A Hybrid of Artificial Fish Swarm Algorithm and Particle Swarm Optimization for Feedforward Neural Network Training

Huadong Chen[1] Shuzong Wang[1] Jingxi Li[1] Yunfan Li[2]

[1] Research Inst. of New Weaponry Technology & Application, Naval Univ. of Engineering, Wuhan 430033, P. R. China

[2] Commanding Communications Academy, Wuhan 430010, P. R. China

Abstract

A hybrid of artificial fish swarm algorithm (AFSA) and particle swarm optimization (PSO) is used to training feedforward neural network. After the two algorithms are introduced respectively, the hybrid algorithm based on the two is expressed. The hybrid not only has the artificial fish behaviors of swarm and follow, but also takes advantage of the information of the particle. An experiment with a function approximation is simulated, which proves that the hybrid is more effective than AFSA and PSO.

Keywords: Artificial fish-swarm algorithm, Particle swarm optimization, Artificial neural networks

1. Introduction

Artificial neural network (ANN) has been acknowledged as an intelligent universal mechanism of dealing with function approximation, pattern recognition, process estimation and prediction, optimization design and other applications. Many applications of ANN are dependent on the effective training of network weights and biases, so, ANN training is very important to the realization of ANN function. The research on training algorithms of ANN is an interesting field and faster algorithms are being developed. In general, backpropagation is an ordinary method used for ANN training [1]. As intelligence algorithms are heuristic and stochastic, and they are less likely to get stuck in local minimum, many those algorithms, such as simulated annealing [2] and genetic algorithm [3] have been proposed to training ANN.

With the development of swarm intelligence algorithms such as ant colony optimization (ACO) [4], Particle swarm optimization (PSO) [5] and Artificial fish swarm algorithm (AFSA) [6], it developed largely the training of ANN. Considering the application of swarm intelligence algorithms to train ANN in above references, a hybrid of PSO and AFSA to train multilayer feedforward neural network (MFNN) is proposed in this paper, which can solve the problem effectively.

2. Particle swarm optimization

PSO is a relatively recent heuristic search method whose mechanics are inspired by the swarming or collaborative behavior of biological populations. PSO moves from a set of points to another set of points in a single iteration with likely improvement using a combination of deterministic and probabilistic rules. The PSO and its many versions have been popular in academia and industry, mainly because of its intuitiveness, ease of implementation, and the ability to effectively solve highly nonlinear, mixed integer optimization problems that are typical of complex engineering systems.

The basic PSO algorithm can be consists of two steps. The first is to generate particles' positions and velocities; the second is to update velocities and positions. As the first is relatively easy, it is elided here. The second which is the kernel of PSO will be detailed introduced in the following [7].

The second step is to update the velocity and position of all particles at generation *t+1* according to the particles information at generation *t*. Firstly, the velocity of Particle *j* is updated as expressed in Formula 1.

$$V = \omega V_t^j + c_1(P_t^j - X_t^j) + c_2(P_t^{gen} - X_t^j) \ . \qquad \textbf{(1)}$$

Where, V_t^j is the velocity of Particle *j* at generation *t*; X_t^j is the position of Particle *j* at generation *t*; P_t^{gen} is the best position in the current swarm over generation *t* determined by the fitness or objective function value; P_t^j is the best position of Particle *j* over generation *t* determined by the fitness or objective function value; ω is inertia factor, which is often the number between 0 and 1; c_1 is self

confidence factor, which is often the random number between 0 and 2; c_2 is swarm confidence factor, which is often the random number between 0 and 2.

After velocity update, the position of Particle j can be computed by Formula 2.

$$X = X_t^j + V_{t+1}^j \ . \quad (2)$$

The velocity update formula includes some random parameters, c_1 and c_2, which can ensure good coverage of the design space and avoid get into local optima. The basic PSO is executed by above update generation by generation, when the optimal value is satisfied it can stop.

3. Artificial fish swarm algorithm

Artificial fish swarm algorithm (AFSA) is a novel method to search global optimum, which is typical application of behaviorism in artificial intelligence. It is a random search algorithm based on simulating fish swarm behaviors which contains searching behavior, swarming behavior and chasing behavior. It constructs the simple behaviors of artificial fish (AF) firstly, and then makes the global optimum appear finally based on animal individuals' local searching behaviors. The AFSA can search global optimum effectively and has a certain adaptive ability for searching space. Before constructing the artificial fish (AF) model, we introduce some definitions first [7], [8].

X_i is the position of AF *i*; $y = f(X)$ is the fitness or objective function at Position *X*, which can represent food concentration; $d_{ij} = \|X_i - X_j\|$ represents the distance between the AF *i* and *j*; *Visual* and δ represent the visual distance and crowd factor of the AF respectively; *nf* is the number of its fellow within the *Visual*; *Mstep* is the maximum step of the AF moving; *step* is a random positive number within *Mstep*; $S = \{X_j \mid \|X_i - X_j\| < Visual\}$ is the set of AF *i* exploring area at the present position. The typical behavior of AF is expressed as followings:

3.1. Behavior of searching food

In general, the fish stroll at random. When the fish discover a water area with more food, they will go quickly toward the area. Let us assume that X_i is the AF state at present, and $X_j \in S$. The behavior of follow can be expressed as the following:

$$prey(X_i) = \begin{cases} X_i + step \dfrac{X_j - X_i}{\|X_j - X_i\|} & if \quad y_j > y_i \\ X_i + step & else \end{cases} . \quad (3)$$

3.2. Behavior of swarm

In the process of swimming, the fish will swarm spontaneously in order to share the food of the swarm. Let us assume that X_i is the AF state at present, and $X_c = \sum_{X_j \in S} X_j / nf$. The behavior of swarm to AF *i* can be expressed in Formula 4.

$$swarm(X_i) = \begin{cases} X_i + step \dfrac{X_c - X_i}{\|X_c - X_i\|} & if \quad \dfrac{y_c}{nf} > \delta y_i \\ prey(X_i) & else \end{cases} . \quad (4)$$

3.3. Behaviors of Agents

When one fish of the fish swarm discovers more food, the other fish will share with it. Let us assume that X_i is the AF state at present, and $y_{\max} = \max\{f(X_j) \mid X_j \in S\}$. The behavior of follow can be expressed in Formula 5.

$$follow(X_i) = \begin{cases} X_i + step \dfrac{X_{\max} - X_i}{\|X_{\max} - X_i\|} & if \quad \dfrac{y_{\max}}{nf} > \delta y_i \\ prey(X_i) & else \end{cases} . \quad (5)$$

According to the character of the problem, the AF evaluates the environment at present, and then selects an appropriate behavior. For example, behaviors of follow and swarm are both simulated, the better of improved its state will be executes. This process indicates the flexibility of AFSA.

4. Numerical Experiments

MFNN is one of ANN models most widely used at present. The typical performance function that is used for training MFNN is the mean sum of squares of the network errors [1]. In this paper, we take three layers feedforward neural networks for an example. The output layer is linear and the nonlinear action function of hidden layer nerve cell is hyperbolical tangential function represented by tansig(*x*). The algorithm is provided with a set of examples of proper network behavior:

$\{P_1, Q_1\}, \{P_2, Q_2\}, \cdots, \{P_i, Q_i\}, \cdots, \{P_n, Q_n\}$,

Where, P_i is a vector input to network, and Q_i is the corresponding target output. As each input is applied to the network, the output is compared to the target. The algorithm should adjust the network parameters (weights and biases) in order to minimize the mean square error:

$$\begin{cases} F(W^1, W^2, B^1, B^2) = E[(Q-Y)^T (Q-Y)] \\ Y = W^2 \times \text{tansig}(W^1 \times P + B^1) + B^2 \end{cases} . \quad (6)$$

Where, W^1 and W^2 are the network weights of the hidden layer and the output layer respectively; B^1 and

B^2 are the network biases of the hidden layer and the output layer respectively; Y is the output of network. The MFNN training process is to get the minimal value of $F(W^1, W^2, B^1, B^2)$ by adjusting the connecting weights W^1 and W^2, and biases B^1 and B^2 of nerve cells.

By comparison of PSO and AFSA, we can find that their methods to search the optimal solution are different: PSO is to search the optimal solution by the best position of particles and swarm in call-board; AF is to search the optimal solution by information of AFs within the visual of the AF. According to above difference, considering the character of each algorithm, a new hybrid algorithm is introduced as followings.

The hybrid of the two is implemented by *nf*. When *nf*>*nf0* (*nf0* is a limitative number of the AF), there are enough information around the AF, then the AFSA can be executed. When *nf* <*nf0*, there are not enough information around the AF, then the PSO can be executed. PSO can search the best position by the best position of particles and swarm in call-board.

5. Experiment and result

In order to prove the generalization of HAP, here it is used for training of ANN with a function approximation. HAP is compared with the standard PSO and AFSA, to perform a comparison in terms of convergence and final obtained result. In the experiment, the function is $t = \sin(\pi \times p)$, where, $p \in [-1,1]$; the structure of three layer feedforward neural networks is 1-4-1 [1]. The population (Particle or AF) number of each algorithm is 50, and the generation is 500. Parameters of AFSA are defined as follows: *visual* = 6.2, δ =0.01, *Mstep*=1. Besides above parameters, Parameter *nf0* of HAP is defined as follows: *nf0*=7. As the three algorithms all have stochastic performance, 20 computations are simulated in order to compare the different optimization approaches for training the ANN. The statistic performance comparisons of the three algorithms are also shown in Table 1 and Fig.1.

Algorithm	MSE(Ave)	MSE(Best)	CPU Time(s)
PSO	0.2407	0.0625	4.06
AFSA	0.2447	0.0383	7.41
HAP	0.1991	0.0243	6.34

Table 1**:** Mean squared error value and CPU time of HAP, PSO and AFSA.

The error values computed by HAP are better than those gotten by PSO and AFSA in Fig. 1. We can conclude that the HAP convergence rate and stability is better than the PSO and AFSA from Fig. 1.

From the Table 1, we can conclude that the PSO is the most quickly when computing but his results are the worst, and that HAP has the best results with less CPU time.

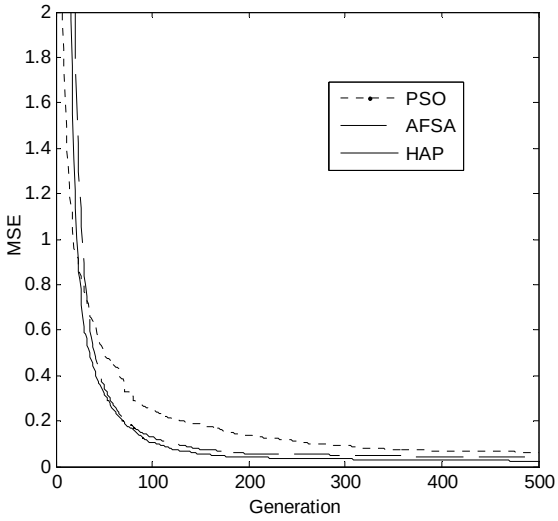

Fig. 1: Shows the mean square error (MSE) of HAP AFSA and PSO versus the generation**.**

6. Conclusions

In this paper, MFNN training by a new algorithm, HAP, is proposed. HAP is a hybrid of PSO and AFSA, it has the advantages of the two at the same time. When the information of the AFSA is enough, AFSA will be executed, otherwise, the PSO will be executed. With the above performance, HAP is a good algorithm to training MFNN. To demonstrate the performance of HAP, designs of function approximation with three-layer ANN are simulated. The results in Table 1 and Fig. 1 demonstrate that the HAP has better global astringency and stability than AFSA and PSO.

References

[1] M.T. Hagan, H.B. Demuth, and M. Beale, *Neural Network Design*, PWS Publishing Company, Boston, 1996.

[2] T. Tambouratzis, A simulated annealing artificial neural network implementation of the n-queens problem. *Int. Joint of Intelligent Systems*, pp. 739-751, 1997.

[3] V. Petridis, S. Kazarlis, and A. Papaikonomou, A genetic algorithm for training recurrent neural network. *Int. Joint Conf. On Neural Network*, pp. 2706-2709, 1993.

[4] J.B. Li and Y.K. Chung, A novel back-propagation neural network training algorithm

designed by an ant colony optimization. *IEEE/PES Transmission and Distribution Conference & Exhibition*, pp. 1-5, 2005.

[5] R. Mendes, P. Cortez, M. Rocha, and J. Neves, Particle swarms for feedforward neural network training. *Proc. Int. Joint Conf. Neural Networks*, pp. 1895–1899, 2002.

[6] C. R. Wang, C.L. Zhou and J. W. Ma, An improved artificial fish-swarm algorithm and its application in feedforward neural networks. *Proc. of the Fourth Int. Conf. on Machine Learning and Cybernetics*, pp. 2890-2894, 2005.

[7] S. Gao and J.Y. Yang, *Swarm intelligence algorithms and applications*. China Waterpower Press, Beijing, 2006.

[8] X.L. Li, Z.J. Shao and J.X. Qian, An optimizing method based on autonomous animate: fish-swarm algorithm, *System Engineering Theory and Practice*, 11: 32-38, 2002.