Comparing Performances of Cuckoo Search Based Neural Networks

Nazri Mohd Nawi[1,2], Abdullah Khan[2], M.Z. Rehman[2],
Tutut Herawan[3,4], and Mustafa Mat Deris[2]

[1] Software and Multimedia Centre (SMC)
[2] Faculty of Computer Science and Information Technology
Universiti Tun Hussein Onn Malaysia
86400, Parit Raja, Batu Pahat, Johor, Malaysia
[3] Department of Information System
University of Malaya
50603 Pantai Valley, Kuala Lumpur, Malaysia
[4] AMCS Research Center, Yogyakarta, Indonesia
{nazri,mmustafa}@uthm.edu.my, hi100010@siswa.uthm.edu.my,
zrehman862060@gmail.com, tutut@um.edu.my

Abstract. Nature inspired meta-heuristic algorithms provide derivative-free solutions to solve complex problems. Cuckoo Search (CS) algorithm is one of the latest additions to the group of nature inspired optimization heuristics. In this paper, Cuckoo Search (CS) is implemented in conjunction with Back propagation Neural Network (BPNN), Recurrent Neural Network (RNN), and Levenberg Marquardt back propagation (LMBP) algorithms to achieve faster convergence rate and to avoid local minima problem. The performances of the proposed Cuckoo Search Back propagation (CSBP), Cuckoo Search Levenberg Marquardt (CSLM) and Cuckoo Search Recurrent Neural Network (CSRNN) algorithms are compared by means of simulations on OR and XOR datasets. The simulation results show that the CSRNN performs better than other algorithms in terms of convergence speed and Mean Squared Error (MSE).

Keywords: Back propagation, Neural network, Cuckoo search, Local minima, Meta-heuristic optimization.

1 Introduction

An Artificial Neural Network (ANN) is a data processing model that emulates the biological nervous systems operations to processes data [1, 2]. ANN consists of a large number of tremendously integrated processing elements (neurons) functioning together to solve 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 this complex processing quality, many new

T. Herawan et al. (eds.), *Recent Advances on Soft Computing and Data Mining*
SCDM 2014, Advances in Intelligent Systems and Computing 287,
DOI: 10.1007/978-3-319-07692-8_16,

algorithms have been proposed that inherit the architecture of ANNs in the recent decades. The Back propagation (BP) algorithm is one such architecture of ANN by Rumelhart [10] well-known for training a multilayer feed-forward ANN [11]. However, BP algorithm suffers from two major drawbacks: low convergence rate and instability. They are caused due to getting trapped in a local minimum or due to the overshooting of the minimum of the error surface [12-14]. In recent years, a number of studies have attempted to overcome these problems. They fall into two main categories. The first category uses heuristic techniques developed from an analysis of the performance of the standard steepest descent algorithm. They include the gradient descent with adaptive learning rate, gradient descent with momentum, gradient descent with momentum and adaptive learning rate, and the resilient algorithm. In the standard steepest descent, the learning rate is fixed and its optimal value is always hard to find [11, 12]. The second category uses standard numerical optimization techniques. This includes conjugate gradient [15, 16], quasi-Newton, and Levenberg-Marquardt (LM) algorithm. In the conjugate gradient algorithms, search is performed along conjugate directions. However, one limitation of this procedure, which is a gradient-descent technique, is that it requires a differentiable neuron transfer function. Also, as neural networks generate complex error surfaces with multiple local minima, the BP fall into local minima in place of a global minimum [17, 18]. Many methods have been proposed to speed up the back propagation based training algorithms by fixing the proper learning rate and the momentum value for each layer at the time of training [19]. Different initialization techniques [20, 21] and cost optimization techniques [22], and global search technique such as hybrid PSO-BP [23], Artificial Bee Colony [24-26], Evolutionary algorithms (EA) [27], Particle Swarm Optimization (PSO) [28], Differential Evolution (DE) [29], Ant Colony, and Back propagation [30], Genetic algorithms (GA) [31], have been introduced to intensify the rate of convergence. Cuckoo Search (CS) is a new meta-heuristic search algorithm, developed by Yang and Deb [32] which imitates animal behavior and is valuable for global optimization [33, 34]. The CS algorithm has been applied alone to solve several engineering design optimization problems, such as the design of springs and welded beam structures, and forecasting [33, 35].

In this paper, Cuckoo Search (CS) is implemented in conjunction with Back propagation Neural Network (BPNN), Recurrent Neural Network (RNN), and Levenberg Marquardt back propagation (LMBP) algorithms to achieve faster convergence rate and to avoid local minima problem. The performances of the proposed Cuckoo Search Back propagation (CSBP), Cuckoo Search Levenberg Marquardt (CSLM) and Cuckoo Search Recurrent Neural Network algorithms are compared with other similar hybrid variants based on Mean Squared Error (MSE), CPU time, accuracy and convergence rate to global minima.

The remaining paper is organized as follows: Learning algorithms are presented in the Section 2, while Section 3 deals with the simulation results and discussions and finally the paper is concluded in the Section 4.

2 Learning Algorithms

2.1 Levy Flight

Levy Flights have been used in many search algorithms [37]. In Cuckoo Search algorithm levy flight is an important component for local and global searching [38]. Levy Flights is a random walk that is characterized by a series of straight jumps chosen from a heavy-tailed probability density function [37]. In statistical term, it is a stochastic algorithm for global optimization that finds a global minimum [38]. Each time Levy Flights processes, step length can be calculated by using Mantegna's algorithm as given in the Equation 1.

$$S = \frac{u}{|v|^{\frac{1}{\beta}}} \tag{1}$$

Note that u and v are drawn from normal distribution with respect to these two random variables;

$$u \sim N(0.\sigma_u{}^2), \qquad v \sim N(0.\sigma_v{}^2) \tag{2}$$

The $\sigma_u{}^2$ and $\sigma_v{}^2$ present in Equation 2 are the variance of distributions which come from Equation 3;

$$\sigma_u = \left\{ \frac{\Gamma(1+\beta).\sin(\tau.\beta/2)}{\Gamma[(1+\beta)/2].\beta.2^{\frac{(\beta-1)}{2}}} \right\}^{\frac{1}{\beta}}, \qquad \sigma_v = 1 \tag{3}$$

The symbol ~ in Equations 2 means the random variable obeys the distribution on right hand side; that is, samples should draw from the distribution.

2.2 Cuckoo Search (CS) Algorithm

Cuckoo Search (CS) algorithm is a new meta-heuristic technique proposed by Xin-She Yang [32]. This algorithm was stimulated by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds. Some host nest can keep direct difference. If an egg is discovered by the host bird as not its own, it will either throw the unknown egg away or simply abandon its nest and build a new nest elsewhere. The CS algorithm follows three idealized rules:

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

In this case, the host bird can either throw the egg away or abandon the nest, and build a completely new nest. The Rule (c) defined above can be approximated by the fraction pa $\in [0, 1]$ of the n nests that are replaced by new nests (with new random solutions). When generating new solutions x^{t+1} for a cuckoo i, a Levy flight is performed;

$$x_i^{t+1} = x_i^t + \alpha \oplus \text{levy}(\lambda) \,, \tag{4}$$

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. The Levy flight essentially provides a random walk while the random step length is drawn from a Levy distribution as shown in the Equation 5:

$$\text{Lavy} \sim u = t^{-\lambda} \,, 1 < \lambda \leq 3 \tag{5}$$

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 and a heavy tail.

2.3 Levenberg Marquardt (LM) Algorithm

One reason for selecting a learning algorithm is to speed up convergence. The Levenberg-Marquardt (LM) algorithm is an approximation to Newton's method to accelerate training speed. Benefits of applying LM algorithm over variable learning rate and conjugate gradient method were reported in [39]. The LM algorithm is developed through Newton's method. Assume the error function is:

$$E(t) = \frac{1}{2} \sum_{i=1}^{N} e_i^{\,2}(t) \,, \tag{6}$$

where, $e(t)$ is the error; N is the number of vector elements, then:

$$\nabla E(t) = J^T(t) e(t) \tag{7}$$

$$\nabla^2 E(t) = J^T(t) J(t) + S(t) \,, \tag{8}$$

where, $\nabla E(t)$ is the gradient; $\nabla^2 E(t)$ is the Hessian matrix of E(t);

$$S(t) = \sum_{i=1}^{N} e_i(t) \nabla^2 e_i(t) \tag{9}$$

where $J(t)$ is the Jacobian Matrix;

$$J(t) = \begin{bmatrix} \frac{\partial v_1(t)}{\partial t_1} & \frac{\partial v_1(t)}{\partial t_2} & \cdots & \frac{\partial v_1(t)}{\partial t_n} \\ \frac{\partial v_2(t)}{\partial t_1} & \frac{\partial v_2(t)}{\partial t_2} & \cdots & \frac{\partial v_2(t)}{\partial t_n} \\ \vdots & & & \\ \frac{\partial v_n(t)}{\partial t_1} & \frac{\partial v_n(t)}{\partial t_2} & \cdots & \frac{\partial v_n(t)}{\partial t_n} \end{bmatrix} \tag{10}$$

For Gauss-Newton Method:

$$\nabla t = -[J^T(t)J(t)]^{-1}J(t)e(t) \tag{11}$$

For the Levenberg-Marquardt algorithm as the variation of Gauss-Newton Method:

$$\nabla t = -[J^T(t)J(t) + \mu I]^{-1}J(t)e(t) \tag{12}$$

where $\mu > 0$ and is a constant; I is identity matrix. So that the algorithm will approach Gauss- Newton, which should provide faster convergence.

3 Results and Discussions

Basically, the main focus of this paper is to compare of different algorithm on based of error, accuracy in network convergence. Before going to discussing the simulation results, there are certain things that need be explained such as tools and technologies, network topologies, testing methodology and the classification problems used for the entire experimentation. The discussion is as follows;

3.1 Preliminary Study

In order to illustrate the performance of the algorithm in training ANN. The simulation experiment is performed on Intel Pentium 3.00 GHz CPU with a 2-GB RAM. The software used for simulation is MATLAB 2008a. In this paper these following algorithms are compared analyzed and simulated on 2-bit XOR, 3-bit XOR and 4-bit OR datasets;

a) Cuckoo Search Levenberg Marquardt (CSLM) algorithm [40],
b) Cuckoo Search Back propagation (CSBP) algorithm [41],
c) Cuckoo Search Recurrent neural network (CSRNN) [42],

The performance measure for each algorithm is based on the Mean Square Error (MSE), standard deviation and accuracy. 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 keep fixed to 5. 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. A total of 20 trials are run for each dataset. The network results are stored in the result file for each trial. CPU time, average accuracy, and Mean Square Error (MSE) are recorded for each independent trial on XOR and OR datasets.

3.2 2-Bit XOR Dataset

The first test problem is the 2 bit XOR Boolean function of two binary inputs and a single binary output. In the simulations, we used 2-5-1, feed forward neural network

for two bit XOR dataset. The parameters range for the upper and lower band is set to [5,-5] respectively, for the CSLM, CSBP, and CSRNN algorithm. Table 1 shows the CPU time, number of epochs, MSE, Standard Deviation (SD), and Accuracy for the 2 bit XOR test problem with 5 hidden neurons. Figure1 shows the 'MSE performances vs. Epochs' of CSLM, CSBP, and CSRNN algorithms for the 2-5-1 network structure. Although, CSLM, CSBP, and CSRNN performed well on 2-Bit XOR but the convergence rate of CSRNN was better. The CSRNN converged in 7 epochs and took 0.78 CPU seconds. Figure 1 shows the graphical representation of the MSE, SD, Epoch, and Accuracy. From Figure 1, it's conformed that CSRNN is better in terms of convergence and CPU time than the CSBP and CSLM algorithms.

Table 1. CPU Time, Epochs, MSE, Accuracy, Standard Deviation (SD) for 2-5-1 ANN Architecture

Algorithm	CSRNN	CSBP	CSLM
CPU Time	0.78	21.22	15.80
Epochs	7	134	145
MSE	0	0	0
SD	0	0	0
Accuracy (%)	100	100	100

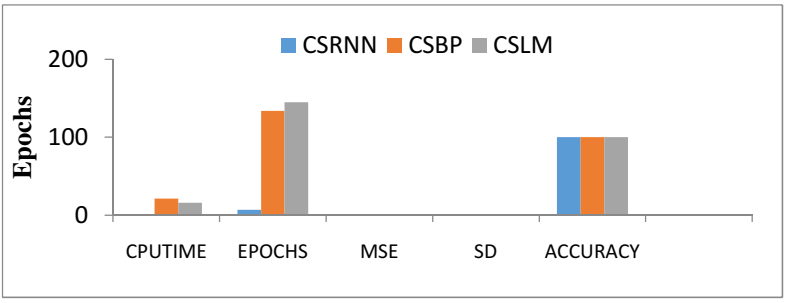

Fig. 1. Performance Comparison of CSRNN, CSLM, and CSBP algorithms on the basis of CPU Time, Epochs, MSE, Accuracy, Standard Deviation (SD) for 2-5-1 ANN Architecture

3.3 3-Bit XOR Dataset

In the second phase, we used 3 bit XOR dataset consisting of three inputs and a single binary output. For the three bit input we apply 3-5-1, network architecture. The parameter range is same as used for two bit XOR problem, for the 3-5-1 the network has twenty connection weights and six biases. Table 2 shows the CPU time, number of epochs, the MSE standard deviation (SD) and accuracy for CSRNN, CSBP, and CSLM algorithms. From Table 2, it is clearly visible that CSRNN converged with a superior 0.82 CPU cycles, 8 epochs and an MSE of 0 with 100 percent accuracy while CSBP and CSLM follows behind. Figure 2 illustrates the performance of the proposed algorithms.

Table 2. CPU Time, Epochs, MSE, Accuracy, Standard Deviation (SD) for 3-5-1 ANN Architecture

Algorithm	CSRNN	CSBP	CSLM
CPU Time	0.82	149.53	80.36
Epochs	8	938	671
MSE	0	5.4E-04	7.5E-07
SD	0	0.00134	3.14E-06
Accuracy (%)	100	98.7351	99.99

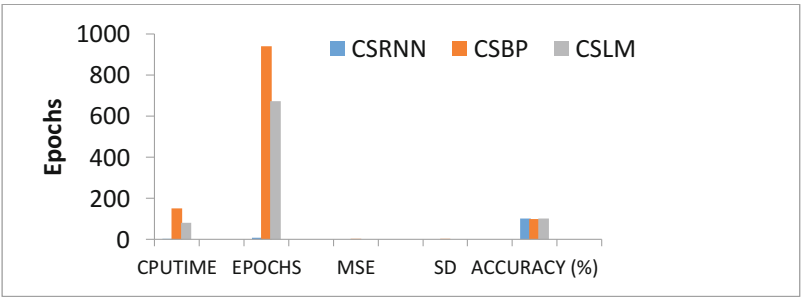

Fig. 2. Performance Comparison of CSRNN, CSLM, and CSBP algorithms on the basis of CPU Time, Epochs, MSE, Accuracy, Standard Deviation (SD) for 3-5-1 ANN Architecture

3.4 4-Bit OR Dataset

The third dataset is based on the logical operator OR which indicates whether an operand is true or false. If one of the operand has a nonzero value the result has a value equal to 1, otherwise it has a 0 value. The network architecture used here is 4-5-1 in which the network has twenty five connection weights and six biases. Table 3, illustrates the CPU time, epochs, and MSE performance and accuracy of the used algorithms, such as CSRNN, CSBP, and CSLM algorithms respectively. Figure 3, shows the graphical representation for the 4-5-1 network architecture of CSRNN, CSBP, and CSLM algorithms. In Figure 3, we can see that the hybrid Cuckoo Search algorithms achieved 0 MSE with 100 percent accuracy. The simulation results show that the CSRNN has better performance than others algorithms in terms of epochs, and CPU time.

Table 3. CPU Time, Epochs, MSE, Accuracy, Standard Deviation (SD) for 4-5-1 ANN Architecture

Algorithm	CSRNN	CSBP	CSLM
CPU Time	1.83	8.48	6.16
Epochs	13	51	55
MSE	0	0	0
SD	0	0	0
Accuracy (%)	100	100	100

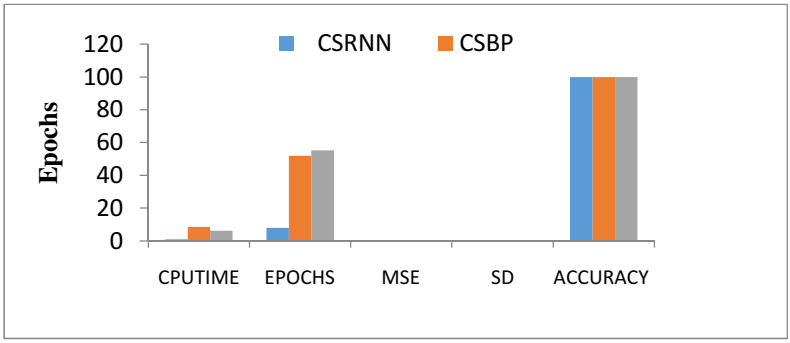

Fig. 3. Performance Comparison of CSRNN, CSLM, and CSBP algorithms on the basis of CPU Time, Epochs, MSE, Accuracy, Standard Deviation (SD) for 4-5-1 ANN Architecture

4 Conclusions

Nature inspired meta-heuristic algorithms provide derivative free solution to optimize complex problems. A new meta-heuristic search algorithm, called Cuckoo Search (CS) is an optimization algorithm developed by Xin-She Yang [32]. In this paper, CS is incorporated with Back propagation Neural Network (BPNN), Recurrent Neural Network (RNN), and Levenberg Marquardt back propagation (LMBP) algorithms to achieve faster convergence rate and to avoid local minima problem. The performances of the proposed Cuckoo Search Back propagation (CSBP), Cuckoo Search Levenberg Marquardt (CSLM) and Cuckoo Search Recurrent Neural Network (CSRNN) algorithms are verified by means of simulation on three datasets such as 2-bit, 3-bit XOR and 4-bit OR. The simulation results show that the proposed CSRNN algorithm is far better than the CSBP and CSLM algorithms in terms of Mean Squared Error (MSE), Convergence rate and accuracy.

Acknowledgments. This work is supported by University of Malaya High Impact Research Grant no vote UM.C/625/HIR/MOHE/SC/13/2 from Ministry of Higher Education Malaysia.

References

1. Radhika, Y., Shashi, M.: Atmospheric Temperature Prediction using Support Vector Machines. Int. J. of Computer Theory and Engineering 1(1), 1793–8201 (2009)
2. Akcayol, M.A., Cinar, C.: Artificial neural network based modeling of heated catalytic converter performance. J. Applied thermal Engineering 25, 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. Int. J. of Computer Science 8(6-2), 1694–1814 (2011)
4. Kosko, B.: Neural Network and Fuzzy Systm, 1st edn. 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. J. Neural Networks 16(3-4), 335–348 (2003)
6. Coppin, B.: Artificial Intelligence Illuminated, USA. Jones and Bartlet Illuminated Series, ch. 11, pp. 291–324 (2004)
7. Basheer, I.A., Hajmeer, M.: Artificial neural networks: fundamentals, computing, design, and application. J. of Microbiological Methods 43(1), 3–31 (2000)
8. Zheng, H., Meng, W., Gong, B.: Neural Network and its Application on Machine fault Diagnosis. In: ICSYSE, pp. 576–579 (1992)
9. Rehman, M.Z., Nawi, N.M.: Improving the Accuracy of Gradient Descent Back Propagation Algorithm (GDAM) on Classification Problems. Int. J. of New Computer Architectures and their Applications (IJNCAA) 1(4), 838–847 (2012)
10. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning Representations by Back–Propagating Errors. J. Nature 323, 533–536 (1986)
11. Lahmiri, S.: A comparative study of backpropagation algorithms in financial prediction. Int. J. of Computer Science, Engineering and Applications (IJCSEA) 1(4) (2011)
12. Nawi, M.N., Ransing, R.S., AbdulHamid, N.: BPGD-AG: A New Improvement of Back-Propagation Neural Network Learning Algorithms with Adaptive Gain. J. of Science and Technology 2(2) (2011)
13. Wam, A., Esm, S., Esa, A.: Modified Back Propagation Algorithm for Learning Artificial Neural Networks. In: 8th NRSC, pp. 345–352 (2001)
14. Wen, J., Zhao, J.L., Luo, S.W., Han, Z.: The Improvements of BP Neural Network Learning Algorithm. In: 5th WCCC-ICSP, vol. 3, pp. 1647–1649 (2000)
15. Lahmiri, S.: Wavelet transform, neural networks and the prediction of s & p price index: a comparative paper of back propagation numerical algorithms. J. Business Intelligence 5(2) (2012)
16. Nawi, N.M., Ransing, R.S., Salleh, M.N.M., Ghazali, R., Hamid, N.A.: An improved back propagation neural network algorithm on classification problems. In: Zhang, Y., Cuzzocrea, A., Ma, J., Chung, K.-i., Arslan, T., Song, X. (eds.) DTA and BSBT 2010. CCIS, vol. 118, pp. 177–188. Springer, Heidelberg (2010)
17. Gupta, J.N.D., Sexton, R.S.: Comparing back propagation with a genetic algorithm for neural network training. J. International Journal of Management Science 27, 679–684 (1999)
18. Rehman, M.Z., Nawi, N.M.: Studying the Effect of adaptive momentum in improving the accuracy of gradient descent back propagation algorithm on classification problems. J. Int. Journal of Modern Physics: Conference Series 9, 432–439 (2012)
19. Yam, J.Y.F., Chow, T.W.S.: Extended least squares based algorithm for training feed forward networks. J. IEEE Transactions on Neural Networks 8, 806–810 (1997)
20. Yam, J.Y.F., Chow, T.W.S.: A weight initialization method for improving training speed in feed forward neural networks. J. Neurocomputing 30, 219–232 (2000)
21. Yam, J.Y.F., Chow, T.W.S.: Feed forward networks training speed enhancement by optimal initialization of the synaptic coefficients. J. IEEE Transactions on Neural Networks 12, 430–434 (2001)
22. Kwok, T.Y., Yeung, D.Y.: Objective functions for training new hidden units in constructive neural networks. J. IEEE Transactions on Neural Networks 8, 1131–1147 (1997)
23. Zhang, J., Lok, T., Lyu, M.: A hybrid particle swarm optimization back propagation algorithm for feed forward neural network training. J. Applied Mathematics and Computation 185, 1026–1037 (2007)

24. Shah, H., Ghazali, R., Nawi, N.M., Deris, M.M.: Global hybrid ant bee colony algorithm for training artificial neural networks. In: Murgante, B., Gervasi, O., Misra, S., Nedjah, N., Rocha, A.M.A.C., Taniar, D., Apduhan, B.O. (eds.) ICCSA 2012, Part I. LNCS, vol. 7333, pp. 87–100. Springer, Heidelberg (2012)
25. Shah, H., Ghazali, R., Nawi, N.M.: Hybrid ant bee colony algorithm for volcano temperature prediction. In: Chowdhry, B.S., Shaikh, F.K., Hussain, D.M.A., Uqaili, M.A. (eds.) IMTIC 2012. CCIS, vol. 281, pp. 453–465. Springer, Heidelberg (2012)
26. Karaboga, D.: Artificial bee colony algorithm. J. Scholarpedia 5(3), 6915 (2010)
27. Yao, X.: Evolutionary artificial neural networks. J. International Journal of Neural Systems 4(3), 203–222 (1993)
28. Mendes, R., Cortez, P., Rocha, M., Neves, J.: Particle swarm for feedforward neural network training. In: Proceedings of the International Joint Conference on Neural Networks, vol. 2, pp. 1895–1899 (2002)
29. Lonen, I., Kamarainen, I.J., Lampinen, J.I.: Differential Evolution Training Algorithm for Feed-Forward Neural Networks. J. Neural Processing Letters 17(1), 93–105 (2003)
30. Liu, Y.-P., Wu, M.-G., Qian, J.-X.: Evolving Neural Networks Using the Hybrid of Ant Colony Optimization and BP Algorithms. In: Wang, J., Yi, Z., Żurada, J.M., Lu, B.-L., Yin, H. (eds.) ISNN 2006. LNCS, vol. 3971, pp. 714–722. Springer, Heidelberg (2006)
31. Khan, A.U., Bandopadhyaya, T.K., Sharma, S.: Comparisons of Stock Rates Prediction Accuracy using Different Technical Indicators with Back propagation Neural Network and Genetic Algorithm Based Back propagation Neural Network. In: The First International Conference on Emerging Trends in Engineering and Technology, pp. 575–580 (2008)
32. Yang, X.S., Deb, S.: Cuckoo search via Lévy flights. In: World Congress on Nature & Biologically Inspired Computing, India, pp. 210–214 (2009)
33. Yang, X.S., Deb, S.: Engineering Optimisation by Cuckoo Search. J. of Mathematical Modelling and Numerical Optimisation 1(4), 330–343 (2010)
34. Tuba, M., Subotic, M., Stanarevic, N.: Modified cuckoo search algorithm for unconstrainedoptimization problems. In: The European Computing Conference, pp. 263–268 (2011)
35. Tuba, M., Subotic, M., Stanarevic, N.: Performance of a Modified Cuckoo Search Algorithm for Unconstrained Optimization Problems. J. Faculty of Computer Science 11(2), 62–74 (2012)
36. Chaowanawate, K., Heednacram, A.: 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)
37. Pavlyukevich, I.: Levy flights, non-local search and simulated annealing. J. of Computational Physics 226(2), 1830–1844 (2007)
38. Walton, S., Hassan, O., Morgan, K., Brown, M.: Modified cuckoo search: A new gradient free optimisation algorithm. J. Chaos, Solitons& Fractals 44(9), 710–718 (2011)
39. Hagan, M.T., Menhaj, M.B.: Training Feedforward Networks with the Marquardt Algorithm. J. IEEE Transactions on Neural Networks 5(6), 989–993 (1994)
40. Nawi, N.M., Khan, A., Rehman, M.Z.: A New Cuckoo Search based Levenberg-Marquardt (CSLM) Algorithm. In: Murgante, B., Misra, S., Carlini, M., Torre, C.M., Nguyen, H.-Q., Taniar, D., Apduhan, B.O., Gervasi, O. (eds.) ICCSA 2013, Part I. LNCS, vol. 7971, pp. 438–451. Springer, Heidelberg (2013)
41. Nawi, N.M., Khan, A., Rehman, M.Z.: A New Back-propagation Neural Network optimized with Cuckoo Search Algorithm. In: Murgante, B., Misra, S., Carlini, M., Torre, C.M., Nguyen, H.-Q., Taniar, D., Apduhan, B.O., Gervasi, O. (eds.) ICCSA 2013, Part I. LNCS, vol. 7971, pp. 413–426. Springer, Heidelberg (2013)
42. Nawi, N.M., Khan, A., Rehman, M.Z.: A New Optimized Cuckoo Search Recurrent Neural Network (CSRNN) Algorithm. In: Sakim, H.A.M., Mustaffa, M.T. (eds.) ROVISP. LNEE, vol. 291, pp. 335–341. Springer, Heidelberg (2013)