Available online at www.sciencedirect.com

SciVerse ScienceDirect

Physics Procedia 25 (2012) 536 – 542

Physics Procedia

2012 International Conference on Solid State Devices and Materials Science

The Application of Imperialist Competitive Algorithm based on Chaos Theory in Perceptron Neural Network

Xiuping Zhang

Department of Computer Science
Tonghua Normal University,
Tonghua, 134002 China

Abstract

In this paper, the weights of a Neural Network using Chaotic Imperialist Competitive Algorithm are updated. A three-layered Perseptron Neural Network applied for prediction of the maximum worth of the stocks changed in TEHRAN's bourse market. We trained this neural network with CICA, ICA, PSO and GA algorithms and compared the results with each other. The consideration of the results showed that the training and test error of the network trained by the CICA algorithm has been reduced in comparison to the other three methods.

Keywords: Imperialist Competitive Algorithm; Perceptron Neural Network; chaotic

1. Introduction

The Artificial Neural Networks are one of the modern computational methods proposed to solve increasingly complex problems so far [1,2]. Neural Networks have the ability to perform tasks such as pattern recognition, classification, optimization, function approximation and control duties. A Neural Network is characterized by its pattern of connections between the neurons (called its architecture), its method of determining the weights on the connections (called its training, or learning, algorithm), and its activation Function. Training is accomplished by presenting a sequence of training vectors, or patterns, each with an associated target output vector. The weights are then adjusted according to a learning algorithm. The optimization method used to determine weight adjustments has a large influence on the performance of NNs. While gradient descent is a very popular optimization method, it is plagued by slow convergence and susceptibility to local minima. Therefore, other approaches to improve NN training introduced. These methods include global optimization algorithms, such as Simulated Annealing [3], Genetic Algorithms [4,5,6], Particle Swarm Optimization algorithms [7,8,9] and other Evolutionary Algorithms[10]. Recently, a new Evolutionary Algorithm has been proposed by Atashpaz-Gargari and lucas [11], in 2007 which has inspired from a socio-political phenomenon, called Imperialist Competitive Algorithm. In this algorithm, the colonies move towards the imperialists by a certain angle. In this paper, a multi-layered perception Neural Network was trained using Chaotic Imperialist Competitive Algorithm

1875-3892 © 2012 Published by Elsevier B.V. Selection and/or peer-review under responsibility of Garry Lee
doi:10.1016/j.phpro.2012.03.123

(CICA). Here, the Neural Network trained by CICA, ICA, PSO and GA algorithms used for prediction of the maximum worth of the stocks changed in TEHRAN's bourse market [12] and compared their results with each other. The consideration of results indicated that the training and test error of the network which trained by CICA algorithm has been reduced in comparison to the other methods. The rest of this paper is organized as follows. In section two, ICA algorithm and chaos theory will be introduced. The proposed algorithm for training the Neural Network will be presented in section three. Section four, is devoted to the empirical results of Neural Network training with proposed algorithm and its compression with the results obtained by ICA, PSO [13] and GA training algorithms. The last section concludes the paper.

2. Introduction of imperialist competitive algorithms (ica)

In this section, we introduce ICA algorithm and chaos theory.

2.1 Imperialist Competitive Algorithm (ICA)

Imperialist Competitive Algorithm (ICA) is a new evolutionary algorithm in the Evolutionary Computation field based on the human's socio-political evolution. The algorithm starts with N initial countries and the best of them (countries with minimum cost) chosen as the imperialists. The remaining countries are colonies that each belong to an empire. The initial colonies belong to imperialists in convenience with their powers. The imperialist countries absorb the colonies towards themselves using the absorption policy. The absorption policy shown in Fig.1, makes the main core of this algorithm and causes the countries move towards to their minimum optima. The imperialists absorb these colonies towards themselves with respect to their power that described in (1). The total power of each imperialist is determined by the power of its both parts, the empire power plus percents of its average colonies power.

$$TC_n = \cos(imperfalist_n) + \xi mean:\{\cos tCalantes\,of \quad empire_n)\}\} \tag{1}$$

Where TC_n is the total cost of the nth empire and ξ is a positive number which is considered to be less than one.

$$x \sim \bigcup(0, \beta \times d)\} \tag{2}$$

In the absorption policy, the colony moves towards the imperialist by x unit. The direction of movement is the vector from colony to imperialist, as shown in Fig. 1, in this figure, the distance between the imperialist and colony shown by d and x is a random variable with uniform distribution. Where β is greater than 1 and is near to 2. So, a proper choice can be $\beta = 2$. In our implementation is $\pi/4(Rad)$ respectively.

$$\beta \sim \bigcup(-r, r) \tag{3}$$

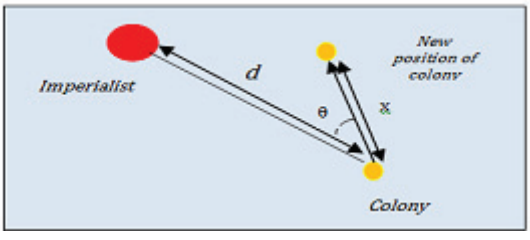

Figure 1. Moving colonies toward their imperialist.

In ICA algorithm, to search different points around the imperialist, a random amount of deviation is added to the direction of colony movement towards the imperialist. In Fig. 1, this deflection angle is shown as θ , which is chosen randomly and with an uniform distribution. While moving toward the imperialist countries, a colony may reach to a better position, so the colony position changes according to position of the imperialist. In this algorithm, the imperialistic competition has an important role. During the imperialistic competition, the weak empires will lose their power and their colonies. after a while all the empires except the most powerful one will collapse and all the colonies will be under the control of this unique empire.

2.2 Chaos Theory

In the chaos theory, a behavior between rigid regularity and randomness based on pure chance is called a chaotic system, or chaos. Chaos appears to be stochastic but it occurs in a deterministic non-linear system under deterministic conditions [14]. Chaotic map is very important for optimization problem. Moreover, it has a very sensitive dependence upon its initial condition and parameter. A chaotic map is a discrete-time dynamical system. Since the ICA algorithm suffers from being trapped at local optima, chaotic local search has been introduced to overcome the local optima and speed up the convergence.

3. Chaotic Imperialist Competitive Algorithm

In this paper, a new Imperialist Competitive Algorithm using the chaos theory is proposed. The primary ICA algorithm uses a local search mechanism as like as many evolutionary algorithms. Therefore, the primary ICA may fall into local minimum trap during the search process and it is possible to get far from the global optimum. To solve this problem we increased the exploration ability of the ICA algorithm, using a chaotic behavior in the colony movement towards the imperialist's position. So it is intended to improve the global convergence of the ICA and to prevent it to stick on a local solution.

3.1 Definition of chaotic angle in the movement of colonies towards the imperialist

In this paper, to enhance the global exploration capability, the chaotic maps are incorporated into ICA to enhance the ability of escaping from a local optimum. The angle of movement changed in a chaotic way during the search process. Adding this chaotic behavior in the imperialist algorithm absorption policy we make the conditions proper for the algorithm to escape from local peaks. Chaos variables are usually generated by the some well-known chaotic maps [15,16]. Table 1, shows the mentioned chaotic maps for adjusting θ parameter (Angle of colonies movement towards the imperialist's position) in the proposed algorithm.

Where, α is a control parameter. The θ is a chaotic variable in k^{th} iteration which belongs to interval of (0,1). During the search process, no value of θ is repeated.

TABLE I. Chaotic Maps.

CM1	$\theta_{n+1} = a\theta_n(1-\theta_n)$
CM2	$\theta_{n+1} = a\theta^2{}_n \sin(\pi\theta_n)$
CM3	$\theta_{n+1} = \theta_n + n - (a/2\pi)\sin(2\pi\theta_n) \bmod(1)$
CM4	$\theta_{n+1} = \begin{cases} 0, \theta_n = 0 \\ 1/X_n \bmod(1), \theta_n \in (0,1) \end{cases}$

The ICA algorithm performance mainly depends on its parameters, and it often lead to be trapped in local optimum. Larger amounts of the θ, lead to a global exploration, while smaller values of the θ, redound fine-tuning of the current search area. Thus, proper control of θ is very important to find the optimum solution accurately. Therefore, we controlled the θ parameter using the chaotic maps.

4. Neural Network Traning Using the Chaotic Imperialist Competitive Algorithm

So far, many different Evolutionary Algorithms have been used for training the Neural Networks. In this paper, we used the Chaotic Imperialist Competitive Algorithm for updating the weights in the training phase. In the following, the proposed Chaotic Imperialist Competitive Algorithm described briefly.

4.1 Updating the Neural Network Weights using the CICA

In this paper, the Chaotic Imperialist Competitive Algorithm is used for updating neural network weights. We apply a three-layered Perceptron Neural Network including an input layer, a hidden and an output layer. The number of input nodes set to 7, hidden nodes to 5, and one node in the output layer. We considered the weights of network training phase as the variables of an optimization problem. The Mean Square Error (MSE) used as a cost function in CICA algorithm. The goal in proposed algorithm is minimizing this cost function. The Neural Network weights initialized randomly and served as initial population for the CICA algorithm to optimize the cost function. In following, we analyze the results of proposed methods for prediction of the maximum worth of the stocks changed in TEHRAN's bourse market.

5. Analysis and Consideration of Empirical Results

In this paper, we evaluate the proposed method performance in comparison to ICA, PSO and GA algorithms for training a three layered Perceptron Neural Network. We applied this trained network on the data of TEHRAN's bourse market. The inputs of this network are the volume of changed stocks, the last price, the least price and the most price in different times. The output of this network is the approximation of the most price of the changed stocks in TEHRAN's bourse market. In these simulations, we used a three-layered Perseptron Neural Network containing an input layer with 7 nodes, a hidden layer with 5 nodes and an output layer with one node. In these simulations, the parameters β , a and b respectively are set to 2,0.5 and 0.2. The number of imperialists and the colonies are considered 8 and 80. In the PSO algorithm, the parameters α_1 and α_2 are fix to 1.5 and the number of the particle is 80. Determining this amount for c1 and c2 we have given equal chance to social and cognition components take part in search process. In GA the population size is 80, the mutation and crossover rate are respectively set to 0.01 and 0.5. The results of these experiments are presented in Table 2 and 3. The results of these simulations are seen in following charts.

As we can see in the below charts, the proposed algorithm has better results than the ICA, PSO and GA algorithms. The dataset include of 1155 instances. Using Holdout method (The holdout method splits the data into two mutually exclusive sets, sometimes referred to as the training and test sets) we apply 80% of instance data for training the Neural Network and the remaining 20 % for testing. The neural network trained by CICA, ICA, PSO and GA algorithms and the results compared with each other. Result in Table 2 shows that the training and test error of network trained by CICA algorithm is less than the other methods.

The comparative result of the Neural Network trained by CICA, ICA, PSO and GA algorithm is shown in Table 2. From the comparative results, it observed that the CICA proposed algorithm both in training and test phase has the least error rate. The run time of the proposed algorithm is less than PSO and GA but a little more than primary ICA.

TABLE II. Compare Results

	Train Error	Test Error	Train correlation	Test correlation	Time of Training
GA	0.0031	0.0063	0.9909	0.9784	1151.88s
POS	0.0016	0.0016	0.9949	0.9943	1363.97s
ICA	0.0021	0.0014	0.9936	0.9973	1165.57s
CICA	0.7596×10^{-4}	8.4305×10^{-4}	0.9970	0.9792	1173.25s

As it can be seen in Fig. 3, although the CICA with first chaotic map (CM1) obtained very competitive results in comparison with other algorithms, they come with the cost of additional computation time. However, for many applications, the training time is less important than generalization.

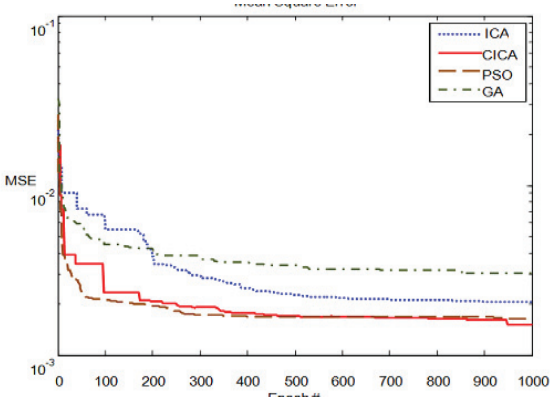

Figure 2. Comparing the mean square error for CICA with CM1.

In Fig. 4, the results of training phase for CICA with second chaotic map (CM2) show that in comparison with the other three algorithms this algorithm has better performance.

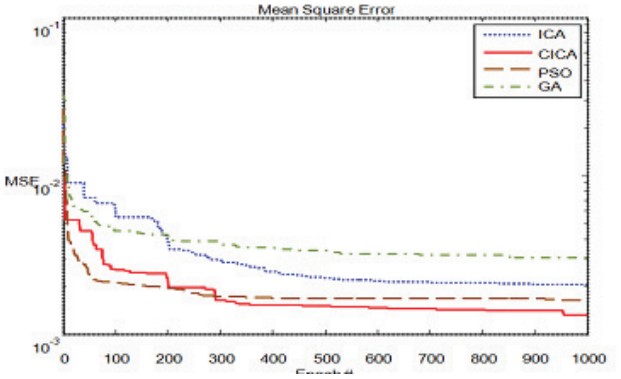

Figure 3. Comparing mean square error for CICA with CM2.

In Fig5, the comparison of Mean Square Error (MSE) of Neural Network trained by CICA with third chaotic map (CM3), ICA, PSO and GA indicated that the proposed algorithm trained very well rather than the other algorithms.

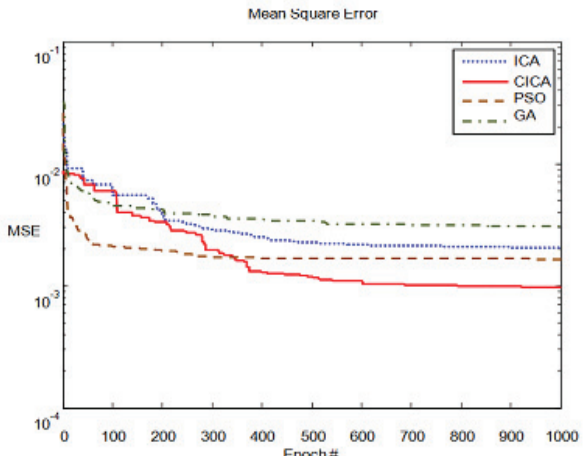

Figure 4. Comparing mean square error for CICA with CM3.

In Fig 6, the CICA algorithm with fourth chaotic map (CM4) has better performance than ICA, PSO and GA algorithms.

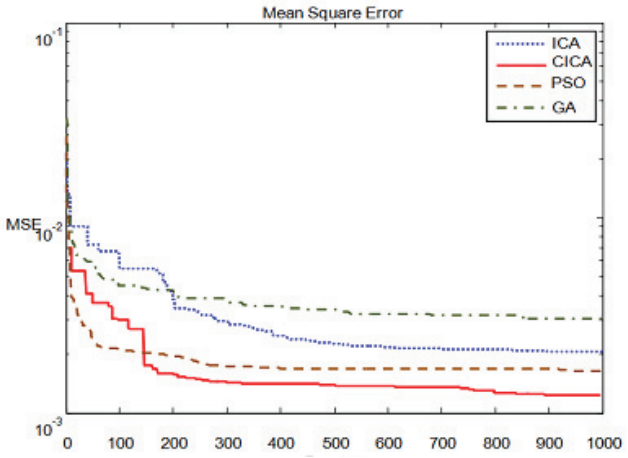

Figure 5. Comparing mean square error for CICA with CM4.

Table 2, shows that the CICA proposed algorithm with third Chaotic Map, has better performance in training and test phase. Hence, it is the best choice to use for training Neural Network.

TABLE III. Compare Results

	Train Error	**Test Error**
CICA-CM1	2.8427×10^{-1}	0.0012
CICA-CM2	2.0353×10^{-1}	0.0023
CICA-CM3	9.7596×10^{-1}	8.4305×10^{-1}
CICA-CM4	1.9475×10^{-1}	9.5637×10^{-1}

6. Conclusion

The run time of training algorithm by proposed CICA has been reduced in comparison to PSO and GA algorithms. In future, some other models can suggested to increase the velocity of convergence of the CICA training algorithm and also different Data Sets can be applied to verify the gained result.

References

[1] J.X. Xie, C.T. Cheng, K.W. Chau, Y.Z. Pei, "A hybrid adaptive time-delay neural network model for multi-step-ahead prediction of sunspot activity", International Journal of Environment and Pollution 28 (3–4) , pp.364–381, 2006.

[2] K.W. Chau, "Reliability and performance-based design by artificial Neural network", Advances in Engineering Software 38 (3) pp.145149,2007.

[3] B.E. Rosen and J.M. Goodwin, "Optimizing Neural Networks Using Very Fast Simulated Annealing", Neural, Parallel & Scientific Computations, pp.383–392, 1997.

[4] C.L. Wu, K.W. Chau, "A flood forecasting neural network model with genetic algorithm", International Journal of Environment and Pollution 28(3–4) pp. 261–273 ,(2006).

[5] N. Muttil, K.W. Chau, "Neural network and genetic programming for modelling coastal algal blooms", International Journal of Environment and Pollution 28 (3–4) pp. 223–238, 2006.

[6] P.A. Castillo, J. Carpio, J.J. Merelo, A. Prieto, V. Rivas, G. Romero, G-Prop, "global optimization of multilayer perceptrons using Gas", Neurocomputing 35 , pp.149–163, 2000.

[7] J. Kennedy and R.C. Eberhart, "Particle swarm optimization", in: Proceedings of IEEE International Conference on Neural Networks, Piscataway: IEEE, pp. 1942–1948, 1995.

[8] K. Lei, Y. Qiu and Y. He, "A New Adaptive Well-Chosen Inertia Weight Strategy to Automatically Harmonize Global and Local Search Ability in Particle Swarm Optimization", ISScAA, 2006.

[9] Y. Da, X.R. Ge, "An improved PSO-based ANN with simulated annealing technique", Neurocomput. Lett. 63 pp. 527–533, 2005.

[10] H. Sarimveis and A. Nikolakopoulos, "A Line Up Evolutionary Algorithm for Solving Nonlinear Constrained Optimization Problems", Computers & Operations Research, 32(6):pp.1499–1514, 2005.

[11] E. Atashpaz-Gargari and C. Lucas, "Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition", IEEE Congress on Evolutionary Computation (CEC 2007).pp 4661-4667, 2007.

[12] http://www.IRBourse.com, The dataset for training the Neural Network.

[13] P. J. Angeline, "Evolving Fractal Movie". Proc. 1st An. Conf. on Genetic Programming, MIT Press, Cambridge, MA, pp. 503–511.

[14] M. Suneel, "Chaotic sequences for secure CDMA", Ramanujan Institute for Advanced Study in Mathematics, pp. 1–4, 2006.

[15] C. Bing-rui and F. Xia-ting, "Self-adapting Chaos-genetic Hybrid Algorithm with Mixed congruential Method", Forth International Conference, pp. 674-677,2008.

[16] M. Gao, J. Xu, J. Tian and H. Wu, "Path Planning for Mobile Robot based on Chaos Genetic Algorithm", Forth International Conference, pp. 409-413,2008.