machines

MDPI

Article

Investigation of an Optimized Linear Regression Model with Nonlinear Error Compensation for Tool Wear Prediction

Lihua Shen [1,2], Baorui Du [3], He Fan [1,2] and Hailong Yang [3,*]

[1] College of Mechanical and Electrical Engineering, Shenyang Aerospace University, Shenyang 110136, China; 20190016@sau.edu.cn (L.S.)
[2] Key Laboratory of Rapid Development & Manufacturing Technology for Aircraft, Shenyang Aerospace University, Ministry of Education, Shenyang 110136, China
[3] The Institute of Engineering Thermophysics, Chinese Academy of Sciences, Beijing 100094, China
* Correspondence: yanghailong22@iet.cn

Abstract: To solve the problem of insufficient accuracy in tool wear process modeling and Remaining Useful Life (RUL) estimation, this study proposes a two-stage prediction method. Firstly, a linear prediction benchmark model is constructed: Support Vector Regression (SVR) is used to preliminarily model the tool wear process, obtaining initial prediction results and their error distribution. Building on this foundation, an Autoencoder (AE) is introduced to establish a nonlinear mapping relationship for the errors, achieving effective compensation of the SVR prediction results and establishing the SVR–AE prediction model. To further enhance model performance, the Ant Colony Optimization (ACO) algorithm is utilized to optimize three key parameters: the number of training epochs, batch size, and hidden layer dimensions, ultimately establishing the ACO–SVR–AE optimization model. Experimental validation demonstrates that on the PHM2010 dataset, compared to the Support Vector Regression (SVR) and Autoencoder (AE) models, the proposed method achieves average reductions of 26.1% in Mean Squared Error (MSE) and 14.5% in Mean Absolute Error (MAE). Compared to traditional random forest and neural network models, the MSE and MAE show average reductions of 32.3% and 25.3%. By combining linear modeling with nonlinear error compensation, this method provides an integrated optimization approach to prediction tasks in complex industrial scenarios.

Keywords: tool wear prediction; linear regression; Support Vector Regression (SVR); Autoencoder; Ant Colony Optimization algorithm (ACO); error compensation

check for updates

Academic Editor: Angelos P. Markopoulos

Received: 11 March 2025
Revised: 17 April 2025
Accepted: 18 April 2025
Published: 24 April 2025

Citation: Shen, L.; Du, B.; Fan, H.; Yang, H. Investigation of an Optimized Linear Regression Model with Nonlinear Error Compensation for Tool Wear Prediction. *Machines* **2025**, *13*, 355. https://doi.org/10.3390/machines13050355

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

In contemporary manufacturing, tool wear represents a critical issue that demands considerable attention. Excessive tool wear not only compromises machining precision but also diminishes product yield rates, thereby substantially escalating production costs. Tool wear prediction is inherently challenging due to the dynamic coupling of time-varying cutting parameters, heterogeneous workpiece materials, and environmental vibrations. Traditional mechanistic models, while interpretable, may encounter challenges in fully capturing the nonlinear interdependencies among these factors and could exhibit constrained real-time adaptability to dynamic degradation patterns, especially when handling asynchronous multi-sensor data streams. Consequently, developing a novel tool wear prediction method that improves accuracy remains a critical need in industrial applications.

Currently, the mainstream methods for predicting tool wear can be categorized into three groups: physical model-based approaches [1], data-driven methods [2], and intelli-

gent algorithm-based techniques [3]. The physical model-based method predicts the wear state of tools by analyzing and modeling physical phenomena during the cutting process, such as cutting force and temperature. Zhang et al. (Mechanical Systems and Signal Processing, 2023) [4] established an accurate physical model of tool wear based on the relationship between cutting force and tool wear, achieving effective prediction of tool wear. Yang et al. [5] constructed a model correlating tool wear rate with cutting parameters based on wear influencing factors and accurately predicted the tool wear curve by stretching the abscissa. Zhang et al. (Metals, 2023) [6] proposed a tool wear rate model based on the positive feedback relationship between tool geometry and wear rate. While these models provide interpretable insights, they rely on assumptions that may reduce their applicability to complex real-world conditions. Data-driven methods, including Convolutional Neural Networks (CNNs) and power signal-based models, have gained traction due to their ability to extract patterns from multi-sensor data. For instance, Zhang et al. (The International Journal of Advanced Manufacturing Technology, 2024) [7] demonstrated the effectiveness of CNNs in mapping vibration and current signals to tool wear states. Similarly, Wang et al. (The International Journal of Advanced Manufacturing Technology, 2023) [8] achieved promising results using power signals. However, their performance appears dependent on the availability of comprehensive labeled datasets, and they face challenges in generalizing across diverse machining scenarios. Recent advancements, such as the integration of physical and data-driven models by Fan et al. [9], aim to mitigate these limitations but still lack robustness in noisy environments. Zhou et al. [10] developed a monitoring method employing a Multi-scale Edge-labeled Graph Neural Network (MEGNN), though its technical emphasis resides in the feature learning potential of the deep graph architecture for few-shot classification scenarios. Intelligent algorithms, including deep neural networks and attention-enhanced LSTM networks, further push the boundaries of prediction accuracy. Zhao et al. [11] utilized multi-sensor fusion with deep learning to achieve high precision, while Tian et al. [12] improved temporal feature extraction using attention mechanisms. Despite their success, computational complexity remains a consideration, and there are overfitting risks. Zhang et al. (Sensors (Basel, Switzerland), 2016) [13] proposed a wireless triaxial accelerometer method using wavelet denoising and NFNs to predict tool wear and remaining life, showing improved accuracy over traditional neural networks. However, it relies on single vibration sensors without integrating multi-source data (e.g., cutting forces), and subsequent studies could explore enhanced multi-sensor fusion strategies. Recent studies, such as Bampoula et al. [14], highlight the potential of LSTM autoencoders and transformer encoders for condition monitoring, yet their application to tool wear prediction remains underexplored. Similarly, Cakir et al. [15] compared machine learning algorithms for predictive maintenance but overlooked the unique challenges of tool wear dynamics. Wu et al. [16] demonstrated the efficacy of transformers in time-series forecasting, offering insights for improving sequential data modeling in tool wear contexts. Support Vector Regression (SVR)-based methods, optimized via evolutionary algorithms [17] or hybridized with particle filters [18], have shown robustness in specific scenarios. For example, Kong et al. [19] combined kernel PCA with v-SVR to enhance feature fusion, while Benkedjouh et al. [20] employed nonlinear feature reduction for life prediction. Nevertheless, their application to tool wear prediction warrants deeper investigation to fully exploit the nonlinear relationships and dynamic couplings inherent in tool wear data. Chen et al. [21] developed a DBO-optimized 1DCNN–LSTM model for high-accuracy prediction of bearing surface roughness, demonstrating the strengths of deep learning models in dynamic signal processing, while requiring substantial labeled data and complex network architectures.

Despite remarkable progress in recent studies, critical aspects of tool wear research remain underexplored, First, existing models predominantly focus on either linear (e.g.,

Support Vector Regression, SVR) or nonlinear (e.g., deep learning) relationships, with limitations in exploiting their synergistic potential, thereby limiting the characterization of complex tool wear dynamics. Second, although data-driven methods might benefit from more sophisticated signal processing techniques, their prediction accuracy could still be compromised under real-world machining conditions. Additionally, data-driven methods are highly susceptible to environmental and operational noise, leading to compromised prediction accuracy in practical machining scenarios. Conventional neural network training typically employs gradient-based algorithms, which may face challenges related to local optima convergence and computational efficiency, especially when operating under noisy or non-stationary conditions. These limitations collectively hinder the development of robust and efficient prediction systems. Mishra et al. [22] proposed an unsupervised GMM approach that demonstrates high versatility in tool condition clustering, but this method relies on physical features and did not quantify prediction errors.

The integration of Support Vector Regression (SVR) and Autoencoder (AE) in this work is motivated by two critical research gaps in tool wear prediction: (1) Existing methods predominantly focus on either the linear relationship (e.g., SVR) or nonlinear modeling (e.g., deep learning) and have not been fully exploited in current implementations to address the intertwined linear–nonlinear dynamics of tool wear caused by time-varying cutting parameters, material heterogeneity, and environmental noise. While standalone linear models lack the capacity to capture complex wear patterns, purely nonlinear approaches often overfit limited industrial data. (2) Conventional Neural Network parameter optimization relies on gradient-based methods, which suffer from local optima and inefficiencies under noisy, varying operational conditions. Existing prediction methods often rely on single models that predominantly focus on either linear or nonlinear modeling alone, thereby failing to comprehensively capture complex data characteristics and resulting in limited prediction accuracy. To address this limitation, this study proposes a hybrid model integrating Support Vector Regression (SVR) and Autoencoder (AE). The proposed model explicitly captures linear patterns in data through SVR while leveraging the AE to extract nonlinear latent features. Their synergistic interaction effectively resolves the critical issue of inadequate representation of combined linear and nonlinear features within multidimensional data. Furthermore, to enhance model performance, we incorporate the Ant Colony Optimization (ACO) algorithm for parameter optimization. Experimental results demonstrate high prediction accuracy, which validates the feasibility of the proposed method.

In view of this, the study focuses on the in-depth exploration of complex multi-sensor data, specifically triaxial cutting force signals (in the X, Y, and Z directions), vibration signals, and acoustic emission signals, aiming to improve prediction accuracy in dynamic machining environments. To better capture both the nonlinear and the linear characteristics of the tool wear dataset, this study proposes a two-stage prediction framework: Ant Colony Optimization–Support Vector Regression–Autoencoder (ACO–SVR–AE), which synergistically integrates nonlinear error compensation with swarm intelligence optimization. The rationale for combining linear Support Vector Regression (SVR) and Autoencoder lies in their complementary capabilities: Linear SVR is adept at capturing linear relationships and exhibits robustness with limited training data, whereas Autoencoder effectively models intricate nonlinear patterns and mitigates input noise. The framework initially leverages Support Vector Regression (SVR) to estimate linear trends, followed by Autoencoder-based compensation of nonlinear residuals. Additionally, the Ant Colony Optimization (ACO) algorithm is introduced to optimize Autoencoder parameters, significantly improving convergence efficiency and predictive accuracy. Extensive experiments on the PHM2010 high-speed milling dataset validate the proposed model against multiple benchmarks, including linear Support Vector Regression (SVR), standalone Autoencoder, and conven-

tional models (e.g., random forest, neural networks). The results highlight its exceptional performance in both prediction precision and noise robustness, demonstrating practical potential for industrial applications.

The paper is organized as follows: Section 1 introduces the research background of tool wear prediction and identifies limitations of existing approaches. Section 2 provides theoretical foundations for Support Vector Regression (SVR), Autoencoder, and ACO. Section 3 details the proposed ACO–SVR–AE framework, including its two-stage linear-nonlinear modeling architecture and parameter optimization strategy. Section 4 validates the model's superiority through comprehensive experiments on PHM2010 datasets, demonstrating significant improvements in MAE/MSE metrics and noise robustness. Section 5 concludes with research contributions and future directions.

2. Theoretical Background

This section will introduce the theories related to linear Support Vector Regression, the Autoencoder neural network, and the Ant Colony Optimization algorithm. Support Vector Regression (SVR), specifically Support Vector Regression (SVR) based on the linear kernel function, uses the insensitive loss function and the kernel function to seek a balance between the empirical risk and the model complexity. It determines the hyperplane parameters by solving a convex quadratic programming problem to achieve regression prediction. Autoencoder, as an unsupervised learning algorithm, goes through the encoding and decoding processes, and extracts the features of tool wear data using reconstruction loss for learning. The Ant Colony Optimization (ACO) algorithm simulates the foraging behavior of ants. It optimizes the key parameters through the pheromone update and positive feedback mechanism to improve the performance of the prediction model. Figure 1 illustrating the overall data flow from input, to SVR, to Autoencoder and ACO.

Key Components Explained:

(1) Input and Feature Extraction. Raw sensor signals are transformed into multi-dimensional features (time domain, frequency domain, and time–frequency domain representations) to characterize tool wear patterns.
(2) Stage 1—Linear SVR Prediction: A linear kernel SVR generates initial wear predictions using the extracted features. Residuals (discrepancies between predicted and actual wear values) are systematically quantified.
(3) Stage 2—Nonlinear Error Compensation: Autoencoder (AE) processes residuals through an encoder–decoder architecture to model complex error patterns.
(4) ACO Optimization: This dynamically adjusts the AE hyperparameters (training epochs, batch size, hidden layer dimensions) via a pheromone-mediated search.
(5) Final Output: Compensated residuals from the optimized AE are integrated with the initial SVR predictions, enhancing accuracy through linear–nonlinear synergy.

2.1. Support Vector Regression (SVR) Based on the Linear Kernel Function

Support Vector Regression (SVR) is a powerful regression algorithm based on the principle of support vector machines, aiming to find the optimal regression function on the training set [23]. Different from traditional regression, it introduces an insensitive loss function (ε-insensitive loss) to find a hyperplane that separates the data with the maximum margin within the allowable error range. As long as the error between the predicted value and the true value is within ε, no loss is counted.

When predicting, the new sample data are input into the trained model, and the predicted value is obtained through the hyperplane equation. Support Vector Regression (SVR) has significant advantages. It has good robustness to noise and outliers. Reasonably adjusting the penalty parameter C and the insensitive coefficient ε can effectively

avoid overfitting. Support Vector Regression (SVR) based on the linear kernel function is especially unique: It has high computational efficiency, so in large-scale data processing, compared with complex kernel functions, it is computationally simple and has a short training time. It also has strong interpretability, so it works in the original feature space and can intuitively present the linear relationship between features and prediction results, which is convenient for understanding the decision-making mechanism [18]. In addition, it has good generalization ability, so for linearly separable data, it can effectively avoid overfitting and ensure the stability of prediction for unknown data. It also has simple parameter adjustment, involving fewer parameters, making it easier to adjust the parameters and quickly find the optimal parameter combination.

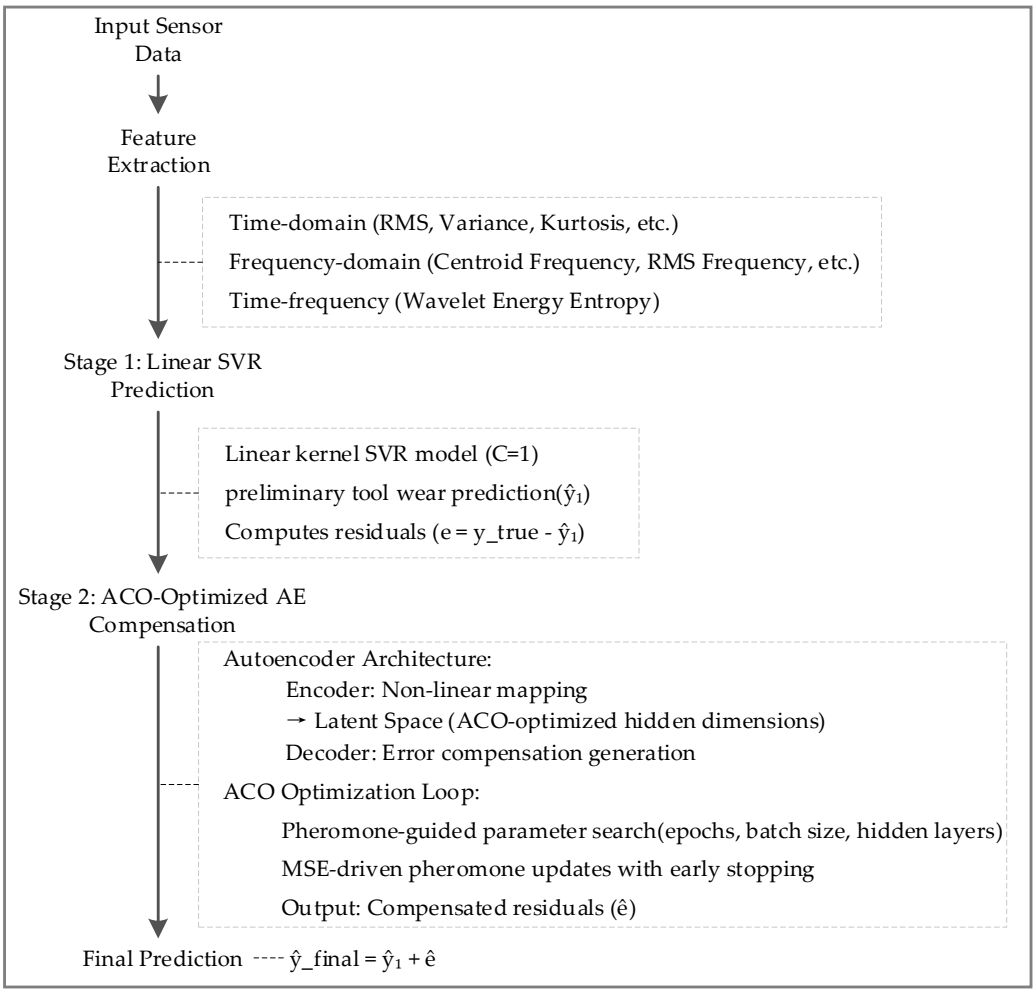


Figure 1. Data flow diagram of the ACO–SVR–AE framework.

The process of the Support Vector Regression algorithm is as follows:

(1) Collect and pre-process data: Collect relevant data and perform pre-processing operations such as data cleaning and normalization to improve the stability and convergence speed of the algorithm.

(2) Select the kernel function: According to the characteristics of the data, select an appropriate kernel function (such as linear kernel, polynomial kernel, Gaussian kernel, etc.) to map the low-dimensional data into a high-dimensional space to solve non-linear problems.

(3) Set hyperparameters C, ε, etc.: The hyperparameter C controls the degree of penalty for errors, and ε defines the width of the insensitive loss function. These parameters have an important impact on the performance of the model.

(4) Construct and solve the optimization problem: Construct the optimization objective function of Support Vector Regression, usually by balancing between the regularization and the loss function, and then use an optimization algorithm (such as the Sequential Minimal Optimization algorithm, SMO) to solve this optimization problem.
(5) Obtain the optimal solutions w and b: By solving the optimization problem, obtain the parameters of the model, that is, the weight vector w and the bias b.
(6) Predict new samples: Input the new sample data into the trained model.
(7) Output prediction results: Calculate the predicted values according to the model and output them.

2.2. Autoencoder for Non-Linear Error Compensation

Basic Principles of Autoencoder

Autoencoder is an unsupervised learning algorithm widely used in tasks such as data dimensionality reduction, feature extraction, and reconstruction [24]. Unsupervised feature learning can automatically extract the intrinsic features of data from unlabeled data. In situations where labeled data are scarce and difficult to obtain, this method has significant advantages. Its principle is based on neural networks and consists of two parts: an encoder and a decoder [25]. The encoder maps the input data to a low-dimensional representation, whereby the decoder reconstructs this low-dimensional representation into the original input. The autoencoder is composed of an encoding network and a decoding network. The encoder obtains the representation of the input vector. The decoder is used to minimize the residual vector between the input vector and the output vector (which represents the output of the decoder), thereby reconstructing the original input vector. Moreover, the autoencoder can encode the representation of the input layer to the hidden layer and then decode it to the output layer. The parameters of the two networks are learned through a reconstruction loss function. Since the reconstruction loss is small enough, the representation can retain most of the information of the input vector. The dimension of the input vector is m, and the number of input samples is n. Figure 2 shows the basic structure of the autoencoder.

The expression of the encoder is:

$$y = f_{encoder}(W_e x + b_e) \tag{1}$$

where $f_{encoder}$ is an activation function, such as sigmoid, tanh, identity, etc.; W_e is a weight matrix of size $m' \times m$; and b_e is a bias vector with dimension m'.

The expression of the decoder is:

$$\hat{x} = f_{decoder}(W_d y + b_d) \tag{2}$$

where $f_{decoder}$ is also an activation function, W_d is a weight matrix of size $m \times m'$, and b_d is a bias vector with dimension m.

The encoder f (Equation (1)) is implemented with tanh activations in all hidden layers, and the decoder f (Equation (2)) combines tanh in hidden layers with a linear activation in the final layer, following standard regression network conventions.

Loss function: The auto-encoder obtains appropriate parameter values by minimizing the loss function. The loss function is expressed as:

$$\theta = (W_e, W_d, b_e, b_d) \tag{3}$$

$$\begin{aligned} J(\theta) &= L(x, \hat{x}) + \lambda \|W\|^2 \\ &= \sum_{i=1}^{n} \|x_i - \hat{x}_i\|^2 + \lambda(\|W_e\|^2 + \|W_d\|^2) \end{aligned} \tag{4}$$

where $\lambda\|W\|^2$ is a regularization term, which is minimized to avoid overfitting [24].

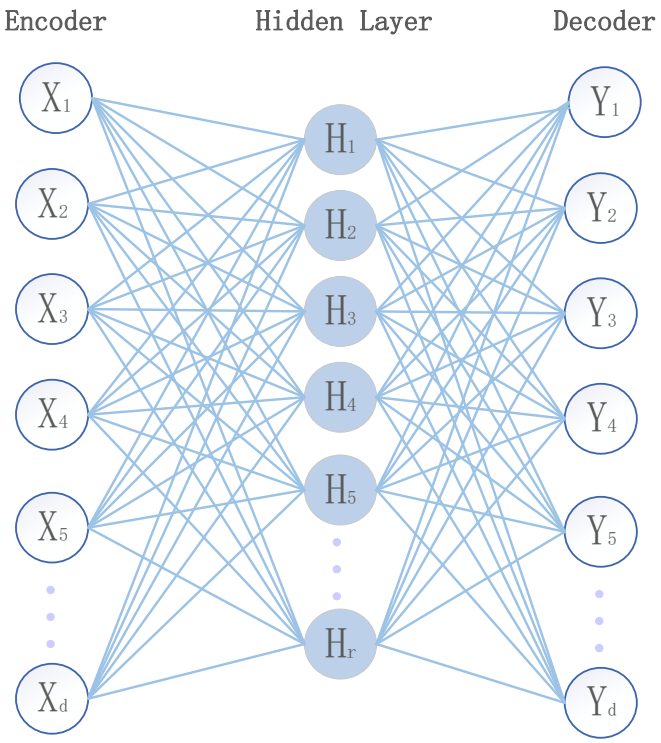


Figure 2. The fundamental structure of the autoencoder.

2.3. Ant Colony Optimization (ACO)

Ant Colony Optimization (ACO) is a heuristic optimization algorithm that simulates the foraging behavior of ants [26]. It has a unique principle for parameter optimization. First, the algorithm initializes a group of ants. Each ant represents a solution of parameter combinations, such as the learning rate of a machine learning model. Ants 'search' in the parameter space and choose the next parameter value based on the pheromone concentration and heuristic information. The higher the pheromone concentration, the greater the probability of the parameter combination being selected. After one search, the pheromone is updated according to the objective function value, increasing the pheromone of the path with the optimal objective function value. After multiple iterations, positive feedback is formed, and the ant colony converges to a relatively optimal parameter combination. After verification, it can be used as optimized parameters to solve practical problems. Figure 3 shows the flowchart for parameter optimization of the Ant Colony Optimization (ACO) algorithm. The process of Ant Colony Optimization (ACO) optimizing its key parameters (the number of training epochs, batch size, and the dimensions of the hidden layer) is as follows:

(1) Initialize parameters: Not only do the parameters of the ant colony algorithm itself need to be initialized, such as the number of ant colonies, initial pheromone values, heuristic factors, etc., but also the the model parameters that need to be optimized need to be set, namely the number of training epochs, batch size, and the dimensions of the hidden layer.

(2) Construct the initial population: Randomly generate a batch of individuals containing different combinations of the number of training epochs, batch size, and the dimension of the hidden layer as the initial population to start the optimization.

(3) Judge the training-rounds condition: Check whether the current number of training rounds is less than the expected number of training rounds. If so, continue the optimization process; otherwise, output the optimal parameter combination.

(4) Model training and performance calculation: For each individual, use its corresponding batch size and dimension of the hidden layer to train the model, and calculate the performance indicators of the model, such as accuracy, loss value, etc., to evaluate the quality of the parameter combination.

(5) Update the pheromone: According to the model performance indicators, increase the pheromone on the path of the parameter combination corresponding to the individual with better performance so that subsequent ants are more inclined to choose these relatively optimal parameter combination directions.

(6) Select a new parameter combination: Ants choose a new combination of the number of training epochs, batch size, and the dimensions of the hidden layer according to the pheromone concentration on the path and heuristic information with a certain probability. This process generates new individuals.

(7) Local search optimization: Conduct local search optimization on the newly generated individuals. By fine-tuning the parameter combination, it is possible to find a better solution.

(8) Update the number of training rounds: Increase the number of training rounds by one to carry out the next round of optimization iterations.

(9) Select the optimal parameter combination: When the number of training rounds reaches the expected number of training rounds, select the parameter combination corresponding to the individual with the best performance from all individuals; that is, obtain the optimal the number of training epochs, batch size, and the dimensions of the hidden layer.

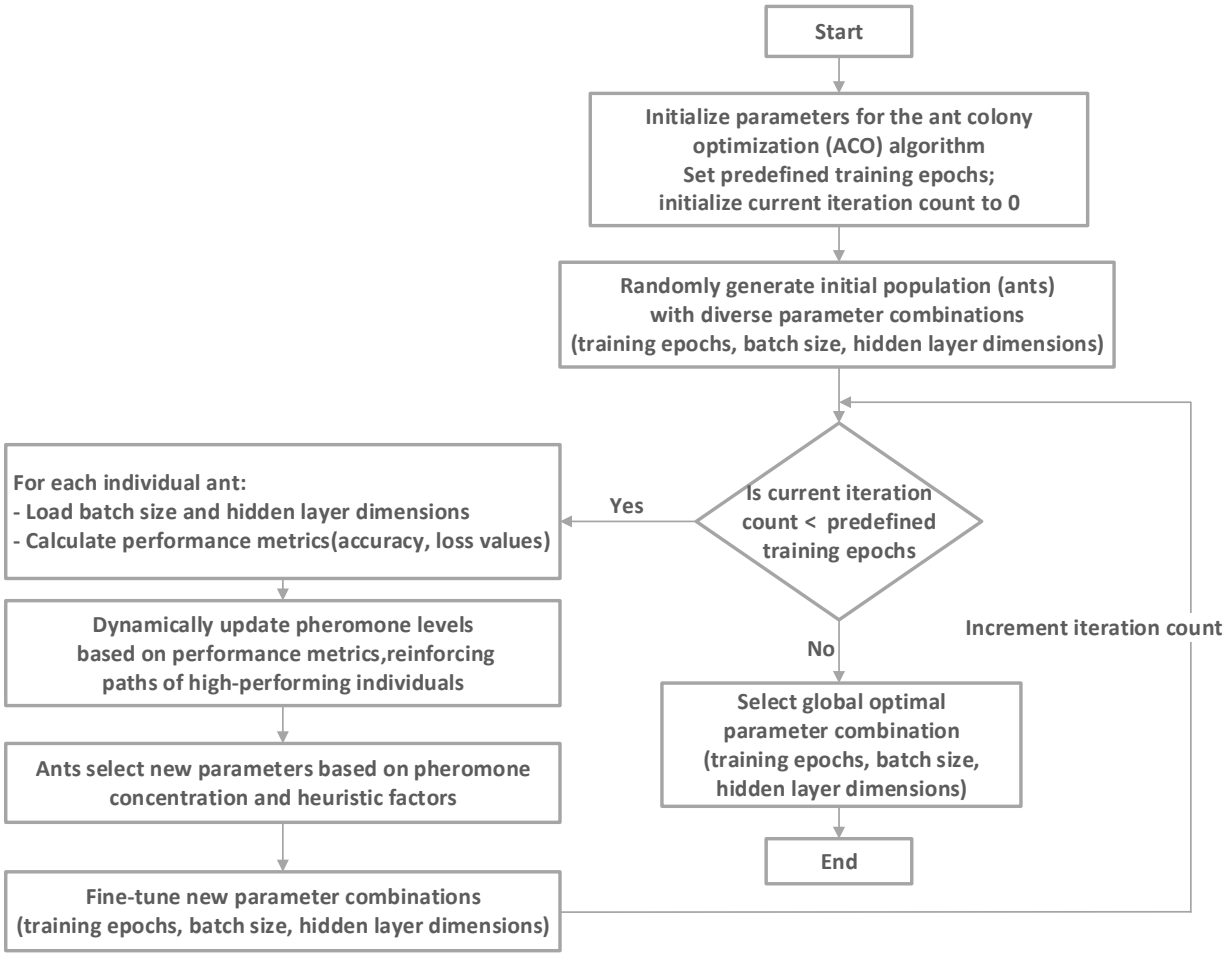


Figure 3. Flowchart of ACO parameter optimization.

3. Tool Wear Prediction Model with Linear Regression Optimized by Ant Colony Algorithm and Non-Linear Error Compensation

3.1. Prediction Model Framework

This paper proposes a two-stage hybrid prediction framework, named ACO–SVR–Autoencoder, integrating a linear kernel Support Vector Regression (SVR) with Autoencoder-based error compensation. In the first stage, an SVR model with linear kernel is applied to generate preliminary predictions, followed by a systematic calculation of residual errors. Building upon the initial SVR predictions, we first construct the baseline SVR–AE model by integrating a shallow Autoencoder (AE) for error compensation. With its demonstrated learning ability, the autoencoder can uncover complex error mapping relationships, thereby achieving effective compensation for the prediction error. Due to the limited sample size of the experimental dataset, deeper neural networks are prone to overfitting. Shallow autoencoders reduce model complexity by decreasing the number of network layers, enabling more stable feature learning with limited data. This study employs a four-layer shallow Autoencoder model for tool wear prediction. The Autoencoder model automatically learns compressed representations by reconstructing input data, demonstrating the capability to extract low-dimensional sensitive features related to tool wear from raw vibration/acoustic emission signals. To enhance the baseline SVR–AE model, the Ant Colony Optimization (ACO) algorithm is subsequently applied to systematically optimize three critical AE parameters: training epochs, batch size, and hidden layer dimensions. This optimization process upgrades the baseline architecture into the ACO–SVR–AE model, where ACO-driven parameter tuning synergizes with AE's nonlinear learning capability.

Ant Colony Optimization (ACO) mimics ants' pheromone foraging behavior. Each "ant" in our framework encodes an Autoencoder parameter set. Through pheromone-driven reinforcement of high-performance solutions and probabilistic parameter space exploration, ACO converges to optimal configurations. This collective intelligence mechanism avoids local minima and complements gradient-based Autoencoder training. A large number of experimental results show that this two-stage method combining Support Vector Regression (SVR) and Autoencoder demonstrates excellent prediction performance on multiple different datasets. Compared with using SVR or Autoencoder alone, its advantages are very significant. In addition, the prediction accuracy of the model optimized by the Ant Colony Optimization algorithm is further improved. This method exhibits moderate robustness in handling noise interference and complex data distributions, which is achieved through the complementary integration of SVR's linear modeling stability and the autoencoder's nonlinear feature adaptability. This gives it broad application potential in many fields and is expected to provide an efficient and reliable prediction solution for practical problems. The model framework diagram of this method is presented in Figure 4.

3.2. Optimized Non-Linear Error Compensation Model

Step 1: Feature Extraction

1. Time Domain Features: Root Mean Square (RMS), variance, maximum value, minimum value, skewness, kurtosis, and peak-to-peak value.
2. Frequency Domain Features: Centroid frequency, average frequency, Root Mean Squared frequency, and standard deviation of frequency.
3. Time–Frequency Domain Features: Wavelet energy entropy. The relevant formulas are presented in Table 1.

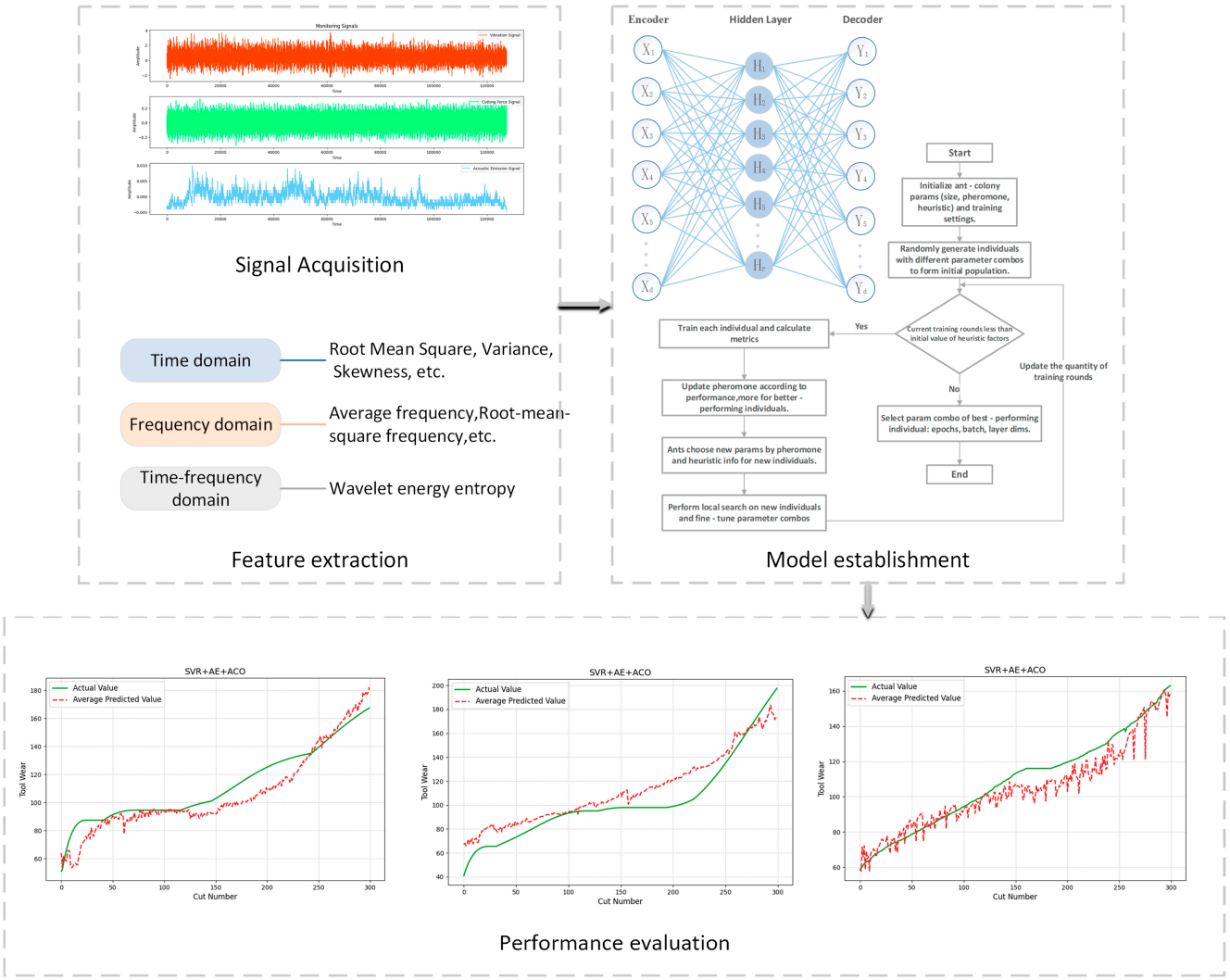


Figure 4. Diagram of the model framework.

Table 1. Experimental conditions.

Hardware Condition	Parameters
CNC milling machine	Roders TechRFM760 (Röders GmbH, Soltau, Germany)
Workpiece material	Nickel-based superalloy 718
Tool	3-tooth ball nose milling cutter
Data acquisition card	NI DAQ Data acquisition card (National Instruments, Austin, TX, USA)
Wear gauge	LEICA MZ12 microscope (Leica Microsystems, Wetzlar, Germany)

Step 2: Parameter Initialization

The regularization parameter CC is set to 1.

The number of training epochs is set to 30, the batch size is 24, and the heuristic information factor ranges between [0, 1].

The error between the predicted value in the first stage and the true value is utilized as the output data for the Autoencoder model. The Autoencoder model is trained to learn the nonlinear distribution patterns inherent in the error data. The encoder extracts latent features from the error data through nonlinear mapping, converting them into a low-dimensional representation in the latent space. These latent space features encapsulate the essential information of the error data while eliminating redundant noise. The decoder generates the error compensation value based on the low-dimensional features. By mini-

mizing the reconstruction error, the Autoencoder model accurately captures the nonlinear patterns of the error.

Step 3: ACO Optimizes the Parameters of the Autoencoder

For the hyperparameters of the Autoencoder model, such as the number of training epochs, batch size, and the dimensions of the hidden layer, the Ant Colony Optimization (ACO) algorithm is employed to conduct a global optimization. This approach mitigates the limitations associated with manual parameter adjustment. Ants select the parameter combinations of the Autoencoder model based on pheromone trails and heuristic information. The probability that ant k selects the parameter combination at the t-th iteration can be expressed as follows:

$$p_{kj}^{t} = \frac{\left[\tau_j^t\right]^{\alpha}\left[\eta_j\right]^{\beta}}{\sum_{l=1}^{m}\left[\tau_l^t\right]^{\alpha}\left[\eta_l\right]^{\beta}} \tag{5}$$

where τ_l^t is the pheromone concentration on parameter combination j at the t-th iteration, η_j is the heuristic information of parameter combination j, α and β are parameters controlling the relative importance of pheromone and heuristic information, and m is the total number of parameter combinations.

The pheromone concentration is dynamically adjusted based on the predicted Mean Squared Error (MSE) to enhance the search path for optimal parameter combinations. Additionally, an early stopping strategy is incorporated to improve computational efficiency. If no improvement is observed over five consecutive iterations, the search process is terminated prematurely to achieve a balance between efficiency and accuracy.

4. Experimental Results and Analysis

4.1. Introduction of the Dataset

The data are from the Prognostics and Health Management (PHM) Society's Prognostics and Health Management Competition for High-speed CNC Machine Tool Cutting Tools [27]. The data size: The dataset contains 315 files, each corresponding to a full milling cycle from tool initiation to failure. Each run recorded three synchronized sensor signals (force, vibration, and acoustic emission).

The data type: The dataset comprise time-series sensor signals (numerical data including cutting forces in Newtons, vibration in g, and acoustic emission in volts), tool wear labels (numerical values in micrometers quantified via post-process microscopy), and metadata capturing operational parameters such as spindle speed, feed rate, and depth of cut.

The data source: Collected by the PHM Society for the 2010 Data Challenge.

The experimental conditions are presented in Table 1. The PHM2010 dataset was used to verify the effectiveness of the method proposed in this paper. The equipment for collecting experimental data is shown in Figure 5. The spindle speed of the CNC milling machine was 10,400 r/min, the cutting depth in the Y-direction (radial) was 0.125 mm, the feed rate was 1555 mm/min, and the cutting depth along the Z-axis was 0.2 mm. The PHM2010 dataset contains 945 data points from three tool life experiments. Measurements included triaxial cutting forces (F_X, F_Y, F_Z), vibration, acoustic emission (AE) signals, and spindle motor current (21 statistical features per pass), with flank wear width (VB, μm) as the target variable. Subsequently, after the three signals collected during the cutting process were amplified by a signal amplifier, the original time domain signals were collected. The signal sampling frequency was 50 kHz, and a total of 7 signals were collected.

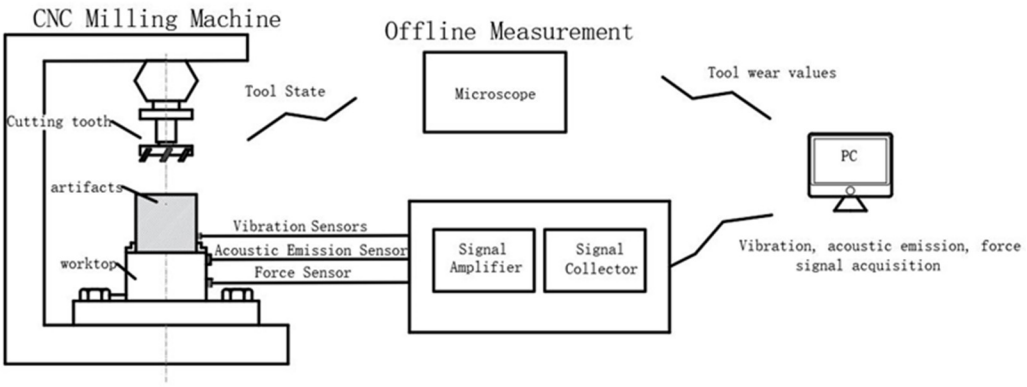


Figure 5. Schematic diagram of the experimental setup.

Experimental data: Acoustic emission signals, milling force signals, and vibration signals in three directions, namely the tool feed direction X, the spindle radial direction Y, and the spindle axial direction Z, were collected. Under fixed working conditions, full life-cycle data of six tools (C1, C2, C3, C4, C5, and C6) were obtained, with each tool cutting 315 times. Off-line wear measurements were carried out for each of the three milling cutters, C1, C4, and C6, and the average wear amount of the three cutting edges of each milling cutter was taken as the tool wear result. In this paper, three groups of experiments were set up, as shown in Table 2. Specifically, any two of these three tools were used for training, and the other one was used for testing.

Table 2. Partitioning of datasets.

Experiments	Training Set (80%)	Validation Set (20%)	Test Set
E1	C1 + C4 (80% data)	C1 + C4 (20% data)	C6
E2	C4 + C6 (80% data)	C4 + C6 (20% data)	C1
E3	C1 + C6 (80% data)	C1 + C6 (20% data)	C4

Open data provide us with abundant experimental conditions, tool wear data, multi-sensor signal data, dataset structure, and other information. In Table 3, "flute" represents the groove wear data.

Table 3. Partial flute wear value (unit: μm).

Cut	Flute_1	Flute_2	Flute_3
1	32.32	48.90	37.72
2	37.91	49.57	37.72
3	43.09	50.30	37.72
4	47.86	51.08	37.85
5	52.25	51.91	38.17
6	56.28	52.77	38.62
7	59.98	53.67	39.17
8	63.35	54.60	39.83
9	66.43	55.56	40.59
10	69.25	56.53	41.43
11	71.74	57.53	42.34
12	74.02	58.55	43.32
13	76.06	59.57	44.37
14	77.89	60.61	45.47
15	79.51	61.65	46.62

4.2. Data Pre-Processing and Feature Extraction

During milling operations, the data collected by sensors often fluctuate significantly. Such signals are interfered with by factors such as the environment and temperature, and cannot be effectively used to monitor the wear state of the milling cutter, thus making them invalid signals. Moreover, the signals collected by the sensors also contain redundant information. Therefore, to build a reliable tool wear monitoring model, it is necessary to eliminate the useless information in the original signals. In this experiment, each tool ran 315 times. The amount of signal data collected in each run varied greatly, with a maximum of 230,000 data points and a minimum of 120,000 data points. To obtain stable signals, the number of data points selected each time was controlled to between 40,000 and 90,000. The original data presented as a natural time-series signal, which depicts the relationship between physical signals or mathematical functions changing over time. Based on this characteristic, we could extract some statistical time domain features from the original data and use them as discriminative features to input into the deep learning system. In practical applications, statistical features such as the mean value, Root Mean Square (RMS), standard deviation, and variance are widely used. Given the stationarity of this signal, skewness and kurtosis were also extracted and incorporated into the model as input features [28]. Next, the frequency domain is a coordinate system used to describe the frequency characteristics of signals. In mechanical failures, periodic pulses are common, and the main frequency components contain information and discriminative features. By means of the Fast Fourier Transform (FFT), the time domain vibration signal can be converted into a frequency domain vibration signal, clearly showing the frequency characteristics of the signal [29]. For non-stationary signals, time–frequency domain features are of great practical value.

In the signal processing process, to convert a one-dimensional signal into a two-dimensional signal that combines time and frequency, the Short-Time Fourier Transform (STFT), wavelet transform/decomposition, and Empirical Mode Decomposition (EMD) are all commonly used and effective methods. The time frequency domain, simply put, integrates the concepts of the time domain and the frequency domain. Its principle is to sequentially extract the corresponding frequency domain information within different time windows in the time domain as time progresses so as to analyze the signal features more comprehensively and in detail [30]. Wavelet analysis, STFT, and Hilbert–Huang Transform all belong to time frequency domain methods, which can analyze the waveform signals in both the time domain and the frequency domain simultaneously. In addition, some research attempts to fuse the features of the time domain, frequency domain, and time–frequency domain together to present the feature information more comprehensively [31]. In this paper, 12 features were extracted from each signal, including those in the time, frequency, and time–frequency domains. These features reflect the characteristics of the sensor signals from different perspectives. In the time domain, 7 feature quantities, namely the Root Mean Square, variance, maximum value, minimum value, skewness, kurtosis, and peak-to-peak value, were extracted. In the frequency domain, 4 feature quantities, namely the centroid frequency, average frequency, Root mean Square frequency, and frequency standard deviation, were extracted. In the time–frequency domain, wavelet energy entropy was extracted. A total of 84 feature quantities were extracted to form a feature set. The specific characteristic information is shown in Table 4.

Table 4. List of extracted features.

Domain	Features	Expression	
Statistical	RMS	$Z_{rms} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} {z_i}^2}$	(6)
	Variance	$Z_{\text{var}} = \frac{1}{n}\sum_{i=1}^{n} (z_i - \bar{z})^2$	(7)
	Maximum	$Z_{\max} = \max(z)$	(8)
	Minimum	$Z_{\min} = \min(z)$	(9)
	Skewness	$Z_{\text{skew}} = \frac{1}{N}\sum_{i=1}^{N} \frac{(s_i - \bar{s})^3}{{\rho_t}^3}$	(10)
	Kurtosis	$Z_{kurt} = \frac{1}{N}\sum_{i=1}^{N} \frac{(s_i - \bar{s})^4}{{\rho_t}^4}$	(11)
	Peak-to-peak	$Z_{p-p} = \max(z) - \min(z)$	(12)
Frequency	Gravity frequency	$F_{FC} = \frac{\sum_{n=1}^{N} \overline{u}(n)u(n)}{2\pi\sum_{n=1}^{N} u(n)^2}$	(13)
	Average frequency	$F_{MF} = \frac{1}{N}\sum_{n=1}^{N} U(n)$	(14)
	RMSF	$F_{RMSF} = \sqrt{\frac{\sum_{n=1}^{N} u(n)^2}{4\pi^2 u(n)^2}}$	(15)
	RVF	$F_{RVF} = \sqrt{\frac{\sum_{n=1}^{N} (-F_{FC})^2 u(n)}{\sum_{n=1}^{N} u(n)}}$	(16)
Time–frequency	Wavelet energy	$E_{WT} = \sum_{i=1}^{N} \omega t_{\phi}^2(i)/N$	(17)

4.3. Effectiveness of ACO–SVR–Autoencoder

MAE and MSE are two main methods for evaluating the prediction accuracy of a model. In this study, the Mean Absolute Error (MAE) and the Mean Squared Error (MSE) were adopted. Specifically, the MAE and MSE of all time periods in the test sequence were calculated to achieve an overall assessment of the prediction quality of the entire test-set sequence, thus enhancing comparability.

The Mean Absolute Error is the average of the absolute values of the differences between the predicted values and the true values. Its calculation formula is as follows:

$$MAE = \frac{1}{\mathrm{n}}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{18}$$

In this formula, n represents the number of samples, y_i is the true value of the i-th sample, and $\hat{y}_i$ is the predicted value of the i-th sample.

The advantage of MAE is that it is simple to calculate and has an intuitive meaning. It measures the error in the unit of the original data and can reflect the average magnitude of the prediction error. The Mean Squared Error is the average of the squares of the differences between the predicted values and the true values. Its calculation formula is:

$$MSE = \frac{1}{\mathrm{n}}\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{19}$$

Compared with the MAE, the MSE amplifies the impact of larger errors by squaring the errors. This makes the model pay more attention to reducing larger errors during the training process, as larger errors contribute more to the MSE.

In this study, the MAE and MSE were calculated to evaluate the performance of the Ant Colony Optimization–Support Vector Regression–Autoencoder tool wear prediction method (ACO—VR–Autoencoder). In the first stage, after using the linear Support Vector Regression (SVR) with a linear kernel to predict the data, the MAE and MSE between the predicted values and the true tool wear values were calculated to measure the preliminary prediction error of the Support Vector Regression (SVR) model. In the second stage, the Autoencoder was used to perform non-linear modeling on this error and achieve error

compensation. Then, the MAE and MSE between the compensated predicted values and the true values were calculated again. By comparing the changes in the indicators before and after, the effect of error compensation and the effectiveness of the entire two-stage prediction method can be evaluated. Through in-depth analysis of these error indicators, a more comprehensive understanding of the model's performance at different stages can be obtained, providing strong data support for further model optimization.

The parameters of the Support Vector Regression (SVR) and the Autoencoder were determined through experiments, and a prediction model based on the combination of linear Support Vector Regression and Autoencoder was constructed and tested on the dataset of a high-speed CNC milling machine. This method aimed to reveal the long-term trend of the data while minimizing the impact of short-term oscillations. The effectiveness of this method was verified by comparison with other algorithms.

Figure 6 presents the visualization of experimental results from three datasets used for validating the ACO–SVR–Autoencoder method. C1 served as the test set, while C4 and C6 served as the training sets. C4 served as the test set, while C4 and C6 served as the training sets. C6 served as the test set, while C1 and C4 served as the training sets.

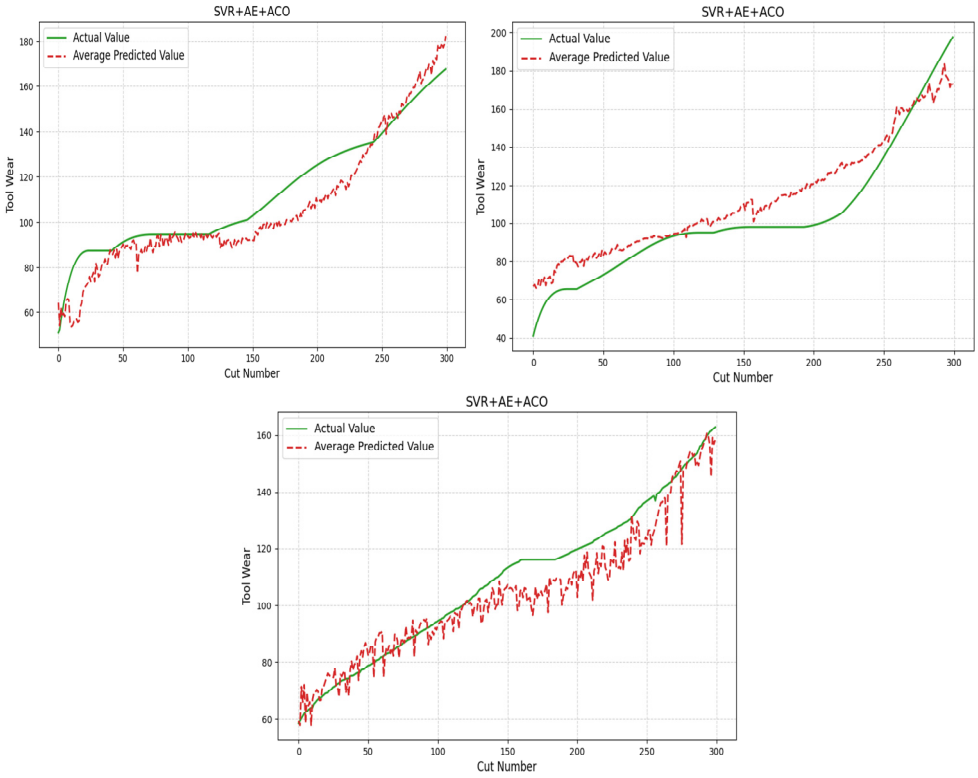


Figure 6. Visualization of prediction results.

4.3.1. Parameter Determination Process

The configuration of model hyperparameters was determined through a systematic experimental validation approach. Specifically, we implemented a stratified cross-validation strategy by reserving 20% of the original training set as an independent validation set to assess the generalization capabilities of different parameter combinations. Multiple rounds of iterative experiments were conducted to monitor variations in both loss functions and evaluation metrics on the validation set. The regularization coefficient (C = 1) was selected to optimally balance model complexity with fitting performance. The determination of hidden layer neuron numbers (100/140) achieved an effective compromise between representational capacity and overfitting prevention. Training for 30 epochs with a batch size of 24 was optimized to strike a balance between convergence rate and computational resource

utilization. The heuristic factor range [0, 1] was validated through controlled experiments, demonstrating its effectiveness. All hyperparameter settings were ultimately finalized based on the consistent performance stability observed across the validation cycles.

(1) Determination of Parameters for Linear Support Vector Regression (SVR)

The linear kernel was selected as the kernel function for linear Support Vector Regression (SVR). The regularization parameter was explicitly set to 1.

During the experiment, relatively good results were achieved by adjusting the regularization parameter C through cross-validation; given the need for subsequent error modeling, the remaining hyperparameters were configured using the methodology proposed in reference [32].

(2) Determination of Parameters for Autoencoder

The parameters of the Autoencoder model were determined using the Ant Colony Optimization (ACO) algorithm. The following parameter ranges were defined:

The number of training epochs for the Autoencoder ranged from 10 to 100.

The batch size of the Autoencoder ranged from 16 to 128.

The number of pre-training epochs for the Autoencoder ranged from 10 to 50.

The number of neurons in the hidden layers of the Autoencoder ranged from 50 to 120 and from 120 to 300 for different architectures.

These parameters were randomly generated in each iteration of the ACO algorithm and subsequently updated and optimized based on the model's performance, as evaluated by the Mean Squared Error (MSE).

(3) Determination of Parameters for the Ant Colony Optimization (ACO) Algorithm

The parameters of the Ant Colony Optimization (ACO) algorithm were manually configured as follows:

The number of ants was set to 10.

The number of iterations was set to 10.

The pheromone importance factor was set to 1.

The heuristic information importance factor was set to 2.

The pheromone evaporation rate was set to 0.5.

Given that nearly all prediction models necessitate random parameter initialization, in order to ensure the reliability and stability of the experimental results, each model was tested 30 times. The specific experimental parameter settings are elaborated on in Table 5. Moreover, in this experiment, two metrics, namely the Mean Absolute Error (MAE) and the Mean Squared Error (MSE), were chosen to comprehensively assess the prediction performance of the models.

Table 5. Setting experimental parameters.

Model	Parameters
Support Vector Regression (SVR)	The regularization parameter C is set to 1.
Autoencoder	The pre-training loss function is the mean squared error, and the dimensions of the hidden layers are set to 100 and 140 respectively. The number of training rounds is set to 30, and the batch size is set to 24.
Neural network	The dimensions of the other two hidden layers (if applicable, clarify the context) are set to 70 and 140.
ACO	The range of the heuristic information factor is set to within [0, 1].

The parameter configurations are grounded in established literature and validated through empirical studies:

Regularization (C = 1): We adopted dropout regularization (applied after the second hidden layer), as recommended by Srivastava et al. (2014) [33], to prevent overfitting in deep autoencoders. A dropout rate of 0.5 (equivalent to C = 1 in our framework) was selected via grid search on the validation data to balance model capacity and generalization.

Hidden Layers (100→140): The hierarchical architecture followed the layer-wise dimension increase proposed by Zhang et al. (2011) [34], where incremental expansion (input→100→140) enhances feature disentanglement. Ablation studies showed that smaller layer dimensions (e.g., 50→120) increased reconstruction MSE by 32% compared to baseline, while larger dimensions (200→300) caused validation loss deterioration (overfitting).

Training Protocol (30 epochs, batch size = 24):

Pre-training (25 epochs): This aligns with Hinton (2006) [35], where greedy layer-wise pre-training stabilizes weight initialization.

Batch Size 24: This follows the small-batch heuristic from Kingma (2014) [36], balancing gradient noise and convergence efficiency for Autoencoders (AEs).

Heuristic Factor [0, 1]: Input normalization to a unit range aligns with guidance from Yann et al. [37] (2015) on feature scaling for neural networks, ensuring domain knowledge inputs (e.g., expert scores) do not dominate latent representations.

4.3.2. Reasons for Algorithm Selection

(1) Reasons for Selecting Linear Support Vector Regression (SVR)

Support Vector Regression (SVR) excels in handling linear relationships and has moderate robustness. In tool wear prediction, data such as the cutting speed of the tool and the amount of wear often exhibit a linear trend. Support Vector Regression (SVR) can accurately capture these linear features, providing a reliable basic prediction for the entire prediction process. This not only reduces errors caused by improper handling of linear relationships but also lays a solid foundation for subsequent error compensation work. As a result, Support Vector Regression (SVR) serves as a crucial first step in the overall prediction framework.

(2) Reasons for Selecting Autoencoder

Autoencoder has excellent capabilities in non-linear modeling and unsupervised learning. Tool wear is comprehensively affected by various factors such as cutting force and temperature, and there are complex non-linear relationships between these factors and the amount of wear. Autoencoder can automatically learn the feature representation of input data, deeply explore the complex non-linear relationships in prediction errors, accurately model the errors, and effectively compensate for the deficiencies of Support Vector Regression (SVR)'s linear prediction. For example, Windrim et al. [38] demonstrated the superiority of Autoencoders in unsupervised feature learning for hyperspectral data, with their model capturing non-linear dependencies and improving prediction accuracy compared with traditional linear methods. Consequently, this significantly improves the overall prediction accuracy, complementing Support Vector Regression (SVR)-based linear prediction.

(3) Reasons for Selecting Ant Colony Optimization (ACO) Algorithm

The global search ability of ACO and its characteristics of simulating intelligent behavior make it an ideal choice for optimizing the parameters of Autoencoder. In a complex parameter space, ACO can conduct extensive searches, avoid getting trapped in local optimal solutions, and help find the parameter combination that optimizes the performance of Autoencoder, giving full play to Autoencoder's non-linear modeling capabilities. By

simulating the cooperation and information-exchange mechanism of ant colonies, ACO continuously updates pheromones during the iterative process to guide the search direction and can adaptively adjust the search strategy. It optimizes the key parameters of Autoencoder (the number of training epochs, batch size, and the dimensions of the hidden layer), avoiding the limitations and blindness of manual parameter adjustment, maximizing the performance of Autoencoder, and ensuring that its non-linear modeling and error compensation effects reach the best state. Thus, it significantly improves the accuracy of tool wear prediction as well as the performance and stability of the entire prediction method. ACO's optimization of Autoencoder further enhances the overall effectiveness of the combined algorithm. In summary, the three algorithms cooperate with each other, giving full play to their respective advantages, jointly forming the ACO–SVR–Autoencoder method, which realizes efficient data prediction and error compensation. Through comparative research, this paper aims to better illustrate the superiority of the ACO–SVR–Autoencoder method in improving prediction accuracy, thereby providing robust support for solving practical problems. To comprehensively verify the effectiveness of this optimized two-stage combined error-compensation prediction method, we carefully designed comparative experiments. The final prediction results of the ACO–SVR–Autoencoder method were rigorously compared with two categories of baseline approaches (as illustrated in Figures 6–8): (1) single-stage methods (predictions using solely linear Support Vector Regression (SVR) or Autoencoder alone) and (2) classic machine learning algorithms such as neural networks and random forests. Comparative analysis demonstrated that the proposed two-stage combination method achieved significantly higher prediction accuracy than both single-stage methods and traditional algorithms. This marked improvement underscores the practical feasibility of the two-stage framework in real-world applications.

4.4. Analysis of the Prediction Results of the Optimized Linear Regression and Non-Linear Error Compensation Model

The innovations achieved in the prediction method of this study are mainly reflected in the following three key aspects:

First, a two-stage prediction method, ACO–SVR–Autoencoder (Ant Colony Optimization–Support Vector Regression–Autoencoder), is proposed through the integration of linear regression and non-linear error compensation. Specifically, the SVR model, utilizing a linear kernel function, demonstrates superior performance in modeling linear relationships by efficiently capturing data trends, thereby establishing a robust prediction baseline. Conversely, the autoencoder architecture excels at extracting complex non-linear features through its deep representation learning capability. This hybrid approach leverages complementary strengths—linear modeling fidelity and non-linear residual compensation—to achieve significant improvements in prediction accuracy. Second, in the error compensation stage, Autoencoder demonstrates unique advantages. It can deeply explore the complex error mapping relationships and effectively tackle the challenges confronted by traditional methods in dealing with non-linear problems. This powerful non-linear processing ability enables the prediction model to compensate for errors more accurately, thus further optimizing the prediction results. Finally, the introduction of the Ant Colony Optimization (ACO) algorithm to optimize the key parameters of Autoencoder is another innovative highlight of this study. ACO breaks through the limitations of traditional parameter adjustment methods. By simulating the intelligent behavior of ant colonies, it can adaptively search for the optimal solution in the complex parameter space, maximizing the performance of Autoencoder. This not only further enhances the effectiveness of the prediction model but also significantly improves the model's stability, ensuring good prediction performance under different data conditions. In conclusion, the integrated

ACO–SVR–Autoencoder optimization approach delivers an efficient, accurate, and stable solution for the prediction field.

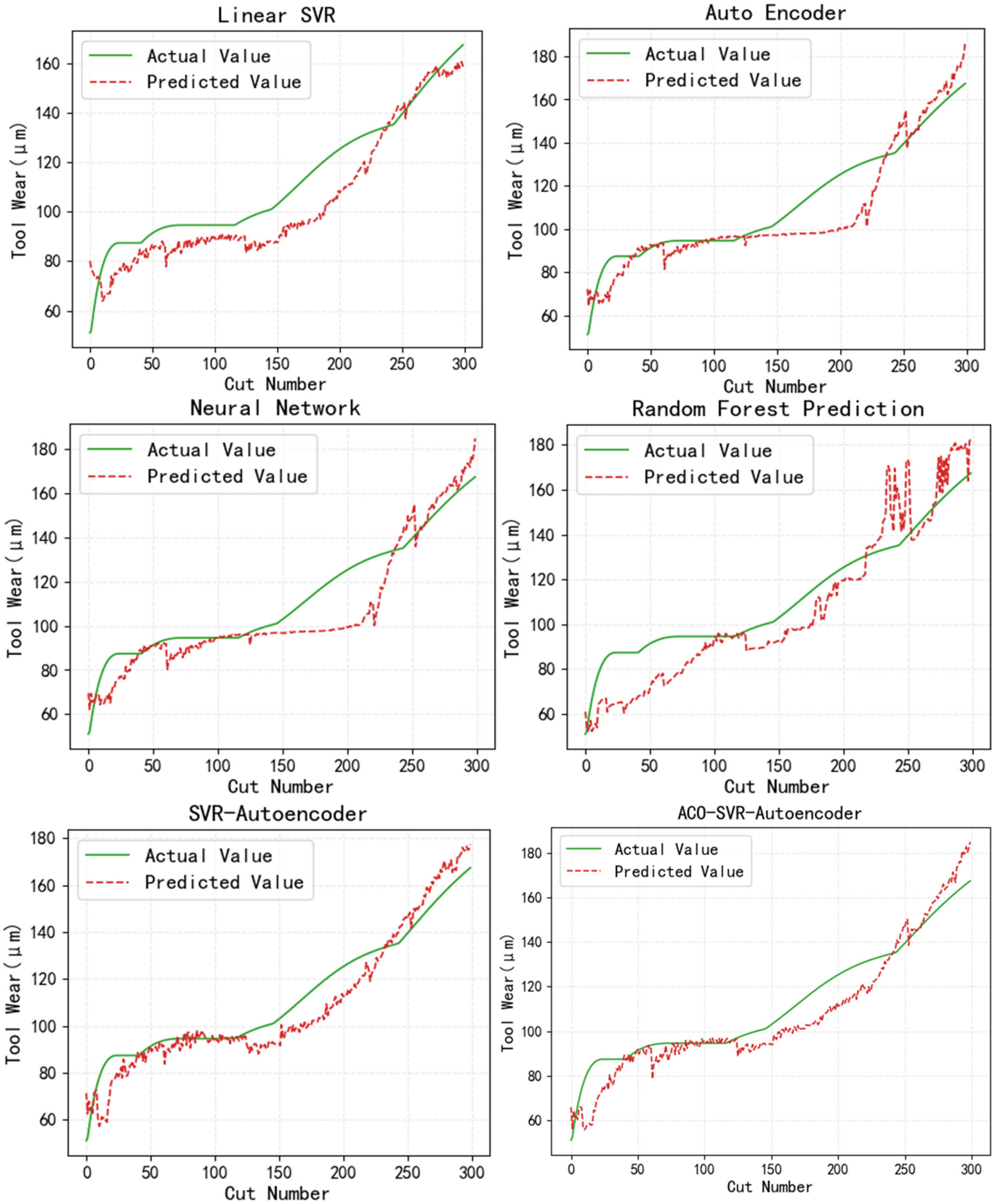


Figure 7. E1 compares the prediction results of various methods.

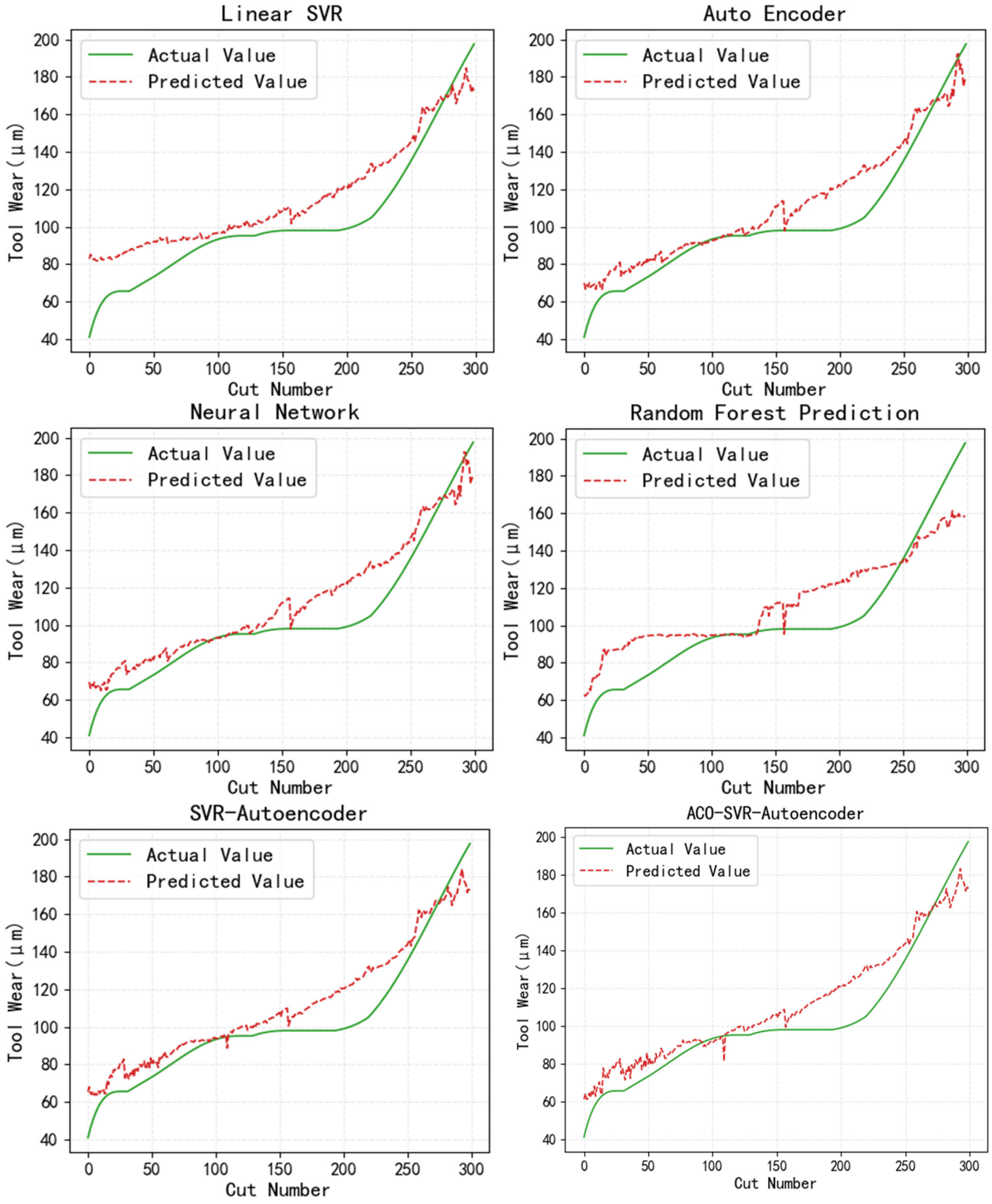


Figure 8. E2 compares the prediction results of various methods.

To comprehensively evaluate the effectiveness of this optimized two-stage combined error compensation prediction method, comparative experiments were designed. The proposed methodology was evaluated on the PHM2010 dataset to validate its performance characteristics under standardized benchmarking conditions. The final prediction

results obtained through the proposed method were compared with those from single approaches [11] (Support Vector Regression (SVR) and Autoencoder) and other classical algorithms (neural networks [39] and random forest [40]). The comparative analysis revealed that the prediction results obtained through the two-stage combined approach demonstrated enhanced performance compared to those generated by standalone methods or classical algorithms. These findings indicate the effectiveness of the two-stage integrated prediction framework.

E1: The test set used in this experiment was C1, with C4 and C6 serving as the training sets. As depicted in Figure 7, ACO–SVR–Autoencoder outperformed conventional methods (Linear Support Vector Regression (SVR), Autoencoder, random forest, neural networks) and our baseline SVR–Autoencoder in prediction accuracy. While the baseline SVR–Autoencoder achieved the lowest MSE (83.87, Table 6), demonstrating its effectiveness in reducing large-magnitude errors, ACO–SVR–Autoencoder attained the lowest MAE (7.50), signifying superior robustness in minimizing systematic deviations. The marginally higher MSE of the ACO–SVR–Autoencoder (88.75 vs. 83.87) can be attributed to the Ant Colony Optimization (ACO) algorithm's focus on MAE minimization during hyperparameter selection.

Table 6. The prediction results of various methods in E1.

Method	Linear SVR	Autoencoder	Neural Network	Random Forest	SVR–Autoencoder	ACO–SVR–Autoencoder
MSE	125.76	144.54	154.06	292.99	83.87	88.75
MAE	9.59	8.88	9.26	14.45	7.54	7.50

Although the absolute error reduction from ACO integration appears modest, its advantages extend beyond direct metric improvements. First, ACO automates the optimization of key Autoencoder hyperparameters (training epochs, batch size, and hidden layer dimensions), eliminating manual tuning while ensuring model stability. Second, the ACO-optimized framework exhibited enhanced generalization, as evidenced by its lower performance variance across repeated trials, indicating reduced sensitivity to initialization and noise.

E2: The test set was C4, and the training sets were C1 and C6. Figure 8 shows the visualization of the prediction results of each method. As can be seen from Table 7, when comparing the tabular data, traditional methods such as Linear Support Vector Regression (SVR), Autoencoder, random forest, and neural networks performed worse than the SVR–Autoencoder and ACO–SVR–Autoencoder proposed in this study in terms of MSE and MAE metrics. Among them, ACO–SVR–Autoencoder had the lowest MSE and MAE, demonstrating better prediction accuracy and stability.

Table 7. The prediction results of various methods in E2.

Method	Linear SVR	Autoencoder	Neural Network	Random Forest	SVR–Autoencoder	ACO–SVR–Autoencoder
MSE	248.87	165.47	173.82	289.62	147.78	146.91
MAE	13.74	10.26	10.60	14.40	10.10	10.04

E3: The test set was C6, and the training sets were C1 and C4. Figure 9 shows the visualization of the prediction results of each method. The proposed ACO–SVR–Autoencoder method demonstrated enhanced prediction accuracy compared to conventional approaches, including linear Support Vector Regression (SVR), Autoencoder, random forest, and neural

networks (Table 8). Although the baseline SVR–Autoencoder framework achieved comparable performance metrics (MSE: 96.50; MAE: 8.11), ACO–SVR–Autoencoder obtained minimal prediction errors, with an MSE of 96.412 and an MAE of 7.75, demonstrating superior capability in reducing error fluctuations. This performance improvement primarily originated from the ACO optimization algorithm's effective tuning of critical parameters in the Autoencoder architecture. The experimental results confirm the effectiveness of the proposed approach in enhancing prediction accuracy.

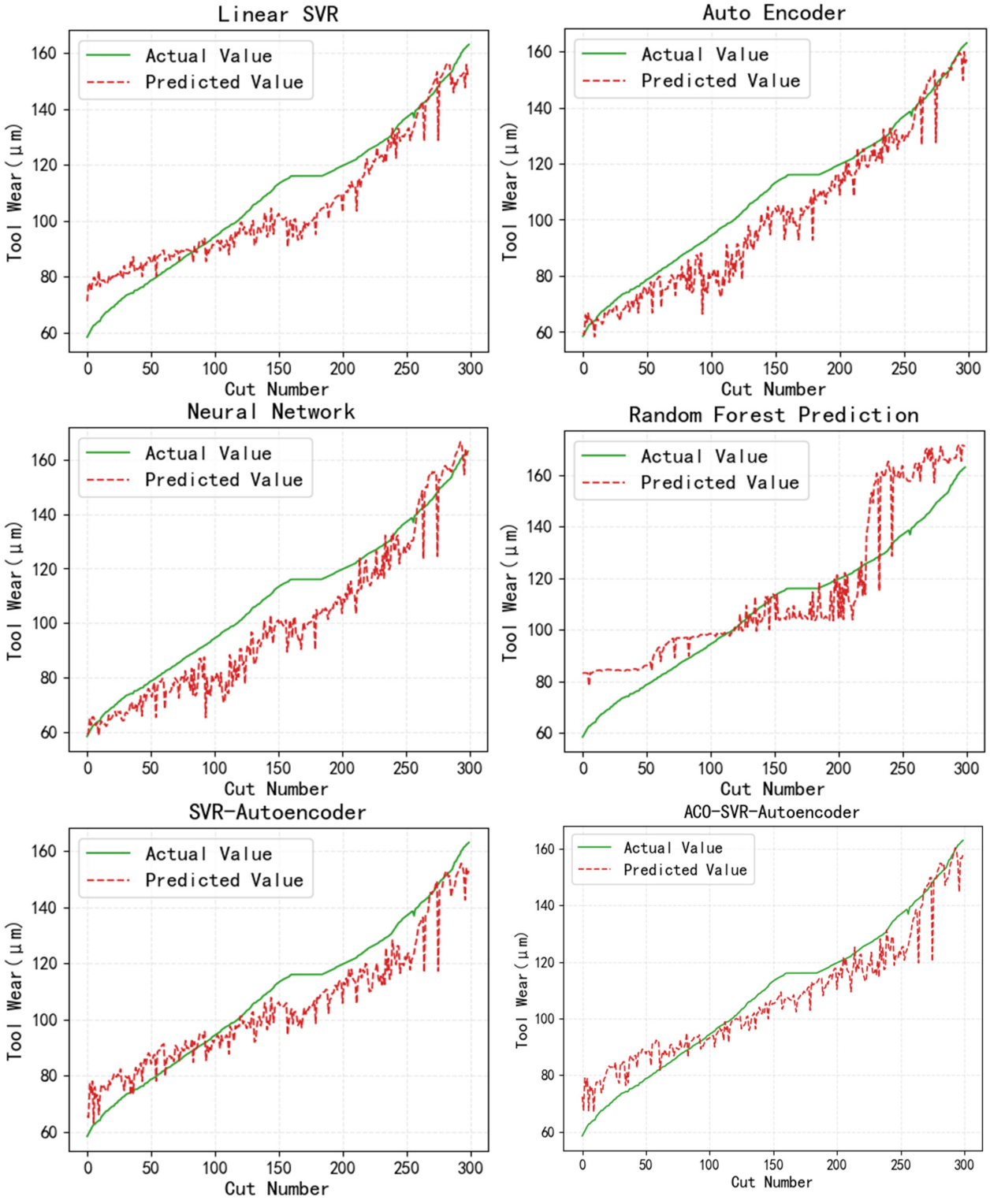


Figure 9. E3 compares the prediction results of various methods.

Table 8. The prediction results of various methods in E3.

Method	Linear SVR	Autoencoder	Neural Network	Random Forest	SVR–Autoencoder	ACO–SVR–Autoencoder
MSE	91.93	128.47	216.42	190.40	96.50	96.41
MAE	8.07	8.75	12.23	11.34	8.11	7.75

In this study, we employed two distinct models: Support Vector Regression (SVR) was first utilized to model the linear components of the data, followed by an Autoencoder model to capture non-linear patterns. This hybrid framework effectively integrated both linear and non-linear data characteristics. Additionally, an intelligent optimization algorithm was applied to adaptively tune the model parameters based on the dataset's intrinsic properties. Consequently, this methodology significantly improved the predictive accuracy of the integrated framework.

This study validated the effectiveness of the Ant Colony Optimization (ACO) algorithm through three sets of comparative experiments, using evaluation metrics including Mean Squared Error (MSE) and Mean Absolute Error (MAE). The results show that while the MSE displayed an increased error in one experimental group, the MAE exhibited reduced errors across all three groups. This indicates that the ACO algorithm is practically effective. The findings confirm that the ACO algorithm can adaptively optimize model parameters based on different datasets, providing insights for applications in other domains.

5. Conclusions

This study proposes a two-stage prediction method based on linear regression and non-linear error compensation. The effectiveness of the model is verified by comparing the results of three groups of experiments. The research conclusions are as follows. To address accuracy limitations in tool wear process modeling and remaining useful life (RUL) estimation, this study proposes a two-stage ACO–SVR–AE framework that integrates linear regression and nonlinear error compensation. The methodology operates in three phases:

(1) Linear Modeling: A baseline Support Vector Regression (SVR) model predicts tool wear trends, generating preliminary results and residual error distributions.
(2) Nonlinear Compensation: An Autoencoder (AE) is employed to learn and model the nonlinear error patterns, thereby compensating for the residuals in SVR predictions and forming the integrated SVR–AE hybrid model.
(3) Parameter Optimization: The Ant Colony Optimization (ACO) algorithm adaptively tunes three critical AE hyperparameters—the number of training epochs, batch size, and hidden layer dimensions, building the ACO–SVR–AE model.

Experimental results on the PHM2010 dataset demonstrate the feasibility of the proposed method. Compared to standalone Support Vector Regression (SVR) and Autoencoder (AE) models, the framework achieves significant average reductions of 26.1% in Mean Squared Error (MSE) and 14.5% in Mean Absolute Error (MAE). When benchmarked against traditional approaches such as random forest and neural networks, the improvements are even more pronounced, with a 32.3% lower MSE and a 25.3% lower MAE. By synergistically integrating linear modeling and nonlinear error compensation, this method offers an innovative solution for predictive tasks in complex industrial systems.

This optimized two-stage framework provides an effective solution for practical prediction challenges, particularly in scenarios requiring both linear pattern capture and non-linear residual correction.

The proposed algorithm's performance may degrade in scenarios with significant variations in working conditions (e.g., cutting speed, feed rate, or depth of cut). If training

and testing data derive from distinct parameter settings, the statistical distributions of vibration/acoustic emission signals may diverge substantially. As the model is trained on condition-specific signal patterns, such divergence would require retraining with new data to ensure accuracy in altered environments.

Future work will focus on developing systematic prediction and optimization models for the cutting process, explicitly accounting for variations in working conditions and cutting parameters.

Author Contributions: Conceptualization, B.D. and H.Y.; methodology, L.S.; software, L.S. and H.F.; validation, H.F.; formal analysis, B.D. and L.S.; investigation, L.S. and H.Y.; resources, B.D.; data curation, H.F.; writing—original draft preparation, L.S. and H.F.; writing—review and editing, L.S.; visualization, H.F.; supervision, B.D. and H.Y.; project administration, B.D.; funding acquisition, H.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the project: CAM software development for parts machining, grant number: E41T0B01.

Data Availability Statement: These data were derived from the following resources available in the public domain: PHM Society. PHM Society 2010 PHM Society conference data challenge (https://www.phmsociety.org/competition/PHM/10 (accessed on 1 January 2023) [27].

Acknowledgments: We would like to express our appreciation for the support provided by the research community, which has been instrumental in facilitating this work.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Sun, M.; Wang, X.; Guo, K.; Huang, X.; Sun, J.; Li, D.; Huang, T. Tool wear monitoring based on physics-informed Gaussian process regression. *J. Manuf. Syst.* **2024**, *77*, 40–61. [CrossRef]
2. Xie, Z.; Zhang, Z.; Chen, J.; Feng, Y.; Pan, X.; Zhou, Z.; He, S. Data-driven unsupervised anomaly detection of manufacturing processes with multi-scale prototype augmentation and multi-sensor data. *J. Manuf. Syst.* **2024**, *77*, 26–39. [CrossRef]
3. Munaro, R.; Attanasio, A.; Del Prete, A. Tool wear monitoring with artificial intelligence methods: A review. *J. Manuf. Mater. Process.* **2023**, *7*, 129. [CrossRef]
4. Zhang, X.; Gao, Y.; Guo, Z.; Zhang, W.; Yin, J.; Zhao, W. Physical model-based tool wear and breakage monitoring in milling process. *Mech. Syst. Signal Process.* **2023**, *184*, 109641. [CrossRef]
5. Yang, C.; Shi, Y.; Xin, H.; Zhao, T.; Zhang, N.; Xian, C. Tool wear prediction model based on wear influence factor. *Int. J. Adv. Manuf. Technol.* **2023**, *129*, 1829–1844. [CrossRef]
6. Zhang, Z.; Liu, Z.; Ren, X.; Zhao, J. Prediction of Tool Wear Rate and Tool Wear during Dry Orthogonal Cutting of Inconel 718. *Metals* **2023**, *13*, 1225. [CrossRef]
7. Zhang, Z.; Jia, L.; Luo, M.; Wu, B.; Zhang, D. A data-driven method for prediction of surface roughness with consideration of milling tool wear. *Int. J. Adv. Manuf. Technol.* **2024**, *134*, 4271–4282. [CrossRef]
8. Wang, Q.; Chen, X.; An, Q.; Chen, M.; Guo, H.; He, Y. A tool wear prediction and monitoring method based on machining power signals. *Int. J. Adv. Manuf. Technol.* **2023**, *129*, 5387–5401. [CrossRef]
9. Fan, C.; Zhang, Z.; Zhang, D.; Luo, M. Tool wear prediction based on a fusion model of data-driven and physical models in the milling process. *Int. J. Adv. Manuf. Technol.* **2024**, *133*, 3673–3698. [CrossRef]
10. Zhou, Y.; Zhi, G.; Chen, W.; Qian, Q.; He, D.; Sun, B.; Sun, W. A new tool wear condition monitoring method based on deep learning under small samples. *Measurement* **2022**, *189*, 110622. [CrossRef]
11. Zhao, R.; Yan, R.; Chen, Z.; Mao, K.; Wang, P.; Gao, R.X. Deep learning and its applications to machine health monitoring. *Mech. Syst. Signal Process.* **2019**, *115*, 213–237. [CrossRef]
12. Tian, J.; Wang, S. Tool wear prediction based on XLSTM combined with the ECA attention mechanism model. *J. Mech. Sci. Technol.* **2025**, *39*, 519–530. [CrossRef]
13. Zhang, C.; Yao, X.; Zhang, J.; Jin, H. Tool Condition Monitoring and Remaining Useful Life Prognostic Based on a Wireless Sensor in Dry Milling Operations. *Sensors* **2016**, *16*, 795. [CrossRef] [PubMed]
14. Bampoula, X.; Nikolakis, N.; Alexopoulos, K. Condition Monitoring and Predictive Maintenance of Assets in Manufacturing Using LSTM-Autoencoders and Transformer Encoders. *Sensors* **2024**, *24*, 3215. [CrossRef]

15. Cakir, M.; Guvenc, M.A.; Mistikoglu, S. The experimental application of popular machine learning algorithms on predictive maintenance and the design of IIoT based condition monitoring system. *Comput. Ind. Eng.* **2021**, *151*, 106948. [CrossRef]
16. Wu, N.; Green, B.; Ben, X.; O'Banion, S. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv* **2020**, arXiv:2001.08317.
17. Wang, J.; Liu, H.; Qi, X.; Wang, Y.; Ma, W.; Zhang, S. Tool wear prediction based on SVR optimized by hybrid differential evolution and grey wolf optimization algorithms. *CIRP J. Manuf. Sci. Technol.* **2024**, *55*, 129–140. [CrossRef]
18. Li, Y.; Huang, X.; Tang, J.; Li, S.; Ding, P. A steps-ahead tool wear prediction method based on support vector regression and particle filtering. *Measurement* **2023**, *218*, 113237. [CrossRef]
19. Kong, D.; Chen, Y.; Li, N.; Tan, S. Tool wear monitoring based on kernel principal component analysis and v-support vector regression. *Int. J. Adv. Manuf. Technol.* **2017**, *89*, 175–190. [CrossRef]
20. Benkedjouh, T.; Medjaher, K.; Zerhouni, N.; Rechak, S. Health assessment and life prediction of cutting tools based on support vector regression. *J. Intell. Manuf.* **2015**, *26*, 213–223. [CrossRef]
21. Chen, B.; Zha, J.; Cai, Z.; Wu, M. Predictive modelling of surface roughness in precision grinding based on hybrid algorithm. *CIRP J. Manuf. Sci. Technol.* **2025**, *59*, 1–17. [CrossRef]
22. Mishra, D.; Pattipati, K.R.; Bollas, G.M. Gaussian mixture model for tool condition monitoring. *J. Manuf. Process.* **2024**, *131*, 1001–1013. [CrossRef]
23. Gao, K.; Xu, X.; Jiao, S. Tool wear prediction based on kernel principal component analysis and least square support vector machine. *Meas. Sci. Technol.* **2024**, *35*, 106129. [CrossRef]
24. Qin, Y.; Liu, X.; Yue, C.; Zhao, M.; Wei, X.; Wang, L. Tool wear identification and prediction method based on stack sparse self-coding network. *J. Manuf. Syst.* **2023**, *68*, 72–84. [CrossRef]
25. He, J.; Yin, C.; He, Y.; Pan, Y.; Wang, Y. Deep multi-task network based on sparse feature learning for tool wear prediction. *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.* **2024**, *238*, 6231–6241. [CrossRef]
26. Sankar, B.R.; Umamaheswarrao, P. Multi objective optimization of CFRP Composite Drilling Using Ant Colony Algorithm. *Mater. Today: Proc.* **2018**, *5*, 4855–4860. [CrossRef]
27. PHM Society. 2010 PHM Society Conference Data Challenge. Available online: www.phmsociety.org/competition/phm/10 (accessed on 1 January 2023).
28. Zhang, H.; He, Q. Tacholess bearing fault detection based on adaptive impulse extraction in the time domain under fluctuant speed. *Meas. Sci. Technol.* **2020**, *31*, 074004. [CrossRef]
29. Hashim, S.; Shakya, P. A spectral kurtosis based blind deconvolution approach for spur gear fault diagnosis. *ISA Trans.* **2023**, *142*, 492–500. [CrossRef]
30. Zhang, X.; Ma, Y.; Pan, Z.; Wang, G. A novel stochastic resonance based deep residual network for fault diagnosis of rolling bearing system. *ISA Trans.* **2024**, *148*, 279–284. [CrossRef]
31. Liu, X.; Li, J.; Bo, L.; Yang, F. Feature-oriented unified dictionary learning-based sparse classification for multi-domain fault diagnosis. *J. Signal Process.* **2024**, *221*, 109485. [CrossRef]
32. Smola, A.J.; Schölkopf, B. A tutorial on support vector regression. *Stat. Comput.* **2004**, *14*, 199–222. [CrossRef]
33. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
34. Zhang, J.; Luan, X.; Liu, F. Multi-manifold NIRS modelling via stacked contractive auto-encoders. *Can. J. Chem. Eng.* **2021**, *99*, 1363–1373. [CrossRef]
35. Hinton, G.E. Reducing the Dimensionality of Data with Neural Networks. *Science* **2006**, *313*, 504–507. [CrossRef]
36. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
37. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
38. Windrim, L.; Ramakrishnan, R.; Melkumyan, A.; Murphy, R.J.; Chlingaryan, A. Unsupervised feature-learning for hyperspectral data with autoencoders. *Remote Sens.* **2019**, *11*, 864. [CrossRef]
39. Wong, S.Y.; Chuah, J.H.; Yap, H.J.; Tan, C.F. Dissociation artificial neural network for tool wear estimation in CNC milling. *Int. J. Adv. Manuf. Technol.* **2023**, *125*, 887–901. [CrossRef]
40. Wu, D.; Jennings, C.; Terpenny, J.; Gao, R.X.; Kumara, S. A comparative study on machine learning algorithms for smart manufacturing: Tool wear prediction using random forests. *J. Manuf. Sci. Eng.* **2017**, *139*, 071018. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.