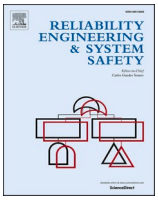
Reliability Engineering and System Safety 215 (2021) 107871

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

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



# Remaining useful life prediction based on intentional noise injection and feature reconstruction

Lei Xiao [a,*], Junxuan Tang [a], Xinghui Zhang [b], Eric Bechhoefer [c], Siyi Ding [a]

[a] College of Mechanical Engineering, Donghua University, Shanghai 201620, China
[b] China North Vehicle Research Institute, Fengtai District, Beijing, 100072, China
[c] Green Power Monitoring Systems, Inc, VT 05753, USA

ARTICLE INFO

*Keywords:*
Remaining useful life
Feature reconstruction
Noise injection
Aero-engines

ABSTRACT

The accurate remaining useful life (RUL) prediction is the foundation of prognostics and health management (PHM). The accuracy of RUL prediction model depends on not only the quality and quantity of degradation feature but also the prediction model. In most of the existing deep-learning based RUL prediction models, noise is considered harmful and has to be removed. Further, the correlation among sensory measurements is ignored. However, noise can boost the prediction performance if judiciously used. This paper proposes a new RUL prediction method where noise is intentionally added into a long short-term memory (LSTM) network. Additionally, correlation analysis is conducted among the sensory measurements to construct new degradation features as the inputs of the LSTM network. Validation of the proposed method was carried out on the C-MAPSS aero-engine lifetime dataset. Finally, the proposed RUL prediction model is compared to other the-state-of-the-art techniques.

## 1. Introduction

Modern industrial manufacturing requires a high level of reliability and availability to be profitable. Prognostics and health management (PHM) is an effective way to manage equipment usage and guarantee its reliability. In PHM, prognostics facilitates providing an estimation of the remaining useful life (RUL) of degrading equipment [1]. PHM is the foundation of making an informed balance of plant decisions to avoid production breakdowns, reduce maintenance costs, and achieve availability [2]. Given its importance, the RUL prediction for critical equipment has been widely investigated.

Among the existing data-driven approaches, many stem from deep learning (DL) techniques, such as Convolutional Neural Networks (CNN), Deep Belief Network (DBN), and Long Short-Term Memory (LSTM) neural networks and their variants, have been successfully applied to RUL prediction [3-12]. Kim and Liu [13] proposed a Bayesian deep learning framework for interval estimation of RUL. Xia et al. [13] developed an ensemble framework for RUL estimation based on convolutional bi-directional LSTM with multiple time windows. Xiao et al. [14] designed a RUL prediction method based on LSTM network to solve the prediction problem under different conditions. The LSTM network, which is based on recurrent neural network (RNN), can be regarded as a directed graph with cycles or feedbacks. LSTM network makes an output stream from the incipient output and the internal memory state. Due to the mechanism, LSTM network and its variants have been widely adopted to predict RULs for engineered systems [5-6, 10-11].

During the training of an RNN, recurrent backpropagation (RBP) is usually adopted to optimize the connection weights and parameters according to some given performance measures [15]. Backpropagation (BP) can be regarded as a special case of the generalized expectation maximization (EM) algorithm [16]. According to the noise EM (NEM) theorem, if noise is added to the system, the performance of EM can be improved [17]. Many theoretical studies and experimental results have demonstrated that injecting noise into neural networks can speed up the convergence of the training process and enhance the output from neural networks [16, 18-21]. The noise can be separately injected into the input, hidden and output layers of a neural network [16, 18, 22]. However, there is few precedent for the RUL prediction boosted by the injection of intentional noise into a neural network. In most existing RUL prediction methods, noise, which is generated by measurement errors or interaction among components, is always regarded as superfluous and removed from the observations. However, this ignores the performance benefit of whiting from adding noise. In the field of PHM, it has been demonstrated that noise injection can boost the detection performance

* Corresponding author.
*E-mail address:* leixiao211@dhu.edu.cn (L. Xiao).

https://doi.org/10.1016/j.ress.2021.107871
Received 7 September 2020; Received in revised form 27 May 2021; Accepted 14 June 2021
Available online 23 June 2021

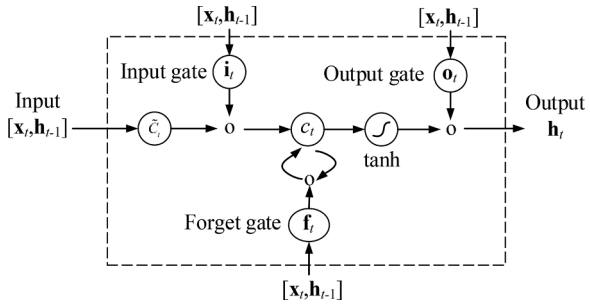



**Fig. 1.** A LSTM unit

of faults for engineered systems [23-27]. However, the attempt of RUL prediction in consideration of noise injection is few. This paper is a preliminary attempt. A new RUL prediction model is proposed and boosted by the intentional noise injection based on the NEM theorem.

The accuracy of RUL prediction results is not only impacted by the prediction model, but also the inputs of the prediction model. The inputs can be in different forms, such as soft processing of condition monitoring data. Soft processing means smoothing and normalization. Smoothing aims at removing measurement errors from the condition monitoring data. Normalization aims at avoiding the impacts caused by different orders of magnitudes between data inputs on the prediction models [3-4].

The other type of inputs to an RUL prediction model is performing feature extraction. The intent of the extracted features is to reflect the degradation of equipment. The commonly extracted features include time-domain features, frequency-domain spectra, and time-frequency-domain features [28, 29]. The third type of inputs is dimensionality reduction of the feature set. One of the common ways to feature-dimension reduction is feature fusion [30-31]. Other common methods about feature-dimension reduction are the adoption of principal component analysis (PCA) and its variants [5, 32]. Even though these degradation features are originally obtained from the same sensory measurements, the contribution to degradation/health reflection is inequal [31]. Thanks to the concern, a feature fusion-selection method is proposed in this paper.

To improve the accuracy of RUL prediction, this paper demonstrates a new RUL prediction method. This method is divided into two phases, which are named as the feature reconstruction and noisy prediction, respectively. In the first phase, the correlation among sensory measurements is analyzed by using correlation analysis and hypothesis testing. The features selected from correlation analysis and hypothesis testing are used to construct new features. The selected features are sensitive to degradation reflection and have high trendability. The selected features are fed into an LSTM network. During training of the network, white noise is conditionally and intentionally injected into the output layer of the LSTM network based on the NEM theorem. This "pollutes" the output during the training process. Therefore, the RUL prediction method is named as *noisy prediction* for the second phase. This is also the main difference from most of existing deep learning-based RUL prediction methods. Though noise is injected into the network, the prognosis is robust, and the RUL prediction accuracy is improved. Injecting noise into a neural network is an effective and convenient way to improve the robustness even though the parameters and models are not fully and well optimized. Usually, parameter optimization is a complex and time-consuming work. The main contributions of this paper are summarized:

- This paper is a preliminary attempt that a new RUL prediction model is proposed and boosted by the intentional noise injection based on the NEM theorem. The noise is injected into the output layer of an LSTM network, deliberately "whitening" the output.
- The correlation among sensory measurements is analyzed, rather than between sensory measures and RUL. New degradation features are constructed according to the results of correlation analysis. Besides, features are selected according to the contribution to trendability.

The residual paper is organized as follows. Section 2 introduces the basic theory about the LSTM network. The proposed RUL prediction method is described in Section 3. Validation by using the C-MAPSS dataset about engine lifetime is conducted in Section 4. The proposed method is compared with other methods in Section 5. The whole paper is concluded in Section 6.

## 2. The basic theory of LSTM

The primary temporal modeling tool in this study is the LSTM network for time sequence information modeling. LSTM regulates the path of information flow by gating mechanism as shown in Fig. 1 according to Ref. [33]. Due to the gating mechanism, an LSTM unit can remove or add information to the cell state [6]. In Fig. 1, $\mathbf{x}_t$ denotes the inputs of an LSTM unit at the current time $t$. $\mathbf{h}_t$ indicates the output of the LSTM unit at the current time. $\widetilde{\mathbf{c}}_t$ and $\mathbf{c}_t$ represent the candidate state of memory unit and its updated internal state at current time $t$, respectively. $\mathbf{i}_t$, $\mathbf{f}_t$ and $\mathbf{o}_t$ stand for the input, forget, and output gates, respectively.

The forget gate determines which data will be dropped out from the cell state. The input gate decides which states will be updated. Output gate selects which part of the cell states will be outputted. In an LSTM unit, the cell state is also important and related to the output from the LSTM unit. The information process in the gates can be formulated as follows,

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

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

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

$$\mathbf{c}_t = \mathbf{f}_t * \mathbf{c}_{t-1} + \mathbf{i}_t * \widetilde{\mathbf{c}}_t, \tag{4}$$

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

$$\mathbf{h}_t = \mathbf{o}_t * \tanh(\mathbf{c}_t), \tag{6}$$

where $\mathbf{W}_{\mathrm{f}}$, $\mathbf{W}_{\mathrm{i}}$, $\mathbf{W}_{\mathrm{c}}$, and $\mathbf{W}_{\mathrm{o}}$ represent the input weights of the forget gate, input gate, cell state layer, and output gate, respectively. $\mathbf{R}_{\mathrm{f}}$, $\mathbf{R}_{\mathrm{i}}$, $\mathbf{R}_{\mathrm{c}}$, and $\mathbf{R}_{\mathrm{o}}$ stand for the recurrent weights of the forget gate, input gate, cell state layer, and output gate, respectively. $\mathbf{b}_{\mathrm{f}}$, $\mathbf{b}_{\mathrm{i}}$, $\mathbf{b}_{\mathrm{c}}$ and $\mathbf{b}_{\mathrm{o}}$ denote the bias. The notation "•" and "*" mean the matrix multiplication and element-wise multiplication.

## 3. The proposed RUL prediction method

The framework of the proposed RUL prediction method based on the LSTM network, noise injection, and feature reconstruction is illustrated in Fig. 2. The proposed method is divided into two phases which are named as feature reconstruction and noisy prediction, respectively.

### 3.1. Feature reconstruction

Condition monitoring is conducted to collect the data to reflect machine degradation. The raw condition monitoring data needs some essential preprocesses, such as constant value remove. The constant values do not reflect the machine degradation and are meaningless for a time-series regression problem. If the constant values are imported into a neural network, they will generate negative impact on the output. Then smoothing is conducted to eliminate the measurement error of raw

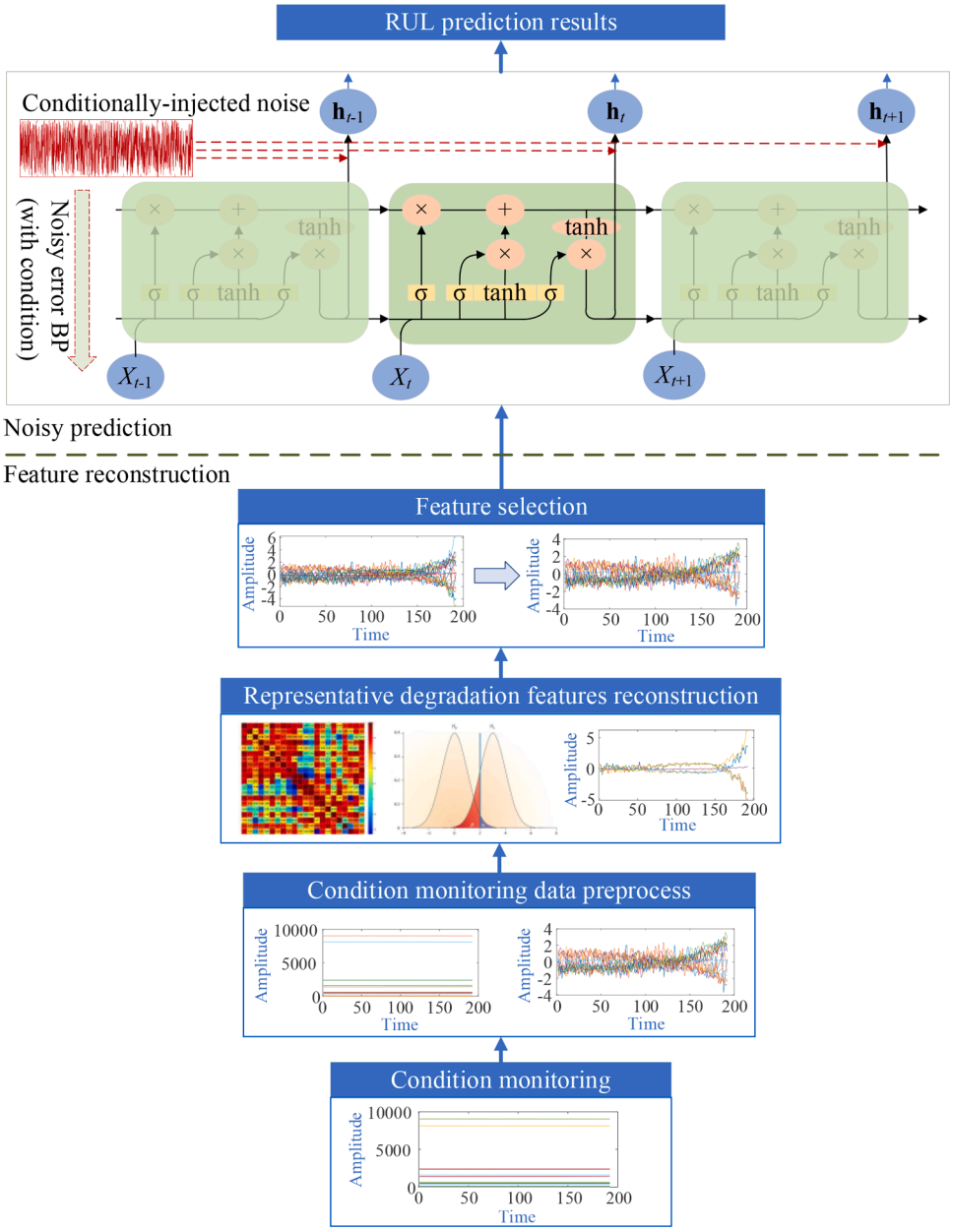



**Fig. 2.** The framework of the proposed RUL prediction method

condition monitoring data by moving mean algorithm. After smoothing, Z-score normalization as Eq. (7) is conducted to eliminate the impact caused by a broad range of orders of magnitudes of condition monitoring data.

$$x_l^* = \frac{x_l - \mu_l}{\sigma_l}, \tag{7}$$

In Eq. (7), $x_l$* denotes the $l^{\text{th}}$ degradation feature $x_l$ after normalization, $l$=1,2, …, $L$. Here, $L$ represents the total number of degradation features from raw condition monitoring data. The notation $\mu_l$ and $\sigma_l$ denote the mean value and the standard deviation of the $l^{\text{th}}$ degradation feature matrix, respectively.

Correlation analysis is conducted on the normalized features. This is also different from the prior RUL prediction method. The correlation coefficients among the features are calculated as follows,

$$\rho\left(x_{l_1}^*, x_{l_2}^*\right) = \frac{\sum\left(x_{l_1}^* - \overline{x_{l_1}^*}\right)\left(x_{l_2}^* - \overline{x_{l_2}^*}\right)}{\sqrt{\sum\left(x_{l_1}^* - \overline{x_{l_1}^*}\right)^2\left(x_{l_2}^* - \overline{x_{l_2}^*}\right)^2}}, \tag{8}$$

where $\rho(x_{l_1}^*, x_{l_2}^*)$ denotes the correlation coefficient of the $l_1^{\text{th}}$ and $l_2^{\text{th}}$ normalized degradation features.

In this paper, the correlation coefficient is valid only if the correlation is significant. Hypothesis testing is conducted after each correlation analysis to quantify this assumption. If the result from hypothesis testing is smaller than the significance level (default is 0.05), then the correlation between the two degradation feature vectors is considered significant.

Predetermined values are given for the correlation analysis and hypothesis testing. If the calculated correlation coefficient is greater than the given value and the $p$-value obtained from the hypothesis testing is less than a threshold, the degradation features are considered to be significantly correlated. Then they are operated by element-wise

multiplication to construct new degradation features as,

$$x_{\text{new}} = x^*_{l_1} * x^*_{l_2} \tag{9}$$

where $x_{\text{new}}$ denotes the newly constructed degradation features through the normalized degradation features $x^*_{l_1}$ and $x^*_{l_2}$. Here, the "element-wise multiplication" operations can enhance or weaken the correlation characteristics between the selected degradation features $x^*_{l_1}$ and $x^*_{l_2}$. After multiplication, more tendency information behind the degradation features is mined.

It is worth noting that the newly constructed degradation feature also needs to be normalized to maintain the same scale of the other degradation features. The normalization algorithm is also Z-score. After normalization, $x^*_{\text{new}}$ denotes the normalized degradation feature which is newly constructed.

Suppose that there are $M$ degradation features including the original and newly constructed features after correlation analysis and hypothesis testing. But not all the degradation features are used, because they have different time sensitivities and trendability for a time-serial prediction problem. The features which have less contribute to RUL prediction should be removed. A new trendability function is defined to select the proper features for the prognosis. In Ref. [34], trendability is used to measure how well each parameter in a population which can be described by the same underlying function. The trendability function deploys the correlation analysis and its calculation is given as follows according to Ref. [34],

$$C_{\text{Trendability}} = \min\left(\left|\rho\left(x^*_m, x^*_{m''}\right)\right|\right) \tag{10}$$

where $C_{\text{Trendability}}$ means the trendability coefficient. $m$=1, 2, …, $M$, $m''$=1, 2, …, $M$.

To make the trendability function benefit for feature selection, $C_{\text{Trendability}}$ is redefined in this paper as,

$$C^m_{\text{Trendability}} = \max\left(\left|\rho\left(x^*_m, x^*_{m'}\right)\right|\right), \qquad \text{if } m = m'', \text{ then } \rho\left(x^*_m, x^*_{jm''}\right) = 0 \tag{11}$$

where $m$=1, 2, …, $M$, $m''$=1, 2, …, $M$. Iteration is conducted on $m$ to compare the correlation between the $m^{\text{th}}$ feature and the rest features. The maximum absolute value of correlation coefficients obtained from the $m^{\text{th}}$ features compared with the rest features is selected as the trendability coefficient of the $m^{\text{th}}$ feature.

Noting that, correlation coefficient can be positive or negative, therefore, the absolute operator is considered in Eq. (11). The greater absolute value of $C^i_{\text{Trendability}}$ means more correlation between the current feature with other feature(s). Then all the correlation coefficients are sorted in descending order. The contribution to correlation representation of the $m^{\text{th}}$ degradation feature then calculated as:

$$w_m = \frac{C^m_{\text{Trendability}}}{\sum_{m=1}^{M} C^m_{\text{Trendability}}}, \tag{12}$$

where $w_m$ denotes the weight of correlation representation of the sorted $m^{\text{th}}$ degradation feature.

The weights of correlation representation are summed, and a predetermined value is given to select the degradation features as follows,

$$\arg \sum_{m=1}^{m^*} w_m \geq w_{\text{given}} \tag{13}$$

where $w_{\text{given}}$ denotes the critical value of the weights. There are $m^*$ degradation features that are selected in total, and the summation of the weights from the $m^*$ degradation features is greater than $w_{\text{given}}$. In fact, there is no theoretical standard or empirical guidance about how to determine $w_{\text{given}}$. The determination of $w_{\text{given}}$ is a trade-off between prediction result accuracy and training time consumption of neural network. If $w_{\text{given}}$ is set to a greater value, more degradation features are selected as the input samples for training a LSTM network. Thus, more accurate results can be obtained but the training process of the network is time-consuming. Otherwise, if $w_{\text{given}}$ is set to a smaller value, less degradation features are selected for training a LSTM network and the training process of the network is shortened but the results accuracy may be impacted.

The other degradation features are abandoned as they have less ability to be correlative with other degradation features, namely, they are hard to be described by the same underlying function. They will not be used for training a LSTM network or predicting RULs of the machines. There are ($M$-$m^*$) degradation features ignored, which may be from the original raw condition monitoring data or the newly constructed degradation features. The process of feature construction and selection is formed as Algorithm 1.

**Data:** $L$ normalized degradation feature set $\{x^*_1, x^*_2, \cdots, x^*_L\}$ from the raw condition monitoring data. The predetermined $\rho_{\text{given}}$ for correlation analysis, the predetermined $p_{\text{given}}$ value for hypothesis testing of the significant check, and the predetermined $w_{\text{given}}$ for feature selection.

**Result:** The selected degradation features, including the original condition indexes which are extracted from raw condition monitoring data and the newly constructed ones.

It is noting that there is no theoretical standard for the determination of $\rho_{\text{given}}$ and $p_{\text{given}}$. However, there is the empirical guidance. In terms of $\rho_{\text{given}}$, it is usually set to greater than 0.7 in Pearson correlation analysis. If the obtained correlative coefficient has a greater value, the correlation between the two elements is stronger. In this paper, the threshold $\rho_{\text{given}}$ is set to 0.9. $p_{\text{given}}$ is usually set to 0.1, 0.05 or 0.01. In this paper, it is set to 0.05. If the obtained $p$-value is less than $p_{\text{given}}$ and the obtained $\rho$-value is greater than $\rho_{\text{given}}$, it means that the two different degradation features are significantly correlative.

### 3.2. Noisy prediction

The RUL prediction can be regarded as a regression problem with respect to the time-series degradation features. In an LSTM regression network, the activation function at the output layer usually adopts the identity function. The target vector $\mathbf{y}$, which is the RUL matrix of machines in this paper, is a Gaussian random vector with mean vector $\mathbf{a}^{\mathbf{y}}$ and identity covariance matrix $\mathbf{I}$. According to Ref. [18], the likelihood function of a regression neural network is as follows,

$$p(\mathbf{y}|\mathbf{x}^*, \boldsymbol{\theta}) = \frac{1}{(2\pi)^{K/2}} \exp\left\{ -\frac{\| \mathbf{y} - \mathbf{a}^{\mathbf{y}} \|^2}{2} \right\}, \tag{14}$$

where $||\bullet||$ denotes the Euclidean norm, $K$ denotes the number of output neurons, $\boldsymbol{\Theta}$ denotes the parameter set of the neural network, and $\mathbf{x}^*$ denotes the normalized selected degradation features.

The squared error $E$ of regression in the network can be calculated as follows,

$$E = \frac{1}{2} \sum_{k=1}^{K} \left(y_k - a^i_k\right)^2, \tag{15}$$

where $a^i_k$ denotes the output from the $k^{\text{th}}$ neuron in the output layer at the $i^{\text{th}}$ epoch.

As noted, noise can be beneficial to the result from the LSTM network if it is carefully injected into the network. This condition can be formulated as follows,

$$E_{\mathbf{y},\mathbf{h},\mathbf{n}|\mathbf{x},\boldsymbol{\theta}} - E_{\mathbf{y},\mathbf{h}|\mathbf{x},\boldsymbol{\theta}} \leq 0, \tag{16}$$

where $\mathbf{n}$ denotes the intentionally-added noise matrix.

Combining Eq. (14) with Eq. (15), the inequality condition which is Eq. (16) is rewritten as follows,

$$\frac{1}{2} \sum_{k=1}^{K} \left(y_k + n_k - a^i_k\right)^2 - \frac{1}{2} \sum_{k=1}^{K} \left(y_k - a^i_k\right)^2 \leq 0, \tag{17}$$

where $n_k$ denotes the intentionally-added noise for the $k^{\text{th}}$ neuron.

**Table 1**
The C-MAPSS dataset of turbofan engines

| Dataset | FD001 | FD002 | FD003 | FD004 |
|---|---|---|---|---|
| Train trajectories | 100 | 260 | 100 | 249 |
| Test trajectories | 100 | 259 | 100 | 248 |
| Maximum life span (cycles) | 362 | 378 | 525 | 543 |
| Minimum life span (cycles) | 128 | 128 | 145 | 128 |
| Operating conditions | 1 | 6 | 1 | 6 |
| Fault conditions | 1 | 1 | 2 | 2 |

**Proof.** The noise is beneficial to the prediction results from a neural network if the following positivity condition holds according to Ref. [18].

$$E_{\mathbf{y},\mathbf{h},\mathbf{n}|\mathbf{x},\boldsymbol{\theta}}\left[\ln\frac{p(\mathbf{y}+\mathbf{n},\mathbf{h}|\mathbf{x},\boldsymbol{\theta}^{(i)})}{p(\mathbf{y},\mathbf{h}|\mathbf{x},\boldsymbol{\theta}^{(i)})}\right]\leq 0. \tag{18}$$

According to Bayesian probability,

$$\frac{p(\mathbf{y}+\mathbf{n},\mathbf{h}|\mathbf{x},\boldsymbol{\theta}^{(i)})}{p(\mathbf{y},\mathbf{h}|\mathbf{x},\boldsymbol{\theta}^{(i)})}=\frac{p(\mathbf{y}+\mathbf{n}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)})p(\mathbf{h}|\mathbf{x},\boldsymbol{\theta}^{(i)})}{p(\mathbf{y}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)})p(\mathbf{h}|\mathbf{x},\boldsymbol{\theta}^{(i)})}=\frac{p(\mathbf{y}+\mathbf{n}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)})}{p(\mathbf{y}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)})}. \tag{19}$$

Taking logarithm to Eq. (18),

$$\ln\frac{p(\mathbf{y}+\mathbf{n}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)})}{p(\mathbf{y}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)})}=\ln p\left(\mathbf{y}+\mathbf{n}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)}\right)-\ln p\left(\mathbf{y}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)}\right). \tag{20}$$

Substituting Eq. (13) into Eq. (19),

$$\ln p\left(\mathbf{y}+\mathbf{n}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)}\right)-\ln p\left(\mathbf{y}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)}\right)=\ln\left[2\pi^{-\frac{K}{2}}\exp\left\{-\frac{\|\mathbf{y}+\mathbf{n}-\mathbf{a}^{\mathbf{y}}\|^2}{2}\right\}\right]-\ln\left[2\pi^{-\frac{K}{2}}\exp\left\{-\frac{\|\mathbf{y}-\mathbf{a}^{\mathbf{y}}\|^2}{2}\right\}\right]. \tag{21}$$

Rewritten Eq. (20),

$$\ln\frac{p(\mathbf{y}+\mathbf{n}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)})}{p(\mathbf{y}|\mathbf{h},\mathbf{x},\boldsymbol{\theta}^{(i)})}=\frac{\|\mathbf{y}-\mathbf{a}^{\mathbf{y}}\|^2}{2}-\frac{\|\mathbf{y}+\mathbf{n}-\mathbf{a}^{\mathbf{y}}\|^2}{2}=\frac{\mathbf{n}^T}{2}(2\mathbf{a}^{\mathbf{y}}-2\mathbf{y}-\mathbf{n}). \tag{22}$$

Therefore, the noise is beneficial if the following inequality holds:

$$E_{\mathbf{y},\mathbf{h},\mathbf{n}|\mathbf{x},\boldsymbol{\theta}^*}\left[\mathbf{n}^T(2\mathbf{a}^{\mathbf{y}}-2\mathbf{y}-\mathbf{n})\right]\leq 0. \tag{23}$$

No matter which optimization algorithm is used to update the parameter set $\boldsymbol{\Theta}$ in a neural network, the squared error of regression is essential. As illustrated in Fig. 2, the selected degradation features should be imported into the LSTM network to train and predict the RULs of machines. During the training process, the LSTM network is conditionally "whitened" by intentionally adding noise. The noise injection satisfies the NEM theorem, which is proofed above. To clarify, Algorithm 2 gives the update rule during the given epochs of the training process, focusing on the RUL prediction with intentionally added noise.

**Data:** $D$ input sets$\{\mathbf{X}_1,\mathbf{X}_2,\cdots,\mathbf{X}_D\}$ of selected degradation features where an input set$\mathbf{X}_d=\{\mathbf{x}_d^{(1)},\mathbf{x}_d^{(2)},\cdots,\mathbf{x}_d^{(T)}\}$. The corresponding target output sets $\{\mathbf{Y}_1,\mathbf{Y}_2,\cdots,\mathbf{Y}_D\}$ of machine RULs were a target output set $\mathbf{Y}_d=\{\mathbf{y}_d^{(1)},\mathbf{y}_d^{(2)},\cdots,\mathbf{y}_d^{(T)}\}$. Where the number of epochs $I$, the number of iteration $J$, the lifetime of a machine $T$, and the batch size $M$.

In Algorithm 2, both the forward pass calculation and the back-propagation calculation are related to the structure of the LSTM network. To reduce the processing time and decrease the calculation complexity, a one-layer LSTM network is adopted, as the deeper structure of the LSTM network needs more time for training. For a rapid and accurate RUL prediction, the optimization algorithm used for the network is the root mean square prop (RMSprop).

## 4. Validation

### 4.1. Data description

The popular C-MAPSS dataset (a simulated turbofan engine dataset provided by NASA [35]) is used to validate the effectiveness of the proposed method. The information about the dataset is listed in Table 1. Each sub-dataset can be further divided into train and test trajectories. Each trajectory indicates the cycle records of an engine which are the snapshots of data taken during its operational cycle. A cycle record is 24-dimensional including 3-dimensional operational settings and 21-dimensional sensor measurements.

In this paper, two indicators are used to evaluate the performance of the proposed prediction method. Eq. (24) stresses that the early prediction is better than the late prediction. Eq. (25) which means the root mean square error (RMSE) is also widely used to evaluate the performance of a regression model. In Eqs. (24)-(25), $P_d$ and $R_d$ denote the predicted and associated real RULs of Engine $d$.

$$E_{\text{score}}=\begin{cases}\sum_{d=1}^{D}\exp\left(-\frac{P_d-R_d}{13}\right)-1 & \text{if } P_d-R_d<0\\ \sum_{d=1}^{D}\exp\left(\frac{P_d-R_d}{10}\right)-1 & \text{if } P_d-R_d>0\end{cases} \tag{24}$$

$$E_{\text{RMSE}}=\sqrt{\frac{1}{d}\sum_{d=1}^{D}(P_d-R_d)^2} \tag{25}$$

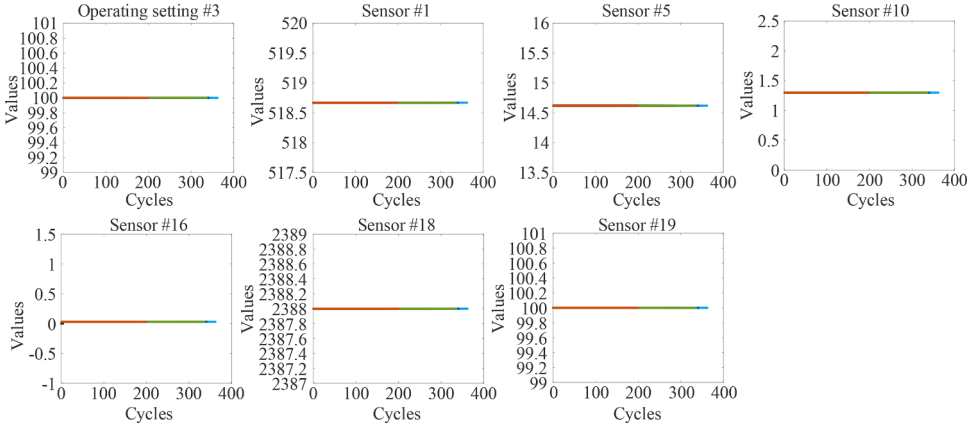



**Fig. 3.** Constant measurements of all the engines in the training dataset of FD001

**Table 2**
The dimension change of degradation features in different sub-datasets

| Dataset | FD001 | FD002 | FD003 | FD004 |
|---|---|---|---|---|
| Dimension of the data from the original dataset | 24 | 24 | 24 | 24 |
| Dimension of raw degradation features | 17 | 24 | 18 | 24 |
| Dimension of constructed degradation features | 5 | 106 | 7 | 106 |
| Dimension of the inputs of an LSTM network | 19 | 124 | 22 | 124 |

### 4.2. Approach validation

As noted in Section 3, constant bias values in the operating settings and sensory measurements should be removed. In terms of FD001, the third operating setting and the sensory measurements from Sensors 1, 5, 10, 16 18, and 19 should be removed. The constant records from all the engines in the training dataset are shown in Fig. 3 where the colored lines stand for the measurements from different engines during their lifetimes. Only 17-dimensional records, including operating settings and sensory measurements, are remained. In terms of FD003, the third operating setting and the sensory measurements from Sensors 1, 5, 16, 18, and 19 should be removed. Therefore, only 18-dimensional records, including operating settings and sensory measurements, are used. All the operating settings and sensory measurements of the engines from FD002 and FD004 are retained (i.e., no elements are removed from FD002 and FD004).

The remained operating settings and sensory measurements can be regarded as the raw degradation features of engines. Since they are extracted from raw condition monitoring data. These raw degradation features are conducted by one-step moving average for smoothing and Z-score normalization to make the orders of magnitudes of the degradation features at the same level.

Correlation analysis is conducted on the raw degradation features. The given $\rho_{\text{given}}$ is set to 0.9, and the given $p_{\text{given}}$ is set to 0.05. Therefore, if a correlation coefficient from two different raw degradation features is greater than 0.9 and the corresponding result from hypothesis testing is less than 0.05, it indicates the two raw degradation features are significantly correlated. Then new features are constructed according to the correlation analysis and hypothesis testing. It is worth noting that there are 5, 106, 7, and 106 degradation features newly constructed for the four sub-datasets.

The newly constructed degradation features after normalization are combined with the raw degradation features. Mixed with the normalized raw degradation features, the dimensions of degradation features for different sub-datasets are 22, 130, 25 and 130 (including 17, 24, 18 and 24 raw degradation features and 5, 106, 7 and 106 constructed degradation features). Trendability coefficient of each feature is calculated. The given $w_{\text{given}}$ is set to 0.95. If $w_{\text{given}}$ is set to a greater value, more features are selected and fed into a LSTM network. Otherwise, less features are selected. The selected features have more probability to be described by a same underlying function and more correlation coefficients among the features. According to $w_{\text{given}}$, there are 19, 124, 22, and 124 degradation features selected with respect to Datasets FD001, FD002, FD003 and FD004, respectively. Then the selected features are imported into an LSTM network. The whole process about the dimension change of degradation features is summarized in Table 2.

The output of an LSTM network is the RULs of engines. Its label value has a significant impact on prediction performance. It has proved that piecewise linear labeling (with 125 cycles) of the C-MAPSS engine dataset is effective and beneficial [3, 12]. The intentional-added noise obeys Gaussian distribution $\mathbf{n}$~N(0,0.1). Since too large amplitude of the noise may cause fluctuation. In addition, noise dominates the parameter optimization rather than the real output samples during the training process.

The minimum batch size is set to 20. The initial learning rate is set to 0.01, and the learning rate is reduced by a factor of 0.3 every 30 iterations during training a LSTM network. The maximum epoch is set to 100. Even though a one-layer LSTM network is adopted, dropout is still used to reduce the training process of LSTM network. The number of units in the fully-connected layer is set to 30. The dropout rate is set to 0.5 according to Ref. [36]. The other important element related to the LSTM network construction is the number of hidden units. It is usually determined according to empirical experience, but its value significantly impacts the RUL prediction accuracy. To choose a reasonable value, ten-fold cross validation is conducted. The results from cross validation are shown in Fig. 4. From Fig. 4, the proper numbers of hidden units respectively are 80, 50, 80 and 50 for FD001, FD002, FD003 and FD004.

The RUL prediction results from the LSTM networks with intentionally-added noise are listed in Table 3. To avoid the impacts caused by the random parameters in an LSTM network, the prediction is run ten times. The best indicators from the ten predictions for each sub-dataset are marked in green as well.

To compare the real and predicted RUL more clearly, Fig. 5 is given to show the real RULs (blue lines), predicted RULs (red lines) and the 0.1 interval for real RULs (green lines). It is worth noting that the results in Fig. 5 are from one prediction and the engines are randomly selected. From Fig. 5, the predicted RULs show good tendency which is well

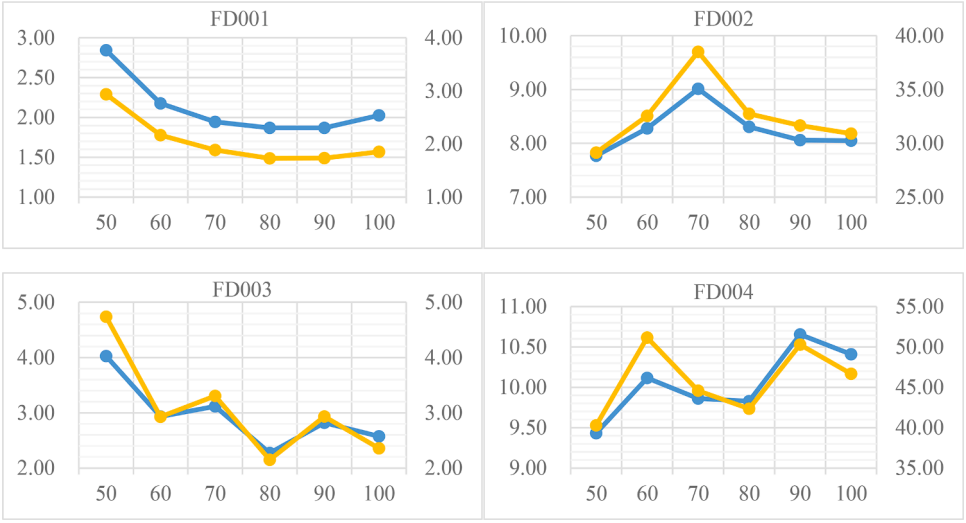



**Fig. 4.** Average $E_{\text{RMSE}}$ (blue line) and $E_{\text{score}}$ (yellow line) for different numbers of hidden units

**Table 3**
The RUL prediction results from LSTM networks with noise injection

| Dataset | FD001 | | FD002 | | FD003 | | FD004 | |
|---|---|---|---|---|---|---|---|---|
| Indicators | $E_{score}$ | $E_{RMSE}$ | $E_{score}$ | $E_{RMSE}$ | $E_{score}$ | $E_{RMSE}$ | $E_{score}$ | $E_{RMSE}$ |
| 1 | 202.12 | 12.44 | 1880.39 | 17.99 | 236.41 | 10.87 | 2743.39 | 20.48 |
| 2 | 245.19 | 12.61 | 1494.28 | 17.11 | 276.36 | 11.54 | 3618.98 | 20.80 |
| 3 | 218.38 | 11.97 | 1807.76 | 17.21 | 413.66 | 13.71 | 2471.78 | 19.26 |
| 4 | 215.30 | 12.73 | 1584.66 | 18.29 | 259.10 | 11.25 | 2636.95 | 19.73 |
| 5 | 187.95 | 11.55 | 1271.47 | 16.02 | 304.30 | 11.88 | 3889.27 | 21.54 |
| 6 | 244.81 | 12.72 | 1666.11 | 18.01 | 362.45 | 12.03 | 2112.92 | 19.38 |
| 7 | 261.14 | 13.24 | 1456.95 | 18.25 | 369.42 | 11.63 | 3288.09 | 21.44 |
| 8 | 198.57 | 12.07 | 1748.07 | 18.11 | 486.78 | 11.24 | 3742.32 | 21.34 |
| 9 | 252.69 | 13.28 | 1921.73 | 18.52 | 457.82 | 11.38 | 2336.41 | 19.40 |
| 10 | 202.89 | 12.20 | 1627.63 | 17.78 | 459.57 | 12.85 | 3548.29 | 20.91 |
| **Avg.** | **225.13** | **12.51** | **1647.93** | **17.72** | **362.59** | **11.84** | **3038.84** | **20.43** |

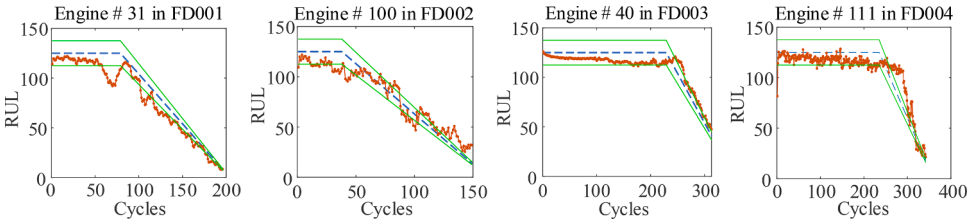



**Fig. 5.** The comparison between the real and predicted RULs of engines from one prediction

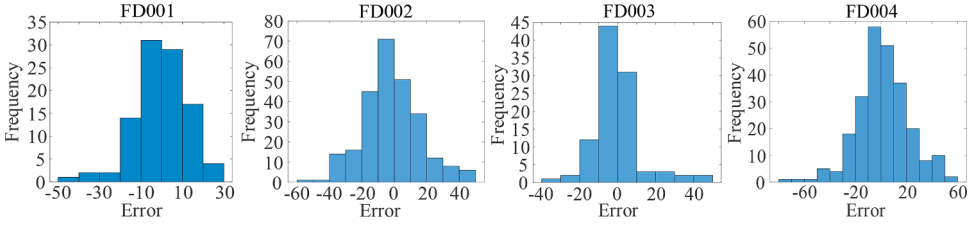



**Fig. 6.** The histograms of prediction errors from one prediction

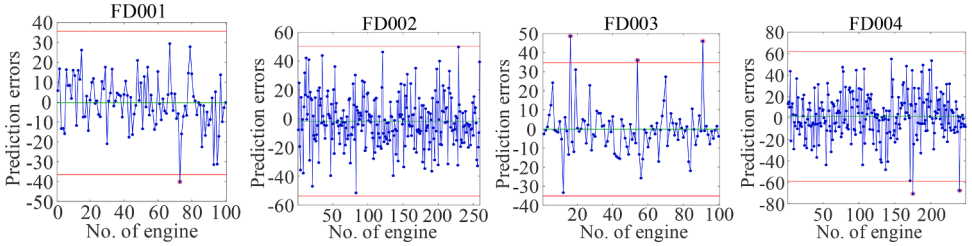



**Fig. 7.** The control charts of RUL prediction errors of engines in different sub-datasets

**Table 4**
The times of noise injection for the sub-datasets.

| Dataset | FD001 | FD002 | FD003 | FD004 |
|---|---|---|---|---|
| Iterations | 500 | 1300 | 500 | 1200 |
| Times of noise injection | 257 | 671 | 245 | 600 |
| Percentage of noise injection | 51.40% | 51.62% | 49.00% | 50.00% |

matched with the real RULs. Histograms, which are also obtained from once prediction, are given in Fig. 6 where most of the prediction errors approximate 0.

A Shewhart control chart which is Fig. 7 is given as the overview of prediction errors for all the engines. In Fig. 7, the green line represents the average value of the RUL prediction errors, and the red lines represents the associated upper and lower control limits which can be obtained according to

$$\begin{aligned} v_{\text{ucl}} &= \mu_{\text{error}} + \alpha\sigma_{\text{error}} \\ v_{\text{lcl}} &= \mu_{error} - \alpha\sigma_{\text{error}} \end{aligned} \tag{26}$$

where $v_{\text{ucl}}$ and $v_{\text{lcl}}$ denote the values of upper and lower control limits, respectively. $\mu_{\text{error}}$ and $\sigma_{\text{error}}$ denote the mean value and standard deviation of RUL prediction error, respectively. $\alpha$ is a parameter to determine the "distance" from the center line to the boundaries. Here, $\alpha$ is set to 3.

From Fig. 7, only 6 prediction errors among the 707 prediction errors are violations which are beyond the upper or lower control limits. Thus, 99.15% (e.g. (707-6)/707=99.15%) prediction errors are in control among all the predicted RULs of engines.

During training a network, the noise is intentionally injected into the output layer of the LSTM network according to the NEM theorem and Algorithm 2. In consideration of the minibatch size and the maximum epoch, there are 5, 13, 5 and 12 batches divided for datasets FD001, FD002, FD003 and FD004, respectively. Correspondingly, there are 500,

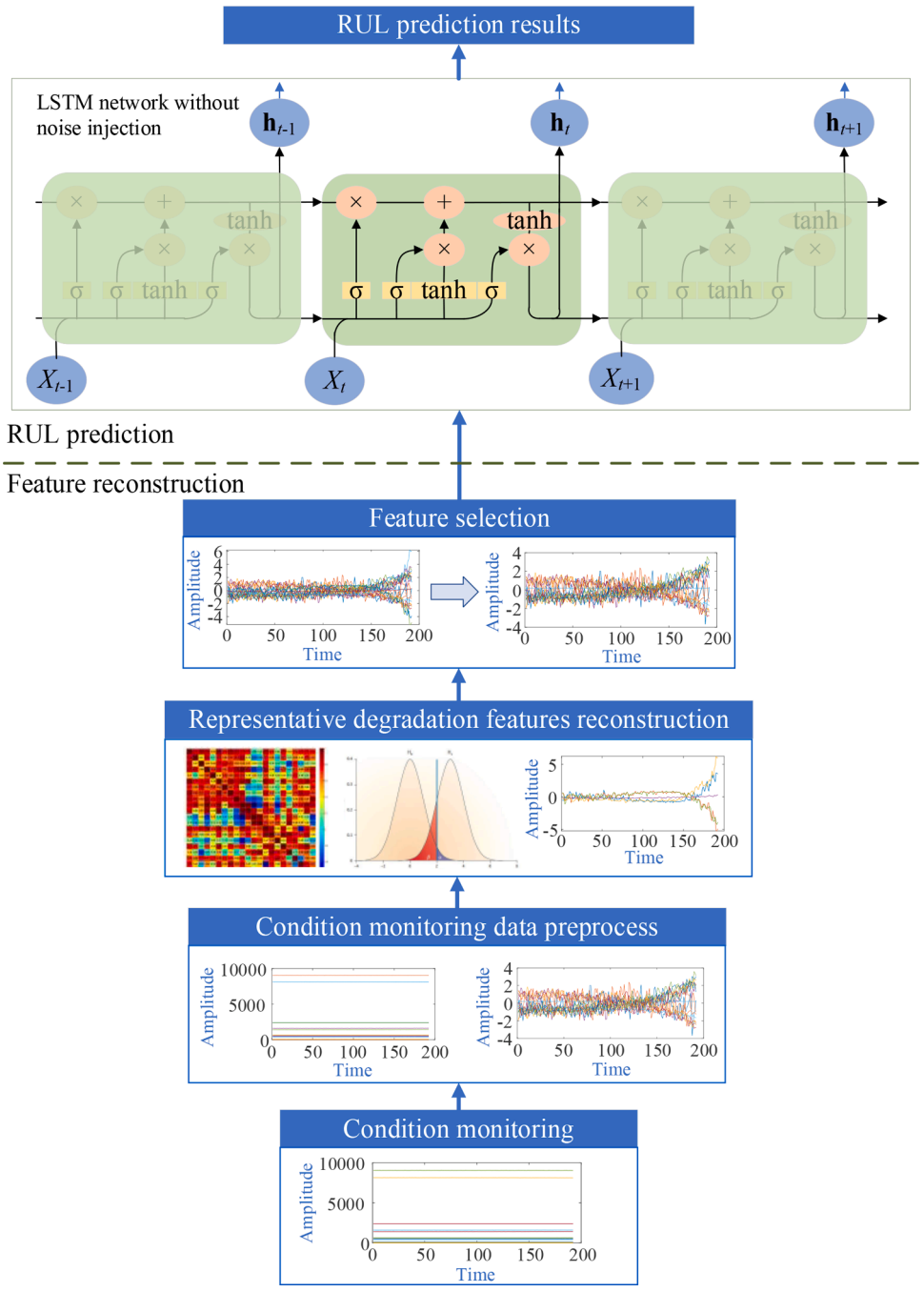



**Fig. 8.** The procedure of the RUL prediction method without noise injection

1300, 500 and 1200 iterations for training the four sub-datasets. The times of noise injection during training the LSTM networks for the four sub-datasets are listed in Table 4. For example, noise is injected 257 times during the training process for dataset FD001. From Table 4, noise is injected into the four datasets for different times. This is caused by the following two reasons: (1) the four datasets are trained separately and (2) the noise is randomly generated during each training process, therefore the times of iteration related to noise injection is random. Even though there is a variation about the times of noise injection for the four datasets, the injected noise works almost half of the whole training process.

## 5. Comparison and discussion

### 5.1. Validation of noise injection and feature reconstruction

There are two main contributions in this paper: one contribution is that noise is intentionally injected into the neural network when predicting the RUL of a machine. The other contribution is that new degradation features are constructed and selected according to correlation analysis, hypothesis testing and trendability. To validate the effectiveness of noise injection, a comparative experiment is conducted where noise is not injected during training the LSTM networks. That is to say; Algorithm 2 is not executed. But the rest procedures for RUL prediction are unchanged. The procedure of the comparative experiment is illustrated in Fig. 8. Note that the processes of feature reconstruction are

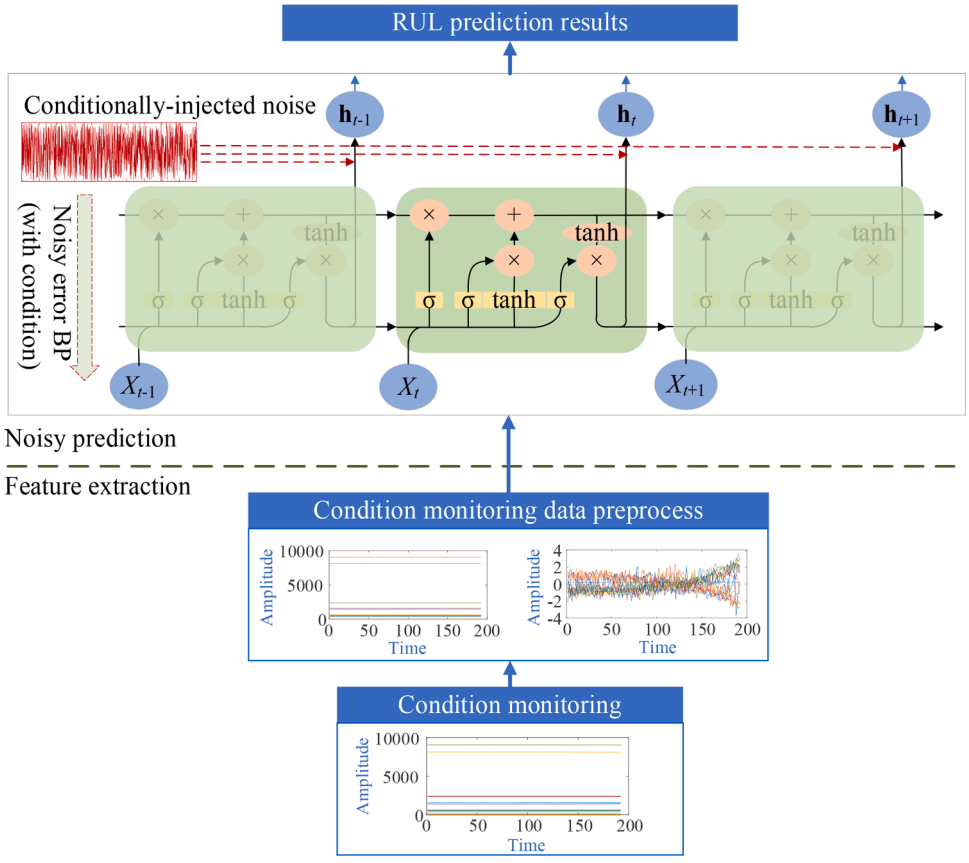



**Fig. 9.** The procedure of RUL prediction without constructed features

**Table 5**
RUL prediction results from different comparative experiments

| Methods | | Proposed method | | | Case 1 | | | Case 2 | | | Case 3 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Indicators | | Best results | Mean | STD | Mean | STD | $I_{MV}$ | Mean | STD | $I_{MV}$ | Mean | STD | $I_{MV}$ |
| FD001 | $E_{score}$ | 187.95 | 225.13 | 25.93 | 278.03 | 61.79 | 47.93% | 308.87 | 64.85 | 64.34% | 318.01 | 104.24 | 69.20% |
| | $E_{RMSE}$ | 11.55 | 12.51 | 0.55 | 13.86 | 1.36 | 20.00% | 14.89 | 1.318 | 28.92% | 15.04 | 1.48 | 30.22% |
| FD002 | $E_{score}$ | 1271.47 | 1647.93 | 203.06 | 1702.22 | 407.51 | 33.88% | 24058.53 | 42210.58 | 1792.18% | 2819.36 | 606.32 | 121.74% |
| | $E_{RMSE}$ | 16.02 | 17.72 | 0.75 | 17.74 | 0.99 | 10.74% | 18.57 | 2.46 | 15.92% | 18.66 | 1.01 | 16.48% |
| FD003 | $E_{score}$ | 236.41 | 362.59 | 90.69 | 542.56 | 171.77 | 129.50% | 733.90 | 120.48 | 210.44% | 726.22 | 230.38 | 207.19% |
| | $E_{RMSE}$ | 10.87 | 11.84 | 0.85 | 14.00 | 1.46 | 28.79% | 14.86 | 0.86 | 36.71% | 14.85 | 2.00 | 36.61% |
| FD004 | $E_{score}$ | 2112.92 | 3038.84 | 649.57 | 4681.55 | 1705.03 | 121.57% | 11073.40 | 8262.65 | 424.08% | 5821.84 | 3660.63 | 175.54% |
| | $E_{RMSE}$ | 19.38 | 20.43 | 0.911 | 20.73 | 0.87 | 6.97% | 22.26 | 1.96 | 14.86% | 20.93 | 1.79 | 8.00% |
| **Overall** | $\boldsymbol{E_{score}}$ | **3808.75** | **5274.49** | - | **7204.36** | - | **89.15%** | **36174.70** | - | **849.78%** | **9685.43** | - | **154.29%** |
| | $\boldsymbol{E_{RMSE}}$ | **16.17** | **17.42** | - | **17.91** | - | **10.76%** | **19.05** | - | **17.81%** | **18.56** | - | **14.78%** |

**Table 6**
The dimension change of degradation features in different sub-datasets

| Dataset | Original features | | | | Features from the proposed method | | | |
|---|---|---|---|---|---|---|---|---|
| | $C_{monotonicity}$ | $C_{prognosability}$ | $C_{trendability}$ | $C_{\sum}$ | $C_{monotonicity}$ | $C_{prognosability}$ | $C_{trendability}$ | $C_{\sum}$ |
| FD001 | 0.13 | 0.75 | 0.75 | 1.63 | 0.12 | 0.72 | 0.85 | 1.67 |
| FD002 | 0.15 | 0.52 | 0.97 | 1.64 | 0.07 | 0.57 | 1.00 | 1.63 |
| FD003 | 0.18 | 0.53 | 0.80 | 1.51 | 0.15 | 0.55 | 0.87 | 1.57 |
| FD004 | 0.15 | 0.51 | 0.99 | 1.64 | 0.07 | 0.54 | 1.00 | 1.60 |

unchanged.

In the second comparative experiment, the condition monitoring data is preprocessed then imported into the noisy LSTM network. The correlation analysis and hypothesis testing are not conducted. In other words, Partial Algorithm 1 is not performed. The procedure of RUL prediction in this comparative experiment is illustrated in Fig. 9. As shown in Fig. 9, the noise injection remains.

In the feature reconstruction process, the selected degradation features are operated by element-wise multiplication. The degradation features can be regarded as vectors. The operators among vectors basically includes plus, minus, multiplication, and division. The operators of plus and minus can be regarded as the same. The operators of multiplication and division can be regarded as the same except the zero values in the vector which is regarded as denominator. Multiplication can be

**Table 7**
The RUL prediction errors from different methods

| Methods | Year | FD001 | | FD002 | | FD003 | | FD004 | |
|---|---|---|---|---|---|---|---|---|---|
| | | $E_{score}$ | $E_{RMSE}$ | $E_{score}$ | $E_{RMSE}$ | $E_{score}$ | $E_{RMSE}$ | $E_{score}$ | $E_{RMSE}$ |
| Proposed method (overall) | 2021 | 225.13 | 12.51 | 1647.93 | 17.72 | 362.59 | 11.84 | 3038.84 | 20.43 |
| (best) | 2021 | 187.95 | 11.55 | 1271.47 | 16.02 | 236.41 | 10.87 | 2112.92 | 19.38 |
| DCNN [3] | 2018 | 273.7 | 12.61 | 10412 | 22.36 | 284.1 | 12.64 | 12466 | 23.31 |
| ANN-evolutionary [37] | 2019 | 337 | 14.39 | 12599 | 29.09 | 533 | 15.42 | 18526 | 34.74 |
| RNN [38] | 2019 | 262 | 14.72 | 6953 | 29 | 452 | 17.72 | 15069 | 33.43 |
| semi-supervised DL [39] | 2019 | 231 | 12.56 | 3366 | 22.73 | 251 | 12.10 | 2840 | 22.66 |
| DSCN [40] | 2019 | 260.67 | 10.95 | 4367.56 | 20.47 | 246.55 | 10.62 | 5168.45 | 22.64 |
| multi-scale DCNN [41] | 2020 | 196.22 | 11.44 | 3747 | 19.35 | 241.89 | 11.67 | 4844 | 22.22 |
| LSTM [42] | 2020 | 398.7 | 15.8 | 3493.2 | 21.4 | 584.2 | 16.0 | 3203.4 | 22.4 |
| RNN [43] | 2020 | NA | 14.57 | NA | 23.20 | NA | 14.92 | NA | 28.72 |
| DLSTM [12] | 2020 | 655 | 18.33 | NA | NA | NA | NA | NA | NA |
| CBLSTM [4] | 2020 | 304.29 | 12.66 | NA | NA | NA | NA | NA | NA |

roughly divided into scalar dot product, cross product and element-wise multiplication. The result of dot product of two vectors can be a value which is not related to time. The cross product requires the vectors with 3 dimensions; thus, it is not suitable for the feature reconstruction in a RUL prediction problem. The rank of product of two vectors maybe a way to obtain new degradation features, however, it needs huge calculation consumption if the size of vector is large. Therefore, element-wise multiplication is an easier way to reconstruct degradation feature. In the third comparative experiment, the plus operator is considered to construct new features, namely, the "*" in Eq. (9) is replaced by "+". In addition, noise injection remains in this comparative experiment.

### 5.2. Discussion about the comparative experiments

The three comparative experiments are run ten times to avoid the randomness of parameters in the LSTM network. Here, all the settings remain the same with the settings in Section 4. These comparative experiments are named as: Case 1: prediction without noise injection. Case 2: prediction without feature construction and selection. Case 3: feature construction based on plus operator.

The results from the three comparative experiments are listed in Table 5 where $I_{MV}$ denotes the improvements of mean values. STD denotes the standard deviations. From Table 5, the reduction of mean values and standard deviations by comparing the proposed method with the three cases can illustrate the effectiveness of noise injection, effectiveness of feature reconstruction and selection, and effectiveness of element-wise multiplication when constructing new degradation features, respectively. Since mean value and standard deviation are the widely-adopted statistical characterizes for comparison. $I_{MV}$ in Table 5 are calculated as Eq. (27). Specifically, the $I_{MV}$ of $E_{score}$ and $E_{RMSE}$ can be calculated according to Eqs. (28)-29).

$$\frac{E_{comparative\_experiment} - E_{proposed\_method}}{E_{proposed\_method}} \times 100\% \tag{27}$$

$$\frac{\sum_{m=1}^{4} E^{m}_{score\_comparative} - \sum_{m=1}^{4} E^{m}_{score\_proposed}}{\sum_{m=1}^{4} E^{m}_{score\_proposed}} \times 100\% \tag{28}$$

$$\frac{\sqrt{\frac{1}{\sum_{m=1}^{4} N^{m}} \times \sum_{m=1}^{4} N^{m} \times \left(E^{m}_{RMSE\_comparative}\right)^{2}} - \sqrt{\frac{1}{\sum_{m=1}^{4} N^{m}} \times \sum_{m=1}^{4} N^{m} \times \left(E^{m}_{RMSE\_proposed}\right)^{2}}}{\sqrt{\frac{1}{\sum_{m=1}^{4} N^{m}} \times \sum_{m=1}^{4} N^{m} \times \left(E^{m}_{RMSE\_proposed}\right)^{2}}} \times 100\% \tag{29}$$

**Table 8**
The comparison of overall $E_{score}$ and $E_{RMSE}$

| Methods | Year | Overall $E_{score}$ | $I_{MV}$ $E_{score}$ | Overall $E_{RMSE}$ | $I_{MV}$ $E_{RMSE}$ |
|---|---|---|---|---|---|
| The proposed method (best) | 2021 | 3808.75 | NA | 16.17 | NA |
| DCNN [3] | 2018 | 23435.80 | 515.31% | 20.47 | 26.60% |
| ANN-evolutionary [37] | 2019 | 31995.00 | 740.04% | 28.22 | 74.55% |
| RNN [38] | 2019 | 22736.00 | 496.94% | 27.84 | 72.22% |
| semi-supervised DL [39] | 2019 | 6688.00 | 75.60% | 20.31 | 25.62% |
| DSCN [40] | 2019 | 10043.23 | 163.69% | 19.14 | 18.37% |
| multi-scale DCNN [41] | 2020 | 9029.11 | 137.06% | 18.66 | 15.42% |
| LSTM [42] | 2020 | 7679.50 | 101.63% | 20.38 | 26.06% |

In Eqs. (27)-(29), $E_{comparative\_experiment}$ and $E_{proposed\_method}$ denote the prediction error indicators from the comparative experiment and our proposed method, respectively. $E^{m}{}_{score\_comparative}$ and $E^{m}{}_{score\_proposed}$ denote the $E_{score}$ from the comparative experiments and our proposed method for the $m^{th}$ sub-dataset, analogously. $N^{m}$ represents the number of train trajectories for the $m^{th}$ sub-dataset.

According to Ref. [34], an ideal prognostic parameter has three key characteristics: monotonicity, prognosability and trendability, respectively. Monotonicity (formulated by Eq. (30)) characterizes the underlying positive or negative trend of the parameter. Prognosability (formulated by Eq. (31)) gives a measure of the variance in the critical failure value of a population of systems [34]. Trendability (redefined in this paper as Eq. (32)) indicates the degree to which the parameters of a population of systems have the same underlying shape. Each characteristic should be in the interval of [0,1].

$$C_{monotonicity} = \text{mean}\left(\left|\frac{\#(d/dx)^{+}}{n-1} - \frac{\#(d/dx)^{-}}{n-1}\right|\right) \tag{30}$$

$$C_{prognosability} = \exp\left(-\frac{\sigma_{fails}}{\text{mean}\left(\left|v_{fails} - v_{starts}\right|\right)}\right) \tag{31}$$

$$C_{trendability} = \max(|\rho|) \tag{32}$$

In Eqs. (30)-(32), $C_{monotonicity}$, $C_{prognosability}$ and $C_{trendability}$ denote the monotonicity, prognosability amd trendability characteristics,

**Algorithm 1**
The process of new degradation feature construction and selection

**While:** epoch $l_1$: 1→($L$-1) do

**While:** epoch $l_2$: $l_1$+1→$L$ do

- Compute correlation coefficient $\rho\left(x_{l_1}^*, x_{l_2}^*\right)$ and compared with $\rho_{\text{given}}$

If $\rho\left(x_{l_1}^*, x_{l_2}^*\right) \geq \rho_{given}$

- Conduct the hypothesis testing

If $p\left(x_{l_1}^*, x_{l_2}^*\right) \leq p_{given}$

- Construct a new degradation feature $x_{\text{new}} = x_{l_1}^* * x_{l_2}^*$
- Normalize degradation feature $x_{\text{new}}^* = \frac{x_{\text{new}} - \mu_{\text{new}}}{\sigma_{\text{new}}}$

Else

- Do nothing

End

Else

- Do nothing

End

- Calculate the correlation coefficient of each feature $C_{\text{Trendability}}^m = \max\left(\left|\rho\left(x_m^*, x_{m^*}^*\right)\right|\right)$
- Sort the degradation features according to $C_{\text{Trendability}}^m$ in descending order
- Calculate the contribution of each degradation feature to correlation representation

$$w_m = \frac{C_{\text{Trendability}}^m}{\sum_{m=1}^{M} C_{\text{Trendability}}^m}$$

- Select the degradation features $\arg \sum_{m=1}^{m^*} w_m \geq w_{\text{given}}$

**End**

**End**

respectively. $\#(d/dx)^+$ and $\#(d/dx)^-$ mean the number of positive and negative values when calculating the differential of the degradation features. $n$ means the number of data points of a degradation feature vector. $\sigma_{\text{fails}}$ stands for the standard deviation of failures values. $v_{\text{fails}}$ and $v_{\text{starts}}$ are the failure values and start values, respectively. $\rho$ means the correlation coefficient.

The three characteristics of the features used in the proposed method are calculated and listed in Table 6. The greater values represent that the features have better monotonicity, prognosability and trendability. The overall suitability of the dataset is improved from 6.42 to 6.47. Here, the original features are the only features used in Case 2. The improvement of overall suitability is about 0.78%. Even though the improvement seems small, the prediction accuracy is improved a lot by comparing the results from the proposed method and Case 2 according to Table 5.

### 5.3. Comparison with the-state-of-the-art methods

In the past decades, many outstanding RUL prediction methods have been developed and some of them are also validated by the C-MAPSS aero-engine lifetime dataset. The validation results from different RUL prediction methods are listed in Table 7. It is worth noting that the results marked in the green stand for the best results from the ten runs of the proposed method. They are the same as the values listed in Table 3. In addition, the presented/published year of these methods are listed. In Table 7, "NA" means that the corresponding values are not provided in the references.

From Table 7, our proposed method shows smallest prediction errors if comparing the results in green with the results from other methods. Even though the indicator $E_{\text{score}}$ is not calculated in Refs. [12] and [4],

**Algorithm 2**
The training process of noisy LSTM network for RUL prediction.

**Result:** Trained weight set $\boldsymbol{\Theta}$.

Initialize the network weight set $\boldsymbol{\Theta}$

**While:** epoch $i$: $1 \rightarrow I$ do

**While:** iteration $j$: $1 \rightarrow J$ do

Select a batch of $G$ samples $\left\{\mathbf{x}^{(g)}, \mathbf{y}^{(g)}\right\}_{g=1}^{G}$

**Forward pass calculation**

- Compute the activation of the three gates in the given batch.
- Compute the memory cell activation $\mathbf{s}_d^{(t)}$ and the output RUL set $\mathbf{a}_d^{\mathbf{y}(t)}$.

**Add noise**

- Generate noise vector $\mathbf{n}$:

if $\mathbf{n}^T\left(2\mathbf{a}_d^{\mathbf{y}} - 2\mathbf{y}_d - \mathbf{n}\right) \le 0$

- Add the noise: $\mathbf{y}_d \leftarrow \mathbf{y}_d + \mathbf{n}$

else

- Do nothing

End

**Backpropagation calculation**

Initialize: $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) = 0$

- Compute the training error $E(\boldsymbol{\theta})$
- Compute the error gradient $\nabla_{\boldsymbol{\theta}}^{(t)} E(\boldsymbol{\theta})$
- Update the parameter set $\boldsymbol{\Theta}$ using gradient information

**End**

**End**

the prediction errors from our proposed method are less than the values from the other methods where only the sub-dataset FD001 is used for validation. Even though all the sub-datasets are used in the rest comparative reference, the proposed method can obtain the smallest $E_{\text{score}}$ and $E_{\text{RMSE}}$ (except Refs. [41-42] only with respect to $E_{\text{RMSE}}$ in FD001 or/and FD003). To make the different methods comparable, the overall $E_{\text{score}}$ and $E_{\text{RMSE}}$ are calculated and compared in Table 8. From Table 8, the overall $E_{\text{score}}$ and $E_{\text{RMSE}}$ from our proposed method are smallest. The improvements of overall $E_{\text{score}}$ and $E_{\text{RMSE}}$ are greater than 75% and 15%. Merged the discussion about Tables 7 and 8, the out-performance of our proposed method compared with the-state-of-the art methods is demonstrated.

## 6. Conclusions

In this paper, a new RUL prediction method is proposed. There are two main contributions to the proposed method. Firstly, a series of new degradation features are reconstructed based on the correlation analysis and hypothesis testing, which are conducted among the raw degradation features. However, not all the constructed features are used for training an LSTM network in this paper. A feature is selected according to its trendability representation ability. In the feature construction phase, element-wise multiplication is considered rather than other operators due to (1) the dimensions of output from other operators and (2) the accuracy of the prediction results. Element-wise multiplication can make two degradation feature vectors more relative. In the feature

selection phase, trendability is considered rather than linear/exponential fitting of degradation features. This is due to two aspects: (1) the linear/exponential fitting takes much more time on fitting especially when the scale of dataset is large, and (2) the accuracy of RUL oriented to linear fitting and exponential fitting is not as good as the accuracy oriented on the proposed method. The other contribution is that the LSTM network is intentionally injected by artificially-generated noise during the training process, which is different from most of the existing deep learning-based RUL prediction methods. The artificially-generated noise is intentionally injected into the output layer of a LSTM network; therefore, the target output is "polluted" and "whiten" by the noise. The noise injection does not decrease the prediction performance, in fact improve the RUL estimation. A series of comparative experiments are conducted to validate the effectiveness of our proposed method with respect to the noise injection, degradation feature reconstruction and selection method, and the-state-of-the-art methods. From comparisons, our proposed new RUL prediction method outperforms.

This study is a preliminary exploration of noisy-EM positivity condition on RUL prediction. Even though it shows improved performances, further research should be conducted. For example, exploring more effective way to reduce the result oscillation, selecting and reconstructing more suitable features for prognostics, or exploring better rules for noise injection into the different layers of a neural network.

## Author Statement

Lei Xiao has made substantial contributions to the conceptions, design, acquisition and analysis for the work;

Junxuan Tang has carried on all the experiments in this manuscript;

Xinghui Zhang has proposed some conceptions in this paper;

Eric Bechhoefer has done the professional check on this paper;

Siyi Ding has proposed some suggestions about the revision of draft.

All persons who have made substantial contributions to the work reported in this manuscript.

## Declaration of Competing Interest

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work, there is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled, "Remaining useful life prediction based on intentional noise injection and feature reconstruction"

## Acknowledgments

The authors gratefully acknowledge the support by the National Natural Science Foundation of China (52075094 and 51705321), the Fundamental Research Funds for the Central Universities (2232019D3-29), the support of the China Postdoctoral Science Foundation (2017M611576), and the Initial Research Funds for Young Teachers of Donghua University.

## References

[1] Khelif R, Chebel-Morello B, Malinowski S, Laajili E, Fnaiech F, Zerhouni N. Direct remaining useful life estimation based on support vector regression. IEEE T IND ELECTRON 2017;64:2276–85.

[2] Xia T, Dong Y, Xiao L, Du S, Pan E, Xi L. Recent advances in prognostics and health management for advanced manufacturing paradigms. RELIAB ENG SYST SAFE 2018;178:255–68.

[3] Li X, Ding Q, Sun J. Remaining useful life estimation in prognostics using deep convolution neural networks. RELIAB ENG SYST SAFE 2018;172:1–11.

[4] Xia T, Song Y, Zheng Y, Pan E, Xi L. An ensemble framework based on convolutional bi-directional LSTM with multiple time windows for remaining useful life estimation. COMPUT IND 2020;115:103182.

[5] Miao H, Li B, Sun C, Liu J. Joint learning of degradation assessment and rul prediction for aeroengines via dual-task deep lstm networks. IEEE T IND INFORM 2019;15:5023–32.

[6] Wu Y, Yuan M, Dong S, Lin L, Liu Y. Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. NEUROCOMPUTING 2018;275:167–79.

[7] Zhang C, Lim P, Qin AK, Tan KC. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. IEEE T NEUR NET LEAR, 28; 2017. p. 2306–18.

[8] Deutsch J, He D. Using deep learning-based approach to predict remaining useful life of rotating components. IEEE Transactions on Systems, Man, and Cybernetics: Systems 2018;48:11–20.

[9] Yang B, Liu R, Zio E. Remaining useful life prediction based on a double-convolutional neural network architecture. IEEE T IND ELECTRON 2019;66:9521–30.

[10] Shi Z, Chehade A. A dual-LSTM framework combining change point detection and remaining useful life prediction. RELIAB ENG SYST SAFE 2021;205:107257.

[11] Da Costa PRDO, Akçay A, Zhang Y, Kaymak U. Remaining useful lifetime prediction via deep domain adaptation. RELIAB ENG SYST SAFE 2020;195:106682.

[12] Wu J, Hu K, Cheng Y, Zhu H, Shao X, Wang Y. Data-driven remaining useful life prediction via multiple sensor signals and deep long short-term memory neural network. ISA T 2020;97:241–50.

[13] Kim M, Liu K. A Bayesian deep learning framework for interval estimation of remaining useful life in complex systems by incorporating general degradation characteristics. IISE TRANS 2020;53:326–40.

[14] Xiao L, Duan F, Tang J, Abbott D. A noise-boosted remaining useful life prediction method for rotating machines under different conditions. IEEE T INSTRUM MEAS 2021;70:1–12.

[15] Hochreiter S, Schmidhuber J. Long short-term memory. NEURAL COMPUT 1997;9:1735–80.

[16] Audhkhasi K, Osoba O, Kosko B. Noise-enhanced convolutional neural networks. NEURAL NETWORKS 2016;78:15–23.

[17] Osoba O, Mitaim S, Kosko B. The noisy expectation–maximization algorithm. FLUCT NOISE LETT 2013;12:1350012.

[18] Adigun O, Kosko B. Noise-boosted bidirectional backpropagation and adversarial learning. NEURAL NETWORKS 2019;120:9–31.

[19] Osoba O, Kosko B. Noise-enhanced clustering and competitive learning algorithms. NEURAL NETWORKS 2013;37:132–40.

[20] Adigun O, Kosko B. Using noise to speed up video classification with recurrent backpropagation. In: 2017 International Joint Conference on Neural Networks (IJCNN) 2017 International Joint Conference on Neural Networks (IJCNN). IEEE; 2017. p. 108–15.

[21] Weninger F, Erdogan H, Watanabe S, Vincent E, Roux JLe, Hershey JR, Schuller B. In: Speech enhancement with LSTM recurrent neural networks and its application to noise-robust ASR, International Conference on Latent Variable Analysis and Signal Separation; 2015.

[22] Adigun O, Kosko B. Using noise to speed up video classification with recurrent backpropagation. IEEE; 2017. p. 108–15.

[23] Xiao L, Tang J, Zhang X, Xia T. Weak fault detection in rotating machineries by using vibrational resonance and coupled varying-stable nonlinear systems. J SOUND VIB 2020;478:115355.

[24] Xiao L, Zhang X, Lu S, Xia T, Xi L. A novel weak-fault detection technique for rolling element bearing based on vibrational resonance. J SOUND VIB 2019;438:490–505.

[25] Lu S, He Q, Wang J. A review of stochastic resonance in rotating machine fault detection. MECH SYST SIGNAL PR 2019;116:230–60.

[26] Qiao Z, Lei Y, Li N. Applications of stochastic resonance to machinery fault detection: A review and tutorial. MECH SYST SIGNAL PR 2019;122:502–36.

[27] Xiao L, Bajric R, Zhao J, Tang J, Zhang X. An adaptive vibrational resonance method based on cascaded varying stable-state nonlinear systems and its application in rotating machine fault detection. NONLINEAR DYNAM 2021.

[28] Sun C, Ma M, Zhao Z, Tian S, Yan R, Chen X. Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing. IEEE T IND INFORM 2019;15:2416–25.

[29] Chen Y, Peng G, Zhu Z, Li S. A novel deep learning method based on attention mechanism for bearing remaining useful life prediction. APPL SOFT COMPUT 2020;86:105919.

[30] Chen C, Xu T, Wang G, Li B. Railway turnout system RUL prediction based on feature fusion and genetic programming. MEASUREMENT 2020;151:107162.

[31] Guo L, Li N, Jia F, Lei Y, Lin J. A recurrent neural network based health indicator for remaining useful life prediction of bearings. NEUROCOMPUTING 2017;240:98–109.

[32] Chen J, Jing H, Chang Y, Liu Q. Gated recurrent unit based recurrent neural network for remaining usefullife prediction of nonlinear deterioration process. Reliability Engineering and System Safety 2019;185:372–82.

[33] Zheng S, Ristovski K, Farahat A, Gupta C. Long Short-Term Memory Network for Remaining Useful Life estimation. Dallas, TX, United states: IEEE; 2017. p. 88–95.

[34] Coble JB. Merging Data Sources to Predict Remaining Useful Life – An Automated Method to Identify Prognostic Parameters. University of Tennessee; 2010.

[35] Saxena A, Goebel K, Simon D, Eklund N. Damage propagation modeling for aircraft engine run-to-failure simulation. IEEE; 2008. p. 1–9.

[36] Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors. 2012.

[37] Laredo D, Chen Z, Schütze O, Sun J. A neural network-evolutionary computational framework for remaining useful life estimation of mechanical systems. NEURAL NETWORKS 2019;116:178–87.
[38] Zhao S, Zhang Y, Wang S, Zhou B, Cheng C. A recurrent neural network approach for remaining useful life prediction utilizing a novel trend features construction method. MEASUREMENT 2019;146:279–88.
[39] Listou Ellefsen A, Bjørlykhaug E, Æsøy V, Ushakov S, Zhang H. Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. RELIAB ENG SYST SAFE 2019;183:240–51.
[40] Wang B, Lei Y, Li N, Yan T. Deep separable convolutional network for remaining useful life prediction of machinery. MECH SYST SIGNAL PR 2019;134:106330.
[41] Li H, Zhao W, Zhang Y, Zio E. Remaining useful life prediction using multi-scale deep convolutional neural network. APPL SOFT COMPUT 2020;89:106113.
[42] Deng K, Zhang X, Cheng Y, Zheng Z, Jiang F, Liu W, Peng J. A remaining useful life prediction method with long-short term feature processing for aircraft engines. APPL SOFT COMPUT 2020;93:106344.
[43] Xia M, Zheng X, Imran M, Shoaib M. Data-driven prognosis method using hybrid deep recurrent neural network. APPL SOFT COMPUT 2020;93:106351.