A Novel Probabilistic-PSO Based Learning Algorithm for Optimization of Neural Networks for Benchmark Problems

SUDHIR G.AKOJWAR[1], PRAVIN R. KSHIRSAGAR[2]
[1]Department of Electronics and Telecommunication Engineering.
[1]Gondwana University
[1]Government College of Engineering,
[1]Chandrapur (M.S)-442403,
[1]INDIA
[1]sudhirakojwar@gmail.com
[2]Department of Electronics Engineering.
[2]Gondwana University
[2]Rajiv Gandhi College of Engineering, Research and Technology, Chandrapur (M.S)-442403,
[2]INDIA
[2]pravinrk88@yahoo.com

Abstract: - This paper approbates a modern and stentorian version of standard particle swarm optimization (PSO) for optimization of initial weights and biases for multi layer feed forward neural networks (MLFFNN) with back propagation (BP). The combination of probabilistic-PSO and MLFFNN sevenfold help in fast convergence of MLFFNN in assortment and sortilege to various benchmark problems by alienating the imperfection of backpropagation of being stuck at local minima or local maxima. The propane probabilistic-PSO differs from the standard PSO in velocity and position parameters. In velocity parameters only particle best value is make use of for cicerone the particle to gait towards the pursuit in the search space, while in standard PSO both particle best and global best values are considered for adjudging the new velocity of the particle. A new parameter introduced which called as the probability parameter (P0), which adjudges if the standard PSO is that instead of using same random number, different particles use different random numbers to soar in search space. The proposed method used to detect the initial weights and biases for MLFFNN with BP, once the optimum value for initial weights and biases estimated the MLFFNN then used for classification and sortilege of various neural network benchmark problems. The benchmarking databases for neural network contain various datasets from various different domains. All datasets represent sensible issue, which can call diagnosis tasks, and all the datasets consist of real world data. The results for accuracy of the proposed probabilistic-PSO method compared with existing methods.

Key-Words: - Probabilistic-PSO, multi layer feed forward neural networks (MLFFNN), back propagation (BP), local minima, local maxima, convergence, optimum weights and biases, benchmark problems.

1. Introduction

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr. Ebehart and Dr. Kennedy in 1995, embolden by social demeanour of bird hording or fish schooling. It has many similarities with evolutionary computations techniques like genetic algorithm (GA). However, PSO do not have evolution operators such as crossover and mutation. The standard PSO consists of two equations for velocity and position upgrade of each particle in the problem search space as shown below:

$$V_i^{(u+1)} = w*V_i^{(u)} + C_1*\text{rand}(\,)*(p_{best}i - P_i^{(u)}) + C_2*\text{rand}(\,)*(g_{best}i - P_i^{(u)}) \quad (1)$$

$$P_i^{(u+1)} = P_i^{(u)} + V_i^{(u+1)} \quad (2)$$

Where rand ()*(p_{best}i $-P_i$(u)) is called particle memory influence, rand()*(g_{best}i-P_i(u)) is swarm influence , V_i (u) is the velocity of i^{th} particle at iteration 'u'. $V_{min} \le V_i(u) \le V_{max}$. V_{max} determines the resolution, or fitness, with which regions searched between the present position and the target position. If V_{max} is too high, particles may fly past good solutions. If V_{min} is too small, particles may not explore sufficiently beyond local solutions. In many experiences with PSO, V_{max} often set at 10-20% of the dynamic range on each dimension. The constants $C_1$and C_2 pull each particle towards p_{best} and g_{best} positions. Low values allow particles to roam far from the target regions before tugged back. On the other hand, high values result in abrupt

movement towards, or past, target regions. The acceleration constants C_1 and C_2 are often set to be 2.0 according to experiences Suitable selection of inertia weight 'ω' provides a balance between global and local explorations, thus requiring less iteration on average to find a sufficiently optimal solution.
Conventional PSO developed in 1995; yet several variants of PSO had been developing. The basic algorithm involves two steps to update the particle position to move particles in the search space. This model was able to solve single modal issues efficiently. Thereafter inertial weight parameter introduced as one as important variant to poise the local and global search of the particles. Many researches had been carrying out disquisition to ameliorate the performance in various ways. The following are the parameters considered by various researchers for performance improvement of the conventional PSO.

1.1 Inertial weight (w):

The inertial weight was inclusive in, to rein the fortuitously change in velocity of the particles subsequently at the end of each iteration [2]. Earlier the inertial weight considered constant, but some researchers substantiate that the value of 'w' must change iteration. It is experimentally found that value of inertia weight between 0 and 1 provides excellent results and it should be decreased as iterations increase [3]. Fuzzy logic used to generate value of 'w' for corresponding iteration [4]. 'W' was set to 0 and used only when re-initialization was required [5].

1.2 Neighborhood:

The search space of each particle is fixing in standard PSO. Afterwards many researchers propounded that the search space should considered pursuance to the cost function. Large neighbourhood is more appropriate for simple problems or single dimension problems and a small neighbourhood is more accurate for complex or multimodal problems [7, 8]. A neighbourhood selected based on the fittest particle for the corresponding iteration [9]. The global best and local best uniformly combined to detect the neighbourhood in iteration [10].

1.3 Use of other evaluation techniques:

Many researchers have effectively utilized evolution algorithms such as ACO and GA along with PSO to ameliorate the performance of PSO [13, 14]. Crossover and mutation used to eliminate the least fitness value. The possibility of being stuck at local minima or local maxima is detracting by using GA along with PSO [15].

1.4 Constants C_1 and C_2:

In standard PSO, the constants were keeping as a fixed value. However, many researchers incorporated different formulae to evaluate and change the values of these constants with iteration. Clerc and Kennedy ameliorate convergence velocity, which guaranteed convergence by varying the values of C_1 and C_2 [6].

2. Proposed probabilistic - PSO algorithm

The N-dimension equation for the velocity provides larger search space due to independent or different values of rand () numbers for each dimension. The minimum and the maximum value of velocity has limited by $\pm V_{max}d$.
In particle swarm algorithm, changes in the velocity are stochastic and one undesirable effect of this is that the uncontrolled particle's trajectory expanded into wider cycles in the problems space. This can result in swarm explosion and thus divergence. To address this drawback of the original PSO model, a threshold called V_{max} on the particles velocity introduced. The motivation was to dampen the oscillation of the particles by restricting them to a maximum allowed value. The threshold as applied to the particle velocity.
In this algorithm, only one swarm population is utilizing to detect the initial weights and biases for a MLFFNN. Here each swarm particle is update by comparing with any two randomly selected neighbourhood particles. If the fitness of neighbourhood particle is greater than particle's own fitness, then the particle updated and if particle fitness is greater than the two neighbourhood particles then the neighbour's parameters are discarded and the particle uses its own value for update.
The modified formula for velocity up-gradation is

$$V_i^{(u+1)} = w*V_i^{(u)} + C* rand_i()^u * (P_{best}i^u - P_i^u) \quad ....(3)$$

The above equation for velocity update differs from the standard PSO equation as elaborated below:
In standard PSO equation two constants C_1 and C_2 are used with a fixed value as $C_1 = C_2 = 2$. Here only one constant 'C' is used with a fixed value as C= 1.45
Here no global best term used to update the particle velocity. Particle velocity updated using its own

velocity and neighbourhood fitness.
The inertial weight is not constant as in standard PSO. It varied by the formula:

$$w = w_{max} - \left[\frac{Wmax - Wmin}{ITERmax}\right] * ITER \quad (4)$$

Where, ITER stands for number of iteration.
The particle learning strategy [19] is dependent on probability 'P_0'. The probability value P_0 takes a different value for each particle. For each dimension, a different random number generated. If the random number is greater than P_0, then the particle will learn from its own P_{best} value, otherwise it will learn from one of its randomly selected neighbour's P_{best} value.
The effectiveness of this P_0 is that, it can easily solve any multimodal problem without being stuck at any local minima. Thus, it increases the diversity of the PSO.
The expression for P_0 given by,

$$P_0 = 0.05 + 0.45 * \frac{e^{(\frac{10(k-1)}{Ps-1}-1)}}{e^{10}-1} \quad (5)$$

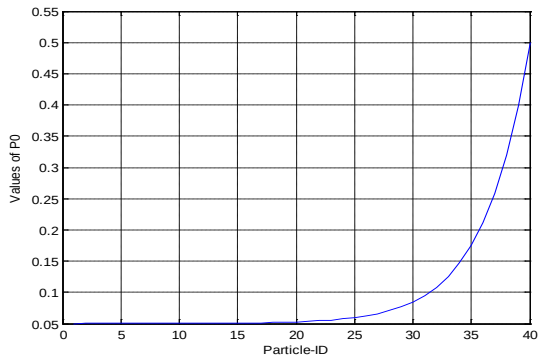

Fig.1 – Each particle's P_0 with population = 40

Where, Ps is the population size and k is the iteration number. The graph below shows values of P_0 assigned when population size is 40. Each particle in the population from 1-40 has a P_0 value ranging from 0.05-0.5.
The modified formula for position update is:

$$P_i^{(u+1)} = P_i^{(u)} + 0.8 * V_i^{(u+1)} \quad (6)$$

The only difference in position update formula from the standard PSO is that, here the velocity is restricted to 80% of its original value. This reduces the search space and hence can be utilized to solve complex multimodal problems also.
The probabilistic learning techniques shows in flowchart in figure 2. The algorithm for modified PSO shown below:
Initialize the basic parameters of PSO and MLFFNN such as, number of iterations, error tolerance, number of particles in PSO, constant 'C', inertial weights 'w_{max}' and 'w_{min}'.
Randomly initialize the position and velocity of each particle corresponding to initial weights and biases for MLFFNN.
Initialize the local best and global best values for each particle and in iteration to zero.
FOR iteration 1 to $ITER_{max}$
DO

- Calculate fitness value of each particle;
- Calculate mean square error of each particle;
- Update the local best of each particle;
- Update global best of each particle;
- Calculate inertial weight using equation (4);

Calculate

- Update local best value of each particle according to its probability of learning P_0;
- Update the velocity of each particle using equation (3);learning probability P_0 using equation (5);
- Update the position of each particle using equation (6);
- Repeat step 4 till itermax is reached;
- Once itermax is reached stop the process.

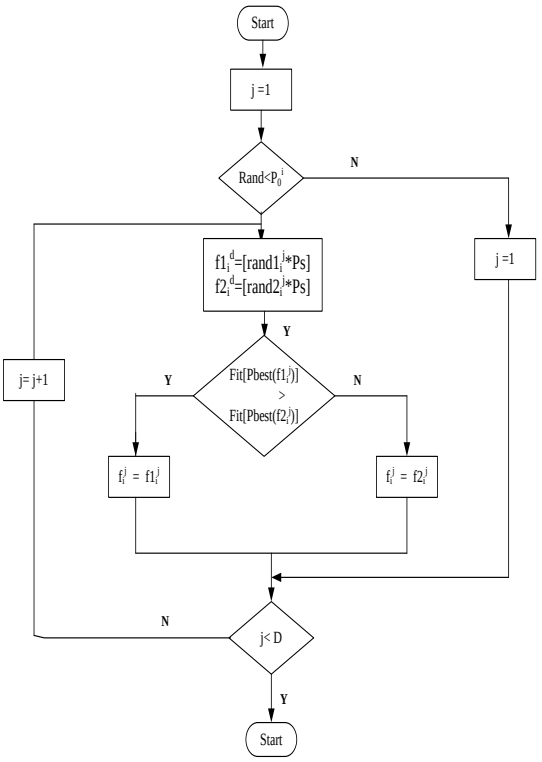

Fig. 2 – Probabilistic learning Strategy of Particles.

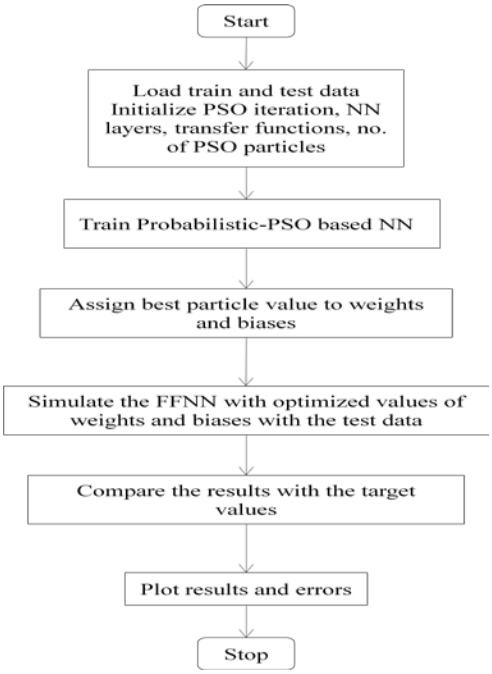

Fig. 3 – Flowchart for the proposed system Using Probabilistic – PSO

3. Results

The propend method used to detect the initial weights and biases for MLFFNN with BP; once the optimum value for initial weights and biases estimated, the MLFFNN then used for assortment and sortilege of various neural network benchmark problems. The benchmarking databases for neural network contain various datasets from various different domains. All datasets represent realistic problems, which called diagnosis tasks, and all the datasets consist of real world data. The results for accuracy of the proposed probabilistic-PSO method compared with extant methods.

In PSO, method selection of parameters c_1, c_2 and w is very much important. The best result obtained when c_1 = 2.01 and c_2= 2.01 and. These results are similar when w chosen according to the formula used. Here the maximum value of w chosen 0.9 and minimum value is chosen 0.4.the velocity limits are selected as v_{max}= 0.5*P_{max} and the minimum velocity is selected as v_{min}= -0.5*P_{min}. There are 10 no of particles are selected in the population.

The results demonstrate that the propane probabilistic-PSO rehearing algorithm when used with MLFFNN yields better assortment accuracy to NN benchmark problems than extant methods as shown in table 2 and also the results of proposed algorithm is validated using different performance parameters such as precision, recall and F-measure as shown in table 3.

Methods	Iris	Diabetes	Breast Cancer	Heart	Wine
PSO	97	78.63	-	--	-
PSO+ FFNN	93.81	77.5	97.4	84.4	95.6
PSO+ CMD	96.22	79.6	89.32	79	97.22
PSO+ GRNN	93.5	76.8	97.2	82.2	94.8
SA+PSO+ BP	100	-	99.66	-	-
S+PSO+ MLFFNN	94.32	-	94.43	-	-
PSO+ELM	94.9	88.4	-	82.7	97.11
SA+BPN	-	81.2	89.1		
GA+JPSNN	-	89.26	-	89.33	94.21
Hybrid PSO+NN	-	87.32	79.88	88.30	
Probabilistic-PSO & MLFFNN	**100**	**93.75**	**99.97**	**96.67**	**99.98**

Table 2 – Classification Accuracy of existing methods, Probabilistic-PSO & MLFFNN over different classification database

Data bases	Class	Precision(P),Recall(R)&F-measure(F)								
		P	R	F	P	R	F	P	R	F
Wine	3	100	100	100	100	100	100	100	100	100
Diabetic	2	80.508	76	78.18	59.459	65.672	62.96	-	-	-
Breast Cancer	2	100	99.074	99.49	97.619	100	98.79	-	-	-
Heart	2	87.097	90.10	88.58	86.967	83.333	85.74	-	-	-
Iris	3	100	100	100	100	100	100	100	100	100

Table 3–Performance of Probabilistic-PSO based on P, R & F for three classes and two classes

4. Conclusions

The proposed probabilistic-PSO based learning algorithm for MLFFNN shows better accuracy than other extant techniques. In the probabilistic PSO, a particle not only learns from its own local best but also from its two neighbouring particles, and thus, the search space is further widens. This results in diversifying the problem solving competence of the PSO. In addition, the restriction of position update formula to 80% of velocity alienates the probability of PSO stuck at the local minima.

The results of benchmark problems for MLFFNN show that probabilistic-PSO based learning algorithm has a wide diversity of solving problems with higher accuracy.

The standard g_{best}, P_{best} are propend which are efficient for space search problems. PSO approaches share information about a best solution found by the swarm or a neighbourhood of particles. Sharing this information introduces a bias in the swarm's search, forcing it to converge on a single solution. When the Influence of a current best solution removed, each particle traverses the search space individually. PSO algorithm proposed to mimic behaviours of social animals more closely through both social interaction and environmental interaction.

In this study, we investigated different PSO algorithm on search optimization and its comparison.

References:

[1] Particle Swarm Optimization, James Kennedy and Russell Eberhart-0-7803-2768-3/95/*IEEE* 1942.

[2] Y. Shi and R. C. Eberhart, A modified particle swarm optimizer, *in Proc. IEEE Congr. Evol. Comput.*, 1998.

[3] Y. Shi and R. C. Eberhart, Parameter selection in particle swarm optimization, *in Proc. 7th Conf. Evol. Programming, New York,* 1998, pp. 591–600.

[4] Y. Shi and R. C. Eberhart, Particle swarm optimization with fuzzy adaptive inertia weight, *in Proc. Workshop Particle Swarm Optimization*, Indianapolis, IN, 2001, pp. 101–106.

[5] Ratnaweera, S. Halgamuge, and H. Watson, Self-organizing hierarchical particle swarm optimizer with time varying accelerating coefficients, IEEE Trans. Evol. Comput, vol. 8, pp. 240–255, Jun. 2004.

[6] M. Clerc and J. Kennedy, The particle swarm-explosion, stability, and convergence in a multidimensional complex space, *IEEE Trans. Evol. Comput*, vol. 6, no. 1, pp. 58–73, Feb. 2002.

[7] J. Kennedy, Small worlds and mega-minds: Effects of neighbourhood topology on particle swarm performance, in Proc. Congr. Evol. Comput., 1999, pp. 1931–1938.

[8] J. Kennedy and R. Mendes, Population structure and particle swarm performance, *in Proc. IEEE Congr. Evol. Comput. Honolulu*, HI, 2002, pp. 1671–1676.

[9] P. N. Suganthan, Particle swarm optimizer with neighborhood operator, *in Proc. Congr. Evol. Comput., Washington, DC*, 1999, pp. 1958–1962.

[10] K. E. Parsopoulos and M. N. Vrahatis, UPSO—A unified particle swarm optimization scheme, *in Lecture Series on Computational Sciences*, 2004, pp. 868–873.

[11] R. Mendes, J. Kennedy, and J. Neves, The fully informed particle swarm: Simpler, maybe better, *IEEE Trans. Evol. Comput.*, vol. 8, pp. 204–210, Jun. 2004.

[12] T. Peram, K. Veeramachaneni, and C. K. Mohan, Fitness-distance-ratio based particle swarm optimization, *in Proc. Swarm Intelligence Symp.*, 2003, pp. 174–181.

[13] P. J. Angeline, Using selection to improve particle swarm optimization, *in Proc. IEEE Congr. Evol. Comput., Anchorage, AK,* 1998, pp. 84–89.

[14] T. M. Blackwell and P. J. Bentley, Don't push me! Collision-avoiding swarms, *in Proc. IEEE Congr. Evol. Comput. Honolulu, HI,* 2002, pp. 1691–1696.

[15] X. Xie, W. Zhang, and Z. Yang, A dissipative particle swarm optimization, *in Proc. Congr. Evol. Comput. Honolulu, HI,* 2002, pp. 1456–1461.

[16] K. E. Parsopoulos and M. N. Vrahatis, On the computation of all global minimizers through particle swarm optimization, *IEEE Trans. Evol. Comput.* vol. 8, pp. 211–224, Jun. 2004.

[17] Jing Liang, Ponnuthural N. Sunanthan, Kai Qin and Baskar Subramanian, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE transactions on evolutionary computation*, vol. 10, No. 3, June 2006.

[18] Pravin Kshirsagar and Dr. Sudhir Akojwar, Novel Approach for Classification and Prediction of Non Linear Chaotic Databases, *International Conference on Electrical, Electronics, and Optimization Techniques*, March 2016.

[19] Jing Liang, Ponnuthural N. Sunanthan, Kai Qin and Baskar Subramanian, Evaluation of comprehensive learning particle swarm optimizer, *in Proc. 11th Int. Conf. Neural Information Processing, Science City, Calcutta, India,* Nov. 2004. [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan.

[20] Lutz Prechelt, Fakultat Fur. 1994. A set of Neural Network Benchmark Problems and Benchmarking Rules. *Technical Report 21/94, University of Germany*.

[21] Pravin Kshirsagar and Sudhir Akojwar, "Hybrid Heuristic Optimization for Benchmark Datasets", *International Journal of Computer Application* (0975-8887), Vol.146-No.7, July 2016.