Measurement 182 (2021) 109685

Contents lists available at ScienceDirect

Measurement

journal homepage: www.elsevier.com/locate/measurement

Remaining useful life prediction for multi-sensor systems using a novel end-to-end deep-learning method

Yuyu Zhao, Yuxiao Wang *

College of Electronic Information and Automation, Civil Aviation University of China, Tianjin 300300, China

ARTICLE INFO

Keywords:
Multi-sensor data
Remaining useful life
Deep learning
Attention mechanism
End-to-end model

ABSTRACT

Remaining useful life (RUL) prediction plays a crucial role in ensuring reliability and safety of modern engineering systems. For complicated systems, the indirect manner of the conventional RUL prediction approaches restricts their universality and accuracy. The challenge to realize accurate RUL estimation consists in the direct exploration of the potential relationship between the RUL and the numerous data from multiple monitoring sensors. Motivated by this fact, a novel end-to-end RUL prediction method is proposed based on a deep learning model in this paper. The long short-term memory (LSTM) encoder-decoder is employed as the main frame of the model to deal with multivariate time series data. Then a two-stage attention mechanism is developed to realize adaptive extraction and evaluation of the input features and temporal correlation. On this basis, the RUL prediction is obtained by a multilayer perceptron. The proposed model can selectively focus on the critical information without any prior knowledge, which is of great significance to enhance the RUL prediction accuracy. The effectiveness and superiority of the proposed method is experimentally validated through a turbofan engine dataset and compared with the state-of-the-art methods.

1. Introduction

Reliability of engineering systems is extremely important in industrial applications. Nowadays, the modern engineering systems tend to be highly integrated and complicated. A small malfunction or failure of any component may cause severe damage to the whole system, raise financial loss and even bring security risks to the environment and users. Due to these problems and the rapid development of sensor technology, prognostics and health management (PHM) technique has been developed and is now widely applied to ensure the reliability and efficiency of engineering systems [1–3]. PHM technique contributes to perform condition-based maintenance rather than follow the traditional practice of scheduled maintenance, the key process of which is remaining useful life (RUL) prediction. RUL prediction provides useful information for making maintenance plan before a failure occurs for engineering systems, which is of great significance to prevent sudden accidents and ensure reliable operation [4,5].

There have been many researches on the RUL prediction problem. In general, the RUL prediction methods can be distributed into two categories, i.e., mechanism model-based and data-driven methods [6,7]. Mechanism model-based methods rely on accurate prior knowledge of system failure mechanisms [8,9], which is usually unavailable owing to the high complexity of modern engineering systems. Besides, differentiated failure mechanisms of different systems lead to poor universality and adaptability of these methods. It is difficult and not practical to acquire satisfactory RUL prediction for complicated systems by using the mechanism model-based methods. Instead, owing to the improving availability of condition monitoring data, data-driven methods tend to be more promising to predict RUL for complicated systems.

Various sensors are utilized to conduct condition monitoring, the readings of which can indicate the health states of the engineering systems. On this basis, data-driven methods aim to explore the potential relationship between the multi-sensor data and the RUL of the systems [10]. Statistical method is one common way to establish the relationship. The statistical method makes predictions by using statistical models or stochastic process models to describe the health degradation of the systems, such as Gaussian mixture model [11], Wiener process [12,13], Brownian motion [14,15]. However, this method is quite dependent on a correct assumption of the data statistical characteristics. Besides, the relationship between the sensor data and the RUL value are generally obtained through indirect ways, of which the typical one is based on the health indicator (HI) [16]. This may give rise to the risk of

* Corresponding author.
E-mail address: wangyx@cauc.edu.cn (Y. Wang).

https://doi.org/10.1016/j.measurement.2021.109685
Received 26 March 2021; Received in revised form 16 May 2021; Accepted 30 May 2021
Available online 15 June 2021

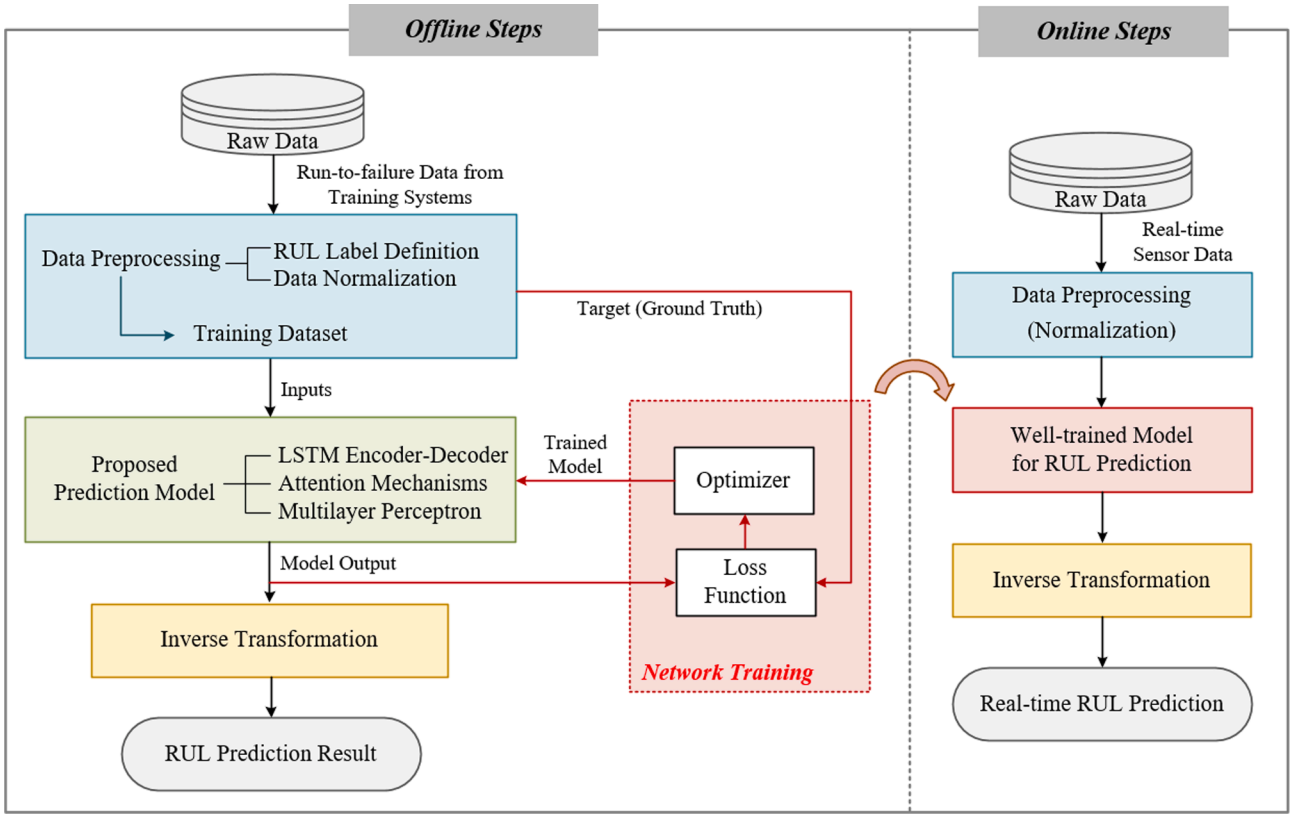


Fig. 1. Brief flowchart of the proposed RUL prediction method.

information loss in the process of transformation.

As an alternative, RUL prediction based on machine learning algorithms attracts more and more attention. In existing investigations, the conventional machine learning algorithms, such as support vector machine (SVM) [17,18], support vector regression (SVR) [19], extreme learning machine (ELM) [20], random forest (RF) [21], have been widely used in RUL prediction tasks. However, such methods usually require prior feature engineering based on relevant expertise of the system, which may lead to poor prediction performance for complicated systems with multiple sensors. As a branch of machine learning, deep learning models are capable of obtaining high-level abstractions automatically from large-scale data [22,23]. Therefore, deep learning methods have been introduced into the field of RUL prediction [2]. For example, [24] proposed a deep convolutional neural network (CNN) based regression approach for RUL prediction. In [25–27], deep belief networks (DBN) and long short-term memory (LSTM) networks were utilized to model the health trend and predict RUL for lithium-ion batteries and aircraft engines. In [28], after extracting the important features from the raw data by an auto-encoder (AE), a gated recurrent unit (GRU) was employed to select the information from the sequences to forecast RUL. However, these methods fail to construct an integrated model mapping the multi-sensor data directly to the RUL prediction result. This reduces the adaptability of them, and the stepwise learning of the model parameters makes it difficult to obtain optimal results. Although the relevant features can be extracted using these methods, the extracted features are treated equally without importance evaluation. Given that the multiple sensors correlate differently with health degradation, treating the features equally will influence or weaken the RUL prediction efficiency and accuracy [29]. In addition, the temporal correlation between the current RUL and the historical data of different time intervals is varying, thus a dynamic and adaptive evaluation of the temporal correlation is also of great significance to enhance the RUL prediction accuracy. However, it has not been considered in previous studies on RUL prediction. RUL prediction for complicated engineering systems with multi-sensor data still face challenges.

Motivated by these facts, the multi-sensor prognostics problem for complicated engineering systems is investigated in this paper. To overcome the drawbacks of the aforementioned approaches, this paper attempts to develop a practical end-to-end method so that it can take full use of multi-sensor data and achieve high-accuracy RUL prediction. The main frame of the proposed prediction model is constructed based on a LSTM network, owing to its capability of dealing with time series and learning long-term dependencies [30,31]. Since attention mechanism provides an efficient way to focus on the more critical information among the numerous input information for the current task [32–35], a two-stage attention mechanism is designed as an important part of the model. Through this design, the relevant input features as well as the temporal correlation can be extracted and evaluated dynamically and adaptively. On this basis, the proposed end-to-end model can map the multi-sensor data directly to the RUL prediction result with high accuracy.

The remainder of this paper is as follows. In Section 2, we briefly introduce the RUL prediction problem and provide an overview of the proposed method. Section 3 presents the data preprocessing methods involved in the RUL prediction task. In Section 4, an end-to-end RUL prediction model is proposed, which consists of a frame of LSTM encoder-decoder, a two-stage attention mechanism and a multilayer perceptron. In Section 5, experiments are conducted on a publicly available dataset to illustrate the effectiveness and superiority of the proposed method. Section 6 concludes this paper and outlines areas for future studies.

2. Problem statement and method overview

Multiple sensors of various types are mounted on or inside modern engineering systems to monitor the operational and environmental

states, for example, temperature, vibration, and pressure. These sensor readings are recorded in real time and can indicate health degradation of the systems, which makes RUL prediction possible.

We consider the scenario where the health conditions of a set of engineering system instances (denoted by $\boldsymbol{U}$) are monitored and stored in multi-sensor data till the end of life. For each system $u \in \boldsymbol{U}$, the collected multi-sensor data is a multivariate time series, denoted by $\boldsymbol{X}^{(u)} = \left\{\boldsymbol{x}_1^{(u)}, \boldsymbol{x}_2^{(u)}, \cdots, \boldsymbol{x}_L^{(u)}\right\}$. Here L represents the last life cycle of the system, and $\boldsymbol{x}_t \in \mathbb{R}^n$ is an n-dimension vector corresponding to readings for n sensors at time t, i.e., $\boldsymbol{x}_t^{(u)} = \left[x_{tj}^{(u)}\right]_{j=1}^{n}$. With the set of run-to-failure sensor readings $\{\boldsymbol{X}^{(u)} | u \in \boldsymbol{U}\}$, our goal is to construct a nonlinear mapping as follows:

$$f_{\Theta_p} : X^{(u)}(t,l) \rightarrow R_t^{(u)}, t \in \{l, l+1, \cdots, L^{(u)}\} \tag{1}$$

where $\boldsymbol{\Theta}_p$ is a set of parameters to be learned. $\boldsymbol{X}^{(u)}(t,l)$ is a subsequence of length l for time series $\boldsymbol{X}^{(u)}$ starting from time $t-l+1$, that is

$$\boldsymbol{X}^{(u)}(t,l) = \left\{\boldsymbol{x}_{t-l+1}^{(u)}, \boldsymbol{x}_{t-l+2}^{(u)}, \cdots, \boldsymbol{x}_t^{(u)}\right\} \tag{2}$$

$R_t^{(u)}$ is the target RUL value at time t. The model construction procedures are performed offline. Once the nonlinear mapping model is well trained, it can be utilized to predict RUL for other system instances $\boldsymbol{U}^*$ in real time. For each system $u^* \in \boldsymbol{U}^*$, the predicted RUL is given by

$$\begin{aligned} \widehat{R}_t^{(u^*)} &= f_{\Theta_p}(X^{(u^*)}(t,l)) \\ &= f_{\Theta_p}\left(x_{t-l+1}^{(u^*)}, x_{t-l+2}^{(u^{*U})}, \cdots, x_t^{(u^*)}\right), t \in \{l, l+1, \cdots, L^{(u^*)}\} \end{aligned} \tag{3}$$

In other words, with the trained model, the RUL of a currently operating system can be estimated if the historical and the current multi-sensor data is available.

As analyzed above, our RUL prediction method can be summarized into two parts, a brief flowchart of the proposed method is demonstrated in Fig. 1.

In the offline part, the run-to-failure sensor readings are preprocessed and regarded as the inputs of the prediction model. The computed model outputs are then compared with the target RUL values, with which the model can be trained. Based on the above steps, we obtain the well-trained prediction model.

In the online part, the multi-sensor reading of an operating system are collected and recorded in real time. Then the current and historical sensor readings are preprocessed and fed into the well-trained prediction model. By conducting inverse transformation to the model output, the real-time RUL prediction can be achieved.

In the frame of the proposed method, the main issues involved, i.e., data preprocessing, design of the model architecture and RUL prediction, will be elaborated in the following Sections.

3. Data preprocessing

As shown in Fig. 1, data preprocessing is an essential step in both the offline and the online parts for RUL prediction. Thus, the involved data preprocessing methods will be presented in this section.

3.1. Definition of RUL label

RUL is defined as the period from the current time to the time at which a system fails to work. As mentioned above, the multi-sensor readings and the corresponding RUL labels of system instances $\boldsymbol{U}$ constitute the training dataset, with which the nonlinear mapping function $f_{\boldsymbol{\Theta}_p}$ can be learned. In actual systems, the run-to-failure sensor readings are collected and recorded without specific RUL value. But the RUL label can be determined by the difference between the maximum life cycle and the current time as follows:

$$R_t^{(u)} = L^{(u)} - t, \quad u \in U \tag{4}$$

where $R_t^{(u)}$ is the RUL label of system u at time t.

In most cases, health degradation of an engineering system will remain negligible until some period of operating time [24]. Based on this fact, the RUL label is considered as constant at the early stage of the system life cycles. The truncated RUL label can be formulated as:

$$R_t^{(u)} = \begin{cases} L_T, & if\, t \leqslant L^{(u)} - L_T \\ L^{(u)} - t, & if \;\; t > L^{(u)} - L_T \end{cases}, \quad u \in \boldsymbol{U} \tag{5}$$

where L_T represents the truncation threshold of the RUL. $L^{(u)} - L_T$ corresponds to the specific time point, before which there is little degradation in system u. As given by (4) and (5), the RUL curve changes from the original linear decline to a nonlinear decline.

3.2. Data normalization

Generally, the multi-sensor readings corresponding to various features, are different physical quantities with different ranges and units. If these sensor readings are utilized to train the prediction model directly, they might behave badly and even lead to non-convergence. In practice, we transform the data to center it by removing the mean value of each feature, then scale it by dividing non-constant features by their standard deviation:

$$\widetilde{x}_{tj}^{(u)} = \frac{x_{tj}^{(u)} - \mu_j}{\sigma_j}, \quad u \in \boldsymbol{U} \tag{6}$$

where μ_j and σ_j are mean and standard deviation for the j-th sensor readings over all cycles from all system instances in $\boldsymbol{U}$. Centering and scaling happen independently on each feature by computing the relevant statistics.

Considering the difference of order of magnitude between the sensor data and the RUL value, the RUL label is also normalized to improve the model training performance. The normalized RUL can be obtained by:

$$\widetilde{R}_t^{(u)} = \frac{R_t^{(u)} - R_{\min}}{R_{\max} - R_{\min}}, \quad u \in \boldsymbol{U} \tag{7}$$

where $R_{\max}$ and $R_{\min}$ are the maximum and minimum values of the RUL for all system instances in $\boldsymbol{U}$.

For brevity, the tilde symbol will be omitted, and hereafter notation x will refer to the normalized sensor data. In addition, the normalized RUL will be denoted by y in the following.

After normalization, the multivariate time series data should be further processed to satisfy the requirements of model training and RUL prediction in the following sections. This is achieved based on sliding time window technique. A fixed-length time window l is used to enclose multivariate data points sampled at consecutive cycles, and the time window will shift one cycle each time till the end of life. The form of the obtained subsequences can be expressed as (2).

Then the dataset used for training the RUL prediction model can be obtained. The training dataset contains a number of input-target pairs described as:

$$\left\{\left(\boldsymbol{X}^{(u)}(t,l), y_t^{(u)}\right) \middle| t \in \{l, l+1, \cdots, L^{(u)}\}, u \in \boldsymbol{U}\right\} \tag{8}$$

In addition, it is to be noted that the aforementioned parameters for normalization should be stored, so as to later re-apply the same transformation on the testing dataset or obtain the predicted RUL by inverse-transformation.

4. Proposed RUL prediction model

As analyzed in Section 2, RUL prediction can be deemed as a problem

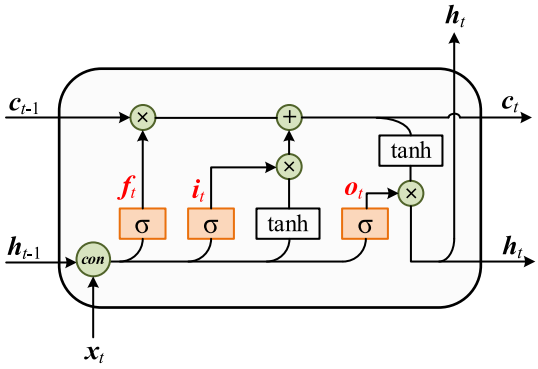


Fig. 2. Diagram of LSTM unit.

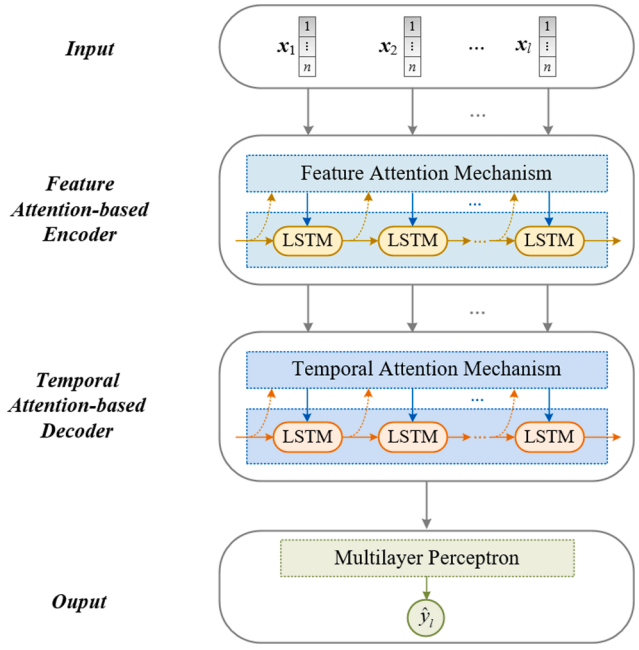


Fig. 3. Graphical illustration of the proposed model.

of multivariate time series prediction. LSTM network, as an excellent variation of recurrent neural network, is suitable to deal with sequential data and is capable of learning long-term dependencies [22]. Therefore, the deep LSTM network is adopted to process the multi-sensor readings in this paper. Compared with the vanilla RNN, the LSTM unit shown in Fig. 2 utilizes a combination of a memory cell and three kinds of gates: input gate, forget gate, and output gate. The three gates provide a way to optionally let information through, thus the LSTM is able to remove or add information to the cell state.

In practical engineering, the input features provided by multiple sensors contribute to the health degradation differently. Meanwhile, the final RUL prediction results have different dependencies on the input at different time steps, and the temporal correlation may vary with the changing of the degree of the health degradation. Thus, it is essential to focus on the more critical information among the numerous input information, so as to acquire the satisfactory accuracy for RUL prediction. Motivated by this fact, attention mechanisms are designed as an important part of the prediction model to select the most relevant input features as well as extract the temporal correlation adaptively.

In this paper, the LSTM encoder-decoder [36] is employed as the main frame of the prediction model to deal with time series data, besides, a feature attention mechanism and a temporal attention mechanism are designed in the encoder and decoder respectively. In addition, a multilayer perceptron is utilized to obtain the final prediction output. A graphical illustration of the proposed model is shown in Fig. 3. These main modules of the proposed model will be demonstrated in more detail below.

4.1. Multi-sensor data fusion via feature attention mechanism

For complicated engineering systems, the monitored feature is generally of different degree of correlation with health degradation. Treating all the features equally may influence or weaken the RUL prediction efficiency and accuracy. Thus, a feature attention mechanism is designed in the encoder to adaptively extract the relevant multi-sensor data and evaluate each feature at each time step. On this basis, fusion of the multi-sensor data can be achieved without any prior knowledge. A brief illustration of the proposed attention mechanism is shown in Fig. 4.

As shown in Fig. 4, the feature attention mechanism mainly consists of a multilayer perceptron and a softmax layer. At each time step, each input feature is scored via the multilayer perceptron, by referring to the previous hidden state of the encoder:

$$p_{tj} = \boldsymbol{v}_p^{\mathrm{T}} \tanh\left(\boldsymbol{W}_p \left[\boldsymbol{h}_{t-1}^{(E)}; \boldsymbol{x}^j\right] + \boldsymbol{b}_p\right) \tag{9}$$

where p_{tj} represents the attention score of the j-th sensor data at time t. $\boldsymbol{x}^j = \left[x_{1j}, x_{2j}, \cdots, x_{lj}\right]^{\mathrm{T}} \in \mathbb{R}^l$ is the time series from the j-th sensor. $\boldsymbol{h}_{t-1}^{(E)} \in \mathbb{R}^m$ is the previous hidden state of the encoder, here m is the number of encoder hidden neurons. $\boldsymbol{v}_p \in \mathbb{R}^l$, $\boldsymbol{W}_p \in \mathbb{R}^{l\times(m+l)}$ and $\boldsymbol{b}_p \in \mathbb{R}^l$ are the weight matrices and the bias vector to be learned. Then the attention weight of each input feature can be determined via the softmax layer:

$$\omega_{tj} = \frac{exp\left(p_{tj}\right)}{\sum_{j=1}^{n} exp\left(p_{tj}\right)} \tag{10}$$

where ω_{tj} is the attention weight of the j-th sensor data at time t, which represents the importance of the j-th sensor data. With the attention weights, we can take importance-based sampling for input data with:

$$\widetilde{\boldsymbol{x}}_t = \boldsymbol{\omega}_t \odot \boldsymbol{x}_t \tag{11}$$

where $\widetilde{\boldsymbol{x}}_t$ corresponds to the newly computed input at time t. $\boldsymbol{\omega}_t = \left[\omega_{tj}\right]_{j=1}^{n} = \left[\omega_{t1}, \omega_{t2}, \cdots, \omega_{tn}\right] \in \mathbb{R}^n$ is the attention weight vector at time t. $\odot$ stands for elementwise multiplication.

By feeding the newly computed input $\widetilde{\boldsymbol{x}}_t$ into the LSTM network, the hidden state at time t can be updated as follows:

$$\begin{aligned}
\boldsymbol{i}_t^{(E)} &= \mathrm{sigmoid}\left(\boldsymbol{W}_i^{(E)}\left[\boldsymbol{h}_{t-1}^{(E)}; \widetilde{\boldsymbol{x}}_t\right] + \boldsymbol{b}_i^{(E)}\right) \\
\boldsymbol{f}_t^{(E)} &= \mathrm{sigmoid}\left(\boldsymbol{W}_f^{(E)}\left[\boldsymbol{h}_{t-1}^{(E)}; \widetilde{\boldsymbol{x}}_t\right] + \boldsymbol{b}_f^{(E)}\right) \\
\boldsymbol{o}_t^{(E)} &= \mathrm{sigmoid}\left(\boldsymbol{W}_o^{(E)}\left[\boldsymbol{h}_{t-1}^{(E)}; \widetilde{\boldsymbol{x}}_t\right] + \boldsymbol{b}_o^{(E)}\right) \\
\boldsymbol{c}_t^{(E)} &= \boldsymbol{f}_t^{(E)} \odot \boldsymbol{c}_{t-1}^{(E)} + \boldsymbol{i}_t^{(E)} \odot tanh\left(\boldsymbol{W}_c^{(E)}\left[\boldsymbol{h}_{t-1}^{(E)}; \widetilde{\boldsymbol{x}}_t\right] + \boldsymbol{b}_c^{(E)}\right) \\
\boldsymbol{h}_t^{(E)} &= \boldsymbol{o}_t^{(E)} \odot tanh\left(\boldsymbol{c}_t^{(E)}\right)
\end{aligned} \tag{12}$$

where $\boldsymbol{i}_t^{(E)}, \boldsymbol{f}_t^{(E)}, \boldsymbol{o}_t^{(E)}$ are the input gate, forget gate and output gate of the encoder LSTM unit, respectively. $\boldsymbol{c}_t^{(E)}$ is the cell state of the encoder, and $\left[\boldsymbol{h}_{t-1}^{(E)}; \widetilde{\boldsymbol{x}}_t\right] \in \mathbb{R}^{m+n}$ is a concatenation of the previous hidden state and the current input $\widetilde{\boldsymbol{x}}_t$. $\boldsymbol{W}_i^{(E)}, \boldsymbol{W}_f^{(E)}, \boldsymbol{W}_o^{(E)}, \boldsymbol{W}_c^{(E)} \in \mathbb{R}^{m\times(m+n)}$, and $\boldsymbol{b}_i^{(E)}, \boldsymbol{b}_f^{(E)}, \boldsymbol{b}_o^{(E)}, \boldsymbol{b}_c^{(E)} \in \mathbb{R}^m$ are the weight matrices and the bias vectors to be learned.

With the proposed feature attention mechanism, the encoder can selectively focus on certain features rather than treating all the senor data equally.

4.2. Temporal correlation extraction via temporal attention mechanism

In the above subsection, the feature attention-based encoder is applied to extract and evaluate the relevant information from the multi-

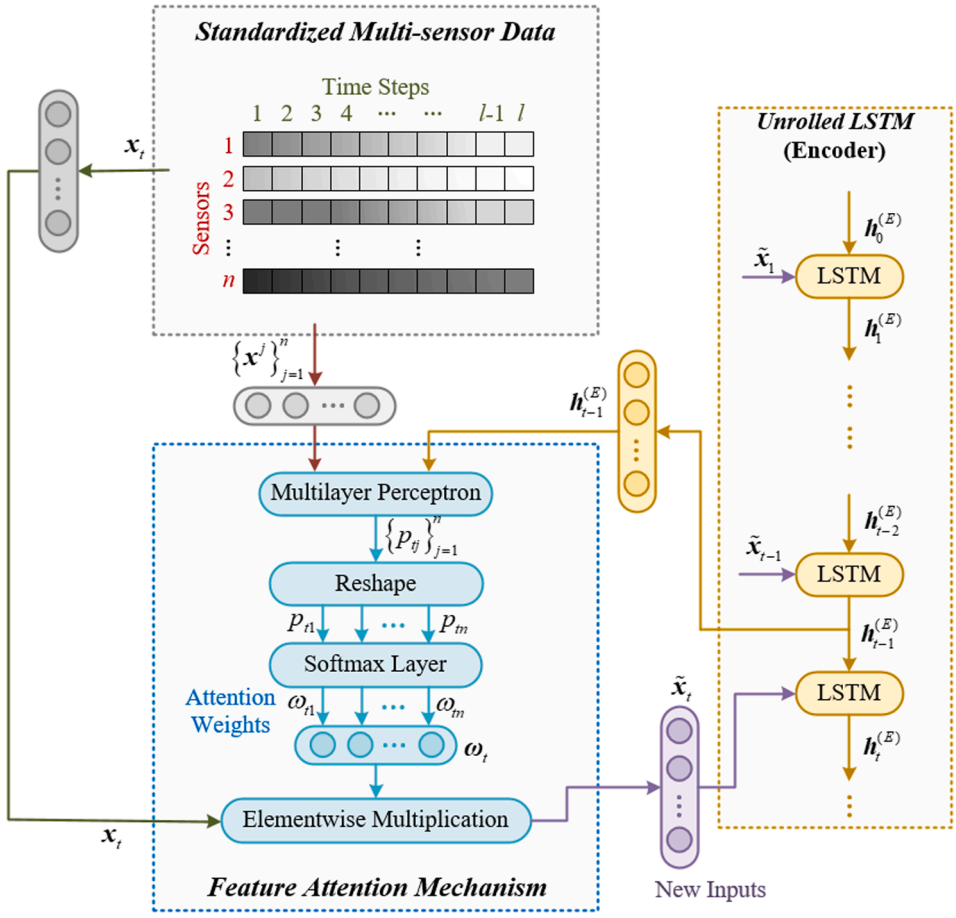


Fig. 4. Graphical illustration of the feature attention-based encoder.

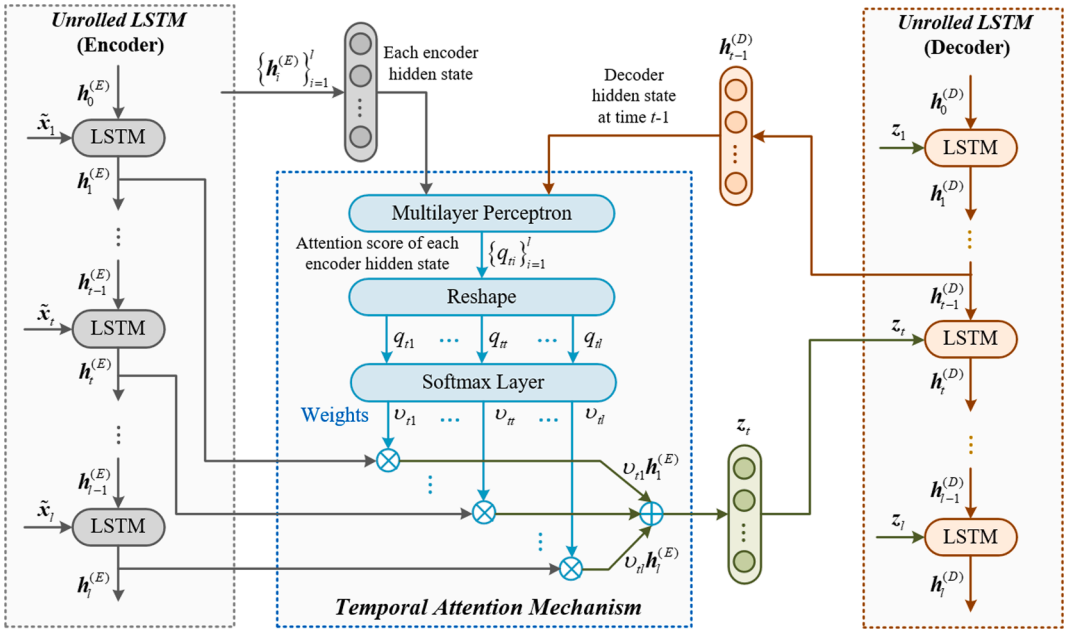


Fig. 5. Graphical illustration of the temporal attention-based decoder.

sensor data, which maps the multi-sensor sequence to an output sequence (i.e. the encoder hidden states). These encoder hidden states $\boldsymbol{h}_i^{(E)}, i \in \{1, 2, \cdots, l\}$ correspond to the fusion results of multiple sensors at different time steps, which contribute to the final prediction differently.

As a result, a temporal attention mechanism is designed in the decoder, which is utilized to capture the dynamic temporal correlation between different time intervals. A graphical illustration of the proposed temporal attention-based decoder is shown in Fig. 5.

Table 1
Illustration of model training process.

Training Procedures
Input: training dataset consists of samples of normalized input features and RUL labels ***Output***: trained model parameters Θ_p Initialization. **While** stopping criterion not met **do:** Compute feature attention weights and generate new features according to (9)-(11). Update encoder hidden states according to (12). Compute temporal attention weights and fuse encoder hidden states according to (13)-(15). Update decoder hidden states according to (16). Compute model output according to (17). Compute loss according to (18). Compute gradient and obtain parameter update: $\Delta\Theta_p \leftarrow \mathrm{Adam}\left(\nabla_{\Theta_p}\sum_i L(\Theta_p)\right)$. Apply update: $\Theta_p \leftarrow \Theta_p + \Delta\Theta_p$. **end while**

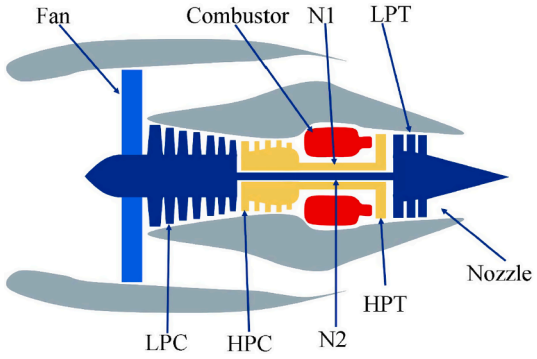


Fig. 6. Simplified diagram of the turbofan engine.

Table 2
Description of multiple sensors.

Sensor	Description	Sensor	Description
T2	Total temperature at fan inlet	phi	Ratio of fuel flow to Ps30
T24	Total temperature at LPC outlet	NRf	Corrected fan speed
T30	Total temperature at HPC outlet	NRc	Corrected core speed
T50	Total temperature at LPT outlet	BPR	Bypass ratio
P2	Pressure at fan inlet	farB	Burner fuel-air ratio
P15	Total pressure in bypass-duct	htBleed	Bleed enthalpy
P30	Total pressure at HPC outlet	Nf_dmd	Demanded fan speed
Nf	Physical fan speed	PCNfR_dmd	Demanded corrected fan speed
Nc	Physical core speed	W31	HPT coolant bleed
epr	Engine pressure ratio	W32	LPT coolant bleed
Ps30	Static pressure at HPC outlet		

Table 3
Performance comparison of different time window length.

Time window length	RMSE	Score
5	15.44	2155.72
10	12.58	1497.13
20	10.67	1102.62
30	9.52	806.18
40	9.59	812.97
50	10.29	916.32

Table 4
Hyperparameter setting of the RUL prediction model.

Hyperparameter	Value	
	FD001/FD003	FD002/FD004
Number of inputs	14/16	20
Time window length	30	30
Number of encoder hidden neurons	20	20
Number of decoder hidden neurons	20	20
Number of MLP hidden neurons	35	(30,15)
Number of outputs	1	1
Initial learning rate	0.001	0.001
Dropout rate	0.2	0.2
Early stopping patience	5	5
Batch Size	128	256

Similar to the feature attention mechanism, the temporal attention mechanism mainly consists of a multilayer perceptron and a softmax layer. At each time step, each encoder hidden state is scored via the multilayer perceptron, by referring to the previous hidden state of the decoder:

$$q_{ti} = \boldsymbol{v}_q^{\mathrm{T}} \tanh\left(\boldsymbol{W}_q\left[\boldsymbol{h}_{t-1}^{(D)}; \boldsymbol{h}_i^{(E)}\right] + \boldsymbol{b}_q\right) \tag{13}$$

where q_{ti} represents the attention score of $\boldsymbol{h}_i^{(E)}$ at output time t. $\boldsymbol{h}_{t-1}^{(D)} \in \mathbb{R}^d$ is the previous hidden state of the decoder, here d is the number of decoder hidden neurons. $\boldsymbol{v}_q \in \mathbb{R}^m$, $\boldsymbol{W}_q \in \mathbb{R}^{m\times(m+d)}$ and $\boldsymbol{b}_q \in \mathbb{R}^m$ are learnable parameters. When i varies from 1 to l, $\boldsymbol{h}_i^{(E)}$ refers to different samples of the multilayer perception input. Thus, feeding each $\boldsymbol{h}_i^{(E)}$ into the multilayer perception yields a corresponding attention score q_{ti}.

Once the attention scores have been obtained, they are fed together into the softmax layer to compute the attention weights. The attention weight of each encoder hidden state can be determined via the softmax layer:

$$v_{ti} = \frac{exp(q_{ti})}{\sum_{i=1}^{l} exp(q_{ti})} \tag{14}$$

Based on this attention mechanism, the relevant encoder hidden states can be adaptively selected across all time steps. Then a weighted sum of all the encoder hidden states can be computed as:

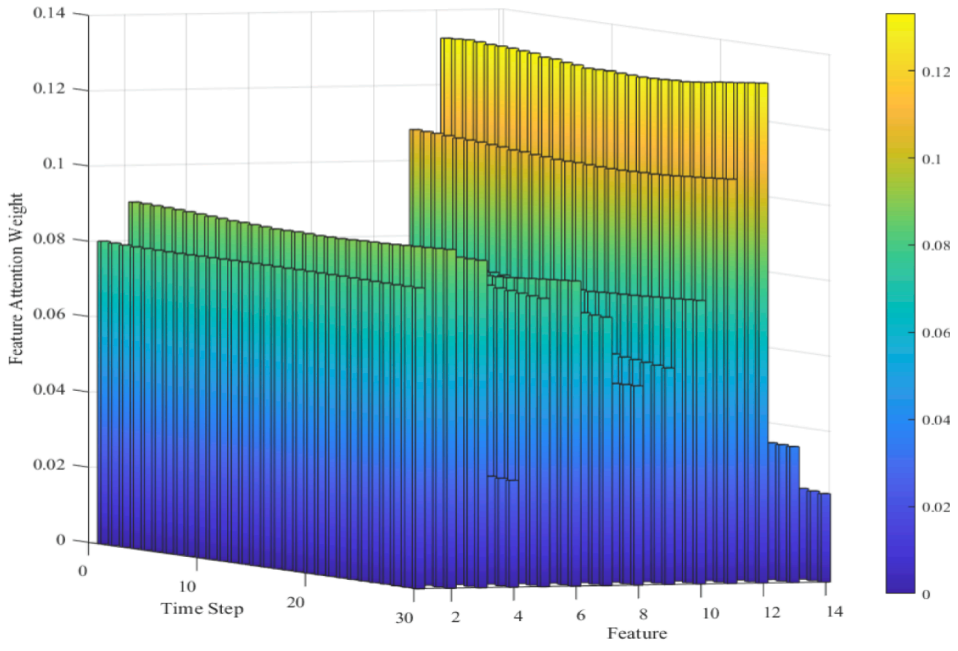
$$z_t = \sum_{i=1}^{l} v_{ti}\boldsymbol{h}_i^{(E)} \tag{15}$$

z_t corresponds to is an adaptive fusion of the encoder hidden states $\boldsymbol{h}_i^{(E)}, i \in \{1, 2, \cdots, l\}$ at output time t. By feeding z_t into the encoder LSTM, the hidden state at time t can be updated as follows:

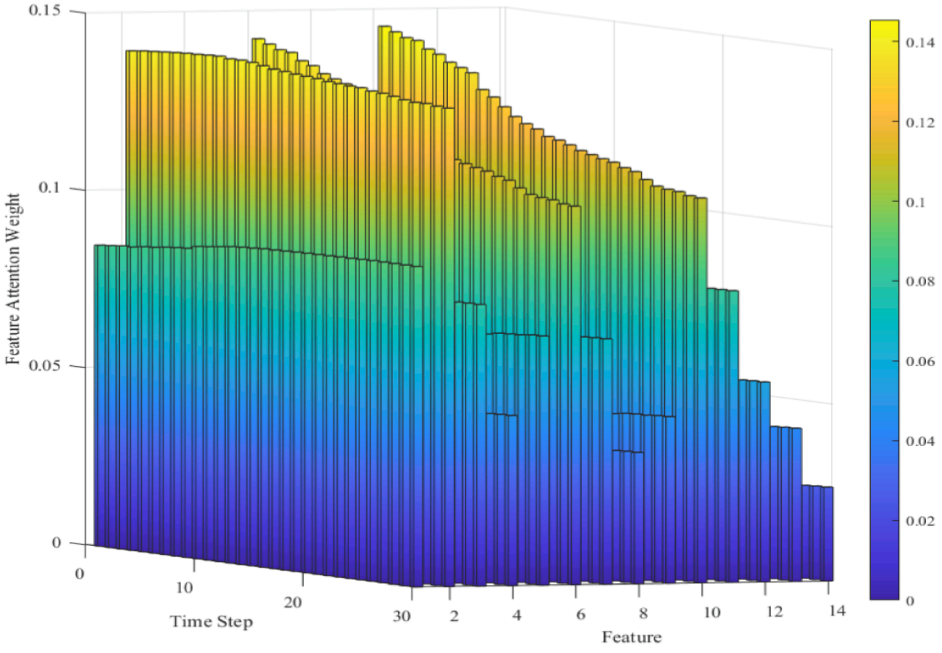
$$\begin{aligned}
\boldsymbol{i}_t^{(D)} &= \mathrm{sigmoid}\left(\boldsymbol{W}_i^{(D)}\left[\boldsymbol{h}_{t-1}^{(D)}; z_t\right] + \boldsymbol{b}_i^{(D)}\right) \\
\boldsymbol{f}_t^{(D)} &= \mathrm{sigmoid}\left(\boldsymbol{W}_f^{(D)}\left[\boldsymbol{h}_{t-1}^{(D)}; z_t\right] + \boldsymbol{b}_f^{(D)}\right) \\
\boldsymbol{o}_t^{(D)} &= \mathrm{sigmoid}\left(\boldsymbol{W}_o^{(D)}\left[\boldsymbol{h}_{t-1}^{(D)}; z_t\right] + \boldsymbol{b}_o^{(D)}\right) \\
\boldsymbol{c}_t^{(D)} &= \boldsymbol{f}_t^{(D)} \odot \boldsymbol{c}_{t-1}^{(D)} + \boldsymbol{i}_t^{(D)} \odot tanh\left(\boldsymbol{W}_c^{(D)}\left[\boldsymbol{h}_{t-1}^{(D)}; z_t\right] + \boldsymbol{b}_c^{(D)}\right) \\
\boldsymbol{h}_t^{(D)} &= \boldsymbol{o}_t^{(D)} \odot tanh\left(\boldsymbol{c}_t^{(D)}\right)
\end{aligned} \tag{16}$$

where $\boldsymbol{i}_t^{(D)}, \boldsymbol{f}_t^{(D)}, \boldsymbol{o}_t^{(D)}$ are the input gate, forget gate and output gate of the decoder LSTM unit, respectively. $\boldsymbol{c}_t^{(D)}$ is the cell state of the decoder. $\boldsymbol{W}_i^{(D)}$, $\boldsymbol{W}_f^{(D)}$, $\boldsymbol{W}_o^{(D)}$, $\boldsymbol{W}_c^{(D)} \in \mathbb{R}^{d\times(m+d)}$, and $\boldsymbol{b}_i^{(D)}, \boldsymbol{b}_f^{(D)}, \boldsymbol{b}_o^{(D)}, \boldsymbol{b}_c^{(D)} \in \mathbb{R}^d$ are learnable parameters.

Given that the hidden state can be deemed as the selective information from the cell state, the previous cell states are not involved in the designed attention mechanisms. Thus compared with existing researches such as [35], the attention mechanisms of our method are designed in a more concise way, which will result in fewer parameters to be learnt and an improvement of model training efficiency.



(a) sample 1

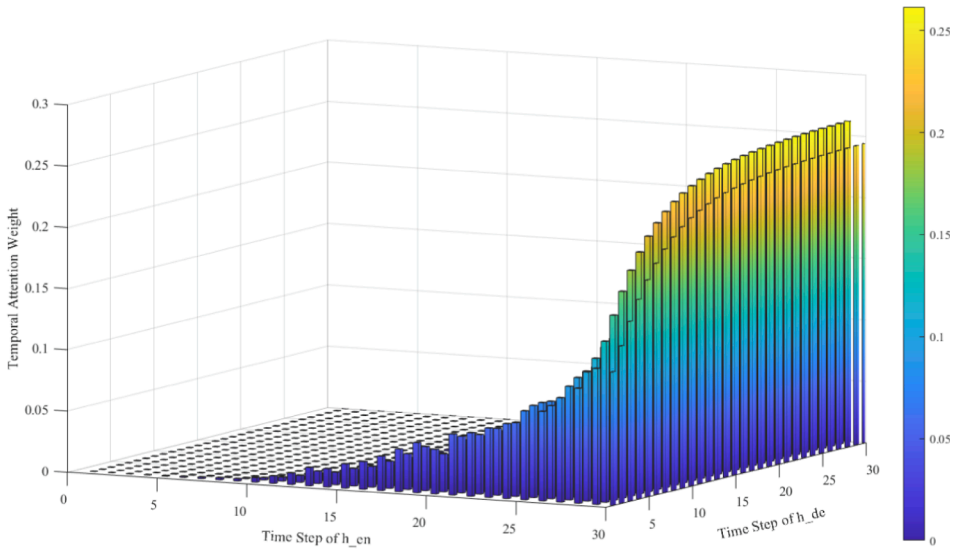


(b) sample 2

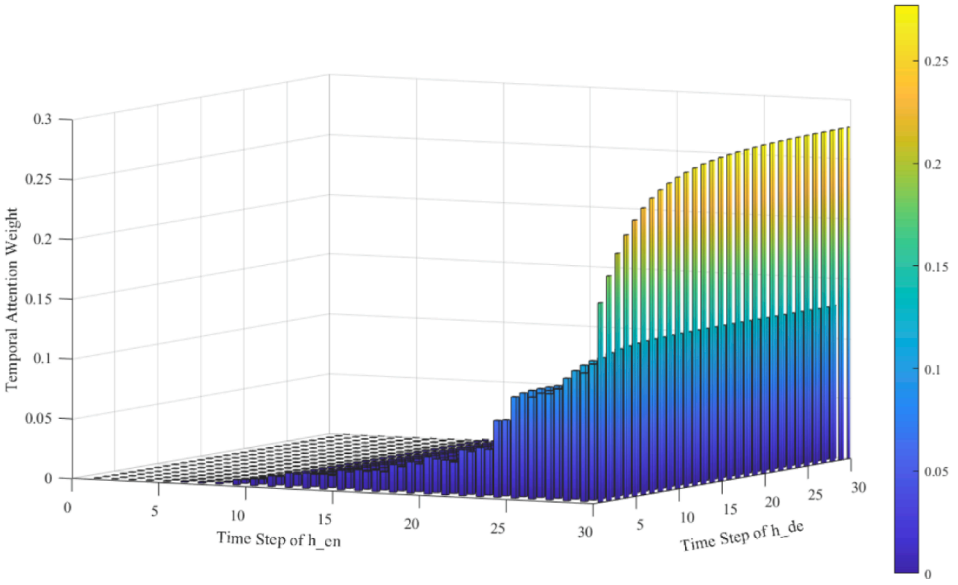
Fig. 7. Feature attention weights of two samples.

4.3. Final output generation via multilayer perceptron

With the encoder-decoder network and the designed attention mechanisms, the relevant feature information as well as the temporal correlation hidden in the multi-sensor data can be extracted. The last decoder hidden state $\boldsymbol{h}_l^{(D)}$ contains information about the whole historical sequence, from which the final RUL prediction result can be obtained. Thus, we adopt a multilayer perceptron (MLP) to generate the final output as follows:



(a) sample 1



(b) sample 2

Fig. 8. Temporal attention weights of two samples.

$$\widehat{y}_l = \boldsymbol{v}_y^{\mathrm{T}} \mathrm{relu}\left(\boldsymbol{W}_y \boldsymbol{h}_l^{(D)} + \boldsymbol{b}_y\right) \tag{17}$$

where $\boldsymbol{v}_y \in \mathbb{R}^r, \boldsymbol{W}_y \in \mathbb{R}^{r\times d}, \boldsymbol{b}_y \in \mathbb{R}^r$ are parameters to be learned, which map the decoder hidden state $\boldsymbol{h}_l^{(D)}$ to the final output $\widehat{y}_l$. r is the number of the hidden neurons in the multilayer perceptron. Actually, multiple hidden layers can be utilized in the multilayer perceptron, and the calculating process is similar to (17). Then the final predicted RUL value $\widehat{R}_l$ can be obtained by applying inverse transformation to $\widehat{y}_l$.

4.4. Model training

As stated above, (9)-(17) depict the whole calculating process of the RUL prediction model. The performance of the RUL prediction model can be evaluated by the error between the ground truth y_l and the predicted result $\widehat{y}_l$. The goal is to minimize the loss function below:

$$L(\Theta_p) = \frac{1}{N_s}\sum_{i=1}^{N_s}\left(\widehat{y}_l^i - y_l^i\right)^2 \tag{18}$$

where N_s represents the number of training samples. Θ_p refers to the set of model parameters to be learned:

$$\Theta_p = \Big\{v_p, W_p, b_p, W_i^{(E)}, W_f^{(E)}, W_o^{(E)}, W_c^{(E)}, b_i^{(E)}, b_f^{(E)}, b_o^{(E)}, b_c^{(E)}, v_q, W_q, b_q, W_i^{(D)}, W_f^{(D)}, W_o^{(D)}, W_c^{(D)}, b_i^{(D)}, b_f^{(D)}, b_o^{(D)}, b_c^{(D)}, v_y, W_y, b_y\Big\} \tag{19}$$

In the training process of the proposed model, the Adaptive Moment Estimation (Adam) [37] algorithm is adopted to optimize the whole network. As the proposed end-to-end model architecture is smooth and

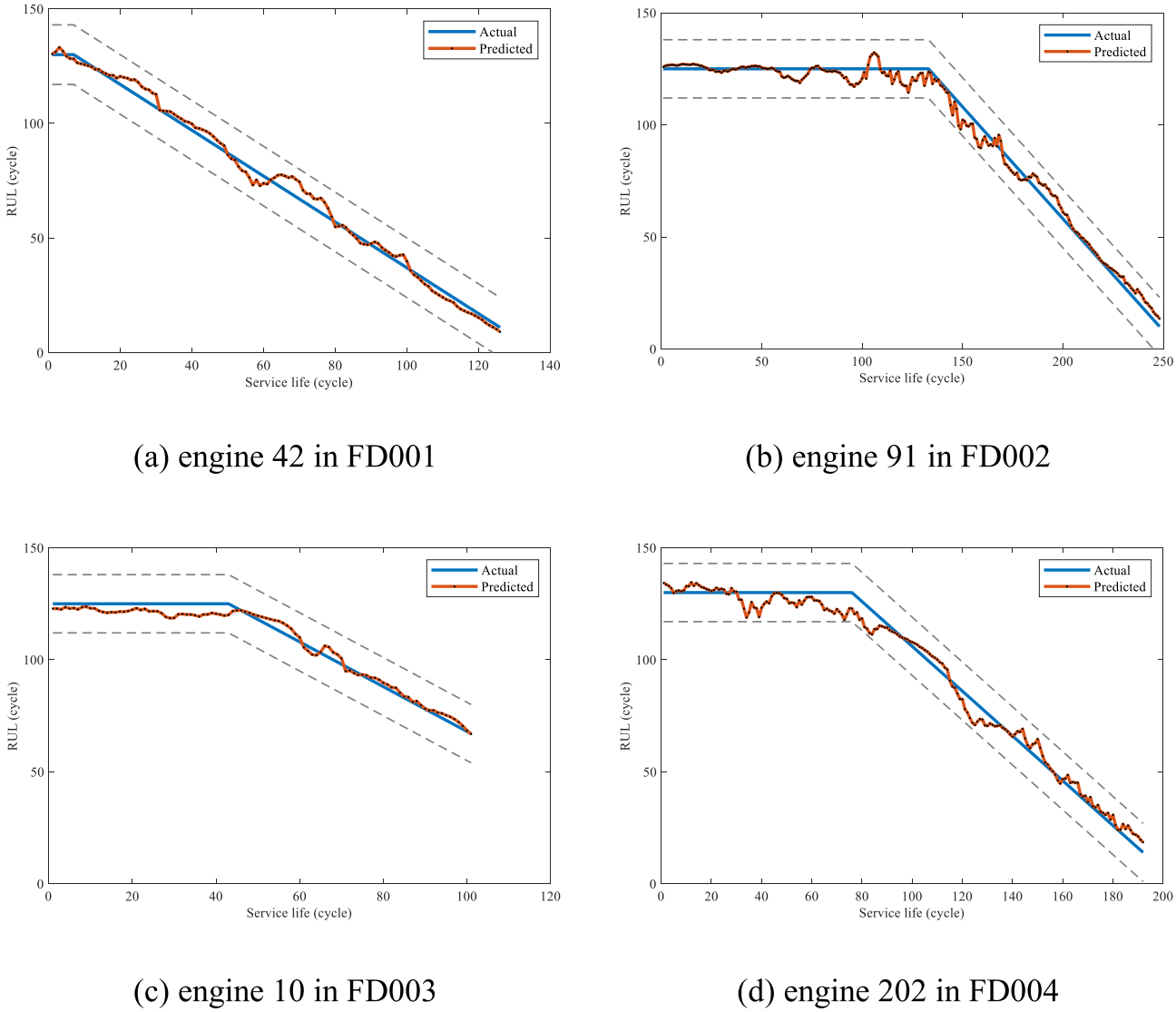


Fig. 9. Examples of RUL prediction results.

differentiable, the parameters in can be jointly learned based on back propagation (through time) algorithm to minimize the objective function given by (18). The model training procedures are presented in Table 1.

In addition, dropout is adopted owing to its powerful ability for preventing neural networks from overfitting. The performance on the cross validation set is monitored to perform the early stopping mechanism, which can prevent model from overfitting as well.

Once the model has been well trained, the nonlinear function f_{Θ_p}, mapping the multivariate time series from multiple sensors to the current RUL value, can be determined according to (9)-(17).

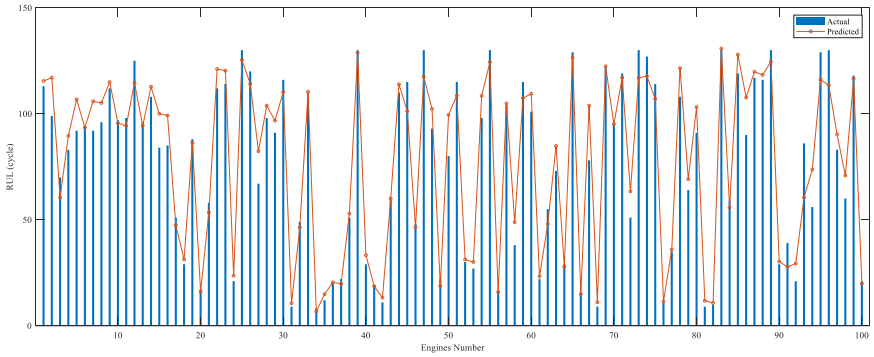
5. Experimental verification

In this section, the performance of the proposed RUL prediction method is investigated by conducting experiments on a publicly available dataset: C-MAPSS Turbofan Engine Dataset [38]. A simplified diagram of the turbofan engine is presented in Fig. 6 [39].

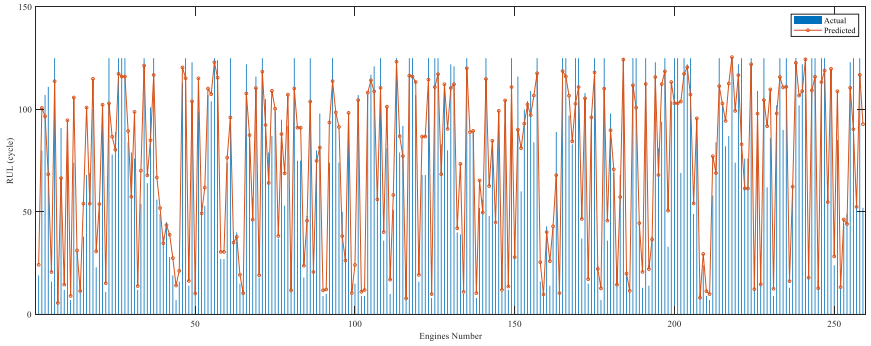
5.1. C-MAPSS turbofan engine dataset

5.1.1. Dataset overview

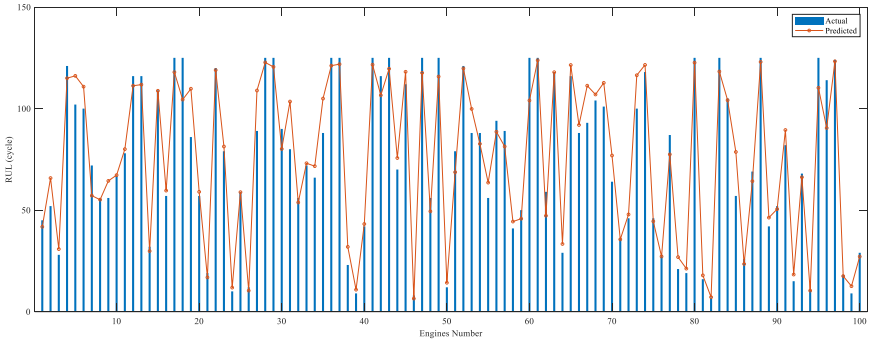
C-MAPSS turbofan engine dataset is a publicly available dataset provided by the Prognostics Center of Excellence at NASA Ames Research Center, which contains four sets corresponding to four different combinations of operational conditions and fault modes [38]. All the four sets are adopted to demonstrate the effectiveness of the proposed method. Due to limited space, detailed discussions on the first set FD001 will be presented as an example. Besides, the final prediction results on all the four sets will be given in the following. The set FD001 is further divided into two subsets. In train_FD001, readings of 21 sensors from 100 turbofan engines are recorded till the end of life. In test_FD001, similar sensor readings from 100 testing engines are provided, which end some time prior to system failure. Description of the 21 sensors is listed in Table 2 [39]. The actual RUL values are given in RUL_FD001. The other three sets are similar to FD001 except that FD002 and FD004 are more challenging due to more engines and more operational conditions involved. The detailed description of them is not given due to limited space, which can be found in [38]. Each engine has a different degree of initial wear and manufacturing variation. The data is contaminated with sensor noise. Our goal is to establish the RUL prediction model based on the data from the training engines, then predict RUL for these testing engines. For FD001, there are a total of 20,631 cycles for training engines, and 13,096 cycles for testing engines.



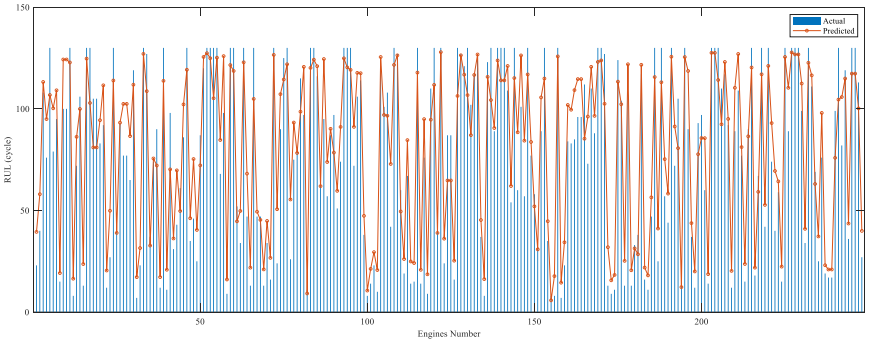
(a) results on FD001



(b) results on FD002



(c) results on FD003



(d) results on FD004

Fig. 10. Final RUL prediction results of all the testing engines.

Table 5
Performance comparison on training, validation and testing sets.

Method	Training set		Validation set		Testing set	
	Loss	RMSE	Loss	RMSE	Loss	RMSE
Vanilla RNN	0.0162	16.065	0.0184	16.524	0.0213	18.969
Vanilla LSTM	0.0111	13.548	0.0145	14.026	0.0183	17.592
RNN with two-stage attention	0.0121	14.146	0.0151	14.373	0.0168	15.951
LSTM with two-stage attention	**0.0106**	**13.012**	**0.0125**	**13.215**	**0.0131**	**14.563**

Table 6
RMSE for various methods on C-MAPSS dataset.

Method	FD001	FD002	FD003	FD004
MLP [24]	37.56	80.03	37.39	77.37
SVR [24]	20.96	42.00	21.05	45.35
ELM [27]	17.27	37.28	18.90	38.43
RF [27]	17.91	29.59	20.27	31.12
CNN [24]	18.45	30.29	19.82	29.16
DBN [27]	15.21	27.12	14.71	29.88
MODBNE [27]	15.04	25.05	12.51	28.66
LSTM with a single attention [29]	14.53	–	–	27.08
Semi-supervised setup [5]	12.56	22.73	12.10	22.66
HDNN [40]	13.02	**15.24**	12.22	**18.16**
1-FCLCNN-LSTM [41]	**11.17**	–	**9.99**	–
BLSTM [42]	–	25.11	–	26.61
AGCNN [43]	12.42	19.43	13.39	21.50
Vanilla RNN	19.88	29.77	19.92	28.95
Vanilla LSTM	16.89	22.43	15.79	26.25
RNN with two-stage attention	15.42	17.16	15.15	22.08
LSTM with two-stage attention	12.63	15.81	12.01	20.73

Table 7
Score for various methods on C-MAPSS dataset.

Method	FD001	FD002	FD003	FD004
MLP [24]	17,972	7,802,800	17,409	5,616,600
SVR [24]	1381.5	589,900	1598.3	371,140
ELM [27]	523.00	498149.97	573.78	121414.47
RF [27]	479.75	70456.86	711.13	46567.63
CNN [24]	1286.7	13,570	1596.2	7886.4
DBN [27]	417.59	9031.64	442.43	7954.51
MODBNE [27]	334.23	5585.34	421.91	6557.62
LSTM with a single attention [29]	322.44	–	–	5649.14
Semi-supervised setup [5]	231	3366	251	2840
HDNN [40]	245.00	1282.42	287.72	1527.42
1-FCLCNN-LSTM [41]	204.00	–	234.00	–
BLSTM [42]	–	4793	–	4971
AGCNN [43]	225.51	1492	227.09	3392
Vanilla RNN	500.75	4052.62	686.36	4986.38
Vanilla LSTM	313.234	2804.57	394.82	3023.88
RNN with two-stage attention	206.05	1525.68	212.83	2228.91
LSTM with two-stage attention	**153.36**	**975.35**	**156.83**	**1471.20**

5.1.2. Data preprocessing

Taking FD001 as an example, the readings of some sensors (i.e. sensors T2, P2, P15, epr, farB, Nf_dmd and PCNfR_dmd) remain constant during the whole life, which indicates that the corresponding features are irrelevant to health degradation. As a result, these sensor readings are removed from the original data and will not be involved in the RUL prediction process.

After removing these constant features, the other 14 features are defined as the model inputs. For data in train_FD001, the input features are normalized according to (6), and the corresponding RUL labels at different time steps are obtained and normalized by (5) and (7). To satisfy the requirements of model training and RUL prediction, sliding time window processing is performed, yielding the input-target pairs described as (8). We randomly select eighty percent of these samples for training and the rest twenty percent for validation to prevent model from overfitting. Conducting similar preprocessing procedures for test_FD001 and RUL_FD001, yields the testing set for RUL prediction. The other three sets can be preprocessed in the similar way.

5.2. Performance metrics

The loss function is utilized to evaluate and guide the model training process, which is calculated using the normalized targets and model outputs. In addition, two common metrics, i.e., root mean square error (RMSE) and score [24], are also adopted to further evaluate the final prediction performance of the proposed method in this paper. It should be noted that the two metrics are calculated using the target and predicted RUL values. Let ΔR_i denotes the difference between the predicted RUL and the actual RUL for the i-th sample, the score is given by

$$score = \sum_{i=1}^{N} s_i$$

$$s_i = \begin{cases} exp(-\Delta R_i/13) - 1, & \Delta R_i < 0 \\ exp(\Delta R_i/10) - 1, & \Delta R_i \geqslant 0 \end{cases}$$

The RMSE assigns equal weights to both early and late predictions. However, the score function imposes more penalization on late predictions than early predictions, as late predictions may lead to catastrophic consequences in practical engineering.

5.3. Hyperparameter setting

Since the proposed RUL prediction model is established based on deep neural networks, the hyperparameters of the model may influence its behavior, such as the time and memory cost of training the model, the quality of the model recovered by the training process and its ability to infer correct results when deployed on new inputs. Searching for good values of the hyperparameters is usually trial and error type. The main hyperparameters of the RUL prediction model are selected after numerous experiments.

Due to the limited space, the selection of the time window length is presented as an example. Considering the large computational cost of trial and error, five percent of the turbofan engines are randomly chosen and utilized to evaluate the performance of different time window lengths. The training process is repeated 10 times. Table 3 presents the average predicted performance of the repeated experiments on FD001. It is to be noted that the RMSE and score values in Table 3 are calculated using the predicted and target RUL values of the selected engines at all the time steps.

Actually, a shorter length of time window is not feasible to contain enough health degradation information, while a longer one increases the complexity of the model that may tend to cause overfitting and reduce the generalization performance. The time window of a moderate length, i.e. length of 30, achieves a better prediction performance, as shown in Table 3. Thus, the time window length is set to 30 in this paper. The other hyperparameters of the RUL prediction model can be determined by conducting the similar experiments. The values of the hyperparameters employed in this paper are listed in Table 4. As seen in Table 4, a relatively simple structure is designed for the sets FD001 and FD003, while a deeper structure is designed for the more challenging sets FD002 and FD004.

5.4. RUL prediction results and discussions

5.4.1. RUL prediction results

With the aforementioned hyperparameter setting, the architecture of the proposed prediction model can be determined. Then the model is trained by the Adam optimizer.

With the trained parameters, the multi-sensor data can be fused by the designed attention mechanisms. Fig. 7 presents the feature attention weights of two samples in FD001 as an example. Since the number of the input features is 14, and the time window length is 30, we obtain 14 × 30 attention weights for each sample. As seen in Fig. 7, the 14 input features correspond to different attention weights, which indicates different importance of these features. For each feature, the attention weights presents relatively slight variations at each time step, which is consistent with our common sense. In addition, the attention weights of different samples are also different. Thus, the feature attention weights provides a dynamic evaluation of the input features, with which the relevant multi-sensor data at each time step could be adaptively extracted.

As stated in the above section, the feature attention-based encoder maps the multi-sensor data to the encoder hidden states. Then the temporal attention mechanism is employed to evaluate the importance of these encoder hidden states. Fig. 8 presents the temporal attention weights of the above two samples. As seen in Fig. 8, the attention weights of the recent encoder hidden states are quite larger than those of the early ones. It indicates that the recent encoder hidden states contributes more to the final prediction, which is consistent with our common sense. Similar to the feature attention, the temporal attention weights are also different for different samples and different time steps. Thus, the temporal attention weights provides a dynamic evaluation of the encoder hidden states, with which the dynamic temporal correlation between different time intervals could be adaptively extracted.

Since the relevant feature information as well as the temporal correlation are extracted, the final RUL prediction results of the testing engines can be obtained. Examples of the RUL prediction results are visualized as shown in Fig. 9. The grey dotted lines represent a ten percent deviation from the maximum RUL value. As seen in Fig. 9, the predicted RUL values match the actual values well for different engines with different degrees of initial wear. The RUL prediction results are basically consistent with the actual values, especially in the last several cycles. In addition, the final RUL values of all the testing engines are also presented in Fig. 10. These results imply the effectiveness of the proposed RUL prediction approach.

5.4.2. Comparisons

To further demonstrate the effectiveness and necessity of the designed modules in the proposed prediction model, several comparison experiments are conducted. We utilize the vanilla RNN, the vanilla LSTM, and RNN with the proposed attention mechanisms for comparison. Firstly, taking FD001 as an example, the results of these models on the training set, the validation set and the testing set are shown in Table 5. In Table 5, all the samples on each set are considered to compute the values of the loss function and the RMSE, which facilitates the detailed comparison of their performance. The vanilla RNN performs worst owing to its problem of gradient disappearance. Without the feature and temporal attention mechanisms, the performance of both the vanilla RNN and the vanilla LSTM models is not satisfactory, especially on the testing set. Such results imply that the proposed attention mechanisms are of great significance to enhance the fitting accuracy and generalization ability of the prediction model. Due to the outstanding capability of learning long-term dependencies of the LSTM, the proposed model obtains better prediction results than the attention-based RNN model. Thus, it can be concluded that both the LSTM encoder-decoder module and the designed attention mechanisms play a crucial role in ensuring the RUL prediction performance.

There have been some state-of-the-art methods for RUL prediction proposed in publicly literatures. The prediction results of these methods on the C-MAPSS turbofan engine dataset have been reported, and their performance were generally evaluated by the RMSE and score of the final RUL values for the testing engines. According to this fact, the two metrics are computed to evaluate the proposed method, and the results are presented in Tables 6 and 7. In addition, the performance of the state-of-the-art methods and the aforementioned comparison methods are also listed in Tables 6 and 7.

In Tables 6 and 7, the bold values represent the best results compared with the others. Compared with [29] that also developing an attention-based deep learning framework, the proposed method presents better performance. Since only one attention mechanism was utilized and it was positioned after the LSTM layer, the model in [29] essentially treated the data from all the sensors equally, which limited the further improvement of its prediction accuracy. The comparison results with [29] imply once again that the proposed two-stage attention plays a crucial role in ensuring prediction performance.

As shown in the two tables, the performance of our proposed method on all the four subsets is outstanding in terms of the score. Although the HDNN and 1-FCLCNN-LSTM methods slightly outperforms the proposed method in terms of the RMSE, the proposed method is still better than most of the rest methods. Compared with other researches, the 1-FCLCNN-LSTM method used a relatively large time window, i.e., 50, which would bring outstanding performance in terms of the RMSE. However, in our work, the network structure is designed in a moderate scale to make a tradeoff between the prediction performance and the computational cost, which facilitates a fair comparison with most researches. A larger time window will further enhance our prediction performance, yet more hidden nodes and deeper structure are required.

Actually, the score function imposes more penalization on late predictions than early predictions, while the RMSE assigns equal weights to both early and late predictions. Since late predictions may lead to catastrophic consequences in practical engineering, we have considered both the two metrics rather than the single RMSE to determine the values of hyperparameters in our work. Our method provides better outcomes especially in terms of the score, thus it could be concluded that the proposed method is good promising as compared with other methods in engineering applications. The above results demonstrate the effectiveness and superiority of the proposed RUL prediction method once again.

6. Conclusions

This paper deals with the RUL prediction problem for complicated engineering systems. A novel end-to-end RUL prediction method is proposed based on deep learning. The salient contributions of the proposed method are as follows:

(1) A feature attention mechanism and a temporal attention mechanism are developed to focus on the more critical information from different sensors and at different time steps, respectively. This two-stage attention mechanism provides an efficient and practical way to extract the relevant features and temporal correlation adaptively, which avoids the requirement of prior knowledge and extra feature engineering.
(2) The proposed end-to-end prediction model directly maps the numerous data from multiple monitoring sensors to the RUL result. The model parameters can be jointly learned owing to the smooth and differentiable architecture, which avoids the intermediate error effectively.
(3) Owing to the proposed two-stage attention mechanism and the end-to-end architecture, the proposed method presents universality and adaptability for various systems.

Therefore, the proposed method can significantly improve the RUL prediction accuracy for complicated systems in comparison with the existing methods. Results of experiments have demonstrated the feasibility and superiority of the proposed method.

In future research, we will extend the proposed method to the application of RUL prediction for engineering systems under multiple fault modes. In addition, the issues of training efficiency and limited training data will also be considered.

CRediT authorship contribution statement

Yuyu Zhao: Conceptualization, Methodology, Software, Writing - original draft, Writing - review & editing. **Yuxiao Wang:** Investigation, Data curation, Supervision, Validation, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The research presented in this document is supported by the National Natural Science Foundation of China under Grant 62003352, and the Fundamental Research Funds for Central Universities, CAUC, under Grant 3122019053.

References

[1] M. Chookah, M. Nuhi, M. Modarres, A probabilistic physics-of-failure model for prognostic health management of structures subject to pitting and corrosion-fatigue, Reliab. Eng. Syst. Saf. 96 (2011) 1601–1610.
[2] B. Rezaeianjouybari, Y. Shang, Deep learning for prognostics and health management: State of the art, challenges, and opportunities, Measurement (163) (2020) 107929.
[3] H. Li, H.Z. Huang, Y.F. Li, et al., Physics of failure-based reliability prediction of turbine blades using multi-source information fusion, Appl. Soft Comput. 72 (2018) 624–635.
[4] S.K. Zeng, M. Pecht, J. Wu, Status and perspectives of prognostics and health management technologies, Acta Aeronaut. Astronaut. Sin. 26 (2005) 626–632.
[5] A.L. Ellefsen, E. Bjorlykhaug, V. Aesoy, et al., Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture, Reliab. Eng. Syst. Saf. 183 (2019) 240–251.
[6] M. Pecht, R. Jaai, A prognostics and health management roadmap for information and electronics-rich systems, Microelectron. Reliab. 50 (2010) 317–323.
[7] L. Liu, X. Song, K. Chen, et al., An enhanced encoder–decoder framework for bearing remaining useful life prediction, Measurement 170 (2020), 108753.
[8] Y.L. Bi, Y.L. Yin, S.Y. Choe, Online state of health and aging parameter estimation using a physics-based life model with a particle filter, J. Power Sources 476 (2020), 228655.
[9] K. Kuhn, R.S. Fertig, A physics-based fatigue life prediction for composite delamination subject to mode i loading, in: Proceedings of the American Society for Composites – 31st Technical Conference, Williamsburg, VA, United states, 2016.
[10] T. Sutharssan, S. Stoyanov, C. Bailey, et al., Prognostic and health management for engineering systems: a review of the data-driven approach and algorithms, J. Eng. 7 (2015) 215–222.
[11] Y. Peng, J. Cheng, Y. Liu, et al., An adaptive data-driven method for accurate prediction of remaining useful life of rolling bearings, Front. Mech. Eng. 13 (2018) 301–310.
[12] Z. Zhang, C. Hu, X. S, et al., Degradation modeling and remaining useful life prediction with bivariate time scale, Acta Automat. Sin. 43 (2017) 1789–1798.
[13] J. Hu, Q. Sun, Z. Ye, et al., Joint modeling of degradation and lifetime data for RUL prediction of deteriorating products, IEEE Trans. Ind. Inf. (2020), https://doi.org/10.1109/TII.2020.3021054.
[14] H. Wang, W. Song, E. Zio, et al., Remaining useful life prediction for lithium-ion batteries using fractional Brownian motion and fruit-fly optimization algorithm, Measurement 161 (2020), 107904.
[15] X. Xi, M. Chen, D. Zhou, Remaining useful life prediction for degradation processes with memory effects, IEEE Trans. Reliab. 66 (2017) 751–760.
[16] H. Yang, Z. Sun, G. Jiang, et al., Remaining useful life prediction for machinery by establishing scaled-corrected health indicators, Measurement 163 (2020), 108035.
[17] Z. Chen, S. Cao, Z. Mao, Remaining useful life estimation of aircraft engines using a modified similarity and supporting vector machine (SVM) approach, Energies 11 (2017) 28.
[18] P.G. Nieto, E. Garcia-Gonzalo, F.S. Lasheras, et al., Hybrid PSO-SVM-based method for forecasting of the remaining useful life for aircraft engines and evaluation of its reliability, Reliab. Eng. Syst. Saf. 138 (2015) 219–231.
[19] R. Khelif, B. Chebel-Morello, S. Malinowski, et al., Direct remaining useful life estimation based on support vector regression, IEEE Trans. Ind. Electron. 64 (2017) 2276–2285.
[20] K. Javed, R. Gouriveau, N. Zerhouni, A new multivariate approach for prognostics based on extreme learning machine and fuzzy clustering, IEEE Trans. Cybern. 45 (2015) 2626–2639.
[21] D. Wu, C. Jennings, J. Terpenny, et al., A comparative study on machine learning algorithms for smart manufacturing: Tool wear prediction using random forests, J. Manuf. Sci. Eng. 139 (2017), 071018.
[22] K. Greff, R.K. Srivastava, J. Koutník, et al., LSTM: a search space odyssey, IEEE Trans. Neural Networks Learn. Syst. 28 (2017) 2222–2232.
[23] D. Holden, J. Saito, T. Komura, A deep learning framework for character motion synthesis and editing, ACM Trans. Graphics 35 (2016) 138.
[24] G. Sateesh Babu, P. Zhao, X. Li, Deep convolutional neural network based regression approach for estimation of remaining useful life, in: Proceedings of 21st International Conference on Database Systems for Advanced Applications, 2016, pp. 214–228.
[25] T. Hu, J. Yu, Life prediction of lithium-ion batteries based on multiscale decomposition and deep learning, J. Zhejiang Univ. (Eng. Sci.) 53 (2019) 1852–1863.
[26] J. Li, Y. Chen, H. Xiang, et al., Remaining useful life prediction for aircraft engine based on LSTM-DBN, Syst. Eng. Electron. 42 (2020) 1637–1644.
[27] C. Zhang, P. Lim, A. Qin, et al., Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics, IEEE Trans. Neural Networks Learn. Syst. 28 (2017) 2306–2318.
[28] Y.W. Lu, C.Y. Hsu, K.C. Huang, An autoencoder gated recurrent unit for remaining useful life prediction, Processes 8 (2020) 1155.
[29] Z. Chen, M. Wu, R. Zhao, et al., Machine remaining useful life prediction via an attention based deep learning approach, IEEE Trans. Ind. Electron. 68 (2021) 2521–2531.
[30] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780.
[31] J.H. Bappy, C. Simons, L. Nataraj, et al., Hybrid LSTM and encoder-decoder architecture for detection of image forgeries, IEEE Trans. Image Process. 28 (2019) 3286–3300.
[32] K. Xu, J.L. Ba, R. Kiros, et al., Show, attend and tell: Neural image caption generation with visual attention, in: Proceedings of 32nd International Conference on Machine Learning, 2015, pp. 2048–2057.
[33] A.M. Rush, S. Chopra, J. Weston, A neural attention model for abstractive sentence summarization, in: Proceedings of EMNLP, 2015.
[34] Y. Li, Z. Zhu, D. Kong, et al., EA-LSTM: Evolutionary attention-based LSTM for time series prediction, Knowl.-Based Syst. 181 (2019), 104785.
[35] Q. Yao, D. Song, H. Chen, et al., A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 2627–2633.
[36] I. Sutskever, O. Vinyals, Q. Le, Sequence to sequence learning with neural networks, Advances in Neural Information Processing Systems 4 (2014) 3104–3112.
[37] R.N. Singarimbun, E.B. Nababan, O.S. Sitompul, Adaptive moment estimation to minimize square error in backpropagation algorithm, in: Proceedings of International Conference of Computer Science and Information Technology, Medan, Indonesia, 2019.
[38] A. Saxena, K. Goebel, Turbofan engine degradation simulation dataset NASA Ames Prognostics Data Repository (http://ti.arc.nasa.gov/project/prognostic-data-repository), NASA Ames Research Center, Moffett Field, CA, 2008.
[39] A. Saxena, G. Ka, D. Simon, et al, Damage propagation modeling for aircraft engine run-to-failure simulation, in: Proceedings of International Conference on Prognostics and Health Management, 2008.
[40] A. Al-Dulaimi, S. Zabihi, A. Asif, et al., A multimodal and hybrid deep neural network model for Remaining Useful Life estimation, Comput. Ind. 108 (2019) 186–196.
[41] C. Peng, Y.F. Chen, Q. Chen, et al., A remaining useful life prognosis of turbofan engine using temporal and spatial feature fusion, Sensors 21 (2021) 418.
[42] C.G. Huang, H.Z. Huang, Y.F. Li, A bidirectional LSTM prognostics method under multiple operational conditions, IEEE Trans. Ind. Electron. 66 (2019) 8792–8802.
[43] H. Liu, Z.Y. Liu, W.Q. Jia, et al., Remaining useful life prediction using a novel feature-attention-based end-to-end approach, IEEE Trans. Ind. Inf. 17 (2021) 1197–1207.