Deep Bidirectional Recurrent Neural Networks Ensemble for Remaining Useful Life Prediction of Aircraft Engine

Kui Hu, Yiwei Cheng, Jun Wu, *Member, IEEE*, Haiping Zhu, and Xinyu Shao

***Abstract*—Remaining useful life (RUL) prediction of aircraft engine (AE) is of great importance to improve its reliability and availability, and reduce its maintenance costs. This article proposes a novel deep bidirectional recurrent neural networks (DBRNNs) ensemble method for the RUL prediction of the AEs. In this method, several kinds of DBRNNs with different neuron structures are built to extract hidden features from sensory data. A new customized loss function is designed to evaluate the performance of the DBRNNs, and a series of the RUL values is obtained. Then, these RUL values are reencapsulated into a predicted RUL domain. By updating the weights of elements in the domain, multiple regression decision tree (RDT) models are trained iteratively. These models integrate the predicted results of different DBRNNs to realize the final RUL prognostics with high accuracy. The proposed method is validated by using C-MAPSS datasets from NASA. The experimental results show that the proposed method has achieved more superior performance compared with other existing methods.**

***Index Terms*—Aircraft engine (AE), bidirectional recurrent neural networks (BDRNNs), deep learning, ensemble learning (EL), prognostics and health management (PHM), remaining useful life.**

I. Introduction

AIRCRAFT engine (AE) is a core part of various kinds of airplanes, and its reliability is very important to ensure the safety of the aircraft [1], [2]. To improve the reliability and availability of the AE and reduce its maintenance costs, prognostics and health management (PHM) has become a research hotspot in the field of the AE recently. PHM of the AE involves analyzing the data acquired by various types of sensors, using the artificial intelligence (AI) algorithm to evaluate health status, forecast the failure before the AE breaks down, and providing a series of maintenance suggestions combined with the existing resource. As an important part of PHM, a well-designed remaining useful life (RUL) prediction method can help engineers arrange maintenance and repairing periodicity of the AE reasonably, which effectively prevents its performance degradation or even catastrophic failure [3]–[6].

In recent years, RUL prediction methods based on AI and big data mining have attracted a wide range of attention. Those methods mainly focus on finding the mapping relationship between massive monitoring data and the corresponding RUL label by using advanced machine learning algorithms. The machine learning algorithms, including artificial neural networks [7], support vector machine (SVM) [8], hidden Markov model [9], etc., are suitable for inferring hidden correlation and causality from monitoring data and learning degradation trend. However, these sophisticated machine learning methods still have some shortcomings.

1) Most machine learning algorithms can only calculate the output according to the current input data [10]. However, in industrial applications, the degradation process occurs in the entire life cycle. The correlations of sequence data at different times will affect the prediction accuracy of the model. Therefore, an algorithm that can capture the time dependence in sequence data for RUL prediction has a very broad prospect.
2) Due to the changing operating environment and working modes, a single machine learning algorithm usually has some limitations in extracting features and fitting curves. It means that the trained machine learning model only has high accuracy in a certain situation and might not work well in other situations.

To overcome the shortcomings mentioned above, a new deep bidirectional recurrent neural networks (DBRNNs) ensemble method is proposed in this article for accurate and robust RUL prediction of the AEs. In this method, multiple bidirectional recurrent neural networks (RNNs) are introduced to capture and mine features, which could utilize information from sequence data in forward and backward directions to capture long-term dependencies. Besides, ensemble learning (EL) is considered as a good cut-in point to increase the robustness of the built model. By integrating several DBRNN models under different conditions using multiple regression decision

Manuscript received 18 February 2021; revised 23 July 2021; accepted 27 October 2021. Date of publication 25 November 2021; date of current version 16 March 2023. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1702300; in part by the National Natural Science Foundation of China under Grant 51875225; and in part by the Key-Area Research and Development Program of Guangdong Province, China, under Grant 2019B090916001. This article was recommended by Associate Editor M. Han. *(Kui Hu and Yiwei Cheng contributed equally to this work.) (Corresponding author: Jun Wu.)*

Kui Hu and Jun Wu are with the School of Naval Architecture and Ocean Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: hu_kui@hust.edu.cn; wuj@hust.edu.cn).

Yiwei Cheng, Haiping Zhu, and Xinyu Shao are with the School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: chengyiwei102@163.com; haipzhu@hust.edu.cn; shaoxy@hust.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2021.3124838.

Digital Object Identifier 10.1109/TCYB.2021.3124838

tree (RDT) models, the generalization of the proposed method and its prediction accuracy under different conditions might be improved.

The main contributions of this article are summarized as the following three points.

1) A novel DBRNN ensemble prognostic model is proposed, which contains two major parts. One is that three kinds of the DBRNNs with different structures are introduced to extract hidden features from monitoring data. Each DBRNN model collects information from both forward and backward data sequences for prediction, which increases data perception ability of the model. The other is that an ensemble model integrating multiple networks is built to improve the overall prediction accuracy and generalization performance.
2) An RDT-based multiple model ensemble algorithm is proposed to fuse prediction values of multiple DBRNNs and infer optimal RUL by the weighted summation of multiple RDT models trained iteratively. Meanwhile, a new customized loss function is designed to evaluate the deviation degree of the DBRNN models. It punishes the later prediction more than the early prediction so as to reduce the accident risk caused by the prediction errors.
3) Key factors, including the number of hidden neurons, the maximum depth, and the number of iterations on RDT, are studied thoroughly. The experimental results demonstrate the superiority of the proposed method to realize accurate RUL prediction of AEs under various operating conditions and failure modes, and the power of combining the EL and deep learning for the RUL prediction of complex dynamical systems.

The remainder of this article is organized as follows. Section II reviews the concept and literatures related to RUL prediction, RNNs, and EL. Section III describes the proposed DBRNN ensemble method in detail. Section IV presents the general process of DBRNN ensemble-based RUL prediction. Section V evaluates the performance of the proposed method. Finally, Section VI draws the conclusion and introduces the future research.

II. Related Works

A. Remaining Useful Life Prediction

RUL prediction is a critical part of PHM [11], [12], which deduces the RUL of equipment in the operating environment by analyzing the health status and degradation trend of the equipment. The entire life cycle process of the equipment from health to failure generally contains three states: 1) normal operation state; 2) degradation state; and 3) failure state. It is noted that the fault of the equipment does not occur instantaneously, but gradually aggravates over time. The performance of the equipment is degraded by the factors, such as the working environment and fatigue aging of the equipment itself, which ultimately leads to the failure of the equipment [13], [14]. At the same time, the RUL of the equipment can be predicted by the degree of performance degradation of the equipment.

According to the literature [15], the RUL prediction methods are classified into three types, that is: 1) model-based method; 2) data-driven method; and 3) hybrid method. Among them, the data-driven prognostic method does not need priori knowledge of the equipment failure mechanism. It only analyzes the health features hidden in the monitoring data reflecting the changes in the health state over time and completes the RUL prediction. Thus, data-driven prognostic methods have attracted wide attention.

Many data-driven RUL prognostic methods have been reported to achieve good results for different equipment. Ahmad *et al.* [16] proposed a hybrid forecasting technology for rolling bearings based on regression adaptive predictive models to learn the evolving trend of the health indicators and used these models to predict the RUL. Li *et al.* [17] constructed a deep convolution neural network (DCNN) model for RUL prediction where the raw sensor sequence data are standardized as the input and the RUL value of the label are fitted. In general, monitoring data used in these methods are typical time series that collect and record health status of the equipment in time sequence. To some extent, it is hard for the methods mentioned above to correlate data at different times. Luckily, it is found that the RNNs have a unique structure to memorize data and correlate data at different times.

B. Recurrent Neural Networks

RNN is a kind of neural network that includes feedforward connection and internal feedback connection [18], [19]. Because of its special network structure, RNN can extract useful information from the previously processed data across time steps and fuse them into the current process. RNN shows a strong advantage in dealing with time series. Many literatures show that RNN-based RUL prediction method has been widely used [20]. Heimes [21] used the RNN structure to realize the RUL prediction. The model was trained by an extended Kalman filter method. The experiments show that the performance of the proposed method is significantly better than that of the multilayer perceptron (MLP)-based method. However, traditional RNN (TRNN) usually faces the problem of memory loss. This is because it has no structure to control memory flow in the circulation layer. There will be a large prediction bias when dealing with long time monitoring sequence.

LSTM [22] and GRU [23] are the improvements of TRNN. These two kinds of neural networks construct unique "gate" structures, which determine the information characteristics passed under the optimal conditions. Zhang *et al.* [24] proposed a lithium battery RUL prediction model based on LSTM, which solved the problem of long-term dependence of degradation data.

The RNN-based RUL prediction method not only improves the accuracy of RUL prediction but also has the characteristics of fast convergence rate and high stability. It plays an important role in the field of reliability evaluation and RUL prediction [25], [26]. Although researchers have been working to improve the structure of TRNN to avoid the problem of gradient disappearance or explosion, it is still necessary

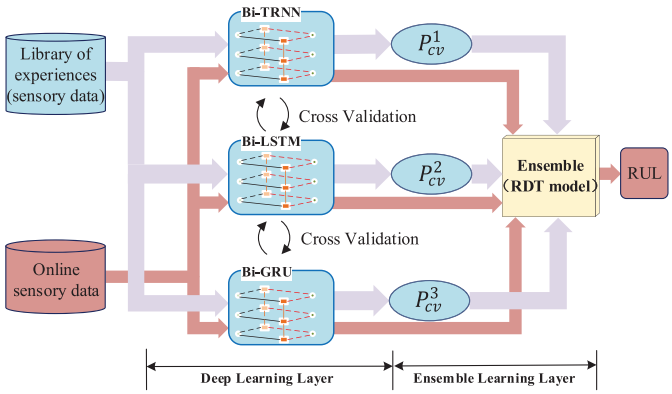

Fig. 1. Architecture of the DBRNN ensemble method.

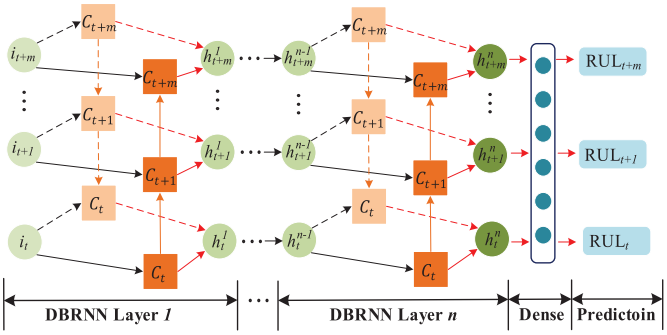

Fig. 2. Structure of DBRNN.

to improve its robustness and accuracy by methods such as EL.

C. Ensemble Learning

It is rarely possible that a single machine learning model consistently performs well in different applications [27]. An EL method that integrates multiple base learners often shows higher accuracy and better generalization [28]. There are two main steps contained in the EL method. One is to generate base learners, and the other is to design a combination strategy to integrate the results of these base learners. In order to ensure high accuracy and generalization of the EL method, base learners are usually required to have enough diversity and generate smaller errors. Besides, a good ensemble strategy has the ability of integrating the results of different base learners so as to obtain better prediction performance than any ensemble member.

The application of EL in RUL prediction has become a research hotspot recently [29]. Rigamonti *et al.* [30] constructed an ensemble echo state networks model for enhancing the performances of the individual networks and providing an estimation of the uncertainty affecting the RUL prediction. Kordestani *et al.* [31] developed an integrated method using an artificial neural network and discrete wavelet transform is via ordered weighted averaging operator to diagnose faults of aircrafts' multifunctional spoiler.

III. Deep Bidirectional Recurrent Neural Networks Ensemble

Some EL methods, such as random forest and adaboost, have the inevitable drawbacks. For example, the random forest tends to cause the overfitting problem in the classification or regression task affected by complex noise [32], and the adaboost is only applied to the classification task [33]. Therefore, the two methods are not suitable for RUL prediction of AE. To solve this problem, a DBRNN ensemble method is proposed, which is revealed in Fig. 1.

As shown in Fig. 1, the proposed DBRNN ensemble method consists of two layers, that is: 1) deep learning layer and 2) EL layer. In the deep learning layer, three kinds of the DBRNNs with different neuron structures are built and their network parameters are determined by backpropagation through the time algorithm and cross-validation method. In the EL layer, the predicted results obtained by the three DBRNN models are reencapsulated. Multiple RDT models are built to relearn the results iteratively. According to the accuracy of the RDT models, the relearned results are weighted and summed to realize the final RUL prediction.

A. Deep Learning Layer

In the deep learning layer, individual ensemble members are constructed as base learners to realize preliminary RUL predictions of AE. The choice of ensemble members is an important issue. Most intelligent models only establish the mapping relationship between the inputs and RUL at a certain time, which ignores the temporal dependency in the monitoring signals. However, the RUL prediction is a typical time-series problem, which means the predicted output is not only related to the current input but also affected by various factors throughout the entire life range. To capture the temporal dependency in the monitoring signals and increase the diversity of ensemble members, three different kinds of DBRNN models are constructed as individual ensemble members. In the proposed method, three specific DRNN structures, including TRNN, LSTM, and GRU, are adopted. Herein, the TRNN mainly focuses on extracting features in adjacent or close-to-distance data. The LSTM is specially designed to deal with long-term dependency, which extracts features from long-span sequence data. GRU can obtain valuable hidden data features by the unique "update gate" structure, which makes the feature extracted by GRU different from that of LSTM. Based on their different characteristics, these three structures are chosen to learn and mine data features from different aspects.

Fig. 2 shows the structure of the DBRNN models. In this structure, the inputs (i.e., $\boldsymbol{i}_t = [i_{t_1}, i_{t_2}, \ldots, i_{t_m}]$) are fed into the RNN cell (i.e., $\boldsymbol{C}_t$) according to the succession sequence, and the hidden state of the current layer is obtained and prepared to be fed into the next BRNN layer. After being processed by the dense layer, the RUL prediction of the base learner is finally obtained. These DBRNNs can continuously iterate and update the hidden state of cells in the time dimension to achieve the flow and in-depth utilization of information in a large time span. The cell structures of different RNNs are shown in Fig. 3.

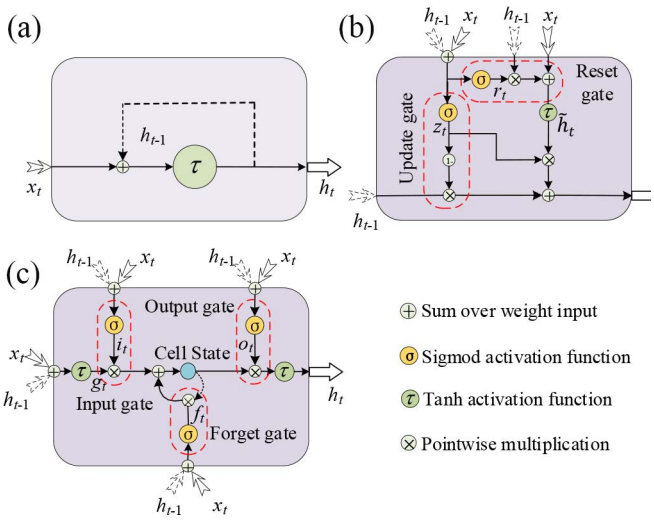

Fig. 3. Cell frame of the DBRNNs. (a) TRNN cell. (b) GRU cell. (c) LSTM cell.

As mentioned above, three types of neurons are adopted to ensure the diversity of the models and make it possible for the models to capture and mine data features from different aspects. The information flow in the TRNN neuron, GRU neuron, and LSTM neuron is shown in Fig. 3(a)–(c), respectively. As shown in Fig. 3(a), the current input is concentrated with the previous hidden state h_{t-1}^{TRNN}, and then fed to the activation function operation to generate a new hidden state h_t^{TRNN}, which means that it could extract information from the input of the previous cell state. The computational process in the TRNN neuron at t time is, respectively, described as

$$\begin{aligned}\boldsymbol{h}_t^{\mathrm{TRNN}} &= \mathbb{H}^{(R)}\left(\boldsymbol{h}_{t-1}^{\mathrm{TRNN}}, \boldsymbol{x}_t, \boldsymbol{\theta}_{\mathrm{TRNN}}\right)\\ &= \tau\left(\boldsymbol{W}_t \cdot \left[\boldsymbol{h}_{t-1}^{\mathrm{TRNN}}, \boldsymbol{x}_t\right] + \boldsymbol{b}_t\right)\end{aligned} \tag{1}$$

where $\mathbb{H}^{(R)}$ defines a nonlinear transformation function in TRNN neuron. θ_{TRNN} is a parameter set of TRNN neurons, respectively. x_t denotes the current input. $\boldsymbol{W}_t$ are weights of the current input x_t and the recurrent input h_{t-1}, and b_t are bias weights of the corresponding node. $\tau(\cdot)$ denotes the tanh function.

Fig. 3(b) shows the structure of the GRU neurons, in which the reset gate determines the output proportion of the new cell state in the previous time, and the update gate determines the output proportion of the new cell state in the current time.

It is seen from Fig. 3(c) that each LSTM neuron has three well-designed structures, including: 1) input gate; 2) forget gate; and 3) output gate. The input gate determines what cell state needs to be updated. The forget gate decides what information will be thrown away from the previous cell state. The output gate resolves which part of the cell state will be exported out of the cells. The computational process in the GRU neuron and the LSTM neuron at t time is, respectively, described as

$$\begin{aligned}\boldsymbol{h}_t^{\mathrm{GRU}} &= \mathbb{H}^{(G)}\left(\boldsymbol{h}_{t-1}^{\mathrm{GRU}}, \boldsymbol{x}_t, \boldsymbol{\theta}_{\mathrm{GRU}}\right)\\ &= \begin{cases} \boldsymbol{z}_t = \sigma\left(\boldsymbol{W}_z \cdot \left[\boldsymbol{h}_{t-1}^{\mathrm{GRU}}, \boldsymbol{x}_t\right] + \boldsymbol{b}_z\right)\\ \mathbf{r}_t = \sigma\left(\boldsymbol{W}_r \cdot \left[\boldsymbol{h}_{t-1}^{\mathrm{GRU}}, \boldsymbol{x}_t\right] + \boldsymbol{b}_r\right)\\ \widetilde{\boldsymbol{h}}_t = \tau\left(\boldsymbol{W}_h \cdot \left[\mathbf{r}_t * \boldsymbol{h}_{t-1}^{\mathrm{GRU}}, \boldsymbol{x}_t\right] + \boldsymbol{b}_h\right)\\ \boldsymbol{h}_t^{\mathrm{GRU}} = (1-\boldsymbol{z}_t) * \boldsymbol{h}_{t-1}^{\mathrm{GRU}} + \boldsymbol{z}_t * \widetilde{\boldsymbol{h}}_t \end{cases}\end{aligned} \tag{2}$$

$$\begin{aligned}\boldsymbol{h}_t^{\mathrm{LSTM}} &= \mathbb{H}^{(L)}\left(\boldsymbol{h}_{t-1}^{\mathrm{LSTM}}, x_t, \theta_{\mathrm{LSTM}}\right)\\ &= \begin{cases} \boldsymbol{g}_t = \tau\left(\boldsymbol{W}_g \cdot \left[\boldsymbol{h}_{t-1}^{\mathrm{LSTM}}, \boldsymbol{x}_t\right] + \boldsymbol{b}_g\right)\\ \boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_i \cdot \left[\boldsymbol{h}_{t-1}^{\mathrm{LSTM}}, \boldsymbol{x}_t\right] + \boldsymbol{b}_i\right)\\ \boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_f \cdot \left[\boldsymbol{h}_{t-1}^{\mathrm{LSTM}}, \boldsymbol{x}_t\right] + \boldsymbol{b}_f\right)\\ \boldsymbol{C}_t = \boldsymbol{C}_{t-1} * \boldsymbol{f}_t + \boldsymbol{g}_t * \boldsymbol{i}_t\\ \boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_o \cdot \left[\boldsymbol{h}_{t-1}^{\mathrm{LSTM}}, \boldsymbol{x}_t\right] + \boldsymbol{b}_o\right)\\ \boldsymbol{h}_t^{\mathrm{LSTM}} = \boldsymbol{o}_t * \tau(\boldsymbol{C}_t) \end{cases}\end{aligned} \tag{3}$$

where $\mathbb{H}^{(G)}$ and $\mathbb{H}^{(G)}$ are nonlinear transformation functions in GRU neuron and LSTM neuron, respectively. $\boldsymbol{\theta}_{\mathrm{GRU}}$ and $\boldsymbol{\theta}_{\mathrm{LSTM}}$ are parameter sets of three kinds of the neurons, respectively. $\boldsymbol{x}_t$ denotes the current input. $\boldsymbol{h}_{t-1}^{\mathrm{GRU}}$ and $\boldsymbol{h}_{t-1}^{\mathrm{LSTM}}$ are the outputs of TRNN neurons, GRU neurons, and LSTM neurons at $t-1$ time, respectively. $\boldsymbol{W}_z$, $\boldsymbol{W}_r$, $\boldsymbol{W}_h$, $\boldsymbol{W}_g$, $\boldsymbol{W}_i$, $\boldsymbol{W}_f$, and $\boldsymbol{W}_o$ are weights of the current input x_t and the recurrent input $\boldsymbol{h}_{t-1}$. $\boldsymbol{b}_z$, $\boldsymbol{b}_r$, $\boldsymbol{b}_h$, $\boldsymbol{b}_g$, $\boldsymbol{b}_i$, $\boldsymbol{b}_f$, and $\boldsymbol{b}_o$ are bias weights of the corresponding node. σ represents the sigmoid function, and $*$ denotes the pointwise multiplication. $\boldsymbol{z}_t$ and $\boldsymbol{r}_t$ are the output of the update gate and reset gate of the GRU cell unit. For the LSTM cell unit, $\boldsymbol{g}_t$, $\boldsymbol{i}_t$, $\boldsymbol{f}_t$, and $\boldsymbol{o}_t$ are the input node, input gate, output of the forget gate, and output gate, respectively. $\boldsymbol{C}_t$ and $\boldsymbol{C}_{t-1}$ are the cell state values at time t and $t-1$, respectively.

During the training process of the DBRNN models, the output is the connection vectors of the forward and backward processes. The output $h_t^{(*)}$ is computed as

$$\boldsymbol{h}_t^{(*)} = \overrightarrow{\boldsymbol{h}_t^{(*)}} \oplus \overleftarrow{\boldsymbol{h}_t^{(*)}} \tag{4}$$

where $\rightharpoonup$ and $\leftharpoonup$ denote the forward and backward processes, respectively, and * can be replaced by TRNN, GRU, and LSTM. The operator $\oplus$ represents the elementwise sum of the outputs of the forward and backward processes. The corresponding hidden vectors are updated as

$$\overrightarrow{\boldsymbol{h}_t^{(*)}} = \mathbb{H}^{(*)}\left(\boldsymbol{x}_t^{(*)}, \boldsymbol{h}_{t-1}^{(*)}, \boldsymbol{\theta}_{(*)}\right) \tag{5}$$

$$\overleftarrow{\boldsymbol{h}_t^{(*)}} = \mathbb{H}^{(*)}\left(\boldsymbol{x}_t^{(*)}, \boldsymbol{h}_{t+1}^{(*)}, \boldsymbol{\theta}_{(*)}\right) \tag{6}$$

where $\mathbb{H}^{(*)}$ is the transition functions in hidden units of three types of the neurons, which is defined by formulas (1)–(3). The proposed DBRNN model consists of the input layer, hidden cell layer, and output layer. The input layer receives the AE sensor signals. After deep mining of multihidden cell layers, the input data are finally processed through a full connection layer to obtain the output results. Under the action of the activation function, the predicted RUL is obtained. This process is defined as

$$o_i = \varphi\left(\boldsymbol{W}_i \boldsymbol{h}_t^{(*)} + \boldsymbol{b}_i\right) \tag{7}$$

where o_i denotes the final output RUL value and φ represents the activation function.

A variety of DBRNN models with different cell structures is utilized to improve the diversity of ensemble members and enhance the generalization of the ensemble model. Meanwhile, the cross-validation technique is adopted to avoid the overfitting of these individual ensemble members [34]. Fig. 4 shows the process of three-fold cross-validation. The sensory dataset of AE is divided into two parts, that is: 1) a subtraining

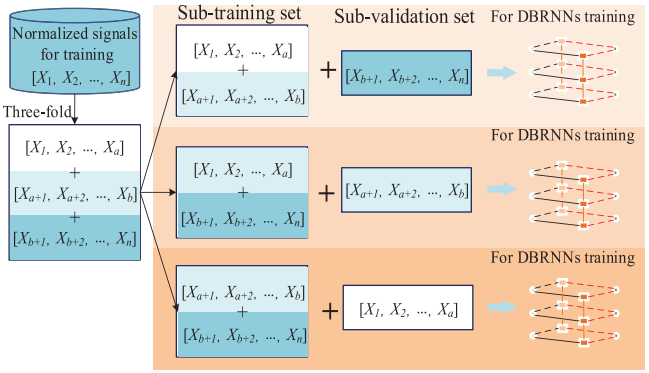

Fig. 4. Three-folds cross validation process.

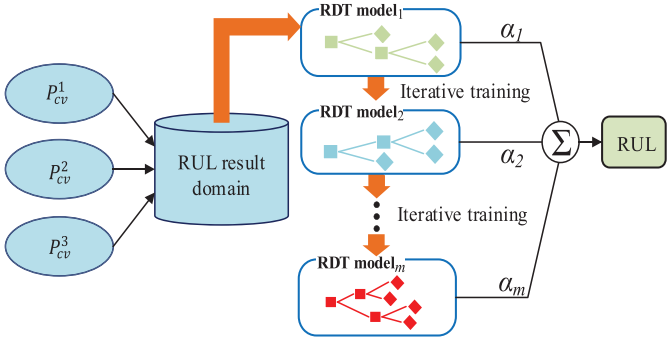

Fig. 5. Specific steps for RDT-based EL.

set and 2) a subvalidation set. $[\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_n]$ represent the sensory data of n AEs where $\boldsymbol{X}_i = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t]$ are the signals sequences, and x_t denotes the multisensory data vector at cycle t. The DBRNN models will undergo three training sessions, in which two-thirds of the AE data are selected for training models and the rest for model validation without repeating in each training session. Taking the first training sessions as an example, the first and second fold of the sensory data, that is, $[\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_a]$ and $[\boldsymbol{X}_{a+1}, \boldsymbol{X}_{a+2}, \ldots, \boldsymbol{X}_b]$, are fed into the DBRNN models for training, and the third fold of the sensory data $[\boldsymbol{X}_{b+1}, \boldsymbol{X}_{b+2}, \ldots, \boldsymbol{X}_n]$) is used for validation, where $a = 1/3n$ and $b = 2/3n$.

The neural network updates the parameters by seeking the partial derivative of the network parameters. The optimization algorithm of the parameters will directly affect the training efficiency and convergence of the DBRNN model. For the traditional optimization algorithm such as stochastic gradient descent (SGD), the learning rate (LR) is fixed during the network training process, which means the LR value is particularly critical. A small LR will lead to slow convergence while a large LR will cause oscillations and fail to converge to the optimal solution [35]. To avoid this problem, a new algorithm called adaptive moment estimation (Adam) is used to optimize the parameters in the deep learning layer of the DBRNN. The Adam algorithm designs an independent adaptive LR for different parameters by analyzing the first moment estimation (FME) and second moment estimation (SME) of the gradient. Therefore, by constraining the global LR, the Adam algorithm has good stability, consumes less configuration resources while ensuring convergence, and does not need to store all global gradients, which is very suitable for processing large-scale data. The updating process of network parameters by the Adam algorithm is expressed as

$$\boldsymbol{g}_t = \nabla_\theta \sum_i L(\boldsymbol{\theta}_{t-1}) \tag{8}$$

$$\boldsymbol{m}_t = \boldsymbol{\lambda}_1 \boldsymbol{m}_{t-1} + (\mathbf{1} - \boldsymbol{\lambda}_1)\boldsymbol{g}_t \tag{9}$$

$$\boldsymbol{n}_t = \boldsymbol{\lambda}_2 \boldsymbol{n}_{t-1} + (\mathbf{1} - \boldsymbol{\lambda}_2)\boldsymbol{g}_t^2 \tag{10}$$

$$\overline{\boldsymbol{m}}_t = \frac{\boldsymbol{m}_t}{\mathbf{1} - \boldsymbol{\lambda}_1^t} \tag{11}$$

$$\overline{\boldsymbol{n}}_t = \frac{\boldsymbol{n}_t}{\mathbf{1} - \boldsymbol{\lambda}_2^t} \tag{12}$$

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \boldsymbol{\omega} \frac{\overline{\boldsymbol{m}}_t}{\sqrt{\overline{\boldsymbol{n}}_t} + \epsilon} \boldsymbol{g}_t \tag{13}$$

where $\boldsymbol{g}_t$ indicates the gradient of the loss function $L(\boldsymbol{\theta})$ to the network parameter set $\boldsymbol{\theta}$. $\boldsymbol{m}_t$ and $\boldsymbol{n}_t$ represent the FME and SME of gradient, respectively. $\overline{\boldsymbol{m}}_t$ and $\overline{\boldsymbol{n}}_t$ are deviation corrections for $\boldsymbol{m}_t$ and $\boldsymbol{n}_t$, respectively. $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$ represent decay rates of FME and SME, respectively. ω denotes the step size and ϵ is the numerical stability constant. $\boldsymbol{\theta}_{t-1}$ is the calculated renewal value of $\boldsymbol{\theta}_t$ at $t-1$ time.

B. Ensemble Learning Layer

In the EL layer, a novel RDT-based method is proposed to integrate the predicted RUL results obtained by the DBRNN models so as to mine the potential characteristics of monitoring signals more comprehensively and achieve a more accurate RUL prediction. The RDT model proposed by Breiman *et al.* [36] has been widely used in the field of statistics and data mining. It uses a completely different way from traditional statistics to build prediction criteria. It is given in the form of the binary tree, which is easy to understand, use, and explain. In many cases, the RDT model is more accurate than the algebra prediction criterion constructed by common statistical methods [37].

By iteratively training multiple RDT models, the predicted RUL results are relearned, and the weights are allocated according to the training errors of the models. After summing up the weighted RUL results, the final RUL prediction is completed.

Fig. 5 indicates the process of the EL, which includes three steps. In the first step, the predicted results of the three DBRNN models are reencapsulated into a new RUL domain $\boldsymbol{X}_o$. In the second step, multiple RDT models are set up and iteratively trained by using the new RUL domain $\boldsymbol{X}_o$. The weight of each model will be updated during the training process. In the third step, the predicted results of each RDT model are integrated with the corresponding weights of the model to realize the final RUL prediction.

Each RDT model divides the input domain into several units, each of which has a corresponding output. The goal of training the RDT model is to find an accurate space partition and obtain the accurate mapping relationship between the partition unit and the output results. When the input characteristics are classified into a certain unit, the corresponding output value will be obtained. The computational procedure is

described as

$$\mathbb{T}_m(\boldsymbol{x}_i) = \begin{cases} (j,s) = \min_{j,s}\left[\min_{c_1} \text{Loss}(y_i, c_1) \right. \\ \qquad \left. + \min_{c_2} \text{Loss}(y_i, c_2)\right] \\ \boldsymbol{R}_k(j,s) = \left\{\boldsymbol{x} \mid x^j \le s\right\} \\ \boldsymbol{R}_{k+1}(j,s) = \left\{\boldsymbol{x} \mid x^j \ge s\right\} \\ \widehat{\mathrm{c}}_k = \frac{1}{N_k}\sum_{x_i \in R_k} y_i, \ \ x_i \in \boldsymbol{R}_k \\ f(x_i) = \sum_{k=1}^{K} \widehat{\mathrm{c}}_m I(x_i \in \boldsymbol{R}_k) \end{cases} \tag{14}$$

where $\mathbb{T}_m$ is the mth RDT model of iterative training. j and s refer to the splitting variable and splitting point, respectively. R_k is the partition of domain, $\widehat{\mathrm{c}}_k$ denotes the output value of the kth partition of domain R_k, and $\widehat{\mathrm{c}}_m$ denotes the average of all the y_i values in the partition R_k. $f(\cdot)$ represents the final output, which is a piecewise function.

In the iterative training process, all training samples are given the same initial weight $\boldsymbol{W}_1 = \left(w_1^1, w_1^2, \ldots, w_1^n\right)$, $w_1^i = (1/n), i = 1, 2, \ldots, n$. Then, the first RDT model $\mathbb{T}_1$ is trained. The error rate α_m of the decision tree model is calculated, and the weight coefficient of $\mathbb{T}_1$ is calculated by the error rate, and the weight distribution of the training samples $\boldsymbol{W}_2$ for the next iterative training is updated. Then, a new round of training is started. The training process stops after reaching the maximum number of iterations M, and finally, obtain m RDT models $\{\mathbb{T}_1, \mathbb{T}_2, \ldots, \mathbb{T}_m\}$ and their corresponding weight coefficients $\{\alpha_1, \alpha_2, \ldots, \alpha_m\}$. Finally, the linear combination of the prediction results of these RDT models is linearly combined to obtain the final RUL prediction. The description of the RDT-based DBRNNs EL is shown in Algorithm 1.

Algorithm 1 RDT-Based DBRNNs EL

Input:
Training dataset $\mathcal{T} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, while $x_i \in X_0$, $y_i \in RUL$, $i = 1, 2, \ldots, n$
M: Maximum number of iterative training
Output:
Trained *RDT*s = $\{\mathbb{T}_1, \mathbb{T}_2, \ldots, \mathbb{T}_m\}$
Step 1) Initialization:
Generate a weight distribution of initial training data:

$$\boldsymbol{W}_1 = \left(w_1^1, w_1^2, \ldots, w_1^n\right), \ w_1^i = \frac{1}{n}, i = 1, 2, \ldots, n$$

Step 2) Iterative training:
for $m = 1, 2, \ldots, M$ do
Step 2.1) Train an *RDT* model $\mathbb{T}_m$ using a dataset with the weight distribution W_1.
Step 2.2) Calculate the maximum error of the model $\mathbb{T}_m$

$$E_m = |y_i - \mathbb{T}_m(x_i)|$$

Step 2.3) Calculate the relative error of each RUL training sample:

$$e_m^i = 1 - exp(\frac{-y_i + \mathbb{T}_m(x_i)}{E_m})$$

Step 2.4) Calculate the error rate:

$$e_m = \sum_{i=1}^{n} e_m^i w_m^i$$

Step 2.5) Calculate the weight coefficient of the model $\mathbb{T}_m$

$$\alpha_m = \frac{e_m}{1 - e_m}$$

Step 2.6) Update the weight distribution of the training data:

$$w_{m+1}^i = \frac{w_m^i}{\sum_{i=1}^{n} w_m^i \alpha_m^{1-e_m^i}} \alpha_m^{1-e_m^i}$$

end for
Step 3) Integration:
Construct a linear combination of the trained *RDT*s according to the calculated weights:

$$RUL(x) = \sum_{m=1}^{M} (ln\frac{1}{\alpha_m})\alpha_m \mathbb{T}_m(x)$$

IV. General Procedure of the Proposed Method

The procedure of the DBRNN ensemble-based RUL prediction consists of two stages, that is: 1) offline modeling and 2) online prediction, which are detailed in Table I. In the procedure, data preprocessing, parameter optimization, and loss function are indispensable.

A. Data Preprocessing

In general, the monitoring data collected by sensors often have different scales and orders of magnitude. A normalization method is adopted to improve the convergence speed and fitting accuracy of the models, which is expressed as

$$\text{Norm}\left(x^{(s,c)}\right) = \frac{x - x_{\min}^{(s,c)}}{x_{\max}^{(s,c)} - x_{\min}^{(s,c)}} \tag{15}$$

where s denotes each kind of sensors and c denotes different operating conditions, and $x_{\min}^{(s,c)}$ and $x_{\max}^{(s,c)}$ are the minimum and maximum values of the sensor signals corresponding to the operation condition c and sensor s.

Since the background noise is unavoidable in data acquisition, the Savitzky–Golay convolution smoothing algorithm is used to reduce the background noise, which is given as

$$\overline{x_k} = \frac{1}{H}\sum_{i=-w}^{+w} x_{k+i} h_i \tag{16}$$

where $\overline{x_k}$ denotes a smooth value at the kth point, and h_i is a smoothing coefficient. H is a smooth window, and $[-w, +w]$ represents the width of windows. The least squares method is used to calculate (h_i/H) by polynomial fitting.

In addition, correlation represents the degree of association between variables. Monotonicity represents the trend of the data degradation over time [38]. In this article, monotonic and correlational analyses are implemented to find the degradation trend and correlation hidden in sensor data, which are expressed as

$$r = \frac{\left|\left(n\sum_n f(t)t - \sum_n f(t)\sum_n t\right)\right|}{\sqrt{\left[n\sum_n f(t)^2 - \left(\sum_n f(t)\right)^2\right]\left[n\sum_n t^2 - \left(\sum_n t\right)^2\right]}} \tag{17}$$

$$m = \frac{\left|\sum_n \sigma(f(t+1) - f(t)) - \sum_n \sigma(f(t) - f(t+1))\right|}{n-1} \tag{18}$$

where r and m represent correlation and monotonicity, respectively. $\sigma(\cdot)$ represents the sign function. n indicates the quantity of the signals.

TABLE I
DESCRIPTION OF THE PROPOSED METHOD

Offline modelling stage	
Step 1:	Sensor data of the AE in the whole life cycle are collected, selected and preprocessed. These data and their corresponding RUL labels constitute the training set for the next training process, that is, $T = \{(x_1, RUL_1), (x_2, RUL_2), \ldots, (x_n, RUL_n)\}$.
Step 2:	The deep learning layer and ensemble learning layer in the DBRNN ensemble method are constructed and relevant parameters are optimized.
Step 3:	The training datasets after three-folds cross-validation are used to train the DBRNN models with back propagation through time algorithm and a unique loss function is designed to evaluate the training error.
Step 4:	The validation datasets are inputted into the trained DBRNN models to obtain the corresponding RUL values.
Step 5:	The predicted results of the DBRNN models are re-encapsulated into the predicted RUL domain $\boldsymbol{X}_o$, which is used for iteratively training multiple RDT models.
Step 6:	Combine all the trained RDT models to establish an ensemble model. The weights of the RDTs are assigned according to the accuracy.
Online prediction stage	
Step 1:	Real-time monitoring data of the AE related sensors is collected. The raw signal is then normalized and smoothed according to formulas (15) and (16).
Step 2:	The preprocessed monitoring data is fed into the well trained DBRNN models to get the predicted results in deep learning layer.
Step 3:	The results are re-encapsulated and put into the ensemble layer to obtain the final RUL predictions by summing the results of the RDT models with assigned weights.

B. Parameters Optimization of DBRNN Ensemble

As is known, different parameter selection might lead to great changes in the structure and complexity of the model. The parameter optimization of the model may help select the model with appropriate complexity, effectively reduce the test error, and improve the generalization of the model.

In the proposed DBRNN ensemble method, multiple hyperparameters, including the number of hidden cell layers and hidden neurons in cell layers, need be adjusted, while the max depth and the iteration numbers of the RDT models in the EL layer also need be confirmed.

In this article, a grid search algorithm is adopted for the optimum hyperparameters of the DBRNN ensemble method. The principle of the grid search is to divide the interval of each parameter value into a series of small intervals, and calculate the target value (usually error) determined by the combination of the corresponding parameter values in order, and select the best one by one so as to obtain the minimum target value and the corresponding optimal parameter value in the interval. For different datasets, the optimal training depth has a significant difference, which is mainly due to the difference in data volume and hidden features of different datasets. This method can ensure that the obtained search solution is globally optimal or near optimal, and avoid significant errors [39].

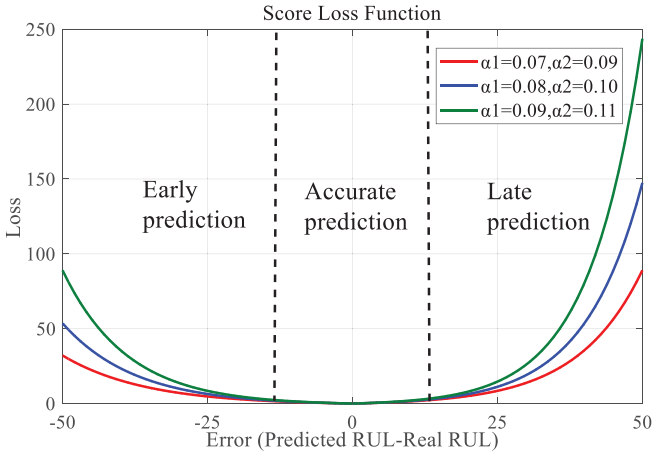

Fig. 6. SLF curve.

C. Loss Function for DBRNNs Training

The loss function evaluates the difference between the actual value and the output value of the model in the iterative training process [45]. The network is trained by minimizing the error between the output and the actual RUL in the training data. Conventional loss functions [such as the mean square error (MSE) and MAE functions] are usually symmetrical, which means that the evaluation values of early prediction and late prediction are the same.

The main goal of RUL prediction is to predict the failure before it occurs to avoid unnecessary downtime and additional maintenance costs. The symmetric loss function is not sensitive to the early and late predictions of the engine, which make it unsuitable for industrial application scenarios. In this article, a novel asymmetric loss function called the score loss function (SLF) is designed to train the DBRNN models, which is expressed as

$$\mathrm{SLF}(Y, f(X)) = \begin{cases} \exp(\alpha_1 |Y - f(X)|) - 1 & Y - f(X) < 0 \\ \exp(\alpha_2 |Y - f(X)|) - 1 & Y - f(X) \geq 0 \end{cases} \quad \alpha_1, \alpha_2 \in [0, 1] \tag{19}$$

where Y and $f(X)$ represent actual RUL label and the output RUL value, respectively. $|Y - f(X)|$ denotes the deviation between the predicted value and the actual value. α_1 and α_2 are penalty factors for advanced prediction, and $\alpha_1 < \alpha_2$.

Fig. 6 shows the SLF curve. It can be seen that the SLF will penalize the later prediction more than early prediction, for the later prediction is regarded as the unsafe prediction. In addition, as the factors α_1 and α_2 decrease, the penalty will increase, which is beneficial for early prediction rather than later prediction.

V. EXPERIMENTAL STUDY

A. Dataset Description

To validate the effectiveness of the proposed RUL prediction method, the dataset generated from the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) test bed is used [26]. The C-MAPSS dataset consists of four subdatasets. As shown in Table II, operating conditions and failure

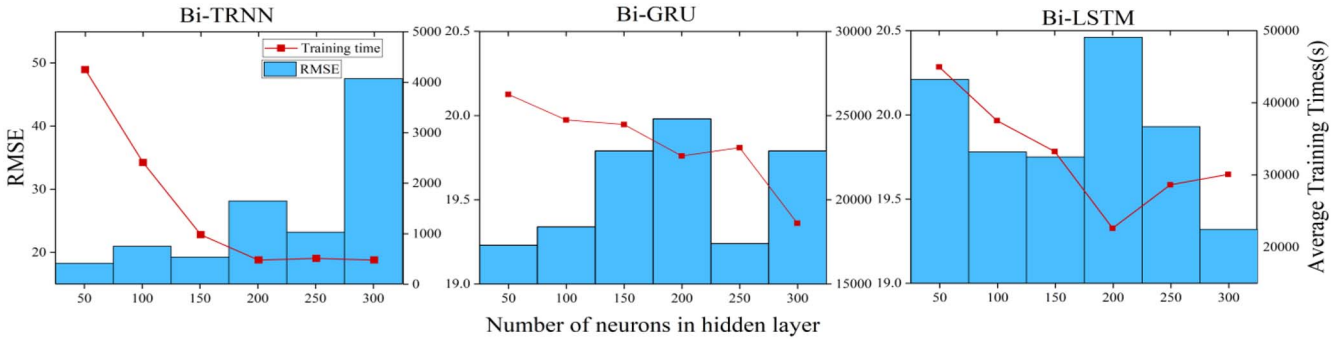

Fig. 7. Prognostic performance of different neuron numbers on FD001.

TABLE II
DATASET DETAILS

Dataset	Training engine	Testing engine	Operating condition	Failure mode
FD001	100	100	1	1
FD002	260	259	6	1
FD003	100	100	1	2
FD004	248	249	6	2

modes of four subdatasets are various. A certain number of training engines and testing engines are included in each subdataset.

Each subdataset consists of monitoring signals acquired from a series of the training engines and the testing engines. These signals are collected by 21 sensors under different operating conditions, including sea-level temperature, Mach number, and altitudes. Different operating conditions have a great impact on the monitoring signals of the sensors. Each engine runs in a healthy state at an early stage, and its performance decreases over time until the engine breaks down. Each engine corresponds to a series of data points sampled by 21 sensors in a life cycle. In the training process, all the training engines are used to train the DBRNN models by cross-validation. In the testing process, only the last set of the sensor signals of each engine in the testing set is selected for prediction.

Given that the engine works steadily at the early age and deteriorates linearly until it fails. Herein, we adopt the usual label processing method, that is, piecewise linear label, which assigns a constant value to the target label of the early monitoring signal (for the C-MAPSS dataset, take 125 as the constant RUL label) [41]–[43].

Two commonly used performance metrics, Score and root MSE (RMSE) [44], are used to evaluate the performance of the proposed method. Both of the performance metrics indicate the deviation degree between the predicted values and the actual values. Lower score and RMSE values mean that the prediction results are closer to the actual values. It also indicates that the prediction accuracy of the model is higher. The formulations of the two metrics are given as

$$\text{Score} = \sum_{i=1}^{N} \begin{cases} \exp\left(-\frac{d_i}{13}\right) - 1, & \text{if } d_i < 0 \\ \exp\left(\frac{d_i}{10}\right) - 1, & \text{if } d_i \geq 0 \end{cases} \tag{20}$$

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} d_i^2} \tag{21}$$

where d_i represents the deviation between the predicted RUL and real RUL of the ith testing engine.

In addition, all the works were accomplished with Anaconda and python 3.6 and executed on a computer with Intel Core i5-4460 (3.20 GHz) CPU, 16-GB memory, and Microsoft Windows 10 Enterprise operating system.

B. Parameters Optimization

Multiple hyperparameters of the proposed method need be adjusted, that is, the hidden cell layers and the number of hidden neurons in the deep learning layer, the max depth of RDT, and the number of RDT models for iterative training in the EL layer.

1) Parameter Optimization of Deep Learning Layer: The performance of the DBRNNs is affected by many factors, such as model structure and training methods. Hence, the grid search algorithm is utilized to explore the optimized parameters in the DBRNN layer.

Taking the optimization of the number of the neurons in the hidden cell layer as an example, comparative experiments are conducted by using three DBRNNs, respectively. For comparison, the penalty factors of the loss function SLF are set to $\alpha_1 = 0.08$ and $\alpha_2 = 0.1$. Fig. 7 shows the average RMSE and training time over ten runs obtained by the DBRNNs on FD001, as the number of hidden neurons increases from 50 to 300. It is found in Fig. 7 that RMSE has a significant negative correlation with the average training time. Lower RMSE usually means higher training accuracy, and higher training accuracy often requires more resource costs. For TRNN and GRU, it is seen in Fig. 7 that as the neurons increase, their RMSE increases and the training time decreases. It means that the overfitting might occur in the training process, especially in TRNN, and this leads to the decreased training time and the lower accuracy of the network models.

Table III shows the relationship between the number of average training epochs of the three DBRNNs and the hidden neurons. The results showed that the epochs decrease as the neurons increase, which means that an excessive number of neurons may cause the model more likely to fall into a local minimum. Besides, compared with the Bi-LSTM network and

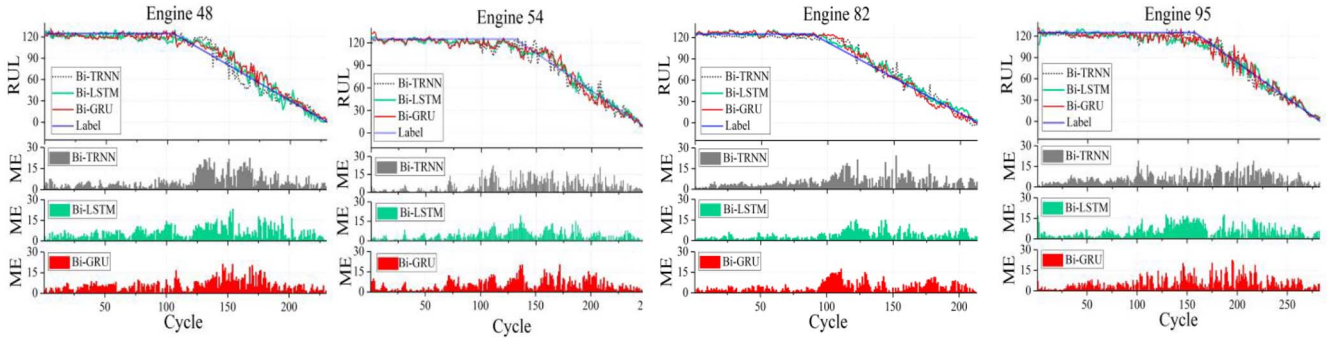

Fig. 8. Comparison of verification results of the three DBRNNs.

TABLE III
EPOCHS OF EARLY STOPPING FOR THREE DBRNNS

Number of neurons	Bi-TRNN	Bi-GRU	Bi-LSTM
50	386	292	346
100	219	275	289
150	89	272	256
200	44	251	174
250	46	257	220
300	44	206	231

TABLE IV
HYPERPARAMETERS OF TRAINING MODEL

Hyper-parameters	Bi-TRNN	Bi-GRU	Bi-LSTM
Neurons	150	250	300
Layers	3	3	3
Dropout	0.5	0.6	0.6
Batch size	8	8	8
Epoch	1000	1000	1000
Early-stop	15	15	15
Initial learning rate	0.001	0.001	0.001

Bi-GRU network, the training time of the Bi-TRNN is one order of magnitude less. This is because the simpler neuron structure of the TRNN will cause the lower computational costs of calculation. Finally, the number of neurons of the Bi-TRNN, Bi-GRU, and Bi-LSTM is set to 150, 250, and 300, respectively.

The results of the hyperparameter optimization are shown in Table IV, and each result is the average of ten repeated experiments. In Table IV, layers denote the number of hidden cell layers in the DBRNN models. Dropout presents the probability of dropping out units in the training process. The batch size is the number of samples selected for training, which has an effect on the training speed of the model. Epoch is the maximum number of training, and early stop denotes the number of iterative epochs in which the loss deviation has not been reduced. Initial LR controls the initial rate of learning progress since the Adam optimizer is adopted to train the DBRNN models.

It is worth noting that the BRNN model with more hidden layers might have a stronger information representation ability theoretically. On the other side, more layers in the BRNN model tend to cause the gradient vanishing in information transmission between different layers and a low convergence efficiency in the model training process. After many experiments, the hidden layer number of the BRNN is determined as 3.

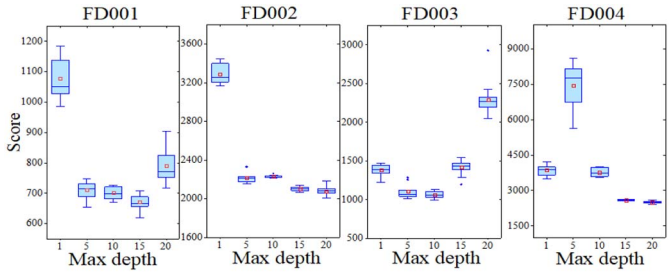

Fig. 9. Results of the RDT with different max depth.

Once the network training is completed, the performance of the model is tested by using the validation set. Fig. 8 shows the prediction results of four validation engines. It is found in Fig. 8 that all the three kinds of DBRNNs can fit the engine's RUL well. After hyperparameter optimization, the predicted mean error (ME) of RUL is within 30. The model also shows higher prediction accuracy for data points in the health state and near failure state.

2) Parameter Optimization of Ensemble Learning Layer: The proposed method uses multiple RDT models to ensemble the DBRNNs. The max depth and iteration number of the RDT model have a very important impact on the performance of the proposed method.

To determine the optimal max depth, the performance of the RDT models with different max depth parameters is compared by using the four subdatasets. Fig. 9 shows the distribution of the score over 15 runs, as the number of max depth increases from 50 to 300 at intervals of 5. The lower the score indicates the less deviation, which means the more accurate the predicted RUL value is. It is found from Fig. 9 that as for datasets FD001 and FD002, the RDT model achieves higher prediction accuracy when the max depth is between 5 and 15. But this value is raised to 20 when it comes to FD002 and FD004. This is because the operation modes of the engines contained in the two datasets are more complex and the RUL prediction domain is larger. So more spatial partitions are needed to improve accuracy.

Once the max depth of the RDT model is set, it is necessary to find the iteration number of the models. Fig. 10 shows the

TABLE V
PERFORMANCE COMPARISONS OF DIFFERENT MODELS ON THE DATASET FD001

Methods	SCORE	RMSE	Model training times (s)	Online average calculation times (s)
Bi-TRNN	692.78	19.09	1068	0.21
Bi-GRU	580.04	18.77	23387	0.36
Bi-LSTM	513.11	18.22	30261	0.28
DBRNNE with MSELF	498.23	18.09	153497	0.71
DBRNNE with SLF	459.89	17.97	164737	0.73

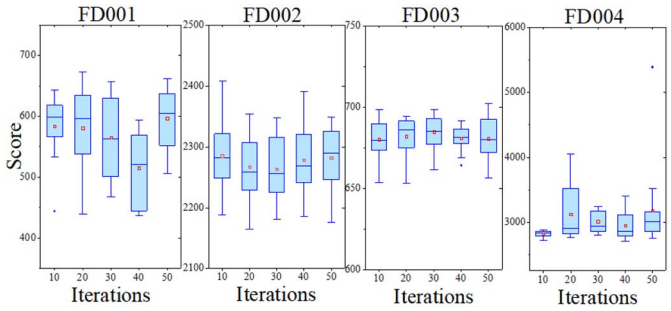

Fig. 10. Results of the RDT with different iteration number.

impact of the iterative number on the prediction accuracy of the four subdatasets. The parameters are changed from 10 to 50 at intervals of 10, and 15 runs are carried out for each group of parameters. The results show that for datasets FD001 and FD003, the predictions have better performance when the number of iterations is about 40. As for FD002, 30 iterations are regarded as the best, and ten iterations have better performance on FD004.

C. Results and Comparison

After parameter optimization, the structure of the models is determined. The steps in Section IV are performed in a sequence to train the constructed models. Then, the sensor signals of the testing engines are fed into the trained DBRNN ensemble model to realize the RUL prediction.

The predicted RUL of the engines in the four testing datasets is shown in Fig. 11, which are sorted by the real RUL from small to large. In this article, an RUL threshold is set, which has been widely used in literature [19], [41]–[43]. The results show that the predicted RUL values are close to the actual RUL values, and the prediction accuracy is even higher when the prediction point is closer to the end of life, which illustrates the remarkable performance of the proposed DBRNN ensemble model in RUL prediction. Compared with FD002 and FD004, the prediction deviation of FD001 and FD003 seems to be smaller, which indicates that the proposed method has higher prediction accuracy under a single operation mode. In addition, due to the introduction of the asymmetric loss function, the predicted RUL value tends to be higher than the actual value, and this is very promising for industrial applications.

Table V compares the prediction results of five models based on two evaluation metrics and time consumption. Among them, DBRNNE with MSELF represents that the model is trained with the traditional MSE loss function, and DBRNNE with SLF represents that the model is trained with the proposed SLF loss function. The experimental results show that the proposed DBRNNE model is superior to all other models in terms of score and RMSE. Since the network structure of Bi-TRNN is the simplest, the prediction results of Bi-TRNN are relatively worse compared with other advanced RNN structures. In addition, compared with DBRNNE with MSE, DBRNNE with SLF has about 8% improvement in the score metric, which means that the proposed SLF can improve the prediction accuracy of the DBRNNE model. In terms of time consumption analysis, since DBRNNE integrates multiple base learners, the model training time will increase significantly. It should be noted that the DBRNNE trained with SLF takes longer training time than that trained with MSELF. This is because the stage of the partial derivative of the SLF is more computationally expensive than MSELF. Considering that the training stage of the prediction model is generally performed offline, more time spent on training is acceptable. However, the online average calculation time of all models is less than 1 s on a normal personal computer, which proves that the proposed DBRNNE model can be applied to industrial applications.

The C-MAPSS dataset used in this article is very popular in the field of prognostic and already has many the latest research results. Many prognostic methods are adopted to compare with the proposed methods, including Earlier CNN [45], multiobjective deep belief networks ensemble (MODBNE) [10], multiobjective MLP ensemble (MOMLPE) [10], multiobjective SVM ensemble (MOSVME) [10], DCNN [19], bidirectional handshaking LSTM (BHLSTM) [40], Markov [46], deep LSTM (DLSTM) [28], Bayes [47], ensemble LSTM (ELSTM) [48], Bi-TRNN, Bi-LSTM, and Bi-GRU.

Table VI shows the comparison results with other published literature on the C-MAPSS dataset, where N/A means the information is unavailable. It is found in Table VI that the proposed method has the lowest score and RMSE on FD002 and FD004, and also has competitive and promising results on FD001 and FD003. The detailed discussions about the result comparison are summarized as follows.

1) Many machine learning algorithms (such as Markov and Bayes) have shown their advantages in this prediction problem. Compared with these traditional machine learning methods, the proposed method shows incomparable advantages.
2) Many methods have very good prognostic results on FD001 and FD003, however, poor performance for FD002 and FD004. This is because FD001 and FD003 have only one operating condition while FD002

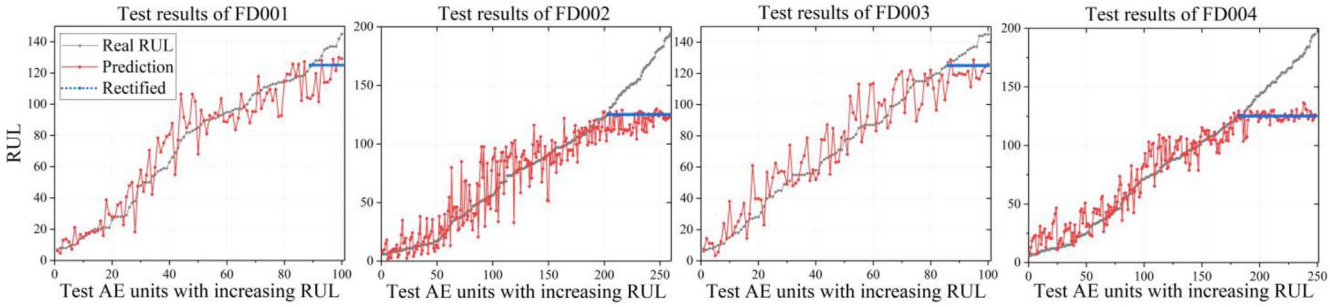

Fig. 11. Sorted RUL prediction of the engines in four testing datasets.

TABLE VI
PERFORMANCE COMPARISONS OF THE PROPOSED METHOD AND THE LATEST RELATED PAPERS ON THE C-MAPSS DATASET

Methods	Year	FD001		FD002		FD003		FD004	
		SCORE	RMSE	SCORE	RMSE	SCORE	RMSE	SCORE	RMSE
Earlier CNN [45]	2016	1286	18.45	13570	30.29	1596	19.82	7890	29.17
MODBNE [10]	2017	334.23	15.04	5585.34	25.02	421.91	12.51	6557.62	28.66
MOMLPE [10]	2017	560.59	16.78	14026.72	28.78	479.85	18.47	10444.35	30.96
MOSVME [10]	2017	7703.33	40.72	316483.31	52.99	22541.58	46.32	141122.19	59.96
DCNN [19]	2018	**273.7**	**12.61**	10412	22.36	**284.1**	**12.64**	12466	23.31
BHLSTM [40]	2019	376.64	N/A	N/A	N/A	1422	N/A	N/A	N/A
Markov [46]	2019	N/A	N/A	N/A	N/A	N/A	N/A	3572	26.85
DLSTM [28]	2020	655	18.33	N/A	N/A	828	19.78	N/A	N/A
Bayes [47]	2020	N/A	N/A	N/A	25.9	N/A	N/A	N/A	N/A
ELSTM [48]	2021	571	18.22	N/A	N/A	839	23.21	N/A	N/A
Bi-TRNN	2021	692.78	19.09	2936.91	21.56	823.49	20.37	3636.00	25.92
Bi-LSTM	2021	580.04	18.77	2435.99	20.93	769.36	19.48	3164.66	20.50
Bi-GRU	2021	513.11	18.22	2322.89	20.54	778.72	19.94	3282.47	23.97
Proposed method	2021	459.89	17.97	**2064.39**	**19.81**	658.12	19.18	**2869.67**	**20.40**

and FD004 have six. More operating conditions usually mean more complex data forms, which puts forward higher requirements for feature extraction and analysis ability of the model. The best performance on FD002 and FD004 among all the state of the arts demonstrates the superiority of the proposed DBRNN ensemble prognostic model.

3) Since FD001 and FD003 are relatively simple, many methods, such as DCNN, MODBNE, and MOMLPE, focus on optimizing the results on these two subdatasets and ignore the further optimize for the others, resulting in excellent results on FD001 and FD003, but relatively poor results on FD002 and FD004. Different from other deep learning methods, the proposed method considers the prediction requirements of four subdatasets at the same time, where network structures are shared on all the subdatasets. This makes the proposed method has the best results on FD002 and FD004 and competitive results on FD001 and FD003 compared with earlier CNN, DCNN, BHLSTM, DLSTM, and three DBRNN models.
4) The prediction results of many other EL methods, such as MODBNE, MOMLPE, MOSVME, and ELSTM, are also recorded in Table V. It is seen that the proposed DBRNN ensemble method achieves promising prediction results compared with these EL methods, which proves the robustness of the proposed method in different working conditions.

VI. CONCLUSION

In this article, a new DBRNN ensemble method is proposed for accurate and robust RUL prediction of the AE. In this method, three kinds of DBRNN models with different neuron structures are set up to extract hidden features from sensory data in both forward and backward directions simultaneously. A new customized loss function is designed to assess the performance of these DBRNN models. The cross-validation and early stopping techniques are introduced to prevent the models from overfitting. Meanwhile, an EL method for RUL estimation is established by using multiple RDT models. The proposed method is evaluated on the C-MAPSS dataset. The experimental results show that the proposed method has a better performance compared with other existing methods.

In the future, we will explore distributed adaptive methods by considering the limited amount of historical monitoring data among the dataset in different operating conditions so that the proposed method might be adaptively adjusted to achieve the diversification and generalization of the application scenarios.

REFERENCES

[1] C. Wang, N. Lu, Y. Cheng, and B. Jiang, "A data-driven aero-engine degradation prognostic strategy," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1531–1541, Mar. 2021, doi: 10.1109/TCYB.2019.2938244.
[2] C. L. P. Chen and T.-H. Guo, "Design of intelligent acceleration schedules for extending the life of aircraft engines," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 5, pp. 1005–1015, Sep. 2007.

[3] V. Raveendran, M. Andresen, and M. Liserre, "Improving onboard converter reliability for more electric aircraft with lifetime-based control," *IEEE Trans. Ind. Electron.*, vol. 66, no. 7, pp. 5787–5796, Jul. 2019.

[4] J. Wu, C. Wu, S. Cao, S. W. Or, C. Deng, and X. Shao, "Degradation data-driven time-to-failure prognostics approach for rolling element bearings in electrical machines," *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 529–539, Jan. 2019.

[5] Y. Z. Zhang, R. Xiong, H. He, and M. G. Pecht, "Lithium-ion battery remaining useful life prediction with box–cox transformation and Monte Carlo simulation," *IEEE Trans. Ind. Electron.*, vol. 66, no. 2, pp. 1585–1597, Feb. 2019.

[6] M. Kordestani, M. Saif, M. E. Orchard, R. Razavi-Far, and K. Khorasani, "Failure prognosis and applications—A survey of recent literature," *IEEE Trans. Rel.*, vol. 70, no. 2, pp. 728–748, Jun. 2021, doi: 10.1109/TR.2019.2930195.

[7] P. Lall, S. Deshpande, and L. Nguyen, "ANN based RUL assessment for copper-aluminum wirebonds subjected to harsh environments," in *Proc. IEEE Int. Conf. Prognost. Health Manag. (ICPHM)*, 2016, pp. 1–10.

[8] L. Zhang, Z. Liu, D. Luo, J. Li, and H. Huang, "Review of remaining useful life prediction using support vector machine for engineering assets," in *Proc. Int. Conf. Qual. Rel. Risk Maint. Safety Eng. (QR2MSE)*, Jul. 2013, pp. 1793–1799.

[9] S. Adams, P. A. Beling, and R. Cogil, "Feature selection for hidden Markov models and hidden semi-Markov models," *IEEE Access*, vol. 4, pp. 1642–1657, 2016.

[10] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos, "Supervised learning of semantic classes for image annotation and retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 3, pp. 394–410, Mar. 2007.

[11] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp.2306–2318, Oct. 2017.

[12] Y. Cheng, M. Lin, J. Wu, H. Zhu, and X. Shao, "Intelligent fault diagnosis of rotating machinery based on continuous wavelet transform-local binary convolutional neural network," *Knowl. Based Syst.*, vol. 216, Mar. 2021, Art no. 106796.

[13] F. Xue, P. Bonissone, A. Varma, W. Z. Yan, N. Eklund, and K. Goebel "An instance-based method for remaining useful life estimation for aircraft engines," *J. Failure Anal. Prevent.*, vol. 8, no. 2, pp. 199–206, Apr. 2008.

[14] J. Wu, Y. Su, Y. Cheng, X. Shao, C. Deng, and C. Liu, "Multi-sensor information fusion for remaining useful life prediction of machining tools by adaptive network based fuzzy inference system," *Appl. Soft Comput.*, vol. 68, pp. 12–23, Jul. 2018.

[15] K. Javed, R. Gouriveau, and N. Zerhouni, "A new multivariate approach for prognostics based on extreme learning machine and fuzzy clustering," *IEEE Trans. Cybern.*, vol. 45, no. 12, pp. 2626–2639, Dec. 2015.

[16] W. Ahmad, S. A. Khan, and J.-M. Kim, "A hybrid prognostics technique for rolling element bearings using adaptive predictive models," *IEEE Trans. Ind. Electron.*, vol. 65, no. 2, pp. 1577–1584, Feb. 2018.

[17] X. Li, Q. Ding, and J. Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Rel. Eng. Syst. Safety*, vol. 172, pp. 1–11, Apr. 2018.

[18] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, 1990.

[19] Y. Cheng, H. Zhu, J. Wu, and X. Y. Shao, "Machine health monitoring using adaptive kernel spectral clustering and deep long short-term memory recurrent neural networks," *IEEE Trans. Ind. Informat.*, vol.15, no. 2, pp. 987–997, Feb. 2019.

[20] J. Chen, H. Jing, Y. Chang, and Q. Liu, "Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process," *Rel. Eng. Syst. Safety*, vol. 185, pp. 372–382, May 2019.

[21] F. O. Heimes, "Recurrent neural networks for remaining useful life estimation," in *Proc. Int. Conf. Prognost. Health Manag.*, Dec. 2008, pp. 1–6.

[22] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[23] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, and H. Schwenk, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. EMNLP*, Jun. 2014, pp. 1724–1734.

[24] Y. Z. Zhang, R. Xiong, H. He, and M. G. Pecht, "Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 5695–5705, Jul. 2018.

[25] S. Zhao, Y. Zhang, S. Wang, B. Zhou, and C. Cheng, "A recurrent neural network approach for remaining useful life prediction utilizing a novel trend features construction method," *Measurement*, vol. 146, pp. 279–288, Nov. 2019.

[26] J. Wu, K. Hu, Y. Cheng, H. Zhu, and Y. Wang, "Data-driven remaining useful life prediction via multiple sensor signals and deep long short-term memory neural network," *ISA Trans.*, vol. 97, pp. 241–250, Feb. 2020.

[27] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *Proc. IEEE Int. Conf. Prognost. Health Manag.*, Dec. 2008, pp. 1–9.

[28] C. Hu, B. D. Youn, P. Wang, and J. T. Yoon, "Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life," *Rel. Eng. Syst. Safety*, vol. 103, pp. 120–135, Jul. 2012.

[29] P. Lim, C. K. Goh, K. C. Tan, and P. Dutta, "Multimodal degradation prognostics based on switching Kalman filter ensemble," *IEEE Trans. Neural Netw. Learn. Syst*, vol. 28, no. 1, pp. 136–148, Jan. 2017.

[30] M. Rigamonti, P. Baraldi, E. Zio, I. Roychoudhury, and S. Pol, "Ensemble of optimized echo state networks for remaining useful life prediction," *Neurocomputing*, vol. 281, no. 15, pp. 121–138, Mar. 2018.

[31] M. Kordestani, M. F. Samadi, M. Saif, and K. Khorasani, "A new fault diagnosis of multifunctional spoiler system using integrated artificial Neural network and discrete wavelet transform methods," *IEEE Sensors J.*, vol. 18, no. 12, pp. 4990–5001, Jun. 2018.

[32] A. Liaw and M. Wiener, "Classification and regression with random forest," *R News*, vols. 2–3, pp. 18–22, Dec. 2002.

[33] X. Li and W. Lei, "AdaBoost with SVM-based component classifiers," *Eng. Appl. Artif. Intell.*, vol. 21, no. 5, pp. 785–795, 2008.

[34] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.

[35] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2014, pp. 1–15.

[36] L. I. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and regression trees (CART)," *Biometrics*, vol. 40, no. 3, p. 358, 1984.

[37] K. E. Fabricius and G. De, "Classification and regression trees: A powerful yet simple technique for ecological data analysis," *Ecology*, vol. 81, no. 11, pp. 3178–3192, 2000.

[38] C.-G. Huang, H.-Z. Huang, and Y.-F. Li, "A bi-directional LSTM prognostics method under multiple operational conditions," *IEEE Trans. Ind. Electron.*, vol. 66, no. 11, pp. 8792–8802, Nov. 2019.

[39] P. C. Bhat, H. B. Prosper, S. Sekmen, and C. Stewart, "Optimizing event selection with the random grid search," *Comput. Phys. Commun.*, vol. 228, pp. 245–257, Jul. 2018,

[40] A. Elsheikh, S. Yacout, and M. S. Ouali, "Bidirectional handshaking LSTM for remaining useful life prediction," *Neurocomputing*, vol. 323, no. 5, pp. 148–156, Jan. 2019.

[41] Z. L. Li, Z. X. Zheng, and R. Outbib, "Adaptive prognostic of fuel cells by implementing ensemble echo state networks in time varying model space," *IEEE Trans. Ind. Electron.*, vol. 67, no. 1, pp. 379–389, Jan. 2020.

[42] Z. Zhao, L. Bin, X. Wang, and W. Lu, "Remaining useful life prediction of aircraft engine based on degradation pattern learning," *Rel. Eng. Syst. Safety*, vol. 164, pp. 74–83, Aug. 2017.

[43] E. Ramasso, "Investigating computational geometry for failure prognostics," *Int. J. Prognost. Health Manag.*, vol. 5, no. 1, pp. 1–18, 2014.

[44] J. B. Coble and J. W. Hines, "Prognostic algorithm categorization with PHM challenge application," in *Proc. Int. Conf. Prognost. Health Manag.*, 2008, pp. 1–11.

[45] G. S. Babu, P. Zhao, and X. L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *Proc. Int. Conf. Database Syst. Adv. Appl.*, 2016, pp. 214–228.

[46] D. Tang, J. Cao, and J. Yu, "Remaining useful life prediction for engineering systems under dynamic operational conditions: A semi-Markov decision process-based approach," *Chin. J. Aeronaut.*, vol. 32, no. 3, pp. 627–638, 2019.

[47] W. Peng, Z. Ye, and N. Chen, "Bayesian deep-learning-based health prognostics toward prognostics uncertainty," *IEEE Trans. Ind. Electron.*, vol. 67, no. 3, pp. 2283–2293, Mar. 2020.

[48] Y. Cheng, H. Zhu, J. Wu, S. W. Or, and X. Shao, "Remaining useful life prognosis based on ensemble long short-term memory neural network," *IEEE Trans. Instrum. Meas.*, vol. 70, Oct. 2020, Art no. 3503912.

Kui Hu received the B.S. degree in mechanical engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2018, where he is currently pursuing the master's degree with the School of Naval Architecture and Ocean Engineering.

His research interests include health monitoring, and remaining useful life prediction for equipment.

Yiwei Cheng received the B.S. degree in marine engineering from Dalian Maritime University, Dalian, China, in 2016. He is currently pursuing the Ph.D. degree with the School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan, China.

His main research interests include big data analytics, health monitoring, intelligent fault diagnosis and remaining useful life prediction for equipment diagnosis, and remaining useful life prediction for equipment.

Jun Wu (Member, IEEE) received the B.S. degree from the Hubei University of Technology, Wuhan, China, in 2001, and the M.S. and Ph.D. degrees in mechanical engineering from the Huazhong University of Science and Technology (HUST), Wuhan, in 2004 and 2008, respectively.

He is currently a Full Professor with the School of Naval Architecture and Ocean Engineering, HUST. He was a Visiting Scholar with the Department of Aeronautics and Astronautics, Stanford University, Stanford, CA, USA, from 2014 to 2015, and 2019, where he conducted technical research in the area of structure health monitoring. He has more than 70 publications and holds 15 patents. His research interests include equipment health monitoring, fault diagnosis, and remaining useful life prediction.

Prof. Wu received several awards for his teaching activities.

Haiping Zhu received the B.S. degree from the Lanzhou University of Technology, Lanzhou, China, in 1998, and the M.S. and Ph.D. degrees in mechanical engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2001 and 2005, respectively.

He is currently a Full Professor with the School of Mechanical science and Engineering, HUST. He has more than 80 publications and the award of 8 patents. His research interests include modeling and optimization of manufacturing system, reliability analysis and maintenance decisions, and intelligent manufacturing and digital workshop applications.

Prof. Zhu receives several awards for his teaching activities.

Xinyu Shao received the B.S. and Ph.D. degrees in mechanical engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1990 and 1998, respectively.

He was a visiting Ph.D. student with the Department of Industrial and Manufacturing Systems Engineering, University of Michigan–Dearborn, Dearborn, MI, USA, from 1995 to 1998. He is currently a member of the Chinese Academy of Engineering and a Full Professor with the School of Mechanical Science and Engineering, HUST. He is also the Director of the National Engineering Research Center for the Digitization of Manufacturing Equipment. His research interests include digital and intelligent manufacturing, concurrent engineering, and quality/reliability engineering.

Prof. Shao was a recipient of the Ministry of Education's Chang Jiang Scholars Program Professor in 2004, the National Science Fund for Distinguished Young Scholars in 2008, and the Second-Grade State Scientific and Technological Progress Prizes in 2001, 2008, and 2012.