STRUCTURE OPTIMIZATION OF ARTIFICIAL NEURAL NETWORKS USING SWARM INTELLIGENT

Assist. Prof. Dr. HANAN A. R. AKKAR
Department of Electrical and Electronic Engineering
University of Technology
Baghdad\Iraq
Email: hnn_aaa@yahoo.com

ABSTRACT

The structure is a very important aspect in neural network design, it is not only impossible to determine an optimal structure for a given problem, it is even impossible to prove that a given structure is optimal. In this paper, PSO (Particle Swarm Optimization) are used to construct best ANN (Artificial Neural Network) architectures, and find an optimal pattern of connections and weights to reduce structure complexity by minimizing the number of connection weights in a Feed Forward Artificial Neural Network (FFANN). They are called Particle Swarm Optimization-Neural Network systems (PSONN). PSONN systems are examined through theoretical analysis and computer simulation using MATLAB package. They are tested by several different examples, where the tests show that PSO a more efficient and automated search method can be used to find an optimal topology of ANN. The best and trained network with few numbers of iteration is provided using PSONN for finding an optimal structure. Finally, a simpler network, faster training with higher accuracy than full connected network is obtained by using PSONN for finding optimal connections and weights.

Keywords: Artificial neural networks, Particle swarm optimization, Pattern recognition.

أمثلة هيكلية الشبكات العصبية الاصطناعية باستخدام الحشود الذكية

الاستاذ المساعد الدكتور حنان عبد الرضا عكار
قسم الهندسة الكهربائية والالكترونية/ الجامعة التكنولوجية
بغداد / العراق

الملخص

تم في هذا البحث استخدام افضلية الحشد الجزيئي(PSO) لبناء افضل هيكلية للشبكات العصبية عن طريق تقليل ربط الاوزان ل(FFANN) وتدريب الشبكة العصبية بنفس النظام المسمى(PSONN). الهيكلية في تصميم الشبكات العصبية من الامور المهمة وتعتبر من اهم المشاكل التي يعاني منها المصمم لذلك من المهم البحث عن افضل تصميم للشبكات العصبية. تم الاختبار باستخدام الطرق النظرية والتمثيل بواسطة حقيبة MATLAB. حيث تم الاختبار باستخدام عدة امثلة وجد من خلالها ان PSOكفوء بايجاد افضل تصميم من حيث تقليل التعقيد في الربط بين خلية عصبية واخرى مما سرع في عمل الشبكة ولم يؤثر على ادائها وفي نفس الوقت تدريب الشبكة العصبية باستخدام PSO مما حسن اداء الشبكات العصبية بشكل كبير.

1. INTRODUCTION

Particle Swarm Optimization (PSO) is a relatively new evolutionary algorithm that may be used to find optimal (or near optimal) solutions to numerical and qualitative problems. Particle Swarm Optimization was originally developed by a social psychologist (James Kennedy) and an

electrical engineer (Russell Eberhart) in 1995 [**kennedy 2001**], and emerged from earlier experiments with algorithms that modeled the flocking behavior seen in many species of birds. Although there were a number of such algorithms getting quite a bit of attention at the time, Kennedy and Eberhart became particularly interested in the models developed by biologist Frank Heppner [**Mendes 2002**]. Heppner studied birds in flocking behaviors mainly attracted to a roosting area. In simulations, birds would begin by flying around with no particular destination and spontaneously formed flocks until one of the birds flew over the roosting area.

Due to the simple rules the birds used to set their directions and velocities, a bird pulling away from the flock in order to land at the roost would result in nearby birds moving towards the roost. Once these birds discovered the roost, they would land there, pulling more birds towards it, and so on until the entire flock had landed. Finding a roost is analogous to finding a solution in a field of possible solutions in a solution space. The manner in which a bird who has found the roost, leads its neighbors to move towards it, increases the chances that they will also find it. Eberhart and Kennedy revised Heppner's methodology so that particles could fly over a solution space and land on the best solution simulating the birds' behavior [**Clerc 2006**].

Neural network is an artificial intelligence model originally designed to replicate the human brain's learning process. A typical network is composed of a series of interconnected nodes and the corresponding weights. It aims at simulating the complex mapping between the input and output. The training process is carried out on a set of data including input and output parameters. The learning procedure is based on the training samples and the testing samples are used to verify the performance of the trained network. During the training, the weights in the network are adjusted iteratively till a desired error is obtained [**Blum 2008, Shi 2004**]. This capability of learning from data without a prior knowledge makes neural networks particularly suitable for classification and regression tasks in practical situations.

The most popular supervised learning technique in ANN is Back-Propagation BP algorithm. The back-propagation algorithm involves the gradual reduction of the error between model output and the target output. It develops the input to output, by minimizing a Mean Square Error cost function measured over a set of training examples. The problem with neural networks is that a number of parameters have to be set before any training can begin. However, there are no clear rules as to how to set these parameters. Yet these parameters determine the success of the training. Therefore, PSO is used to find their optimum values [**Su 2008, Engelbrecht 2007**].

A particle represents the weight vector of NN, including all biases. The dimension of the search space is therefore the total number of weights and biases. The fitness function is the Mean Squared Error (MSE) over the training set. Changing the position means updating the weight of the network in order to reduce the error. All the particles update their position by calculating the new velocity, which they use to move each particle to the new position. The new position is a set of new weights used to obtain the new error. For PSO, the new weights are adapted even though no improvement is observed. This process is repeated for all the particles. The particle with the lowest error is considered as the global best particle so far. The training process continues until satisfactory error is achieved by the best particle or computational limits are exceeded. When the training ends, the weights are used to calculate the classification error for the training patterns. The same set of weights is used then to test the network using the test patterns [8, 9].
In this paper the ability of Particle Swarm Optimization PSO in designing Artificial Neural Networks (ANN) is discussed. The Multi-Layer Perceptron (MLP) is taken into account as an ANN structure to be optimized [**Chandramouli 2006**].

2. PARTICLE SWARM OPTIMIZATION

In Particle Swarm Optimization, the particles are "flown" through the problem space by following the current optimum particles. Each particle keeps track of its coordinates in the problem space which are associated with the best solution (fitness) that it has achieved so far. This implies that each particle has a memory, which allows it to remember the best position on the feasible search space that it has ever visited. This value is commonly called *pbest*. Another best value that is

tracked by the particle swarm optimizer is the best value obtained so far by any particle in the neighborhood particle. This location is commonly called *gbest* [**Wang 2008**].

The particles learn over time in response to their own experience and the experience of the other particles in their group. Each particle keeps track of its best fitness position in hyperspace that has achieved so far. This value is called personal best or p_{best}. The overall best value obtained by so far by any particle in the population is called global best or g_{best} . During each epoch (or iteration) every particle is accelerated towards its own personal best as well as in the direction of the global best position. This is achieved by calculating a new velocity term for each particle based on the distance from its personal best, as well as its distance from the global best position. These two components ('personal' and 'global' velocities) are then randomly weighted to produce the new velocity value for this particle, which will in turn affect the next position of the particle during the next epoch. **Figure (1)** shows the basic PSO procedure.

2.1 Implementation Steps of PSO

PSO is basically developed through simulation of bird flocking in two-dimension space. The position of agent is represented by xy-axis position and also the velocity is expressed by V_x (the velocity of x-axis) and V_y (the velocity of y-axis). Modification of agent position is realized by position and velocity information.

PSO is a population-based stochastic optimization technique modeled on the social behaviors in animals or insects, bird flocking, fish schooling and animal herding. PSO initially has population of random solutions each potential solution called particle, is flown through the problem space, the particles have memory and each particle keeps track of previous best position and corresponding fitness. The previous best value is called as 'Pbest' it also has an other value called 'Gbest' which is the best value of all the particles Pbest in the swarm. The basic concept of PSO technique lies in accelerating each particle toward its Pbest and the Gbest locations at each time step. The main equations in PSO described as below:

$$V_i(t+1) = \alpha V_i(t) + c_1 r_1 (P_{best\ i}(t) - X_i(t)) + c_2 r_2 (G_{best}(t) - X_i(t)) \quad (1)$$

$$(X_i(t+1) = X_i(t) + V_i(t+1) \quad (2)$$

Where α are random numbers in the range [0, 1], c_1 and c_2 are positive constants, r_1 and r_2 represents the rate of the position change (velocity) for particle *i*, $V_i(t)$ represent the ith particle, and $X_i(t)$ is the inertial weight. The PSO algorithm is simple in concept easy to implement and computational efficient. The original procedure for implementing PSO is as follows:

1. Initialize the swarm in solution space $t = 0$, $P_i \in P(t)$ is random position $X_i(t)$ within the solution space and the velocity $V_i(t) = (0 \cdots 0)$
2. Evaluate fitness of individual particles. Fitness of $P_i(t) = f(X_i(t))$.
3. Modify Gbest, Pbest and velocity.
   If $f(X_i(t)) > Pbest_i$ then $Pbest_i = f(X_i(t))$ and $Xpbest_i = X_i(t)$
   If $f(X_i(t)) >$ Gbest then $Gbest = f(X_i(t))$ and $Xgbest = X_i(t)$

Update velocity according to equation (1).

4. Move each particle to a new position according to equation (2).
5. Repeat until convergence or a stopping condition is met or maximum number of iteration. Figure (1) shows the basic procedure of PSO algorithm[**Venayaga 2007**]

2.2 The Success and Limitation of PSO

PSO is a computational intelligence based technique that is not largely affected by the size and nonlinearity of the problem, and can converge to the optimal solution in many problems where most analytical methods fail to converge. It can therefore, be effectively applied to different optimization problems. It's an optimization algorithm that using only primitive mathematic calculation. So the advantage of the PSO over many of the other optimization algorithms is its ability to handle optimization problems with multiple local optima reasonably well and relative simplicity of implementation the calculation speeds and fast.

Moreover, PSO has advantages over other similar optimization techniques, namely the following:

1) PSO is easier to implement and there are fewer parameters to adjust.
2) In PSO, every particle remembers its own previous best value as well as the neighborhood best; therefore, it has a more effective memory capability than other optimization technique.
3) PSO is more efficient in maintaining the diversity of the swarm (more similar to ideal social interaction in a community), since all the particles use the information related to the most successful particle in order to improve themselves

There are not many parameter need to be turned in PSO. Here is a list of the parameters and their typical values

- The number of particles, the typical range is about 10 to 40. Actually for most of the problems 10 particles is large enough to get good results. For some difficult or special problems, one can try 100 or 200 particles as well.
- Dimension of particles, it is determined by the problem to be optimized.
- Range of particles, it is also determined by the problem to be optimized, we can specify different ranges for different dimension of particles.
- Learning factors, c1 and c2 usually equal to 2.
- The stop condition, the maximum number of iterations the PSO execute and the minimum error requirement [**Venayaga 2007**].

3. COMPUTER SIMULATION RESULTS

This algorithm is tested with MATLAB package:-

A. Patterns of Arithmetic Operation: Four patterns of 3x3 pixels are shown in **Figure (2)**. In this test a neural net of (9) input neurons (they are equal to the length of input vector), (4) hidden neurons and (4) output neurons (they are equal to the number of patterns used in training set). The parameters of PSO are adjusted to bring the network out of its local minima. So C1=C2=2, r1=0.4, r2=0.9, and α =0.2 r_2 =0.2 and α =0.2, the inertial weight decrease from 0.9 to 0.2 linearly. The initial weights randomly generated between [-12,12] and biases between [-8,8] with maximum velocity [Vmax=18]. The Mean Square Error (MSE=10e-7) against the maximum number of iterations (epoch=1500) are shown in **Figure (3)** and **Figure (4)** illustrate the Accuracy performance of the PSO.

B. Patterns of English Characters: Ten patterns of 7x7 pixels are shown in **Figure (5)**. A neural network of (49) input neurons, (10) hidden neurons and (10) output neurons is used in this test). Before training process, the parameters of training algorithm (PSO) set to swarm size=25 particles, C1=C2=2, r1=0.8 and 2 and r2=0.2, the inertial weight decrease from o.9 to 0.2 linearly. The initial weights randomly generated between [-10, 10] and biases between [-9, 9] with maximum velocity [Vmax=20].The maximum number of iterations (epoch) =1500 and Mean Square Error (MSE=10e-7). It can be seen that, the best network for each test can be obtained in a small number of iteration (generation) with low number of epoch for training with PSO. **Figures (6)** show the curves of the MSE against the number of iterations of the training the optimal network. **Figure (7)** illustrate the accuracy performance of the PSO.

4. PSONN FOR OPTIMAL CONNECTIONS AND WEIGHTS

Optimization is used to refer to reducing structural complexity by minimizing the number of weights in a Feed Forward Artificial Neural Network (FFANN). Such optimization offers advantages in terms of simpler networks, faster training and better generalization ability to avoid over fitting due to over sized network. Optimization of FFANN structure includes the selection of appropriate inputs and outputs, and the determination of the number of connections between each neuron inside the network. In other words, the goal is to have a network with a low connectivity, both for speed of operation, and for a better ability to generalize. The goodness of an individual is determined by two parameters: the MSE and the total possible connections. The MSE can be scaled by the ratio of the actual connections (C_a) to total possible connections (C_t) [**Wang 2008**]. The fitness function which can be used in PSO is:

$$Fitnessfun\ ction = MSE \times \frac{C_a}{C_t} \tag{3}$$

It can be seen that equation3 is the best one to achieve good effect on the algorithm to get optimal connections and weights and reaches the lower value of MSE than the fully connected neural network with a very few number of iteration. **Figures (8) and (9)** show the neural networks for the two tests (A) and (B) respectively with optimum connections obtained from equation3.

5. CONCLUSIONS

The merits of PSO algorithm can be summarized as follows: Firstly, with parallel search strategy, it can locate the global optimization consistently. Secondly, its velocity – displacement search model is simple and easy to implement. Thirdly, few parameters should be considered and set up. Moreover, the information flow with single direction can absolutely speed up the convergence process. Finally, variable inertial weight can effectively guarantee the compromise between global search and local search.

By using PSONN for designing an optimal neural network, the optimal structure, including number of hidden layers, number of hidden neurons, is found. At the end of the process not only the best network structure for a particular application but also the trained network with little number of epochs is provided.

PSONN for finding optimum connections and weights uses PSO to reduce structural complexity by minimizing the number of connection weights in a FFANN. Tests show that by this algorithm, a simplex network, faster training with higher accuracy than the old network (full connected network) is obtained.

The equation (3) used to find the optimal pattern of connections and weights by scaling the MSE by the ratio of the actual connections (C_a) to total possible connections (C_t), therefore PSO is an optimization algorithm reduces the connections of ANN by 30%. It directs the search through the intelligence generated from cooperation and competition among the individuals.

6. REFERENCES

1. A. P. Engelbrecht, " Computational Intelligence: An Introduction", John Wiley & Sons Ltd, 2007.

2. C. Blum, D. Merkle,"***Swarm Intelligence***"Springer-Verlag Berlin Heidelberg, 2008.

3. G. K. Venayaga moorthy and R. G. Harley, "Swarm Intelligence for Transmission System

Control", IEEE1-4241298-6, 2007.

4. J.Kennedy and R.C Eberhart,"***Swarm Intelligence***" San Francisco: Morgan Kaufmann Publishers, **2001**.

5. K. Chandramouli and E.Izquierdo, "Image Classification using Chaotic Particle Swarm Optimization", IEEE 142440481, ICIP2006.

6. L. Wang, X. Wang, J. Fu and L. Zhen, "ANovel Probability Binary Partical Swarm Optimization Algorithm and its Application", Journal of software, Vol. 3, No.9, 2008.

7. M. Clerc, "***Particle Swarm Optimization***" ISTE Ltd, 2006.

8. T. Su, J. Jhang and C.Hou," ***A Hybrid Artificial Neural Network and Particle Swarm Optimization for Function Approximation***", International Journal of Innovative Computing, Information and Control, Volume 4, Number 9, September, 2008.

9. R. Mendes, P. Cortez, M. Rocha and J. Nevers,"***Particle Swarm for Feed forward Neural Network Training***", IEEE, June, 2002.

10. Y. Shi, "***Particle Swarm Optimization***", IEEE Neural Network Society, February, 2004.

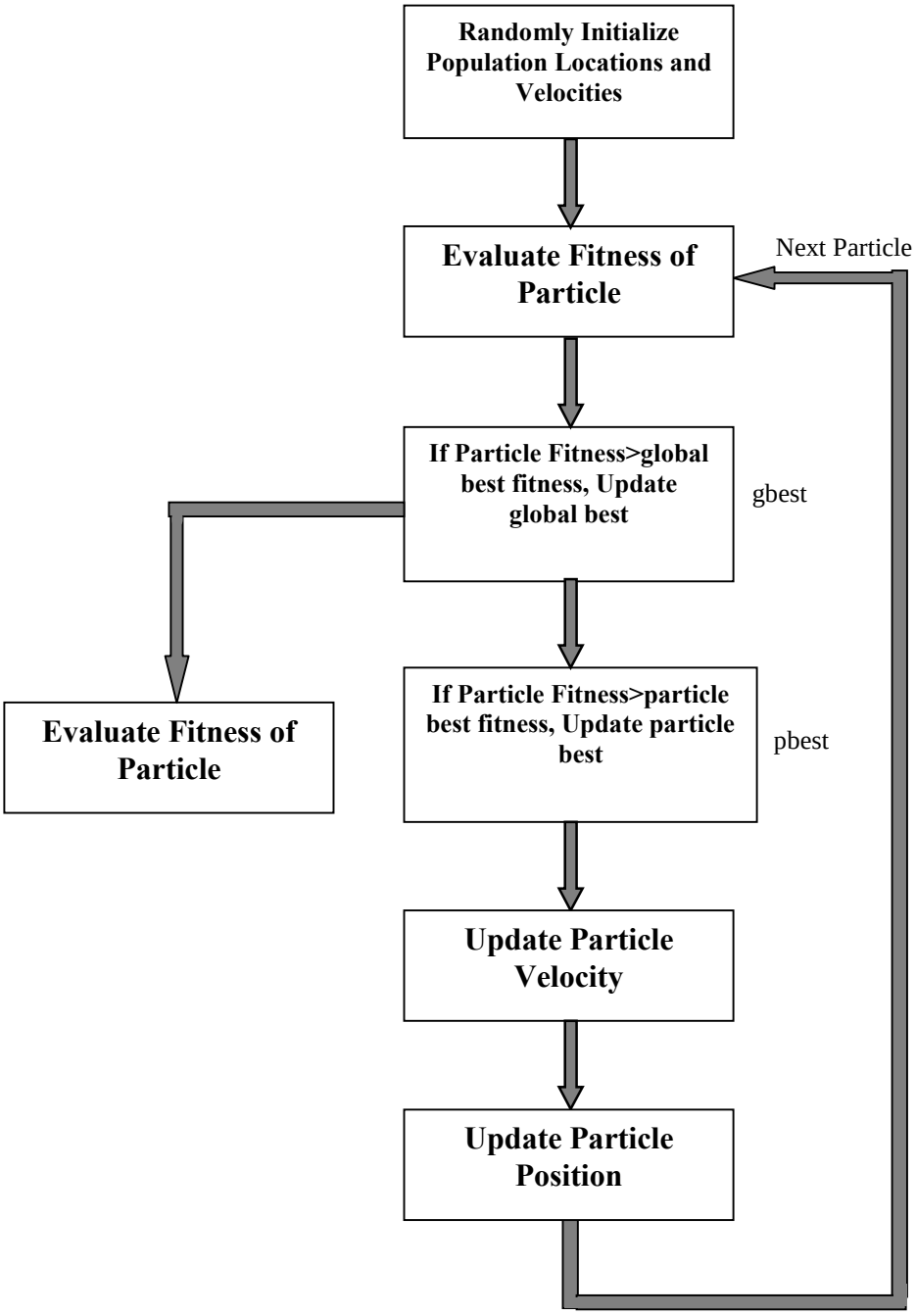
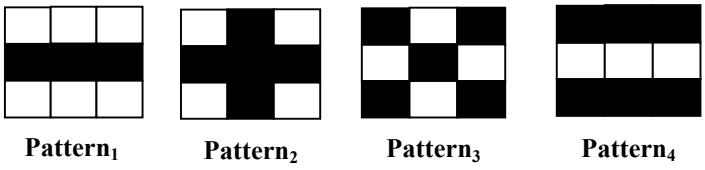


Figure (1) Basic PSO Procedures [Shi 2004].



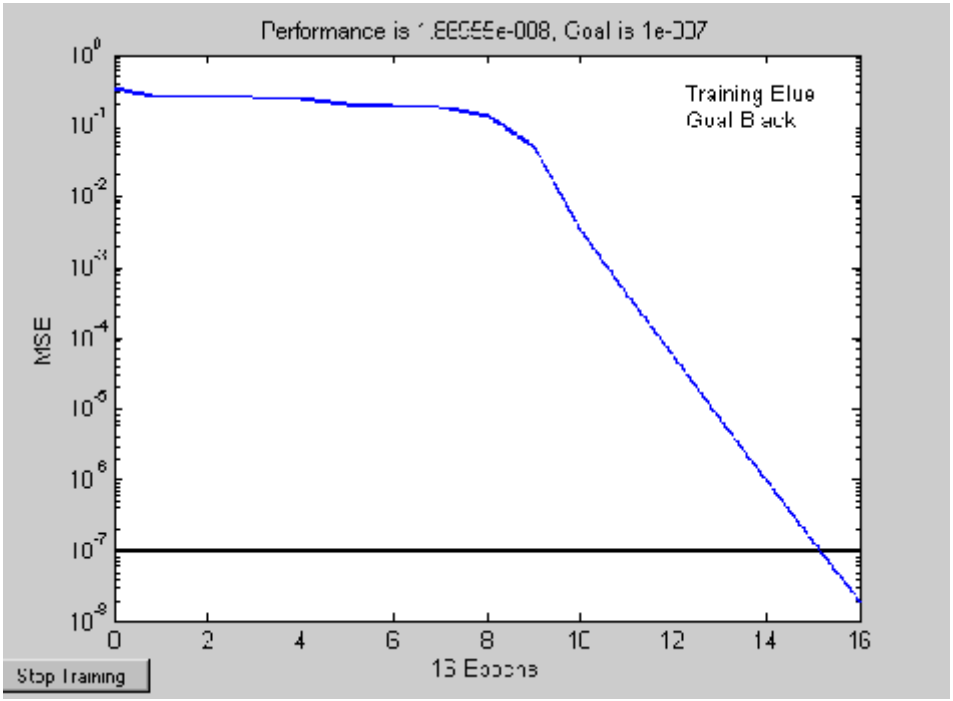
(a)

$$P = \begin{pmatrix} -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ -1 & -1 & 1 & 1 \end{pmatrix}$$

b

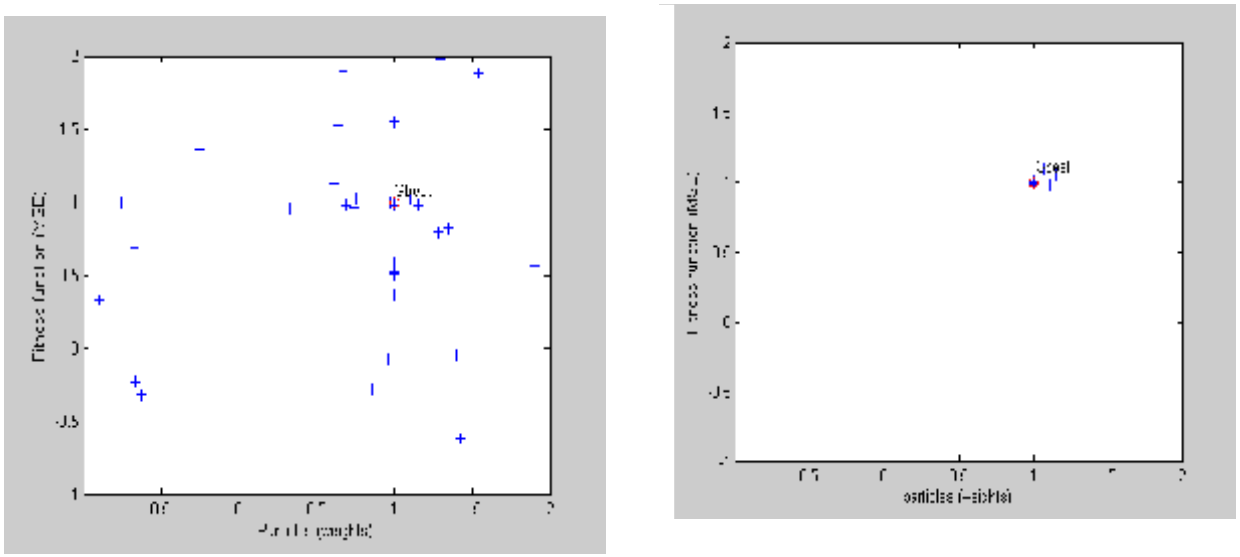
.

Figure (2) The test: a-the patterns, b-the patterns' matrix.



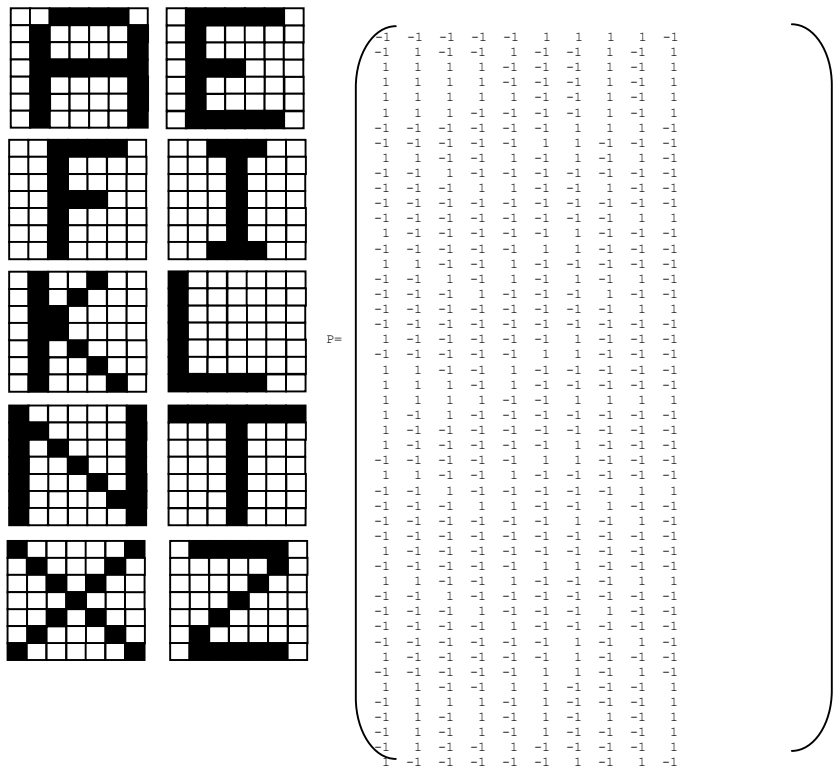
.

Figure (3): Training MSE with iterations for optimum network for test (A).



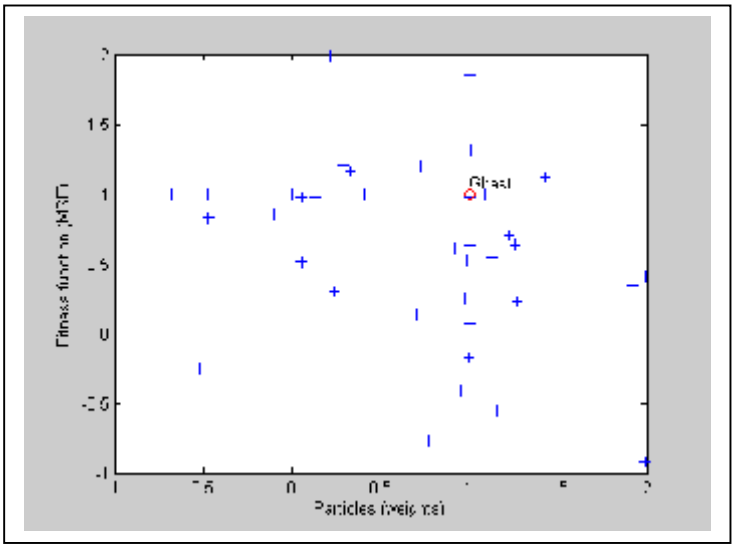
(a) (b)

Figure (4) Accuracy performance PSO (a) After 5 iterations (b) After 1500 iterations

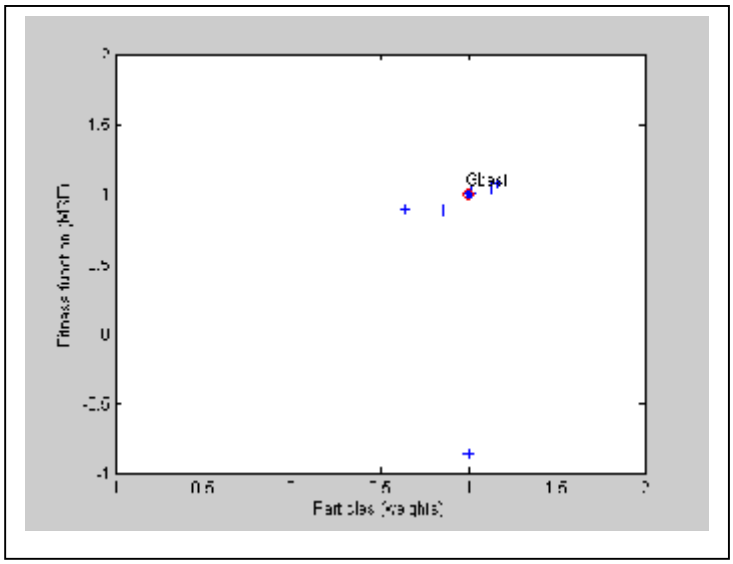


(a) **(b)**

Figure (5) the test patterns: (a) The patterns, (b) the patterns' matrix.



(a)



(b)

Figure (7) Accuracy performance of PSO

a- After 5 iterations

b- After 1500 iterations

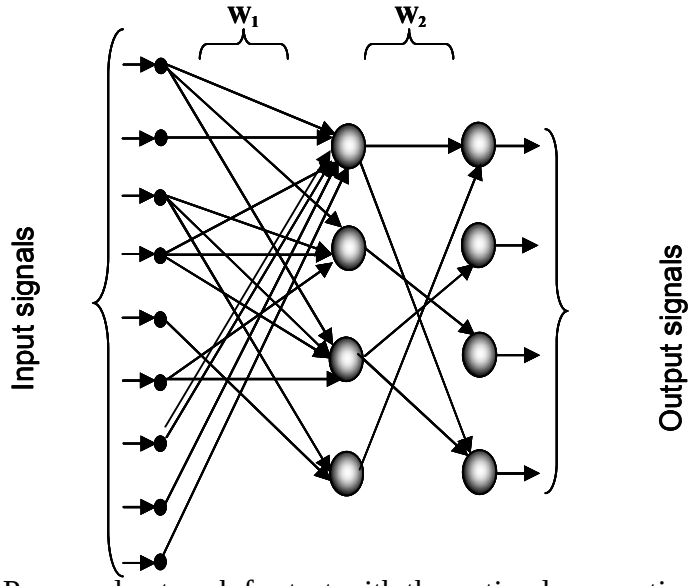


Figure (8) The Proposed network for test with the optimal connections by using PSO.

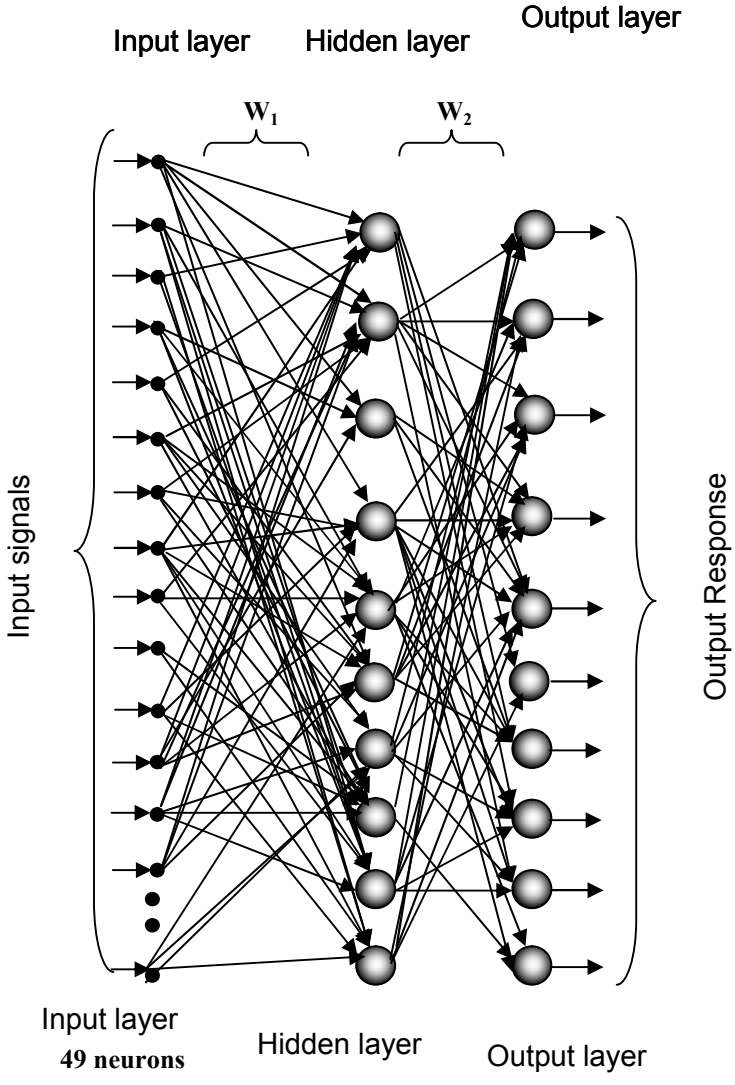


Figure (9) the proposed ANN for test with the optimal connections by using PSO.