Wear Value Prediction of CNC Turning Tools based on ν-GSVR with A New Hybrid Evolutionary Algorithm

Min-Liang Huang[1], Jr-syu Yang[2], Jheng-yu Wu[1,*], and Shih-Hsing Chang[1,*]

[1] Department of Mechanical and Electro-Mechanical Engineering, Tamkang University, New Taipei City, Taiwan, R.O.C.
[*] Corresponding author. E-mail: 605350122@s05.tku.edu.tw, prof.chang@msa.hinet.net

Received: Oct. 30, 2019; Accepted: Apr. 04, 2020

The dimensional accuracy of the workpiece will exceed the tolerance, therefore, to predict how many workpieces have been cut, the turning tool must be replaced is the important issue in machining field. To deal well with the normally distributed random error existed in the wear value prediction of CNC turning tools, this paper introduces the -Support Vector Regression (ν-GSVR) model with the Gaussian loss function to the prediction field of short-term wear value. A new hybrid evolutionary algorithm (namely CCGA) is established to search the appropriate parameters of the ν-GSVR, coupling the Chaos Map, Cloud model and Genetic Algorithm. Consequently, a new forecasting approach for the short-term wear value prediction of CNC turning tools, combining ν-GSVR model and CCGA algorithm, is proposed. The forecasting process considers the wear value prediction of CNC turning tools during the first few time intervals, the turning tool wear value for the spindle revolution, cutting depth and feed rate. It is used to verify the forecasting performance of the proposed model. The experiment indicates that the model yield more accurate results than the compared models in forecasting the short-term wear value on the turning tools. In this way, we can figure out how many turning tools to prepare for similar workpieces, which can reduce the stock of turning tools, and reduce the labor costs on quality inspection of workpieces during this period.

Keywords: Wear Value Prediction; CNC Turning Tools; Support Vector Machine; Support Vector Regression; Gaussian loss function; Genetic Algorithm; Chaos Map; Cloud model

http://dx.doi.org/10.6180/jase.202006_23(2).0021

1. Introduction

Wear value prediction of CNC turning tools the key index in reducing the stock of turning tools, and the labor costs on quality inspection of workpieces [1]. Data-insufficient wear value is also the key technology for intelligent flexible manufacturing systems; therefore, the less-data wear value prediction of CNC turning tools has a great practical significance. According to different principles, the current less-data wear value of CNC turning tools prediction methods can be classified into two types. The first type is the prediction method based on the determination of mathematical models, including the early common regressive moving average model (ARMA) [2], the later more complex ARIMA model [3] with higher accuracy and the Kalman filter model [4]. The second type is the knowledge-based intelligent model prediction method, including fuzzy theory [5], wavelet theory [6], chaos theory [7] and neural networks [8–10]. The traditional calculation method is simple and fast, but it does not reflect the uncertainty and nonlinearity of the wear value of CNC turning tools. Therefore, it is ineffective in handling complex wear value prediction problems. The neural network algorithm is a typical representation of the second type method, because the neural network algorithm cannot overcome deficiencies in empirical risk minimization. In other words, when the sample size of study is limited, the accuracy is difficult to guarantee; but when the sample size of study is large, it is easy to fall into the trap of over-fitting, so the neural network algorithm is versatile. Imbalanced data is another common problem, how to solve the problem of misclassification of minority class samples [11] and predict them accurately has become a new challenge.

A SVMs-rebalance methodology [12]. is a graphical prediction method link with a kernel version of multidimensional scaling in high-dimensional feature space.

IShort-term load forecasts are needed for the efficient management of power systems like turing tools, so a new SVD(singular value decomposition)-based exponential smoothing formulation may be a good choice [13].

Having overcome the inherent defects in the neural network [14], the SVR has the advantages of a global optimum, a simple structure, strong small sample promotion ability and it is based on structural risk minimization criterion. Therefore, it can solve the problems such as small sample, nonlinearity, high dimension, local minimum, and has been successfully applied to short-term wear value prediction of CNC turning tools prediction [14]. But the standard SVR model cannot effectively deal with the noise arising from the flow sequence due to random factors. Zhu et al [15] presented the short-term traffic flow prediction model based on wavelet analysis and SVR, which had better prediction capabilities. Wavelet analysis has an unparalleled advantage in the handling of noise, but it is complex process and very time-consuming. Different from other predictions, short-term traffic prediction is done online in real time, and has high real-time prediction requirements. That is to say the computational time is more "expensive" within the accuracy level required of wear value prediction. Based on the Gaussian function which can effectively handle normally distributed random errors, Wu [16] proposed a Gaussian loss function based ν-support vector regression (ν-GSVR), which achieved good filtering results in the application of predicting product sales. This paper applies the ν-GSVR model to forecast short-term wear value of CNC turning tools.

However, the SVR model does not give a method for optimizing the combination of model parameters, and there is certain cross-error [17] in the commonly used cross validation. GA [18] in the optimization of model parameters has global optimization, robustness and self-adaptability [16, 19]. But standard GA has some deficiencies. While providing an evolution opportunity for the individuals in the population, the random operation inevitably causes degradation in the group, leading to some dependence on the algorithm and the initial population and is prone to "precocious" or local convergence. The direction of genetic evolution is random and uncontrollable leading to slow searches for the optimal solution or a satisfactory solution. To quickly and accurately search for the optimal parameter combinations of ν-GSVR model, the standard GA has been improved through the implementation of Cat Map and Cloud models. This paper proposes Chaos Cloud genetic algorithm (CCGA), to optimize the parameter combinations of ν-GSVR model. Finally, a new approach for predicting short-term wear value of CNC turning tools is established, combining ν-GSVR with CCGA (CCGA-ν-GSVR). The proposed model considers the relationship between the short-term wear value and the real wear value during the first few time intervals.

The rest of this paper is organized as follows. Section 2 describes the ν-GSVR model. Section 3 provides CCGA based on Cat Maps, Cloud model and GA algorithm. Section 4 introduces the proposed CCGA-ν-GSVR forecasting model. Section 5 illustrates a numerical example to reveal the forecasting performance of the proposed forecasting model. The conclusions are given in Section 6.

2. ν-support Vector Regression Model with Gaussian Loss Function

2.1. ν-support vector regression based ϵ-loss function

Suppose training set $T = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, where x_i is a d-dimensional input variable, and y_i is the corresponding output value. Through a nonlinear mapping function $\Phi(x) = \Phi(x1), \Phi(x2), \ldots \Phi(xl)$, SVR model maps the sample into a high dimensional feature space R^{df}, in which the optimal decision function is constructed as follows:

$$f(x) = w^T.\phi(x) + b, w \in R^{df}, b \in R \quad (1)$$

Where: w is weight vector, b is bias value, and fitting function $f(x)$ minimizes the following objective function (structural risk):

$$min\left[\frac{1}{2}\|\omega\|^2 + C \cdot R_{emp}\right] \quad (2)$$

Where $\frac{1}{2}\|\omega\|^2$ is the expression for the complexity of the decision function; the second item, empirical risk R_{emp}, is for the training errors; C is a regulatory factor used to adjust the ratio between the model complexity and the training error R_{emp}. The training error $R_{emp} = \frac{1}{l}\sum_{i=1}^{l}|y_i - f(x_i)|$ can be measured with ϵ, the insensitive loss function defined by $c(x_i, y_i, f(x_i)) = max\{0, |y_i - f(x_i)| - \epsilon\}$.

In the standard support vector regression (ϵ-SVR) model, ϵ insensitive factor controls the sparsity of the solutions and the generalization of models. However, it is very difficult to reasonably determine the value of ϵ in advance. Therefore, Scholkopf et al [19] presented ν-SVR by introducing a parameter ν into the ϵ-SVR model. At this point, the ν-SVR model with the ϵ-insensitive loss function is shown as follows:

$$\begin{aligned} \min_{w,b,\epsilon,\zeta^*} \quad & \tau(w, b, \epsilon, \zeta^*) = \frac{1}{2}\|\omega\|^2 \\ & + C \cdot (\nu \cdot \epsilon + \frac{1}{l}\sum_{i=1}^{l}(\zeta_i + \zeta_i^*)) \end{aligned} \quad (3)$$

$$s.t.\begin{cases}(w.x_i+b)-y_i\le\epsilon+\zeta_i\\ y_i-(w.x_i+b)\le\epsilon+\zeta_l^*\\ \zeta^{(*)}\ge 0,\epsilon\ge 0\end{cases}\quad(4)$$

Where: $\zeta^{(*)}=(\zeta_1,\zeta_1^*,\ldots,\zeta_l,\zeta_l^*)$ is a slack variable, w is the d-dimensional row vector, $C(C\le 0)$ is a penalty coefficient, deciding the balance between confidence risk and experience risk; $\nu\epsilon[0,1]$ is the upper bound of the proportion of error samples in the total number of training samples and the lower bound of the proportion of support vectors in the total number of training samples; unlike standard SVR, ϵ is present as the variable of optimization problem, and its value will be given as part of the solution.

2.2. ν-SVR model based on Gaussian loss function (ν-GSVR)

The standard SVR model with -insensitive loss function cannot deal with the random error (white noise) in normal distributed prediction sequences; so the standard SVR theoretically does not guarantee the accuracy of time series prediction problems containing white noise. The Gaussian function is in accord with the characteristics of normally distributed noise, thus it can minimize the effects of normally distributed noise as the loss function of SVR to a certain extent.

LSSVR (Least Squares Support Vector Regression) uses the ν-insensitive function as the loss function, uses the sum of the squares of the slack variables and changes the inequality constraints into equality constraints, which aims to simplify the solution of SVR. The ν-GSVR is the establishment of the relationship between slack variables and the loss function and normally distributed noise under the condition that the inequality constraints are not changed, and slack variable is also squared [20]. At this point, the ν-GSVR optimization problem is:

$$\begin{aligned}&\min_{w,b,\epsilon,\zeta^*}\quad \tau(w,\epsilon,\zeta^*)=\tfrac{1}{2}\|\omega\|^2\\ &+C\cdot(\nu\cdot\epsilon+\tfrac{1}{l}\textstyle\sum_{i=1}^{l}\tfrac{1}{2}(\zeta_i^2+\zeta_i^{*2}))\end{aligned}\quad(5)$$

$$s.t.\begin{cases}(w.x_i+b)-y_i\le\epsilon+\zeta_i\\ y_i-(w.x_i+b)\le\epsilon+\zeta_l^*\\ \zeta^{(*)}\ge 0,\epsilon\ge 0\end{cases}\quad(6)$$

Where: $\zeta^{(*)}=(\zeta_1,\zeta_1^*,\ldots,\zeta_l,\zeta_l^*)$ is the slack variable; w is d-dimensional row vector; $C(C>0)$ is the penalty coefficient; ν value range is [0,1]; ϵ is present as optimization variable; its value will be given as part of the solution. Literature [20] makes a detailed proof on the existence and uniqueness of ν-GSVR model solution.

The calculation steps for -GSVR model are:

Step 1: Suppose the known training set $T=\{(x_1,y_1),\ldots,(x_l,y_l)\}$, where $x_i\epsilon R^d$, $y_i\epsilon R$, $i=1,\ldots,l$;

Step 2: Select the appropriate positive ν and C, and the kernel function $K(x_i,y_i)$;

Step 3: Construct and solve the optimization problem, and the basic structure is shown in Figure 1;

$$\begin{aligned}\min_{a,a^*}w(\alpha,\alpha^*)=&\tfrac{1}{2}\textstyle\sum_{i,j=1}^{l}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)K(x_i,x_j)\\ &-\textstyle\sum_{i=l}^{l}(\alpha_i^*-\alpha_i)y_i+\tfrac{l}{2C}\textstyle\sum_{i=l}^{l}(\alpha_i^{*2}-\alpha_i^2)\end{aligned}\quad(7)$$

$$s.t.\begin{cases}\sum_{i=l}^{l}(\alpha_i^*-\alpha_i)=0\\ \sum_{i=l}^{l}(\alpha_i^*-\alpha_i)\le Cv\\ 0\le\alpha_i,\alpha_i^*\le C/l, i=1,\ldots,l\end{cases}\quad(8)$$

Get the optimal solution $\alpha^{(*)}=(\alpha,..,\alpha_l,\alpha_1^*,\ldots,\alpha_l^*)$

Step 4: Construct decision function

$$f(x)=\sum_{i=1}^{l}(\alpha_i^*-\alpha_i)K(x_i,x)+b\quad(9)$$

Where b is calculated as follows; select the two components a_j or a_k in the open interval $(0,C/l)$, then:

$$\begin{aligned}&\epsilon=\textstyle\sum_{i=1}^{l}(a_i^*-a_i)K(x_i\cdot x_j)+b-y_j\\ or\quad &\epsilon=y_k-\textstyle\sum_{i=1}^{l}(a_i^*-a_i)K(x_i\cdot x_j)-b\end{aligned}\quad(10)$$

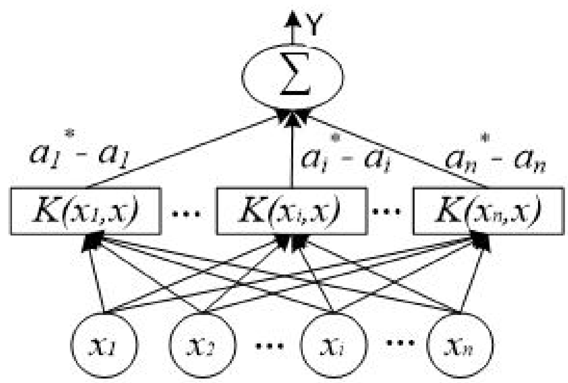

Fig. 1. The architecture of ν-GSVR

3. A new hybrid optimization algorithm

The parameter selection in SVR prediction model determines the generalization performance of the model, but there is no effective way to determine the optimal parameter combination, and there is cross error in traditional cross validation [17]. In view of this, through the improvement on the standard genetic algorithm (SGA) based on Cat Map and Cloud models, this paper presents CCGA to determine the parameter combinations of ν-GSVR model.

3.1. Standard Genetic Algorithm

By the joint action of reproduction, crossover, mutation and other genetic operators, GA makes the population continuously evolve and eventually arrive at the optimal solution. Due to the self-organization, self-adaptation, self-learning and essential parallel characteristics of GA, it has been widely used in parameter estimation, pattern recognition, machine learning, neural networks, industrial control and many other areas; however the shortcomings of the GA including the slow search for optimal solution or satisfactory solution and the easiness to fall in "pre-maturation" prevent its use in a wide range of applications. Based on this, the scholars from various countries have conducted in-depth studies on the encoding of GA, the determination of control parameters, the mechanism of action operators and have presented numerous improved methods [21–23], but in the process of application in large-scale complex parameter optimization, there are still defects in search speed and optimization accuracy. Based on the advantages of Cat Map such as good ergodicity and uniformity, resistance to fall into small cycles and fixed points, as well as the characteristics of cloud droplets of the Cloud models such as randomness and stable orientation, this paper carries out the following improvements on the SGA.

3.2. Initialization of parent population with Cat Map

Chaos optimization approach is a global optimization technique [24–26], using the nature of chaos such as ergodicity and initial value sensitivity. The current existing chaos optimization approaches have mostly used the Logistic Map as a chaotic sequence generator. The probability density of the chaotic sequence generated by the Logistic Map obeys the Chebyshev distribution similar to the 'bath tub' curve, and such a distribution hinders the global search capability and efficiency of the algorithm. To overcome the shortcomings of the Logistic Map, the Cat Map with the advantages of good ergodicity and uniformity, its resistance to fall into small cycles and fixed points, is used in the initialization of the parent population of GA.

3.2.1. Cat Map

The two-dimensional Cat Map equation is:

$$\begin{cases} x_{n+1} = (x_n + y_n) mod 1 \\ y_{n+1} = (x_n + 2y_n) mod 1 \end{cases} \tag{11}$$

Where: $x \bmod 1 = x - [x]$; now, the two Lyapunov indices of Cat Map are $L_1 = ln((3+\sqrt{5})/2) > 0$ and $L_1 = ln((3+\sqrt{5})/2) < 0$, indicating that Cat Map has chaotic characteristics.

3.2.2. Analysis of chaotic characteristics of Cat Map and Logistic Map

$$x_{n+1} = u \cdot x_n \cdot (1 - x_n) \tag{12}$$

Where x_n is the n^{th} iteration value of variable x and u is a control parameter. When $u = 4$, the system completely is in a chaotic state [27, 28]. Furthermore x0 can take any initial values except for 0.25, 0.5 and 0.75 within the interval of (0,1).

To analyze the chaotic characteristics of Cat Map and Logistic Map, suppose the initial values of Logistic Map are 0.2, 0.4, 0.6 and 0.8 whilst the initial values of Cat Map are 0, 0.2, 0.4, 0.6, 0.8 and 1. After 50,000 iterations the distribution graph for the two maps within the range of [0,1] was obtained. The statistics for the values with the maximum and minimum number of occurrences were carried out and the results obtained are shown in Table 1:

The statistics in Table 1 show the maximum frequency of the Logistic Map within (0;0.01] and [0.99;1) exceeds 3,000 times, while the average value frequency within (0.01,0.99) interval is only about 500 times. So, when the optimal solution is in the midrange, the application of the Logistic Map in the generation of the parent population makes it difficult to ensure an efficient search for the optimal solution. Meanwhile the maximum frequency of occurrence for the Cat Map is ≈560 times and the minimum frequency is ≈440 times, this indicates that the Cat Map is more evenly distributed. Secondly the initial values of Cat Map can range between 0 and 1, which is not permissible for the Logistic Map. Therefore the Cat Map has better chaotic distribution characteristics, and its application in the initialization of parent population for GA is able to better able to maintain the population diversity for an ergodic search theoretically.

3.3. Crossover and Mutation based on Cloud model

Cloud models have the characteristics of random and bias stability [27]. The random property avoids a local minimum solution, and the bias stability aids the positioning of the global optimum. Thus, introduction of Cloud models into the GA can reduce optimization time-consuming and improve the ability to avoid falling into local minimums when used with both the basic cloud generated algorithm and the Y-condition cloud generated algorithm which perform the mutation operation and the crossover operation respectively.

Suppose T is the language value in the domain u, map $C_T(x) : u \to [0,1]$, $\forall x \in u$, $x \to C_T(x)$, then the distribution of $C_T(x)$ on u is called the membership cloud under T. In the case of obeying the normal distribution, $C_T(x)$ is known as the normal Cloud models [28]. The overall characteristics of the Cloud models can be represented by

Table 1. Comparison on chaotic distribution of Logistic Map and Cat Map

	Logistic map				Cat map					
initial value	0.2	0.4	0.6	0.8	0	0.2	0.4	0.6	0.8	1
max frequency	3222	3241	3239	3284	559	563	547	556	546	559
min frequency	283	290	289	291	450	440	452	437	458	450

the three digital features including desired E, entropy S and hyper entropy H.

E is the expectation of spatial distribution of cloud droplet in the domain as well as the point that is the most able to represent the qualitative concept. S represents the measurable granularity of the qualitative concept, and the greater the entropy S the larger the concept. H is the uncertain measurement of entropy and is jointly determined by the randomness and fuzziness of the entropy.

The Cloud models have the characteristics of the uncertainty with certainty and stability with change, and thus reflect the basic principle of the evolution of a species in nature. The Cloud models parameter E represents the parent's good individual genetic characteristics and the offspring's inheritance from the parent. The entropy S and hyper-entropy H indicate the uncertainty and fuzziness of the inheritance process, giving the mutation characteristics of the species during the evolutionary process.

The algorithm or hardware for the generation of cloud droplets is called the cloud generator [28]. The basic cloud generator and Y-condition cloud generator are calculated as follows:

Normal cloud generator: Input the three digital features including E, S and H, as well as n, the number of cloud droplets; output the quantitative values of n cloud droplets and the certainty of the representative concept.

1. To generate $S1$, a normal random number with the expectations of S and the standard deviation of H;

2. To generate x_i, the normal random number with the expectations of E and the standard deviation of $S1$;

3. Suppose x_i should be a specific quantitative value of qualitative concept, called cloud droplets, the calculation $\mu_i = e^{-(x_i-E)/2*(S1)^2}$ is conducted:

4. Suppose μ_i should be the certainty for the qualitative concept of x_i;

5. $\{x_i, \mu_i\}$ completely reflect the conversion process from qualitative to quantitative;

6. Repeat steps 1 to 5, till the n cloud droplets (x_i, μ_i) are all generated.

Y-condition cloud generator: input the three digital features including E, S, H, and n, the number of cloud droplets; output the quantitative values of n cloud droplets and the certainty of the representative concept.

1. To generate S^*, a normal random number with the expectations of S and the standard deviation of H;

2. Calculate cloud droplet x_i, $x_i = E \pm S\sqrt{-2ln(\mu_0)}$

3. Repeat steps 1 to 2, till the n cloud droplets (x_i, μ_0) are all generated.

3.4. CCGA

The shortcomings of standard GA are overcome by adopting real-coding, using Cat Map for the generation of initial population, using the Y-condition cloud of normal Cloud models to achieve crossover operation and using the basic cloud to achieve mutation operation. The computing process of CCGA is as follows:

Step 1: Generation of initial population by Cat Map: use equation (11) to produce the initial population, in order to make it be as uniformly distributed as possible in the solution space, overcome the heterogeneity of the initial population generated by random sequence and improve the diversity of the population.

Step 2: Select each individual genes as the ν-GSVR model parameters to calculate the individual fitness.

Step 3: Selection, copying and migration.

1 Copy the best individual to the next generation;

2 Select an elite population, and copy it;

3 Replace the worst individual by a randomly generated individual.

Step 4: Y-condition cloud crossover.

1 Randomly generate membership μ_0 according to the uniform distribution;

2 If μ_0 is less than crossover probability p_c, E is be calculated by:

$$E = \frac{fitness(i)}{fitness(i)+fitness(j)} \cdot x_i + \frac{fitness(j)}{fitness(i)+fitness(j)} \cdot x_j \tag{13}$$

Where: x_i and x_j respectively are the parent individuals of the crossover operation; $fitness(i)$ and $fitness(j)$ are the fitnesses of the two parent individuals;

3 S = variable search range/ $c1$;

4 H=S/$c2$;

5 Use the Y-condition cloud generator to produce two offspring individuals.

Step 5: Mutation of normal cloud.

1 take the original individual as E;

2 S = variable search range/$c3$;

3 H=S/$c4$;

4 Activate normal cloud generator to produce cloud droplets (x_i, μ_i);

5 If μi is less than the mutation probability p_m, x_i will be regarded as the post-mutation individual.

Step 6: Go to Step 2, till it meets the optimal stopping conditions, where $c1 - c4$ is the control parameter.

3.5. Analysis of effects of control parameter value

According to "3δ" rule [29], S value determines the horizontal width of cloud cover, that is, it determines the search range of individual during crossover and mutation operation. The values of $c1$ and $c3$ generally are recommended to be within the interval [6,6P] where P is size of the population. To a certain extent too great a value for H may lead to the loss of "bias stability" in the Cloud models. However too small a value for H may lead to the loss of "randomness". Therefore, the values of $c2$ and $c4$ are recommended to be within [1, 10]. Although S and H are the important parameters for the Cloud models, after several generations, the randomness of E and μ conceals the difference in the evolution result due to the difference between the values. That is the difference in value among the control parameters $c1 - c4$ within a certain range do not have a major effect on the final evolution performance. Comprehensively considering the optimization, speed and accuracy of an algorithm, this experiment takes $c1 = c3 = 3P$, $c1 = c4 = 6$. Of course, in the process of practical application the values of parameters $c1 - c4$ can be properly determined according to a variable search range, population size and search accuracy.

4. CCGA in Selecting Parameters of ν-GSVR Model

The procedure of the CCGA-ν-GSVR model is as follows: first step, Normalize the training data set and the value range of parameters; second step, use the evolution individuals generated by CCGA as ν-GSVR parameters; third step, according to the normalized training data, conduct training to calculate the regression values; forth step, determine whether the current parameter combinations meet the parameter optimization accuracy requirements, if the requirement is met, stop the parameter optimization. The optimal parameter combination obtained will be used as the parameter of ν-GSVR for traffic flow prediction. Otherwise, repeat steps 2 to 4 until the accuracy of parameter optimization is met. The specific calculation process is shown in Figure 2.

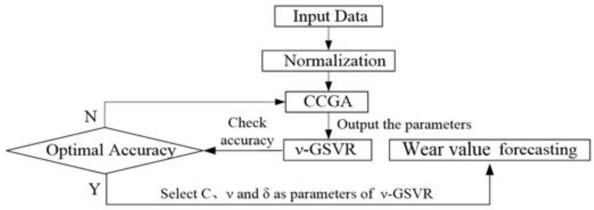

Fig. 2. The process of CCGA-ν-GSVR

To ensure the CCGA optimization efficiency, this paper uses the reciprocal of the root mean square error as fitness function, and then the fitness function is calculated as follows.

$$fitness = \frac{1}{\sqrt{RMSE}} = \frac{1}{\sqrt{\frac{1}{n}\sum_{t=1}^{n}(Y^*(t) - Y(t))^2}} \quad (14)$$

Where n is the number of input samples, $Y*(t)$ is the regressed value for the sample data, $Y(t)$ is the actual value for the sample data.

In view of the radial basis function's good ability to learn in the application process of SVR [14–16, 30], this paper selects radial basis function as the kernel function of ν-GSVR model. The radial basis function is expressed as the following equation.

$$K(x_i x_j) = exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\delta^2}\right) \quad (15)$$

5. The numerical test

Wear value is characteristic of cutting tools, and the distribution in time is continuous, that is, the wear value at the next time of CNC turning tools is intrinsically linked with the wear value in the first few intervals of CNC turning tools. Meanwhile, the wear value is a part of the larger wear value network, and the wear value will be Wear value is characteristic of cutting tools, and the distribution in time is continuous, that is, the wear value at the next time of CNC turning tools is intrinsically linked with the wear value in the first few intervals of CNC turning tools. Meanwhile, the wear value is a part of the larger wear value network, and the wear value will be influenced by the wear value in the previous time CNC turning tools. In addition, the spindle revolution, cutting depth and feed rate affect the turning tool wear value to a large extent. Therefore, we can select wear value for the first few intervals of the section, the previous time wear value and turning tool conditions at the time to predict the wear value of the turning tool for the next time step. influenced by the wear value in the previous time CNC turning tools. In addition, the spindle revolution, cutting depth and feed rate affect the turning tool wear value to a large extent. Therefore, we can select wear value for the first few intervals of the section, the previous time wear value and turning tool conditions at the time to predict the wear value of the turning tool for the next time step.

Taking the data detected from CNC turning tools of Taishan Vocational Training Site for the numerical test. Assuming that t represents the current time, $Y(t+1)$ represent the wear value for the next time step, while $X1(t)$, $X2(t)$ and $X3(t)$ represent the wear value for the three upstream sections. The turning tool condition is set as the sixth influencing factor, in this case $X4(t)$, which is quantified as 1 for low spindle revolution, 0.75 for medium low spindle revolution, 0.5 medium spindle revolution, 0.25 for medium high spindle revolution, and 0 for high spindle revolution. The result of tentative calculation indicates that the forecasting performance is best, when the first two times intervals of the wear value are select as the input vector. At this point, we obtained the six influencing factors of the wear value $Y(t+1)$, $X = \{Y(t-1), Y(t), X_1(t), X_2(t), X_3(t), X_4(t)\}$.

Turning data was detected by Mr. Chen of Taishan Vocational Training Site . The sample window of 10 minutes gave 90 sets of data. The first 72 data from the first four times were used as training samples for the model. The remaining 18 data from the machining were used to test the prediction accuracy of the model.

The proposed CCGA-ν-GSVR model has been implemented in Matlab7.1 programming language. The experiments are made on a 1.81 GHz Core(TM) 2 CPU personal computer with 2.0GB memory under Microsoft Windows XP professional. Model initialization: C value ranges [0.01, 1000], ν value ranges [0.01,1], δ value ranges [0.01,1], genetic population size *popsize*=60, maximum evolution generation $G_{max} = 50$, crossover probability P_c=0.3, mutation probability P_m=0.8, relative error of adjacent-generation optimal individual fitness E=0.00001. The combination of optimal parameters of ν-GSVR are obtained by the CCGA, C=584.5, ν=0.87 and δ=0.21. The developed model was used to forecast the wear value. The chart for the comparison between actual wear value and model fitness value is described in Figure 3. The actual wear value and model prediction results are shown in Table 2. The results show that the fitness results are basically consistent with the actual wear value variation curve, where the root mean square error of regressed value is less than 3, the root mean square error of prediction is less than 13, and the prediction accuracy satisfies the actual application requirements.

To compare the parameter optimization efficiency of CCGA, this paper also used SGA to optimize the parameters of ν-GSVR model. Using the same computer, the same population size, crossover probability and mutation probability, the statistical results for the fitness values and genetic algebra of the two algorithms are plotted against time in Figure 4.

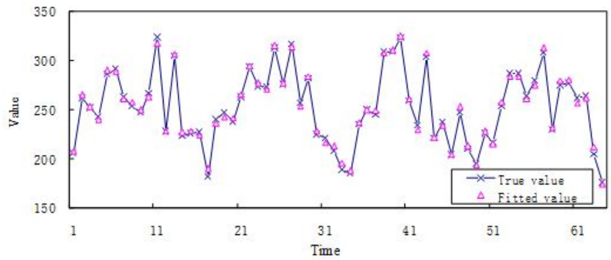

Fig. 3. Fitting chart of CCGA-ν-GSVR

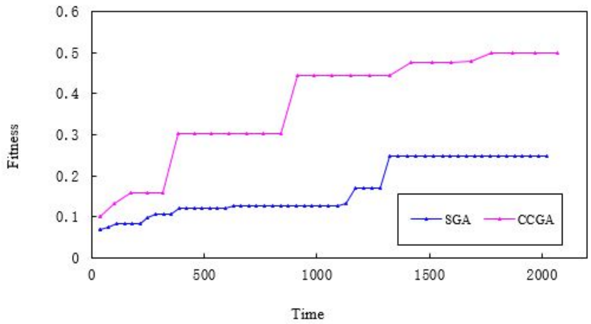

Fig. 4. Comparison of training.

The analyses as shown in Figure 4 illustrate that, within the running time of 2000s, the use of Cat Map for the initialization of parent population and the introduction of cloud crossover and mutation operations, the running time of CCGA for each iteration is more time consuming than SGA. This extra computational time results in the CCGA solving fewer generations than the SGA model for a given period of time. However, based on the ergodic characteristics of Cat Map, the maximum fitness value of the CCGA search for optimal solution is double that of the SGA algorithm. The introduction of cloud theory-based genetic manipulation reduced the range of maximum fitness value and minimum fitness value so that more individuals from CCGA search can be concentrated in the range of the optimal solution. The overall efficiency to find the optimal solution using CCGA is significantly higher than that of SGA algorithm, indicating that CCGA is more suitable for the optimization of ν-GSVR model parameters.

To compare the performance of the CCGA-ν-GSVR model in wear value forecasting, this paper selects the ARIMA model, the PSO-BP neural network model pro-

Table 2. Forecasting result of CCGA-ν-GSVR model.*

Peak period	$X_1(t)$	$X_2(t)$	$X_3(t)$	$X_4(t)$	Y(t-1)	Y(t)	Y(t+1) Actual	Y(t+1) Forecasting	Y(t+1) Error
17:30	111	228	124	0	176	182	195	202	7
17:40	107	238	119	0.25	182	195	259	250	-9
17:50	135	251	119	0	195	259	188	196	8
18:00	126	241	148	0	259	188	233	219	-14
18:10	128	246	148	0.25	188	233	221	213	-8
18:20	148	264	138	0.5	233	221	267	258	-9
18:30	152	263	162	0.5	221	267	291	284	-7
18:40	146	294	142	0	267	291	310	297	-13
18:50	139	263	161	0.25	291	310	295	286	-9
19:00	141	270	145	0.25	310	295	228	220	-8
19:10	147	267	132	0	295	228	266	254	-12
19:20	123	246	142	0	228	266	282	271	-11
19:30	116	255	123	0	266	282	233	240	7
19:40	130	238	137	0	282	233	237	231	-6
19:50	101	233	132	0	233	237	235	221	-14
20:00	102	231	98	0.25	237	235	180	196	16

*: "17:30" denotes the 17:30 on 4 April 2014, and so on.

posed by Ye [9], the WD-SVM prediction model proposed by Zhu [15], and the improved CCGA-ν-SVR prediction model based on the GSVMR model proposed by Ren [30] as comparison models. The four models were calculated in the same computer using Matlab 7.1. Taking into account the fact that the increase in optimization time improves the optimization effect, the model parameters should be selected for each model to ensure the same maximum optimization time is achieved. In order to ensure the selection accuracy of SVR parameters the root mean square error was used to compare the performance of each method.

The comparison of the real wear value and the forecasted results of the various modeling methods are given in Table 3.

It can be seen from Table 3 that the SVR model is superior to ARIMA and PSO-BP in the capacity of the wear value forecasting. In terms of noise handling during in the prediction process the WA-SVM model firstly carried out noise reduction on the traffic flow sequence through wavelet analysis; eliminating the effects of random errors in the sequence of traffic flow in the SVM model. This has proven beneficial as the prediction accuracy was significantly superior to that of unfiltered CCGA-ν-SVR model. From the comparison of prediction results between CCGA-ν-SVR model and CCGA-ν-GSVR model, it can be seen that the proposed CCGA-ν-GSVR model also played a role in weakening these random errors, and the prediction accuracy showed a better correlation with the measured results than that of the CCGA-ν-SVR model. Although the noise reduction effect is inferior to wavelet analysis, the computational complexity of wavelet analysis is not conducive to the actual operation. For real-time wear value prediction, the real-time performance of algorithms is the determining factor for a given accuracy condition. In this sense, the proposed CCGA-ν-GSVR model is more suitable for the short-term wear value prediction of CNC turning tools.

6. Conclusions

Cost and wear of turning tools are important keys to the prediction process. The real-time accurate prediction for the wear value of CNC turning tools can effectively reduce the stock of turning tools, and the labor costs on quality inspection of workpieces. In this paper, a new approach for predicting short-term wear value based on the CCGA-ν-GSVR method is proposed. based on the CCGA-ν-GSVR method is proposed. The numerical test based on detected data is used for elucidating the forecasting accuracy of proposed method. The prediction results of the proposed CCGA-ν-GSVR forecasting scheme are compared to those of the CCGA-ν-GSVR, CCGA-ν-SVR, WA-SVR, PSO-BP and ARIMA. The numerical test indicates that the proposed scheme obtain better prediction result and outperforms the five competing models. In short, the proposed forecasting scheme is a valid approach for short-term wear value predication.

In future researches, more influence factors should be

Table 3. Forecasting result of CCGA-ν-GSVR model.*

Peak period	Actual	CCGA-ν-GSVR	CCGA -ν- SVR	WA-SVR	PSO-BP	ARIMA
17:30	195	202	204	189	221	216
17:40	259	250	246	266	237	236
17:50	188	196	176	195	161	208
18:00	233	219	251	220	201	249
18:10	221	213	211	215	193	245
18:20	267	258	279	274	296	243
18:30	291	284	276	285	264	322
18:40	310	297	321	300	345	332
18:50	295	286	312	301	266	328
19:00	228	220	210	235	258	208
19:10	266	254	279	255	298	294
19:20	282	271	299	290	249	302
19:30	233	240	224	239	258	256
1940	237	231	225	248	209	203
19:50	235	221	218	222	267	261
20:00	180	196	189	168	226	194
RMSE		**10.31**	**17.39**	**8.89**	**30.51**	**37.04**

considered, and other hybrid optimization methods should study for more appropriate parameters of ν-GSVR model, for improving the performance of short-term wear value forecasting.

References

[1] N Shang, M Qin, and Y Wang. A BP neural network method for short-term traffic flow forecasting on crossroads. *Computer Applications and Software*, 23(2):32–33, 2006.

[2] D R Huang, J Song, D C Wang, J Cao, and W Li. Forecasting Model of Traffic Flow Based on ARMA and Wavelet Transform. *Dynamics of Continuous Discrete & Impulsive Systems*, 42(36):1298–1302, 2006.

[3] C Han, S Song, and C Wang. A real-time short-term traffic flow adaptive forecasting method based on ARIMA model. *Journal of System Simulation*, 7(16):1530–1534, 2004.

[4] Yibing Wang, Markos Papageorgiou, and Albert Messmer. Real-time freeway traffic state estimation based on extended Kalman filter: A case study. *Transportation Science*, 41(2):167–181, 2007.

[5] Hong Li Lan. Short-term traffic flow prediction for highway tunnel based on fuzzy clustering analysis. *Computer Applications and Software*, 27(1):151–153, 2010.

[6] D Hu, J Xiao, and C Che. Lifting wavelet support vector machine for traffic flow prediction. *Application Research of Computers*, 24(8):276–278, 2007.

[7] Hong Xie and Zhonghua Liu. Short-term traffic flow prediction based on embedding phase-space and blind signal separation. In *2008 IEEE International Conference on Cybernetics and Intelligent Systems, CIS 2008*, 2008.

[8] N Shang, M Qin, Y Wang, Z Cui, Y Cui, and Y Zhu. A BP neural network method for short-term traffic flow forecasting on crossroads. *Computer Applications and Software*, 23(2):33–35, 2006.

[9] Hong Sen Yan and Duo Xu. An approach to estimating product design time based on fuzzy ν-support vector machine. *IEEE Transactions on Neural Networks*, 18(3):721–731, 2007.

[10] Hanli Liu, Chenghu Zhou, Axing Zhu, and Lin Li. Multi-population genetic neural network model for short-term traffic flow prediction at intersections. *Acta Geodaetica et Cartographica Sinica*, 38(4):363–368, 2009.

[11] Che Chang Hsu, Kuo Shong Wang, and Shih Hsing Chang. Bayesian decision theory for support vector machines: Imbalance measurement and feature optimization. *Expert Systems with Applications*, 38(5):4698–4704, 2011.

[12] Che Chang Hsu, Kuo Shong Wang, Hung Yuan Chung, and Shih Hsing Chang. A study of visual behavior of multidimensional scaling for kernel perceptron algorithm. *Neural Computing and Applications*, 26(3):679–691, 2015.

[13] James W. Taylor. Short-term load forecasting with exponentially weighted methods. *IEEE Transactions on Power Systems*, 27(1):458–464, 2012.

[14] Jianrong Cao and Anni Cai. A robust shot transi-

tion detection method based on support vector machine in compressed domain. *Pattern Recognition Letters*, 28(12):1534–1540, 2007.

[15] Sheng-Xue Zhu, Jun Zhou, and Xu Bao. Short-term Traffic Forecast Based on WD and SVM. *University of Science and Technology of Suzhou (Engineering and Technology)*, 20(3):80–85, 2007.

[16] Qi Wu and Hong Sen Yan. Forecasting method based on support vector machine with Gaussian loss function. *Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS*, 15(2), 2009.

[17] Hong Sen Yan and Duo Xu. An approach to estimating product design time based on fuzzy ν-support vector machine. *IEEE Transactions on Neural Networks*, 18(3):721–731, 2007.

[18] B Abdelhadi, A. Benoudjit, and N. Nait-Said. Self-adaptive genetic algorithms based characterization of structured model parameters. In *Proceedings of the Annual Southeastern Symposium on System Theory*, volume 2003-Janua, pages 181–185, 2003.

[19] Bernhard Schölkopf, Alex J. Smola, Robert C. Williamson, and Peter L. Bartlett. New support vector algorithms. *Neural Computation*, 12(5):1207–1245, 2000.

[20] Qi Wu, Rob Law, Edmond Wu, and Jinxing Lin. A hybrid-forecasting model reducing Gaussian noise based on the Gaussian support vector regression machine and chaotic particle swarm optimization. *Information Sciences*, 238:96–110, 2013.

[21] Zong Fei Zhang. Novel improved quantum genetic algorithm. *Computer Engineering*, 36(6):181–183, 2010.

[22] Changkyu Choi and Ju-Jang Lee. Chaotic local search algorithm. *Artificial Life and Robotics*, 2(1):41–47, mar 1998.

[23] Jun Feng Yao, Chi Mei, and Xiao Qi Peng. Application research of the chaos genetic algorithm(CGA) and its evaluation of optimization efficiency. *Zidonghua Xuebao/Acta Automatica Sinica*, 28(6):935–942, 2002.

[24] Dun Wei Gong, Mei Qiang Zhu, Xi Jin Guo, and Ming Li. Genetic algorithm based on chaotic mutation to deal with premature convergence. *Kongzhi yu Juece/Control and Decision*, 18(6):686–689, 2003.

[25] Fang Wang, Yong-Shou Dai, and Shao-Shui Wang. Modified chaos-genetic algorithm. *Computer Engineering and Applications*, 46(6), 2010.

[26] Qingzhang Lü, Guoli Shen, and Ruqin Yu. A chaotic approach to maintain the population diversity of genetic algorithm in network training. *Computational Biology and Chemistry*, 27(3):363–371, 2003.

[27] Li DY, Meng HJ, and Shi XM. Membership clouds and membership clouds generator. *Journal of Computer Research and Development*, 2005.

[28] Xing-Sheng Li and De Yi Li. A new method based on cloud model for discretization of continuous attributes in rough sets. *Pattern Recognition and Artificial Intelligence*, 16(1):33–37, 2003.

[29] Bo Liu, Ling Wang, Yi Hui Jin, Fang Tang, and De Xian Huang. Improved particle swarm optimization combined with chaos. *Chaos, Solitons and Fractals*, 25(5):1261–1271, 2005.

[30] Qi-Liang Ren, Xiao Song Xie, and Qiyuan Peng. GSVMR model on short-term forecasting of city road traffic volume. *Journal of Highway and Transportation Research and Development*, 2(52):135–138, 2008.