CSLM: Levenberg Marquardt based Back Propagation Algorithm Optimized with Cuckoo Search

Nazri Mohd. Nawi, Abdullah Khan & M. Z. Rehman

Software and Multimedia Centre, Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia (UTHM).
P.O. Box 101, 86400 Parit Raja, BatuPahat, Johor DarulTakzim, Malaysia
Email: hi100010@siwa.uthm.edu.my

Abstract. Training an artificial neural network is an optimization task, since it is desired to find optimal weight sets for a neural network during training process. Traditional training algorithms such as back propagation have some drawbacks such as getting stuck in local minima and slow speed of convergence. This study combines the best features of two algorithms; i.e. Levenberg Marquardt back propagation (LMBP) and Cuckoo Search (CS) for improving the convergence speed of artificial neural networks (ANN) training. The proposed CSLM algorithm is trained on XOR and OR datasets. The experimental results show that the proposed CSLM algorithm has better performance than other similar hybrid variants used in this study.

Keywords: *artificial neural networks; back propagation; cuckoo search; levenberg marquardt; local minima.*

1 Introduction

An Artificial Neural Network (ANN) is a data processing model that is based on the biological nervous systems of a human brain [1]-[2]. The main constituent of this representation is the new foundation of the data processing system. It consists of a large number of tremendously interrelated processing elements known as neurons, all functioning together in order to solving many complex real world problems [3]. ANN have been effectively implemented in all engineering fields such as biological modeling, decision and control, health and medicine, engineering and manufacturing, marketing, ocean exploration and so on [4]-[9]. Because of the delightful appearance of artificial neural networks, a large number of applications have been proposed in recent decades. The back propagation (BP) algorithm that was introduced by Rumelhart [10] is the well-known method for training a multilayer feed-forward artificial neural networks [11]. However, the BP algorithm suffers from two major drawbacks: low convergence rate and instability. They are caused by a possibility of being trapped in a local minimum and prospect of overshooting the minimum of the error surface [12]-[14].

Received September 23rd, 2013, 1st Revision October 1st, 2013, 2nd Revision November 4th, 2013, Accepted for publication November 13th, 2013.

In recent years, many improved learning algorithms have been proposed to overcome the flaws of gradient descent based systems. These algorithms include a direct enhancement method using a poly tope algorithm [14], a global search procedure such as evolutionary programming [15], and genetic algorithm (GA) [16]. The standard gradient-descent BP is not path driven, but population driven. However, the improved learning algorithms have explorative search topographies. Therefore, these approaches are expected to evade local minima often by promoting exploration of the search space. The Stuttgart Neural Network Simulator (SNNS) [17], which was developed in the recent past use many different algorithms including Error Back Propagation [13], Quick prop algorithm [18], Resilient Error Back Propagation [19], Back percolation, Delta-bar-Delta, Cascade Correlation [20] etc. All these algorithms are derivatives of steepest gradient search, so ANN training is relatively slow. For fast and efficient training, second order learning algorithms have to be used. The most effective method is Levenberg Marquardt (LM) algorithm [21], which is a derivative of the Newton method. This is quite multifaceted algorithm since not only the gradient but also the Jacobian matrix should be calculated. The LM algorithm was developed only for layer-by-layer ANN topology, which is far from optimal [22]. LM algorithm is ranked as one of the most efficient training algorithms for small and medium sized patterns. LM algorithm is coined as one of the most successful algorithm in increasing the convergence speed of the ANN with MLP architectures [23]. It is a good combination of Newton's method and steepest descent [24]. It Inherits speed from Newton method but it also has the convergence capability of steepest descent method. It suits specially in training neural network in which the performance index is calculated in Mean Squared Error (MSE) [25] but still it is not able to avoid local minimum [26]-[27].

In order to overcome these issues this study proposed a new algorithm that combines Cuckoo Search (CS) and Levenberg Marquardt Back propagation (LMBP) algorithms to train neural network for XOR and OR datasets. The proposed CSLM algorithm reduces the error and improved the performance by escaping from local minima.

The remaining of the paper is organized as follows. Section two describes the Cuckoo Search algorithm. In Section three, the implementation of the proposed CSLM algorithm is elaborated. In Section Four, the performance of the proposed CSLM on some experimental data is discussed. The paper is finally concluded in Section Five.

2 The Cuckoo Search Algorithm

Xin-She Yang [28] proposed a metaheuristic Cuckoo Search (CS) algorithm based on the forceful parasitic behavior of some cuckoo species by laying their eggs in the nests of other bird species. Sometimes the host bird cannot differentiate between its own and cuckoo eggs. But, if an egg is discovered by the host bird as not its own, then it either throw these unknown eggs away or simply leave its nest. Some species in cuckoo are very specialized in the impersonating the color and pattern of the eggs of the host species. This reduces the chances of their eggs being abandoned and thus increases the chances of their survival. The CS algorithm follows the three basic rules:

1. Each cuckoo lays one egg at a time, and put its egg in randomly chosen nest;
2. The best nests with high quality of eggs will carry over to the next generations;
3. The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability pa [0, 1].

In this case, the host species can either throw the egg away or build a completely new nest somewhere else. The last assumption can be approximated by the fraction *Pa* of the *n* nests that are replaced by new nests (with new random solutions). For a maximization problem, the quality or fitness of a solution can simply be proportional to the value of the objective function. In this algorithm, each egg in a nest represents a solution, and a cuckoo egg represents a new solution, the aim is to use the new and potentially better solutions (cuckoos) to replace a not so good solution in the nests. Based on these three rules, the basic steps of the Cuckoo Search (CS) can be summarized as the following pseudo code:

```
Step 1: Generate initial population of N host nest $i = 1...N$
Step 2: While ( $f \min < MaxGeneration$ ) or (stop criterion)
Step 3: Do Get a Cuckoo randomly by Levy flights and evaluate its fitness $F_i$ .
Step 4: Choose randomly a nest $j$ among $N$ .
Step 5: If $F_i > F_j$ Then
Step 6: Switch $j$ by the new solution, End If
Step 7: A segment ( $p_a$ ) of worse nest are abandoned and new ones are built.
Step 8: Keep the optimal solutions (or nest with quality solutions).
Step 9: Rank the solutions and find the current best.
Step 10: end while
```

When creating new solutions x^{t+1} for a cuckoo i , a Levy flight is performed

$$x^{t+1} = x_i^{\,t} + \alpha \oplus levy(\lambda) \qquad (1)$$

where $\alpha > 0$; is the step size which should be related to the scales of the problem of interest. The product $\oplus$ means entry wise multiplications. The random walk via Levy flight is more effective in exploring the search space as its step length is much longer in the long run. The Levy flight essentially provides a random walk while the random step length is drawn from a Levy distribution.

$$Levy \sim u = t^{-\lambda},\ 1 < \lambda \leq 3 \qquad (2)$$

This has an infinite variance with an infinite mean. Here the steps essentially construct a random walk process with a power-law step-length distribution with a heavy tail. Some of the new solutions should be generated by Levy walk around the best solution obtained so far, this will speed up the local search. However, a substantial fraction of the new solutions should be generated by far field randomization whose locations should be far enough from the current best solution. This will make sure the system will not be trapped in a local minimum.

3 The Proposed CSLM Algorithm

The proposed method known as Cuckoo Search based Levenberg-Marquardt (CSLM) algorithm is given in Figure 1. Cuckoo Search (CS) is a metaheuristic algorithm that starts with a random initial population. It works with three basic rules i.e. selection of the best source by keeping the best nests or solutions, replacement of host eggs with respect to the quality of the new solutions or cuckoo eggs produced based randomization via Levy flights globally (exploration) and discovering of some cuckoo eggs by the host birds and replacing according to the quality of the local random walks (exploitation) [29]. In the figure, each cycle of the search consists of several steps initialization of the best nest or solution, the number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability, p_a [0, 1].

In this algorithm, each best nest or solution represents a best possible solution (i.e., the weight space and the corresponding biases for NN optimization in this study) to the considered problem and the size of a food source represents the quality of the solution. The initialization of weights was compared with output and the best weight cycle was selected by cuckoo. The cuckoo would continue searching until the last cycle to find the best weights for networks. The solution that was neglected by the cuckoo was replaced with a new best nest. The main idea of this combined algorithm is that CS is used at the beginning stage of searching for the optimum. Then, the training process is continued with the LM algorithm. The LM algorithm incorporates the Newton method and gradient descent method. The flow diagram of CSLM is shown in Figure 1. In the first stage CS algorithm finish its training then LM algorithm starts training with the weights generated by CS algorithm and the LM train the network until the stopped condition is satisfied.

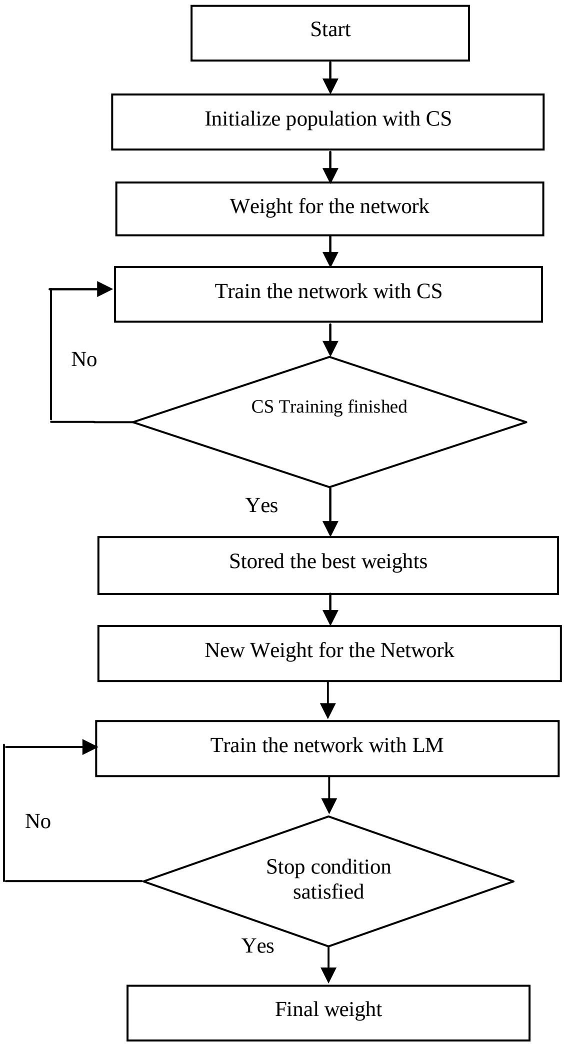

Figure 1 The Proposed CSLM Algorithm.

4 Results and Discussions

4.1 Preliminaries

In order to illustrate the performance of the proposed algorithm, CSLM is trained on 2-bit, 3-bit XOR and 4-bit OR datasets. The simulation experiments were performed on a 1.66 GHz AMD E-450 APU with Radeon and 2 GB RAM using MATLAB 2009b software. The proposed CSLM algorithm is compared with Artificial Bee Colony Levenberg Marquardt algorithm (ABC-LM), Artificial Bee Colony Back Propagation (ABC-BP) algorithm and simple back propagation neural network (BPNN) based on MSE and maximum epochs was set to 1000. The three layer feed forward neural network are used for each problem; i.e. input layer, one hidden layer, and output layers. The number of hidden nodes is formed of five neurons. In the network structure the bias nodes are also used and the log sigmoid activation function is placed as the activation function for the hidden and output layers nodes. For each algorithm, 20 trials are repeated.

4.2 Two Bit Exclusive-OR Problem

The first test problem is the exclusive OR (XOR) which is a Boolean function of two binary input to a single binary output. In the simulation we used 2-5-1, feed forward neural network structure for two bit XOR problem. For all algorithms, the parameters range is set to [5,-5], [5,-5], [5,-5], [1,-1] respectfully. Table 1 shows the CPU time, number of epochs, accuracy, SD and the MSE for the 2 bit XOR test problem with five hidden neurons. Figure 2 shows the MSE of the proposed CSLM algorithm and ABC-LM algorithm for the 2-5-1 network structure. From Table 1, it can be clearly seen that the proposed CSLM algorithm converged to the global minima within 126 epochs, while ABC-LM needs much time and number of epochs to converge. Figure 3 illustrates the average accuracy for the CSLM, ABC-BP, ABC-LM, and conventional BPNN algorithms.

Table 1 CPU time, Epochs and MSE for 2-5-1 NN architecture.

Algorithm	BPNN	ABC-BP	ABC-LM	CSLM
CPUTIME	42.643	172.33	123.95	14.41
EPOCHS	1000	1000	1000	126
MSE	0.220	2.39E-04	0.125	0
SD	0.0105	6.7E-05	1.5E-06	0

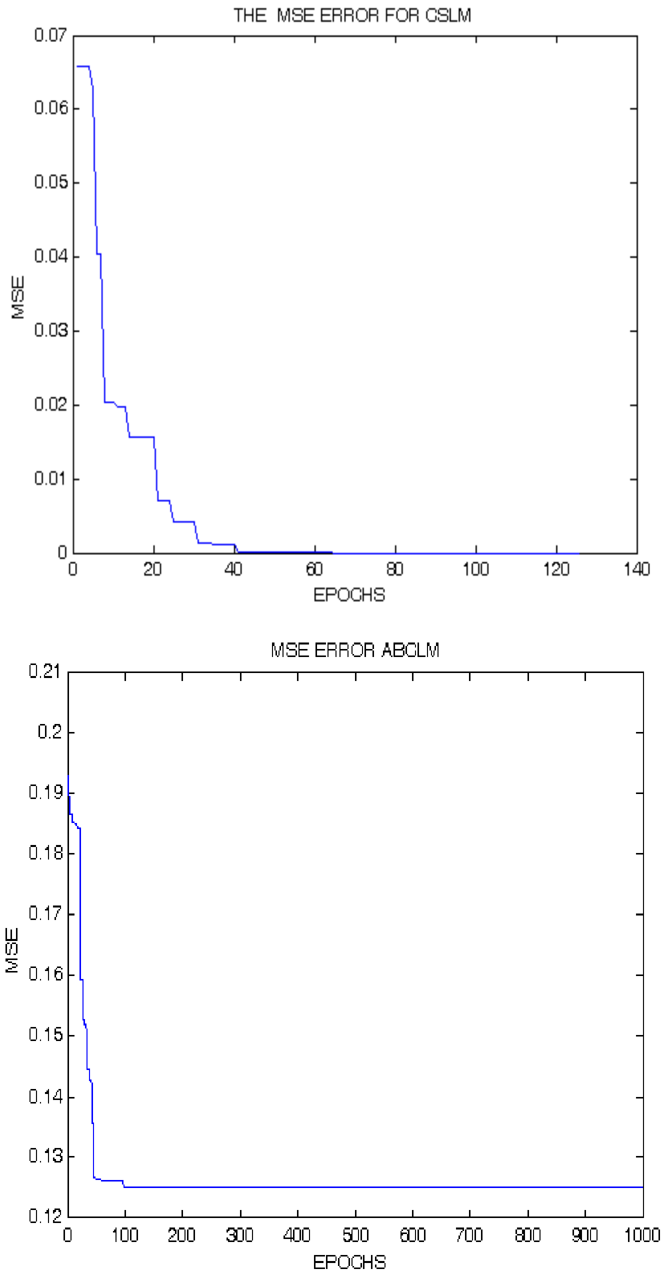

Figure 2 MSE for (a) CSLM and (b) ABC-LM on 2 Bit XOR and 2-5-1 NN architecture.

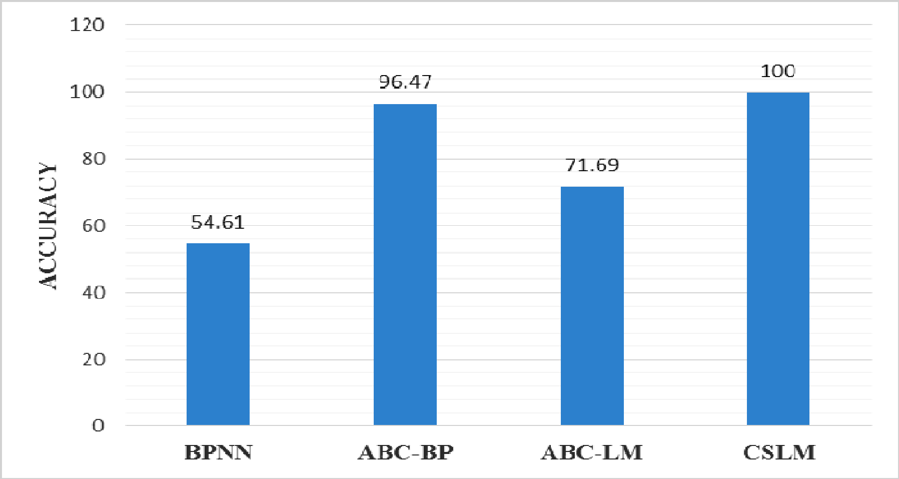

Figure 3 Average accuracy for 2-5-1 NN architecure.

4.3 Three Bit Exclusive-OR Problem

In the second phase, we used 3 bit XOR dataset which comprises of three inputs and a single binary output. The parameter range is same as used for two bit XOR problem. For the 3-5-1 the network, it has twenty connection weights and six biases. Table 2 shows the CPU time, number of epochs, the MSE standard deviation (SD) of the MSE, and accuracy for the 3 bit XOR test problem with 5 hidden neurons. Figure 4, displays the 'MSE performances vs. Epochs' of CSLM, and ABC-LM algorithms for the 3-5-1 network structure. While Figure 5, shows the average accuracy performance of the CSLM, ABC-LM, ABC-BP and BPNN, algorithms. For the three bit XOR, the CSLM algorithm converges to global minima within 671 epoch and 99.9% accuracy. From Table 2, it is clear that the proposed CSLM algorithm has better performance than BPNN, ABC-BP, ABC-LM in terms of MSE, Epochs, CPU time.

Table 2 CPU time, Epochs and MSE for 3-5-1 NN architecture.

Algorithm	BPNN	ABC-BP	ABC-LM	CSLM
CPUTIME	50.03	172.33	123.79	80.36
EPOCHS	1000	1000	1000	671
MSE	0.25	0.08	0.01	7.5E-07
SD	0.00064	0.0187	0.0055	3.14E-06

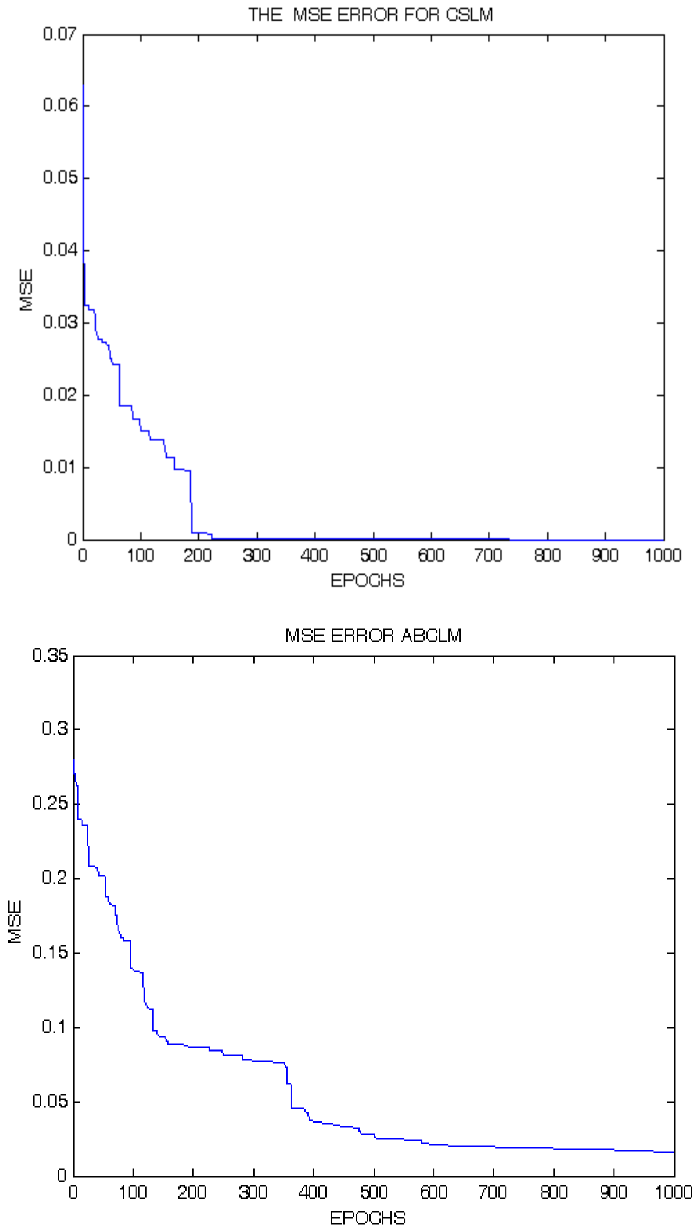

Figure 4 MSE for (a) CSLM and (b) ABC-LM on 3 BIT XOR and 2-5-1 NN architecture.

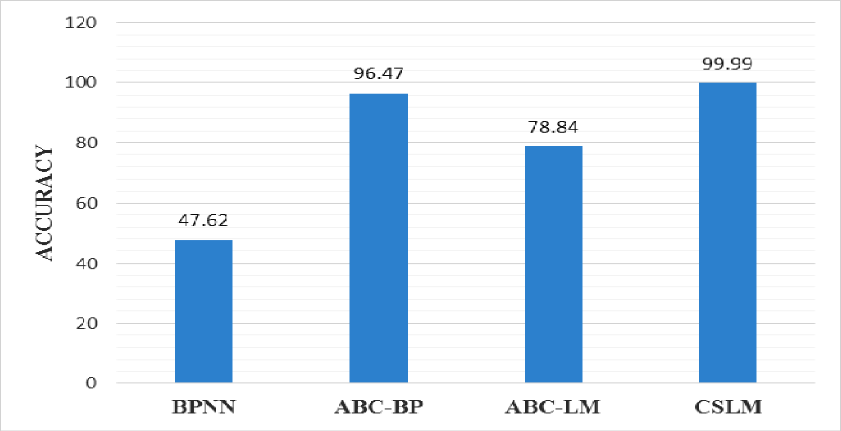

Figure 5 Average accuracy for 3-5-1 NN architecture.

4.4 The 4 Bit OR Problem

The third problem is 4 bit OR which consists of four inputs and a single binary output. The network structure is same as the 2 and 3 bit XOR problem. In 4 bit OR, if the number of inputs all is 0, the output is 0, otherwise the output is 1. Again for the four bit input we apply 4-5-1, feed forward neural network structure. For the 4-5-1 network structure it has twenty five connection weights and six biases. Table 3 illustrates the CPU time, epochs, and MSE performance of the proposed CSLM, ABC-BP, ABC-LM and BPNN algorithms respectively. Figure 6, shows the 'MSE performance vs. Epochs' for the 4-5-1, network structure of the proposed CSLM, and ABC-LM algorithms. From the Table 3, it can be observed that the proposed CSLM algorithm convergence rate is very fast than the other techniques in terms of CPU time and number of epochs. In Figure 7, the average accuracy comparison of the proposed CSLM, and ABC-BP, ABC-LM and BPNN algorithms can be seen. The proposed CSLM can be seen converging to global minima with 100% accuracy in Figure 7.

Table 3 CPU time, Epochs and MSE for 4-5-1 ANN architecture.

Algorithm	BPNN	ABC-BP	ABC-LM	CSLM
CPUTIME	63.280	162.494	118.7274	6.16
EPOCHS	1000	1000	1000	43
MSE	0.0527	1.91E-10	1.82E-10	0
SD	0.0084	1.5E-10	2.31E-11	0

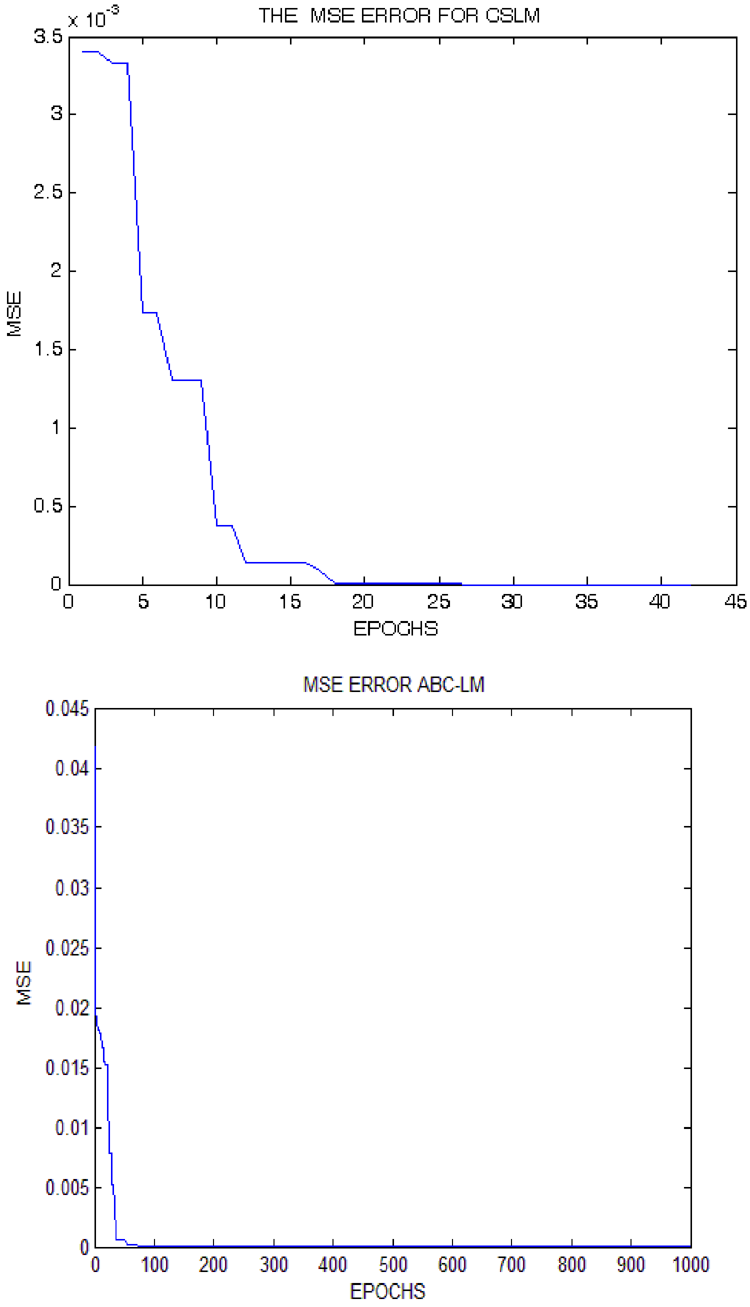

Figure 6 MSE for (a) CSLM and (b) ABC-LM on 4 Bit OR and 2-5-1 NN architecture.

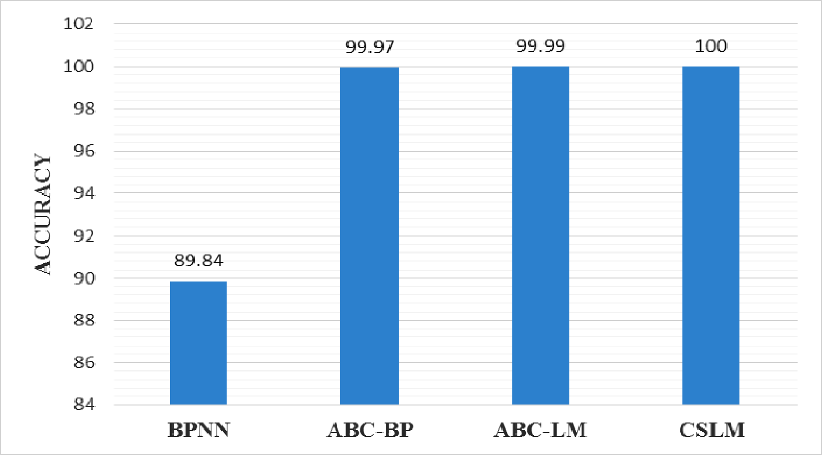

Figure 7 Average accuracy for 3-5-1 NN architecture.

5 Conclusion

In this paper, an improved Cuckoo Search and Levenberg Marquardt back propagation (CSLM) algorithm is proposed. The main idea of CSLM is that the CS is used at the beginning stage for generating the optimal weights, then LM continues the training by inheriting the best weights from CS algorithm [30]. The proposed CSLM algorithm is trained on XOR and OR datasets. The experimental results show that the proposed CSLM algorithm has better performance than ABC-LM, ABC-BP and BPNN algorithms in terms of MSE, number of epochs, (Standard Deviation) SD and accuracy.

Acknowledgements

The authors would like to thank Office of Research, Innovation, Commercialization and Consultancy Office (ORICC), Universiti Tun Hussein Onn Malaysia (UTHM) and Ministry of Higher Education (MOHE) Malaysia for financially supporting this Research under Fundamental Research Grant Scheme (FRGS) vote no. 1236.

References

[1] Radhika, Y. & Shashi, M., *Atmospheric Temperature Prediction using Support Vector Machines*, International Journal of Computer Theory and Engineering, **1**(1), pp. 1793-8201. 2009.

[2] Akcayol, M.A. & Cinar, C., *Artificial Neural Network Based Modeling of Heated Catalytic Converter Performance*, Journal of Applied Thermal Engineering, **25**, pp. 2341-2350, 2005.

[3] Shereef, K.I. & Baboo, S.S., *A New Weather Forecasting Technique using Back Propagation Neural Network with Modified Levenberg-Marquardt Algorithm for Learning*, IJCSI International Journal of Computer Science, **8**(6), pp. 1694-0814, 2011.

[4] Kosko, B., *Neural Network and Fuzzy Systems, 1st ed.*, Prentice Hall of India, 1994.

[5] Krasnopolsky, V.M. & Chevallier, F., *Some Neural Network application in environmental sciences. Part II: Advancing Computational Efficiency of Environmental Numerical Models*, Neural Networks, **16**(3-4), pp. 335-348, 2003.

[6] Coppin, B., *Artificial Intelligence Illuminated*, Jones and Bartlet Illuminated Series, USA, Chapter 11, pp. 291- 324, 2004.

[7] Basheer, I.A. & Hajmeer, M., *Artificial Neural Networks: Fundamentals, Computing, Design, and Application*, Journal of Microbiological Methods, **43**(1), pp. 03-31, 2000.

[8] Zheng, H., Meng, W. & Gong, B., *Neural Network and its Application on Machinery Fault Diagnosis*, IEEE International Conference on Systems Engineering (ICSYSE), pp. 576-579, 17-19 September, Kobe, Japan, 1992.

[9] Rehman, M.Z. & Nawi, N.M., *Improving the Accuracy of Gradient Descent Back Propagation Algorithm* (*GDAM*) *on Classification Problems*, International Journal of New Computer Architectures and their Applications (IJNCAA), **1**(4), pp.838-847, 2012.

[10] Rumelhart D.E., Hinton G.E. & Williams R.J., *Learning Representations by Back-Propagating Errors*, Nature, **323**, pp. 533-536, 1986.

[11] Lahmiri, S., *A Comparative Study of Backpropagation Algorithms in Financial Prediction*, International Journal of Computer Science, Engineering and Applications (IJCSEA), **1**(4), 2011.

[12] Nawi, N.M., Ransing, R.S. & AbdulHamid, N., *BPGD-AG: A New Improvement of Back-Propagation Neural Network Learning Algorithms with Adaptive Gain*, Journal of Science and Technology, **2**(2), 2011.

[13] Ahmed, W., Saad, E. & Aziz, E., *Modified Back Propagation Algorithm for Learning Artificial Neural Networks*, The 18th National Radio Science Conference (NRSC), pp. 345-352, 27-29 March, Mansoura, Egypt, 2001.

[14] Wen, J. Zhao, J.L., Luo. S.W. & Han, Z., *The Improvements of BP Neural Network Learning Algorithm*, 5th Int. Conf. on Signal Processing WCCC-ICSP, pp.1647-1649, 21-25 August, Beijing, China, 2000.

[15] Salchenberger, L.M., Cinar, E.M. & Lash N.A., *Neural Networks: A New Tool for Predicting Thrift Failures*, Decision Sciences, **23**(2), pp. 899-916, 1992.

[16] Sexton, R.S., Dorsey, R.E. & Johnson, J.D., *Toward Global Optimization of Neural Networks: A Comparison of the Genetic Algorithm and Backpropagation*, Decision Support Systems, **22**, pp. 171–186, 1998.

[17] SNNS (Stuttgart Neural Network Simulator), http://wwwra.informatik. unituebingen.de/SNNS/, 25th January, 2013.

[18] Fahlman, S.E., *Faster-Learning Variations on Back Propagation: An Empirical Study*, 1988 Connectionist Models Summer School by Scott E. Fahlman, pp. 38-51, San Mateo, CA, 1988.

[19] Riedmiller, M. & Braun. H., *A Direct Adaptive Method for Faster Back Propagation Learning: The RPROP Algorithm*, IEEE International Conference on Neural Networks (ICNN93), 28th March-1st April, San Francisco, CA, 1993.

[20] Fahlman, S.E. & Lebiere, C., *The Cascade-Correlation Learning Architecture*, Advances in Neural Information Processing Systems, **2**, pp. 524-532, San Mateo, Calif, 1990.

[21] Hagan, M.T. & Menhaj, M.B., *Training Feed Forward Networks with the Marquardt Algorithm*, IEEE Trans. on Neural Networks, **23**, pp. 899-916, 1994.

[22] Wilamowski, B.M., Cotton, N., Hewlett, J. & Kaynak, O., *Neural Network Trainer with Second Order Learning Algorithms*, 11th International Conference on Intelligent Engineering Systems, Budapest, Hungary, IEEE, 2007.

[23] Hagan, M.T. & Menhaj, M.B., *Training Feed Forward Networks with the Marquardt Algorithm*, IEEE Trans. Neural Network., **5**(6), pp. 989-993, 1994.

[24] Xiao-ping, C., Chang-hua, H., Zhi-qiang, Z. & Ying-jie, L., *Fault Prediction for Inertial Device Based on LMBP Neural Network*, Electronics Optics & Control, **12** (6), pp.38-41, 2005.

[25] Haykin, S., *Neural Networks*, Beijing, China Machine Press, pp. 501-522, 2004.

[26] Nawi, N.M., Khan, A. & Rehman, M.Z., *A New Levenberg Marquardt Based Back Propagation Algorithm Trained with Cuckoo Search*, ICEEI, UKM, 2013.

[27] Yan, J., Cao, H., Wang, J., Liu, Y. & Zhao, H., *Levenberg-Marquardt Algorithm Applied to Forecast the Ice Conditions in Ningmeng Reach of The Yellow River*, 5th International Conference on Natural Computation, pp. 184-188, 14-16 August, Tianjin, China, 2009.

[28] Yang, X. & Deb, S., *Cuckoo Search via Lévy Flights*, Proceedings of World Congress on Nature & Biologically Inspired Computing, India, pp. 210-214, 2009.

[29] Yang, X. & Deb, S., *Cuckoo Search via Levy Flights*, Nature and Biologically Inspired Computing (NaBIC 2009), pp. 210-214, 2009.

[30] Nawi, N.M., Khan, A. & Rehman, M.Z., *A New Levenberg-Marquardt based Back-propagation Algorithm trained with Cuckoo Search*, 4th International Conference on Electrical Engineering and Informatics (ICEEI), 24-25 June, Bangi, Malaysia, 2013.