IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON ADVANCES IN PROGNOSTICS AND SYSTEM HEALTH MANAGEMENT

Received April 28, 2019, accepted May 22, 2019, date of publication May 28, 2019, date of current version June 21, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2919566

A Directed Acyclic Graph Network Combined With CNN and LSTM for Remaining Useful Life Prediction

JIALIN LI[1], XUEYI LI[1], AND DAVID HE[2]
[1]School of Mechanical Engineering and Automation, Northeastern University, Shenyang 110819, China
[2]The Department of Mechanical and Industrial Engineering, The University of Illinois at Chicago, Chicago, IL 60607, USA

Corresponding author: David He (davidhe@uic.edu)

ABSTRACT Accurate and timely prediction of remaining useful life (RUL) of a machine enables the machine to have an appropriate operation and maintenance decision. Data-driven RUL prediction methods are more attractive to researchers because they can be deployed quicker and cheaper compared to other approaches. The existing deep neural network (DNN) models proposed for the applications of RUL prediction are mostly single-path and top-down propagation. In order to improve the prognostic accuracy of the network, this paper proposes a directed acyclic graph (DAG) network that combines long short term memory (LSTM) and a convolutional neural network (CNN) to predict the RUL. Different from the existing prediction models combined with CNN and LSTM, the method proposed in this paper combines CNN and LSTM organically instead of just using CNN for feature extraction. Moreover, when a single timestamp is used as an input, padding the signals in the same training batch would affect the prediction ability of the developed model. To overcome this drawback, the proposed method generates a short-term sequence by sliding the time window (TW) with one step size. In addition, based on the degradation mechanism, the piece-wise RUL function is used instead of the traditional linear function. In the experimental test, the turbofan engine degradation simulation dataset provided by NASA is used to validate the proposed RUL prediction model. By comparing with the existing methods using the same dataset, it can be concluded that the prediction method proposed in this paper has better prediction capability.

INDEX TERMS Remaining useful life prediction, long-short-term memory network, convolutional neural networks, turbofan engine.

I. INTRODUCTION

The prognostics and health management (PHM) of the mechanical equipment has received much attention, and the prediction of remaining useful life (RUL) is the core of the PHM [1], [2]. By predicting the RUL of the machine, it is possible to adjust the mechanical operation and propose a maintenance strategy in a targeted manner [3]. There are three types of methods for predicting the RUL of mechanical equipment: model-based prognostics, data-driven prognostics, and hybrid approaches. Model-based prediction uses a physical understanding (physical model) of the system to predict the RUL. It can be further divided into micro-level models [4] and macro-level models [5] based on the modeling physics. Micro-level models, also known as damage propagation models, need to consider assumptions and simplifications in uncertainty management, which can impose significant limitations on the method. A macro-level model is a simplified representation of the system. It defines the relationship between input variables, state variables, and system output. Data-driven prognostics typically use pattern recognition and machine learning techniques to detect the state of the system [6]. A data-driven method is suitable for applications in the complex system, as it does not require a comprehensive understanding of the system. Modeling strategies for data-driven prediction methods can be of two types: 1) modeling cumulative damage and then inferring the damage threshold, 2) learning the RUL directly from the data.

The classical data-driven prediction methods usually use a stochastic model to describe the degradation process

The associate editor coordinating the review of this manuscript and approving it for publication was Dong Wang.

of the system. Considering it is difficult to calculate the closed-form solution due to the state-dependent model, Li *et al.* [7] proposed a general expression of age and state-dependent models to describe the system degradation processes. As different operating conditions and health conditions would lead to different degradation processes of the system, it makes the RUL prediction difficult. In order to solve this problem, a wiener-process-model (WPM)-based method [8] for RUL prediction was proposed by considering unit-to-unit variability. To predict the RUL of mechanical equipment with complex system structure and harsh operating environment, Hu *et al.* [9] combined the unscented Kalman filter with a particle filter to replace the standard particle filter, and used Markov chain Monte Carlo to improve the prediction accuracy. Qian *et al.* [10] proposed a RUL prediction method that combines enhanced phase space warping (PSW) with an improved Paris crack propagation model. In their method, PSW is enhanced by multidimensional autoregressive (AR) models to improve accurate defect tracking and the Paris crack growth model is modified by time segmentation algorithm for real-time RUL prediction.

In the past decade, due to the rise of deep learning (DL), data-driven prediction methods have focused more on the use of flexible models such as various types of neural networks (NN). Yan *et al.* [11] combined deep denoising autoencoder (DDA) and regression operation and proposed a device electrocardiogram (DECG) concept for predicting the RUL of industrial equipment. Guo *et al.* [12] found that the accuracy of predicting bearing RUL is greatly influenced by health indicators, but existing signal extraction methods cannot meet the requirements. Therefore, 6 new similarly related features and 8 classical time-frequency features were combined to form a feature set and the monotonicity and correlation metrics were used to select the most sensitive features. Finally, the selected sensitive features were used as input to the recurrent neural network (RNN). The echo state networks (ESNs) have also been commonly used for mechanical RUL prediction. Rigamonti *et al.* [13] resorted to ESNs to improve the performance of individual ESN and used the improved ESN for RUL prediction.

The main disadvantage of the data-driven method is that it has a wider confidence interval than other methods, and it requires a large amount of data for training. Moreover, it is difficult to get run-to-failure data, especially for new systems, because running a system to failure can be a lengthy and rather expensive process. So the public databases are usually used to verify the proposed model, such as battery dataset and turbofan engine degradation simulation dataset provided by the prognostics CoE at NASA Ames, FEMTO bearing dataset provided by FEMTO-ST institute.

Reusable lithium-ion batteries (LIB) have become a core component of the energy supply for most devices. Therefore, it is necessary to predict the RUL of LIB [14]. Ren *et al.* combined the autoencoder with deep neural network (DNN) to predict the RUL of LIB [15]. In order to solve the problem that the battery capacity cannot be measured in operation, Liu and Chen [16] proposed a new method combining indirect health index (HI) and multi-Gaussian process regression (GPR) model. Most existing LIB RUL prediction models have been developed using offline training data. However, the load current, temperature, and state of charge of the electric lithium-ion battery vary with the working conditions. In this case, a well-trained prediction model is not suitable for practical applications. To address this problem, Zhang *et al.* [17] proposed a RUL prediction model combining the Box–Cox transformation (BCT) and Monte Carlo (MC) simulation. In their model, BCT transforms the available capacity data and builds a linear model between the transformed capacities and cycles. MC simulation generates RUL prediction uncertainty.

Bearings are the core components of the mechanical equipment and their RUL prediction is also necessary. There are many applications for validating models using the FEMTO bearing dataset [18]. In order to overcome the problem of feature extraction methods separated from RUL prediction models, Ren *et al.* [19] proposed a multi-scale dense gate multiplexing unit network (MDGRU) to predict the RUL of bearings. Ren *et al.* [20] proposed a new method based on deep convolution neural network (DCNN) to predict RUL of bearings. A new feature extraction method was presented to obtain the spectrum-principal-energy-vector. To solve the problem that the two stages are mutually independent, Wang *et al.* [21] proposed a new model with stage correlation for RUL prediction. Many other methods to predict bearing RUL using various feature extraction methods and DL architectures have been reported [22], [23].

Aircraft PHM is one of the most important applications as any short-term faults in the equipment can have a significant impact on safety operation of the aircrafts. Khelif *et al.* [24] used support vector regression to model the direct relationship among sensor values or HIs and estimate the RUL directly from the sensor values without estimating the degradation state or failure threshold. Ordóñez *et al.* [25] combined the auto-regressive integrated moving average (ARIMA) model and support vector machine algorithms to predict turbofan engine RUL. Many engine RUL prediction models have been developed by establishing a degradation model [26]–[28]. Among all methods for engine RUL prediction, DNN-based methods account for the majority. Zhang *et al.* [29] employed a multi-objective evolutionary algorithm to evolve multiple DBNs simultaneously subject to accuracy and diversity as two conflicting objectives. Li *et al.* [30] proposed a new deep CNN (DCNN) prediction model to predict turbofan engine RUL. Long short term memory (LSTM) [31], [32] is a kind of RNN, which can solve the problem of gradient disappearance and explosion in long sequence training. Wu *et al.* [33] used the vanilla LSTM network to obtain good RUL prediction in the case of complicated operations, model degradation and strong noise. Ellefsen *et al.* [34] established a semi-supervised model for RUL prediction to provide high RUL prediction accuracy, even with reduced amounts of labeled training data.

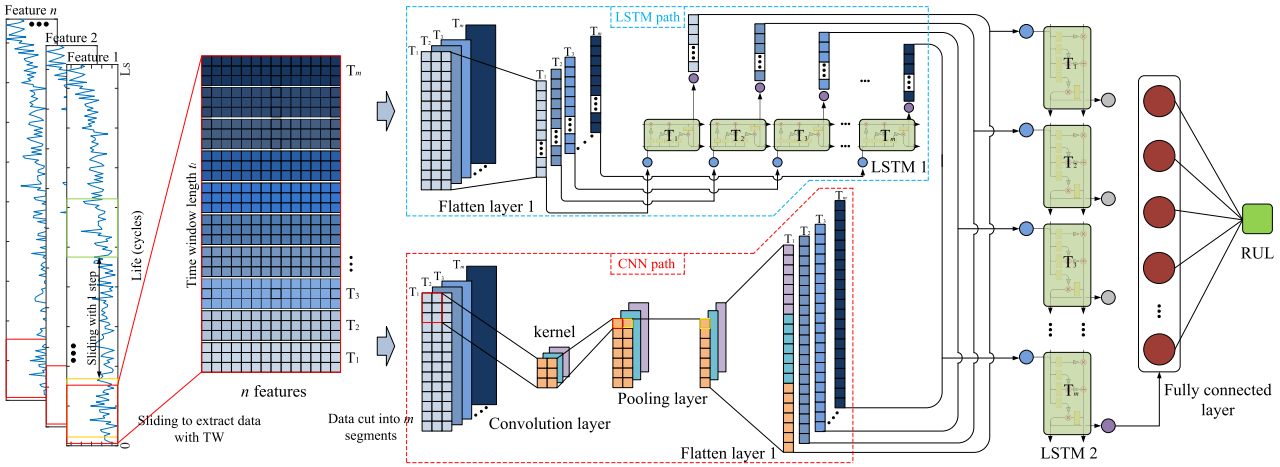


FIGURE 1. Framework of proposed DAG network.

Zhang *et al.* [35] combined transfer learning and Bi-directional long short term memory (BLSTM) network for RUL prediction. In their approach, the model can first be trained by different but related datasets and then fine-tuned by target dataset.

This paper proposes a DAG network based on LSTM and CNN to improve the accuracy of the RUL prediction of the machines. Moreover, a sliding time window (TW) is used to extract data so that the input of the prediction model has the same length of short-term time series. And the piece-wise RUL function is applied instead of the traditional RUL function. The rest of the paper is organized as follows: In Section 2, the methodology of the proposed method is introduced. In Section 3, the validation dataset, the preparation of the data, and the model evaluation method are described. In Section 4, the validation results of the proposed method using the validation database is reported. Finally, Section 5 concludes the paper.

II. METHODOLOGY

A. FRAMEWORK OF PROPOSED DAG NETWORK

This paper presents a DAG network combining CNN and LSTM networks for estimating the RUL of mechanical equipment. Although the combination of LSTM and CNN for RUL prediction has been reported in the literature, the method presented in this paper is different. Different from the DAG structure presented in this paper, the existing prediction models of using both LSTM and CNN are all combined in a serial manner. Hinchi and Tkiouat [36] used a combination of CNN and LSTM networks, first applying CNN to extract signal features and then input them into the LSTM network. However, when the CNN is used as the feature extractor, the extracted features have a great influence on the training of the LSTM network, and the CNN cannot be corrected according to the prediction error of the LSTM. To overcome this limitation, the method presented in this paper places CNN and LSTM in parallel into the DAG network. As shown in Fig. 1, the DAG network contains two paths: LSTM path and CNN path. There is no correlation between the two paths, but the output of both paths affect the RUL prediction. The constructed DAG network is a holistic model that can correct each parameter in the network according to the predicted error. The model structure of the two paths can increase the stability and accuracy of the prediction, and the compact network structure saves time and convenience in training. The collected data is simply processed and input into the DAG network to train the prediction network. The specific process is described as follows:

1) Data preparation and model development: The first step is to simply process the health-to-failure data as an input to the DAG network. As shown in Fig.1, the signal has n features and the signal length is Ls cycles, i.e., machine life span. The data is extracted by sliding the time window (TW) with the length of t_l cycles, and the sliding step size is one cycle. The size of the array extracted each time by TW is $t_l \times n$ (length of TW $\times$ numbers of features), and the number of arrays is Ls-t_l (life span−time window length). So now the input data size is $\{t_l \times n\}$, the sampling size is Ls-t_l, and the output is the corresponding RUL [Ls-t_l, Ls-t_l-1...1]. Then, the input data is transposed and cut into m pieces (T_1, $T_2 \ldots T_m$) along the column direction to obtain the data size $\{n \times (t_l/m)\} \times m$. The processed data is input into the two paths of the DAG network and the obtained results will be summed and continue to propagate forward.
2) Process of path 1: The data input to path 1 first goes through a flatten layer, and the output data size is $(n \times t_l/m) \times m$. The output data of the flatten layer will be entered as time series data into the LSTM 1 network containing m cells. Let the LSTM 1 network contain u_1 nodes. So the output data size of the LSTM 1 network is $u_1 \times m$.
3) Process of path 2: The data input to path 2 is convoluted first. The convolution operation is set as follows: filter size is $[k_1, k_2]$, i.e., convolution kernel size, the number

of kernels is n_k, and stride of the filter is $[s_1, s_2]$. The output of the convolution will be used as the input to the pooling layer. Pooling size is set to $[p_1, p_2]$ and stride is $[ps_1, ps_2]$. The data is finally processed through the flatten layer. The output data size should be the same as path 1.

4) Sum the outputs of two paths and continue to propagate forward: The output vectors of the two paths will be summed by elements-wise, which requires the output of both paths to have the same dimension. The combined data will be entered into the LSTM 2 network with the number of nodes as u_2. The output of the last cell of the LSTM 2 network will be input to a fully connected layer. The output node of the fully connected layer is one, which gives the value of the estimated RUL.
5) Correct the DAG network based on training error between the actual output and the ideal output.
6) Repeat Step 2) – 5) until the maximum number of training epochs is reached.
7) Finally, the trained network is used to predict the RUL.

B. LSTM NETWOK

The long-short term memory (LSTM) network [37] is a special recurrent neural network (RNN) proposed to solve the problem of gradient dispersion of the RNN. Fig. 2 shows the information transfer and update between the LSTM cells. Compared with the RNN network, the cell state of the LSTM has changed. It consists of a long-term state of $\mathbf{C}_t$ and a short-term state $\mathbf{h}_t$. The change in LSTM cell status relies on three control gates: forget gate, input gate, and output gate.

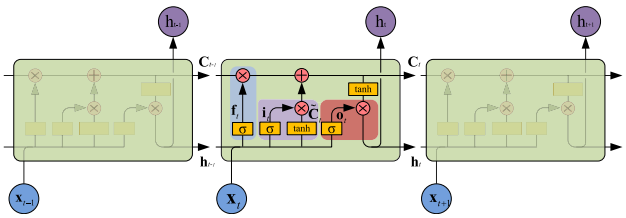


FIGURE 2. Diagram of LSTM cell.

The forget gate $\mathbf{f}_t$ is realized by (1), and its function is to selectively forget the information of the previous LSTM cell state.

$$\mathbf{f}_t = \sigma(\mathbf{W}_\mathrm{f} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_\mathrm{f}) \tag{1}$$

where $\sigma(\ )$ is activation function sigmoid $= 1/(1+ \exp(-x))$, $\mathbf{W}_\mathrm{f}$ is the weight matrix of the forgot gate, $\mathbf{h}_{t-1}$ is the short-term state of previous LSTM cell, $\mathbf{x}_t$ is the input of t-th LSTM cell, and $\mathbf{b}_\mathrm{f}$ is the bias vector of forgot gate.

The input gate consists of two parts, which are realized by (2) and (3). The vector $\mathbf{i}_t$ generated by (2) determines which information in the short-term state $\mathbf{h}_{t-1}$ is used to update the new cell state. The $\tilde{\mathbf{C}}_t$ generated by (3) will be added to the long-term cell state after being filtered by $\mathbf{i}_t$.

$$\mathbf{i}_t = \sigma(\mathbf{W}_\mathrm{i} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_\mathrm{i}) \tag{2}$$

$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{W}_\mathrm{c} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_\mathrm{c}) \tag{3}$$

where $\mathbf{W}_\mathrm{i}$ and $\mathbf{W}_\mathrm{c}$ are the weight matrix of the input gate, $\mathbf{b}_\mathrm{i}$ and $\mathbf{b}_\mathrm{c}$ are the bias vector of input gate, and activation function $\tanh = (\exp(x) - \exp(-x))/(\exp(x)+\exp(-x))$.

Then update the long-term state of $\mathbf{C}_t$ based on the output of the forget gate and the input gate.

$$\mathbf{C}_t = \mathbf{f}_t \otimes \mathbf{C}_{t-1} + \mathbf{i}_t \otimes \tilde{\mathbf{C}}_t \tag{4}$$

where $\otimes$ is the element-wise multiplication, and $\mathbf{C}_{t-1}$ is the long-term state of the previous LSTM cell.

The output gate also consists of two parts, which are realized by (5) and (6).

$$\mathbf{o}_t = \sigma(\mathbf{W}_\mathrm{o} \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t] + \mathbf{b}_\mathrm{o}) \tag{5}$$

$$\mathbf{h}_t = \mathbf{o}_t \otimes \tanh(\mathbf{C}_t) \tag{6}$$

where $\mathbf{W}_\mathrm{i}$ is the weight matrix of the output gate and $\mathbf{b}_\mathrm{i}$ is the bias vector of input gate.

C. 2D-CONVOLUTIONAL LAYER

Inspired by the visual center of a cat, the idea of convolutional neural networks (CNN) [38] has gradually emerged after several generations' efforts. The CNN has achieved good results in both speech analysis and image recognition, among which the LeNet and the Alex-Net are widely known for their super high image recognition accuracy. The weight-sharing structure of CNN makes it more similar to biological neural networks, reducing the complexity of the network model and reducing the number of weights. Moreover, CNN can use a 2-dimensional (2D) array as an input, thus avoiding the complicated feature extraction and data reconstruction process in the traditional recognition algorithms. The convolutional layer is the core of the CNN that includes convolution operations and activation operations.

The convolution operation is shown in Fig. 3. A sliding convolutional filter moves vertically and horizontally to extract data from 2D-input. The size of the filter should be the same as the size of the convolution kernel. The filter size in Fig. 3 is [3, 2], and the sliding step size for traversing the input vertically and horizontally is [2, 2]. There are three types of convolution kernels, corresponding to three feature maps. One convolution operation is as follows: the result matrix of the dot product of the convolution filter and the convolution kernel is summed, and then a bias term is added. With the translation of the convolution filter in the horizontal and vertical directions, the convolution operation is repeated to obtain a complete feature map.

The convolution operation and the activation operation can be calculated by (7) and (8).

$$z_{ij}^n = \mathrm{sum}(\mathbf{k}_n \otimes \mathbf{xf}_{ij}) + b_n \tag{7}$$

$$\mathbf{Y}^n = \varphi(\mathbf{Z}^n) \tag{8}$$

where Z_{ij}^n is the output of convolution operations, n represents the n-th feature maps, i and j correspond to the number of steps of the convolution filter in the vertical and horizontal

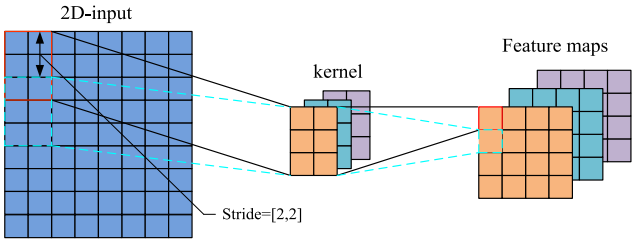


FIGURE 3. 2D-convolution operation.

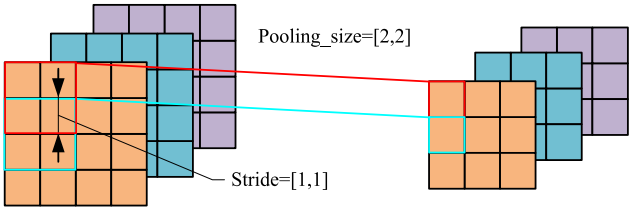


FIGURE 4. Pooling operation.

directions, sum(*) operation adds all the elements in *, $\mathbf{k}_n$ is the convolution kernel matrix, $\mathbf{xf}_{ij}$ is the filter matrix, b_{n} is the bias term, and $\varphi(\ )$ is an activation function.

D. POOLING LAYER

The main purpose of the pooling layer is to compress the input by down sampling without affecting the input quality. On one hand, the pooling operation can simplify the computational complexity of the network by compressing the input. On the other hand, feature compression is performed to extract the main features. Fig. 4 shows the pooling layer operation, where the input matrix dimension is [3, 3], pooling size is [2, 2], and pooling sliding step size for traversing the input vertically and horizontally is [1, 1]. There are many types of pooling operations, such as average pooling, maximum pooling, and overlapping pooling.

The average pooling operation and the maximum pooling operation can be calculated by (9) and (10).

$$d_{ij}^{n} = \text{Ave\_pooling}(\mathbf{yf}_{ij}) \tag{9}$$

$$d_{ij}^{n} = \text{Max\_pooling}(\mathbf{yf}_{ij}) \tag{10}$$

where d_{ij}^{n} is the output of pooling operation, n represents the n-th feature maps, i and j correspond to the number of steps of the pooling filter in the vertical and horizontal directions, $\mathbf{yf}_{ij}$ is the pooling filter matrix, Ave_pooling(*) operation takes average over all the elements in *, and Max_pooling(*) operation selects the maximum element in *.

E. FLATTEN LAYER AND FULLY CONNECTED LAYER

The role of the flatten layer is to flatten several 2D or 3D data. The output is a flatten layer a one-dimensional vector. Fig. 5 shows the flatten operation, turning the data into one dimension in order to make it as an input to the fully connected layer.

The fully connected layer connects all the nodes between the adjacent layers, so that the features extracted from the front can be integrated. Due to its fully connected nature, the full connected layer usually has the largest amount of parameters. The calculation process of the fully connected layer can be expressed by (11).

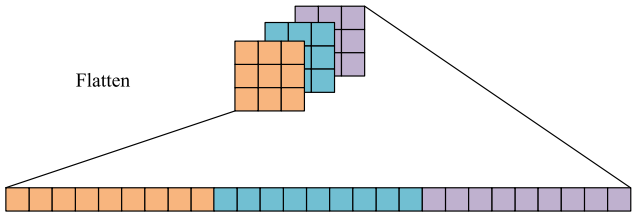


FIGURE 5. Flatten operation.

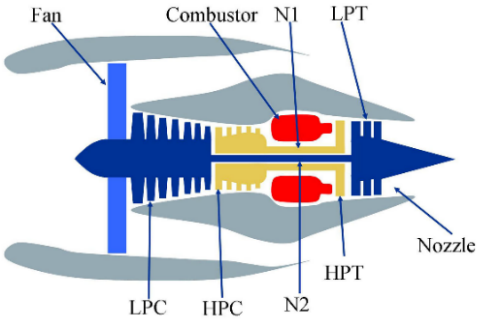


FIGURE 6. A simplified diagram of the simulation engine in C-MAPSS [39].

$$\mathbf{H} = \varphi(\mathbf{W}_{\mathrm{fc}}\mathbf{s} + \mathbf{b}_{\mathrm{fc}}) \tag{11}$$

where $\mathbf{W}_{\mathrm{fc}}$ is the weight matrix of the fully connected layer, $\mathbf{b}_{\mathrm{fc}}$ is the bias vector, $\mathbf{s}$ is the inputs, and $\mathbf{H}$ is the output matrix.

III. EXPERIMENTAL TEST

In this section, the C-MAPSS simulated turbofan engine dataset is used to validate the proposed DAG prediction model. The main contents of this section include: description of the C-MAPSS database, sensors data selection, data normalization, definition of piece-wise RUL function, and model evaluation.

A. C-MAPSS DATASET DESCRIPTION

The degradation data of the turbofan engine used in this paper was simulated by C-MAPSS developed by NASA [39]. A simplified diagram of the simulation engine is shown in Fig. 6. The main components includes: fan, low pressure compressor (LPC), high pressure compressor (HPC), combustor, high pressure turbine (HPT), low pressure turbine (LPT), and nozzle. C-MAPSS was developed on MATLAB software and Simulink environments. It includes many editable input parameters that allow the user to enter specific values, such as Fuel flow, Fan flow modifier, and Fan pressure-ratio modifier, etc. The input to the C-MAPSS contains 14 factors that affect the degradation of the turbofan engine, and the output of the simulation model represents the health condition of the turbofan engine.

TABLE 1. 21 Sensor outputs of the simulation engine running.

	Symbol	Description	Units	Trend
1	T2	Total Temperature at fan inlet	°R	~
2	T24	Total temperature at LPC outlet	°R	↑
3	T30	Total temperature at HPC outlet	°R	↑
4	T50	Total temperature LPT outlet	°R	↑
5	P2	Pressure at fan inlet	psia	~
6	P15	Total pressure in bypass-duct	psia	~
7	P30	Total pressure at HPC outlet	psia	↓
8	Nf	Physical fan speed	rpm	↑
9	Nc	Physical core speed	rpm	↑
10	Epr	Engine pressure ratio	--	~
11	Ps30	Static pressure at HPC outlet	psia	↑
12	Phi	Ratio of fuel flow to Ps30	pps/psi	↓
13	NRf	Corrected fan speed	rpm	↑
14	NRc	Corrected core speed	rpm	↓
15	BPR	Bypass ratio	--	↑
16	farB	Burner fuel-air ratio	--	~
17	htBleed	Bleed enthalpy	--	↑
18	NF_dmd	Demanded fan speed	rpm	~
19	PCNR_dmd	Demanded corrected fan speed	rpm	~
20	W31	HPT coolant bleed	lbm/s	↓
21	W32	LPT coolant bleed	lbm/s	↓

A description of the 21 simulation outputs of C-MAPSS is shown in Table 1. The legend of column 5 'Trend' represents the degradation trend of the output, where ↑ indicates that the parameter is ascending with time, ↓ indicates that the parameter is descending with time, and ~ indicates that the parameter is irregular with time. In this paper, 14 outputs with regular trend [40] are selected as inputs to the DAG network.

The C-MAPSS dataset can be divided into four sub-datasets according to different operating conditions and fault modes. A description of four sub-datasets is given in Table 2. Each sub-dataset contains training data, test data, and the actual RUL corresponding to the test data. The training data contains all the engine data from a certain health state to the fault, while the test data is a piece of data before the engine running fault. Moreover, the training and test data respectively contain a certain number of engines with different initial health states. Due to the different initial health states of the engines, the running cycles of different engines in the same database are different. Taking the FD001 database as an example, the test dataset contains 100 engines, with a maximum running cycle of 303 and a minimum running cycle of 31. In order to test all the engine in the test set, the sliding window length is usually smaller than the minimum running cycles in the test set. The data size obtained by the sliding TW processing was 30×14, the training samples was 17731, and the test sample size was 100.

B. DATA NORMALIZATION

According to Table 2, 14 out of 21 sensor outputs were selected for RUL prediction, and the output value of these 14 sensors ranged from a tens to thousands. Commonly used standardization methods: linear normalization and z-score normalization can be realized according to (12) and (13), respectively.

TABLE 2. Description of the C-MAPSS dataset.

Sub-datasets	FD001	FD002	FD003	FD004
Engines in training set	100	260	100	249
Engines in test set	100	259	100	248
Max/min cycles for training	362/128	378/128	525/145	543/128
Max/min cycles for test	303/31	367/21	475/38	486/19
Operating condition	1	6	1	6
Fault modes	1	1	2	2
TW length	30	21	36	18
Training samples	17731	48558	21120	56815
Test samples	100	259	100	248

$$y_i^c = (x_i^c - \bar{x}^c)/\sigma^c \tag{12}$$

$$y_i^c = (x_i^c - x_{\min}^c)/(x_{\max}^c - x_{\min}^c) \tag{13}$$

where x_i^c is the i-th output of sensors c, $\bar{x}^c$ is the average value of all outputs of sensor c, σ^c is the standard deviation of all outputs of sensor c, $x_{\min}^c$ is the minimum value of sensor c output, $x_{\max}^c$ is the maximum value of sensor c output, and y_i^c is the normalized data. The data normalization processing method selected in this paper is the z-score normalization.

C. RUL TARGET FUCTION

The RUL defines the time that the equipment can still operate. It is generally considered that the RUL decreases linearly with time. However, in practical application, the degradation of the equipment at the beginning of operation is not obvious. As shown in Fig. 7(a), the signals have not significant trend within the assumed health range of green. This paper applies a piece-wise linear function to represent the RUL, as shown in Fig. 7(b). According to [41] based on the CMAPSS dataset, the largest RUL value in piece-wise linear function was set to 125.

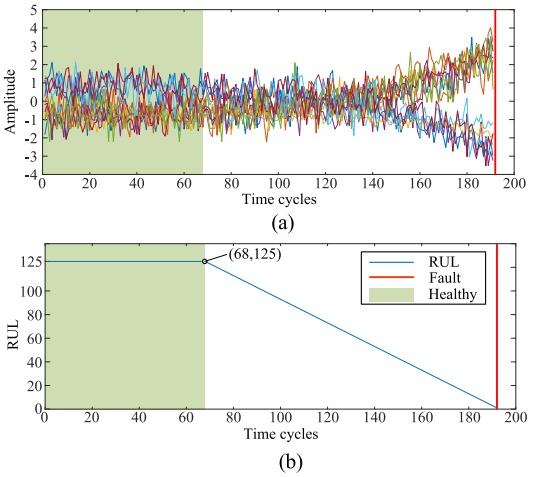


FIGURE 7. Engine #1 in FD001 dataset: (a) normalized 14 signals as inputs of DAG network and (b) piece-wise linear RUL function.

D. MODEL EVALUATION

When using the trained model to predict the RUL, there is a prediction error ($h = \mathrm{RUL}^{\mathrm{predict}} - \mathrm{RUL}^{\mathrm{actual}}$) between the predicted RUL and the actual RUL. The model can be evaluated by an evaluation method using the prediction errors. Two commonly used evaluation methods [42] are described as follow:

1) RMSE: For the evaluation of the RUL prediction model, RMSE is a commonly used method with the same penalties for early and late predictions.

$$\mathrm{RMSE} = \sqrt{\frac{1}{N} \cdot \sum_{j=1}^{N} h_j^2} \tag{14}$$

where h_j is the prediction error, N is the test sample size.

2) Score: The scoring function was first proposed at the international conference on prognostics and health management (PHM08) to be used to evaluate the data challenge model. Mathematical calculations can be achieved by (15), which have different penalties for early and late predictions.

$$\mathrm{score} = \sum_{j=1}^{N} s_j, \quad s_j = \begin{cases} \mathrm{e}^{-\frac{h_j}{13}} - 1, & h_j < 0 \\ \mathrm{e}^{\frac{h_j}{10}} - 1, & h_j \geq 0 \end{cases} \tag{15}$$

Figure 8 compares the two evaluation methods. The similarity between the two methods is that the closer the prediction error is to 0, the smaller the output. The difference between the two methods is that RMSE has the same penalties for early and late predictions, while score is different.

IV. RESULTS AND ANALYSIS

A. PREDICTED FINAL RUL OF EACH TEST ENGINE

The proposed DAG network was used to predict the RUL of the engine of the C-MAPSS datasets. Taking the FD001 database as an example, the TW length was set to 30, and the data size obtained after data cutting was $\{14 \times 3\} \times 10$. The LSTM 1 cell node was set to 21, containing 10 cells. The output of each cell was saved and input to the next layer.

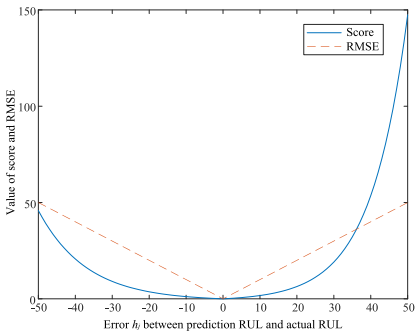


FIGURE 8. Comparison of two evaluation methods: Score and RMSE.

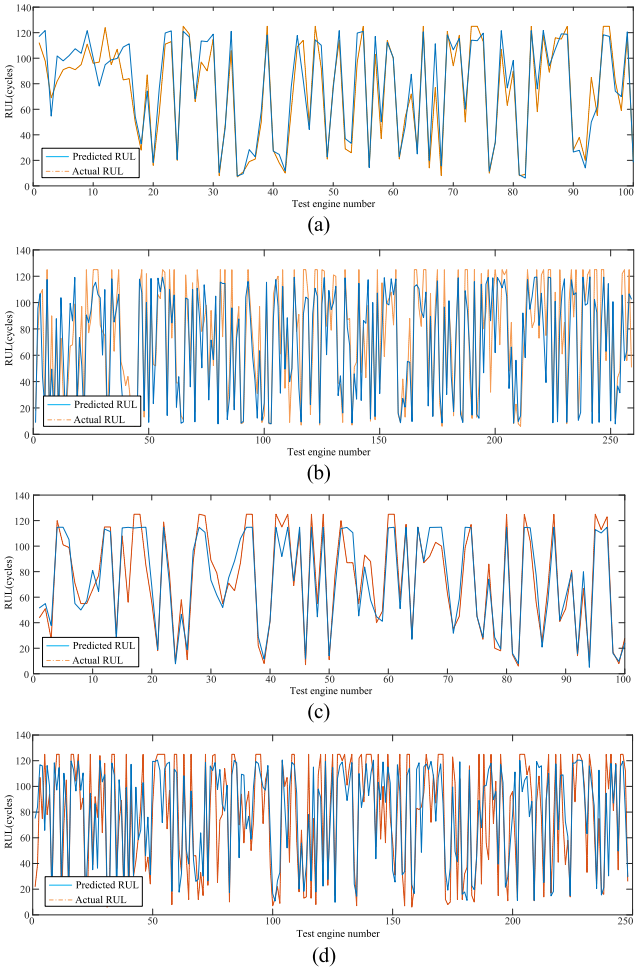


FIGURE 9. Final RUL predicted for each engine by DAG network: (a) FD001 dataset, (b) FD002 dataset, (c) FD003 dataset, and (d) FD004 dataset.

The size of the convolution kernel in path 2 was [3, 2], the number of kernel was 3, and the sliding step size was [2, 2]. The pooling size was set to [1, 2] and stride was [1, 2]. The cell node of LSTM 2 was set to 10, and the output mode was 'last mode', i.e., only the output of the last cell was saved. Finally, a fully connected layer with an output node of 1 was used. When the model was back propagated, 'rmsprop' was used as the optimizer, the learning rate was set to 0.005, and the mini-batch was set to 100. The sliding TW lengths of different sub-datasets were different, as shown in Table 2.

The DAG network was trained with four sub-datasets in the C-MAPSS dataset and tested with all test engines. The test results of each test engine in the test set are shown in Fig. 9. The 4 graphs correspond to 4 sub-datasets, the horizontal axis is the number of engines, the vertical axis is the RUL, the blue solid line in the graph represents the predicted RUL, and the yellow dash-dotted line represents the actual RUL. The number of test engines in datasets FD001 and FD003 is less than the remaining two sub-datasets. Moreover, it can be seen roughly from Fig. 9 that the coincidence degree between the predicted RUL curve and the actual RUL curve

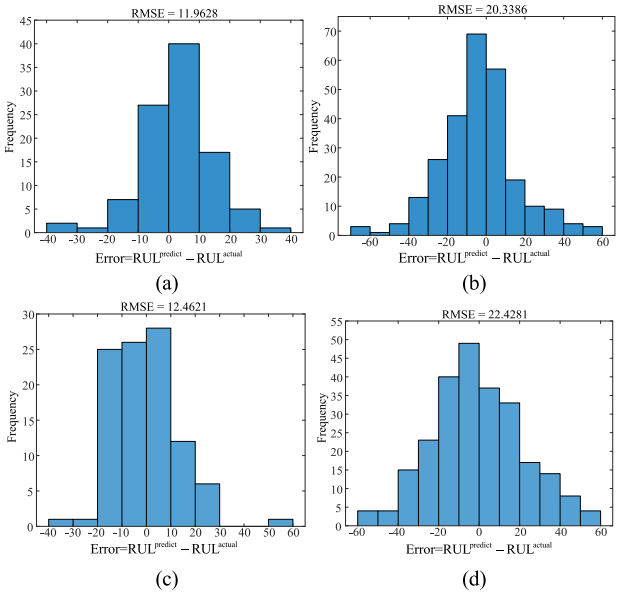


FIGURE 10. Distribution histogram of prediction error: (a) FD001 dataset, (b) FD002 dataset, (c) FD003 dataset, and (d) FD004 dataset.

in graphs (a) and (c) is better than graphs (b) and (d). This observation can be further confirmed by Fig. 10.

Fig. 10 shows the test engines error distribution histogram of the 4 test datasets, including the RMSE of each test dataset. The horizontal axis represents the error between the predicted RUL and the actual RUL. The vertical axis represents the number of engines corresponding to the error region. It can be seen from the figure that the error spans of FD001 and D003 are smaller, and the corresponding RMSE values are also lower. The prediction error distribution of FD001 and FD003 is concentrated between $[-20, 20]$, and the prediction errors of other two datasets are concentrated between $[-40, 40]$. According to Table 2, the datasets FD002 and FD004 contain data of six working conditions. They are more complicated than the other two datasets and the RUL prediction challenge for these two datasets is even greater. The reason why the prediction result of FD001 is slightly better than FD003 is that there are two fault modes in FD003.

The prediction methods validated by the C-MAPSS dataset in the past 4 years are presented in Table 3. The RMSE and score of the different prediction methods are compared in Table 4 and Table 5.

The second column in Table 4 is the year in which the prediction method was proposed. It can be found that the prediction results are gradually getting better as the new methods are proposed. It can be seen from Table 4 that the RMSE values of the 4 sub-datasets obtained by the proposed method are the lowest of all methods in the table. Similarly, the scores of the predicted RUL results are shown in Table 5. One can see from Tale V that the proposed method has the lowest scores except for in dataset FD003. It can be seen from Fig. 8 that the lower the score and RMSE, the better the RUL prediction of the model. The score of proposed method for FD003 dataset is higher than several of the compared methods because there

TABLE 3. The existing prediction methods validated by CMAPSS dataset.

Methods	Description
MLP[43]	Standard Multi-layer Perceptron algorithm
SVR[43]	Standard Support Vector Regression algorithm
RVR[43]	Standard Relevance Vector Regression algorithm
CNN[43]	CNN with 2 convolution layers + 2 pooling layers + fully connected layer
LSTM[44]	Long Short-Term Memory+ feed forward neural networks
ELM[29]	Standard Extreme Learning Machines
DBN[29]	Standard deep belief network
MODBNE[29]	A Multiobjective evolutionary algorithm integrated with the traditional DBN
BLSTM[35]	2 Bi-directional LSTM layers + flatten layer+ fully connected layer
RNN[30]	5 recurrent layers+ fully connected layer
DCNN[30]	5 convolution layers + flatten layer + fully connected layer
BiLSTM[45]	2 bidirectional LSTM layers+2 fully connected layer

TABLE 4. Compare the prediction RMSE with other methods.

Methods	Years	RMSE			
		FD001	FD002	FD003	FD004
MLP[43]	2016	37.56	80.03	37.39	77.37
SVR[43]	2016	20.96	42.0	21.05	45.35
RVR[43]	2016	23.80	31.30	22.37	34.34
CNN[43]	2016	18.45	30.29	19.82	29.16
LSTM[44]	2017	16.14	24.49	16.18	28.17
ELM[29]	2017	17.27	37.28	18.47	30.96
DBN[29]	2017	15.21	27.12	14.71	29.88
MODBNE[29]	2017	15.04	25.05	12.51	28.66
BLSTM[35]	2018	14.26	21.7	16.33	25.9
RNN[30]	2018	13.44	24.03	13.36	24.02
DCNN[30]	2018	12.61	22.36	12.64	23.31
BiLSTM[45]	2018	13.65	23.18	13.74	24.86
Proposed method	2019	**11.96**	**20.34**	**12.46**	**22.43**

TABLE 5. Compare the prediction scores with other methods.

Methods	Score			
	FD001	FD002	FD003	FD004
MLP[43]	1.80×10^4	7.80×10^6	1.74×10^4	5.62×10^6
SVR[43]	1.38×10^3	5.90×10^5	1.60×10^3	3.71×10^5
RVR[43]	1.50×10^3	1.74×10^4	1.43×10^3	2.65×10^4
CNN[43]	1.29×10^3	1.36×10^4	1.60×10^3	7.89×10^3
LSTM[44]	3.38×10^2	4.45×10^3	8.52×10^2	5.55×10^3
ELM[29]	5.23×10^2	4.98×10^5	5.74×10^2	1.21×10^5
DBN[29]	4.18×10^2	9.03×10^3	4.42×10^2	7.95×10^3
MODBNE[29]	3.34×10^2	5.59×10^3	4.22×10^2	6.56×10^3
RNN[30]	3.39×10^2	1.43×10^4	3.47×10^2	1.43×10^4
DCNN[30]	2.74×10^2	1.04×10^4	$\mathbf{2.84\times10^2}$	1.25×10^4
BiLSTM[45]	2.95×10^2	4.13×10^3	3.17×10^2	5.43×10^3
Proposed method	$\mathbf{2.29\times10^2}$	$\mathbf{2.73\times10^3}$	5.35×10^2	$\mathbf{3.37\times10^3}$

is one large late prediction in the 100 test engines. As can be seen from Fig. 9(c) and Fig. 10(c), the RUL estimate of engine #16 is a late prediction, and the prediction error is within the range of 50-60, which is the reason for the larger score.

Fig. 11 shows the effect of the TW length on the prediction results of the developed model. The figure shows the boxplot of each engine final RUL prediction error and the RMSE under different TW lengths. It can be seen from the figure that as the length of the TW increases, the engine prediction

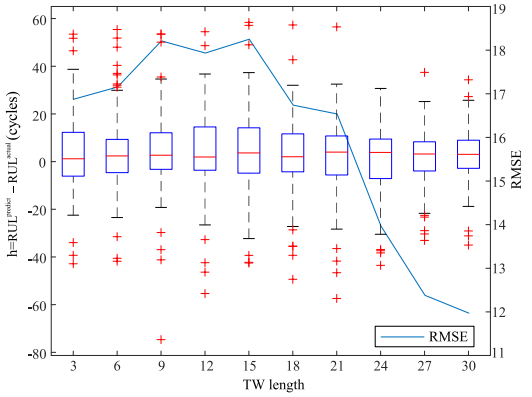


FIGURE 11. Each engine prediction error boxplot and RMSE curves of FD001 dataset under different TW length.

error is increasingly concentrated near zero and the RMSE also drops. It can be concluded that increasing the length of the time window can enhance the prediction accuracy of the model.

B. PREDICTED FULL LIFE CYCLES OF EACH TEST ENGINE

The previous section presents the results of using the model to predict the final RUL of the each test engine. Next, this section presents a complete prediction of the degradation process for each test engine. Fig. 12 shows the predicted degradation process and the actual degradation process for the 4 randomly selected engines from all the 100 test engines. During the test, the sliding TW was used to process the data. The predicted RUL corresponds to the life of the last time series in the TW. So the abscissa of the prediction curve does not start from 0. And the length of the blank before the prediction curve is equal to the length of the TW. The value at the top of the graph is the RMSE of the predicted result of the engine. Fig. 13-15 presents the predicted degradation process of 4 engines for the other three sub-datasets. The RUL prediction results of Fig. 12(c), (d), and Fig. 14(d) are very good, especially for the prediction of the RUL at the last cycle. Observing these three graphs, it can found that when the remaining period of the test engine is large, the time cycle span is also large. In contrast, it can be concluded by comparing Figures 14(b) and (d) that the smaller the predicted RUL and the time cycle span, the larger the prediction error. The longer the engine runs, the more pronounced the degradation of the engine. When the model is trained, the 'memory' of the model for the late stage of degradation is deeper than the early stage. So the predicted result is getting better and better with time cycles. Both Fig. 13(a) and Fig. 14(c) show that the prediction results in the later period of the time cycle are better than the early stage.

The actual RUL in Fig. 12(b) and 13(c) is a straight line, but the predicted RUL curve is different. The reason for this may be that this paper replaces the original linear function with a piecewise linear RUL function. So that it is possible for the engine to have a degradation trend but the corresponding RUL has not changed. Therefore, it can be inferred that the

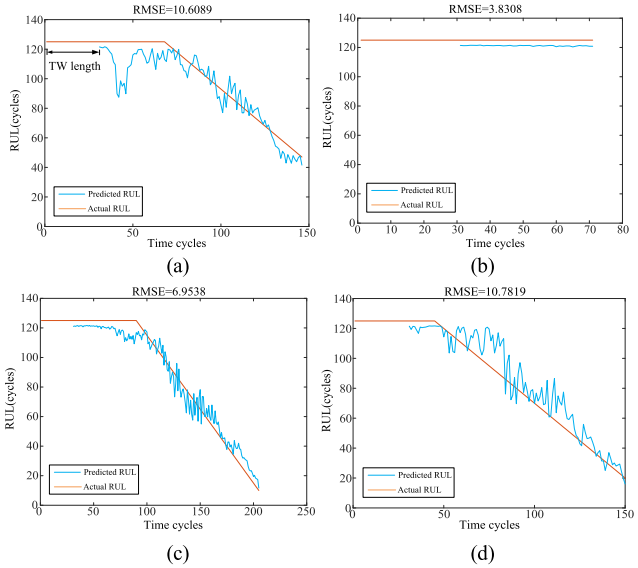


FIGURE 12. Predicted RUL of 4 engines in FD001 dataset: (a) engine # 46, (b) engine # 65, (c) engine # 76, and (d) engine # 92.

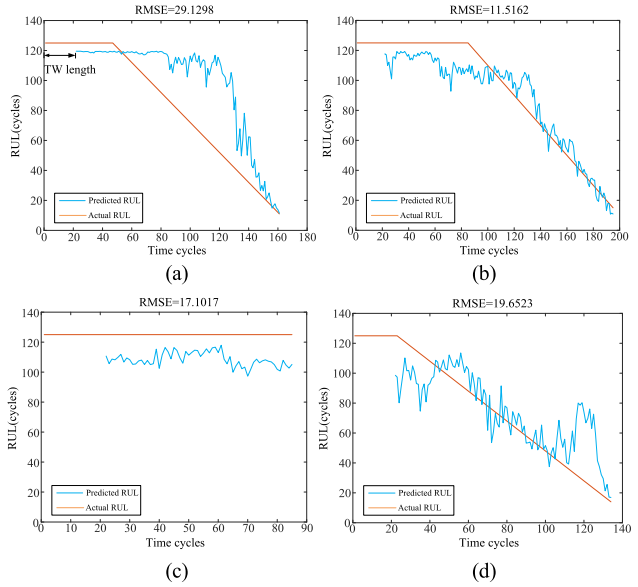


FIGURE 13. Predicted RUL of 4 engines in FD002 dataset: (a) engine # 9, (b) engine # 45, (c) engine # 170, and (d) engine # 182.

engine in Fig. 12(b) is healthy in an earlier state, while the engine in Fig. 13(c) is going to degrade. Fig. 15(b) can verify the previous inference that the predicted RUL is maintained at the early time cycle. And the predicted RUL has changed when there is a degradation trend but the actual RUL has not decreased.

Next, the prediction results of the three methods are compared, where method 1 is LSTM only without WT processing, method 2 is CNN only without data cutting, and method 3 is the proposed DAG network. Fig. 16 shows a boxplot of the RMSE of all test engine prediction results for the three methods. It can be seen from the figure that the RMSE

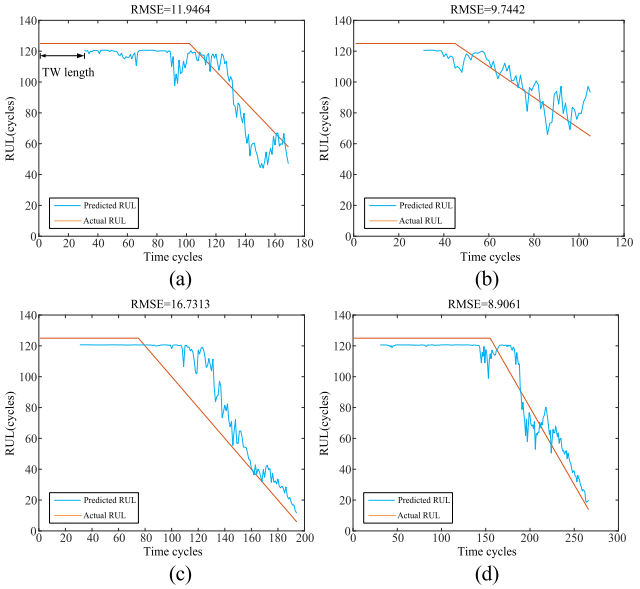


FIGURE 14. Predicted RUL of 4 engines in FD003 dataset: (a) engine # 25, (b) engine # 34, (c) engine # 85, and (d) engine # 92.

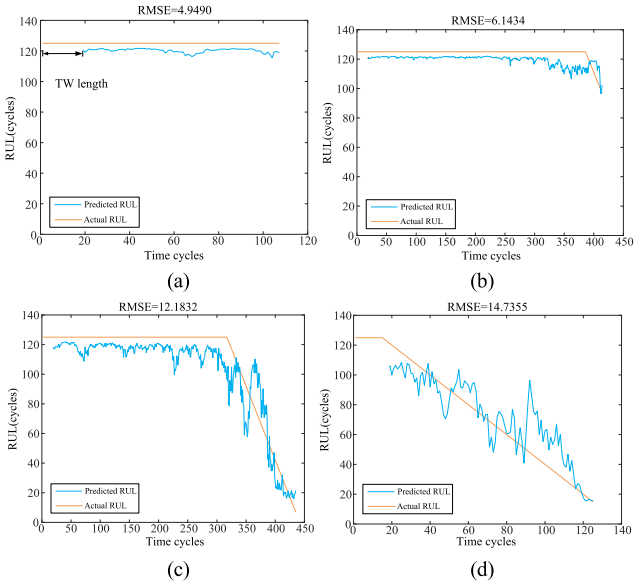


FIGURE 15. Predicted RUL of 4 engines in FD004 dataset: (a) engine # 33, (b) engine # 41, (c) engine # 135, and (d) engine # 183.

distribution region of the proposed method is lower and relatively concentrated. Method 1 without TW processing has the largest RMSE distribution span, which indicates that the prediction of early time cycles is not good when the entire time sequence is used as input.

Fig. 17 presents the distribution of all prediction engine score for the three methods. Compared to RMSE, the score distribution of each test engine is more dispersed. This is because the score for each test engine is the cumulative of all cycle errors of the engine. So the number of predicted cycles of the test engine will affect the score. Observing the results

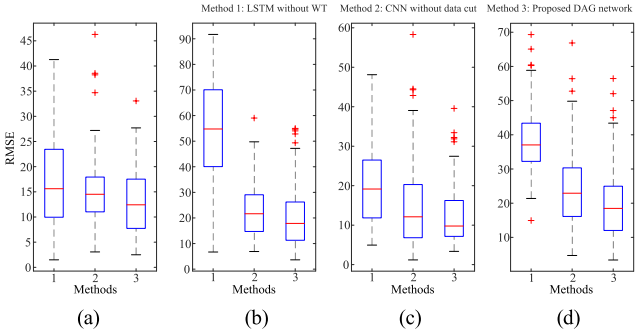


FIGURE 16. Comparison of 3 methods in terms of RMSE boxplot for each engine of the 4 sub-datasets: (a) FD001 dataset, (b) FD002 dataset, (c) FD003 dataset, and (d) FD004 dataset.

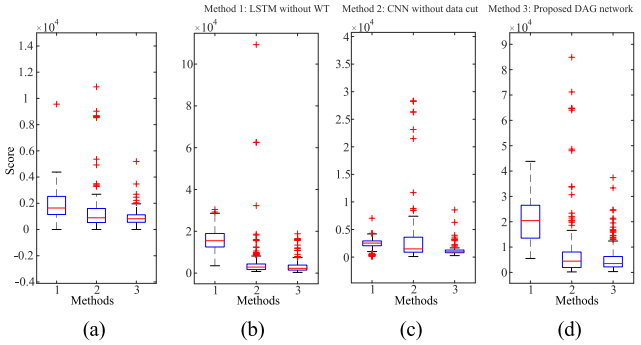


FIGURE 17. Comparison of 3 methods in terms of scores box plot for each engine of the 4 sub-datasets: (a) FD001 dataset, (b) FD002 dataset, (c) FD003 dataset, and (d) FD004 dataset.

TABLE 6. Comparison the training time of different methods.

Methods	Model description	Time
LSTM	2 LSTM layers +2 Fully connected layers	304.35s
CNN	2 Convolution layers + 2 Pooling layers+ 2 Fully connected layers	81.84s
The proposed method	DAG network	138.17s

of all the 4 datasets, it is not difficult to find that the proposed DAG method has the best prediction results.

The computational time can be reflected in the training time of the diagnostic model. Therefore, the training time is an important indicator for evaluating the quality of the diagnostic model. The FD001 dataset was first processed by TW, and then the obtained data was input into three models: LSTM, CNN, and the proposed method. The training epochs and training mini-batch were set to 40 and 200, respectively for all three models. Table 6 provides the training time for the 3 models tested. It can be seen from Table 6 that the training time of 81.84s for the CNN model is the shortest. The training time of 138.17s for the proposed method is shorter than the training time of 304.35s for the LSTM model. Because the LSTM network involves processing time series data, the training time of the proposed method and the LSTM

model is longer than the CNN model. The proposed method reduces the time series length of the LSTM network through data cutting process and flatten layer. Moreover, the CNN path and the LSTM path are in a parallel connection, the proposed method has a shorter propagation path than the LSTM methods. Therefore, the proposed method takes less training time in comparison with the LSTM model.

V. CONCLUSIONS

In this paper, a new DAG network structure including LSTM and CNN was proposed. The sliding TW was applied to process the raw data into a data table with the same length of time sequence. And the traditional linear RUL function was replaced by the improved piece-wise linear function according to the fault occurrence mechanism. The turbofan engine degradation simulation dataset provided by NASA was used to validate the developed prediction model. The following conclusions can be drawn:

1) Compared with the RUL prediction methods present in recent 4 years using the C-MAPSS dataset validation, the overall prediction results of the proposed method in terms of RMSE and score are better than other methods. This shows that the prediction capability of the proposed method is better than the existing RUL prediction methods.
2) Increasing the length of the sliding TW can improve the prediction accuracy of the developed model.
3) Since the performance degradation in the later period of the operation is more obvious, the model has a deeper 'memory' of the late stage of degradation during the training. As a result, when the whole life cycle of each engine is predicted, the prediction accuracy at the late stage of degradation is better than that at the early stage.
4) Compared with using CNN or LSTM methods alone, the prediction accuracy of the proposed two-path DAG network is better.

REFERENCES

[1] D. Wang, K.-L. Tsui, and Q. Miao, "Prognostics and health management: A review of vibration based bearing and gear health indicators," *IEEE Access*, vol. 6, pp. 665–676, 2018. doi: 10.1109/ACCESS.2017.2774261.

[2] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring," *Mech. Syst. Signal Process.*, vol. 115, pp. 213–237, Jan. 2019. doi: 10.1016/j.ymssp.2018.05.050.

[3] X.-S. Si, W. Wang, C.-H. Hu, and D.-H. Zhou, "Remaining useful life estimation—A review on the statistical data driven approaches," *Eur. J. Oper. Res.*, vol. 213, no. 1, pp. 1–14, Aug. 2011. doi: 10.1016/j.ejor.2010.11.018.

[4] W. K. Yu and T. A. Harris, "A new stress-based fatigue life model for ball bearings," *Tribol. Trans.*, vol. 44, no. 1, pp. 11–18, Jan. 2001. doi: 10.1080/10402000108982420.

[5] J. Luo, K. R. Pattipati, L. Qiao, and S. Chigusa, "Model-based prognostic techniques applied to a suspension system," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 5, pp. 1156–1168, Sep. 2008. doi: 10.1109/TSMCA.2008.2001055.

[6] A. Mosallam, K. Medjaher, and N. Zerhouni, "Data-driven prognostic method based on Bayesian approaches for direct remaining useful life prediction," *J. Intell. Manuf.*, vol. 27, no. 5, pp. 1037–1048, Oct. 2016. doi: 10.1007/s10845-014-0933-4.

[7] N. Li, Y. Lei, L. Guo, T. Yan, and J. Lin, "Remaining useful life prediction based on a general expression of stochastic process models," *IEEE Trans. Ind. Electron.*, vol. 64, no. 7, pp. 5709–5718, Jul. 2017. doi: 10.1109/TIE.2017.2677334.

[8] N. Li, Y. Lei, T. Yan, N. Li, and T. Han, "A Wiener-process-model-based method for remaining useful life prediction considering unit-to-unit variability," *IEEE Trans. Ind. Electron.*, vol. 66, no. 3, pp. 2092–2101, Mar. 2019. doi: 10.1109/TIE.2018.2838078.

[9] Y. Hu, S. Liu, H. Lu, and H. Zhang, "Online remaining useful life prognostics using an integrated particle filter," *Proc. Inst. Mech. Eng. O, J. Risk Rel.*, vol. 232, no. 6, pp. 587–597, Dec. 2018. doi: 10.1177/1748006X17751494.

[10] Y. Qian, R. Yan, and R. X. Gao, "A multi-time scale approach to remaining useful life prediction in rolling bearing," *Mech. Syst. Signal Process.*, vol. 83, pp. 549–567, Jan. 2017. doi: 10.1016/j.ymssp.2016.06.031.

[11] H. Yan, J. Wan, C. Zhang, S. Tang, Q. Hua, and Z. Wang, "Industrial big data analytics for prediction of remaining useful life based on deep learning," *IEEE Access*, vol. 6, pp. 17190–17197, 2018. doi: 10.1109/ACCESS.2018.2809681.

[12] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, "A recurrent neural network based health indicator for remaining useful life prediction of bearings," *Neurocomputing*, vol. 240, pp. 98–109, May 2017. doi: 10.1016/j.neucom.2017.02.045.

[13] M. Rigamonti, P. Baraldi, E. Zio, I. Roychoudhury, K. Goebel, and S. Poll, "Ensemble of optimized echo state networks for remaining useful life prediction," *Neurocomputing*, vol. 281, pp. 121–138, Mar. 2018. doi: 10.1016/j.neucom.2017.11.062.

[14] L. Zhang, Z. Mu, and C. Sun, "Remaining useful life prediction for lithium-ion batteries based on exponential model and particle filter," *IEEE Access*, vol. 6, pp. 17729–17740, Mar. 2018. doi: 10.1109/ACCESS.2018.2816684.

[15] L. Ren, L. Zhao, S. Hong, S. Zhao, H. Wang, and L. Zhang, "Remaining useful life prediction for lithium-ion battery: A deep learning approach," *IEEE Access*, vol. 6, pp. 50587–50598, 2018. doi: 10.1109/ACCESS.2018.2824352.

[16] J. Liu and Z. Chen, "Remaining useful life prediction of lithium-ion batteries based on health indicator and Gaussian process regression model," *IEEE Access*, vol. 7, pp. 39474–39484, 2019. doi: 10.1109/ACCESS.2019.2905740.

[17] Y. 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. doi: 10.1109/TIE.2018.2808918.

[18] L. Zhao and X. Wang, "A deep feature optimization fusion method for extracting bearing degradation features," *IEEE Access*, vol. 6, pp. 19640–19653, 2018.

[19] L. Ren, X. Cheng, X. Wang, J. Cui, and L. Zhang, "Multi-scale dense gate recurrent unit networks for bearing remaining useful life prediction," *Future Gener. Comput. Syst.*, vol. 94, pp. 601–609, May 2019. doi: 10.1016/j.future.2018.12.009.

[20] L. Ren, Y. Sun, H. Wang, and L. Zhang, "Prediction of bearing remaining useful life with deep convolution neural network," *IEEE Access*, vol. 6, pp. 13041–13049, 2018. doi: 10.1109/ACCESS.2018.2804930.

[21] H. Wang, Y. Zhao, and X. Ma, "Remaining useful life prediction using a novel two-stage wiener process with stage correlation," *IEEE Access*, vol. 6, pp. 65227–65238, 2018. doi: 10.1109/ACCESS.2018.2877630.

[22] L. Ren, J. Cui, Y. Sun, and X. Cheng, "Multi-bearing remaining useful life collaborative prediction: A deep learning approach," *J. Manuf. Syst.*, vol. 43, pp. 248–256, Apr. 2017. doi: 10.1016/j.jmsy.2017.02.013.

[23] L. Ren, Y. Sun, J. Cui, and L. Zhang, "Bearing remaining useful life prediction based on deep autoencoder and deep neural networks," *J. Manuf. Syst.*, vol. 48, pp. 71–77, Jul. 2018. doi: 10.1016/j.jmsy.2018.04.008.

[24] R. Khelif, B. Chebel-Morello, S. Malinowski, E. Laajili, F. Fnaiech, and N. Zerhouni, "Direct remaining useful life estimation based on support vector regression," *IEEE Trans. Ind. Electron.*, vol. 64, no. 3, pp. 2276–2285, Mar. 2017. doi: 10.1109/TIE.2016.2623260.

[25] C. Ordóñez, F. S. Lasheras, J. Roca-Pardiñas, and F. J. de Cos Juez, "A hybrid ARIMA–SVM model for the study of the remaining useful life of aircraft engines," *J. Comput. Appl. Math.*, vol. 346, pp. 184–191, Jan. 2019. doi: 10.1016/j.cam.2018.07.008.

[26] Z. Li, K. Goebel, and D. Wu, "Degradation modeling and remaining useful life prediction of aircraft engines using ensemble learning," *J. Eng. Gas Turb. Power*, vol. 141, no. 4, p. 041008, Nov. 2018. doi: 10.1115/1.4041674.

[27] F. Khan, O. F. Eker, A. Khan, and W. Orfali, "Adaptive degradation prognostic reasoning by particle filter with a neural network degradation model for turbofan jet engine," *Data*, vol. 3, no. 4, p. 49, Nov. 2018. doi: 10.3390/data3040049.

[28] Z. Zhao, B. Liang, X. Wang, and W. Lu, "Remaining useful life prediction of aircraft engine based on degradation pattern learning," *Reliab. Eng. Syst. Safe.*, vol. 164, pp. 74–83, Aug. 2017. doi: 10.1016/j.ress.2017.02.007.

[29] 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. doi: 10.1109/TNNLS.2016.2582798.

[30] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Rel. Eng. Syst. Saf.*, vol. 172, pp. 1–11, Apr. 2018. doi: 10.1016/j.ress.2017.11.021 .

[31] J. Zhang, P. Wang, R. Yan, and R. X. Gao, "Long short-term memory for machine remaining life prediction," *J. Manuf. Syst.*, vol. 48, pp. 78–86, Jul. 2018. doi: 10.1016/j.jmsy.2018.05.011.

[32] A. Elsheikh, S. Yacout, and M.-S. Ouali, "Bidirectional handshaking LSTM for remaining useful life prediction," *Neurocomputing*, vol. 323, pp. 148–156, Jan. 2019. doi: 10.1016/j.neucom.2018.09.076.

[33] Y. T. Wu, M. Yuan, S. Dong, L. Li, and Y. Liu, "Remaining useful life estimation of engineered systems using vanilla LSTM neural networks," *Neurocomputing*, vol. 275, pp. 167–179, Jan. 2018. doi: 10.1016/j.neucom.2017.05.063.

[34] A. L. Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, and H. Zhang, "Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture," *Rel. Eng. Syst. Saf.*, vol. 183, pp. 240–251, Mar. 2019. doi: 10.1016/j.ress.2018.11.027.

[35] A. Zhang, H. Wang, S. Li, Y. Cui, Z. Liu, G. Yang, and J. Hu, "Transfer learning with deep recurrent neural networks for remaining useful life estimation," *Appl. Sci.*, vol. 8, no. 12, p. 2416, Nov. 2018. doi: 10.3390/app8122416.

[36] A. Z. Hinchi and M. Tkiouat, "Rolling element bearing remaining useful life estimation based on a convolutional long-short-term memory network," *Procedia Comput. Sci.*, vol. 127, pp. 123–132, 2018. doi: 10.1016/j.procs.2018.01.106.

[37] E. Chemali, P. J. Kollmeyer, M. Preindl, R. Ahmed, and A. Emadi, "Long short-term memory networks for accurate state-of-charge estimation of Li-ion batteries," *IEEE Trans. Ind. Electron.*, vol. 65, no. 8, pp. 6730–6739, Aug. 2018. doi: 10.1109/TIE.2017.2787586.

[38] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, "Recent advances in convolutional neural networks," *Pattern Recognit.*, vol. 77, pp. 354–377, May 2018. doi: 10.1016/j.patcog.2017.10.013.

[39] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage propagation modeling for aircraft engine run-to-failure simulation," in *Proc. Int. Conf. Prognostics Health Manage.*, Denver, CO, USA, 2008, pp. 1–9. doi: 10.1109/PHM.2008.4711414.

[40] C. Zheng, W. Liu, B. Chen, D. Gao, Y. Cheng, Y. Yang, X. Zhang, S. Li, Z. Huang, and J. Peng, "A data-driven approach for remaining useful life prediction of aircraft engines," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Maui, HI, USA, 2018, pp. 184–189. doi: 10.1109/ITSC.2018.8569915.

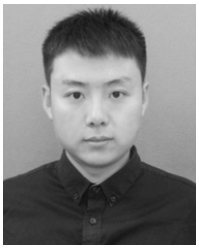
[41] F. Li, L. Zhang, B. Chen, D. Gao, Y. Cheng, X. Zhang, Y. Yang, K. Gao, Z. Huang, and J. Peng, "A light gradient boosting machine for remainning useful life estimation of aircraft engines," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Maui, HI, USA, 2018, pp. 3562–3567. doi: 10.1109/ITSC.2018.8569801.

[42] S. Wang, X. Zhang, D. Gao, B. Chen, Y. Cheng, Y. Yang, W. Yu, Z. Huang, and J. Peng, "A remaining useful life prediction model based on hybrid long-short sequences for engines," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Maui, HI, USA, 2018, pp. 1757–1762. doi: 10.1109/ITSC.2018.8569668.

[43] G. S. Babu, P. Zhao, and X.-L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *Database Syst. for Adv. Appl.*, vol. 9642, S. B. Navathe, W. Wu, S. Shekhar, X. Du, X. S. Wang, and H. Xiong, Eds. Cham, Switzerland: Springer, 2016, pp. 214–228.

[44] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *Proc. IEEE Int. Conf. Prognostics Health Manage. (ICPHM)*, Dallas, TX, USA, Jun. 2017, pp. 88–95. doi: 10.1109/ICPHM.2017.7998311.

[45] J. Wang, G. Wen, S. Yang, and Y. Liu, "Remaining useful life estimation in prognostics using deep bidirectional LSTM neural network," in *Proc. Prognostics Syst. Health Manage. Conf. (PHM-Chongqing)*, Chongqing, China, 2018, pp. 1037–1042. doi: 10.1109/PHM-Chongqing.2018.00184.



JIALIN LI was born in Shenyang, China. He received the B.S. degree from the School of Mechanical Engineering, Shenyang University of Technology, China, in 2015, and the M.S. degree from the School of Mechanical Engineering and Automation, Northeastern University, China, in 2017, where he is currently pursuing the Ph.D. degree. His current research interests include data-driven methods to fault diagnosis, pattern recognition, deep learning, remaining useful life estimation, and their application to critical components of mechanical equipment.

XUEYI LI was born in Harbin, China. He received the bachelor's degree in vehicle engineering from the Heilongjiang Institute of Technology, China, in 2013, and the master's degree in mechanical engineering from Northeastern University, China, in 2017, where he is currently pursuing the Ph.D. degree with the School of Mechanical Engineering and Automation. He was with Beiyou Electronic Control Fuel Injection System (Beijing) Stock Company, Ltd., for two years. His current research interests focus on prognostic and health management, especially on the method of mechanical fatigue diagnosis and residual life prediction based on the integration of various sensing technologies.

DAVID HE received the B.S. degree in metallurgical engineering from the Shanghai University of Technology, China, in 1982, the MBA degree from the University of Northern Iowa, in 1990, and Ph.D. degree in industrial engineering from the University of Iowa, in 1994. He is currently a Professor and the Director of the Intelligent Systems Modeling & Development Laboratory, Department of Mechanical and Industrial Engineering, The University of Illinois at Chicago. His research areas include: machinery health monitoring, diagnosis and prognosis, complex systems failure analysis, quality and reliability engineering, and manufacturing systems design, modeling, scheduling, and planning.