Optimizing Back-Propagation using PSO_Hill_A* and Genetic Algorithm

Priyanka sharma[1] and Asha Mishra[2]
[1]M.Tech Scholar of computer science & Engineering , BSAITM, Faridabad
[2]Department of computer science & Engineering , BSAITM, Faridabad

Abstract:
Back propagation(BP) is used to solve real world problems which use the concept of multilayer perceptron(MLP). BP have the disadvantage of trapped in local minima, slow convergence rate and more error prone. To optimize BP Algorithm we used Particle swarm optimization(PSO) and Genetic algorithm(GA). Limitation of PSO_Hill and PSO_A* is overcomes when these algorithms are combined and on the basis of strength of these two algorithm we proposed a new PSO_Hill_A* algorithm which is used to optimize and enhance learning process in terms of convergence rate and accuracy. GA is a kind of method to simulate and to search the optimal solution, GA can have four operations including Encoding, selecting, crossover and Mutation. To optimize and improve BP, we proposed two architecture: 1) Use of PSO_Hill_A* before and after hidden layer. 2) Use of GA before and after hidden layer.

Keywords:
PSO_Hill_A, BPA, GA, PSO_Hill, PSO_A**

1. Introduction:

Several modern heuristic tools that facilitate the solution of optimization problems which were previously difficult or impossible to address have evolved in the last two decades. These tools include evolutionary computation, simulated annealing, tabu search, and particle swarm, among others. Recently, the genetic algorithm (GA) and particle swarm optimization (PSO) techniques emerged as promising algorithms for handling optimization problems. With the development of artificial intelligence in recent years, some approaches have been presented to using ANNs with a back propagation(BP) algorithm [2], Genetic Algorithm (GA) [2, 3-5] and Particle Swarm Optimization (PSO) [4, 5-6] methods.

Back propagation is a gradient-based method. Although the BP algorithm has solved a number of practical problems, but firstly it easily gets trapped in local minima especially for complex function approximation problem, so that back propagation may lead to failure in finding a global optimal solution. Second, the convergent speed of the BP algorithm is too slow even if the learning goal, a given termination error, can be achieved.

To improve the performance of the original BP algorithm[8], researchers have concentrated on the following two factors: (1) selection of better energy function; (2) selection of dynamic learning rate and momentum. However, these improvements have not removed the disadvantages of the BP algorithm getting trapped into local optima in essence. In particular, with feed-forward neural networks structure becoming more complex; its convergent speed will be even slower [6].

GA is a optimization method based on the Darwinian principles of biological evolution, reproduction, and “the survival of the fittest” [2]. GA maintains a set of candidate solutions called population and repeatedly modifies them. At each step, the GA selects individuals from the current population to be parents and uses them produce the children for the next generation. Over successive generations, the population evolves toward an optimal solution

Genetic algorithm seems to be good methods to solve optimization problems, when applied to problems consisting of more number of local optima, the solution from GA are just near global optimum areas. Also, it takes long simulation time to obtain the solution. Moreover, when the number of parameter is more, optimization problem is complex and coding chromosomes with more genes for increasing algorithm accuracy. It should be noted that optimization of neural network architecture design, including selecting the number of input variables, input nodes and the number of hidden neurons, to improve forecasting performance is becoming more and more important and desirable.

PSO is inspired by the capability of flocks of birds, schools of fish, and herds of animals to adapt to their environment. In PSO, a set of randomly generated solutions propagates in the design space toward the optimal solution over a number of iterations. Recently, a new evolutionary computation technique, PSO technique finds the optimal solution using a population of particles. Each particle represents a candidate solution to the problem. Some of the attractive features of the PSO include ease of implementation and that no gradient information is required [10].

The PSO algorithm has a strong ability to find the most optimistic result, but it has a disadvantage of easily getting into a local optimum [11-12]. The BP algorithm, on the contrary, has a strong ability to find local optimistic result.

Manuscript received May 5, 2014
Manuscript revised May 20, 2014

In this paper, we used genetic algorithm and particle swarm optimization (PSO_Hill_A*) algorithm to improve BP algorithms is proposed and to eliminate the known drawbacks of the ANN trained by the BP.

2. Back Propagation Algorithm (BPA)

Back-Propagation (BP) algorithm was proposed in 1986 by Rumelhart, Hinton and Williams for setting weights and hence for the training of Multi-Layer Perceptrons (MLP) [14]. Back propagation is a gradient-based method. The back- propagation algorithm is used in layered feed-forward ANNs [21],[1]. The BP algorithm propagates backward the error between the desired signal and the network output through the network. After providing an input pattern, the output of the network is then compared with a given target pattern and the error of each output unit calculated. This error signal is propagated backward, and a closed-loop control system is thus established. The weights can be adjusted by a gradient descent-based algorithm[13]. BP algorithm has solved a number of practical problems, but it easily gets trapped in local minima especially for complex function approximation problem, so that back propagation may lead to failure in finding a global optimal solution. Second, the convergent speed of the BP algorithm is too slow even if the learning goal, a given termination error, can be achieved. The important problem to be stressed is that the convergent behavior of the BP algorithm depends very much on the choices of initial values of the network connection weights as well as the parameters in the algorithm such as the learning rate and momentum. To improve the performance of the original BP algorithm, researchers have concentrated on the following two factors:
(1) selection of better energy function;
(2) selection of dynamic learning rate and momentum.
However, these improvements have not removed the disadvantages of the BP algorithm getting trapped into local optima in essence. In particular, with feed-forward neural networks structure becoming more complex; its convergent speed will be even slower [6].

3. Genetic algorithm (GA)

Genetic algorithm is a search technique based on the concept of evolution[15], and in particular with the concept of the survival of the fittest[16]. The application of genetic algorithm on neural network make an hybrid neural network where the weights of the neural network are calculated using genetic algorithm approach. From all the search spaces of all the possible weights, the genetic algorithm will generate new points of the possible solution. The first step to calculate its values, is to define the solution domain with a genetic representation (problem encoding) and a fitness function to determine the better solutions. Those two component of a genetic algorithm are the only problem dependent of the genetic algorithm approach[25]. Once an initial population of elements (chromosomes or genotype[17]) have been created, the techniques used by this algorithms to converge to a solution of the problem are related to the evolutionary theory:
1. Selection: some chromosomes of the current population are selected to breed a new generation. A small number are selected randomly while the others are selected depending on how they fit better with a fitness function.

2. Genetic operations: once the first chromosomes have been defined by those that fits better the fitness function, the rest of the population is going to be created using genetic operations. The fitness of those new chromosomes will also be checked and compared with the worst chromosomes of the last generation in order to decide who will stay into the population.
(a) Mutation: modifying one or more bits (gens) of some chromosomes of the population.
(b) Crossover: crossing two or more chromosomes of the population between
them.
A pseudo-code for this algorithm is:

```
1. Creation of the initial population.
2. while (!solution)
(a) Evaluate the fitness of all the chromosomes of the population.
(b) The best chromosomes will be selected to reproduce, using mutation and crossover.
(c) Substitute the worst chromosomes of the previous generation by the new produced chromosomes.
```

This process was referred to as Simple Genetic Algorithm[18]. Finally, the fittest chromosome will be selected as a solution.

4. Particle Swarm Optimization (PSO)

Particle swarm optimization algorithm was introduced by Kennedy and Eberhart in 1995 [19], [20]. The algorithm consists of a swarm of particles flying through the search space. Particle Swarm Optimization (PSO) is a swarm-based intelligence algorithm [19] influenced by the social behavior of animals such as a flock of birds finding a food source or a school of fish protecting themselves from a predator. A particle in PSO is analogous to a bird or fish flying through a search (problem) space. The movement of each particle is co-coordinated by a velocity which has both magnitude and direction. Each particle position at any instance of time is influenced by its best position and the position of the best particle in a problem space. The performance of a particle is measured by a fitness value, which is problem specific. Each particle will have a fitness value, which will be evaluated by a fitness function to be optimized in each generation. Each particle knows its best position L_best and the best position so far among the entire group of particles G_best[21]. The L_best of a particle is the best result (fitness value) so far reached

The particle swarm optimization (PSO) algorithm is based on the evolutionary computation technique. PSO optimizes an objective function by conducting population-based search. The population consists of potential solutions, called particles, similar to birds in a flock. The particles are randomly initialized and then freely fly across the multi-dimensional search space. While flying, every particle updates its velocity and position based on its own best experience and that of the entire population. The updating policy will cause the particle swarm to move toward a region with a higher object value. Eventually, all the particles will gather around the point with the highest object value. PSO attempts to simulate social behavior, which differs from the natural selection schemes of genetic algorithms.

PSO processes the search scheme using populations of particles, which corresponds to the use of individuals in genetic algorithms. Each particle is equivalent to a candidate solution of a problem. The particle moves according to an adjusted velocity, which is based on that particle's experience and the experience of its companions.

5. Related work

Priyanka Sharma, Asha Mishra [1] proposed two algorithm PSO_Hill and PSO_A* to optimize Back propagation algorithm (BPA) which have the problem of complexity, local minima so we are using Particle Swarm optimization (PSO) algorithms to reduce and optimize BPA. PSO_Hill and PSO_A* algorithms are analyzed and evaluated on the basis of their advantages, applied to feed forward neural network(FNN) for back propagation algorithm(BPA) which is a gredient desent technique. where BPA is used for non_linear problems. These non_linear problems are improved by a PSO_Hill and PSO_A* algorithms.

H. Shayeghi, H.A. Shayanfar, G. Azimi [8]: As accurate Short Term Load Forecasting(STLF) is very important for improvement of the management performance of the electric industry. An issue of the optimal design of a neural network based short term load forecaster is addressed. A new hybrid evolutionary algorithm combining the Particle Swarm Optimization (PSO) algorithm and Back Propagation (BP) algorithm, referred to as HPSOBP algorithm, is proposed to evolve the optimum large neural network structure, connecting weights and bias values for one-day ahead electric load forecasting problem. The hybrid algorithm can make use of not only strong global searching ability of the PSO algorithm, but also strong local searching ability of the BP algorithm.

M. Conforth and Y. Meng [22] propose a swarm intelligence based reinforcement learning (SWIRL) method to train artificial neural networks (ANN). Basically, two swarm intelligence based algorithms are combined together to train the ANN models. Ant Colony Optimization (ACO) is applied to select ANN topology, while Particle Swarm Optimization (PSO) is applied to adjust ANN connection weights. To evaluate the performance of the SWIRL model, it is applied to double pole problem and robot localization through reinforcement learning. Extensive simulation results successfully demonstrate that SWIRL offers performance that is competitive with modern neuro-evolutionary techniques, as well as its viability for real-world problems.

Y. Karpat and Tugrul Ozel [23] propose a concept of particle swarm optimization, which is a recently developed evolutionary algorithm, is used to optimize machining parameters in hard turning processes where multiple conflicting objectives are present .The relationships between machining parameters and the performance measures of interest are obtained by using experimental data and swarm intelligent neural network systems (SINNS). The results showed that particle swarm optimization is an effective method for solving multi-objective optimization problems, and an integrated system of neural networks and swarm intelligence can be used in solving complex machining optimization problems.

James kennedy and Russell Eberhart [24] propose a concept for the optimization of nonlinear functions using particle swarm methodology is introduced. The evolution of several paradigms is outlined, and an implementation of one of the paradigms is discussed. Benchmark testing of the paradigm is described, and applications, including nonlinear function optimization and neural network training, are proposed. The relationships between particle swarm optimization and both artificial life and genetic algorithms are described.

Sergi Perez [25]: Natural networks have been used during several years to solve classification problems. The performance of a neural network depends directly on the design of the hidden layers, and in the calculation of the weights that connect the different nodes. On this project, the structure of the hidden layer is not modified and calculation of the weights of the system. In order to obtain a feasible result, the weights of the neural network are calculated due a function cost. A genetic algorithm approach is presented and compared with the gradient descent in failure rate and time to obtain a solution.

6. Proposed work

6.1 Proposed Variants for optimization: 1) PSO_Hill_A*
Improving and optimizing BPA by using PSO_Hill_A* .
Improving and optimizing BPA by using GA .

6.1.1 PSO_Hill_A* Algorithm:

```
1. initialize initial position of particle Xj , weight wj
   and velocity Vj, P_loc→0,p_glob→0,n→0
2. Vj(t+1)=Vj+Xj*Wj
3. if goal Vj(t+1)= actual Vj(t+1) then terminate with
   success
4. otherwise
5. if ( j<n)
6. {
7. compare p_loc with P_glob
8. {
9. If p_glob position is better than p_loc
10.Set P_current==p_glob
11. terminate with success
12. }
13.}
14. compute fitness function of each particle
15. if it is best solution among all the particles,
    terminate with success
16. otherwise
17. return to step 2
```

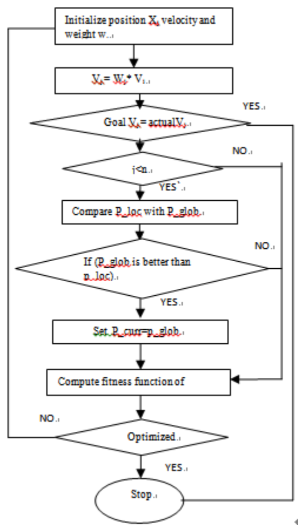
6.1.2 PSO_hill_A*Advantage:

1) Less c.p.u time
2) More Stability
3) Fitness function
4) More optimized
5) High computational speed
6) Strong ability in global search
7) Higher Accuracy
8) Learning achieved from particle own experienced
9) Learning achieved from experience of cooperation between particles

6.1.3 PSO_Hill_A* Diagram:

6.2 Genetic algorithm

- Encoding: Real to binary number value encoding
- Selecting: Roulette wheel selection of two parents for mutation
- Fitness: Evaluate the fitness of all the chromosomes of the population..
- Crossover: We are using single point crossover.
- Mutuation: Mutation prevent it to be trapped in local minima and maintain diversity in population. We are using flipping in which mutuation chromosome is randomly generated.


Initialize position Xj velocity and weight w
Vj= Wj* Vj
YES
Goal Vj= actualVj
NO
j<n
YES
Compare P_loc with P_glob
NO
If (P_glob is better than p_loc)
YES
Set P_curr=p_glob
Compute fitness function of
NO
Optimized
YES
Stop

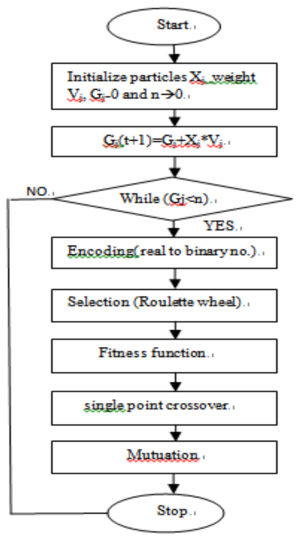

Start
Initialize particles Xj, weight Vj, Gj-0 and n→0
Gj(t+1)=Gj+Xj*Vj
NO
While (Gj<n)
YES
Encoding(real to binary no.)
Selection (Roulette wheel)
Fitness function
single point crossover
Mutuation
Stop

Fig. Diagram of Genetic Algorithm

6.3 Optimizing methods of back propagation algorithm

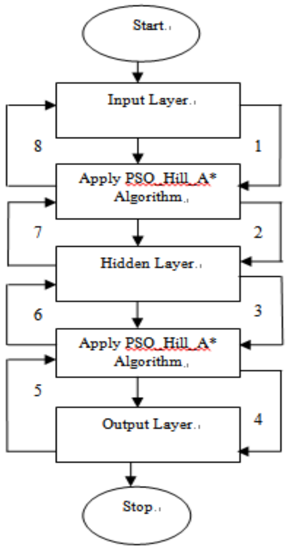
1) Improving and optimizing BPA by using PSO_Hill_A* .
2) Improving and optimizing BPA by using GA

Diagram step description of Optimizing BPA by using PSO_Hill_A* and GA :

1. Less error propagated in 1, 2, 3 and 4 step from I/P layer to PSO_Hill_A*/GA block, PSO_Hill_A*/GA block to hidden layer, hidden layer to PSO_Hill_A*/GA block and PSO_Hill_A*/GA block to o/p layer .
2. Error minimized in 5, 6, 7 and 8 step from O/P layer to PSO_Hill_A*/GA block,

PSO_Hill_A*/GA block to hidden layer, Hidden layer to PSO_Hill_A*/GA block and PSO_Hill_A*/GA block to I/p layer.

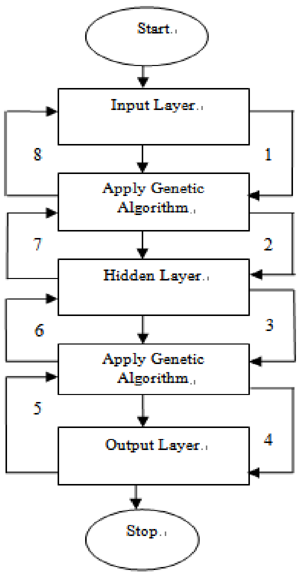
6.3.1 Diagram Of Optimizing BPA by using PSO_Hill_A* : In this flow chart we are using PSO_Hill_A* algorithm before and after hidden layer to minimize the error that are pass through from input to hidden layer and hidden to output layer and less error is propagated from output to hidden layer and hidden to input layer. Hence, this architecture optimize and improve BPA.

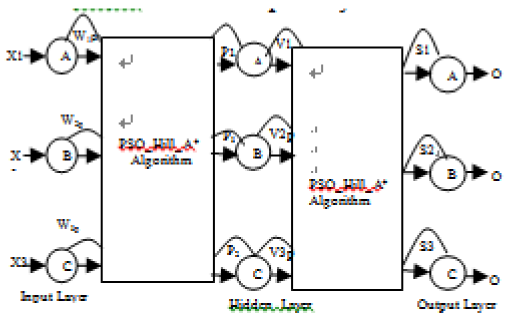


6.3.2 Improving and optimizing BPA by using GA: In this flow chart we are using GA before and after hidden layer to minimize the error that are pass through from input to hidden layer and hidden to output layer and less error is propagated from output to hidden layer and hidden to input layer. Hence, this architecture optimize and improve BPA.

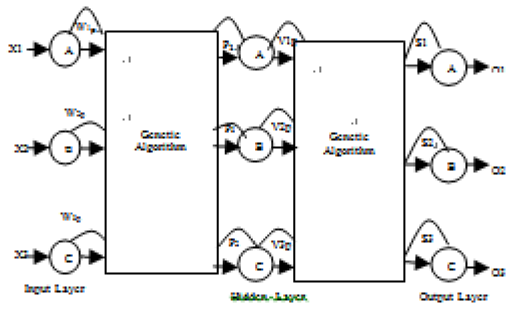
Notation's of Architecture's:

1. Subscript 1,2,3: denotes the notations of neuron 1, 2 and 3
2. O/p: Output , I/P: input
3. X_1, X_2,X_3: Input particles
4. W_{1p},w_{2p},w_{3p}: weight from I/P layer neuron 1 to PSO block/ GA block (before hidden layer)
5. P_1,P_2,P_3: O/p of PSO_Hill_A*/GA algorithm (before hidden layer)
6. V_{1p},V_{2p},V_{3p}: weight from hidden layer neuron 1 to PSO block(after hidden layer)
7. S_1,S_2,S_3: O/p of PSO_Hill_A*/GA algorithm (after hidden layer)
8. O_1,O_2,O_3: O/P of output Layer.
9.





6.3.2.1 Architecture of Optimizing BPA by using GA : In this we are using 3 neurons at input, hidden, output layer, input particles and weights. Here front forwarded arrows are used to show the inputs and weights from input to GA block GA block to hidden layer and hidden to GA block and GA block to output layer. We are using arcs to show the error's propagated back warded from output to GA block and GA block to hidden layer and hidden to GA block and GA block to input layer.



6.3.1.1 Architecture of Optimizing BPA by using PSO_Hill_A* : In this we are using 3 neurons at input, hidden, output layer, input particles and weights. Here front forwarded arrows are used to show the inputs and weights from input to PSO_Hill_A* block PSO_Hill_A* block to hidden layer and hidden to PSO_Hill_A* block and PSO_Hill_A* block to output layer. We are using arcs to show the error's propagated back warded from output to PSO_Hill_A* block and PSO_Hill_A* block to hidden layer and hidden to PSO_Hill_A* block and PSO_Hill_A* block to input layer

7. Conclusion

In this paper, variant of the particle swarm optimization scheme PSO_Hill_A* is presented. PSO Variant PSO_Hill_A* algorithm is proposed on the strength of these two algorithm PSO_Hill and PSO_A* with its algorithm, architecture, advantages and disadvantages, which can be used to the optimize the BPA. In this paper, we are optimizing and minimizing error by using two architecture: 1) Use of PSO_Hill_A* before and after hidden layer 2) Use of GA before and after hidden layer. In next paper, we are implementing this proposed work using PSO tool in matlab and GA tool in matlab.

Refrences

[1] Priyanka Sharma, Asha Mishra, "Optimizing Back-Propagation using PSO_Hill and PSO_A*", Int. J. of Scientific and Research Publications.

[2] H. Shayeghi, H.A. Shayanfar and G. Azimi,"Intelligent Neural Network Based STLF", Int. Journal of Intelligent Systems and Technologies, Vol. 4, No. 1, pp. 17-27, 2009.

[3] D. Srinivasan, "Evolving Artificial Neural Networks for Short Term Load Forecasting", Neurocomputing, Vol. 23, pp. 265-276, 1998.

[4] Z.A. Bashir and M.E. EL-Hawary, "Applying Wavelets to Short term Load Forecasting Using PSOBased Neural Networks", IEEE Trans. on Power Systems, Vol. 24, No. 1, pp. 20-27, 2009.

[5] C.F. Juang, "A Hybrid of Genetic Algorithm and Particle Swarm Optimization for Recurrent Network Design, IEEE Trans. on systems, Man, and Cybernetics- Part B: Cybernetics, Vol. 34, No. 2, pp. 997-1006, 2004.

[6] H. Shayeghi, H.A. Shayanfar and G. Azimi, "STLF Based on Optimized Neural Network Using PSO", Int. J. of Electronics, Circuits and Systems, Vol. 3, No. 2, pp.113-123, 2009.

[7] G.C. Liao and T.P. Tsao, "Application of a Fuzzy Neural Network Combined with a Chaos Genetic Algorithm and Simulated Annealing to Short term Load Forecasting", IEEE Trans. Evolutionary Computation, Vol. 10, No. 3, pp. 330-340, 2006.

[8] H. Shayeghi, H.A. Shayanfar, G. Azimi," A hybrid particle swarm optimization back propagation algorithm for short term load forecasting", Int. J .on Technical and Physical Problems of Engineering, Vol 2, Issue 4

[9] A.U. Asar, S.R.U. Hassnain and A. Khan, "Short term Load Forecasting Using Particle Swarm Optimization Based ANN Approach", Proc. of IEEE Int. Joint Conf. on Neural Networks, Orlando, Florida, USA, pp. 6, 2007.

[10] M. Clerc and J. Kennedy, "The Particle Swarm Explosion, Stability, and Convergence in a Multidimentional Complex Space", IEEE Trans. On Evolutionary Computation, Vol. 6, No. 1, pp. 58-73, 2002.

[11] J.R. Zhang, J. Zhang, T.M. Lok and M.R. Lyu, "A Hybrid Particle Swarm Optimization Back Propagation Algorithm for Feed-forward Neural Network Training", Applied Mathematics and Computation, Vol. 185, pp.1026-1037, 2007.

[12] Y. Shi, "Particle Swarm Optimization", Electronic Data Systems, Inc. IEEE Neural Networks Society, pp. 8-13, 2004.

[13] Dr. Hanan A. R. Akkar, Samem Abass Salman," Training Artificial Neural Network Using Back-Propagation & Particle Swarm Optimization for Image Skin Diseases", Eng & Tech Journal, Vol. 29, No. 13, 2011.

[14] D. Graupe, "Principle of Artificial Neural Networks", World Scientific Publishing Co. Pte. Lte., 2007.

[15] Darwin, Ch. (1859) The origin of species by means of natural selection.

[16] Whitley, D. (1994) A Genetic Algorithm Tutorial. Colorado State University, Computer Science Department.

[17] Holland, J. (1975) Adaptation in Natural and Artificial Systems. University of Michigan Press.

[18] Goldberg, D. (1987) Simple Genetic Algorithms and the Minimal, Deceptive Problem. In L.Davis, ed., Pitman Genetic Algorithms And Simulated Annealing.

[19] J.Kennedy and R.Eberhart,"Particle swarm optimization",Proceedings of. IEEE International Conference on Neural Networks, pp. 1942-1948, 1995

[20] V.Selvi and Dr.R.Umarani, "Comparative Analysis of Ant Colony and Particle Swarm Optimization Techniques", International Journal of Computer Applications (0975 – 8887), Volume 5– No.4, August 2010.

[21] Carlos Gershenson, "Artificial Neural Networks for Beginners"

[22] Matthew Conforth and Yan Meng ,"Reinforcement Learning for Neural Networks using Swarm Intelligence" , 2008 IEEE Swarm Intelligence Symposium, St. Louis MO USA, September 21-23, 2008

[23] Yiğit Karpat and Tuğrul Özel, "Swarm-Intelligent Neural Network System (SINNS) Based Multi-Objective Optimization Of Hard Turning" ,Transactions of NAMRI/SME Volume 34, 2006.

[24] James Kennedy' and Russell Eberhart2, "Particle Swarm Optimization", Washington, DC 20212,kennedy_jim@bls.gov,http://www.cs.tufts.edu/comp/150GA/homeworks/hw3/_reading6%201995%20particle%20swarming.pdf, 1995 IEEE.

[25] Sergi Perez, "Apply genetic algorithm to the learning phase of a neural network".