World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:7, No:10, 2013

Particle Swarm Optimization with Interval-valued Genotypes and Its Application to Neuroevolution

Hidehiko Okada

***Abstract*—**The author proposes an extension of particle swarm optimization (PSO) for solving interval-valued optimization problems and applies the extended PSO to evolutionary training of neural networks (NNs) with interval weights. In the proposed PSO, values in the genotypes are not real numbers but intervals. Experimental results show that interval-valued NNs trained by the proposed method could well approximate hidden target functions despite the fact that no training data was explicitly provided.

***Keywords*—**Evolutionary algorithms, swarm intelligence, particle swarm optimization, neural network, interval arithmetic.

I. Introduction

A multi-layered feed forward neural network (NN) with interval-valued weights and biases was proposed in literature [1]. A supervised learning method for the interval NN (INN) was also proposed [1] as an extension of the traditional back propagation. The INN approximately models an interval function $Y = F(\boldsymbol{x})$, where Y is an interval and $\boldsymbol{x}$ is a real vector, by learning data $(x_q, Y_q), q = 1,2, \ldots$. The INN can learn the data in which Y_q include both of real values and interval values, because a real value can be specified as an interval value with zero width (i.e., with the same value of upper and lower limits). As the learning method for the INN, the supervised method was proposed but no unsupervised one has been proposed.

Besides, evolutionary algorithms (EAs) have recently been applied to the unsupervised learning of NNs, known as neuroevolution (NE) [2]-[5]. In NE, weights and biases are tuned by evolutionary operations, not by the back propagation (BP) algorithm. Because NE does not utilize BP, NE does not require errors between NN output values and their target signals. Thus, NE is applicable for problems in which the error function is hard to be determined. EAs have been applied to NE of traditional NNs with real-valued weights and biases, where the genotypes (chromosomes) consist of real numbers or bit strings that encode real numbers.

The author previously proposed extensions of EAs for handling fuzzy-valued genotypes [6]. In this paper, the author applies an instance of the proposed EAs, interval-valued particle swarm optimization (IPSO), to NE of INNs.

Hidehiko Okada is with Department of Intelligent Systems, Faculty of Computer Science and Engineering, Kyoto Sangyo University, Japan (e-mail: hidehiko@cc.kyoto-su.ac.jp).

II. Neural Networks with Interval Weights and Biases

The INN employed in this research is the same as in the literature [1], which is a three-layered feed forward NN with interval weights and biases. Fig. 1 shows its structure. An INN receives an input real vector $\boldsymbol{x}$ and calculates its output interval value O (for the sake of simplicity, the output layer includes a single unit) as follows [1]:

Input layer:

$$o_i = x_i \tag{1}$$

Hidden layer:

$$Net_j = \Sigma_i W_{j,i} o_i + \Theta_j \tag{2}$$

$$O_j = f(Net_j) \tag{3}$$

Output layer:

$$Net = \Sigma_j W_j O_j + \Theta \tag{4}$$

$$O = f(Net) \tag{5}$$

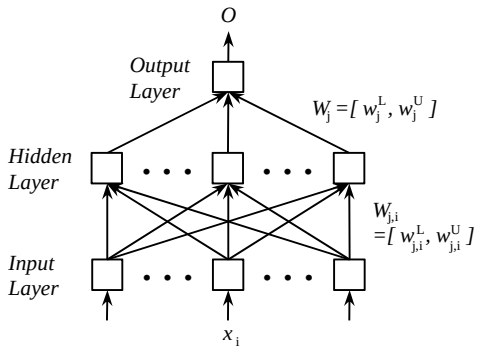

Fig. 1 Neural network with interval weights and biases [1]

In (1)-(5), x_i and o_i are real numbers, while Net_j, Net, $W_{j,i}$, W_j, Θ_j, Θ, O_j and O are intervals. $f(x)$ is the unit activation function which is typically the sigmoidal one: $f(x) = 1/(1 + e^{-x})$. The feed-forward calculation of the INN is based on the interval arithmetic [7] (for more detail, see the literature [1]). $f(x)$ maps an interval input to an interval output as illustrated in Fig. 2.

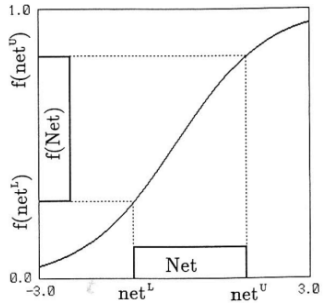

Fig. 2 Input-output relation of each unit in the hidden/output layers [1]

The FNN includes $mn + m$ weights (i.e., mn weights between n input units and m hidden units, and m weights between m hidden units and an output unit) and $m + 1$ biases (= the total number of units in the hidden and output layers). Thus, the INN includes $mn + 2m + 1$ interval variables in total. Our IPSO handles these interval variables as a genotype $\boldsymbol{X} = (X_1, X_2, \dots, X_D)$ where X_i is an interval and $D = mn + 2m + 1$. X_i can be specified by its upper and lower limits or by its center and width values: $X_i = [x_i^L, x_i^U]$ or $X_i = (x_i^c, x_i^w)$ where x_i^L, x_i^U, x_i^c and x_i^w denote the lower limit, upper limit, center and width of X_i respectively.

III. PSO with Interval Genotypes

Our IPSO consists of the same processes as those in the ordinary PSO. Processes of initialization of populations, updates of particles and fitness evaluation are extended so that these processes can handle interval-valued genotypes.

A. Initialization of Population

In the initialization process, $\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_P$ are randomly initialized where P is the population size. Because the elements in $\boldsymbol{X}_a$ (i.e., $X_{a,1}, X_{a,2}, \dots, X_{a,D}$) are weights and biases in an INN in this research, smaller absolute values of $X_{a,i}$ are preferable as initial values. Thus, the initial values for $X_{a,i}$ are randomly sampled from the normal distribution $N(0, \varepsilon)$ or uniformly from an interval $[-\varepsilon, \varepsilon]$ where $\varepsilon$is a small positive number. In the case of using the [lower, upper] model for specifying interval genotype values, two values are sampled per $X_{a,i}$: the smaller (larger) one is set to $x_{a,i}^L$ ($x_{a,i}^U$). In the case of using the (center, width) model, two values are sampled per $X_{a,i}$: one of the two values is set to$x_{a,i}^c$ and the absolute value of the other is set to $x_{a,i}^w$.

B. Fitness Evaluation

To evaluate fitness of an INN as a phenotype instance of the corresponding genotype instance $\boldsymbol{X}_a = (X_{a,1}, X_{a,2}, \dots, X_{a,D})$ where $\boldsymbol{X}_a \in \{\boldsymbol{X}_1, \boldsymbol{X}_2, \dots, \boldsymbol{X}_P\}$, the INN is supplied with several samples of input real vectors and calculates output values. The input values are sampled within the variable domain of application problem. Fitness of the genotype instance $\boldsymbol{X}_a$ is evaluated based on the output values. The method for scoring the fitness based on the output values depends on the problem to which the INN is applied. For example, in a case where the INN is applied to controlling an automated system, some performance measure of the system can be used as the fitness score of the genotype instance corresponding to the INN.

C. Updates of Particles

Let the position vector of a particle, its personal best and its global (or local) best be denoted as $\boldsymbol{X}_r, \boldsymbol{Pbest}_r, \boldsymbol{Gbest}_r$ (or $\boldsymbol{Lbest}_r$). In the case of using the [lower, upper] model, $\boldsymbol{X}_r = (\boldsymbol{X}_{r,1}, \boldsymbol{X}_{r,2}, \dots, \boldsymbol{X}_{r,D})$ and $X_{r,i} = [x_{r,i}^L, x_{r,i}^U]$. Let the velocity for $x_{r,i}^L$ and $x_{r,i}^U$ be denoted as $v1_{r,i}$ and $v2_{r,i}$ respectively. Note that $v1_{r,i}$ ($v2_{r,i}$) is not the lower (upper) limit of an interval so that $v2_{r,i}$ can be smaller than $v1_{r,i}$. $v1_{r,i}$ and $v2_{r,i}$ are updated as:

$$v1_{r,i} = w \cdot v1_{r,i} + c_1 r_1 (pbest_{r,i}^L - x_{r,i}^L) + c_2 r_2 (gbest_{r,i}^L - x_{r,i}^L) \quad (6)$$

$$v2_{r,i} = w \cdot v2_{r,i} + c_1 r_1 (pbest_{r,i}^U - x_{r,i}^U) + c_2 r_2 (gbest_{r,i}^U - x_{r,i}^U) \quad (7)$$

employing the global best model, or as:

$$v1_{r,i} = w \cdot v1_{r,i} + c_1 r_1 (pbest_{r,i}^L - x_{r,i}^L) + c_2 r_2 (lbest_{r,i}^L - x_{r,i}^L) \quad (8)$$

$$v2_{r,i} = w \cdot v2_{r,i} + c_1 r_1 (pbest_{r,i}^U - x_{r,i}^U) + c_2 r_2 (lbest_{r,i}^U - x_{r,i}^U) \quad (9)$$

employing the local best model. The constant values w, c_1, c_2 and the random values r_1, r_2 are the same as those in the ordinary PSO with the real-valued genotypes. Similarly, in the case of using the (center, width) model, $v1_{r,i}$ and $v2_{r,i}$ are updated as:

$$v1_{r,i} = w \cdot v1_{r,i} + c_1 r_1 (pbest_{r,i}^c - x_{r,i}^c) + c_2 r_2 (gbest_{r,i}^c - x_{r,i}^c) \quad (10)$$

$$v2_{r,i} = w \cdot v2_{r,i} + c_1 r_1 (pbest_{r,i}^w - x_{r,i}^w) + c_2 r_2 (gbest_{r,i}^w - x_{r,i}^w) \quad (11)$$

employing the global best model, or as:

$$v1_{r,i} = w \cdot v1_{r,i} + c_1 r_1 (pbest_{r,i}^c - x_{r,i}^c) + c_2 r_2 (lbest_{r,i}^c - x_{r,i}^c) \quad (12)$$

$$v2_{r,i} = w \cdot v2_{r,i} + c_1 r_1 (pbest_{r,i}^w - x_{r,i}^w) + c_2 r_2 (lbest_{r,i}^w - x_{r,i}^w) \quad (13)$$

employing the local best model.

By using the updated $v1_{r,i}$ and $v2_{r,i}$, $X_{r,i}$ is updated as:

$$x_{r,i}^L = x_{r,i}^L + v1_{r,i} \quad (14)$$

$$x_{r,i}^U = x_{r,i}^U + v2_{r,i} \quad (15)$$

or as:

$$x_{r,i}^c = x_{r,i}^c + v1_{r,i} \quad (16)$$

$$x_{r,i}^w = x_{r,i}^w + v2_{r,i} \quad (17)$$

Note that $x_{r,i}^L$ must not be larger than $x_{r,i}^U$ because $x_{r,i}^L$ and $x_{r,i}^U$ are the lower and upper limits of the interval $X_{r,i}$. Similarly, $x_{r,i}^w$ must not be negative because $x_{r,i}^w$ is the width of the interval $X_{r,i}$. If the value of $x_{r,i}^L$ becomes larger than the value

of $x_{r,i}^U$ after the updates by (14) and (15), these values must be repaired to meet the constraint. The repair method can be as follows:

- the value of $x_{r,i}^U$ is assigned to $x_{r,i}^L$,
- the value of $x_{r,i}^L$ is assigned to $x_{r,i}^U$,
- the mean value of $x_{r,i}^L$ and $x_{r,i}^U$ is calculated and assigned to both of $x_{r,i}^L$ and $x_{r,i}^U$, or
- the two values for $x_{r,i}^L$ and $x_{r,i}^U$ are switched.

Similarly, if the value of $x_{r,i}^w$ becomes negative after the updates by (17), the value must be repaired to meet the constraint. The repair method can be as follows:

- the value of $x_{r,i}^w$ is assigned to 0, or
- the absolute value of $x_{r,i}^w$ is assigned to $x_{r,i}^w$.

IV. Application of IPSO to Neuroevolution

The author has been experimentally evaluating the ability of our IPSO in evolving INNs. INNs are challenged to approximately model target interval functions $F(x)$. For the sake of simplicity, x is a real value (so that the INN includes only a single input unit) and $0 \leq x \leq 1$, as in the literature [1].

A target function is $F(x) = [F(x)^L, F(x)^U]$ where,

$$F(x)^L = 0.2\sin(2\pi x) - 0.1x^2 + 0.4 \quad (18)$$

$$F(x)^U = 0.2\sin(2\pi x) + 0.1x^2 + 0.6 \quad (19)$$

The two dotted curves in Fig. 3 show $F(x)^L$ and $F(x)^U$.

The [lower, upper] model is experimentally employed in this experiment. The INN is designed as follows:

- Number of units: 1 input, 10 hidden, 1 output.
- Unit activation function: the sigmoidal one.

The IPSO is designed as follows:

- Population size: 100.
- Number of cycle: 10,000.
- $w = 0.9, c_1 = 1.4, c_2 = 1.4$.
- Initial values of $x_{r,i}^L, x_{r,i}^U$ for the interval weights and biases: uniformly random within $[-1.0,1.0]$.
- Initial values of $v1_{r,i}, v2_{r,i}$: 0.0.
- $-10.0 \leq v1_{r,i}, v2_{r,i} \leq 10.0$
- $-10.0 \leq x_{r,i}^L \leq x_{r,i}^U \leq 10.0$.

The fitness value of $\boldsymbol{X}_i, i = 1,2,\ldots,P$, is calculated as follows. An INN which corresponds to $\boldsymbol{X}_i$ is supplied with 101 real input values $\{0.0,0.01,0.02,\ldots,0.99,1.0\}$ and calculates 101 output intervals $O_{i,j} = [o_{i,j}^L, o_{i,j}^U], \mathrm{j} = 1,2,\ldots,101$. Besides, the same input values are supplied to the target function $F(x)$ and 101 intervals $F(x_{i,j}) = [f_{i,j}^L, f_{i,j}^U], j = 1,2,\ldots,101$, are obtained. Then, the error e_i is calculated as $e_i = \sum_j((o_{i,j}^L - f_{i,j}^L)^2 + (o_{i,j}^U - f_{i,j}^U)^2)$. The value of e_i is used as the fitness of $\boldsymbol{X}_i$ where a smaller value of e_i is better. Note that scores of e_i are not utilized for the process of updating the particles but only for comparing the fitness of the particles: the target function $F(x)$ is completely hidden from the IPSO.

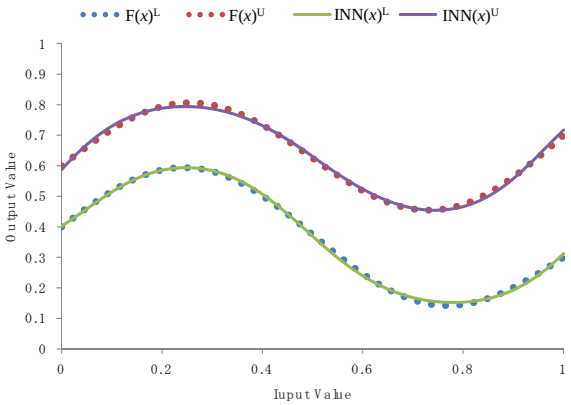

Fig. 3 Target interval function $F(x)$ and output interval by the evolved INN. The two dotted curves show the target function, and the two solid curves show the output interval by the evolved INN. The error was $9.0 * 10^{-3}$

The two solid curves in Fig. 3 show the output interval function by the evolved INN with the smallest error. The result shows that the INN trained by our IPSO could well approximate the target function despite the fact that no training data is explicitly provided.

V. Conclusion

In this paper, the author has applied the interval-valued extension of particle swarm optimization to the neuroevolution of interval-valued neural networks. In the IPSO, values in the genotype are not real numbers but intervals. To handle the interval-valued genotype, the IPSO extends its processes of initialization of populations, fitness evaluation and updates of particles. The experimental result showed that the neural networks evolved by our IPSO approximated well the hidden target function despite the fact that no training data was explicitly provided.

In the future work, the author will further evaluate the ability of the IPSO by experimentally applying it to problems other than neuroevolution.

Acknowledgment

This work is supported by Kyoto Sangyo University Research Grant.

References

[1] H. Ishibuchi, H. Tanaka and H. Okada, An architecture of neural networks with interval weights and its application to fuzzy regression analysis, *Fuzzy Sets and Systems*, vol.57, no.1, pp.27-39, 1993.
[2] D.B. Fogel, L.J. Fogel and V.W. Porto, Evolving neural networks, *Biological Cybernetics*, vol.63, issue 6, pp.487-493, 1990.
[3] X. Yao, Evolving artificial neural networks, *Proc. of the IEEE*, vol.87, issue 9, pp.1423-1447, 1999.
[4] K.O. Stanley and R. Miikkulainen, Evolving neural networks through augmenting topologies, *Evolutionary Computation*, vol.10, no.2, pp.99-127, 2002.
[5] D. Floreano, P. Durr and C. Mattiussi, Neuroevolution: from architectures to learning, *Evolutionary Intelligence*, vol.1, no.1, pp.47-62, 2008.
[6] H. Okada, Proposal of fuzzy evolutionary algorithms for fuzzy-valued genotypes, *Proc. of International Conference on Instrumentation,*

Control, Information Technology and System Integration (SICE Annual Conference) 2012, pp.1538-1541, 2012.
[7] G. Alefeld and J. Herzberger, *Introduction to Interval Computation*, Academic Press, 1983.