IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 4, June 2014.
www.ijiset.com
ISSN 2348 – 7968

Artificial Neural Network Training by using Regrouping Particle Swarm Optimization

Lakshminarayana Pothamsetty[1], Shishir Ranjan[2], Mukesh Kumar kirar[3], Ganga Agnihotri[4]

[1,2,3,4]Department Of Electrical Engineering, MANIT, Bhopal, India
E-mail: lakshminarayanapothamsetty@gmail.com[1], er.shishir.ranjan@gmail.com[2], mukeshkirar@rediffmail.com[3], ganga1949@gmail.com[4]

Abstract

This paper presents a novel methodology for Artificial Neural Network training by using Re-grouping Particle Swarm Optimization (Regrouping-PSO). The Particle Swarm Optimization (PSO) can be used to optimize a particular objective function. The basic PSO is suffered from the stagnation problem. In Regrouping-PSO this stagnation problem can be avoided by automatic regrouping the particles when stagnation is detected. In ANN training by using Regrouping-PSO the objective function is taken as the Mean Square Error (MSE) difference between the actual output and the obtained output. The variables of the objective function are the weights and biases of the neural network. The proposed method can be applied to both pattern recognition and classification problems.

Keywords: *ANN; PSO; Premature Convergence; Stagnation; Re-grouping-PSO*

1. Introduction

Artificial Neural Networks are inspired by the Human brain. A brain is a massively parallel distributed system made up of highly interconnected neural computing elements called as neurons. These neurons have the ability to learn and thereby acquire knowledge and make it available for use. The neurons are interconnected with the other neurons through weights and each neuron has a bias. Generally these weights and biases are adjusted in order to reduce the error between the target and the obtained output. This process of adjusting the weights and biases is known as training [1]. For this minimization of error difference between the target and the obtained output Particle Swarm Optimization is used [2]. PSO is basically an optimization algorithm, inspired by the social behavior of bird flock. In PSO stagnation is the main problem, which occurs after some iteration all the particles have prematurely converged to any particular region of the search space. This stagnation problem can be avoided by using Regrouping-PSO which automatically regroups all the particles around the global best when stagnation is detected [3]. The Re-grouping PSO is applied for three to four times in order to escape from the local well.

2. Artificial Neural Networks

The concepts of ANNs have been around since the 1950 and are biologically inspired from the view of human brain. The complex relationship between the input variables and output variables is established by using the ANN. There are different types of neural networks are present which mainly include are

1) Single layer feed forward neural network
2) Multi layer feed forward neural network
3) Recurrent neural network

The multi layer feed forward neural network is the most commonly used neural network.

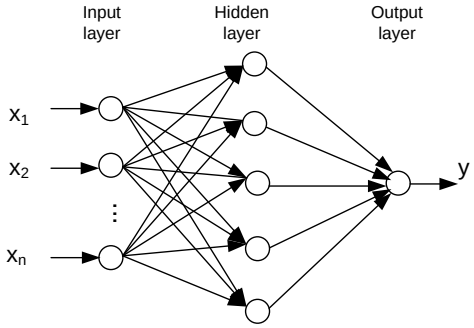

Fig.1. A multi layer neural network with one hidden layer

This neural network is also known as multilayer perceptron (MLP). The MLP consists of one or more hidden layers in addition to the input and output layers. Generally one hidden layer is sufficient [4, 5]. The neurons in each layer are interconnected with other neurons through weights and each neuron has a bias. Generally these weights and biases are adjusted up to the desired accuracy difference the target and the obtained output. This process of adjusting the weights and biases is known as training. The LMBP (Levenberg Marquadict Back Propagation) algorithm is the most commonly used algorithm for the ANN training. The main disadvantage of this method is that the solution will be trapped at local minimum. This disadvantage can be avoided in Regrouping-PSO training.

ISSN 2348 – 7968

3. Particle Swarm Optimization

PSO is used to optimize a particular objective function. That is it will maximize or minimize the objective function. PSO is a population search method. It is inspired from the social behaviour of bird flock [6, 7]. In PSO the concept of fitness is used. Initially we will take random solutions to the particular objective function. These initial solutions are called as particles. Then we will calculate the fitness for all of these particles. The value of the particle for which the objective function is minimum, is known as global best particle. Then all the particles move towards the global best particle with a velocity proportional to the distance between the each particle and global best particle.

In the next iteration the particles will update their position by adding the velocity to the previous particles. Then we will calculate the fitness for the updated particles. The local best particles are those particles which have minimum objective function among the previous particles and the updated particles. The minimum among the local best particles is taken as global best particle [8]. Then the particles move towards the global best particle with a velocity proportional to both distance between the present particle and previous best particle, and distance between the present particle and global best particle. This process is continued until true global minimum is found. The detailed procedure of PSO is given below

Step 1: Initialization of particles

We will randomly initialize the s particles within search space, where s represents the swarm size. Each particle represents a possible solution to the objective function.

Step 2: Velocity updating

Velocity for particle i, at iteration $t+1$ can be updated using velocity contained in previous iteration t, to be represented as

$$\vec{V}_{id}(t+1) = W * \vec{V}_{id}(t) + C_1 * \varphi_1 * (P_{best}^{id} - P_{id}(t)) + C_2 * \varphi_2 * (G_{best}^{id} - P_{id}(t))$$

Where i =1, 2, 3...n, n is the number of particles

d =1, 2, 3 … m, m is the number of input variables to be optimized

W = inertia weight factor varies linearly from $W_{\min}$ to $W_{\max}$, ranges between (0, 1)

C_1 = C_2 = cognitive and social acceleration factors respectively

φ_1 = φ_2 = uniformly distributed random numbers in the range of (0, 1)

Step 3: Position updating

Then the position of the each particle is evaluated as the sum of its previous position and the corresponding updated velocity obtained in the previous step can be represented as

$$\vec{P}_{id}(t+1) = \vec{P}_{id}(t) + \vec{V}_{id}(t+1)$$

Step 4: Memory updating

Update and G_{best}^{id} when the condition is met

$$P_{best}^{id} = P_{id} \quad \text{If} \quad f(P_{id}) > f(P_{best}^{id})$$

$$G_{best}^{id} = G_{id} \quad \text{If} \quad f(G_{id}) > f(G_{best}^{id})$$

These parameters move with an adaptable velocity within the search space and retain its own memory with the best position it ever reached.

Step 5: Termination checking

The algorithm repeats steps 2-4 until certain termination conditions are met, such as a predefined number of iterations or failure to make progress for a certain number of iterations.

2.1 ANN training by using PSO

The PSO can be used as a training algorithm for the neural networks. The number of dimensions of the PSO is taken as the total number of weights and biases of the neural network [9, 10]. The mean square error of the difference between the target and the output is taken as the objective function. The fitness of the i^{th} particle is expressed in terms of an output mean squared error of the neural network as follows

$$f(W_i) = \frac{1}{s}\sum_{l=1}^{o}[\{t_{kl} - p_{kl}(W_i)\}^2]$$

Where,

f is the fitness value, t_{kl} is the target output, p_{kl} is the predicted output, s is the number of training set samples, o is the number of output neurons.

4. Regrouping Particle Swarm Optimization (Regrouping-PSO)

The main problem with the basic PSO is the premature convergence. Premature convergence occurs when all particles move towards to a local minimizer rather than towards global minimizer and the moving of the particles towards global minimizer has stopped. Stagnation is due to premature convergence. When the premature convergence condition met, after certain number of iterations all the particles will move to a particular small region of the search space. And the particles will not come out from the small region of the search space. If this process of moving the particles is continued further then after some time all the particles will act similar to one particle and there is no progress towards

ISSN 2348 – 7968

finding the true global minimum. This stagnation problem can be avoided by using Regrouping-PSO. In Regrouping-PSO when the premature convergence detected the swarm will automatically regroup the particles around the global best and continue the progress of finding the true global minimum [11, 12]. For the illustration of the stagnation problem, consider a search algorithm of minimization of two-dimensional Rastigin function.

Rastrigin function

$$f(X) = An + \sum_{i=1}^{n}\left[x_i^2 - A\cos(2\pi x_i)\right]$$

Where A=10 and n is the number of dimensions of the problem

Here we have considered n=2, two dimensional problem. The Rastrigin function has true global minimum X=0. For this particular problem consider ten particles. The particles motion can be seen in the Fig.2. In Fig.2 we can see that the particles are flying from random initialization to local minimizer [2, 0]. Here stagnation occurs. This stagnation can be detected as follows.

4.1 Detection of premature convergence

In PSO all the particles are move towards the global best. If after certain number of iterations none of those particles will encounter a better solution, all the particles will continue moving towards the unchanged global best. This process of moving the particles towards the unchanged global best will continue until all the particles will occupy the same location in the search space. If all the particles occupy the same location in the search space then there is no progress towards finding the true global minimum [13, 14]. Therefore we will measure the distance between the each particle to other particles so that when the premature convergence condition met the swarm will regroup the particles around the global minimizer which is found up to so far and enable the continued progress towards the true global minimizer.

In each iteration k, we will calculate the maximum Euclidean distance $\delta(k)$ between each particle and the global best as follows

$$\delta(k) = \max_{i=\{1,..s\}} ||\vec{x}_i - \vec{g}(k)||$$

Where, ||. || denotes the Euclidean norm.

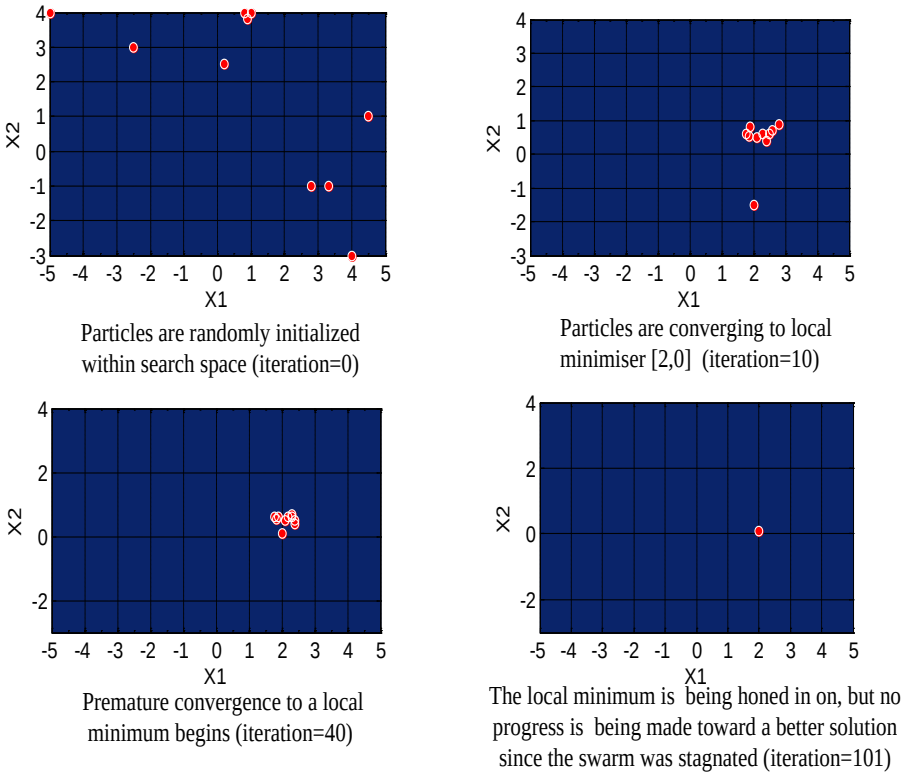

Fig.2 Swarm motion converged to local minimizer [2, 0]

ISSN 2348 – 7968

Let the diameter of the search space

$$diam(\Omega) = \left\| range(\Omega) \right\|$$

Where $range(\Omega)$ is the range of each dimension given by,

$$range(\Omega) = [range_1(\Omega), \ldots, range_n(\Omega)]$$

$$range_j(\Omega) = x_j^U - x_j^L$$

Where, x_j^U and x_j^L are the maximum and minimum values of the dimension j

After certain number of iterations when all the particles are prematurely converged, the swarm will regroup the particles if the normalized swarm radius satisfies the following condition.

$$\delta_{norm} = \frac{\delta(k)}{diam(\Omega)} < \varepsilon$$

Where ε is the stagnation threshold and is generally taken as 1.1*10^-4.

When all the particles are converged prematurely, the swarm is regrouped randomly in a new search space around the global best. Generally the regrouping factor ρ is taken as

$$\rho = \frac{6}{5\varepsilon}$$

When stagnation is detected, we need to regroup the particles around the global best. Here the range in which new particles are to be formed around the global best is different for each dimension and it is calculated as the minimum of the product of the regrouping factor with the maximum distance along dimension j of any particle from global best and the original range of the search space on dimension j [15].

$$range_j(\Omega^r) = \min\left(range_j(\Omega^0),\ \rho \max_{i=\{1,..s\}} \left| x_{i,j}^{r-1} - g_j^{r-1} \right| \right)$$

4.2 Swarm Regrouping

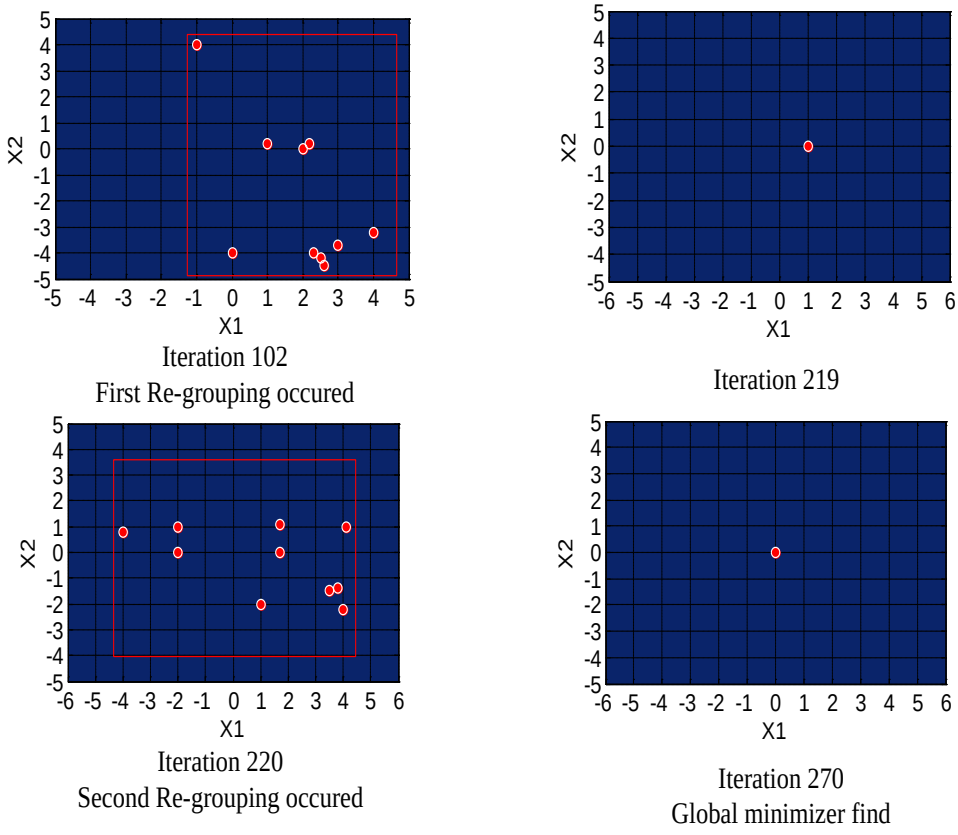

Fig.3 Swarm motion converged to global minimizer [0, 0]

ISSN 2348 – 7968

The swarm is then regrouped by re-initializing particles position as

$$\vec{x}_i = \vec{g}^{r-1} + \vec{r} * range(\Omega^r) - \frac{1}{2} range(\Omega^r)$$

Where,

$$range(\Omega^r) = [range_1(\Omega^r), range_2(\Omega^r), \dots, range_n(\Omega^r)]$$

Here we have used random vector $\vec{r}$ to randomly initialize the particles within new search space with respective lower and upper bounds as

$$x_j^{L,r} = g_j^{r-1} - \frac{1}{2} range_j(\Omega^r)$$
$$x_j^{U,r} = g_j^{r-1} + \frac{1}{2} range_j(\Omega^r)$$

Vector $\vec{g}^{r-1}$ is the global best at the last iteration of the previous regrouping and $\vec{x}_i^{r-1}$ is the position of particle i at the last iteration of the previous grouping. The maximum velocity is recalculated with each regrouping according to the new range per dimension as [16, 17].

$$v_j^{max,r} = \lambda * range_j(\Omega^r)$$

We can see from Fig.3 that the true global minimizer [0, 0] is found by applying the Regrouping-PSO. Similarly when applying Regrouping PSO for ANN training the best values of weights and biases for which the objective function is minimum can be obtained when compared with the weights and biases obtained by PSO. This Regrouping-PSO is used in the ANN training similar to the PSO, where as in the Regrouping –PSO when the premature convergence is detected the swarm will regroup all the particles around the global best. In Regrouping-PSO, four to five times we will regroup the swarm in order to further reduce the mean square error difference between the target and the obtained output, and to escape from the local wells.

5. Conclusions

The Regrouping-PSO can be used as the training algorithm for the neural networks. The objective function of the Regrouping-PSO is taken as the mean square error difference between the target and the obtained output. And the variables of the objective function are taken as the weights and biases of the neural network. In Regrouping-PSO four to five times we will regroup the swarm in order to further reduce the mean square error difference between the target and the obtained output, and to escape from the local wells.

References

[1] S. Rajasekharan and G.A. Vijayalakshmi Pai, “Neural networks, Fuzzy logic and Genetic Algorithms: Synthesis and Applications,” PHI Learning Private Limited, 15th edition, 2003.

[2] J. Upender, C.P. Gupta, G.K. Singh and G. Ramakrishna, “PSO and ANN-based fault classification for protective relaying,” IET Generation, Transmission and Distribution., Vol. 4, Issue no. 10, Jan 2010, pp. 1197–1212.

[3] G.I. Evers and M. Ben Ghalia, "Regrouping particle swarm optimization: A new global optimization algorithm with improved performance consistency across benchmarks," IEEE International Conference on Systems, Man and Cybernetics, Oct. 2009, pp.3901-3908.

[4] A.Karami, “Power system transient stability margin estimation using neural networks,” Electrical Power and Energy systems, vol.33, 2011, pp.983-991.

[5] Chi Zhou, Liang Gao, Haibing Gao and Chuanyong Peng, “Pattern classification and prediction of water quality by neural network with Particle Swarm Optimization,” proceedings of the 6th World Congress on Intelligent Control and Automation, Dalian, China, June 21-23.

[6] Azzam-ul-Aser, Syed Riaz ul Hassnaina and Affan Khanb, “Short term load forecasting using Particle Swarm Optimization based ANN approach,” Proceedings of International Joint Conference on Neural Networks, Orlando, Florida, USA, August 12-17, 2007.

[7] Hua-sheng Zhao, Long Jin and Xiao-yan Huang, “A Prediction of the monthly Precipitation model based on PSO-ANN and its applications,” Third International Joint Conference on Computational Science and Optimization, 2010.

[8] G. Evers, "An automatic regrouping mechanism to deal with stagnation in particle swarm optimization," M.S. Thesis, Electrical Engineering Department, The University of Texas-Pan American, Edinburg, TX, 2009.

ISSN 2348 – 7968

[9] H. Wang, "Opposition-based particle swarm algorithm with Cauchy mutation," in Proceedings of the IEEE Congress on Evolutionary Computation, 2007, pp. 4750-4756.

[10] M. Clerc and J. Kennedy, "The particle swarm explosion, stability, and convergence in multidimensional complex space," IEEE Transactions on Evolutionary computation, vol.6, 2002, pp. 58-73.

[11] B. Liu, "An improved particle swarm optimization combined with chaos," chaos, Solitons and Fractals, vol.25, 2005, pp. 1261-1271.

[12] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," Proc. IEEE International Conference on Neural Networks (Perth Australia), 1995, IEEE Service Center Piscataway, NJ, Aug. 2004, pp. 1942–1948.

[13] M. Todorovski and R. Dragoslav, "An initialization procedure in solving optimal power flow by genetic algorithm," IEEE Trans. on Power Systems, vol. 21, no. 2, May. 2006, pp. 480–487.

[14] Biruk Tessema and Gary G. Yen, "A Self Adaptive Penalty Function Based Algorithm for Constrained Optimization," IEEE Congress on Evolutionary Computation, CEC 2006, pp. 246–253.

[15] J.B. Park, K.S. Lee, J.R. Shin, and K.Y. Lee, "A particle swarm optimization for economic dispatch with non-smooth cost functions," IEEE Trans. on Power Systems, vol. 20, no. 1, Feb. 2005, pp. 34-42.

[16] Ioannis N. Kassabalidis and M. A. El-sharkawi, "Dynamic Security Border Identification Using Enhanced particle swarm optimization," IEEE Trans. on Power Systems, vol. 17, no. 3, Aug. 2002, pp. 723 - 729.

[17] G. Venter and J. Sobieski, "Particle Swarm Optimization," in proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Stuctures, Structual Dynamics, and Materials Conference, Denver, CO, 2012, pp.1-9.