Training Tsukamoto-Type Neural Fuzzy Inference Network Based on Cat Swarm Optimization

Meysam Orouskhani[*1], Mohammad Mansouri[2], Mohammad Teshnehlab[3]

[1] Ph. D. student, Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran, orouskhani@gmail.com

[2] Ph. D. student, Intelligent System Laboratory, Electrical and Computer Engineering Department, K.n.Toosi University, Tehran, Iran, mohammad.mansouri@ee.kntu.ac.ir

[3] Academic member, Intelligent System Laboratory, Electrical and Computer Engineering Department, K.n.Toosi University, Tehran, Iran, teshnehlab@eetd.kntu.ac.ir

Abstract: This paper introduces a new approach for tuning the parameters of the Tsukamoto-type neural fuzzy inference network (TNFIN). In the standard method, the antecedent and consequent parameters of TNFIN are trained by a hybrid learning algorithm include of least square estimation and the gradient descent. In this study, a swarm-based meta-heuristic optimization algorithm, so-called "Cat Swarm Optimization" is used to tune the parameters. Experimental results for prediction of Mackey-Glass model and identification of a nonlinear dynamic system indicate that the performance of the proposed algorithm in comparison with standard method is much better and it shows quite satisfactory results.

Keywords: Cat Swarm Optimization, Tsukamoto Fuzzy Model, Prediction and Identification

1- Introduction

Fuzzy-neural systems with capabilities of a neural network and the fuzzy inference system are most applicable intelligent networks [4]. Fuzzy reasoning is developed to implement fuzzy implication relations. Tsukamoto-type neural fuzzy inference network (TNFIN) is a five-layer feed forward neural fuzzy network which uses Tsukamoto-type fuzzy reasoning. How to train the parameters of TNFIN in order to enhance accuracy in the special application is an important problem. In original method, TNFIN employs least square estimation (LSE) algorithm and gradient descent (GD) method to train the network [10]. Both antecedent and consequent parameters of TNFIN are nonlinear. Therefore, finding an optimal learning step size and calculation of gradient in each step are so difficult that cause a slow convergence. On the other hand, using stochastic leaning algorithms which are based on probability can search the solution space globally without trapping in local minima. Evolutionary computation algorithms which are based on population using different stochastic techniques and in comparison with former method can find the better solution. Evolutionary algorithms and swarm intelligence are two main parts of evolutionary computation [14], which simulate the evolutional stages of creatures and swarm behavior of creatures, respectively. In this paper, in order to train the parameters of TNFIN, one of the recent swarm intelligence algorithms is applied. Cat swarm optimization (CSO) is a swarm intelligence algorithm that simulates the cat's behavior in two modes: seeking and tracing mode [13]. So by taking advantages of CSO, there is no need to calculate the gradient and learning rate and it makes a fast convergence and less error. Experimental results for prediction of Mackey-Glass model and identification of a nonlinear system show that proposed learning method has much better performance. The rest of the article is organized as follows: in section2 the structure of TNFIN and traditional learning algorithm are reviewed. Cat swarm optimization is introduced in section3 and the proposed method for training the parameters of TNFIN is introduced in section4. In section5, experimental results for applications of prediction and identification are showed. Finally, a conclusion is drawn in the last section.

2- Tsukamoto-Type Neural Fuzzy Inference Network (TNFIN)

Both neural networks and fuzzy logic are model-free estimators and share the common ability to deal with the uncertainties and noise [3-11]. Also both of them encode the information in parallel and distribute architectures in a numerical framework. Hence, it is possible to convert fuzzy logic architecture to a neural network and vice versa. This makes it possible to combine the advantages of neural networks and fuzzy logic [4]. A network obtained in this way could use powerful training algorithms that neural networks have at their disposal, to obtain the required parameters not available in the fuzzy logic architecture. Moreover, the network obtained in this way would not remain a black box, because it would have fuzzy logic interpretation capabilities in terms of the linguistic variables [6]. TNFIN combines the two approaches, neural networks and fuzzy systems. Tsukamoto-type neural fuzzy network has five layers with the same type of nodes [10].

Five layers of TNFIN are explained as follow: the first layer is the fuzzification layer. This layer executes a fuzzification process and makes fuzzy outputs. Fuzzification process will be done by membership functions (MFs) as follows:

$$\mu A_i(x) = \frac{1}{1+\left(\frac{x-c_i}{a_i}\right)^2} \tag{1}$$

Here, c_i and a_i are known as primary nonlinear parameters. The second layer executes the fuzzy AND of the antecedent part of the fuzzy rules and the output of this layer multiplies the input signals. Each neuron in this layer calculates the related firing strength.

$$O_{2,i} = w_i = \mu \mathrm{A}_i(\mathrm{x}) \times \mu \mathrm{B}_i(y), i = 1,2 \tag{2}$$

The third layer normalizes the membership functions and neurons in this layer calculate the related normalized firing strength which shows the effect of a rule in final result.

$$O_{3,i} = \overline{w_i} = \frac{w_i}{w_1 + w_2}, i = 1,2 \tag{3}$$

The fourth layer executes the conclusion part of the fuzzy rules. This layer uses Tsukamoto fuzzy reasoning for defuzzification process. In this layer, m and s are called consequent parameters and are used to adjust the shape of membership function of the consequent part. The output of this layer is:

$$O^4_k = O^3_k \times y_k = \left\{\begin{matrix} O^3_k(\mathrm{c}_k - \mathrm{d}_k\sqrt{\frac{1}{O^3_k}} - 1) \to k = odd \\ O^3_k(\mathrm{c}_k + \mathrm{d}_k\sqrt{\frac{1}{O^3_k}} - 1) \to k = even \end{matrix}\right\} \tag{4}$$

Layer 5 computes the output of the fuzzy system by summing up the outputs of the fourth layer which is the defuzzification process.

$$O^5 = \sum_k O^4_k \tag{5}$$

The general learning procedure in neural fuzzy system consists of modifying their parameters by presenting them with the input data and desired output data. Typically, by adjusting the parameters of fuzzy membership functions and the weights of connections between different layers, a certain performance index is optimized. Here the goal is to minimize the following error function E.

$$E = \frac{1}{2} \times \sum_{p=1}^{n} (\mathrm{T(p)} - \mathrm{O}^5(\mathrm{p}))^2 = \sum_{p=1}^{n} E_p \tag{6}$$

Where $O^5(p)$ is the network output for the P_{th} training point, T(p) is the corresponding targeting output, and n is the total number of the points in the training data set. The consequent parameters are identified in the forward pass by means of the LSE method and the antecedent parameters are modified in the backward pass using the GD method.

3- Cat Swarm Optimization

There are many algorithms to find the global solutions of the problems. Some of these optimization algorithms are developed based on swarm intelligence. These algorithms imitate the creature's swarm behavior and model into algorithm, such as Ant colony optimization which imitates the behavior of ants [5-12], Particle swarm optimization which imitates the behavior of birds [9], Bee colony optimization which imitates the behavior of bees [1] and the recent finding, Cat swarm optimization which imitates the behavior of cats [13]. Cat swarm optimization (CSO) is a new optimization algorithm in the field of swarm intelligence [13]. The CSO algorithm models the behavior of cats into two modes: 'Seeking mode' and 'Tracing mode'. In CSO, every cat has its own position composed of d dimensions, velocities for each dimension, a fitness value, which represents the accommodation of the cat to the fitness function, and a flag to identify whether the cat is in seeking mode or tracing mode. The final solution would be the best position of one of the cats. The CSO keeps the best solution until it reaches the end of the iterations. Cat swarm optimization algorithm has two modes in order to solve the problems which are described below:

For modeling the behavior of cats in resting time and being-alert, algorithm uses the seeking mode. This mode has four main parameters which are mentioned as follow: seeking memory pool (SMP), seeking range of the selected dimension (SRD), counts of dimension to change (CDC) and self-position consideration (SPC). The process of seeking mode is describes as follows [13]:

Step1: Make j copies of the present position of cat_k, where j = SMP. If the value of SPC is true, let j = (SMP-1), then retain the present position as one of the candidates.

Step2: For each copy, according to CDC, randomly plus or minus SRD percent the present values and replace the old ones.

Step3: Calculate the fitness values (FS) of all candidate points.

Step4: Calculate the selecting probability of each candidate point by Equation (7);

Step5: Randomly pick the point to move to from the candidate points, and replace the position of cat_k.

$$P_i = \frac{|FS_i - FS_{\max}|}{FS_{\max} - FS_{\min}} \tag{7}$$

Tracing mode is the second mode of algorithm. In this mode, cats desire to trace targets and foods. The process of tracing mode can be described as follows [13]:

Step1: Update the velocities for every dimension according to Equation (8).

Step2: Check if the velocities are in the range of maximum velocity.

Step 3: Update the position of cat_k according to Equation (9).

$$V_{k,d} = V_{k,d} + r_1 \times c_1 \times (X_{best,d} - X_{k,d}) \tag{8}$$

$$X_{k,d} = X_{k,d} + V_{k,d} \tag{9}$$

$X_{best,d}$ is the position of the cat, who has the best fitness value, X_k,d is the position of cat_k, c_1 is an acceleration coefficient for extending the velocity of the cat to move in the solution space and usually is equal to 2.05 and r_1 is a random value uniformly generated in the range of [0,1]. In this paper for updating the velocities of cats, an adaptive inertia weight is added. Adaptive inertia weight and new velocity update equations are expressed in equations (10), (11) where 'w_s' is the first inertia weight, 'i' is the current iteration, 'i_{max}' is the maximum iteration [8].

$$W(\mathrm{i}) = \mathrm{W}_s + \frac{(\mathrm{i}_{max} - \mathrm{i})}{2 \times i_{max}} \tag{10}$$

$$V_{k,d} = W(\mathrm{i}) \times V_{k,d} + r_1 \times c_1 \times (X_{best,d} - X_{k,d}) \tag{11}$$

4- Learning Phase

In this section, the new learning method for training TNFIN is explained. The new algorithm uses CSO for training the nonlinear parameters. Antecedent parameters: {'a', 'b' and 'c'} and consequent parameters: {'m' and 's'} are nonlinear parameters of TNFIN in layer2 and layer4 respectively and are trained by CSO. In this study, 'b' is equal to '1'. The goal of using learning algorithm is minimize the error function which is indicated in Equation (12), where 'y' is desire output (O^5) and 'y_{out}' is the computed output.

$$Error = \frac{1}{2} \times (\mathrm{y} - \mathrm{y}_{out})^2 \tag{12}$$

In the simulations of this paper, TNFIN uses two membership functions for each input. Table1 describes the parameters which are trained by new learning algorithm the values of the parameters of CSO are shown in Table2.

Type	Parameter	No of Parameters
Antecedent	c	Neurons Number × Input Dimension
Antecedent	a	Input Dimension
Consequent	m, s	Rules Number

Table 1. Training parameters in the proposed learning algorithm

Parameter	Value
SMP, SRD, CDC, MR	3, 0.1,1, 0.5
Population Size	40

Table 2. Parameters of CSO

5- Experimental Results

Example1: Predicting a chaotic system; the main goal of this simulation is predict future values of a chaotic time series [2-7].

$$\dot{x}(t) = \frac{0.2 \times x(t-\tau)}{1 + x^{10}(t-\tau)} - 0.1 \times x(t) \tag{13}$$

Predicting the output by using the information of inputs is the goal of problem where inputs and outputs are chosen as follow:

Inputs = [x(t-18), x(t-12), x(t-6), x(t)], Output =[x(t+6)]

Table3 shows the average error in predicting the outputs of Mackey-glass model by using proposed and standard method. It indicates that the proposed method in comparison with the former method has much better performance. Also, figure1 shows the mean square error for this simulation.

Example2: Identification of nonlinear system; in this example the nonlinear plant with multiple time-delays is described [9].

$$y_p(t+1) = f\,(y_p(t), y_p(t-1), y_p(t-2), u_p(t), u_p(t-1)) \tag{14}$$

$$f\,(x_1, x_2, x_3, x_4, x_5) = \frac{x_1 \times x_2 \times x_3 \times x_5 \times (x_3 - 1) + x_4}{1 + {x_2}^2 + {x_3}^2} \tag{15}$$

Here, by applying CSO as a training algorithm, TNFIN is used to identify and model the nonlinear dynamic system and experimental results are shown in Table4 and figure1. Table4 shows the experimental results of applying the proposed method to minimize the mean square error (MSE) of the system identification. It is clear that the minimum error of new method is less than standard method.

Learning Algorithm	MSE
Standard Method	0.1658
Cat Swarm Optimization	0.009961

Table 3. Experimental results for identification of the Mackey-glass plant

Learning Algorithm	MSE
Standard Method	0.2
Cat Swarm Optimization	0.03954

Table 4. Experimental results for identification of the nonlinear dynamic system

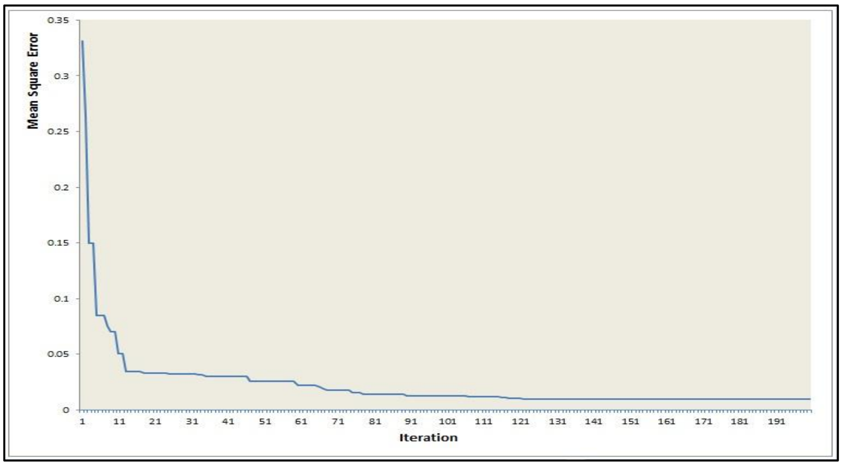

Figure 1. MSE for prediction of the Mackey-Glass plant (Example1)

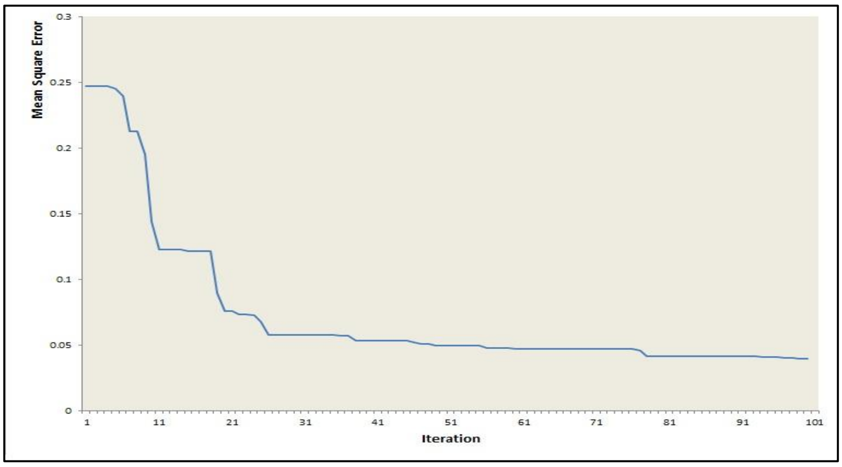

Figure 2. MSE for identification of nonlinear dynamic system (Example2)

Conclusion

In this paper, a new learning method for training the parameters of TNFIN is proposed. This paper used the cat swarm optimization, a swarm-based optimization algorithm, to tune the antecedent and consequent parameters of the TNFIN. The experimental results of the simulation for predicting a chaotic system and identification of a nonlinear dynamic system reveal that the new approach achieves better results in comparison with standard method. Structure of the proposed method and simultaneous local and global search abilities implicate that CSO get the better solutions and can escape from local minimums of the problems and regardless of the runtime, the proposed method gets better performance with less error.

References

[1] D. Karaboga, "An idea based on Honey Bee swarm for numerical optimization", *Technical Report-TR06*, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[2] J.S.R. Jang, "ANFIS: adaptive-network-based fuzzy inference system*"*, *IEEE Trans. Systems Man and Cybernet*, 1993.

[3] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical system using neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp 4–27, 1990.

[4] M. Aliyari Shoorehdeli, M. Teshnehlab, A. Khaki Sedigh, "Training ANFIS as an identifier with intelligent hybrid stable learning algorithm based on particle swarm optimization and extended Kalman filter", *ELSEVIER, Fuzzy Sets and Systems,* 2009.

[5] M. Dorigo, L.M. Gambardella, “Ant colony system: a cooperative learning approach to the traveling salesman problem”, *IEEE Trans. on Evolutionary Computation*. 26 (1), pp 53-66, 1997.

[6] M. Kumar, D.P. Garg, “Intelligent learning of fuzzy logic controllers via neural network and genetic algorithm”, in: *Proc. Japan—USA Symp on Flexible Automation*, 2004.

[7] M.C. Mackey, L. Glass, “Oscillation and chaos in physical control system”, *Science 197*,pp. 287–289, 1997.

[8] M.Orouskhani, M. Mansouri, and M. Teshnehlab, “Average-Inertia weighted Cat swarm optimization*,” LNCS, Berlin Heidelberg: Springer-Verlag*, pp 321– 328, 2011.

[9] R. Eberhart, J. Kennedy, “A new optimizer using particle swarm theory*”, Sixth International Symposium on Micro Machine and Human Science*, pp 39-43, 1995.

[10] R. Shoureshi, Zhi. Hu, “Tsukamoto-type Neural Fuzzy Inference Network”, *Proceeding of American Control Conference*, IEEE, 2000.

[11] R.R. Yager, L.A. Zadeh, “Fuzzy Sets, Neural Networks and Soft Computing”, *Thomson Learning*, 1994.

[12] S.C. Chu, J.F. Roddick, J.S. Pan, “Ant colony system with communication strategies”, *Information Sciences 167*, pp 63-76, 2004.

[13] S.C.Chu, P.W.Tsai and J.S.Pan, “Cat Swarm Optimization,” *LNAI 4099, Berlin Heidelberg: Springer-Verlag*, pp. 854–858, 2006.

[14] T. Weise, *Global Optimization Algorithms, Theory and applications*, Electronic Book, 2009.