ELSEVIER

Journal of Materials Processing Technology 178 (2006) 19–23

Journal of Materials Processing Technology

www.elsevier.com/locate/jmatprotec

PSO-based neural network optimization and its utilization in a boring machine

Jialin Zhou *, Zhengcheng Duan, Yong Li, Jianchun Deng, Daoyuan Yu

Huazhong University of Science and Technology, School of Mechanical Science and Engineering, Luoyu Street 1037, Wuhan 430074, China

Received 24 September 2004; received in revised form 8 March 2005; accepted 19 July 2005

Abstract

This paper presents a particle swarm optimization (PSO) technique in training a multi-layer feed-forward neural network (MFNN) which is used for a prediction model of diameter error in a boring machining. Compared to the back propagation (BP) algorithm, the present algorithm achieves better machining precision with a fewer number of iterations.

Keywords: Diameter error compensation; Particle swarm optimization; Multi-layer feed-forward neural network; Sum of squares error (SSE)

1. Introduction

The prediction of the diameter errors of a boring machining is important in improving machining precision, as it is required for the compensation of on-line machining errors. Much effort has been focused on the modeling and the agile optimization of the machining process, and the artificial neural network technique has been a powerful tool in this area. Artificial neural networks, due to their excellent ability of non-linear mapping, generalization, self-organization and self-learning, are utilized widely in many areas and outstanding achievements have been obtained [1–3]. Neural network models can therefore be implemented to establish a prediction model of diameter errors in a boring machining.

Among neural networks, multi-layer feed-forward neural networks are the most popular, and are implemented with the standardized BP algorithm or its alternatives. As the BP algorithm optimizes a target function by using the gradient descent method, the calculation may overflow or fluctuate between the optima. Another problem is that the convergence of the BP is sensitive to the initial selection of the weights. If the initial sets of weights are not selected properly, the optimization solution could be trapped in a local optimum.

The particle swarm optimization algorithm, proposed by Eberhart and Kenedy [4], is an evolutionary computation technique originating from the behavioral simulation of bird flocking and fish schooling. As PSO possesses the features of straight-forward logic, simple realization and underlying intelligence, it has been developed quickly and utilized widely recently in many areas such as the optimization of discrete and continuous functions, model identification, etc.

In this paper, the implementation of PSO for the self-learning of the neural networks to perform diameter error prediction in a boring machining is presented. In comparison to the BP algorithm, it is verified that the PSO algorithm improves the quality of optimization of the neural networks and error compensations.

2. Introduction to a diameter error compensation system in a boring machining

A complete diameter error compensation system for boring machining, as shown in Fig. 1, consists mainly of the following parts: an on-line diameter metering device, a micro feeding compensation device and a controller. The automatic

* Corresponding author. Tel.: +86 27 87544175.
E-mail address: zhoujialinhust@sina.com (J. Zhou).

0924-0136/$ – see front matter © 2006 Published by Elsevier B.V.
doi:10.1016/j.jmatprotec.2005.07.002

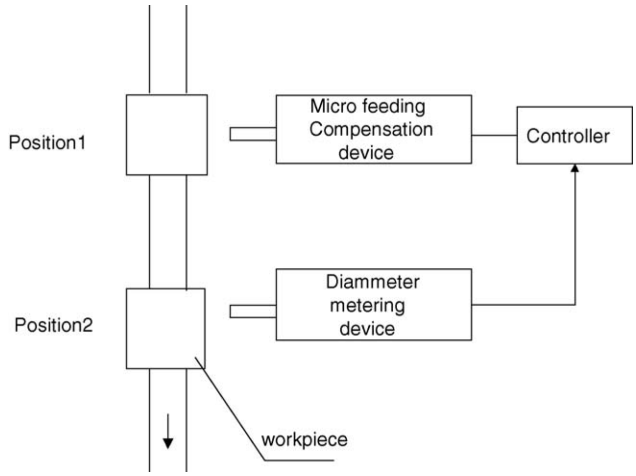

Fig. 1. Error compensation system.

diameter metering device automatically collects the diameters of the machined work-pieces and feeds them into the controller; the controller then predicts the diameter error of the next work-piece through the error prediction model and drives the micro feeding compensation device to realize the diameter compensation. The micro feeding compensation device is essential to achieve a high diameter precision and machining efficiency (the micro feeding compensation device used in the authors' system has been awarded a China patent (no. 02115538.0)).

2.1. Micro feeding compensation device

To realize the micro movement of the boring cutter, an innovative elastic parallelogram, as shown in Fig. 2, is developed and used as the boring rod. The controller predicts the diameter error and sends out a compensation signal to a servo-motor. The servo-motor drives the ball screw device to move backwards and forwards. The slide inside the ball screw device pushes the elastic parallelogram to produce a radical micro deformation. As the boring rod is fixed to the parallelogram, the boring cutter has a radial micro movement synchronized with the deformation and changes the cutting diameter of the work-piece.

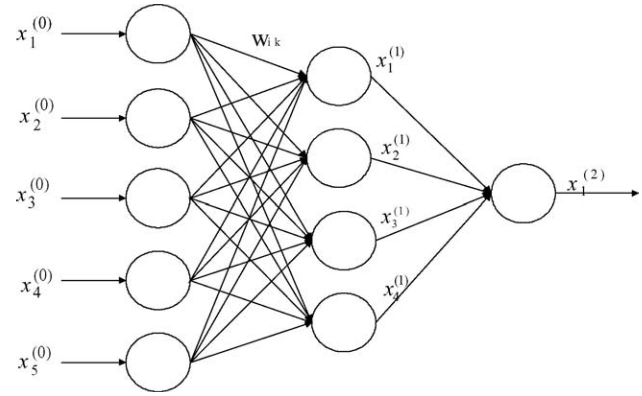

Fig. 3. BP neural network structure.

3. The neural network-based modeling and optimization of diameter error compensation

Hecht-Nielson verified in 1987 that a continuous function in closure space can be approximated to any degree of accuracy by hidden layer neural networks [5]. A three-layer neural network can therefore establish a mapping relation between an n-dimensional space and an m-dimensional space. To develop a diameter error prediction model in a boring machining, the diameter errors can be described mathematically by a non-linear mapping function between the historical data of diameter errors and the "future" diameter errors. The mapping function can be modeled readily by a neural network with three layers: the input layer, the hidden layer and the output layer. The input layer reflects the time series of historical diameter errors; whilst the output layer represents the prediction of the future diameter errors. A $5 \times 4 \times 1$ network structure [6] is shown in Fig. 3.

The activation functions of both the hidden layer and the output layer are Log-Sigmoid functions. When the feedforward neural network (BP network) is implemented, the mapping relationship between the input and the output is described as:

$$s_i^{(q)} = \sum_{j=0}^{n_{q-1}} w_{ij}^{(q)} x_j^{(q-1)}, \quad x_0^{(q-1)} = \theta_i^{(q)},\ w_{i0}^{(q)} = -1 \tag{1}$$

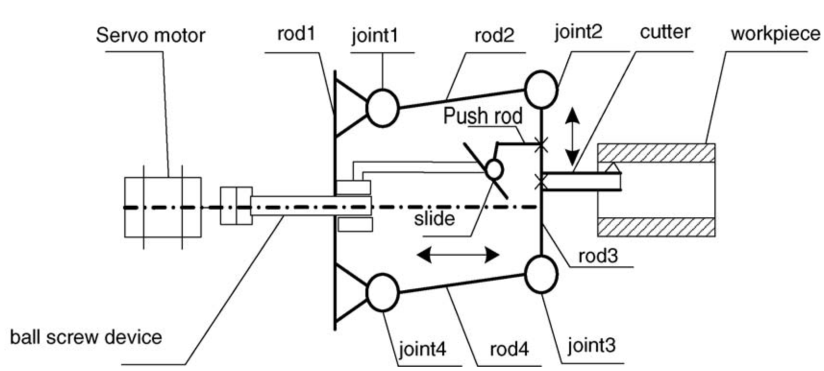

Fig. 2. Elastic parallelogram.

$$x_i^{(q)} = f(s_i^{(q)}) = \frac{1}{1+e^{-s_i^{(q)}}},$$
$$i = 1, 2, \dots, n_q,\ j = 1, 2, \dots, n_{q-1},\ q = 1, 2, \dots, Q \tag{2}$$

where, $w_{ij}^{(q)}$ is the weight between the jth neuron in the $(q-1)$ layer and the ith neuron in the q layer, n_q the number of the neurons in the qth layer, and $f(s_i^{(q)})$ is the activation function.

The basic equation of the BP algorithm is:

$$\Delta w_{ij}(k+1) = \Delta w_{ij}(k) + \eta \frac{\partial E}{\partial w_{ij}} \tag{3}$$

An evolutionary BP algorithm is developed further by combining the additional momentum weight and the learning rate of a standard BP algorithm. Its basic form is as follows:

$$\Delta w_{ij}(k+1) = \mathrm{mc}\Delta w_{ij}(k) + (1-\mathrm{mc})\eta\delta_i p_j \tag{4}$$

$$\Delta b_i(k+1) = \mathrm{mc}\Delta b_i(k) + (1-\mathrm{mc})\eta\delta_i \tag{5}$$

$$\mathrm{mc} = \begin{cases} 0 & \mathrm{SSE}(k) > 1.04\mathrm{SSE}(k-1) \\ 0.95 & \mathrm{SSE}(k) < \mathrm{SSE}(k-1) \\ \mathrm{mc} & \text{others} \end{cases} \tag{6}$$

$$\eta(k+1) = \begin{cases} 1.05\eta(k) & \mathrm{SSE}(k+1) < \mathrm{SSE}(k) \\ 0.7\eta(k) & \mathrm{SSE}(k+1) > 1.04\mathrm{SSE}(k) \\ \eta(k) & \text{others} \end{cases} \tag{7}$$

where mc is the momentum weight, δ_i the error signal item, η the learning rate, k the iteration of learning, p_i the positive output, and b_i is the bias.

The evolutionary BP algorithm increases the speed of convergence and the search of global optima, but its performance is still unsatisfactory. For better optima searching, the PSO technique is adopted.

4. PSO algorithm

The particle swarm optimization algorithm is a relatively new computational intelligence technique to simulate a sort of social behavior. In PSO, a point in the problem space is called a particle, which is initialized with a random position and search velocity. Each particle flies through the problem space and keeps track of its positions and its fitness. Where the latter means the best solution achieved. Its position and velocity are adjusted by its fitness to the environment. Given that a swarm consists of m particles in a D-dimensional problem space, the position and velocity of the ith particle is presented as:

$$s_i = (s_{i1}, s_{i2}, \dots, s_{iD}), \quad i = 1, 2, \dots, m,$$

$$v_i = (v_{i1}, v_{i2}, \dots, v_{iD}),$$

The best position of a particle is denoted by P_{best}, $p_i = (p_{i1}, p_{i2}, \dots, p_{i3})$. Treating the swarm population as a whole, the best position of the best positions of all particles is denoted by G_{best}, in which $p_g = (p_{g1}, p_{g2}, \dots, p_{g3})$.

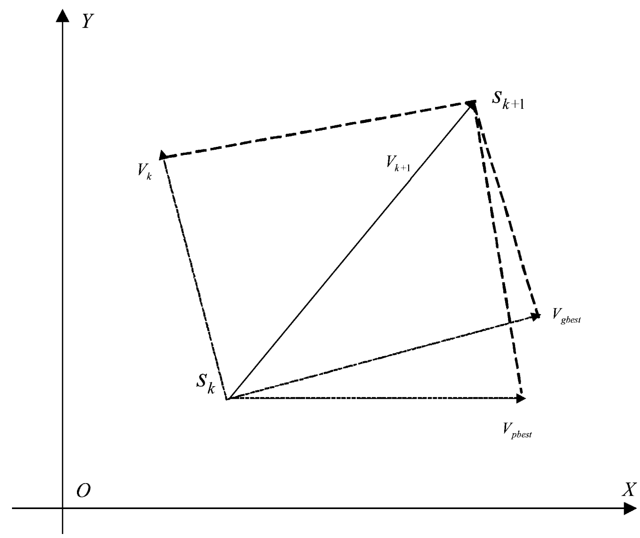

Fig. 4. The illustration of searching in a two-dimensional space.

The PSO algorithm proposed by Kenedy and Eberhart [4] is formulated as:

$$v_{id}^{k+1} = wv_{id}^k + c_1\mathrm{rand}_1(p_{\text{best}} - s_{id}^k) + c_2\mathrm{rand}_2(g_{\text{best}} - s_{id}^k) \tag{8}$$

$$s_{id}^{k+1} = s_{id}^k + v_{id}^{k+1}, \quad i = 1, 2, \dots, m,\ d = 1, 2, \dots, D \tag{9}$$

where, v_{id}^k is the velocity of the ith particle at the kth iteration, w the inertia weight, c_j the accelerating factor, rand the random in a range [0, 1] and s_{id}^k is the current position of the ith particle.

Fig. 4 illustrates the search process of a particle in a two-dimensional space, where s^k is the current position, s^{k+1} the search position next to the current position, v^k the velocity at the current position, v^{k+1} the velocity at the next position, $v_{p_{\text{best}}}$ the best velocity based on the P_{best} and v_p is the velocity based on G_{best}.

To ensure the convergence of the search, Clerc [7] introduced a constriction factor into the standard PSO algorithm. Eq. (8) being converted into:

$$v_{id}^{k+1} = k[v_{id}^k + c_1\mathrm{rand}_1(p_{\text{best}} - s_{id}^k) + c_2\mathrm{rand}_2(g_{\text{best}} - s_{id}^k)] \tag{10}$$

$$k = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \tag{11}$$

where $\varphi = c_1 + c_2$, $\varphi > 4$.

From Eq. (10), k is constricted by c_1 and c_2. Due to k, there is no need of the maximum search velocity $V_{\max}$ and the search convergence is ensured mathematically. In another words, the vibration amplitude of the particle decreases when it is near to the best position. Obviously, the constriction factor in the PSO algorithm can produce the solution better than that of the standard PSO.

5. Neural network training and comparison

For the neural network described in Chapter 3, the training process is performed as below.

5.1. Data collection

There are 210 sets of diameter errors collected from the boring machining, among which 140 sets of data are used for networks training whilst the rest are for verification of the prediction model.

5.2. Evolutionary BP algorithm

As the system is non-linear, the selection of the initial weights is vital to the local optima, the convergence speed and the training time of a neural network. In general, it is expected that the input values to each neuron are close to zero when the initial weights are used. This ensures that the weights of each neuron can be adjusted near to the peak point of its Log-Sigmoid function. Therefore, the weights are selected randomly in the range (0, 1) and the initial bias is set to zero. As the learning rate and momentum weight are tuned adaptively, the initial learning rate is set to 0.01 and the initial momentum weight is set to 0.9. The input data is normalized to be within the range [0, 1].

5.3. The training of the constriction factor PSO

The constriction factor of the PSO algorithm optimizes the weights of the neural networks [8,9]. In the boring machining diameter error prediction model, a three-layer neural network is used: five input neurons, four hidden neurons and one output neuron. There are consequently 24 weights to be optimized. This implies that the swarm of particles flies in a 24-dimensional space to search for the weights with the minimum sum of squares error (SSE). A population of 30 individuals is used. The SSE is used as the fitness of each particle. Particles inside the population are drawn gradually toward the global minimum as the system iterates. When a pre-specified criterion or a maximum number of iterations is reached, the iterations cease. The flow chart of the PSO is shown in Fig. 5.

5.4. Convergence comparison

As shown in Fig. 6, the SSE reaches 0.05 after 50 times of iteration, which is much faster than for an evolutionary BP algorithm, where the SSE does not reach 0.05 in 450 times of iteration.

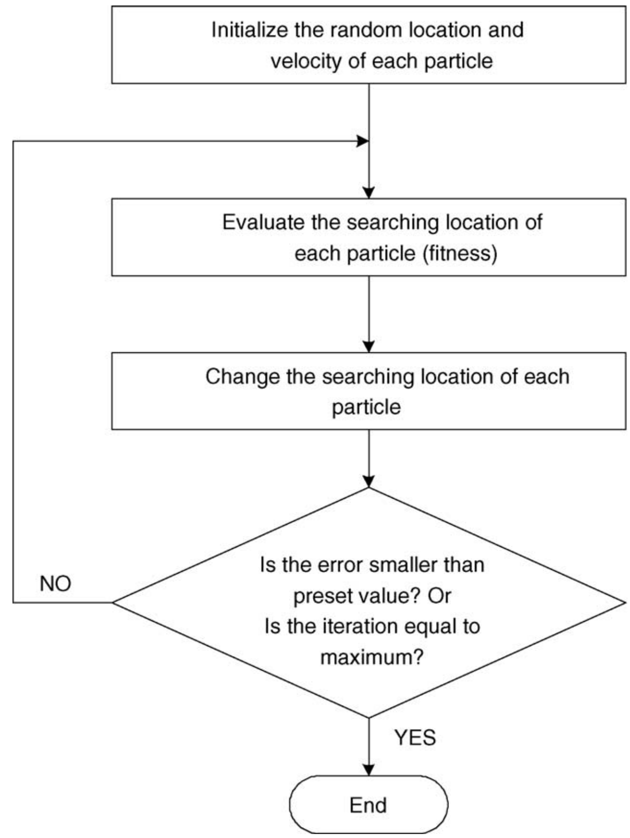

Fig. 5. The flow chart of the PSO algorithm.

5.5. Realistic machining data and processes

To show the effects of the prediction models trained by the BP and the PSO, three compensation cases are used: no error compensation; with the BP-trained prediction model; with the PSO-trained prediction model. In each case, 60 workpieces are machined. The processes of data collection and processing are shown in Tables 1–3.

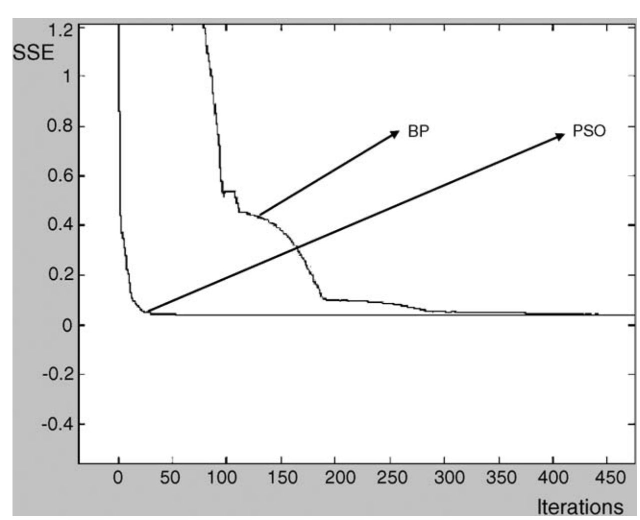

Fig. 6. Convergence velocity comparison of the BP and the PSO.

Table 1
Analysis of the errors without error compensation (μm)

7.2	−5.2	5.6	9.4	9.6	8.2	1.8	7.2	5.2	8
10.8	9.8	4.2	9	7.6	6.4	8.4	−3.2	6.4	8.4
6.2	11.2	4.8	3.8	7.6	6.8	6.6	11.2	8	5
9.6	11.6	7.2	−16.4	5.8	8.8	7	8.2	5.8	7.6
8.6	16	10.8	11.4	4.6	3.8	1	9.8	3	2.4
10.2	9.2	5.6	8.6	7	3.8	6	3.8	6	12
Mean: 6.58					S.D.: 4.56				

Table 2
Analysis of the errors with the BP-trained compensation model (μm)

−2.4	1.2	1.8	0.2	1.6	−1	−4	−4.8	−3.8	−3.6
0.2	−6.2	0.2	−0.8	−1.4	−2.2	−0.4	−4.8	−8.4	−2.2
−2.6	0.4	1	−4.6	−1.4	−1	−1.6	−2.2	3.2	−2
−4.2	−3	−4.4	−0.2	−0.6	−3.4	−2.4	2.8	−2	2.6
3.8	−0.8	−3	−10.4	−5.4	−0.4	−7.6	−2.4	−0.4	0.4
1.4	4.8	1.4	1	−3.4	−3	−2.4	−1.4	−1.4	−2
Mean: −1.63					S.D.: 2.88				

Table 3
Analysis of the errors with the PSO-trained compensation model (μm)

5.8	1.4	−0.6	−4.5	−2.6	1.9	0.1	−1.6	1.4	4.2
8.4	5	1.3	0.1	−2.7	−1.5	1.6	3	5	−0.9
1	−0.6	−1.6	0.8	3.1	1.8	4.5	3.6	2.7	5.6
0.8	−0.6	−1.7	0.1	−0.9	−1.3	1.8	2.8	2	3.6
5	4	3.3	4.4	1.2	−0.8	1.1	0	−0.1	1.1
−2.9	0.8	3.4	5.2	7.8	3.7	4.1	1.7	−1.2	0.2
Mean: 1.53					S.D.: 2.69				

5.6. Machining conditions

The diameter of the work-pieces bored is 25 mm. The material is nodular cast iron QT-400; linear cutting speed, 200 m/min; depth of a single side, 0.15 mm; feed speed, 0.06 mm/r.

6. Conclusion

The paper shows that the networks for diameter error prediction trained by the PSO algorithm or by the BP algorithm both improve the precision of the boring machining, but the neural networks trained by the PSO algorithm perform better than those trained by the BP algorithm. It is expected that the PSO algorithm will become more popular as an optimization tool in many fields in the future.

References

[1] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning internal representations by error backpropagation, in: D.E. Rumelhart, J.L. McClelland (Eds.), Parallel Distributed Processing, MIT Press, 1986.

[2] L. Fausett, Fundamentals of Neural Networks, Prentice-Hall, Englewood Cliffs, NJ, 1994.

[3] K.J. Hunt, D. Sbarbaro, R. Zbikowski, P.J. Gawthrop, Neural networks for control systems—a survey, Automatica 28 (1992) 1083–1112.

[4] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995, pp. 39–43.

[5] R. Hecht-Nielson, Counterpropagation networks, Appl. Opt. 26 (1987) 4979–4984.

[6] J.L. Zhou, J.C. Deng, Study on online forecasting compensation control system for boring process, Chin. Mech. Eng. 14 (23) (2003) 1989–1990.

[7] M. Clerc, The swarm and the queen: towards a deterministic and adaptive particle swarm optimization., in: Proceeding Congress on Evolutionary Computation, Washington, DC, 1999, pp. 1951–1957.

[8] R.C. Eberhart, Y. Shi, Evolving artificial neural networks, in: Proceedings of the International Conference on Neural Networks and Brain, Beijing, PR China, 1998, pp. PL5–PL13.

[9] R.C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in: Proceeding Congress on Evolutionary Computation, San Diego, CA, 2000, pp. 84–88.