International Journal of Engineering Inventions
e-ISSN: 2278-7461, p-ISSN: 2319-6491
Volume 14, Issue 1 [January. 2025] PP: 20-31

Remaining useful life prediction of gear based on GWO-GRU model with multiple feature inputs

Yueqian Hou[1], Chengyang Deng[1], Yuanlong Bai[1], Bo Sun[1], Wenting Hu[1], Lei Wang[1], Hao Wang[1],Xuewei Zhao[2]

[1] *School of Mechanical and Vehicle Engineering, Changchun University, Changchun, CHINA*
[2] *Changchun Branch, Lingyun Industrial Co,CHINA*
Corresponding Author: Yueqian Hou

ABSTRACT: *Gears are essential components of mechanical equipment. Accurate estimation of Remaining Useful Life (RUL) not only enhances equipment reliability and reduces maintenance costs, but also significantly improves economic efficiency and safety in industrial production. Recurrent Neural Networks (RNNs) are particularly adept at handling time series data, effectively capturing patterns and trends during gear operation. However, RNNs often face challenges such as gradient explosion and vanishing, which can adversely affect model training effectiveness and stability. The Gated Recurrent Unit (GRU) model addresses these gradient issues inherent in RNNs. Hyperparameters have a direct impact on model performance and training outcomes. Typically, hyperparameter selection relies on manual tuning by researchers, which often fails to identify the optimal combination, resulting in diminished prediction accuracy. Thus, this paper introduces a method for predicting RUL that utilizes the Grey Wolf Optimizer (GWO) to optimize the hyperparameters of the GRU model. The main research focuses include: (1) Given the long degradation period of gears and the large amount of data, 17-dimensional time-frequency domain feature data are extracted. This approach not only reduces data volume but also better characterizes gear degradation trends, with data processed using the Sliding Window Method (SWM) to create training and test sets; (2) The GWO algorithm is utilized to determine optimal hyperparameter combinations for the GRU model, and the performance of this optimized model is compared with that of other models. Experimental results demonstrate that the GWO-GRU model significantly enhances prediction accuracy compared to alternative models, thereby validating the effectiveness of the proposed method.*

Key words: *Gears; Remaining Useful Life; Grey Wolf Optimizer; Gated Recurrent Unit; Hyperparameters, Multi-feature input*

Date of Submission: 02-01-2025 Date of acceptance: 13-01-2025

I. INTRODUCTION

Gears are crucial components in various mechanical systems, including industrial machines, automotive gearboxes, power transmission systems, and aerospace applications [1]. The performance and reliability of these systems are often directly influenced by the condition of their gears. Over time, gears experience degradation due to factors such as material fatigue, wear, lubrication issues, and external loads, all of which can lead to failures if not monitored and managed properly [2]. Consequently, predicting the RUL of gears has become an essential task in modern industrial maintenance practices. Accurate RUL prediction can significantly improve the operational reliability of machinery, reduce maintenance costs, and prevent unexpected downtime, which can be particularly costly in critical applications such as aircraft engines, manufacturing lines, and heavy-duty machinery[3].

Traditional methods of RUL prediction for gears primarily rely on statistical techniques, physical modeling, or expert-driven rules. These methods typically assume predefined degradation patterns and often struggle to handle complex, non-linear failure modes [4]. As a result, they are limited in their ability to provide accurate and real-time predictions for gears operating under varying conditions. In recent years, machine learning (ML) and deep learning (DL) techniques have emerged as promising alternatives for RUL prediction due to their ability to learn complex patterns directly from data without the need for explicit assumptions about the failure process [5]. Among the various ML techniques, time-series models, particularly RNNs, have gained significant attention for their ability to model sequential data and capture the temporal dependencies inherent in the degradation [6].

In addition, hyperparameters significantly influence the model's performance and training outcomes in neural network modeling. The selection of hyperparameters typically relies on the manual debugging methods employed by researchers, which often fail to yield the optimal combination of hyperparameters [7]. Intelligent optimization algorithms are increasingly favored for model parameter optimization due to their strong global

search capabilities, which enable them to escape local optima and identify global solutions. However, conventional methods like Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) have notable limitations. Both GA and PSO are prone to issues arising from a large number of input parameters, which can make the search process computationally expensive and time-consuming. Furthermore, these algorithms exhibit high sensitivity to parameter selection and initial conditions, often leading to suboptimal solutions or slow convergence [8].

Long Short-Term Memory (LSTM) networks, an advanced variant of RNNs, have shown exceptional performance in handling sequential data, especially when dealing with long-term dependencies\cite{lable18}. Zheng S et al.[9] proposed a LSTM model for RUL prediction, capable of leveraging the full sequence of sensor data and uncovering latent patterns across different operating conditions, fault scenarios, and degradation processes. Yousuf S et al.[10] proposes an LSTM-based regression model for predicting the RUL of Ring Oscillator circuits, utilizing essential electrical features and demonstrating high prediction accuracy with minimal deviation. Xiang S et al.[11] proposed an LSTMP-A model that enhances gear remaining life prediction accuracy by fusing time-domain and frequency-domain features with an attention-based weight amplification mechanism. LSTM uses a gating mechanism to mitigate the issues of vanishing and exploding gradients that traditional RNNs often face, making them effective for modeling complex patterns in data [12]. From the perspective of network architecture, LSTM networks consist of three gates: the input gate, the forget gate, and the output gate. These gates enable more precise control over the flow and retention of information. However, when predicting the remaining lifespan of gears, it is often necessary to analyze time series data with multiple features. This complexity demands more intricate structures, which can lead to reduced computational efficiency. Additionally, when handling multi-feature data, LSTM networks require a greater number of parameters, increasing the likelihood of overfitting.

The GRU has emerged as a promising alternative for gear RUL prediction to overcome these shortcomings. Liu H et al.[13] proposed a RUL perdition method based GRU, validated it with experimental data from three sources and showed better performance compared to related literature by addressing existing issues. Sadiqa Zohra et al.[14] proposed hybrid CNN-GRU model for life prediction of lithium-ion batteries, which effectively improves the prediction accuracy. Wen L et al.[15] proposed a method for predicting the remaining life of bearings that combines GRU and Wiener process, and demonstrated through experiments that it outperforms other deep learning methods, highlighting the practical application of information fusion principles in bearing health assessment. Although the GRU offers advantages over LSTM networks in terms of model simplicity and computational efficiency, its performance is still highly dependent on the selection of optimal hyperparameters [16]. To address this challenge, integrating optimization algorithms, such as the GWO, can significantly improve the performance of GRU models by automating the hyperparameter selection process. GWO is a nature-inspired optimization algorithm that mimics the social hunting behavior of grey wolves. It has shown remarkable effectiveness in solving various optimization problems by efficiently exploring the parameter space and avoiding local optima.

To achieve more accurate predictions of the residual life of gear, multiple features from both the time and frequency domains are typically utilized. When a GRU model encounters multi-feature inputs, the parameter space becomes increasingly complex. The GWO algorithm demonstrates strong global search capabilities, enabling it to effectively navigate this high-dimensional parameter space. Moreover, predicting the residual life of gears often necessitates timely results to inform maintenance decisions. The GWO algorithm converges at a relatively rapid rate, facilitating the identification of optimal GRU parameters in a shorter timeframe. This indicates that an effective prediction model can be established more quickly, allowing for the early initiation of residual life predictions for gears.

This paper proposes the use of GWO to optimize the hyperparameters of the GRU model as figure 1 shown. By leveraging the search capabilities of GWO to identify the optimal hyperparameter configuration, the performance of the GRU model can be significantly improved without the need for manual tuning. This approach combines the strengths of GRU in capturing sequential patterns and the efficiency of GWO in optimizing complex models, resulting in a powerful tool for accurate and reliable RUL prediction for gears.

This paper is organized as follows: Section II describes the related background; Section III presents the extracted 17-dimensional features and discusses the GWO hyperparameter optimization algorithm for the GRU model; Section IV details the experimental setup and evaluates the model's advantages compared to other models using the same data; finally, Section V provides the conclusion of the paper.

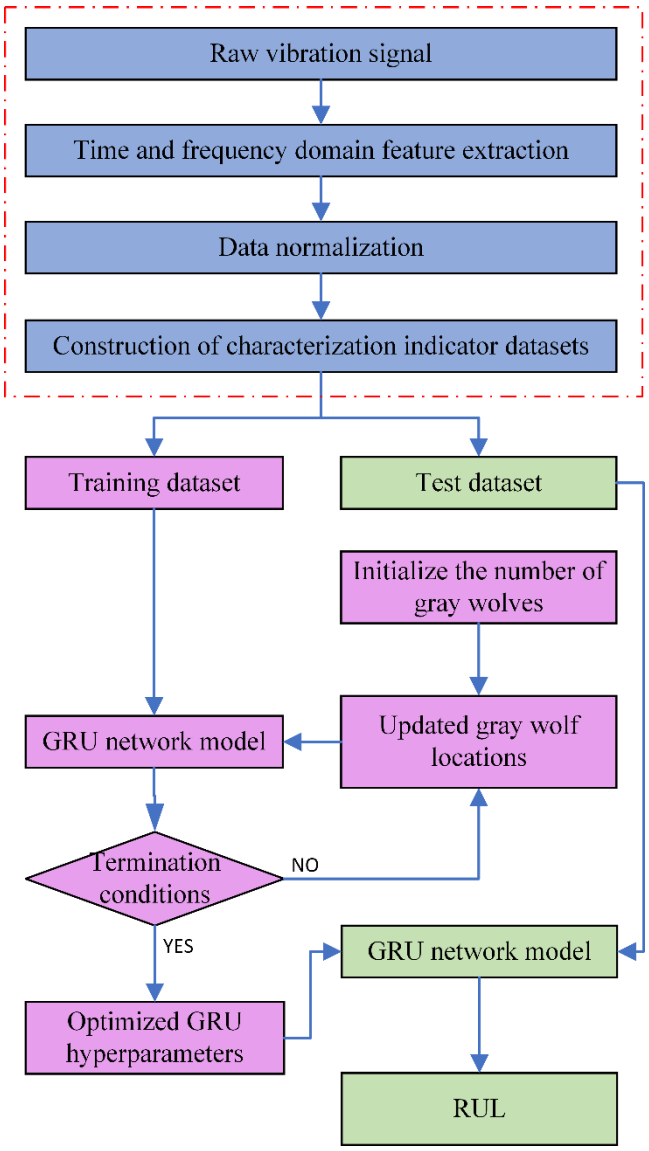


Fig. 1 GWO-GRU Model Flowchart

II. BACKGROUND

GRU Neural Network

The GRU network model introduced by Cho et al.[17] in 2014. The model consists of two inputs and one output: the two inputs are the input value X_t at time t and the hidden state H_{t-1} at the previous time $t-1$; the outputs are the hidden state H_t at time t; the reset gate R_t, the update gate Z_t, and a candidate hidden state $\tilde{H}_t$. The internal structure of the GRU model is shown in Fig. 2.

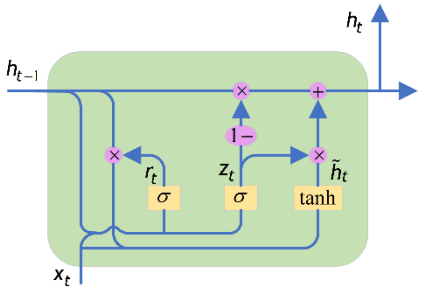


Fig. 2 Basic Structure Diagram of GRU

According to the information flow transfer path of the GRU model, we first need to calculate the update gate Z_t and reset gate R_t , which are shown in the following formulas:

$$R_t = \sigma(W_{rx}X_t + W_{rh}H_{t-1} + b_r)$$
$$Z_t = \sigma(W_{zx}X_t + W_{zh}H_{t-1} + b_z)$$

where σ is the $Sigmoid$ activation function, W_{rx}, W_{zx} and W_{rh}, W_{zh} denote the input value X_t and the weight parameter of the hidden state H_{t-1}corresponding to the reset gate and the update gate, and b_r, b_z is the bias value of the corresponding gating structure. Next, the candidate hidden state $\widetilde{H}_t$ is computed for calculating the hidden state for the next time period, and its calculation formula is shown below:

$$\widetilde{H}_t = tanh(W_{hx}X_t + W_{hh}(R \odot H_{t-1}) + b_h)$$

W_{hx}, W_{hh} are the weight parameters of the candidate hidden state, $tanh$ is the hyperbolic tangent activation function, b_h is the bias value of the candidate hidden state, and $\odot$ is the Hadamard product. The role of the reset gate R_t is used to control whether the candidate hidden state $\widetilde{H}_t$ is dependent on the hidden state of the previous time period H_{t-1}. When $R_t = 0$, the hidden state H_{t-1} of the previous time period is discarded, while the hidden state of the previous time period is retained when $R_t \neq 0$, and the larger the value of R_t, the more information is retained.

Finally, the two values of gate Z_t and $\widetilde{H}_t$candidate hidden state $H_t$are updated to compute the hidden state Ht for use in the output layer, which is computed as shown below:

$$H_t = Z_t \odot H_{t-1} + (1 - Z_t) \odot \widetilde{H}_t$$

The update gate Z_t controls how much historical information is retained by the hidden state H_t from the previous moment and how much information is received from the candidate hidden state $\widetilde{H}_t$, and finally integrates the two into a single output to the hidden state H_t at the current moment.

GWO algorithm

The GWO algorithm, introduced by Mirjalili et al. [18] in 2014, is an intelligent optimization algorithms that emulates the hunting behavior of gray wolves. This algorithm is distinguished by its minimal parameter requirements and improved convergence capabilities. Gray wolves are social animals that live in packs, where a strict hierarchy governs their interactions. Each level of the pack has specific roles and responsibilities, allowing for seamless cooperation during hunting activities. Figure 3 illustrates the social hierarchy within wolf packs.

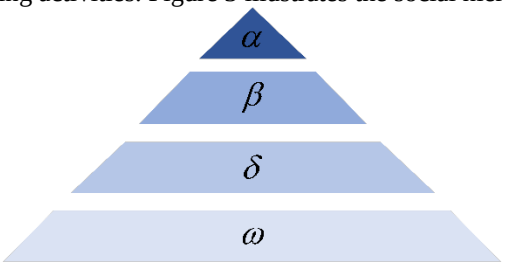


Fig. 3 Gray wolf social class stratification

The gray wolf social system is generally composed of four levels α, β, δ and ω. where a represents the position of the alpha wolf, the leader of the pack responsible for guiding the hunt. This position signifies the optimal solution in the optimization algorithm, corresponding to the highest fitness value; β wolves are the suboptimal solution in the algorithm, with the second highest fitness; δ wolves are the third tier of the pack, which represents the third-best solution in the algorithm, with the third-highest fitness value; The fourth layer of the wolf pack, ω, denotes all individuals in the pack except α, β, and δ, with α, β, and ω leading ω to complete the position update. Therefore, β, δ are generally denoted as the leading wolves. The whole arithmetic process of the gray wolf optimization algorithm can be divided into:

During hunting, wolves can pinpoint the location of their prey and encircle them. This behavior can be simulated by mathematical modeling with the following equation:

$$\vec{D} = |\vec{C} * \vec{X}_p(t) - \vec{X}(t)|$$
$$\vec{D} = |\vec{C} * \vec{X}_p(t) - \vec{X}(t)|$$

where t denotes the number of iterations, $\vec{A}$ and $\vec{C}$ denote the coefficient vectors, $\vec{X}_p$ denotes the position vector of the prey, and $\vec{X}$ denotes the position vector of the gray wolf. Where $\vec{A}$, $\vec{C}$ are calculated as shown below:

$$\vec{A} = 2\vec{a} * \overrightarrow{r_1} - \vec{a}$$
$$\vec{C} = 2\overrightarrow{r_2}$$

where $\vec{a}\,\vec{a}$ is decreasing with the number of iterations, decreasing linearly from 2 to 0, and $\overrightarrow{r_1}, \overrightarrow{r_2}$ denotes a random number between 0 and 1.

GWO is used in practice because the optimal solution for its location is always unknown, under the leadership of the α wolf king and β , δ wolves help the wolf king α to determine the location of the pack and direct the ω wolves to update their locations to complete the hunt. The following equation describes this behavior:

$$\vec{D}_\alpha = |\vec{C}_1 * \vec{X}_\alpha - \vec{X}|$$
$$\vec{D}_\beta = |\vec{C}_2 * \vec{X}_\beta - \vec{X}|$$
$$\vec{D}_\delta = |\vec{C}_3 * \vec{X}_\delta - \vec{X}|$$
$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 * (\vec{D}_\alpha)$$
$$\vec{X}_2 = \vec{X}_\beta - \vec{A}_2 * (\vec{D}_\beta)$$
$$\vec{X}_3 = \vec{X}_\delta - \vec{A}_3 * (\vec{D}_\delta)$$
$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3}$$

where $\vec{X}_\alpha$, $\vec{X}_\beta$, $\vec{X}_\delta$ denote the positions of α, β, δ wolves at the current moment, $\vec{X}_1$, $\vec{X}_2$, $\vec{X}_3$ denote the distances and directions that the ω wolves need to move towards the first three wolves, $\vec{X}_t$ and $\vec{X}_{t+1}$ denote the final positions of the current individual and the current individual in this iteration, $\vec{D}_\alpha$, $\vec{D}_\beta$, $\vec{D}_\delta$ denote the distances between the head wolf and the current individual. $\vec{A}_1$, $\vec{A}_2$, $\vec{A}_3$, $\vec{C}_1$, $\vec{C}_2$, $\vec{C}_3$denote the coefficient vectors.

III. GWO-GRU model prediction methods

Time and frequency features

To address noise interference in the original vibration signals, wavelet denoising preprocessing was applied to the collected signals. Statistical characteristics were extracted as features from both the time domain and frequency domain, using samples of a specific signal length. A total of thirteen time-domain features and four frequency-domain metrics were identified to construct the dataset [19], where t_1 to t_{13} represent the time-domain features, and f_1 to f_4 represent the frequency-domain features. The frequency-domain features were derived by applying a Fast Fourier Transform to the vibration signals, generating the frequency spectrum and envelope spectrum, from which the relevant features were extracted. Table 1 lists the names and calculation formulas of these time-domain and frequency-domain features. The rationale for selecting these specific features is elaborated below:

Table 1: Time-DomainandFrequency-DomainFeatureExtraction

Name	Formula	Name	Formula
Peak-to-peak	$t_1 = MAX\|x(i)\| - MIN\|x(i)\|$	ClearanceFactor	$t_{10} = MAX\|x(i)\|/(\frac{1}{N}\sum_i^N \|x(i)\|^{1/2})^2$
Peak	$t_2 = MAX\|x(i)\|$	SkewnessFactor	$t_{11} = \frac{1}{N}\sum_i^N \|x(i)\|^3 / (\sqrt{\frac{1}{N}\sum_i^N x(i)^2})^3$
Mean	$t_3 = \frac{1}{N}\sum_i^N x(i)$	KurtosisFactor	$t_{12} = \frac{1}{N}\sum_i^N \|x(i)\|^4 / (\sqrt{\frac{1}{N}\sum_i^N x(i)^2})^4$
Variance	$t_4 = \frac{1}{N}\sum_i^N (x(i) - \overline{x})^2$	WaveformFactor	$t_{13} = \sqrt{\frac{1}{N}\sum_i^N x(i)^2} / \frac{1}{N}\sum_i^N \|x(i)\|$
RootMean Square	$t_5 = \sqrt{\frac{1}{N}\sum_i^N x(i)^2}$	Mean Frequency	$f_1 = \frac{1}{N}\sum_j^N X(j)$
AbsoluteMean Amplitude	$t_6 = \frac{1}{N}\sum_i^N \|x(i)\|$	Frequency Center	$f_2 = \sum_j^N (f(j)^2 X(j)) / \sum_j^N X(j)$
ImpulsiveFactor	$t_7 = MAX\|x(i)\| / \frac{1}{N}\sum_i^N \|x(i)\|$	RMS Frequency	$f_3 = \sqrt{\sum_j^N (f(j)^2 X(j)) / \sum_j^N X(j)}$
Standard Deviation	$t_8 = \sqrt{\frac{1}{N}\sum_i^N (x(i) - \overline{x})^2}$	Standard deviation frequency	$f_4 = \sqrt{\sum_j^N ((f(j) - f_c)^2 X(j)) / \sum_j^N X(j)}$
PeakFactor	$t_9 = MAX\|x(i)\| / \sqrt{\frac{1}{N}\sum_i^N x(i)^2}$		

(1) Gear degradation typically occurs over an extended period, generating large volumes of data. By extracting time-frequency domain features, this data can be significantly reduced, thereby enhancing processing efficiency. This approach eliminates the need for continuously deepening network structures for feature extraction, which can complicate network optimization.
(2) In real-time dynamic monitoring, time-frequency domain features of vibration signals can effectively characterize the degradation state of gears.
(3) The feature dataset constructed with time-frequency domain features displays distinct sequential patterns and trends, making it particularly suitable for input into a GRU network structure.
(4) Relying on a single feature to assess gear degradation is limited and may result in the loss of critical information.
(5) The GRU algorithm focuses on learning and analyzing sequential trends and intrinsic connections within the data. It can independently determine which feature information to retain or discard, eliminating the need for specific extraction of sensitive features. In fact, incorporating a variety of features allows the model to learn more

comprehensively, resulting in performance that aligns more closely with real-world scenarios.

Data normalization

Data normalization, also referred to as standardization of data deviation, is a process that scales data to a uniform range. In this study, the min-max normalization method was applied to rescale the data to the range [0, 1]. Normalization is commonly applied to evaluation indicators to mitigate the effects of differing units on the overall dataset, thereby facilitating the weighting and comparison of various indicators. The main considerations following data normalization are outlined below:

$$x^* = \frac{x - Min}{Max - Min}$$

Where: Min is the maximum value among all sample data; Max is the minimum value among all sample data. Construct a new feature dataset after normalizing the data.

Forecasting methodology process

This paper proposes utilizing the processing capabilities of the GRU neural network for predicting gear lifespan based on time series data. The primary approach involves first extracting seventeen feature indices from the original vibration signals to create a seventeen-dimensional feature dataset, followed by normalizing the fea

ture data. The SWM is employed to extract continuous subsequences from the original data by moving a fixed-size window. Each window contains a series of consecutive data points, and as the window slides, overlapping subsequences are generated. This method is widely used in time series prediction and signal processing [20]. In this study, we apply the SWM to process the dataset and construct training and test samples. Figure 4 illustrates the schematic representation of the SWM.

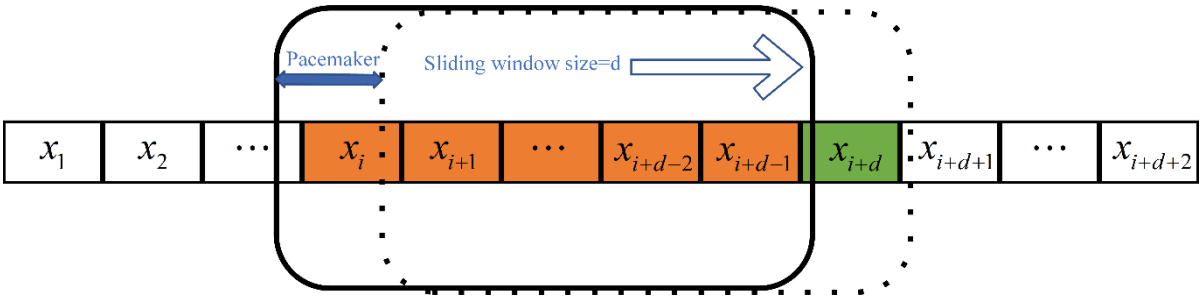


Fig. 4 Schematic diagram of the SWM

The GRU model and the GWO algorithm are constructed, with the GWO population initialized such that each individual represents a unique hyperparameter combination. The loss function is used as the fitness value to evaluate each hyperparameter combination. Based on fitness values, the top three wolves with the highest fitness are selected to guide the search direction of the population. Iterative steps are repeated to narrow the search space, and the optimal hyperparameter combination is identified once fitness values converge. This optimal combination is then applied to train the GRU model. Figure 5 shows the pseudocode for GWO optimization of GRU hyperparameters. Finally, the training data is input into the model for training, and upon completion, the test set is used to assess the model's predictive performance.

Algorithm: GWO-GRU Hyperparameter Optimization Algorithm

Require: Initial population of wolves with random GRU hyperparameters.
Ensure: Optimized hyperparameter configuration for GRU.

Initialize wolf population with random GRU hyperparameters.
Define α , β , and δ as the top three wolves based on fitness.
Set maximum iterations max_iter and control parameter a.
for iteration = 1 to max_iter **do**
    Fitness Evaluation:
    for each wolf in the population **do**
        Train GRU with wolf's hyperparameters.
    Evaluate fitness based on performance (e.g., accuracy, loss).
    end for
    Update α, β, and δ:
    Update the top three wolves (α, β , and δ) based on fitness.
    Position Update:
    for each wolf in the population **do**
        Adjust wolf's hyperparameters based on α, β , and δ.
    end for
    Update Control Parameter a:
    Reduce the value of a as iterations progress to fine-tune exploration and exploitation.
    Check for Convergence:
    if convergence criteria met (e.g., fitness improvement below threshold) **then**
        Stop and return the best wolf configuration.
    end if
end for
Final Training and Evaluation:
Train GRU with the optimized hyperparameters from the best wolf.
Evaluate the final GRU model on the test dataset.
return The best GRU hyperparameter configuration.

Fig. 5 GWO-GRU model pseudocode

IV. Experimental validation

Introduction to data

To assess the feasibility of the proposed algorithm, a publicly available gear life cycle dataset from Chongqing University was employed to test the method outlined in this study [21]. The experiments were conducted using the gear contact fatigue test rig shown in Figure 6, which consists of four components: the gear operating platform, the torque control system, the cooling and lubrication system, and the experimental operating platform. The experimental gears were fabricated from 20CrMnMo material, with a normal modulus of 5 and 24 and 26 teeth, respectively. The lubricant flow rate for the experiment was set at 4 L/h, with the cooling temperature kept at a maximum of 70°C. Under normal operating conditions, the gears exhibit a long lifespan; however, an acceleration experiment was performed to obtain degradation data. An acceleration sensor installed on the gearbox shell recorded the state detection signals of the test gears. The gear speed was set at 1000 r/min, and the torque was maintained at 1300 N·m. The sensor operated at a sampling frequency of 50 kHz, with sampling occurring from the 41st to the 50th second of each minute for 10 seconds. Each sample comprised 500,000 data points, and two sets of gear degradation data were used for the experiment, obtained under the same torque and speed conditions. Most of the samples were collected during the gear break-in period and stable operation, with the last 600 minutes of vibration data used as the degradation data for the gears, resulting in a total of 600 samples in the dataset.

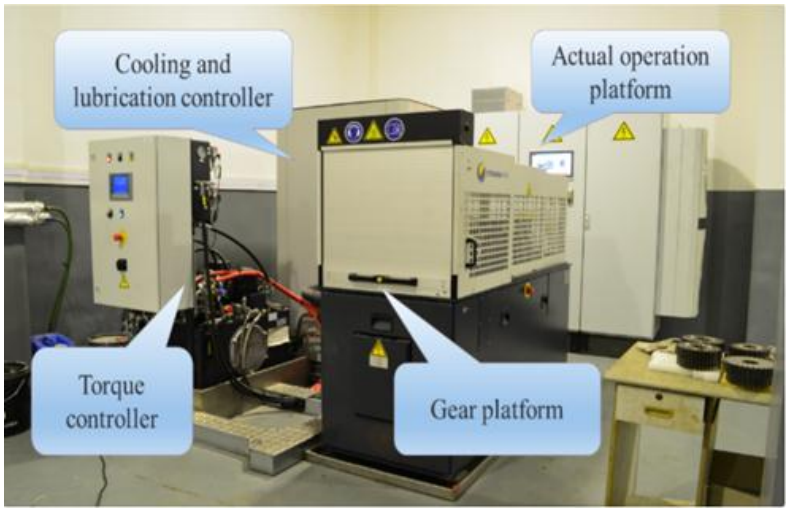


Fig. 6 Accelerated test rig for gear degradation

Each sample consists of 500,000 data points collected over a duration of 10 seconds, resulting in a total dataset of up to 300 million data points. The substantial size of this dataset imposes significant challenges for feature extraction; therefore, the data for each sample undergoes preprocessing in both the time and frequency domains. We calculated 13 features in the time domain and 4 features in the frequency domain for each sample, yielding a total of 17-dimensional features and constructing a feature dataset of 17 × 600 dimensions. Subsequently, the dataset is processed using SWM after normalization, and it is divided into a training set and a test set. The step size for SWM is set to 1, and the window size is set to 30.

Next, the actual RUL of the gears is utilized as a label, with a value of 1 indicating an intact, theoretical state, while a value of 0 signifies that the gear has reached a failure state. This gear characterization dataset comprises a total of 600 data entries, with the cumulative life of the gears amounting to 600 minutes. It is important to note that this duration does not represent the true lifespan of the gears, as it excludes the time spent in stable operation and instead reflects the remaining life of the gears during the degradation phase. For instance, if we consider the current sample to be the 300th data point, the remaining life of the gear would be 300 minutes. Consequently, the corresponding label value for this sample would be calculated as 300/600 = 0.5. This process continues to construct the RUL labels for the gears.

The model was designed using the open-source deep learning framework PyTorch. The hardware and software specifications included an Intel(R) Core(TM) i5-12500H CPU (3.1 GHz), WIN10 64-bit operating system, NVIDIA RTX3050 GPU, Python 3.9, and PyTorch version 1.13. CUDA version 11.6 was installed to enable GPU acceleration.

Indicators for model evaluation

The performance of the constructed model is typically evaluated using three primary metrics: the Mean Absolute Error (MAE),the Root Mean Square Error (RMSE) and the coefficient of determination R^2. The MAE measures the average absolute difference between predicted and actual values, while the RMSE quantifies the deviation between these values by calculating the square root of the sum of the squared differences [22]. R^2, an essential metric, reflects the model's goodness of fit. A combined assessment using these three metrics enables a more precise and comprehensive evaluation of the model's performance. The calculations for these metrics are as follows:

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

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(\bar{y}_i - y_i)^2}$$

Where y_i represents the true value, $\hat{y}_i$ denotes the predicted value, n indicates the number of samples,

and $\bar{y}$ denotes the mean value. The values of MAE and RMSE are inversely related to model accuracy; lower values signify higher prediction accuracy and stronger model performance. The closer the R^2 value is to 1, the better the model fits the data and the higher its accuracy.

Comparative analysis of prediction algorithms

To evaluate the impact of various data split rsetatios on model performance, four distinct ratios were established: 50\% training set and 50\% test set; 70\% training set and 30\% test set; 80\% training set and 20\% test set; and 90\% training set and 10\% test set. The model's performance was assessed using three evaluation metrics: MAE, RMSE, and R², to analyze the effect of the split ratio on the model's performance. Table 2 presents the performance metrics computed based on the test dataset to evaluate the algorithm's performance at different data split ratios.

Table 2: Performance of the Model with Different Data Splits

Ratios	Number of training samples / Number of test samples	MAE	RMSE	R2
5:5	300/300	0.3992	0.4241	-7.6026
7:3	420/180	0.2913	0.2951	-10.5728
8:2	480/120	0.0265	0.0328	0.6782
9:1	530/70	0.0068	0.0087	0.8924

As shown in the table 2, when the training set ratios are 50\% and 70\%, the R² values are negative, indicating that the model is overfitting. However, at the 80\% and 90\% training set ratios, the R^2 values are positive. The R^2 value for the 90\% training set ratio is closer to 1 compared to the 80\% ratio, with MAE and RMSE values approaching 0. Therefore, the first 530 samples were selected as the training set for model training, and the remaining 70 samples were used as the test set for model evaluation.

The GRU model has four main hyperparameters: batch size, learning rate, number of hidden layer units, and number of hidden layers. This paper tests the proposed algorithm against GRU, LSTM, GA-GRU, and PSO-GRU models. First, the GRU model is compared to the LSTM model. The hyperparameters for manual tuning of the GRU model are as follows: batch size of 32, learning rate of 0.001, 64 hidden layer units, and 2 hidden layers. For the LSTM model, the parameters for manual tuning include a feature dimension of 17, a learning rate of 0.001, a hidden state dimension of 64, and 2 layers. Both models were evaluated in the same environment, with the GRU model achieving a training time of 7.69 seconds, significantly less than the LSTM model's training time of 10.47 seconds. Three optimization algorithms-GWO, GA, and PSO—were employed to optimize the four hyperparameters of the GRU model. The initial values for the GWO, GA, and PSO algorithms were established to obtain the combinations of GRU hyperparameters that correspond to the minimum fitness function. The time taken by GWO, PSO, and GA to seek the optimal fitness value with the same number of iterations and initial parameters is 24 min 18 sec, 28 min 17 sec, and 35 min 52 sec, respectively, which shows that the time taken by GWO is significantly less than that of GA and PSO. Table 3 presents the initial parameters for each optimization algorithm.

Table 3: Initialparametersofeachoptimizationalgorithm

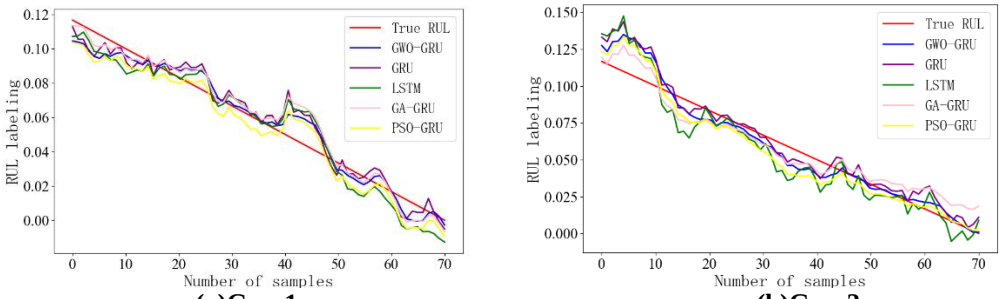
GWO		GA		PSO	
Parameters	Value	Parameters	Value	Parameters	Value
Population Size	10	Population Size	10	Particle Swarm Size	10
Max Iterations	30	Max Iterations	30	Max Iterations	30
Dimension	4	Dimension	4	Dimension	4
		Crossover Rate	0.8	Individual Learning Factor	2
		Variation Rate	0.1	Social Learning Factor	2
				Inertia Weight	0.5

The hyperparameters of the GRU model, optimized using GWO, GA, and PSO, along with the manually tuned GRU parameters, are presented in Table 4.

Table 4: Initialparametersofeachalgorithm

	GWO-GRU	GA-GRU	PSO-GRU	GRU
Batch Size	30	30	30	30
Learning Rate	0.0005	0.001	0.0017	0.001
Number of Hidden Units	95	64	68	64
Number of Hidden Layers	4	3	3	2

Five models were employed to predict two sets of gear data. Figure 7 illustrates the remaining service life prediction curves for these five models. As depicted in the figure, the GWO-GRU model aligns more closely with the actual life curve than the other algorithms. Additionally, this model exhibits a smaller amplitude of predictive curve fluctuations and demonstrates higher prediction accuracy.

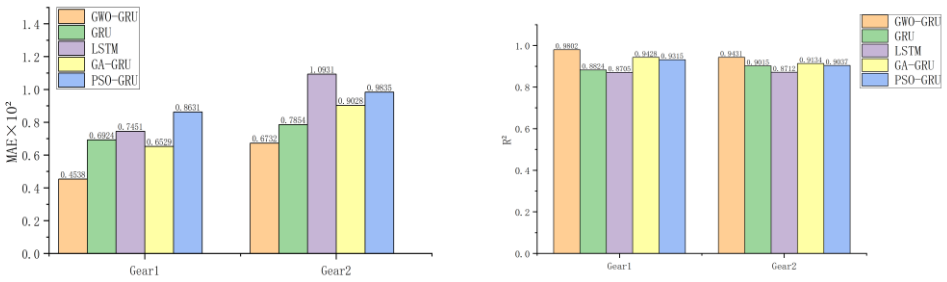


(a)Gear1 **(b)Gear2**

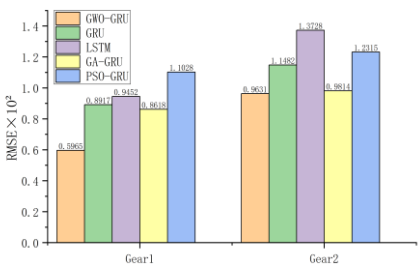
Fig. 7 Comparison of the Predictive Effects of Different Models:(a) Gear1 and (b) Gear2

After optimizing the models, there is some randomness in the prediction process due to the models. Therefore, all five models we trained and tested 100 times and calculated the average of MAE, RMSE and R^2 to enhance the robustness of the models.

The results of the average values of the evaluation metrics for the different models over 100 training and testing sessions are shown in Figure 8, (a) MAE values of the different models on the two sets of data, (b) indicates the RMSE values of the different models on the two sets of data,and (c) shows the R^2 values of the different models on the two sets of data. The figure shows that the MAE values of the GWO-GRU model for gear 1 and gear 2 are 0.004538 and 0.006732, respectively, with RMSE of 0.005965 and 0.009631. Compared to the other four models, the evaluation metrics of the GWO-GRU model converge more closely towards 0, and the GWO-GRU has a greater enhancement of the prediction accuracy. Among them, the MAE values of GWO-GRU model on gear 1 were 34.47\%, 39.14\%, 30.50\% and 47.47\% lower than those of the manually tuned parameterized GRU model, LSTM model, and GA-GRU and PSO-GRU models, and the MAE values on gear 2 were 14.27\%, 38.38\%, 25.44\% and 31.57\%; the RMSE values of the GWO-GRU model on gear 1 were 33.09\%, 36.96\%, 30.74\% and 45.91\% lower than those of the manually-tuned GRU model, the LSTM model, and the GA-GRU and PSO-GRU models, respectively, and those on gear 2 were 16.12\%, 29.84\%, 1.86\%, and 21.83\% lower than those of the manually-tuned GRU model, the LSTM model, and the PSO-GRU model. The R^2 values of the GWO-GRU model on Gear 1 and Gear 2 are 0.9802 and 0.9431, respectively, and the R^2 values are closer to 1 compared to other models, indicating that the model has a significant advantage in prediction accuracy.



(a) MAE **(c) R^2**



(b) RMSE

Fig. 7 Histograms of the Results of the Evaluation Metrics for the Different Models: (a) Mean Absolute Error, (b) Root-Mean-Square Error and (c) coefficient of determination.

In summary, the GWO-GRU model outperforms other models in prediction accuracy. This enhancement is likely attributed to the robust search capabilities of the GWO algorithm in hyperparameter optimization, which enables the GRU model to achieve more accurate parameter settings.

Although the GWO-GRU model demonstrates strong performance across various scenarios, certain limitations remain. For instance, while the GWO algorithm significantly outperforms both the GA and PSO algorithms in convergence efficiency, it still requires approximately 24 minutes to process the 17×600 dataset. A comparison of the MAE, RMSE, and R^2 values for the model on Gear1 and Gear2 indicates better performance on Gear1 than Gear2, suggesting that the model's performance is more dependent on high-quality data.

V. Conclusion

In this paper, a method for predicting the RUL based on multi-feature input is proposed. The GRU is utilized to handle time series data, and the GWO is employed to optimize the hyperparameters of the GRU model. To verify the feasibility of the proposed method, a publicly available gear life cycle dataset was used. Firstly, the impact of different ratios of the training set to the test set on the prediction accuracy was compared and analyzed. It was found that a ratio of 90\% for the training set and 10\% for the test set could achieve the best prediction accuracy. Subsequently, the performance of the GWO-optimized GRU model was compared with the traditional GRU and LSTM models, as well as GRU models optimized using GA and PSO. The experimental results showed that the GWO-GRU model corresponded more closely to the actual gear life curve in comparison to the other models. Additionally, it demonstrated high prediction accuracy and minimal fluctuation. This method can effectively tackle the issue of poor accuracy resulting from manual parameter adjustment and can prevent the occurrence of gradient explosion or disappearance during the prediction process.

In the future, this method can be extended to other mechanical components to improve the utility of the method. In addition, integrating other optimization algorithms to form a hybrid optimization algorithm can enhance the global search capability of GWO, accelerate the convergence speed, and explore the time-frequency domain as a potential feature, which can provide a new perspective for understanding the degradation process of mechanical components.

Acknowledgements

This work was supported by the project “Research on key technology of CNC equipment health management based on data-driven” (No. 2024JB401L07) of Jilin Provincial Department of Science and Technology, and the project “Research on construction of CNC equipment digital twin model based on parameter migration learning and maintenance decision-making” (No. 2023LY501L06) of Jilin Provincial Department of Education.

REFRENCES

[1] D. T. Jelaska, Gears and Gear Drives. Hoboken, NJ, USA: John Wiley Sons, 2012.
[2] K. Feng, J. C. Ji, Q. Ni, et al., "A review of vibration-based gear wear monitoring and prediction techniques," Mechanical Systems and Signal Processing, vol. 182, p. 109605, 2023.
[3] A. Martins, B. Mateus, I. Fonseca, et al., "Predicting the health status of a pulp press based on deep neural networks and hidden Markov models," Energies, vol. 16, no. 6, p. 2651, 2023.
[4] O. Fink, Q. Wang, M. Svensen, et al., "Potential, challenges and future directions for deep learning in prognostics and health management ap plications," Engineering Applications of Artificial Intelligence, vol. 92, p. 103678, 2020.
[5] T. F. De Barrena, J. L. Ferrando, A. García, et al., "Tool remaining useful life prediction using bidirectional recurrent neural networks (BRNN)," The International Journal of Advanced Manufacturing Technology, vol. 125, no. 9, pp. 4027–4045, 2023.
[6] Z. Tian and M. J. Zuo, "Health condition prediction of gears using a recurrent neural network approach," IEEE Transactions on Reliability, vol. 59, no. 4, pp. 700–705, 2010.
[7] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," Neurocomputing, vol. 415, pp. 295–316, 2020.
[8] S.Hochreiter,"Longshort-termmemory,"NeuralComputationMIT-Press, 1997.
[9] S. Zheng, K. Ristovski, A. Farahat, et al., "Long short-term memory network for remaining useful life estimation," in 2017 IEEE International Conference on Prognostics and Health Management (ICPHM), IEEE, 2017, pp. 88–95.
[10] S. Yousuf, S. A. Khan, and S. Khursheed, "Remaining useful life (RUL) regression using long–short term memory (LSTM) networks," Microelec tronics Reliability, vol. 139, p. 114772, 2022.
[11] S. Xiang, Y. Qin, C. Zhu, et al., "Long short-term memory neural network with weightamplificationanditsapplicationinto gearremainingusefullife prediction," Engineering Applications of Artificial Intelligence, vol. 91, p. 103587, 2020.
[12] M. Ma and Z. Mao, "Deep-convolution-based LSTM network for remain ing useful life prediction," IEEE Transactions on Industrial Informatics, vol. 17, no. 3, pp. 1658–1667, 2020.
[13] H. Liu, Y. Li, L. Luo, et al., "A lithium-ion battery capacity and RUL prediction fusion method based on decomposition strategy and GRU," Batteries, vol. 9, no. 6, p. 323, 2023.
[14] S. Zohra, et al., "A hybrid CNN-GRU model for remaining useful life prediction of lithium-ion batteries," Energies, vol. 15, p. 2387, 2022.
[15] L. Wen, S. Su, and X. Li, "GRU-AE-wiener: A generative adversarial network assisted hybrid gated recurrent unit with Wiener model for bearing remaining useful life estimation," Mechanical Systems and Signal Process ing, vol. 220, p. 111663, 2024.
[16] J. Zhou, Y. Qin, D. Chen, et al., "Remaining useful life prediction of bear ings by a new reinforced memory GRU network," Advanced Engineering Informatics, vol. 53, p. 101682, 2022.
[17] K. Cho, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," arXiv preprint arXiv:1406.1078, 2014.

[18] S.Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," Advances in Engineering Software, vol. 69, pp. 46-61, 2014.
[19] S.Xiang,Y.Qin,C.Zhu,Y.Wang,andH.Chen,"Longshort-termmemory neural network with weight amplification and its application into gear remaining useful life prediction," Engineering Applications of Artificial Intelligence, vol. 91, p. 103587, 2020.
[20] P. K. R. Prakarsha and G. Sharma, "Time series signal forecasting using artificial neural networks: An application on ECG signal," Biomed. Signal Process. Control, vol. 76, p. 103705, 2022.
[21] Y.Qin,J. Yang, J. Zhou, H. Pu, X. Zhang, andY.Mao,"Dynamicweighted federated remaining useful life prediction approach for rotating machin ery," Mechanical Systems and Signal Processing, vol. 202, p. 110688, 2023.
[22] W. WangandY.Lu, "Analysis of the Mean Absolute Error (MAE) and the Root MeanSquare Error (RMSE)in assessing rounding model," IOP Con ference Series: Materials Science and Engineering, vol. 324, p. 012049, 2018.