*Sensors and Materials*, Vol. 34, No. 2 (2022) 803–817
MYU Tokyo

S & M 2854

# Using Neural Networks for Tool Wear Prediction in Computer Numerical Control End Milling

Cheng-Hung Chen,[1] Shiou-Yun Jeng,[2] and Cheng-Jian Lin[3,4]*

[1]Department of Electrical Engineering, National Formosa University, Yunlin 632, Taiwan
[2]Department of Business Administration, Asia University, Taichung 413, Taiwan
[3]Department of Computer Science & Information Engineering,
National Chin-Yi University of Technology, Taichung 411, Taiwan
[4]College of Intelligence, National Taichung University of Science and Technology, Taichung 404, Taiwan

(Received September 6, 2021; accepted January 4, 2022)

***Keywords:*** backpropagation neural network, tool wear prediction, linear regression, machine tool, milling

The precision of the machining tool in computer numerical control (CNC) machining is affected by several factors. For example, cutting parameters considerably affect machining accuracy and tool wear. Tool wear results in the manufacture of substandard products. Therefore, predicting tool wear is crucial in CNC machining. In this study, we proposed a backpropagation neural network (BPNN) to predict tool wear. In machine learning, backpropagation is a widely used algorithm for training artificial neural networks. The proposed BPNN considered the variation of tool wear with different cutting parameters, such as the spindle speed, feed, cutting depth, and cutting time. The experimental results revealed that the root mean square error of the BPNN prediction model was less than that of the linear regression prediction model. Furthermore, the proposed model achieved a coefficient of determination ($R^2$) of 0.9964, which indicated that the BPNN model can accurately predict tool wear.

## 1. Introduction

Several measurement techniques have been used to indirectly predict tool wear.(1) For decades, piezoelectric sensors and dynamometers have been widely used to collect data and monitor cutting processes. Previous studies on output response methods have mainly employed dynamometers, accelerometers, acoustic emission, and current sensors. Currently, tool wear prediction is performed using regression analysis, neural networks (NNs), and other methods. Among these, NNs are being increasingly adopted in machine tool wear prediction. Advanced sensors, such as current sensors, dynamometers, infrared sensors, and ultrasonic sensors, can help monitor machine tool processing parameters such as feed rate, spindle speed, and cutting depth.(2–6)

Linear regression is a regression analysis technique that models the relationship between one or more independent and dependent variables by using the least squares function (also known as

*Corresponding author: e-mail: cjlin@ncut.edu.tw
https://doi.org/10.18494/SAM3642

ISSN 0914-4935 © MYU K.K.
https://myukk.org/

the linear regression equation). Such a function is a linear combination of one or more model parameters (that is, regression coefficients). In linear regression, data are modeled using linear predictive functions. Unknown model parameters are then estimated. Patra *et al.*(7) investigated the variation of tool wear under various cutting conditions. Furthermore, they investigated the variation of the thrust force with changes in the peck drilling parameters, such as an increase in the number of holes at different feeds and a decrease in the drilling speed. Typically, NN models exhibit a lower prediction error than regression models. Jose *et al.*(8) used the effects of tool wear and surface roughness during the turning process on acoustic emission and force sensors, using linear regression models to predict tool wear. Patra *et al.*(9) compared artificial NN (ANN) and regression models and developed a microdrilling monitoring system with a three-axis accelerometer, data acquisition mechanism, signal processing modules, and ANNs. Ozel and Karpat(10) proposed an NN model to predict the surface roughness and tool flank wear under various cutting conditions during hard turning. Linear regression models have also been developed to determine process-specific parameters.

Many scholars have used ANNs to predict tool wear. Drouillet *et al.*(11) proposed a remaining useful life technique for predicting tool wear on the basis of machine spindle power values by using NNs. Chang *et al.*(12) proposed a prediction system based on a binary NN (BNN) algorithm to estimate the wear of a turning tool. The input parameters of the BNNs were cutting speed, feed rate, and material removal rate. Akarslan and Hocaoglu(13) categorized tools as undamaged, damaged but usable, and damaged and broken, and used these categories as feature vectors. Guo and Mao used an ANN as a classifier and determined that during the milling process, wear in the primary stage indicates that the tool is weak.(14) They subsequently determined that wavelet NNs can effectively handle various signals with different frequencies. Martinsen *et al.*(15) used ANNs for sensor signal analysis as a replacement for the traditional repeated test process monitoring method. Gong and Yang(16) developed a model based on ANN prediction theory to investigate the relationship between input and output parameters. Gao *et al.*(17) investigated the influence of experimental design on modeling of the tool condition monitoring system based on various NNs. Kaya *et al.*(18) proposed an effective strategy based on ANNs to estimate tool wear. Hazza *et al.*(19) experimented with a ceramic cutting tool heat-treated in accordance with AISI 4340 and with a hardness of 60 hardness Rockwell C (HRC) scale and proposed a novel model to predict tool life for various cutting speeds, feed rates, depths of cut, and rake angles. Luetzig *et al.*(20) improved the performance of data fusion algorithms to achieve accurate final flank wear estimates. Regular tests of a network demonstrated that their algorithm can provide reliable final flank wear estimates. Sanjay *et al.*(21) estimated tool wear by using statistical analysis and various NN structures.

In this study, we proposed a backpropagation neural network (BPNN) to predict tool wear. The proposed BPNN considered the variation of tool wear with different cutting parameters, such as the spindle speed, feed, cutting depth, and cutting time. The approach in this study offers the following advantages: 1) The training data of BPNN can be obtained by milling experiments using an orthogonal table. Simultaneously, tool wear results can be obtained by recording the tool length before and after milling. 2) The BPNN parameters are selected using an orthogonal experimental design. 3) The root mean square error (RMSE) of the BPNN models was smaller than that for linear regression and $R^2$ was larger than that for linear regression. The remainder of this paper is organized as follows. Section 2 introduces the tool wear of a computer numerical control (CNC) machine. The tool wear prediction is described in Sect. 3. Section 4 presents the prediction results of the BPNN model. Section 5 presents the conclusions and future research directions.

## 2. Tool Wear of CNC Machine

### 2.1 Cutting force

The cutting force between the tool and the workpiece is a vital factor that directly affects tool wear. When the tool cuts into the workpiece, the applied force cuts chips of the workpiece, which are then removed from the workpiece surface, as depicted in Fig. 1.

### 2.2 Causes of tool wear

The gradual blunting of a tool to the point that it cannot cut the workpiece is called tool wear. If a worn tool is used, vibration and noise are generated, which cause an uneven cutting force and temperature. Therefore, worn tools must be replaced with new tools to ensure cutting precision and quality. Tool wear is categorized into damage and wear. The types and descriptions of tool damage and wear are presented in Table 1.

### 2.3 Stages of tool wear

Tool wear is divided into the following three stages (Fig. 2). Initial wear stage: The tool surface roughness or surface structure is not resistant to wear. Therefore, the cutting starts faster in the short time of cutting. Normal wear stage: The tool is slightly worn and its surface gradually becomes smooth. The cutting force of the cutting surface is uniform. Therefore, wear is slow. Intense wear stage: High friction causes a sharp increase in the tool temperature. The contact between the tool and the workpiece reduces considerably. The blade becomes dull and loses its cutting ability.

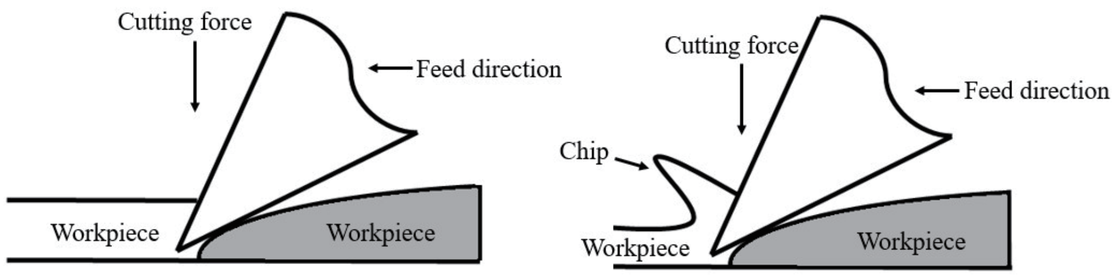

Fig. 1. Schematic of the cutting force.

Table 1
Types of tool damage and wear.

| Tool damage/wear type | | Description |
|---|---|---|
| Damage | Cutting edge or tip collapse | Occurs because of the unsuitable material composition of the tip and the hardness of the workpiece. A large rake angle results in low cutting edge strength, which causes the tool to lose its cutting ability. |
| | Blade or tool break | Occurs when the cutting force is too large or machining is performed inadvertently. The broken tool cannot be used. |
| | Cutting plastic deformation | Typically occurs when the portion to be cut is large and the material is hard. |
| | Thermal break of tool | Occurs because of the repeated thermal expansions and contractions caused by the cutting friction. |
| Wear | Material wear | Attributed to fine particles with high hardness in the processing materials, which result in the formation of grooves on the surface of the tool. |
| | Oxidative wear | As the temperature increases, the oxide generated on the tool surface is cut and rubbed. |
| | Diffusion wear | Caused by the interdiffusion of chemical elements of the workpiece and tool during high-temperature cutting. An unsuitable tool composition accelerates tool wear. |
| Cold welding wear | | A large cutting force and friction exist between the workpiece and the front and back flanks, which is generally more serious at medium cutting speeds. |

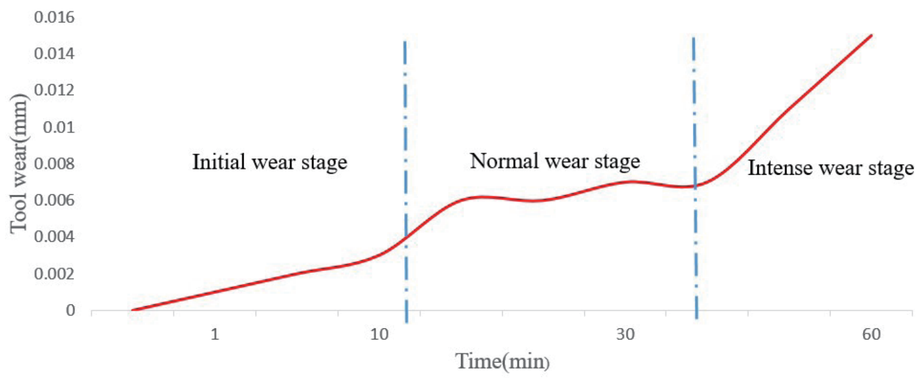

Fig. 2. (Color online) Stages of tool wear.

## 2.4 Cutting parameters and formula

The cutting parameters should be adjusted according to the cutting conditions, and the parameters indirectly affect the quality of cutting. The cutting parameters are as follows.

The depth of cut is the thickness of the tool tip that is milled into the workpiece; if the depth of cut is too large, it may cause extreme tool wear and even tool breakage. The width of the workpiece, tool diameter, number of blades, and cutting width are the other vital parameters of the tool. The cutting depth $d_c$, cutting width $d_w$, diameter of tool $D$, and workpiece are illustrated Fig. 3. The cutting speed depends on the cutting speed of the cutting edge relative to the workpiece, which is determined by the tool diameter and tool revolutions per minute.

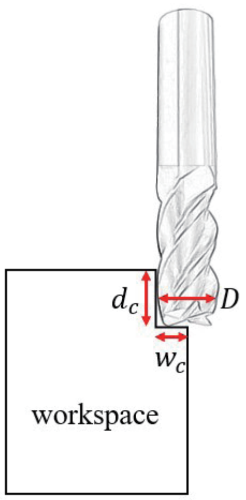

Fig. 3. (Color online) Coding method of chromosomes in GA.

The cutting speed is calculated as follows:

$$V_C = \frac{(\pi \cdot D \cdot N)}{1000} (\mathrm{m/min}), \tag{1}$$

where *Vc* is the cutting speed (m/min), *D* is the tool diameter (mm), and *N* is the spindle speed (rpm).

The feed speed is the speed at which the spindle moves toward the workpiece and is calculated as follows:

$$F = f \cdot z \cdot N, \tag{2}$$

where *F* is the feed rate (mm/min), *f* is the feed rate per blade (mm/min), *z* is number of blades, and *N* is the spindle speed (rpm).

## 3. Tool Wear Prediction

### 3.1 BPNN

In this study, a three-layer BPNN for tool wear prediction was proposed (Fig. 4). The input layer had four neurons, namely, the feed, spindle speed, cutting depth, and time. The numbers of neurons in the hidden layer were determined to be 20, 25, and 30. The output layer had one neuron, namely, tool wear.

The procedure of the proposed BPNN is as follows:

First, determine the number of input layer neurons $x_m$, the number of hidden-layer neurons $h_o$, and the number of output layer neurons $y_n$.

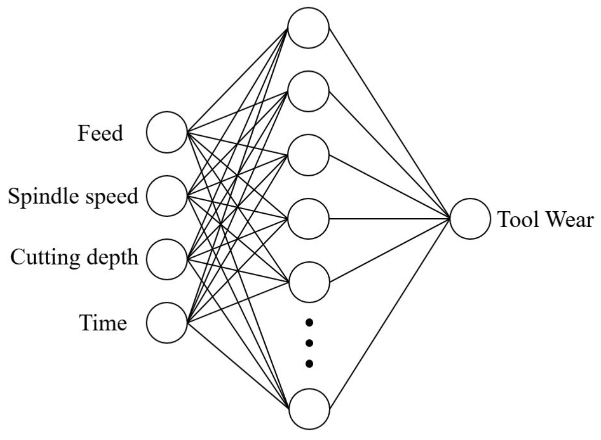

Fig. 4. Proposed BPNN architecture.

Then, randomly set the weight value and the bias of the NN. Here, $w_{ij}$ is the weight value between the input and hidden layers, $w_{jk}$ is the weight value between the hidden and output layers, $\theta h_O$ is the bias of the hidden layer, and $\theta y_n$ is the bias of the output layer.

Calculate the output value of the hidden layer as follows:

$$net\_h_O = \sum_{m=1}^{M} w_{ij} \times x_m - \theta h_o, \tag{3}$$

$$h_o = \frac{1}{1+e^{-net\_h_o}}, \tag{4}$$

where $net\_h_o$ is the weighted product sum of the number of neurons in the hidden layer, $M$ is the total number of input layer neurons, and $h_o$ is the output value of the number of neurons in the hidden layer and is a nonlinear transformation of the input.

Calculate the output value of the output layer using the following formulas:

$$net\_y_n = \sum_{o=1}^{O} w_{jk} \times h_o - \theta y_n, \tag{5}$$

$$y_n = \frac{1}{1+e^{-net\_y_n}}, \tag{6}$$

where $net\_y_n$ is the weighted product sum of the number of neurons in the output layer, $O$ is the total number of hidden-layer neurons, and $y_n$ is the NN output value.

The deviation of the output layer is calculated as follows:

$$\delta y_n = y_n \cdot (1 - y_n) \cdot (T_n - y_n), \tag{7}$$

where $\delta y_n$ is the deviation of the number of neurons in the output layer and indicates the deviation between $y_n$ and the target output value $T_n$.

Calculate the deviation of the hidden layer as follows:

$$\delta h_o = h_o \cdot (1 - h_o) \cdot \sum_{n=1}^{N} w_{jk} \cdot \delta y_n , \tag{8}$$

where $\delta h_o$ is the deviation of the number of neurons in the hidden layer and $N$ is the total number of output layer neurons. This means that the deviation of the output layer is backpropagated to the hidden layer to calculate the deviation.

Calculate the weighted correction value and the bias correction amount between the input layer and hidden layer using the formulas

$$\Delta = \eta \cdot \delta \cdot + \alpha \cdot \Delta , \tag{9}$$

$$\Delta\theta = -\eta \cdot \delta + \alpha \cdot \Delta\theta , \tag{10}$$

where $\Delta$ is the weighted correction value between input layer neurons and hidden layer neurons, $\Delta\theta$ is the bias correction amount of hidden layer neurons, $\eta$ is the learning rate, and $\alpha$ is the inertia factor.

Calculate the weighted correction amount and the bias correction amount between the hidden layer and output layer using the formulas

$$\Delta = \eta \cdot \delta \cdot + \alpha \cdot \Delta , \tag{11}$$

$$\Delta\theta = -\eta \cdot \delta + \alpha \cdot \Delta\theta , \tag{12}$$

where $\Delta$ is the weighted correction value between the hidden layer neurons and output layer neurons, $\Delta\theta$ is the bias correction amount of the output layer neurons, $\eta$ is the learning rate, and $\alpha$ is the inertia factor. Update the weight value between the input layer and the hidden layer using the formula

$$w_{ij} = w_{ij} + \Delta w_{ij}. \tag{13}$$

Update the bias value of the hidden layer using the expression

$$\theta h_o = \theta h_o + \Delta\theta h_o . \tag{14}$$

Update the weight value between the hidden and output layers using the expression

$$w_{jk} = w_{jk} + \Delta w_{jk} . \tag{15}$$

Update the bias value of the output layer using

$$\theta y_n = \theta y_n + \Delta\theta y_n. \tag{16}$$

Calculate the square error of the output neurons using

$$E = \frac{1}{2}\sum_n (T_n - y_n)^2 , \tag{17}$$

where $E$ is the square error of the output neurons, $T_n$ is the target output value, and $y_n$ is the NN output value. After a learning cycle, the smaller the value of $E$, the better the performance of NN learning. A small error between the NN output value and target output value is desirable.

## 3.2 Linear regression

Linear regression typically predicts continuous variables, one or more independent variables, and dependent variables. Then, the least squares method is used to minimize the error of linear regression. The relationship between two variables is determined by fitting a linear equation. In general, just finding a straight line can predict the direction of the data.

Linear regression is categorized into simple linear regression and multivariate linear regression. Simple linear regression has one independent variable, whereas multivariate linear regression has more than one independent variable. Both regression types are used to investigate the relationship between the independent variable ($X$) and the dependent variable ($Y$).

Simple linear regression is expressed as follows:

$$Y = a + b_1 X_1 + \varepsilon , \tag{18}$$

where $a$ is a constant, $b_1$ is the regression coefficient, and $\varepsilon$ is the error value.

Multivariate linear regression is expressed as follows:

$$Y = a + b_1 X_1 + b_2 X_2 + \cdots + b_k X_k + \varepsilon , \tag{19}$$

where $a$ is a constant, $b_1 \cdots b_k$ are the regression coefficients, and $\varepsilon$ is the error value.

Multivariate variable data are calculated using linear regression. In regression analysis, the least squares method is most often used to determine the best match of data, and the sum of squared errors between the predicted data and the actual data are minimized.

The formula used in the least squares method is as follows:

$$Q(a,b) = \sum_{i=1}^{n} (Y_i - a - bX_i)^2 , \tag{20}$$

where $n$ is the total amount of data, $a$ is a constant, $b$ is the regression coefficient, $X_i$ is the independent variable of linear regression, and $Y_i$ is the dependent variable of linear regression.

In this study, a multivariate linear regression was performed for the parameters of tool wear, and four independent variables and one dependent variable were proposed. The independent variables are labeled $X_1$ to $X_4$.

The tool wear multivariate linear regression can be expressed as follows:

$$Y = a + b_1X_1 + b_2X_2 + b_3X_3 + b_4X_4 + \varepsilon\,, \tag{21}$$

where $Y$ is the tool wear, $X_1$ is the feed, $X_2$ is the spindle speed, $X_3$ is the cutting depth, $X_4$ is the time, and $\varepsilon$ is the error value.

## 4. Results and Discussion

### 4.1 Prediction results of BPNN model

The training data of the BPNN are established using an orthogonal table $L_{24}$ design. In the process of milling, the tool wear can be obtained from the difference between the tool length before and after actual milling experiments. Finally, the training data can be built by recording the tool wear. The input of the BPNN is the equipment parameters of the CNC machining during milling including the feed, spindle speed, cutting depth, and time. Furthermore, the output of the BPNN is the measured tool wear in milling experiments. The BPNN target value ($T_n$) and the BPNN output predicted value ($Y_n$) of each training data were analyzed, and the RMSE between the calculations was determined.

The RMSE is the square root of the mean square error (MSE) and is used to determine the accuracy of a prediction, that is, the degree of deviation between the predicted and target values. The smaller the value, the higher the accuracy.

The RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{n}^{N}(T_n - Y_n)^2}\,, \tag{22}$$

where $N$ is the total number of data. The parameters of the BPNN are listed in Table 2.

Table 3 shows the prediction results obtained using the BPNN. The parameters in the BPNN are set to the number of iterations ($g$): 500000, learning rate ($\eta$): 0.5, inertia factor ($\alpha$): 0.8, and

Table 2
Initial parameters of BPNN.

| Number of iterations ($g$) | 100000 | 200000 | 500000 |
|---|---|---|---|
| Learning rate ($\eta$) | 0.1 | 0.2 | 0.5 |
| Inertia factor ($\alpha$) | 0.5 | 0.8 | — |
| Number of hidden-layer neurons ($H_n$) | 20 | 25 | 30 |

Table 3
Prediction results using BPNN.

| No. | $F$ | $S$ | $C$ | $T$ | $T_n$ | $Y_n$ | Error |
|---|---|---|---|---|---|---|---|
| 1 | 3000 | 5000 | 0.3 | 60 | 0.0500 | 0.0469 | −0.0031 |
| 2 | 3000 | 5000 | 0.5 | 60 | 0.0600 | 0.0616 | 0.0016 |
| 3 | 3000 | 5000 | 1 | 60 | 0.1000 | 0.0997 | −0.0003 |
| 4 | 3000 | 5000 | 0.3 | 30 | 0.0200 | 0.0250 | 0.0050 |
| 5 | 3000 | 5000 | 0.5 | 30 | 0.0300 | 0.0284 | −0.0016 |
| 6 | 3000 | 5000 | 1 | 30 | 0.0400 | 0.0384 | −0.0016 |
| 7 | 3000 | 8000 | 0.3 | 60 | 0.0600 | 0.0605 | 0.0005 |
| 8 | 3000 | 8000 | 0.5 | 60 | 0.0800 | 0.0802 | 0.0002 |
| 9 | 3000 | 8000 | 1 | 60 | 0.0900 | 0.0882 | −0.0018 |
| 10 | 3000 | 8000 | 0.3 | 30 | 0.0300 | 0.0303 | 0.0003 |
| 11 | 3000 | 8000 | 0.5 | 30 | 0.0500 | 0.0468 | −0.0032 |
| 12 | 3000 | 8000 | 1 | 30 | 0.0600 | 0.0618 | 0.0018 |
| 13 | 4000 | 5000 | 0.3 | 60 | 0.0500 | 0.0524 | 0.0024 |
| 14 | 4000 | 5000 | 0.5 | 60 | 0.0800 | 0.0798 | −0.0002 |
| 15 | 4000 | 5000 | 1 | 60 | 0.1000 | 0.0985 | −0.0015 |
| 16 | 4000 | 5000 | 0.3 | 30 | 0.0400 | 0.0377 | −0.0023 |
| 17 | 4000 | 5000 | 0.5 | 30 | 0.0500 | 0.0493 | −0.0007 |
| 18 | 4000 | 5000 | 1 | 30 | 0.0600 | 0.0603 | 0.0003 |
| 19 | 4000 | 8000 | 0.3 | 60 | 0.0800 | 0.0794 | −0.0006 |
| 20 | 4000 | 8000 | 0.5 | 60 | 0.1300 | 0.1286 | −0.0014 |
| 21 | 4000 | 8000 | 1 | 60 | 0.1500 | 0.1489 | −0.0011 |
| 22 | 4000 | 8000 | 0.3 | 30 | 0.0200 | 0.0188 | −0.0012 |
| 23 | 4000 | 8000 | 0.5 | 30 | 0.0300 | 0.0345 | 0.0045 |
| 24 | 4000 | 8000 | 1 | 30 | 0.0700 | 0.0685 | −0.0015 |

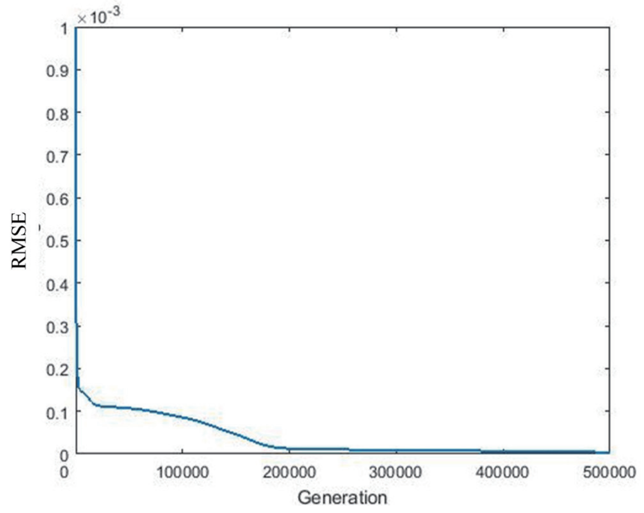

Fig. 5. (Color online) Learning curve ($g$: 500000, $\eta$: 0.5, $\alpha$: 0.8, $H_n$: 20).

number of hidden-layer neurons ($H_n$): 20. Figure 5 shows that the learning curve has decreasing RMSE with increasing number of iterations. The curve illustrates the convergence process and mean square error.

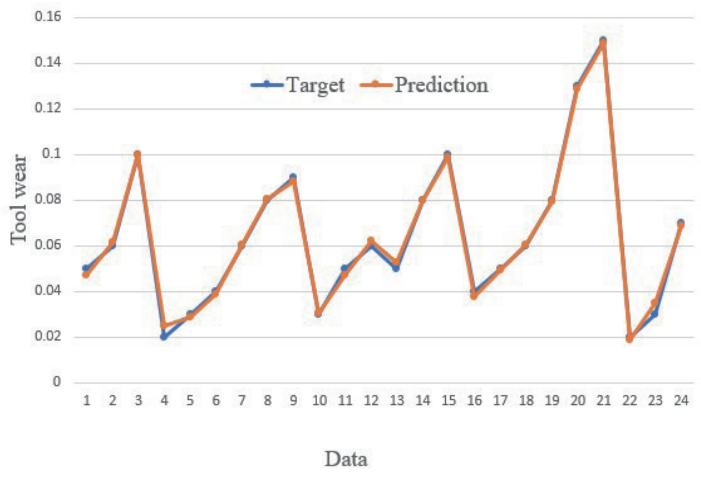

Fig. 6. (Color online) Target and predicted results of the BPNN.

According to the prediction results in Table 3, the target value was similar to the BPNN output value. The measured value of the tool setter used in this experiment was the third digit after the decimal point. Therefore, a small error value was desired. The variation of the target and predicted values is depicted in Fig. 6.

## 4.2 Comparison of BPNN parameters

For the BPNN parameters, the variations in the RMSE of the number of iterations ($g$), learning rate ($\eta$), inertia factor ($\alpha$), and number of hidden-layer neurons ($H_n$) are presented in Table 4.

If the number of hidden-layer neurons is small, the error value will become large during the learning process. In contrast, while the number is becoming larger, the error value will decrease, which causes the convergence speed to also decrease.

If the learning rate is too low, then network vibration can be avoided. However, low learning rates lead to slow convergence. If the learning rate is too large, then the target value can be approached swiftly, but too much correction can occur, which can cause the network to vibrate, to be unable to converge, or even to diverge.

The inertia factor is the variation from the previous weighted value to the next weighted value. Typically, the value is between 0 and 1. The purpose of the inertia factor is to keep the weighted value variation of the neural network in the same direction.

According to the experimental results in Table 4, the RMSEs of experiments 16, 20, and 23 are less than 0.02. The common point inertia factor and the number of hidden-layer neurons in these experiments are 0.8 and 20, respectively.

Next, we analyzed the RMSE variation for the four parameters (Figs. 7–10). In the experiment, 100000, 200000, and 500000 iterations were performed. The learning rate, inertia factor, and number of hidden layers were 0.5, 0.8, and 20, respectively. In Fig. 7, as the number of iterations increased, the RMSE decreased.

Table 4
RMSE results using various parameters in BPNN.

| No. | $g$ | $\eta$ | $\alpha$ | $H_n$ | RMSE |
|---|---|---|---|---|---|
| 1 | 100000 | 0.1 | 0.5 | 20 | 0.043160783 |
| 2 | 100000 | 0.1 | 0.8 | 20 | 0.040845023 |
| 3 | 100000 | 0.2 | 0.5 | 20 | 0.041291525 |
| 4 | 100000 | 0.2 | 0.8 | 20 | 0.039677893 |
| 5 | 100000 | 0.2 | 0.5 | 25 | 0.041327163 |
| 6 | 100000 | 0.2 | 0.5 | 30 | 0.041371707 |
| 7 | 100000 | 0.5 | 0.8 | 20 | 0.031305923 |
| 8 | 100000 | 0.5 | 0.5 | 20 | 0.040108104 |
| 9 | 200000 | 0.1 | 0.5 | 20 | 0.041789529 |
| 10 | 200000 | 0.1 | 0.8 | 20 | 0.039590307 |
| 11 | 200000 | 0.2 | 0.5 | 20 | 0.040648152 |
| 12 | 200000 | 0.2 | 0.5 | 25 | 0.040710524 |
| 13 | 200000 | 0.2 | 0.5 | 30 | 0.040947384 |
| 14 | 200000 | 0.2 | 0.8 | 20 | 0.032222959 |
| 15 | 200000 | 0.5 | 0.5 | 20 | 0.035253436 |
| 16 | 200000 | 0.5 | 0.8 | 20 | 0.013551621 |
| 17 | 500000 | 0.1 | 0.5 | 20 | 0.040009647 |
| 18 | 500000 | 0.1 | 0.8 | 20 | 0.032885653 |
| 19 | 500000 | 0.2 | 0.5 | 20 | 0.033029596 |
| 20 | 500000 | 0.2 | 0.8 | 20 | 0.013749464 |
| 21 | 500000 | 0.2 | 0.5 | 25 | 0.036251435 |
| 22 | 500000 | 0.2 | 0.5 | 30 | 0.035760352 |
| 23 | 500000 | 0.5 | 0.8 | 20 | 0.008338854 |
| 24 | 500000 | 0.5 | 0.5 | 20 | 0.020994665 |

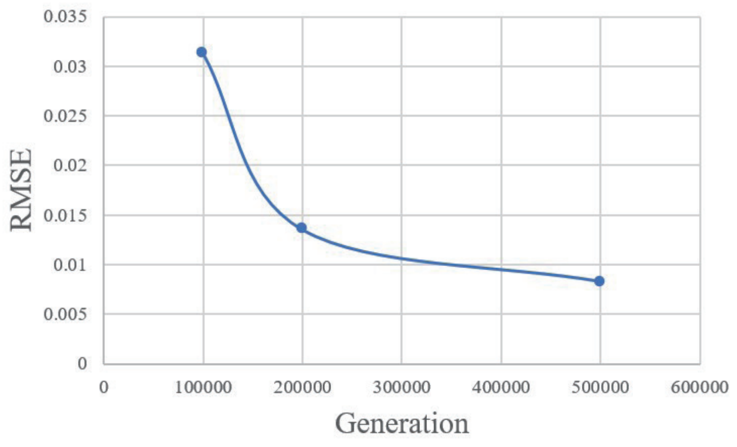

Fig. 7. (Color online) RMSE results using various numbers of iterations.

Here, the learning rates of the experiment were 0.1, 0.2, and 0.5. The iterations, inertia factor, and number of hidden layers were 500000, 0.5, and 20, respectively. Figure 8 shows the RMSE results obtained using various learning rates, and the RMSE decreases as the learning rate increases.

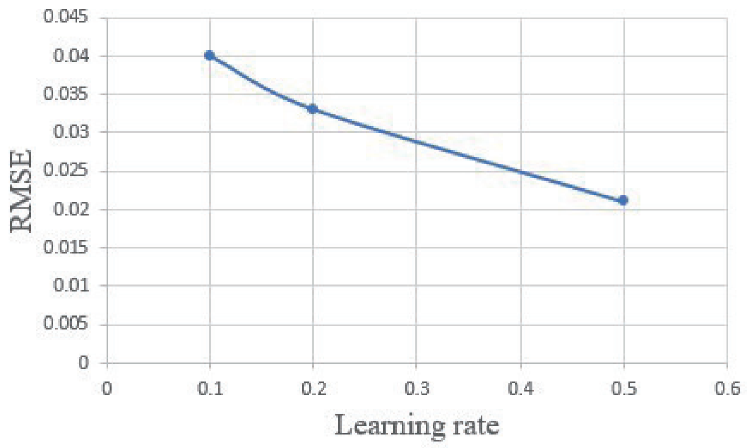

Fig. 8. (Color online) RMSE results using various learning rates.

In this experiment, the inertia factor was varied (0.5 and 0.8) and the number of iterations, learning rate, and number of hidden layers were fixed at 500000, 0.5, and 20, respectively. Figure 9 shows the RMSE results obtained using various inertia factors, and the RMSE decreases as the inertia factor increases. As shown in Figs. 8 and 9, when the values of the learning rate and the inertia factor gradually increase, the RMSE will gradually decrease.

In this experiment, the number of hidden-layer neurons was varied (20, 25 and 30). The number of iterations, learning rate, and inertia factor were fixed at 200000, 0.5, and 0.5, respectively. Figure 10 shows that the number of neurons in the hidden layer did not considerably influence the RMSE. Too many neurons may cause divergence, which in turn increases errors. In the BPNN parameter experiment, the number of hidden layer neurons has the least influence on RMSE during the learning process.

A comparison of the RMSE and coefficient of determination ($R^2$) for the BPNN and linear regression is presented in Table 5. In statistics, $R^2$ is used to measure the difference between dependent and independent variables to explain the relationship between the target and predicted values. Typically, the value is between 0 and 1. If the value is closer to 1, then the predictive ability of the system is better. $R^2$ is calculated as follows:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\Sigma(T_n - Y_n)^2}{\Sigma(T_n - T)^2}, \tag{23}$$

where $SS_{res}$ is the sum of the squares of the residuals of the target and predicted values, $SS_{tot}$ is the error between the target and average values, $Y_n$ is a predicted value, $T_n$ is a target value, and $T$ is the average of the target values.

As shown in Table 5, in which the prediction results of linear regression and the BPNN are compared, the proposed BPNN achieves a small RMSE (0.008338854) and a large $R^2$ (0.9964). Therefore, the experimental results show that the BPNN is better than linear regression.

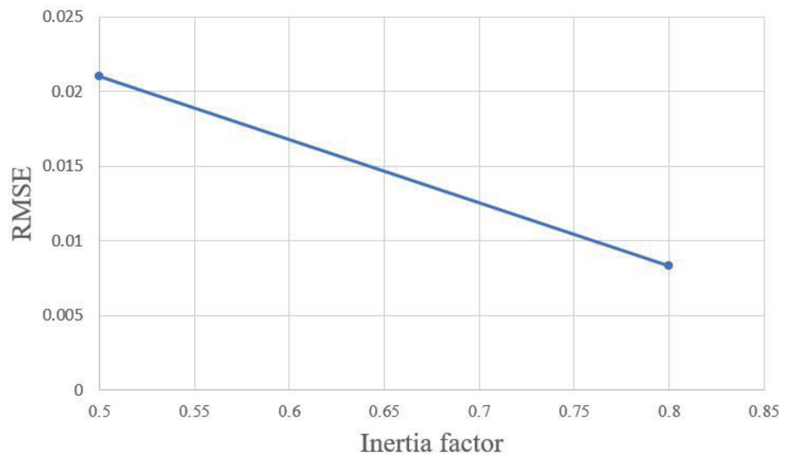

Fig. 9. (Color online) RMSE results using various inertia factors.

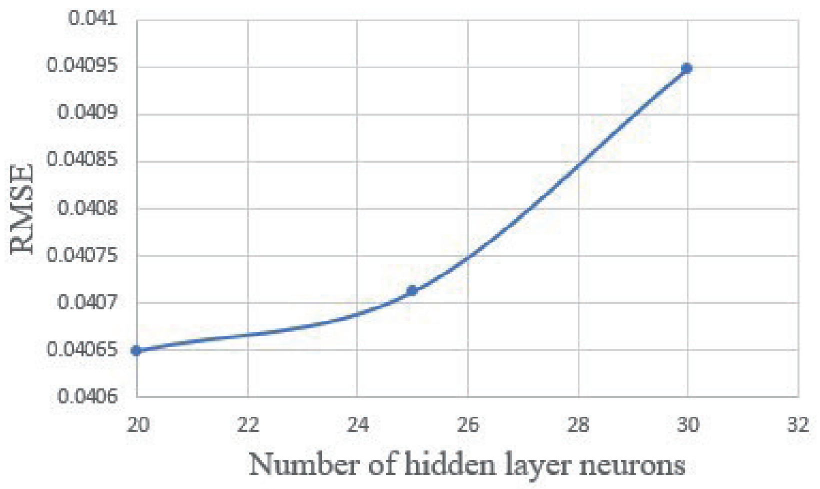

Fig. 10. (Color online) RMSE results using various numbers of hidden-layer neurons.

Table 5
Comparison of prediction results using linear regression and BPNN.

| Model | RMSE | $R^2$ |
|---|---|---|
| Linear regression | 0.021215796 | 0.6422 |
| BPNN | 0.008338854 | 0.9964 |

## 5. Conclusions

In this study, tool wear prediction was performed using BPNN and multiple linear regression models. In tool milling, tool wear is affected by the feed, spindle speed, cutting depth, and time. The BPNN model achieved excellent prediction with 500000 iterations, a learning rate of 0.5, an inertia factor of 0.8, and 20 hidden-layer neurons. The RMSE was smallest with these values.

RMSE and $R^2$ of the BPNN model were smaller and larger than those of linear regression, respectively. Therefore, the BPNN model exhibited excellent tool wear prediction.

## Acknowledgments

The authors would like to thank the Ministry of Science and Technology of the Republic of China, Taiwan, for financially supporting this research under Contract Nos. MOST 110-2221-E-150-011 and MOST 110-2218-E-005-016.

## References

1 R. Corne, C. Nath, M. E. Mansori, and T. Kurfess: J. Manuf. Syst. **43** (2017) 287. https://doi.org/10.1016/j.jmsy.2017.01.004
2 N. Ghosh, Y. B. Ravi, A. Patra, S. Mukhopadhyay, S. Paul, A. R. Mohanty, and A. B. Chattopadhyay: Mech. Syst. Signal Process. **21** (2007) 466. https://doi.org/10.1016/j.ymssp.2005.10.010
3 M. Malekian, S. S. Park, and M. B. G. Jun: J. Mater. Process. Technol. **209** (2009). https://doi.org/10.1016/j.jmatprotec.2009.01.013
4 D. F. Hesser and B. Markert: Manuf. Lett. **19** (2019) 1. https://doi.org/10.1016/j.mfglet.2018.11.001
5 M.V. Vardhan, G. Sankaraiah, and M. Yohan: Mater. Today: Proc. **5** (2018) 7. https://doi.org/10.1016/j.matpr.2018.06.177
6 I. Kandilli, M. Sonmez, H. M. Ertunc, and B. Cakir: Proc. 2007 IEEE Int. Conf. Mechatron. Autom. (IEEE, 2007) 1388–1394.
7 K. Patra, A.K. Jha, T. Szalay, J. Ranjan, and L. Monostori: Precise Eng. **48** (2017) 279. https://doi.org/10.1016/j.precisioneng.2016.12.011
8 B. Jose, K. Nikita, T. Patil, S. Hemakumar, and Dr. P. Kuppan: Mater. Today: Proc. **5** (2018) 8299. https://doi.org/10.1016/j.matpr.2017.11.521
9 K. Patra, A. Jha, and T. Szalay: Proc. 2017 IEEE Conf. Industrial Engineering and Engineering Management (IEEE, 2017) 1–5.
10 T. Ozel and Y. Karpat: Int. J. Mach. Tools Manuf. **45** (2005) 467. https://doi.org/10.1016/j.ijmachtools.2004.09.007
11 C. Drouillet, J. Karandikar, C. Nath, A. C. Journeaux, M. E. Mansori, and T. Kurfess: J. Manuf. Process. **22** (2016) 161. https://doi.org/10.1016/j.jmapro.2016.03.010
12 W. Y. Chang, S. J. Wu, and B. S. Lin: 2018 IEEE Conf. Advanced Manufacturing (IEEE, 2018) 280–283.
13 E. Akarslan and F. O. Hocaoglu: 2014 22nd Signal Processing and Communications Applications Conf. (IEEE, 2014) 622–625.
14 G. Guo and X. Mao: Proc. 2011 IEEE Conf. Electronic & Mechanical Engineering and Information Technology (IEEE, 2011) 1421–1423.
15 K. Martinsen, J. Downey, and I. Baturynska: Procedia CIRP **41** (2016) 933–938.
16 L. X. Gong and M. Z. Yang: 2010 3rd Int. Conf. Information and Computing (IEEE, 2010) 38–41.
17 H. Gao, M. Xu, Y. C. Su, P. Fu, and Q. Liu: 2008 7th World Congr. Intelligent Control and Automation (IEEE, 2008) 6906–6910.
18 B. Kaya, C. Oysu, and H. M. Ertunc: Adv. Eng. **42** (2011) 76. https://doi.org/10.1016/j.advengsoft.2010.12.002
19 M. H. F. A. Hazza, E. Y. T. Adesta, and M. H. Hasan: 2013 IEEE Conf. Advanced Computer Science Applications and Technologies (IEEE, 2013) 70–73.
20 G. Luetzig, M. Sanchez-Castillo, and R. Langari: Proc. Int. Jt. Conf. Neural Netw. 4 (IEEE, 1997) 2359–2363.
21 C. Sanjay, M. L. Neema, and C. W. Chin: J. Mater. Process. Technol. **170** (2005) 494. https://doi.org/10.1016/j.jmatprotec.2005.04.072