A New Cuckoo Search Based Levenberg-Marquardt (CSLM) Algorithm

Nazri Mohd. Nawi, Abdullah Khan, and Mohammad Zubair Rehman

Faculty of Computer Science and Information Technology,
Universiti Tun Hussein Onn Malaysia (UTHM)
P.O. Box 101, 86400 Parit Raja, Batu Pahat, Johor Darul Takzim, Malaysia
nazri@uthm.edu.my, hi100010@siswa.uthm.edu.my,
zrehman862060@gmail.com

Abstract. Back propagation neural network (BPNN) algorithm is a widely used technique in training artificial neural networks. It is also a very popular optimization procedure applied to find optimal weights in a training process. However, traditional back propagation optimized with Levenberg marquardt training algorithm has some drawbacks such as getting stuck in local minima, and network stagnancy. This paper proposed an improved Levenberg-Marquardt back propagation (LMBP) algorithm integrated and trained with Cuckoo Search (CS) algorithm to avoided local minima problem and achieves fast convergence. The performance of the proposed Cuckoo Search Levenberg-Marquardt (CSLM) algorithm is compared with Artificial Bee Colony (ABC) and similar hybrid variants. The simulation results show that the proposed CSLM algorithm performs better than other algorithm used in this study in term of convergence rate and accuracy.

Keywords: Artificial neural network, back propagation, local minima, Levenberg-Marquardt, cuckoo search algorithm.

1 Introduction

Artificial Neural Networks (ANNs) is known for its competence in providing main features, such as: flexibility, ability of learning by examples, and capability to solve problems in pattern classification, function approximation, optimization, pattern matching and associative memories [1],[2]. Due to their powerful capability and functionality, ANN provides an alternative approach for many engineering problems that are difficult to solve by conventional approaches. ANNs are widely used in many areas such as signal processing, control, speech production, speech recognition and business [1]. Among many different neural network models, the multilayer feed- forward neural networks (MLFF) have been mainly used due to their well-known universal approximation capabilities [3]. The mostly popular MLFF training algorithms are the back-propagation (BP) algorithm and Levenberg Marquardt (LM), which are gradient-based methods [4]. Different techniques have been used in finding an optimal network performance for training ANNs such as evolutionary algorithms (EA),

B. Murgante et al. (Eds.): ICCSA 2013, Part I, LNCS 7971, pp. 438–451, 2013.

genetic algorithms (GA), particle swarm optimization (PSO), differential evolution (DE), and back propagation algorithm [5-8]. Therefore, a variety of NN models have been proposed. The most commonly used method to train the NN is based on back propagation [9-10]. The back-propagation (BP) learning has become the most standard method and process in adjusting weight and biases for training an ANNs in many domains [11]. Unfortunately, the most commonly used Error Back Propagation (EBP) algorithm [12-13] is neither powerful nor fast. It is also not easy to find the proper neural network architectures. Moreover another limitation of gradient-descent based technique, is that it requires a differentiable neuron transfer function. Also, as ANN generate complex error surfaces with multiple local minima, the BP fall prey to local minima instead of a global minima [14].

In recent years, many improved learning algorithms have been proposed to overcome the weakness of gradient-based techniques. These algorithms include a direct optimization method using a poly tope algorithm [14], a global search technique such as evolutionary programming [15], and genetic algorithm (GA) [16]. The standard gradient-descent BP is not trajectory driven, but population driven. However, the improved learning algorithms have explorative search features. Consequently, these methods are expected to avoid local minima frequently by promoting exploration of the search space. The Stuttgart Neural Network Simulator (SNNS) which was developed a decade ago and is constantly improving [17]. The SNNS uses 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. Unfortunately, all these algorithms are derivatives of steepest gradient search and training is relatively slow. For fast training, second order learning algorithms have to be used. The most effective method is Levenberg Marquardt algorithm (LM) [21], which is a derivative of the Newton method. This is a relatively complex algorithm since not only the gradient but also the Jacobian Matrix must be calculated. The LM algorithm was developed only for layer-by layer architectures, which is far from optimum [22]. LM algorithm is ranked as one of the most efficient training algorithms for small and medium sized patterns. LM algorithm was successfully implemented for neural network training in [23], but only for multilayer perceptron (MLP) architectures known as LMBP. LM has proved its mettle in improving the convergence speed of the network. It is the due to the good collaboration of Newton's method and steepest descent [24]. Not only it has the speed advantage of Newton's method, but also has the convergence character of the steepest descent method. Even though the LM algorithm is frequently used for fast convergence [26] but still, it is not devoid of local minima problem [26-27], [34].

In order to overcome the issues of slow convergence and network stagnancy, this paper propose a new algorithm that combines Cuckoo Search (CS) developed in 2009 by Yang and Deb [28-29] and Levenberg-Marquardt back propagation (LMBP) algorithm to train ANN for Exclusive-OR (XOR) datasets. The proposed Cuckoo Search Levenberg-Marquardt (CSLM) algorithm helps in reducing the error and avoids local minima. The remaining of the paper is organized as follows.

The remaining paper is organized as follows: Section 2 gives literature review on ANN. Section 3, explains Cuckoo Search via levy flight. In section 4, explain the proposed CSLM algorithm and the simulation results are discussed in section 5 respectively. Finally, the paper is concluded in the Section 6.

2 Artificial Neural Networks

Artificial Neural Networks (ANNs) imitates the learning processes of human cognitive system and the neurological functions of the brain. ANN works by processing information like Biological neurons in the brain and consists of small processing units known as Artificial Neurons, which can be trained to perform complex calculations [30]. An Artificial Neural Network (ANN) consists of an input layer, one or more hidden layers and an output layer of neurons. In ANN, every node in a layer is connected to every other node in the adjacent layer. An Artificial Neuron can be trained to store, recognize, estimate and adapt to new patterns without having the prior information of the function it receives. This ability of learning and adaption has made ANN superior to the conventional methods. Due to its capability to solve complex time critical problems, it has been widely used in the engineering fields such as biological modelling, financial forecasting, weather forecasting, decision modelling, control systems, manufacturing, medicine, ocean and space exploration etc [31-32]. ANN are usually classified into several categories on the basis of supervised and unsupervised learning methods and feed-forward and feed backward architectures [30].

3 Cuckoo Search (CS) Algorithm

Cuckoo Search (CS) algorithm is a novel meta-heuristic technique proposed by Xin-Shen Yang [28]. This algorithm was stimulated by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds. If an egg is discovered by the host bird as not their own then they will either throw the unknown egg away or simply abandon its nest and build a new nest somewhere else. Some other species have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicking the color and pattern of the eggs of a few chosen host species. This reduces the probability of their eggs being abandoned and thus increases their reproductively. The CS algorithm follows the three idealized rules:

- Each cuckoo lays one egg at a time, and put its egg in randomly chosen nest;
- The best nests with high quality of eggs will carry over to the next generations;
- 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 bird can either throw the egg away or abandon the nest, and build a completely new nest. 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;

Step 1: Generate initial population of N host nest i= 1... N
Step 2: ***while*** (f_{min}< MaxGeneration) or (stop criterion)
Do
Step 3: 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*** *Fi >Fj*
Then
Step 6: Replace j by the new solution.
Step 7: ***end if***
Step 8: A fraction (pa) of worse nest are abandoned and new ones are built.
Step 9: Keep the best solutions (or nest with quality solutions).
Step 10: Rank the solutions and find the current best.
Step 11: ***end while***

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

$$x_i^{t+1} = x_i^{t} + \alpha \oplus \text{levy}(\lambda) \tag{1}$$

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 efficient in exploring the search space as its step length is much longer in the long run.

$$\text{Lavy} \sim u = t^{-\lambda}, 1 < \lambda \leq 3 \tag{2}$$

This has an infinite variance with an infinite mean. Here the steps essentially construct a random walk process, 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 local optimum.

4 The Proposed CSLM Algorithm

The CS is a population based optimization algorithm, and similar to many others, meta-heuristic algorithms start with a random initial population, The CS algorithm essentially works with three components: 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) [33]. In Figure 1, 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 *pa* ε [0, 1].

In the proposed Cuckoo Search Levenberg-Marquardt (CSLM) algorithm, each best nest or solution represents a 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 population represents the quality of the solution. The initialization of weights is compared with output and the best weight cycle is selected by cuckoo. The cuckoo will continue searching until the last cycle to find the best weights for the network. The solution that is neglected by the cuckoo is replaced with a new best nest. The main idea of this combined algorithm is that CS algorithm is used at the beginning stage of searching for the optimum to select the best weights. Then, the training process is continued with the LM algorithm using the best weights of CS algorithm. The LM algorithm interpolate between the Newton method and gradient descent method. The LM algorithm is the most widely used optimization algorithm. It outperforms simple gradient conjugate descent and gradient methods in a wide other variety of problems [35]. The flow diagram of proposed CSLM algorithm is shown in

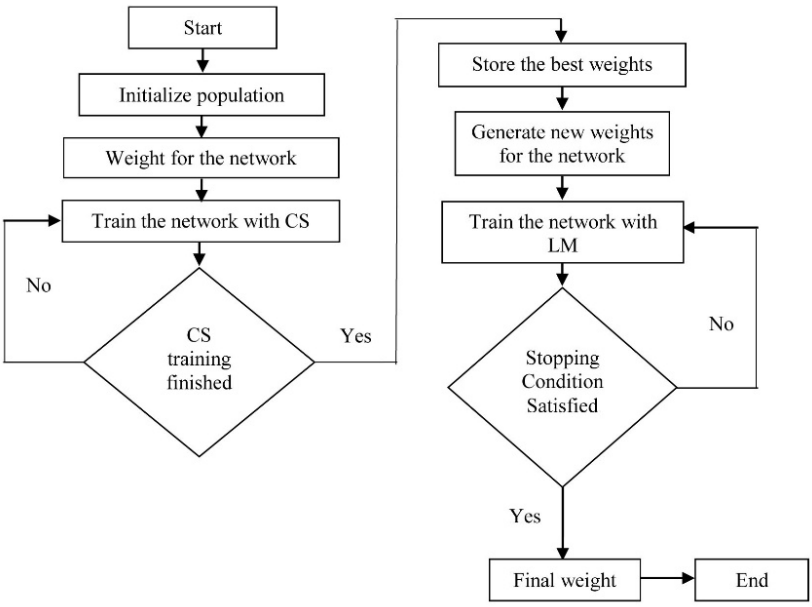

Fig. 1. The Proposed CSLM Algorithm

Figure 1. In the first stage CS algorithm finished its training, then LM Algorithm start training with the weights from CS algorithm and the LM continue to train the network until the stopping criteria or Mean Square Error (MSE) is achieved.

5 Experiments and Results

In ordered to illustrate the performance of the proposed CSLM algorithm in training ANN. CSLM algorithm is tested on 2-bit, 3-bit XOR, and 4-bit OR parity problems. The simulation experiment are performed on an AMD Athlon 1.66 GHz CPU with a 2-GB RAM. The software used for simulation process is MATLAB 2009b.

The proposed CSLM algorithm is compared with Artificial Bee Colony Levenberg- Marquardt (ABCLM), Artificial Bee Colony Back-Propagation (ABCBP) and conventional Back-Propagation Neural Network (BPNN) algorithms respectively. The performance measure for each algorithm is based on the Mean Square Error (MSE). The three layers feed forward neural network architecture (i.e. input layer, one hidden layer, and output layers.) is used for each problem. The number of hidden nodes is varied to 5 and 10 neurons. In the network structure, the bias nodes are also used and the log-sigmoid activation function is applied. . For each problem, trial is limited to 1000 epochs. And MSE criteria is kept to 0. A total of 20 trials are run for each case. The network results are stored in the result file for each trial.

5.1 The 2 Bit XOR Problem

The first test problem is the Exclusive OR (XOR) Boolean function of two binary input to a single binary output as (0 0; 0 1; 1 0 ; 1 1) to (0; 1; 1 ;0). In the simulations, we used 2-5-1, 2-10-1 MLFF network for two bit XOR problem. The parameters range for the upper and lower band is used [5,-5], [5,-5], [5,-5], [1,-1] respectively. For the CSLM, ABCLM, ABCBP and BPNN, Table 1 and Table 2

Table 1. CPU Time, Epochs and MSE for **2-5-1** ANN Structure

Algorithm	ABCBP	ABCLM	BPNN	CSLM
CPUTIME	172.3388	123.9488	42.64347	14.41
EPOCHS	1000	1000	1000	126
MSE	2.39E-04	0.125	0.220664	0
Accuracy (%)	96.47231	71.69041	54.6137	100

Table 2. CPU Time, Epochs and MSE for 2-10-1 ANN Structure

Algorithm	ABCBP	ABCLM	BPNN	CSLM
CPUTIME	197.34	138.96	77.63	18.61
EPOCHS	1000	1000	1000	153
MSE	8.39E-04	0.12578	0.120664	0
Accuracy (%)	96.8	71.876	54.6137	100

shows the CPU time, number of epochs and the MSE for the 2 bit XOR test problem with 5 and 10 hidden neurons. Figure 2 and Figure 3 shows the mean square error of CSLM algorithm and ABCBP algorithm for the 2-5-1 network structure.

The CSLM algorithm avoids the local minima and trained the network successfully within 145, 153 epochs as seen in the Table 1 and Table 2. Both Tables, show that CSLM can converge successfully for almost every kind of network structure. In Figure 2, the CSLM algorithm can be seen to converge within 153 epochs which is quite superior convergence rate as compared to the other algorithms. The ABCBP algorithm is showing a lot of oscillations in the trajectory path and not converging within 100 epochs as shown in the Figure 3. BPNN shows many failures in convergence to the global solution. The average CPU time and MSE of CSLM is also found to be less than BPNN, ABCBP, and ABCLM algorithms.

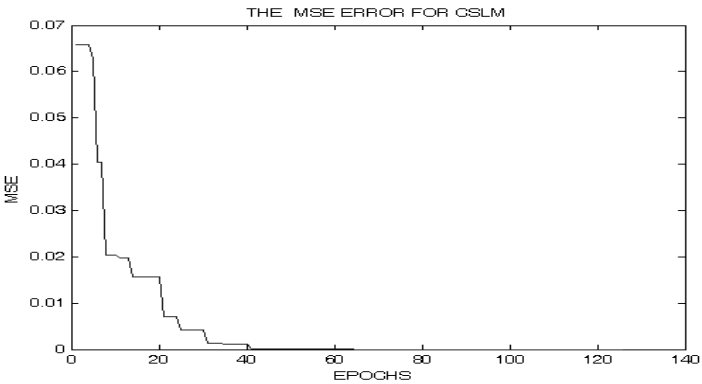

Fig. 2. MSE for CSLM using 2-5-1 ANN Structure

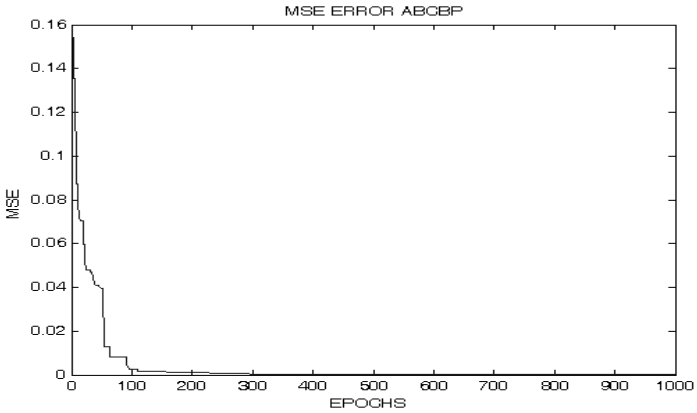

Fig. 3. MSE for ABCBP using 2-5-1 ANN Structure

5.2 The 3 Bit XOR Problem

In the second case we used three bit Exclusive OR test problem of three input to a single binary output as (1 1 1; 1 1 0 ; 1 0 1; 1 0 0; 0 1 1; 0 1 0 ; 0 0 1; 0 0 0) –to-(1; 0; 0; 1; 0; 1; 1; 0). In three bit XOR input if the number of binary inputs is odd, the output is 1, otherwise the output is 0. Also for the three bit input we apply 3-5-1, and 3-10-1, feed forward neural network structure. In Figure 4 and Figure 5 we can see the simulation result of 3 bit XOR problem. The parameter range is same as used for two bit XOR problem. For the 3-5-1 network structure it has twenty connection weights and six biases, and for the 3-10-1 the network has forty connection weights and eleven biases.

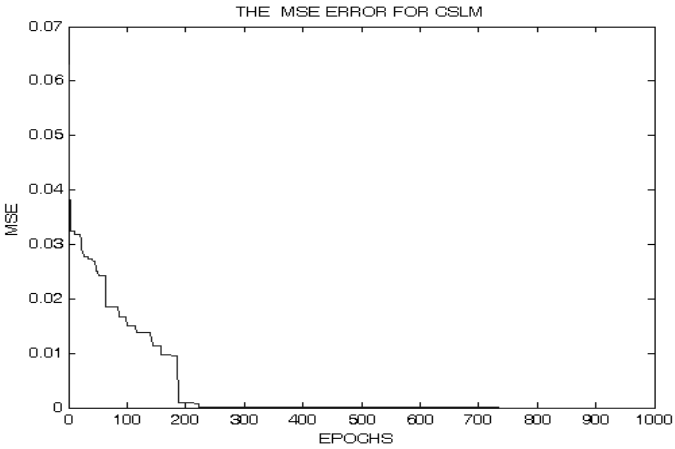

Fig. 4. MSE for CSLM using 3-5-1 ANN Structure

In Figure 4 and Figure 5 we can see the simulation results of 3 bit XOR problem. Figure 4 illustrates the 'MSE vs. Epochs' of CSLM algorithm. While Figure 5 shows the 'MSE vs. Epochs' of the BPNN algorithm. CSLM algorithm can be seen converging with 0 MSE within 80.53 seconds CPU time and 671 epochs in Figure 4. While in Figure 5, ABCLM is seen as converging with 0.0056 MSE and 125.5 seconds CPU time. In three bit XOR dataset, CSLM has fulfilled all criterion during convergence; such as less MSE, less CPU cycles, high Accuracy, and less no of epochs.

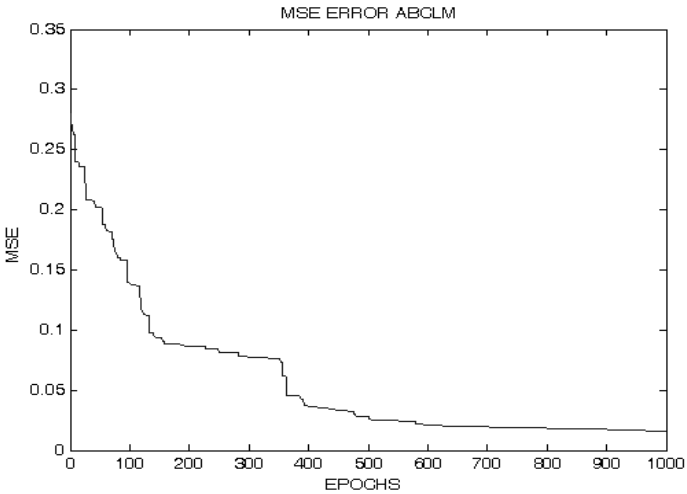

Fig. 5. MSE for BPNN using 3-5-1 ANN Structure

5.3 The 4 Bit OR Problem

The third problem is 4 bit OR. In case we used four bit OR test problem of four input to a single binary output as (1 1 1 1;1 1 1 0;1 1 0 1;1 1 0 0; 1 0 1 1;1 0 0 1; 1 0 0 0;0 1 1 1;0 1 1 0; 0 1 0 1; 0 1 0 0; 0 0 1 1 ; 0 0 1 0; 0 0 0 1;1 0 1 0; 0 0 0 0) –to –(1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 1; 0). The network structure is same as the 2 and 3 bit XOR problem. In 4 bit OR input 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, 4-10-1, feed forward neural network structure. For the 4-5-1 network structure it has twenty five connection weights and six biases, and for the 4-10-1 the network has fifty connection weights and eleven biases. Table 3 and Table 4 illustrates the CPU time, epochs, and MSE performance of the proposed CSLM algorithm, ABCBP, ABCLM and BPNN algorithms respectively. Figure 6, and Figure 7, shows the 'MSE performance vs. Epochs' for the 4-5-1, and 4-10-1 network structure of the proposed CSLM algorithm. While in Figure 8, we see the 'MSE performance vs. Epochs' for the network structure 4-5-1 of ABCBP algorithm, and Figure 9 shows the 'MSE performance vs. Epochs' of ABCLM algorithm for 4-10-1 network structure.

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

Algorithm	ABCBP	ABCLM	BPNN	CSLM
CPU TIME	162.4945	118.7274	63.28089	6.16091
EPOCHS	1000	1000	1000	43
MSE	1.91E-10	1.82E-10	0.052778	0
Accuracy (%)	99.97	99.99572	89.83499	100

Table 4. CPU time, Epochs and MSE error for 4-10-1 ANN structure

Algorithm	ABCBP	ABCLM	BPNN	CSLM
CPU TIME	180.4945	129.7274	67.28089	8.65
EPOCHS	1000	1000	1000	46
MSE	1.67E-10	1.76E-10	0.05346	0
Accuracy (%)	99.47	99.78	89.83499	100

Table 3 illustrates the CPU time, MSE and epochs for the 4-5-1 network structure. While, Table 4 shows the CPU time MSE error and epochs with 4-10-1 network structure. From both Tables, we can see that CSLM algorithm outperforms ABCBP, ABCLM, and BPNN in-terms of CPU time, epochs, MSE, and accuracy. Figure 6

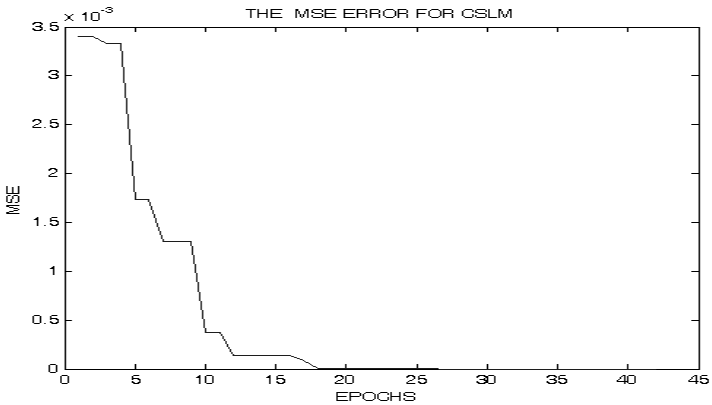

Fig. 6. MSE for CSLM using 4-5-1 ANN Structure

and Figure 7 shows the 'MSE performance vs. Epochs' for the 4-5-1, and 4-10-1 network structure for CSLM algorithm. In both Figures, CSLM is converging within 43, 46 epochs. For the 4-5-1 and 4-10-1 network structure CSLM again has 0 MSE. While in Figure 8 we see that for the network structure 4-5-1, ABCBP is converging within 1000 epochs which is quite a large number if compared with CSLM. Also, ABCLM algorithm for 4-10-1 network structure shows a large MSE and more CPU time then the proposed CSLM algorithm.

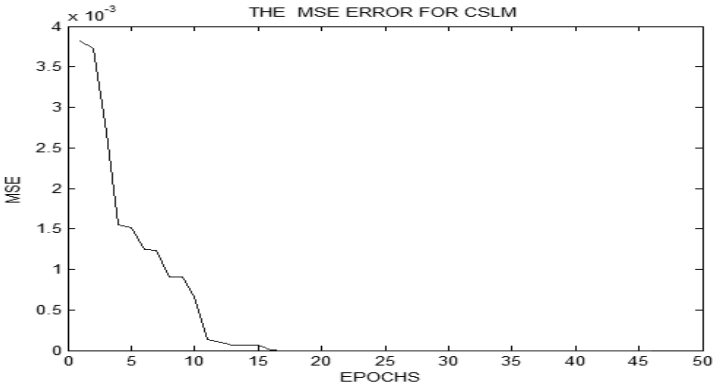

Fig. 7. MSE for CSLM using 4-10-1 ANN Structure

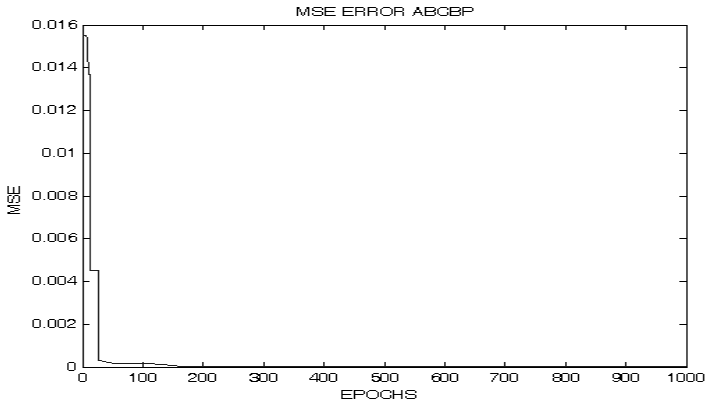

Fig. 8. MSE for ABCBP using 4-10-1 ANN Structure

5.4 Overall Result

The overall MSE results of the algorithms for 2 bit XOR, 4 bit OR problem, are given in above Tables. From the above Tables, it is clear that the CSLM algorithm obtained the best results as compared to the ABCBP, ABCLM, and BPNN in term of CPU time, MSE, and accuracy. the over all results show that CSLM perform better result on 4-bit OR dataset then 2 and 3 bit XOR dataset in term of CPU time and epochs.

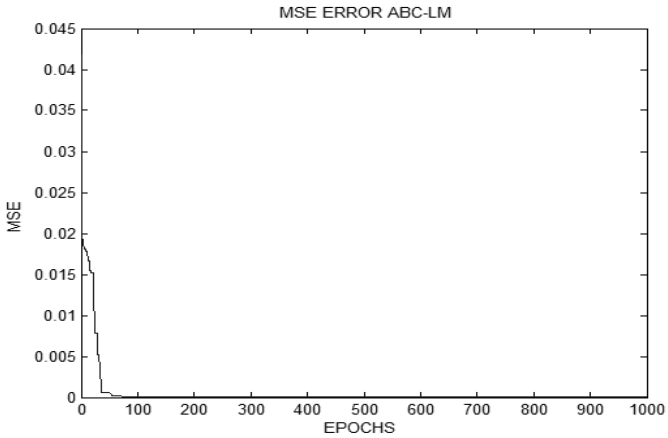

Fig. 9. MSE for ABCLM using 4-10-1 ANN Structure

6 Conclusion

Traditional back propagation optimized with Levenberg marquardt training algorithm has some drawbacks such as getting stuck in local minima, and network stagnancy [4]. In this paper, an improved Levenberg-Marquardt back propagation (LMBP) algorithm integrated and trained with Cuckoo Search (CS) algorithm. In the proposed CSLM algorithm, first CS algorithm trains the network and LM continues training by taking the best weight-set from CS and tries to minimize the training error avoided local minima problem and achieve fast conversances. The proposed CSLM algorithm is used to train MLFF on the 2-bit XOR, 3- Bit XOR and 4-Bit OR benchmark problems. The results show that the CSLM is simple and generic for optimization problems and has better convergence rate and accuracy than the ABCLM, ABCBP and BPNN algorithms. In future this proposed model will be used benchmarks data classification.

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. Dayhoff, J.E.: Neural-Network Architectures: An Introduction, 1st edn. Van Nostrand Reinhold Publishers, New York (1990)
2. Mehrotra, K., Mohan, C., Ranka, S.: Elements of Artificial Neural Networks. MIT Press, Cambridge (1997)

3. Rehman, M.Z., Nazri, M.N.: Studying the Effect of adaptive momentum in improving the accuracy of gradient descent back propagation algorithm on classification problems. International Journal of Modern Physics (IJMPCS) 9(1), 432–439 (2012)
4. Ozturk, C., Karaboga, D.: Hybrid Artificial Bee Colony algorithm for neural network training. In: IEEE Congress of Evolutionary Computation (CEC), pp. 84–88 (2011)
5. Leung, C., Member, C.: A Hybrid Global Learning Algorithm Based on Global Search and Least Squares Techniques for back propagation neural network Networks. In: International Conference on Neural Networks, pp. 1890–1895 (1994)
6. Yao, X.: Evolutionary artificial neural networks. International Journal of Neural Systems 4(3), 203–222 (1993)
7. Mendes, R., Cortez, P., Rocha, M., Neves, J.: Particle swarm for feed forward neural network training. In: Proceedings of the International Joint Conference on Neural Networks, vol. 2, pp. 1895–1899 (2002)
8. Nawi, N.M., Ghazali, R., Salleh, M.N.M.: The development of improved back-propagation neural networks algorithm for predicting patients with heart disease. In: Zhu, R., Zhang, Y., Liu, B., Liu, C. (eds.) ICICA 2010. LNCS, vol. 6377, pp. 317–324. Springer, Heidelberg (2010)
9. Abid, S., Fnaiech, F., Najim, M.: A fast feedforward training algorithm using a modified form of the standard backpropagation algorithm. IEEE Transactions on NeuralNetworks 12, 424–430 (2001)
10. Yu, X., OnderEfe, M., Kaynak, O.: A general backpropagation algorithm for feedforward neural networks learning. IEEE Transactions on Neural Networks 13, 251–259 (2002)
11. Chronopoulos, A.T., Sarangapani, J.: A distributed discrete time neural network architecture forpattern allocation and control. In: Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS 2002), Florida, USA, pp. 204–211 (2002)
12. Wilamowski, B.M.: Neural Networks and Fuzzy Systems. CRC Press (2002)
13. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning Internal Representations by error Propagation. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1 (1986)
14. Gupta, J.N.D., Sexton, R.S.: Comparing backpropagation with a genetic algorithm for neural network training. The International Journal of Management Science 27, 679–684 (1999)
15. Salchenberger, L.M., Cinar, E.M., Lash, N.A.: Neural Networks: A New Tool for Predicting Thrift Failures. Decision Sciences 23(4), 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 Back propagation. Decision Support Systems 22, 171–186 (1998)
17. SNNS (Stuttgart Neural Network Simulator) (2013), `http://wwwra.informatik.unituebingen.de/SNNS/` (Accessed February 2, 2013)
18. Fahlman, S.E.: Faster-Learning Variations on Back-Propagation: An Empirical Study. In: Proceedings of the 1988 Connectionist Models Summer School. Morgan-Kaufmann, Los Altos (1988)
19. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: Proceedings of the IEEE International Conference on Neural Networks, ICNN 1993 (1993)
20. Fahlman, S.E., Lebiere, C.: The Cascade-Correlation Learning Architecture. In: Touretzky, D.S. (ed.) Advances in Neural Information Processing Systems 2. Morgan-Kaufmann, Los Altos (1990)

21. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. IEEE Transactions on Neural Networks 23, 899–916 (1994)
22. Wilamowski, B.M., Cotton, N., Kaynak, O.: Neural Network Trainer with Second Order Learning Algorithms. In: 11th International Conference on Intelligent Engineering Systems, Budapest, Hungary (2007)
23. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. IEEE Trans. Neural Networks 5(6), 989–993 (1994)
24. Cao, X.P., Hu, C.H., Zheng, Z.Q., Lv, Y.J.: Fault Prediction for Inertial Device Based on LMBP Neural Network. Electronics Optics & Control 12(6), 38–41 (2005)
25. Haykin, S.: Neural Networks: A Comprehensive Foundation. China Machine Press, Beijing (2004)
26. Xue, Q., et al.: Improved LMBP Algorithm in the Analysis and Application of Simulation Data. In: 2010 International Conference on Computer Application and System Modeling (2010)
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. In: Fifth International Conference on Natural Computation (2009)
28. Yang, X.S., Deb, S.: Cuckoo search via Lévy flights. In: Proceedings of World Congress on Nature & Biologically Inspired Computing, India, pp. 210–214 (2009)
29. Yang, X.S., Deb, S.: Cuckoo search via levy flights. In: Nature Biologically Inspired Computing (NaBIC), pp. 210–214 (2009)
30. Deng, W.J., Chen, W.C., Pei, W.: Back-propagation neural network based importance-performance analysis for determining critical service attributes. Expert Systems with Applications 4(2) (2008)
31. Kosko, B.: Neural Network and Fuzzy Systems, 1st edn. Prentice Hall of India (1994)
32. Lee, T.: Back-propagation neural network for the prediction of the short-term storm surge in Taichung harbor, Taiwan. Engineering Applications of Artificial Intelligence 21(1) (2008)
33. Yang, X.S., Deb, S.: Engineering optimisation by Cuckoo Search. International Journal of Mathematical Modelling and Numerical Optimisation 1, 330–343 (2010)
34. Kumar, M.P.: Backpropagation Learning Algorithm based on Levenberg Marquardt Algorithm. In: Proceedings of CSCP, pp. 393–398 (2012)
35. Nourani, E., Rahmani, A.M., Navin, A.H.: Forecasting Stock Prices using a hybrid Artificial Bee Colony based Neural Network. In: International Conference on Innovation, Management and Technology Research (ICIMTR 2012), Malacca, Malaysia (2012)
36. Chaowanawatee, K., Heednacram, H.: Implementation of Cuckoo Search in RBF Neural Network for Flood Forecasting. In: Fourth International Conference on Computational Intelligence, Communication Systems and Networks, pp. 22–26 (2012)