Complex & Intelligent Systems (2022) 8:1409–1424
https://doi.org/10.1007/s40747-021-00606-4

ORIGINAL ARTICLE

Machine remaining life prediction based on multi-layer self-attention and temporal convolution network

Zhiwu Shang[1,2] · Baoren Zhang[1,2] · Wanxiang Li[1,2] · Shiqi Qian[1,2] · Jie Zhang[1,2]

Received: 17 June 2021 / Accepted: 3 December 2021 / Published online: 20 December 2021

Abstract

Convolution neural network (CNN) has been widely used in the field of remaining useful life (RUL) prediction. However, the CNN-based RUL prediction methods have some limitations. The receptive field of CNN is limited and easy to happen gradient vanishing problem when the network is too deep. The contribution differences of different channels and different time steps to RUL prediction are not considered, and only use deep learning features or handcrafted statistical features for prediction. These limitations can lead to inaccurate prediction results. To solve these problems, this paper proposes an RUL prediction method based on multi-layer self-attention (MLSA) and temporal convolution network (TCN). The TCN is used to extract deep learning features. Dilated convolution and residual connection are adopted in TCN structure. Dilated convolution is an efficient way to widen receptive field, and the residual structure can avoid the gradient vanishing problem. Besides, we propose a feature fusion method to fuse deep learning features and statistical features. And the MLSA is designed to adaptively assign feature weights. Finally, the turbofan engine dataset is used to verify the proposed method. Experimental results indicate the effectiveness of the proposed method.

Keywords Temporal convolution network · Multi-layer self-attention · Remaining useful life prediction · Feature fusion

Introduction

Condition-based maintenance (CBM) is a maintenance strategy that monitors equipment health conditions in real-time and makes optimal maintenance decisions based on monitoring information [12]. This strategy can avoid unnecessary maintenance plans and ensure the reliability of equipment operation. It has been widely used in recent years [8]. Health prognostics is one of the major tasks in CBM, it can provide important guidance for equipment maintenance. Thus, accurate prediction of remaining useful life (RUL) is significant for preventive maintenance decisions of equipment.

The current RUL prediction methods can be divided into model-based and data-driven methods [12]. The model-based prediction methods are based on the internal working mechanism of the object system and establish the mathematical model that can reflect the physical laws of degradation. The mathematical model can go deep into the essence of the object system and obtain accurate prediction results. However, it is difficult to establish an accurate mathematical model to reflect the physical laws of degradation in practical applications. Establishing a model from the internal mechanism of the system requires a large amount of expert knowledge. This is often impossible to establish an accurate mathematical model, especially when the degradation process is complicated and the degradation mechanism is unclear.

In recent years, with the development of big data and intelligence, data-driven methods have been more and more widely used [21–23]. Data-driven methods can be further divided into statistical model-based methods and artificial intelligence (AI) methods [4]. The statistical model-based

✉ Zhiwu Shang
shangzhiwu@126.com

Baoren Zhang
baoren_zhang@yeah.net

Wanxiang Li
lwxwolf@yeah.net

Shiqi Qian
qianshiqi777@163.com

Jie Zhang
tguzhangjie@163.com

[1] School of Mechanical Engineering, Tiangong University, Tianjin 300387, China

[2] Tianjin Modern Electromechanical Equipment Technology Key Laboratory, Tianjin 300387, China

methods predict the RUL by establishing a statistical model based on empirical knowledge. The statistical models used for RUL prediction include autoregressive model [16], random coefficient model [12], Wiener process model [29], etc. In these methods, the RUL prediction model is constructed by fitting available observations into a random coefficient model or a random process model under the probabilistic method, without relying on any physics or principles.

The AI-based methods attempt to use AI algorithms to learn the mechanical degradation patterns from large amounts of data. It is usually necessary to extract some features that are sensitive to degradation from the raw data through manual methods or deep learning algorithms. Then realize the mapping between features and RUL through AI algorithms. With the advent of the big data era, massive amounts of industrial data have created favorable conditions for AI-based methods [6]. In this paper, we mainly focus on AI-based RUL prediction methods. AI-based methods can be divided into shallow machine learning algorithms and deep learning algorithms [4]. The shallow models used for RUL prediction include support vector machine (SVM) [15, 31], random forest (RF) [33], decision tree (DT) [28], etc. Since the trend of the raw data is unclear and contains noise [14], it is necessary to extract features from the raw data before inputting the model. Zan et al. [31] extracted statistical features in the time domain, frequency domain, and time–frequency domain from bearing vibration signals. Then, multiple features were fused into one fusion feature, and the particle swarm optimization support vector machine was used to predict RUL.

Another widely used AI-based RUL prediction method is the deep learning algorithm. Deep learning methods are representation learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transforms the representation at one level into a representation at a higher, slightly more abstract level [11]. Compared with shallow models, deep learning algorithms can automatically extract representative features from raw data. CNN is one of the most popular deep learning algorithms. Due to its shared parameter convolution kernel, CNN performs well in spatial feature extraction and has been successfully applied to RUL prediction. Babu et al. [1] first used CNN for turbofan engine RUL prediction. Unlike the CNN structure used in computer vision, the convolution and pooling operations in this method were performed along the time dimension of multi-channel data. The results showed that CNN performed better than shallow models such as MLP. Li et al. [13] proposed a multi-scale deep convolution neural network and used raw sensory data as input to the model to predict RUL. Ren et al. [18] proposed a new feature extraction method, named the Spectrum-Principal-Energy-Vector, and input this feature into an eight-layer CNN to predict the RUL of the bearing. Cheng et al. [5] used the Hilbert–Huang transform to construct a new health indicator, named the degradation energy indicator. This indicator was used as the label to train a seven-layer CNN model and predicted the bearing RUL through SVM. However, the receptive field of CNN is limited and easy to happen gradient vanishing problem when the network is too deep. Due to the limited receptive field, it is difficult for the network to capture the features in the long time series and miss some important degradation information. Another disadvantage is that when the network is too deep, gradient explosion and gradient disappearance are easy to occur during training, which makes training more difficult.

Considering these shortcomings of CNN, Bai et al. [2] proposed TCN. TCN increases the receptive field by dilated convolution, so the model can receive more historical information. Meanwhile, TCN uses the residual connection to make the model deeper and extract more abstract features. However, there are few studies on TCN for RUL prediction.

In the above RUL prediction methods based on deep learning, different channel signals or features extracted from the signals are used as input to the model. Then, the deep learning algorithm is used to extract features (hereafter this text, the features extracted by the deep learning algorithm will be abbreviated as deep learning features) from input data and establish the mapping relationship between deep learning features and RUL. However, in the network construction process, they assumed that the input data obtained by different channels at different times contributed equally to the output. But in reality, different channels and different time steps have different contributions to RUL prediction. For example, some channels may contain more degradation information, while some contain less. If this difference is not considered, the model will be affected by irrelevant information, resulting in low prediction accuracy and poor generalization ability. The attention mechanism can relate the features at different locations and assign weights to these features, thereby enhancing the contribution of important features to RUL prediction. Chen et al. [3] introduced the attention mechanism into RUL prediction. They integrated recurrent neural network and attention mechanism to establish an RUL prediction model. The frequency domain features of bearing vibration signals were used as the model input. This method obtained high prediction accuracy.

In this paper, we proposed an RUL prediction method based on MLSA and TCN. The main contributions of this research are as follows:

1. The proposed method integrates MLSA and TCN to extract deep learning features. The proposed method utilizes MLSA for adaptively assigning weights to different channels and different time steps, thereby enhancing the contribution of important channels and time steps to RUL

prediction. And the feature representation of the data is obtained by TCN.

2. A new feature fusion method for RUL prediction is proposed. Studies have shown that manually extracted statistical features also contain rich degradation information [10]. The proposed model can take both deep learning features and statistical features into consideration for RUL prediction. And considering the contribution differences of different source features to RUL prediction, the attention mechanism is used to adaptively assign weights to different source features.
3. To evaluate the proposed method, four experiments are conducted on the turbofan engine dataset. And we select a sample to visualize the prediction process to understand the contribution differences of different features to RUL prediction.

The content of this paper is arranged as follows: "Methodology" introduces details of the proposed method. "Experimental study and analysis" introduces the turbofan engine dataset and the data preprocessing methods. The effectiveness of the proposed method is verified by four experiments. The results are analyzed and discussed in this section. To understand the contribution differences of different features to RUL prediction, the attention weights are visualized. Finally, conclusions are drawn in "Conclusion".

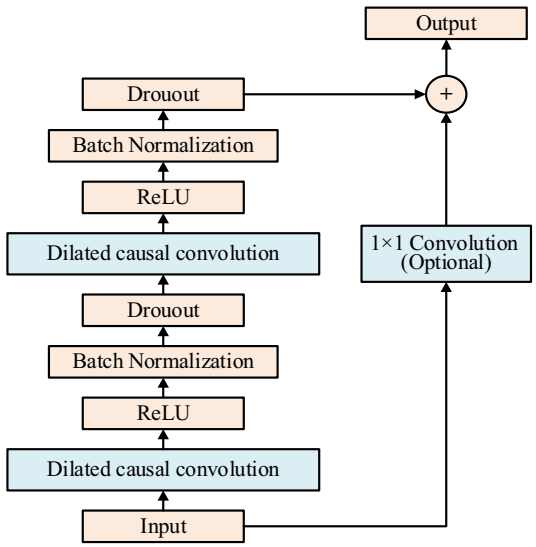

Fig. 1 Structure of a residual block in the TCN [2]

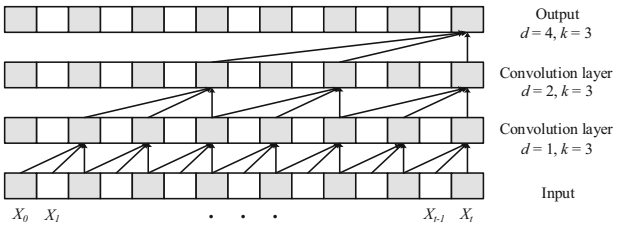

Fig. 2 A dilated causal convolution with dilation factors $d = 1, 2, 4$ [2]

Methodology

Temporal convolution network

Bai et al. [2] proposed TCN in 2018. TCN is composed of several residual blocks. Each residual block contains convolution layers, dropout layers, batch normalization layers and adopts residual connection. The structure of a residual block in the TCN is shown in Fig. 1.

Unlike CNN, TCN uses dilated causal convolution to increase the range of receptive field. A dilated causal convolution with dilation factors $d = 1, 2, 4$ is shown in Fig. 2. The biggest difference between dilated convolution and normal convolution is that holes are injected into the convolution kernel. The hyperparameter of the dilated convolution is the dilation rate d, which indicates the number of holes between adjacent notes in the convolution kernel. When $d = 1$, it means normal convolution operation. When $d = 2$ denotes that the inner interval of the convolution kernel is 2. The size of the dilated convolution kernel k' and the receptive field L can be calculated as follows:

$$k' = d \cdot (k - 1) + 1, \tag{1}$$

$$L = \left(\sum_{D} d \cdot (k - 1) \right) + 1, \tag{2}$$

where k is the convolution kernel size, D is the dilation rate array $\{d_1, d_2, \ldots, d_n\}$, and n is the number of dilated convolution layers. According to Eq. (2), choosing larger kernel size or increasing the dilation rate can increase the range of the receptive field.

By using causal convolution, the output at time t only depends on the values at time t and before in the previous layer, that is:

$$y_t^{n+1} = f(x_1^n, \ldots, x_t^n), \tag{3}$$

where n is the layer number, $f(\cdot)$ represents the convolution operation, y_t^{n+1} represents the output of the $(n + 1)$th layer at time t. Different from the traditional convolution, the causal convolution does not use data of future time.

The residual connection is beneficial for model training. The residual block input and the output of the last layer

are connected through the residual connection, as shown in Eq. (4) [2].

$$O = Activation(X + F(X)), \tag{4}$$

where X is the input of the residual block, $F(X)$ is the output of the last layer of the residual block, O is the output of the residual block, and $Activation(\cdot)$ is the activation function like sigmoid. By residual connection, the input skips many layers and connects to the last layer of the residual block, which protects the integrity of the information to some extent, alleviates gradient explosion and gradient vanishing, and enables the model to extract high dimensional features.

Self-attention mechanism

The idea of attention mechanism [7, 26] originates from human vision. When humans find that a part of a scene often has something they want to observe, they will learn to focus on that part when a similar scene appears again and focus more attention on the useful part. This is a way for humans to quickly select high-value information from massive information using limited processing resources [17]. The attention mechanism in deep learning simulates this process. When the neural network finds the key information of the input data, it will assign a higher weight to the key information to enhance its contribution to the result.

Self-attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence [25]. It has been successfully applied in different research fields [27, 30, 32]. The self-attention mechanism assigns weights to different features based on the dependencies between features. The purpose is to reduce the dependence on external information and use the inherent information within the features to allocate attention as much as possible. The equations of the self-attention mechanism are Eqs. (5)–(7) [4]. The calculation process is as follows:

1. The sample is represented as $H = \{h_1, h_2, \ldots, h_i, \ldots, h_n\}$, $h_i \in R^n$, where n is the sequence length of the feature. First, the ith feature h_i is scored according to the importance of the i-th feature:

   where $\varphi(\cdot)$ is the scoring function, such as sigmoid function and linear function, etc.

   $$s_i = \varphi(W \cdot h_i + b), \tag{5}$$

2. After obtaining the score s_i corresponding to h_i, the score can be normalized by softmax function:

   where α_i is the attention weight assigned by the self-attention mechanism to h_i.

   $$\alpha_i = \text{softmax}(s_i) = \frac{\exp(s_i)}{\sum_i \exp(s_i)}, \tag{6}$$

3. The final output O of the self-attention mechanism is:

   $$O = H \otimes A = \{\alpha_1 h_1, \alpha_2 h_2, \ldots, \alpha_n h_n\}, \tag{7}$$

   where $A = \{\alpha_1, \alpha_2, \ldots, \alpha_n\}$, $\otimes$ is the element-wise multiplication operation.

Procedure of proposed method

This section describes the specific steps of the proposed method. Figure 3 shows the framework of the proposed method. In the deep learning features extraction part, first, the self-attention mechanism is used to adaptively assign weights to different channels, and then the deep learning features are extracted through TCN. After that, the self-attention mechanism is used to adaptively assign weights to different time steps. In the statistical feature extraction part, two statistical features are extracted: the mean value and the trend coefficient. Subsequently, the fusion module is used to fuse features from different source and adaptively assign weights to different source features. Finally, the regression layer is used to predict RUL.

Deep learning features extraction

Before feeding data into the TCN, the self-attention mechanism is used to weight different channels in the channel attention layer. The data sample is expressed as $x = \{x_1, x_2, \ldots, x_t, \ldots, x_{t_{\max}}\}$, x_t represents the channel data at time t, $t_{\max}$ is the maximum time step, $x_t = \{x_{1,t}, x_{2,t}, \ldots, x_{k,t}, \ldots, x_{k_{\max},t}\}$, $x_{k,t}$ represents the value of the kth channel at time t, and $k_{\max}$ is the number of channels.

First, the self-attention mechanism is used to weight different channels. The calculation process is as follows:

1. First, scoring of different channels at time t:

   $$s_t = \varphi(W \cdot x_t + b), \tag{8}$$

   where $\varphi(\bullet)$ is the scoring function, such as sigmoid and linear function, W and b are the weight matrix and bias vector, respectively. The score of different channels at time t can be expressed as $s_t = \{s_{1,t}, s_{2,t}, \ldots, s_{k,t}, \ldots, s_{k_{\max},t_{\max}}\}$.

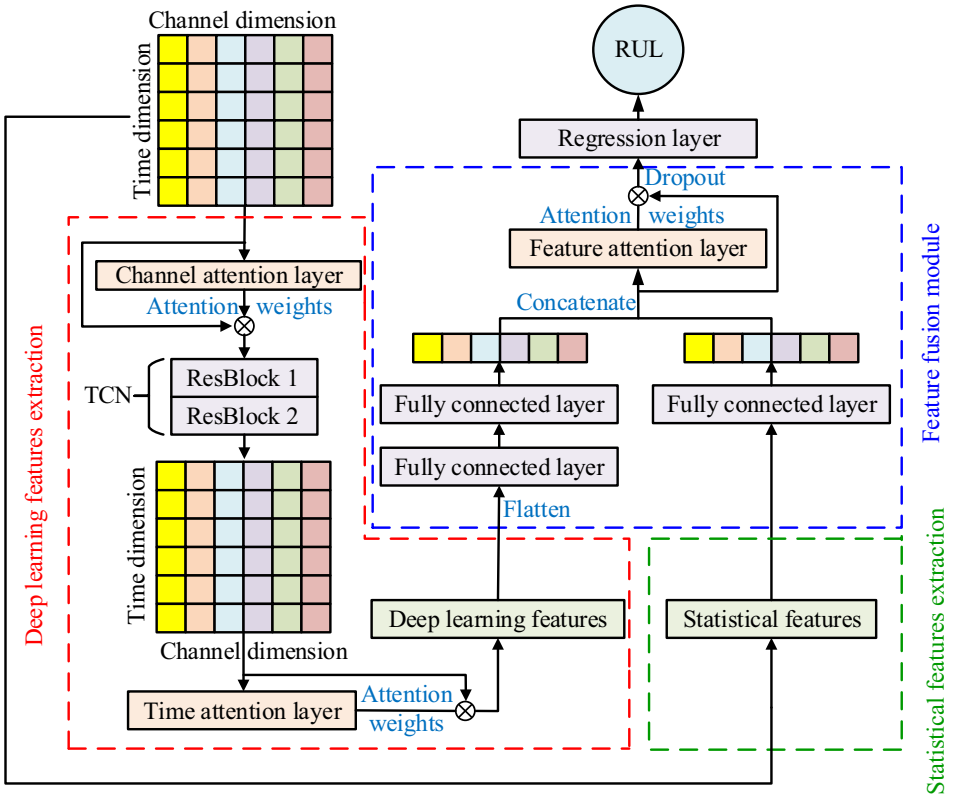

Fig. 3 Framework of the proposed RUL prediction method

2. After obtaining the scores of different channels at time t, the score $s_{k,t}$ corresponding to $x_{k,t}$ can be normalized by the softmax function:

$$\alpha_{k,t} = \text{softmax}(s_{k,t}) = \frac{\exp(s_{k,t})}{\sum_k \exp(s_{k,t})}, \tag{9}$$

where $\alpha_{k,t}$ is the attention weight corresponding to $x_{k,t}$.

3. Take the average of the weights assigned to the kth channel at all time steps, the weight $\overline{\alpha}_k$ corresponding to the kth channel is obtained:

$$\overline{\alpha}_k = \frac{1}{t_{\max}} \sum_t \alpha_{k,t}, \tag{10}$$

4. Finally, the output of the channel attention layer is:

$$\alpha \otimes x = \{\overline{\alpha}_1 x_1, \overline{\alpha}_2 x_2, \ldots, \overline{\alpha}_k x_k, \ldots, \overline{\alpha}_{k_{\max}} x_{k_{\max}}\}^T, \tag{11}$$

where $\alpha = \{\overline{\alpha}_1, \overline{\alpha}_2, \ldots, \overline{\alpha}_{k_{\max}}\}$.

By assigning corresponding weights to different channels, the contribution of channels with richer degradation information is enhanced, while the contribution of insensitive channels is weakened.

Then, the TCN is used to extract the deep learning features. The TCN used in this paper contains two residual blocks.

Table 1 Parameters of TCN

	Layers	Parameters
Residual block 1	1D convolution layer	k: 64, kernel size: 5, stride: 1, d: 1
	Activation function	ReLU
	1D convolution layer	k: 64, kernel size: 5, stride: 1, d: 1
	Activation function	ReLU
Residual block 2	1D convolution layer	k: 64, kernel size: 5, stride: 1, d: 2
	Activation function	ReLU
	1D convolution layer	k: 64, kernel size: 5, stride: 1, d: 2
	Activation function	ReLU

Each residual block is composed of two dilated causal convolution layers, and the input of the residual block is connected with the output of the last layer through the residual connection. The 1-D dilated causal convolution kernel performs on the time dimension of multi-channel data to extract temporal features. The output and input size of the TCN is the same. Due to its shared parameter convolution kernel, the number of parameters and the training time can be greatly reduced. The parameters of TCN used in this paper are experimentally determined and the details of TCN are shown in Table 1.

After obtaining the deep learning features extracted by TCN, the self-attention mechanism is used again to weight different time steps. The output of TCN is expressed as: $x' = \{x'_1, x'_2, \ldots, x'_k, \ldots, x'_{k_{\max}}\}^T$, the data of the kth channel is expressed as: $x'_k = x'_{k,1}, x'_{k,2}, \ldots, x'_{k,t}, \ldots, x'_{k,t_{\max}}\}$. The calculation process is as follows:

1. First, scoring of different time steps:

$$s'_k = \varphi(W \cdot x'_k + b), \tag{12}$$

where $s'_k = \{s'_{k,1}, s'_{k,2}, \ldots, s'_{k,t}, \ldots, s'_{k,t_{\max}}\}$,$s'_{k,t}$ is the score of the kth channel at time t.

2. The score $s'_{k,t}$ can be normalized to attention weight $\beta_{k,t}$ as follows:

$$\beta_{k,t} = \text{softmax}(s'_{k,t}) = \frac{\exp(s'_{k,t})}{\sum_t \exp(s'_{k,t})}, \tag{13}$$

3. Take the average of the weights assigned to all channels at time t, the weight corresponding to the tth time step is calculated as:

$$\overline{\beta}_t = \frac{1}{k_{\max}} \sum_k \beta_{k,t}, \tag{14}$$

4. The output of the time attention layer is:

$$\beta \otimes x' = \{\overline{\beta}_1 x'_1, \overline{\beta}_2 x'_2, \ldots, \overline{\beta}_t x'_i \ldots, \overline{\beta}_{t_{\max}} x'_{t_{\max}}\}, \tag{15}$$

where $\beta = \{\overline{\beta}_1, \overline{\beta}_2, \ldots, \overline{\beta}_{t_{\max}}\}$.

Through the above steps, the deep learning features representation of data is obtained.

Statistical features extraction

Some statistical features contain rich degradation information, such as mean value and trend coefficient, which has been proved to be effective for RUL prediction in [13]. The mean value shows the magnitude of sensory data, and the trend coefficient reflects the degradation rate. In this paper, these two statistical features are extracted and used for RUL prediction. Figure 4 shows an example of these two features. It can be seen, the mean value and trend coefficient increase over time, which well reflects the properties of the raw data.

Feature fusion

After the deep learning features are extracted in the deep learning features extraction part, the deep learning features are fused with the manually extracted statistical features for RUL prediction in the feature fusion module.

The samples for deep learning features are two-dimensional (2-D) matrices, one dimension is the channel dimension and the other is the time dimension. The statistical feature samples are 1-D vectors. The sample shapes of these two features are different and cannot be directly concatenated, so the deep learning features need to be flattened to 1-D firstly. Subsequently, fully connected layers are used to extract more abstract features. After that, two different source features can be concatenated. And the self-attention mechanism is used to adaptively assign weights to the features from different sources.

The process is as follows:

1. Flatten deep learning features by the flatten layer. After that, two fully connected layers are used to extract more abstract features, the output is expressed as $D = \{d_1, d_2, \ldots, d_m\}$. A fully connected layer is used to exact abstract features from statistical features, the output is expressed as $H = \{h_1, h_2, \ldots, h_m\}$. Concatenate two different source features into a new feature set $F = \{D, H\} = \{d_1, d_2, \ldots, d_m, h_1, h_2, \ldots, h_m\} = \{f_1, f_2, \ldots, f_n\}$, where $n = 2\,m$.
2. The self-attention mechanism is used to weight different source features:

$$s_n = \varphi(W \cdot f_n + b), \tag{16}$$

$$\gamma_n = \text{softmax}(s_n) = \frac{\exp(s_n)}{\sum_n \exp(s_n)}, \tag{17}$$

where $\varphi(\cdot)$ is the scoring function, s_n is the score corresponding to the feature f_n, and γ_n is the attention weight corresponding to the feature f_n.

3. The output of the feature fusion module is:

$$F \otimes \gamma = f_1\gamma_1, f_2\gamma_2, \ldots, f_n\gamma_n, \ldots, f_{n_{\max}}\gamma_{n_{\max}}\}, \tag{18}$$

where $\gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_{n_{\max}}\}$.

To prevent over-fitting, dropout is adopted, which is a common regularization method. During the model training

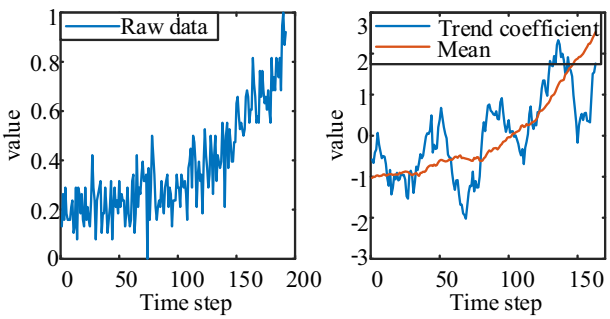

Fig. 4 An example of two statistical features

process, some neurons are randomly hidden, so these neurons will not make an effect. During the testing process, all neurons are activated. In this paper, the dropout ratio is set to 0.2. Finally, the mapping relationship between fusion features and RUL is established through the regression layer.

Since RUL prediction is a typical regression problem, the mean square error (MSE) is selected as the loss function. The Adam optimizer is used to modify the model parameters. To achieve the best effect, the learning rate decay strategy is adopted. The initial learning rate is set to 0.001, which is decayed to 0.0001.

Experimental study and analysis

Dataset, evaluation metrics, experimental results, analysis, and discussion are described specifically in this section. The training and testing process is implemented using Keras running on top of TensorFlow. The computer is configured with an Intel(R)Xeon(R)Gold 6136 CPU, 16 GB RAM, and windows7 64-bit operating system.

Dataset

The widely used NASA turbofan engine dataset [20] is used to evaluate the proposed method. The dataset can be found in [19]. This dataset is generated by C-MAPSS (Commercial Modular Aero-Propulsion System Simulation) software to simulate the degradation process of turbofan engines. This dataset describes the degradation process of turbofan engine. The engine consists of fan, high-pressure turbine (HPT), high-pressure compressor (HPC), low pressure compressor (LPC), low pressure turbine (LPT), nozzle, and combustor as shown in Fig. 5. Twenty-one sensors are deployed at different locations to monitor the condition of the engine. For detailed information on engine modules and channel descriptions, please refer to the literature [20].

This dataset contains four sub-datasets, which are respectively denoted as FD001, FD002, FD003, and FD004. The

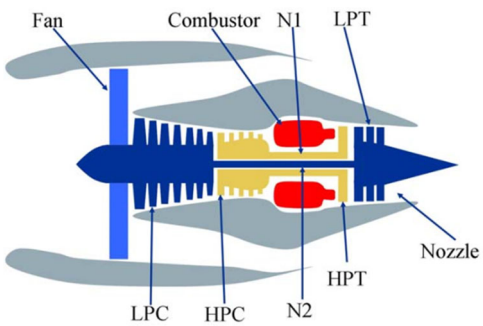

Fig. 5 Simplified diagram of engine simulated in C-MAPSS [20]

Table 2 Details of the turbofan engine dataset

	C-MAPSS			
	FD001	FD002	FD003	FD004
Train numbers	100	260	100	249
Test numbers	100	259	100	248
Operation mode	1	6	1	6
Fault mode	1	1	2	2

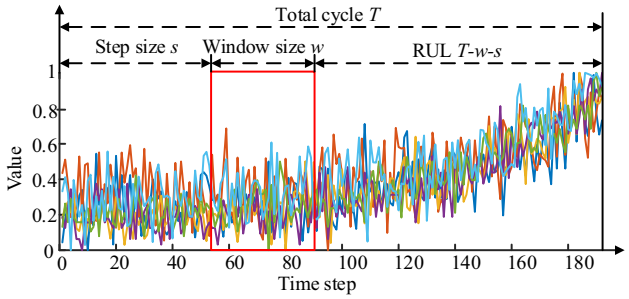

Fig. 6 An example of sliding window to split data

operating conditions and failure modes of each sub-dataset are different. Each sub-dataset consists of three parts: training set, testing set, and RUL label. Each training set and testing set contain 26 columns of data. The first two columns are the engine ID and the number of operating cycles. The next three columns are the three operating parameters: flight altitude, Mach number, and throttle resolver angle. The remaining 21 columns are different channel data. The training set has complete run-to-failure data, while the testing set only provides part of the full life cycle data. The details of the dataset are shown in Table 2.

Data preprocessing

Samples creation

There are dependencies between different time sequences, which are crucial to the problem of sequential processing. To capture this dependence, a sliding window is used to split data along the time dimension [1]. As shown in Fig. 6, a time window with length w is used to split data to obtain training samples. The RUL corresponding to the tth sample is T–w–t, and T is the total cycle of the engine.

Data normalization

Some channels did not change during the whole cycle, indicating these channels are not related to the degradation of engines. The smaller the variance value of the channel data, the less the change. Finally, the channel 1, 5, 6, 10, 16, 18,

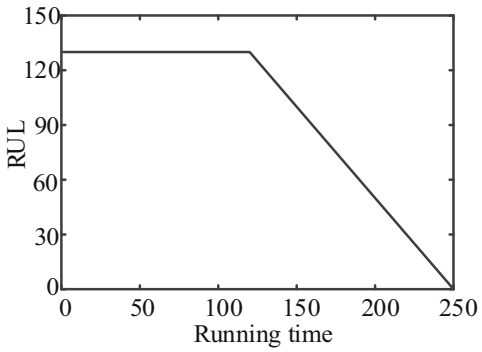

Fig. 7 Piece-wise linear RUL function

and 19 are removed from four sub-datasets. Meanwhile, since the three operating conditions also changed during the engine operation, they are also taken as part of the inputs.

The range of each channel is different. To eliminate its influence on the prediction results, the data of each channel is normalized to the range of [0,1] by Eq. (19) [24]:

$$\hat{x}_{k,t} = \frac{x_{k,t} - \min(x_k)}{\max(x_k) - \min(x_k)} \tag{19}$$

where $\max(x_k)$, $\min(x_k)$ is the maximum and minimum values of the kth channel, respectively.

Besides, the extracted statistical features are standardized.

RUL label settings

In the healthy stage, the turbofan engine runs stably and the degradation is not obvious. Therefore, during the health phase, RUL is set to a constant value. When the fault occurs, the performance of the engine begins to degrade. As the fault becomes more serious, the condition worsens until the RUL drops to 0 and the engine fails completely. In this paper, referring to the literature [9, 24], the RUL label is set as a piece-wise linear function, as shown in Fig. 7. The threshold value is set to 130. The RUL label is set to 130 when the true RUL is greater than 130. For samples with RUL less than 130, the label is set to the corresponding true RUL.

Evaluation metrics

To evaluate the performance of the proposed method, two commonly used evaluation metrics are adopted: scoring function [1] and root mean square error (RMSE) [1].

The scoring function is defined as:

$$\text{Score} = \begin{cases} \sum_{i=1}^{N} (e^{-\frac{\hat{r}_i - r_i}{13}} - 1), & \text{when } \hat{r}_i < r_i \\ \sum_{i=1}^{N} (e^{\frac{\hat{r}_i - r_i}{10}} - 1), & \text{when } \hat{r}_i \geq r_i \end{cases}, \tag{20}$$

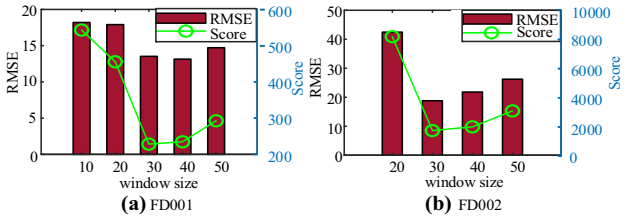

Fig. 8 Results of the proposed method with different window sizes on the two sub-datasets

where $\hat{r}_i$ is the predicted value, r_i is the true value, and Score is the score value.

The scoring function imposes different levels of penalties for early and late predictions. For the case of overestimating RUL ($\hat{r}_i \geq r_i$), the penalty is higher than the case of underestimating RUL ($\hat{r}_i < r_i$). This is because in reality, the consequences of the late prediction are more severe than the early prediction. This asymmetric preference is also in line with the aviation industry's risk aversion attitude towards engine failures. However, relying only on the scoring function sometimes is incomplete, because the appearance of outliers (the difference between the predicted value and the true value is too large) will affect the overall evaluation of the scoring function. Therefore, it needs to be used together with RMSE for evaluation. RMSE can reflect the global error between the predicted value and the true value, and it is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{r}_i - r_i)^2} \tag{21}$$

The smaller the value of Score and RMSE, the better the prediction performance of the model.

Experimental implementation and results

Six experiments are conducted to investigate the performance of the proposed method. The first experiment investigates the impact of different time windows on RUL prediction. The second experiment investigates the impact of different Rectifiers on RUL prediction. The third experiment investigates the impact of different source features on RUL prediction. The fourth experiment investigates the effect of attention mechanism settings. The fifth experiment compares the performance of the proposed method with other RUL prediction methods. The sixth experiment compares the complexity of different methods.

1. The impact of different time windows on RUL prediction: The time window size directly affects the model. Studies have shown that a longer time window contains

Table 3 Performance of models using different Rectifiers

Rectifiers	RMSE		Score		Time cost (s)
	Mean	STD	Mean	STD	
ELU	13.57	0.73	257.26	30.02	33.64
LeakyReLU	13.11	0.44	232.19	14.88	38.33
PReLU	13.25	0.65	237.54	26.71	39.71
ReLU	13.25	0.40	235.52	17.18	31.16

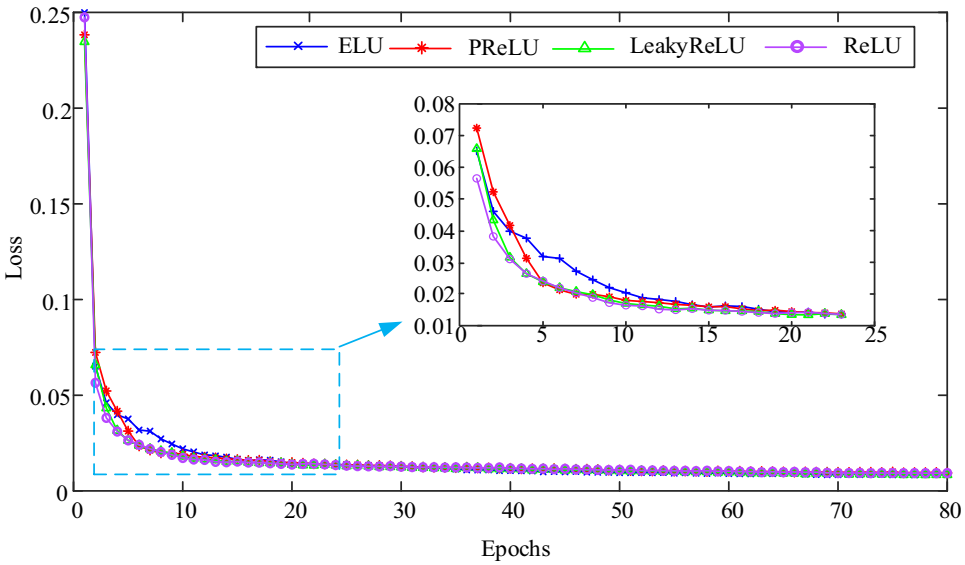

Fig. 9 The loss of models using different Rectifiers

more useful information. However, an excessively long time window will increase the computational complexity and affect the performance of the model. Therefore, choosing an appropriate time window is very important. To investigate the impact of the time window sizes on RUL prediction, different size time windows are used to create data samples. This experiment is conducted on sub-datasets FD001 and FD002. The experimental results are shown in Fig. 8. Fig. 8 shows the changes in the performance of the model under different time windows. The x-axis represents different time windows, and the y-axis is the RMSE and Score obtained by the model at different window sizes. It can be seen that when the time window size is less than 30, as the window size increases, the performance of the model improves. This is because the larger window contains richer degradation information. But when the window size increases to more than 30, the performance of the model begins to degrade. This means that when the window size exceeds 30, it will instead have a negative effect on the RUL prediction. According to the experimental results, the time window size is set to 30, and the model performs best.

2. The impact of different Rectifiers on RUL prediction: Referring to the literature [4], the proposed method uses ReLU as the activation function. To evaluate the impact of different Rectifiers on RUL prediction, the performance of LeakyReLU, PReLU, ELU, and ReLU is compared on FD001. Each model is trained 10 times to eliminate random errors. The experimental results are shown in Table 3. In the table, Mean represents the mean value of 10 training results, STD represents the standard deviation of 10 training results. Time cost represents the average time cost of each iteration. As shown in Table 3, the performance of the four models using different rectifiers is similar. The model using ReLU performs better than the model using ELU and PReLU. Compared with the model using ReLU, the RMSE of the LeakyReLU model is reduced by 1.05%, and the Score is reduced by 1.41%. However, the time cost of each iteration is increased by 23.01%. In addition, we visualized the training loss of models, as shown in Fig. 9. It can be seen from Fig. 9 that the loss of the model using ReLU decreases faster. Therefore, after consideration, we use ReLU as the activation function.
3. The impact of features on RUL prediction: this paper proposes a feature fusion method. To verify its effectiveness for RUL prediction, the model is trained using different features: the model trained using only deep learning features, the model trained using only statistical features, the model trained using two different source features.

Table 4 Results of models trained using different features

Features	FD001		FD002	
	RMSE	Score	RMSE	Score
Deep learning features				
Mean	14.59	269.08	19.79	2609.90
STD	0.54	21.31	1.37	1212.39
Statistical features				
Mean	14.81	440.94	23.48	4549.80
STD	0.76	11.57	1.40	1051.49
Both two features				
Mean	13.25	235.52	19.57	1655.04
STD	0.40	17.18	1.13	174.80

The parameters are the same except for the features used for RUL prediction. Each model is trained 10 times to eliminate random errors. The experimental results are shown in Table 4. It can be seen from the table that the model trained using both two features performs better, which verifies the effectiveness of the two features. In other words, the proposed method makes full use of deep learning features and statistical features to obtain more degradation information, which helps to improve the accuracy of RUL prediction.

4. The impact of attention mechanism settings: the proposed method uses the self-attention mechanism to weight different channels, different time steps, and different source features. This experiment is to verify the effectiveness of each attention mechanism layer. Four sets of comparative experiments are carried out: no attention mechanism, the model with channel attention layer, the model with channel and time attention layer, the proposed model. The parameters are the same except for the settings of the attention mechanism. Each model is trained 10 times. The experimental results are shown in Table 5. According to the experimental results, the three models that use the attention mechanism perform better than the model without the attention mechanism. It shows that adaptively assigning feature weights through the attention mechanism can effectively improve accuracy. In addition, with the increase of attention mechanism layers, the performance of the model gradually improves. This is because by using the attention mechanism weighting different features, the consideration for the differences of different features is more comprehensive. The RMSE and Score of the proposed method are the lowest, achieves the best prediction performance by weighting different channels, different time steps, and different source features.
5. Comparison of different methods: The compared models include shallow models such as SVM, decision tree regression (DTR), random forest (RF), and deep learning models such as deep convolution neural network (DCNN), multi-layer attention convolution neural network (MA-CNN). Among them, MA-CNN is the model that combines the multi-layer attention mechanism and CNN. The experiment is conducted on four sub-datasets FD001, FD002, FD003, and FD004, and each model is trained 10 times. The experimental results are shown in Tables 6 and 7. From the table, the average RMSE of the three deep learning models is 13.94% lower than the RMSE of shallow models, and the average Score is 70.8% lower. It can be seen that the deep learning models perform better than shallow models. And compared with the average values of RMSE and Score of comparative models, the proposed method in this paper reduces RMSE by 14.19% and the Score value by 68.00%, showing its superiority in RUL prediction.

Table 5 Results of models with different attention mechanism settings

Attention settings	FD001		FD002	
	RMSE	Score	RMSE	Score
Without attention mechanism				
Mean	14.99	329.55	24.13	4026.62
STD	0.67	40.46	1.86	1030.72
Model with channel attention layer				
Mean	13.05	255.44	23.10	3148.57
STD	0.62	25.21	1.23	496.48
Model with channel and time attention layer				
Mean	13.17	257.00	20.95	2742.30
STD	0.62	29.44	2.16	1310.78
Proposed model				
Mean	13.25	235.52	19.57	1655.04
STD	0.40	17.18	1.13	174.80

Table 6 RMSE performance of different methods

	FD001		FD002		FD003		FD004	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD
SVR	14.48	–	18.66	–	14.57	–	21.74	–
DTR	18.49	0.56	31.50	0.56	20.55	0.60	35.40	0.49
RF	12.89	0.13	22.18	0.15	14.28	0.14	26.42	0.19
DCNN	14.09	0.59	24.29	1.17	14.85	0.69	26.85	1.25
MA-CNN	13.26	0.61	19.53	1.02	12.97	0.41	22.65	1.82
Proposed	13.25	0.40	19.57	1.13	13.43	0.86	21.69	1.27

Table 7 Score performance of different methods

	FD001		FD002		FD003		FD004	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD
SVR	363.43	–	1579.69	–	470.89	–	3690.72	–
DTR	966.07	301.84	32,411.90	4967.96	1771.17	133.55	41,920.40	4192.43
RF	268.42	10.89	4071.24	179.25	346.04	9.44	16,272.46	1608.33
DCNN	270.23	12.82	3496.08	513.60	367.21	53.23	5968.61	1451.64
MA-CNN	261.55	17.48	2056.38	443.28	297.60	21.92	3228.87	717.70
Proposed	235.52	17.18	1655.04	174.80	239.02	24.33	2414.69	359.75

Table 8 The complexity comparison

Model	FLOPs	Parameters	Time cost (s)	Time complexity
Model 1	3,319,511	80,307	23.35	$O\left(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f + \sum_{r=1}^{R} L_r C_r\right)$
Model 2	3,338,711	80,579	26.38	$O\left(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f + \sum_{r=1}^{R} L_r C_r + N_{ch}^2 L_{ch}\right)$
Model 3	3,371,351	81,509	29.20	$O\left(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f + \sum_{r=1}^{R} L_r C_r + N_{ch}^2 L_{ch} + L_t^2 N_t\right)$
Model 4	1,280,729	58,715	20.57	$O\left(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f\right)$
Model 5	1,307,609	62,199	26.93	$O\left(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f + N_{ch}^2 L_{ch} + L_t^2 N_t\right)$
Model 6	3,372,291	81,929	31.16	$O\left(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f + \sum_{r=1}^{R} L_r C_r + N_{ch}^2 L_{ch} + L_t^2 N_t + N_f^2\right)$

6. Complexity comparison of different methods: This paper proposes a new RUL prediction method based on MLSA and TCN. To evaluate the complexity of the proposed method. This paper uses FLOPs (i.e. the number of floating-point multiplication-adds), parameter size and time cost to evaluate the complexity of the model. Comparison models include TCN (Model1), TCN with channel attention layer (Model 2), TCN with channel and time attention layer (Model 3), DCNN (Model 4), MA-CNN (Model5), and the proposed method (Model 6). The experimental results are shown in Table 8 and Fig. 10.

Table 8 shows the complexity comparison of different methods. The time complexity of DCNN mainly comes from the 1D convolution layer and the fully connected layer. The time complexity of the convolution layer is $O(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c)$, where C is the number of convolution layers, L_c is the output feature length of the cth convolution layer,K_c is the kernel size of the cth convolution layer, N_{c-1} is the number of output channels of the (c-1)-th convolution layer, and N_c is the number of output channels of the cth convolution layer. The time complexity of the fully connected layer is $O(\sum_{f=1}^{F} S_{f-1} S_f)$, where F is the number of fully connected layers,S_{f-1} is the output size of the $(f-1)$th fully connected layer,S_f is the output

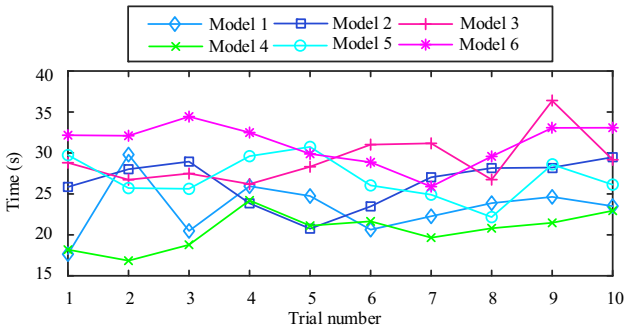

Fig. 10 Detailed results of time cost with different methods for 10 experiments

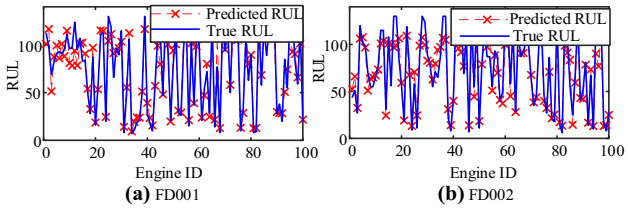

Fig. 11 Results on FD001 and FD002

size of the fth fully connected layer. So, the time complexity of DCNN is $O(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f)$. Besides the 1D convolution layer and the fully connected layer, TCN also uses residual connection. The time complexity of the residual connection is $O(\sum_{r=1}^{R} L_r N_r)$, where R is the number of residual blocks,L_r is the output feature length of the rth residual block,N_r is the number of output channels of the r-th residual block. So, the time complexity of TCN is $O(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f + \sum_{r=1}^{R} L_r N_r)$. The proposed method integrates TCN and MLSA to predict RUL. The time complexity of the channel attention layer is $O(N_{\text{ch}}^2 L_{\text{ch}})$, where N_{ch} is the number of output channels of the channel attention layer,L_{ch} is the output feature length of the channel attention layer. The time complexity of the time attention layer is $O(L_t^2 N_t)$, where L_t is the output feature length of the time attention layer,N_t is the number of output channels of the time attention layer. The time complexity of the feature attention layer is $O(N_f^2)$, where N_f is the output feature number of the feature attention layer. Therefore, the time complexity of the proposed method is $O(\sum_{c=1}^{C} L_c K_c N_{c-1} N_c + \sum_{f=1}^{F} S_{f-1} S_f + \sum_{r=1}^{R} L_r C_r + N_{ch}^2 L_{ch} + L_t^2 N_t + N_f^2)$. The time complexity of Model 2, Model 3, and Model 5 are described in Table 8. To further evaluate the computational complexity of different methods, this paper also uses multiple metrics to evaluate the models. The complexity of models under different metrics is described in Table 8. The detailed results of time cost of different methods are shown in Fig. 10. By comparing Model 1, Model 2, Model 3, and Model 6, it can be found that after using the channel attention layer, the required FLOPs increase by 0.58%. After using the time attention layer, the required FLOPs increase by 0.98%. After using the feature attention layer, the required FLOPs increased by 0.03%. It can be seen that with the use of the attention layer, the computational complexity of the model increases. In addition, the proposed method requires 3.37 MFLOPs, 81.92 k parameters, and takes 31.16 s for each iteration. Compared with MA-CNN and DCNN, the proposed method requires more complexity and training time. However, with the rapid development of computers, the cost gap of this method is narrow.

Analysis

The comparison between the true RUL and the predicted RUL is shown in Figs. 11, 12, 13. It can be seen that the predicted RUL is very close to the real RUL, which proves the feasibility of the proposed method for RUL prediction. FD001 and FD003 have the same sample numbers, but the RMSE and Score of the model on FD001 are lower than on FD003. The main reason is that FD003 contains 6 modes of operation, while FD001 has only one mode, which makes prediction more difficult. FD002 and FD004 have similar sample numbers, but the RMSE and Score of the model on FD002 are lower than on FD004. The main reason is that there are two failure modes in FD004, while there is only one failure mode in FD002, so the former is more difficult to predict.

To understand the contribution differences of different features to RUL prediction, a sample of FD001 is selected to visualize the process of prediction. Figure 14a presents the raw data. Figure 14b shows the weights assigned to the different channels by the self-attention mechanism. It can be seen that the weight assigned to the 12th channel is the highest, indicating that the degradation information in the 12th channel is the most important for RUL prediction.

Figure 15a presents the output of the channel attention layer, which is calculated by multiplying the weights assigned to the different channels with the raw data. The output of this layer is used as the input of TCN. Figure 15b shows the output of TCN. It can be seen that the features extracted by TCN are smoother compared to the input data.

Figure 16a presents the weights of different time steps assigned by the self-attention mechanism. From the figure, the weight assigned to the last time step is the highest. This is because there is more degradation information embedded in the later period, so a higher weight is assigned to the last time step to enhance its contribution to RUL prediction. Figure 16b shows the output of the channel attention layer. It is calculated by multiplying the weights assigned to the different time steps with the output of the TCN.

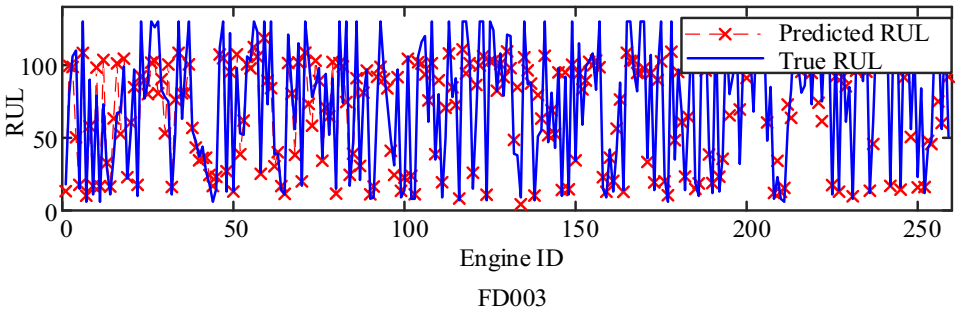

Fig. 12 Results on FD003

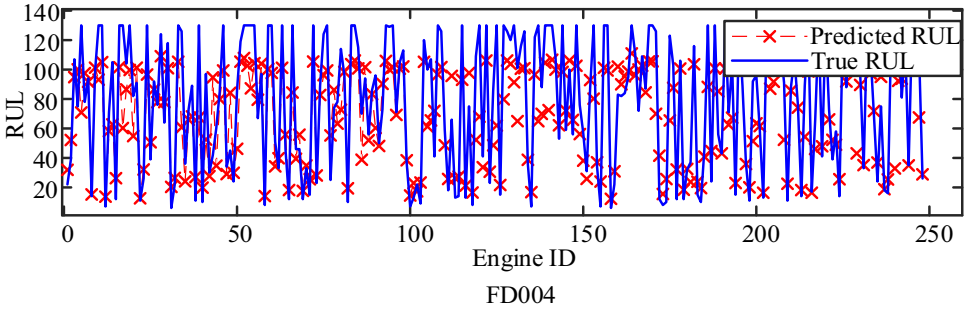

Fig. 13 Results on FD004

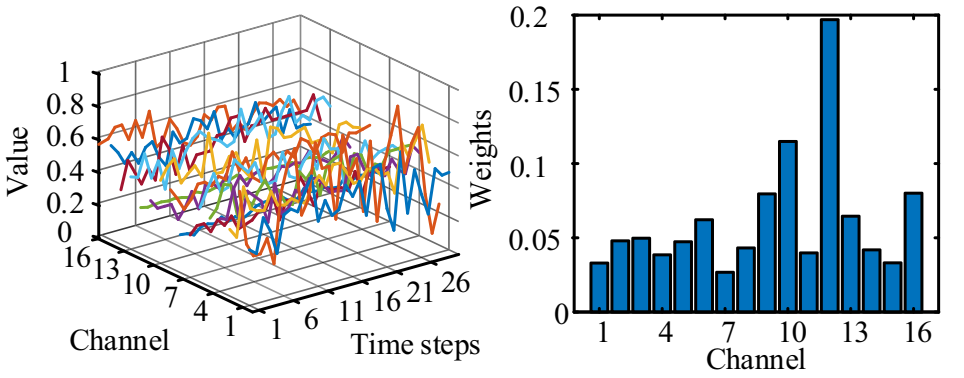

Fig. 14 Raw data and weights of different channel

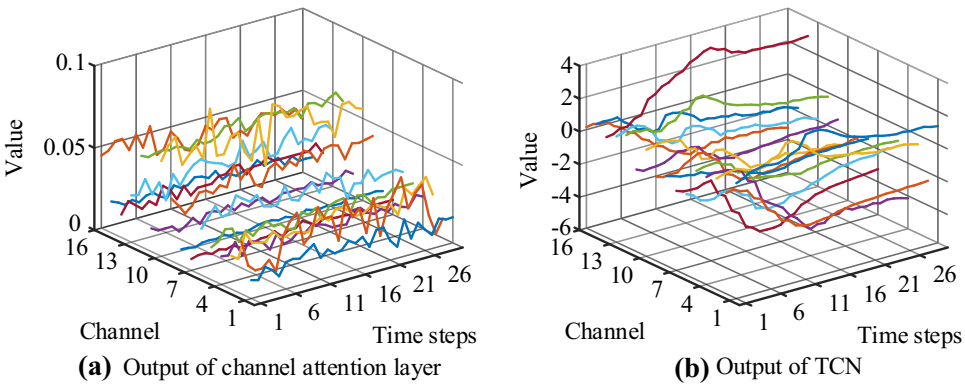

Fig. 15 Output of channel attention layer and output of TCN

Figure 17 shows the weights assigned to the different source features by the self-attention mechanism. The red bars are the weights assigned to the deep learning features, and the blue bars are the weights assigned to the statistical features. Both deep learning features and statistical features are assigned high weights, which indicates these two features

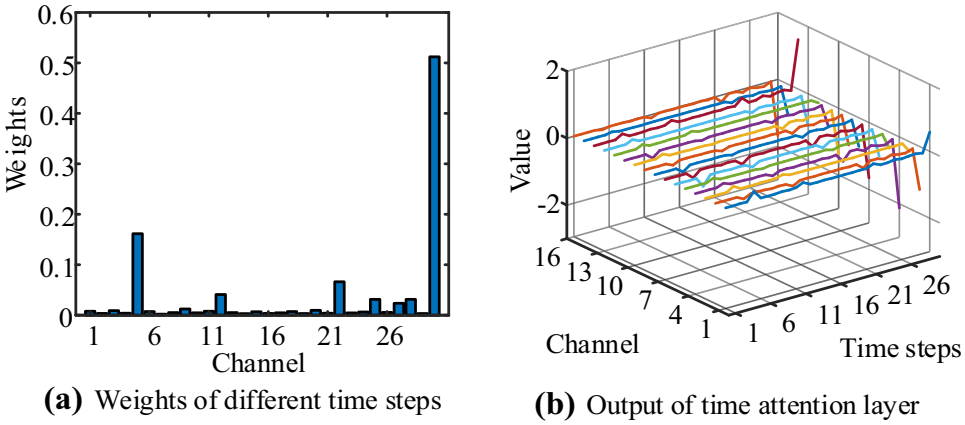

Fig. 16 Weights of different time steps and output of channel attention layer

(a) Weights of different time steps

(b) Output of time attention layer

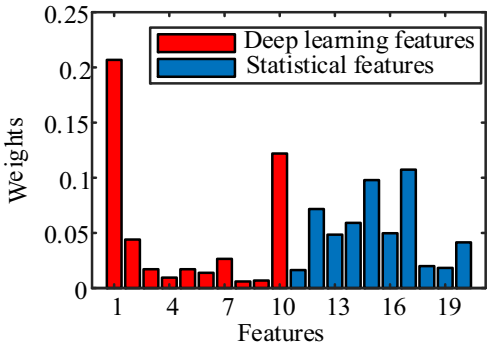

Fig. 17 Weights of different source features

both play an important role in RUL prediction. The weights assigned to each feature are different, indicating that different features contribute differently to the RUL prediction. The larger the weight, the greater its contribution to RUL prediction.

Discussion

In summary, compared with other prediction methods, the proposed method can achieve higher prediction accuracy and stable performance. This paper uses RMSE and the Scoring function as evaluation metrics. We calculate the mean and standard deviation of the two metrics to analyze the accuracy and stability of different methods. Experiment (i) investigates the impact of different time windows on RUL prediction. The results show that a shorter time window contains less useful information, and a too long time window contains redundant information, which will have a negative impact on the result. Experiment (ii) investigates the impact of different Rectifiers on RUL prediction. The results show that the ReLU model performs better on RUL prediction, and the training loss decreases faster. Experiment (iii) investigates the impact of features on RUL prediction. The results show that the proposed method can effectively fuse deep learning features and statistical features. The degradation information is more comprehensive, which helps to improve the accuracy of RUL prediction. Experiment (iv) investigates the impact of attention mechanism settings. The results show that the MLSA can effectively improve prediction performance. Experiment (v) compares the proposed method with other methods commonly used in RUL prediction and verifies the superiority of the proposed. Experiment (vi) compares the complexity of different methods. The results show that the proposed method requires more complexity than other methods. And the proposed method has more parameters and requires longer training time. Meanwhile, the model prediction process is visualized, which further explains the internal mechanism of the proposed method. The above experimental results show that the proposed method can extract high-quality degradation features, thereby achieving accurate RUL prediction. However, the proposed method requires more complexity than the existing models.

Conclusion

This paper proposes a method for RUL prediction based on multi-layer self-attention (MLSA) and temporal convolution network (TCN). First, the self-attention mechanism is used to adaptively assign weights to different channels to enhance the contribution of important channels to RUL prediction. Then we use TCN to extract deep learning features. To weight the contributions of different time steps, the self-attention mechanism is used again to adaptively assign weights to different time steps. Subsequently, we extract the two statistical features, mean and trend coefficient, and concatenate them with the deep learning features. Consider the contribution differences of different source features to RUL prediction, the self-attention mechanism is used again to weight different source features. Finally, the RUL is obtained through the regression layer. To evaluate the performance

of the proposed method, four comparative experiments are conducted on the turbofan dataset. The impacts of time window size, features, and attention mechanism settings on RUL predictions are investigated. And compared with the average value of RMSE and Score of comparative models, the proposed method reduces RMSE by 14.19% and the Score value by 68.00%, which verifies the superiority of the proposed method. In future research, we will focus on reducing the complexity of the proposed method.

Acknowledgements This paper is supported by the National Natural Science Foundation of China and Civil Aviation Administration of China Joint Funded Project (Grant No. U1733108). All the authors would like to thank them.

Author contributions ZS: resources, funding acquisition, project administration, supervision, writing—review and editing. BZ: methodology, software, writing-original draft. WL: data acquisition, investigation. SQ: formal analysis. JZ: data curation.

Funding This work was financially supported by the Civil Aviation Joint Funds of the National Natural Science Foundation of China (Grant No. U1733108).

Availability of data and material The datasets generated and analyzed during the current study are available from the corresponding author on reasonable request.

Code availability The code required to reproduce these findings cannot be shared at this time as the data also forms part of an ongoing study.

Declarations

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

1. Babu G, Zhao P, Li X (2016) Deep convolutional neural network based regression approach for estimation of remaining useful life. In: International conference on database systems for advanced applications.
2. Bai S, Kolter J Z, Koltun V (2018) An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. arXiv:1803.01271v2
3. Chen Y, Peng G, Zhu Z, Li S (2020) A novel deep learning method based on attention mechanism for bearing remaining useful life prediction. Appl Soft Comput 86:105919. https://doi.org/10.1016/j.asoc.2019.105919
4. Chen Z, Wu M, Zhao R, Guretno F, Yan R, Li X (2021) Machine remaining useful life prediction via an attention-based deep learning approach. IEEE Trans Ind Electron 68(3):2521–2531. https://doi.org/10.1109/tie.2020.2972443
5. Cheng C, Ma G, Zhang Y, Sun M, Teng F, Ding H, Yuan Y (2020) A deep learning-based remaining useful life prediction approach for bearings. IEEE/ASME Trans Mechatron 25(3):1243–1254. https://doi.org/10.1109/tmech.2020.2971503
6. Dalzochio J, Kunst R, Pignaton E, Binotto A, Sanyal S, Favilla J, Barbosa J (2020) Machine learning and reasoning for predictive maintenance in Industry 40: current status and challenges. Comput Ind. https://doi.org/10.1016/j.compind.2020.103298
7. Du W, Wang Y, Qiao Y (2018) Recurrent spatial-temporal attention network for action recognition in videos. IEEE Trans Image Process 27(3):1347–1360. https://doi.org/10.1109/TIP.2017.2778563
8. Florian E, Sgarbossa F, Zennaro I (2021) Machine learning-based predictive maintenance: a cost-oriented model for implementation. Int J Prod Econ. https://doi.org/10.1016/j.ijpe.2021.108114
9. Hou M, Pi D, Li B (2020) Similarity-based deep learning approach for remaining useful life prediction. Measurement. https://doi.org/10.1016/j.measurement.2020.107788
10. Khelif R, Chebel-Morello B, Malinowski S, Laajili E, Fnaiech F, Zerhouni N (2017) Direct remaining useful life estimation based on support vector regression. IEEE Trans Ind Electron 64(3):2276–2285. https://doi.org/10.1109/tie.2016.2623260
11. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444. https://doi.org/10.1038/nature14539
12. Lei Y, Li N, Guo L, Li N, Yan T, Lin J (2018) Machinery health prognostics: a systematic review from data acquisition to RUL prediction. Mech Syst Signal Process 104:799–834. https://doi.org/10.1016/j.ymssp.2017.11.016
13. Li H, Zhao W, Zhang Y, Zio E (2020) Remaining useful life prediction using multi-scale deep convolutional neural network. Appl Soft Comput 89:106113. https://doi.org/10.1016/j.asoc.2020.106113
14. Li W, Shang Z, Gao M, Qian S, Zhang B, Zhang J (2021) A novel deep autoencoder and hyperparametric adaptive learning for imbalance intelligent fault diagnosis of rotating machinery. Eng Appl Artif Intell. https://doi.org/10.1016/j.engappai.2021.104279
15. Li Y, Shan X, Zhao W, Wang G (2019) A LS-SVM based approach for turbine engines prognostics using sensor data. In: 2019 IEEE International Conference on Industrial Technology
16. Lin M, Zeng X, Wu J (2021) State of health estimation of lithium-ion battery based on an adaptive tunable hybrid radial basis function network. J Power Sources. https://doi.org/10.1016/j.jpowsour.2021.230063
17. Niu Z, Zhong G, Yu H (2021) A review on the attention mechanism of deep learning. Neurocomputing 452:48–62. https://doi.org/10.1016/j.neucom.2021.03.091
18. Ren L, Sun Y, Wang H, Zhang L (2018) Prediction of bearing remaining useful life with deep convolution neural

network. IEEE Access 6:13041–13049. https://doi.org/10.1109/access.2018.2804930
19. Saxena A, Goebel K (2008) Turbofan engine degradation simulation data set. NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA
20. Saxena A, Goebel K, Simon D, Eklund N (2008) Damage propagation modeling for aircraft engine run-to-failure simulation. In: 2008 International Conference on Prognostics and Health Management.
21. Shang Z, Liao X, Geng R, Gao M, Liu X (2018) Fault diagnosis method of rolling bearing based on deep belief network. J Mech Sci Technol 32(11):5139–5145. https://doi.org/10.1007/s12206-018-1012-0
22. Shang Z, Li W, Gao M, Liu X, Yu Y (2021) An intelligent fault diagnosis method of multi-scale deep feature fusion based on information entropy. Chin J Mech Eng. https://doi.org/10.1186/s10033-021-00580-5
23. Shang Z, Liu X, Liao X, Geng R, Gao M, Yun J (2019) Rolling bearing fault diagnosis method based on EEMD and GBDBN. Int J Perform Eng 15(1):230–240. https://doi.org/10.23940/ijpe.19.01.p23.230240
24. Song Y, Gao S, Li Y, Jia L, Li Q, Pang F (2020) Distributed attention-based temporal convolutional network for remaining useful life prediction. IEEE Internet Things J 8(12):9594–9602. https://doi.org/10.1109/jiot.2020.3004452
25. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser L, Polosukhin I (2017) Attention Is all you need. arXiv:1706.03762v5
26. Xia H, Luo Y, Liu Y (2021) Attention neural collaboration filtering based on GRU for recommender systems. Complex Intell Syst 7(3):1367–1379. https://doi.org/10.1007/s40747-021-00274-4
27. Xiao Z, Xu X, Xing H, Luo S, Dai P, Zhan D (2021) RTFN: a robust temporal feature network for time series classification. Inf Sci 571:65–86. https://doi.org/10.1016/j.ins.2021.04.053
28. Xue Q, Shen S, Li G, Zhang Y, Chen Z, Liu Y (2020) Remaining useful life prediction for lithium-ion batteries based on capacity estimation and Box-Cox transformation. IEEE Trans Veh Technol 69(12):14765–14779. https://doi.org/10.1109/tvt.2020.3039553
29. Yan B, Ma X, Huang G, Zhao Y (2021) Two-stage physics-based Wiener process models for online RUL prediction in field vibration data. Mech Syst Signal Process 152:107378. https://doi.org/10.1016/j.ymssp.2020.107378
30. Yang B, Tu Z, Wong D F, Meng F, Chao L S, Zhang T (2018) Modeling localness for self-attention networks. arXiv:1810.10182
31. Zan T, Liu Z, Wang H, Wang M, Gao X, Pang Z (2021) Prediction of performance deterioration of rolling bearing based on JADE and PSO-SVM. Proc Inst Mech Eng Part C 235(9):1684–1697. https://doi.org/10.1177/0954406220951209
32. Zhang B, Xiong D, Su J (2020) Neural machine translation with deep attention. IEEE Trans Pattern Anal Mach Intell 42(1):154–163. https://doi.org/10.1109/tpami.2018.2876404
33. Zhang C, Lim P, Qin AK, Tan K (2017) Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. IEEE Trans Neural Netw Learn Syst 28(10):2306–2318. https://doi.org/10.1109/TNNLS.2016.2582798