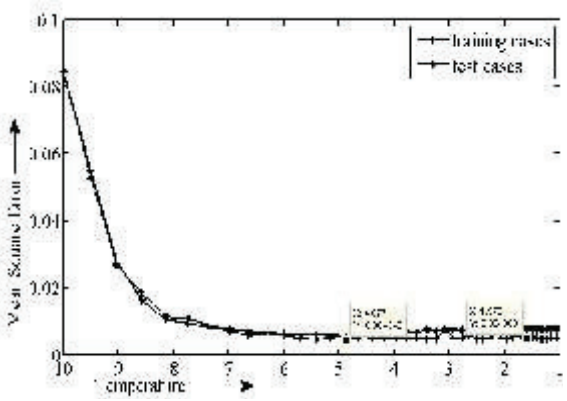
Table 2b. Mapping of perormances of BP and SA (During testing)

Data set	BP		SA (-2,2)	
	Epoch	MSE	T	MSE
1.IRIS	100	0.0062	4.876	*0.0043*
2.WINE	32	0.0037	5.403	0.0050
3.DIABETES	89	0.0980	9.025	*0.0940*
4.TAE	62	0.0558	9.025	*0.0549*
5.GLASS	5	0.0363	9.500	*0.0268*

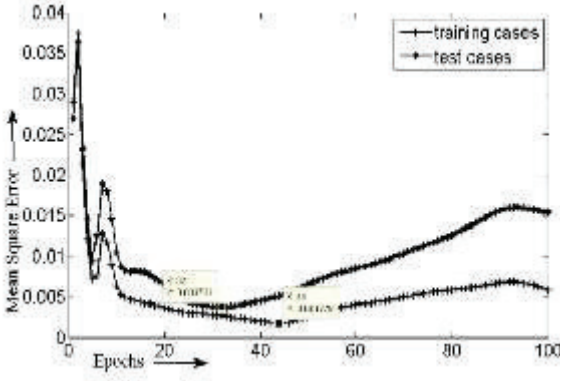
From the table 2 it is seen that while training, SA outperforms BP for all the datasets. Except for WINE data (although the difference is negligible), it has also performed better than BP during testing the network. This observation clarifies that the approach of randomization by SA to find solution in search space leads us to global optimum; whereas, BP might have trapped inside local minima. So the classification task is better performed by SA when compared to BP.



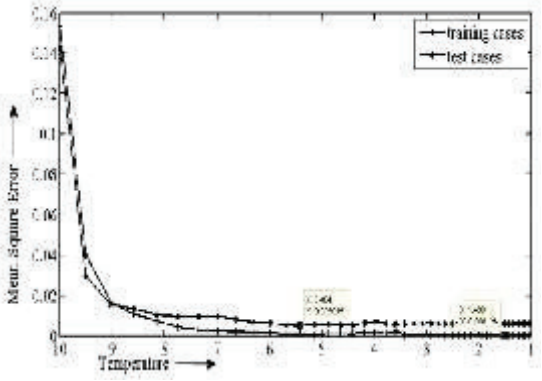
(a) BP



IRIS (b) SA



(a) BP



WINE (b) SA

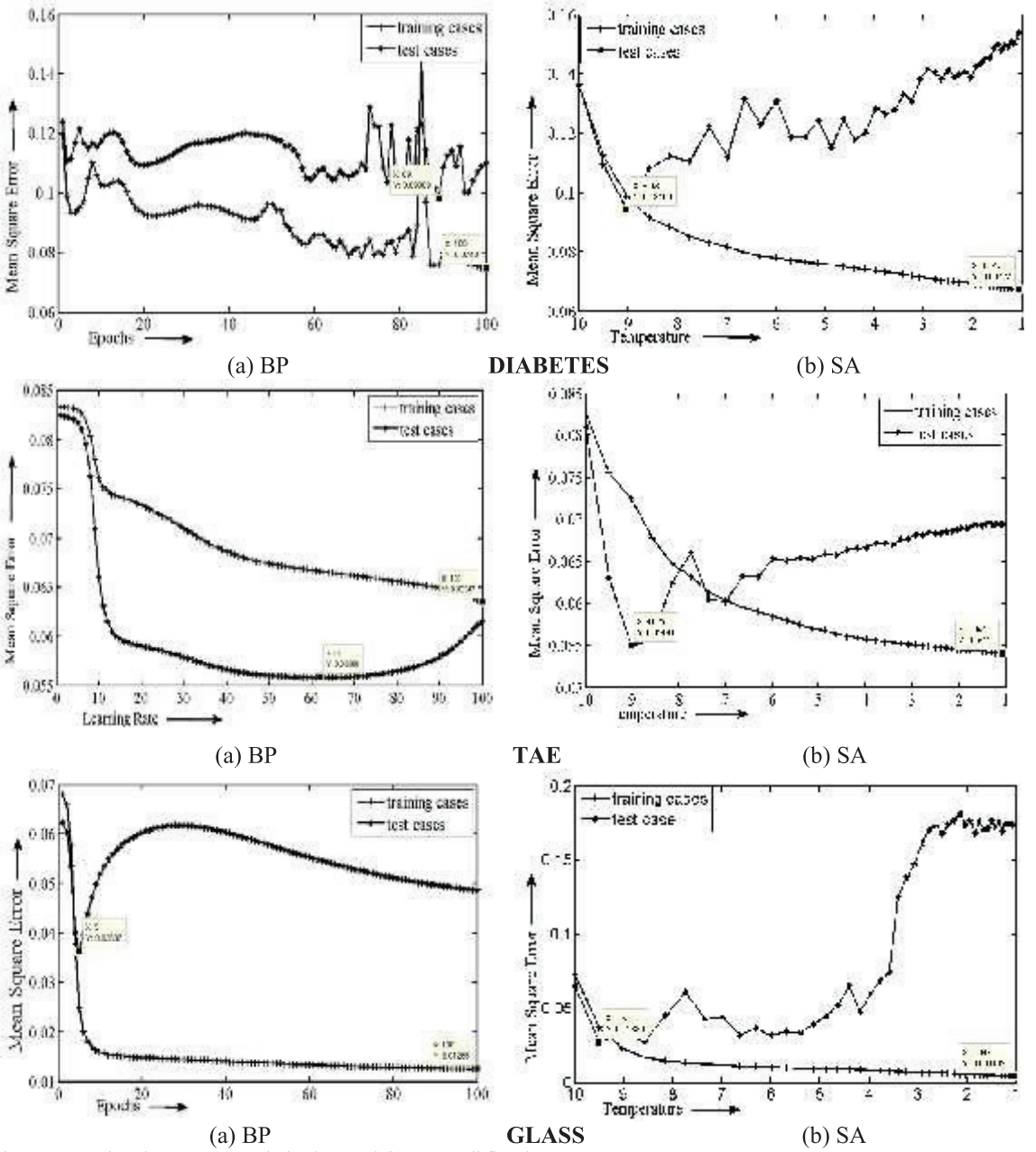


Fig. 2. Comparison between (a) Optimized BP and (b) SA on all five datasets.

4. Conclusions

The paper studies the performance of BP and SA on a FNN-based classifier on five datasets, such as IRIS, WINE, DIABETES, TAE and GLASS. Prior actual comparisons, the best λ for each dataset have been obtained through parametric studies during BP (see fig.1). While comparing, the performances of BP and SA, the study reveals that SA can predict both the exemplary and unknown patterns more

accurately compared to BP, except for WINE data, however, the difference of MSE is negligible. The reason could be that BP has a tendency to be stuck at the local minima due to its well-directed conjugate gradient approach and reliance on the optimum learning rate (obtained during the training). SA, being a random search can have more space and therefore has less chance to be trapped inside the local minima. This is an interesting finding in the field of NN-based classifier optimization.

Authors' future work consists of comparing these two traditional optimization techniques with evolutionary optimization approaches. They are currently working on this issue.

References

[1] Jha G.K., "Artificial Neural Network", IARI, 2006

[2] Abraham A. Chapter 129: "Artificial Neural Netorwk" in Handbook of Measuring System Design, by Sydenham P. H. and Thorn R. (eds.) John Wiley & Sons, 2005 [avaialble in http://www.softcomputing.net/ann_chapter.pdf] last accessed on 14/11/2011

[3] Jung I. and Wang G., "Pattern Classification of Back-Propagation Algorithm Using Exclusive Connecting Network", World Academy of Science, Engineering and Technology 36 2007.

[4] Kirkpatrick S., Gelatt C.D., Vechhi M.P., "Optimization by Simulated Annealing", Science, Vol. 220 (1983) pp. 671-680.

[5] Bailer-Jones D.M.and Bailer-Jones C.A.L., "Modelling Data: Analogies in neural networks, simulated annealing and genetic algorithms" in Model-Based Reasoning: Science, Technology, Values, L. Magnani and N. Nersessian (eds.), New York: Kluer Academic/Plenum Publishers, 2002.

[6] Theodoridis S. and Koutroumbas K., 2009, "Pattern Recognition", Academic Press, London, U.K, p.984.

[7] Skabar A.,Cloete I., "Neural Networks, Financial Trading and the Efficient Market Hypothesis", ACSC, 2002.

[8] Lin J., Guo W., Chen G., Gao H., Fang X.,"A PSO-BPNN-BASED model for energy saving in Wireless Sensor Networks", International Conference on Machine Learning and Cybernetics, Baoding, 12-15 July,2009.

[9] Yu J., Hong X., Liu X., Teng W., "Application of BPNN in analyzing LHM elemental basic structure", Applied Mechanics and Materials Vols. 16-19 (2009) pp 460-470.

[10] Wang H., Kuang X., Liu J.,"Application of back propagation neural network in paleoclimate", International Conference on Information Science and Technology,2011.

[11] "An ensemble of neural networks for weather forecasting", Neural Computing and Application, 2009, Vol 13, pp 112-122.

[12] Dai H., Macbeth C., "Application of back-propagation neural networks to identification of seismic arrival types ", Physics of the Earth and Planetary Interiors, vol. 101, issues 3-4,pp 177-188 .

[13] Chattopadhyay S., Kaur P., Rabhi F. A., Acharya U.R. "An automated system to diagnose the severity of adult depression", EAIT-2011 pp. 121-124

[14] Chattopadhyay S., Kaur P., Rabhi F. A., Acharya U.R. "Neural network approaches to grade adult depression", Journal of Medical Systems, 2011 DOI: : 10.1007/s10916-011-9759-1 (in press)

[15] Kaczmarski K., Antos D.,"Use of Simulated Annealing for optimization of chromatographic separations", ACTA CHROMATOGRAPHICA, no. 17, 2006.

[16] Goldman F.E., Mays L.W. Chapter 5: "Water distribution system operation: application of simulated annealing" In: Mays LW (ed) Water Resources Systems Management Tools. McGraw-Hill, New York, 2005.

[17] Kolahan, F., Abolbashari, M.H. , Mohitzadeh, S. Simulated Annealing Application for Structural Optimization, World Academy of Science, Engineering and Technology. 2007, pp. 35.

[18] http://www.ics.uci.edu [last accessed 14/12/2011]

[19] http://www.heatonresearch.com/node/707 [last accessed 14/12/2011]

[20] Lee C.K., Chung C.H., "The effect of learning rate to an unsupervised neural network for inspection process", Transactions on Information and Communication Technologies, WIT Press, vol. 19, 1997

[21] Bertisimas D. and Tsitsiklis J., "Simulated Annealing", Statistical Science, Vol. 8, No. 1, pp. 10–15, 1993.