Reliability Engineering and System Safety 185 (2019) 372–382

Contents lists available at ScienceDirect

Reliability Engineering and System Safety

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

# Gated recurrent unit based recurrent neural network for remaining useful life prediction of nonlinear deterioration process

Chen Jinglong*, Jing Hongjie, Chang Yuanhong, Liu Qian

*State Key Laboratory for Manufacturing and Systems Engineering, Xi' an Jiaotong University, Xi'an, China*

ARTICLE INFO

*Keywords:*
RUL prediction
PHM
Recurrent neural network
Nonlinear deterioration

ABSTRACT

Remaining useful life (RUL) prediction is a key process for prognostics and health management (PHM). However, conventional model-based methods and data-driven methods for RUL prediction are bad at a very complex system with multiple components, multiple states and therefore extremely large amount of parameters. In order to solve the problem, a general two-step solution is proposed in this paper. In the first step, kernel principle component analysis (KPCA) is applied for nonlinear feature extraction. Then, a novel recurrent neural network called gated recurrent unit (GRU) is presented as the second step to predict RUL. GRU network is capable of describing a very complex system because of its specially designed structure. The effectiveness of the proposed solution for RUL prediction of a nonlinear degradation process is proved by a case study of commercial modular aero-propulsion system simulation data (C-MAPSS-Data) from NASA. Results also show that the proposed method requires less training time and has better prediction accuracy than other data-driven methods.

## 1. Introduction

PHM is a basic requirement for condition-based maintenance in many application domains where safety, reliability, and availability of the systems are considered mission critical [1]. Particularly, RUL prediction is one of the main tasks in PHM. Improving the accuracy of the RUL prediction can not only enhance the safety and reliability, but also prolong service time which decreases the average cost in turn. Therefore, many researchers have studied RUL prediction methods in recent years.

Generally, there are two approaches for RUL prediction: model-based methods and data-driven methods. Model-based methods can be used for a component or a simple system to deduct a more accurate RUL by building a physical failure model while data-driven methods can estimate RUL for a complex system by constructing a simpler data-based model. In order to predict RUL of complex systems, data-driven methods therefore has got more attention recently [2]. Ahmad et al. [3] predicted the RUL of the rolling element bearings using dynamic regression models. Hu et al. [4] proposed a prediction method for the RUL of wind turbine bearings based on the Wiener process. Huang et al. [5] presented an adaptive skew-Wiener process model for RUL prediction. Zhang et al. [6] presented a review on Wiener-process-based methods for RUL prediction and degradation data analysis. Le et al. [7] estimated the RUL with noisy gamma deterioration process. Ling et al. [8] proposed Bayesian and likelihood inferences on remaining useful life in two-phase degradation models under gamma process. Baptista et al. [9] proposed a method for RUL prediction combining data-driven and Kalman filter. Son et al. [10] predicted the RUL based on noisy condition monitoring signals using constrained Kalman filter. Duong et al. [11] presented a method with heuristic Kalman optimized particle filter for RUL prediction. Liu et al. [12] proposed a novel method using adaptive hidden semi-Markov model (HSMM) for multi-sensor monitoring equipment health prognosis. Chen et al. [13] presented a hidden Markov model (HMM) with auto-correlated observations for RUL prediction and optimal maintenance policy. Li et al. [14] proposed an optimal Bayesian control policy for gear shaft fault detection using HSMM. Chen et al. [15] proposed a general solution to nonlinear multistate deterioration modeling with non-homogeneous hidden semi-Markov model (NHSMM) for deterioration level assessment and RUL prediction. Moghaddass et al. [16] presented an integrated framework for online diagnostic and prognostic health monitoring using a multi-state deterioration process. Although these methods are widely used, they have their own limitations. The deterioration process of the equipment is usually nonlinear and multiple, because of the complicated structure and variable work status. The deterioration curve may not follow a typical shape such as exponential or linear function. It is an important challenge of RUL prediction that finding out the rule of the nonlinear deterioration. Wiener process, Gamma process and Kalman

* Corresponding author.
*E-mail address:* jlstrive2008@mail.xjtu.edu.cn (J. Chen).

https://doi.org/10.1016/j.ress.2019.01.006
Received 3 July 2018; Received in revised form 5 December 2018; Accepted 7 January 2019
Available online 08 January 2019
0951-8320/ 

filter perform not very well when the deterioration process is nonlinear. HMM model performs well on nonlinear deterioration process but the training time increases dramatically when multiple system states are concerned.

Another important branch of data-driven methods is artificial intelligence (AI). In recent years, AI, particularly deep learning methods, has achieved outstanding performance in image processing, natural language processing (NLP) and so on. Researchers have also exploited applications of AI methods for RUL prediction. Among deep learning methods, recurrent neural network (RNN) has attracted special attention because its network structure contains recurrent hidden layer, which is very suitable for time series processing and consequently RUL prediction. Guo et al. [17] proposed a recurrent neural network based health indicator for RUL prediction of bearings. Liu et al. [18] proposed a method for fault diagnosis of rolling bearings with recurrent neural network-based auto-encoders. However, RNN cannot link two similar data if they are separated too far away.

In order to overcome the weakness of RNN, long short term memory (LSTM) is proposed, which introduces input gate, output gate and cell state into RNN [19]. LSTM could save long-time memory into cell state and it has been verified as a most mature and efficient method on many tasks. Hinchi and Tkiouat [20] proposed a method based on LSTM for RUL prediction of rolling bearing. Yuan et al. [21] proposed a method for RUL prediction of aero engine using LSTM neural network. Malhotra et al. [22] proposed a method for multi-sensor prognostics by using an unsupervised health index based on LSTM encoder-decoder. However, each memory blocks in LSTM needs an input gate and an output gate. These gates make the training more difficult and increase the training time of the network.

To reduce training time and improve network performance, a simplified but improved LSTM-architecture network, GRU, is proposed [23]. The GRU chooses a new type of hidden unit that merges the forget gate and the input gate into a single update gate and mixes cellular state and hidden state into one state as well. In brief, the number of gates is decreased from 4 in LSTM to 2 in GRU, named update gate and reset gates.

A general two-step solution for RUL prediction of nonlinear deterioration process is proposed to deal with the nonlinearity in deterioration modeling. In the solution, (1) KPCA is applied as the first step for nonlinear feature extraction. By reducing the dimension, over-fitting caused by too many model parameters can be effectively avoided. (2) GRU, a simplified network of LSTM with fewer parameters, is presented to predict RUL. In practice, (3) Sequence-to-one method is applied, increasing the number of samples while avoiding the trouble of variable length sequence input. 4) Sliding average method is applied to smooth the results, increasing the prediction accuracy effectively.

The rest of the paper is organized as follows. Section II describes different RNN structures. In section III, a general two-step solution for RUL prediction is proposed. In Section IV, the C-MAPSS-Data is used to verify the efficiency and accuracy of the proposed method. Finally, conclusion is drawn in Section V.

## 2. Recurrent neural network

In this section, we briefly introduce RNN, LSTM and GRU. A standard neural network usually contains three layers, input layer, hidden layer and output layer. The input set is marked as the vector **x**, and the hidden set is marked as the vector h, and the output set is marked as the vector **y**. Matrix **U** connects input layer and hidden layer, and matrix **V** connects hidden layer and output layer. Any two inputs are totally independent, for the points are not related inside of the layers. When it comes to the projects which are required to concern the relationships of the inputs at different times such as natural language processing and speech recognition, the common forward neural network usually doesn't work, but RNN works perfectly in these parts.

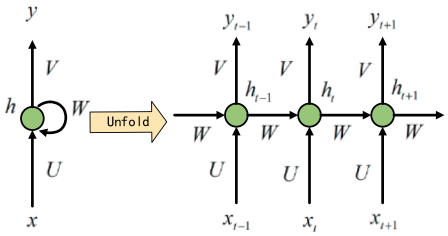



**Fig. 1.** RNN structure.

### 2.1. Standard RNN

From the Fig. 1, it can be seen that RNN is characterized as the self-connected hidden layer. Compared with a feed-forward neural network, RNN is a deep neural network which can process sequence information. RNN consists of a hidden state **h** and an optional output **y** which operates on an input sequence $\mathbf{x} = (x_0, x_1, ..., x_{N-1})^T$. At each time step t, the hidden state $h_t$ is updated as the following function:

$$h_t = f(\mathbf{W}h_{t-1} + \mathbf{U}x_t), \tag{1}$$

where $f$ is a non-linear activation function, such as sigmoid function. **W** is a matrix connected with current hidden layer and previous hidden layer. **U** is a matrix connected with current input layer and current hidden layer.

At each time step t, the output $y_t$ is updated as the following function:

$$y_t = f(\mathbf{V}h_t) \tag{2}$$

where **V** is a matrix connected with current hidden layer and current output layer. The sigmoid function, as a special non-linear activation, is widely used in artificial neural network. The mathematical form of the sigmoid function is shown as the following function:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

RNN can be trained by back-propagation. However, when it comes to long sequence input, the RNN will be hard to train. Because of the specific neural structure, the vanishing gradient problem will arise in long sequence input. RNN can only remember the latest information, but not the earlier one.

### 2.2. Long short term memory (LSTM)

LSTM model was proposed to addresses the vanishing gradient problem by Hochreiter and Schmidhuber [19] in 1997 at first. Then, Graves [24] improved and promoted the LSTM recently. Avoiding long-term dependency problems, the LSTM achieved great success in many tasks and was regarded as one of the most popular and efficient RNN model.

The structure diagram of LSTM will be shown in Fig. 2. The LSTM uses a well-designed memory cell which contains of 4 different gates to replace the activation function of hidden state in RNN. The core of LSTM is the cell state which stores the cell memory, converting the information from the start to the end. The cell state will keep the same, only to change with linear transformation by different gates. Therefore, the vanishing gradient problem can be solved.

The memory cell consists of an input gate, a forget gate and an output gate which are designed to protect and update the cell state. Each gates fetching the current input layer data $x_t$ and previous hidden layer data $h_{t-1}$ contains a sigmoid function to keep the output value from 0 to 1, which decides whether to close or to open. 1 presents that any amount is allowed to pass, and 0 presents that nothing is allowed to pass.

The forget gate is designed to decide which needs to be discarded from the cell state.

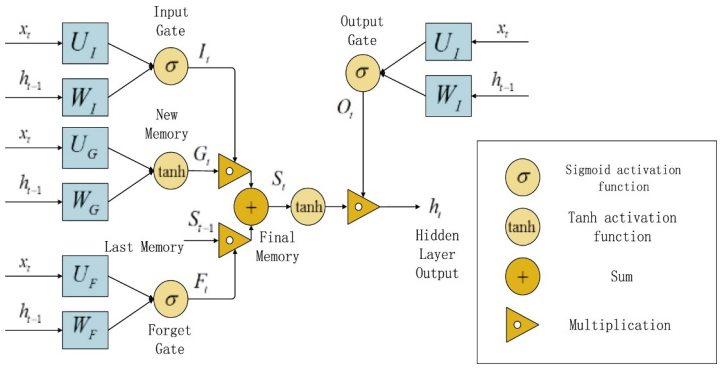



**Fig. 2.** Structure diagram of LSTM.

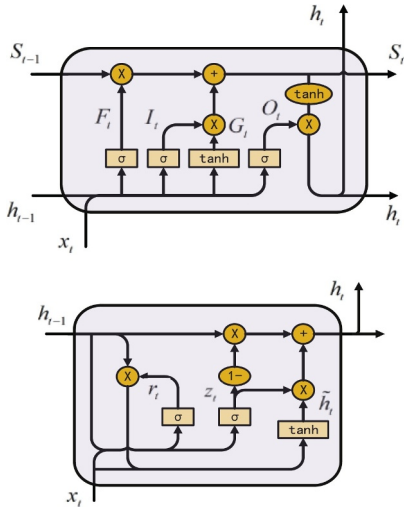


**Fig. 3.** frame of LSTM and GRU.

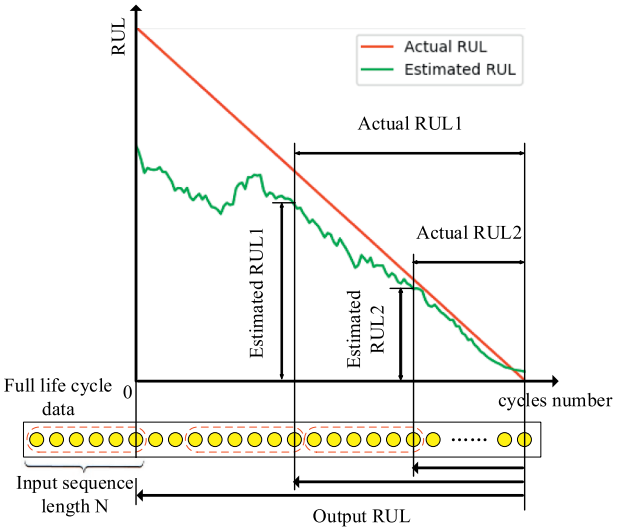



**Fig. 4.** Flow chart.

$$F_t = \sigma(\mathbf{W_F} h_{t-1} + \mathbf{U_F} x_{t-1}) \tag{4}$$

Next, the new information should be stored into the cell state conditionally. A tanh layer is chose to form the new memory and the input gate is designed to decide what should to be added into the cell state.

$$G_t = \tanh(\mathbf{W_G} h_{t-1} + \mathbf{U_G} x_{t-1}) \tag{5}$$

$$I_t = \sigma(\mathbf{W_I} h_{t-1} + \mathbf{U_I} x_{t-1}) \tag{6}$$

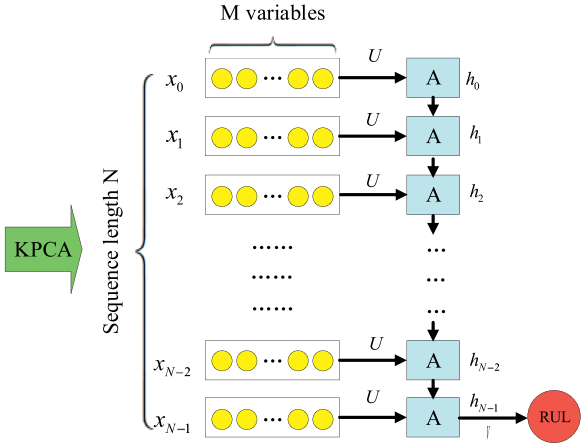



**Fig. 5.** Result style of RUL prediction.

Then, the cell state will be updated by the forget gate, the input gate and the new memory.

$$S_t = S_{t-1} * F_t + G_t * I_t \tag{7}$$

Lastly, the output gate is designed to decide which information will be converted from the cell state into the current hidden layer data.

$$O_t = \sigma(\mathbf{W_O} h_{t-1} + \mathbf{U_O} x_{t-1}) \tag{8}$$

$$h_t = O_t * \tanh(S_t) \tag{9}$$

In the formula above, $\mathbf{W_I}$,$\mathbf{W_F}$,$\mathbf{W_G}$,$\mathbf{W_O}$ are hidden layer weight values between current hidden layer and previous hidden layer, and $\mathbf{U_I}$,$\mathbf{U_F}$,$\mathbf{U_G}$,$\mathbf{U_O}$ are weight values between current input layer and current hidden layer. ⊗ is element by element multiplication operator.

The specific structure of LSTM can be shown as bellow:

### 2.3. Gated recurrent unit (GRU)

GRU model was proposed at 2014 by Cho et al. [23] on statistical machine translation. The GRU chooses a new type of hidden unit that has been motivated by the LSTM unit. It combines with the forget gate and the input gate into a single update gate. It is also mixed with cellular state and hidden state. The final model is simpler than the standard LSTM model and is a very popular variant.

We will describe how the activation of the hidden unit at time step t is computed. Firstly, $r_t$ is computed by

$$r_t = \sigma(W_r h_{t-1} + U_r x_t) \tag{10}$$

where σis the logistic sigmoid function, and $\mathbf{W_r}$, $\mathbf{U_r}$ are weight matrices.

The new remember $\tilde{h}_t$ is generated by $r_t$ with a tanh layer. The function is

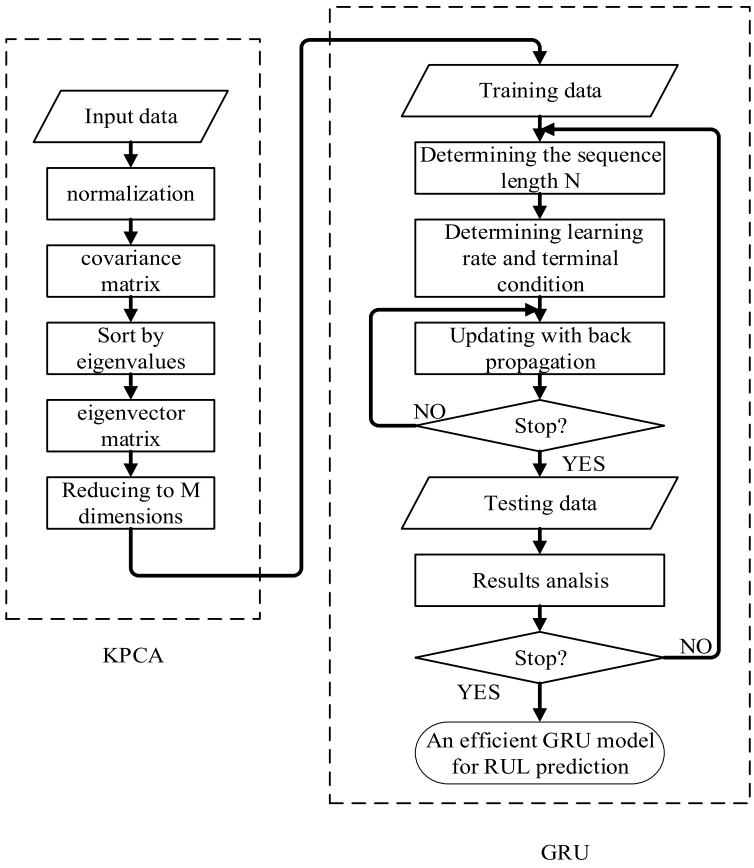



**Fig. 6.** Data processing network diagram.

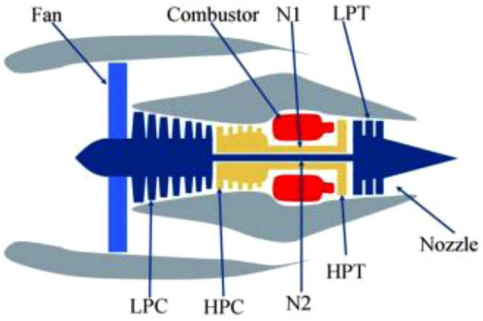



**Fig. 7.** Aircraft engine block diagram.

$$\tilde{h}_t = \tanh(W(r_t * h_{t-1}) + Ux_t) \tag{11}$$

The GRU deletes the remember gate and forget gate in LSTM, and create the $z_t$ to replace them. $z_t$ is computed by

$$z_t = \sigma(W_z h_{t-1} + U_z x_t) \tag{12}$$

In the end, the hidden state value is updated by

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t \tag{13}$$

From Fig. 3, it is easy to be found that GRU contains less variables than LSTM, which makes GRU more compact and efficient.

## 3. Proposed model

In this section, a general solution is proposed for RUL prediction of nonlinear deterioration process.

**Table 1**
The physical meanings of C-MAPSS data sources.

| Symbol | Description | Units |
|---|---|---|
| Parameters available to participants as sensor data | | |
| T2 | Total temperature at fan inlet | °R |
| T24 | Total temperature at LPC outlet | °R |
| T30 | Total temperature at HPC outlet | °R |
| T50 | Total temperature at LPT outlet | °R |
| P2 | Pressure at fan inlet | psia |
| P15 | Total pressure in bypass-duct | psia |
| P30 | Total pressure at HPC outlet | psia |
| Nf | Physical fan speed | rpm |
| Nc | Physical core speed | rpm |
| Epr | Engine pressure ratio (P50/P2) | – |
| Ps30 | Static pressure at HPC outlet | psia |
| phi | Ratio of fuel flow to Ps30 | pps/psi |
| NRf | Corrected fan speed | rpm |
| NRc | Corrected core speed | rpm |
| BPR | Bypass Ratio | – |
| farB | Burner fuel-air ratio | – |
| htBleed | Bleed Enthalpy | – |
| Nf_dmd | Demanded fan speed | rpm |
| PCNfR_dmd | Demanded corrected fan speed | rpm |
| W31 | HPT coolant bleed | lbm/s |
| W32 | LPT coolant bleed | lbm/s |
| Parameters for calculating the Health Index | | |
| T48 (EGT) | Total temperature at HPT outlet | °R |
| SmFan | Fan stall margin | – |
| SmLPC | LPC stall margin | – |
| SmHPC | HPC stall margin | – |

### 3.1. Phase1: KPCA

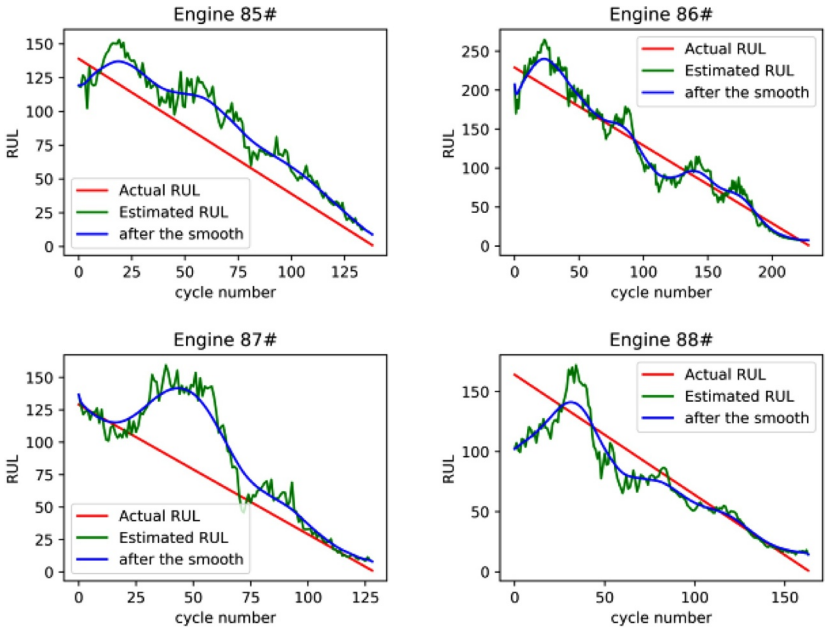
The objective of KPCA is dimensionality reduction and nonlinear feature extraction, by retaining useful information and removing

**Table 2**
RMS error of RUL prediction and training time of 81#-100# engines with GRU in different *N*.

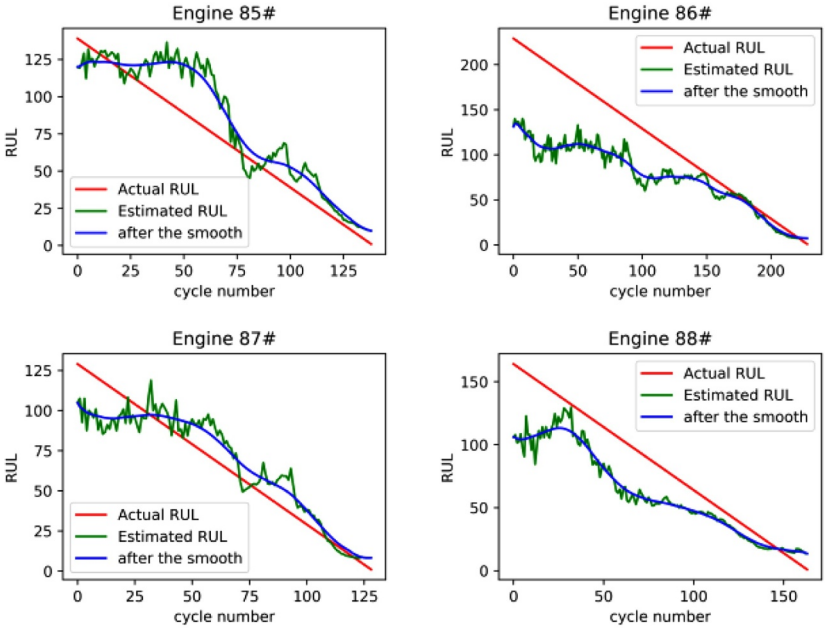
| N | 5 | 10 | 15 | 25 | 50 |
|---|---|---|---|---|---|
| error | 40.8 | 39.3 | 38.4 | 36.3 | 34.2 |
| time(min) | 1.21 | 2.01 | 3.30 | 3.92 | 5.72 |

redundant information. The original data has 24 dimensions from different parts of aero-engine. The data contains many different signals which vary from temperature signals, pressure signals, speed signals, flow signals to dimensionless signals. According to the previous experience, the quality of the supplied signals is directly relevant to the accuracy of RUL prediction. KPCA removes redundant information and reduces the dimensions of data, which makes the data more concise and efficient.

A large amount of information reflecting the state of the machine in RUL prediction is contained in the nonlinear components of the data. While principal component analysis (PCA) can only extract the linear components in the original signal, KPCA is necessary to improve the accuracy of life prediction. KPCA transforms the original data from low-dimensional space to high-dimensional space by mapping function, so as to achieve linear separability. At the same time, through kernel function, KPCA transforms the dot product operation of mapping vectors in high-dimensional space into the operation between original sample space data vectors, which greatly reduces the complexity of the




(a) RUL with GRU




(b) RUL with LSTM

**Fig. 8.** RUL of 85#–88# enginesN.

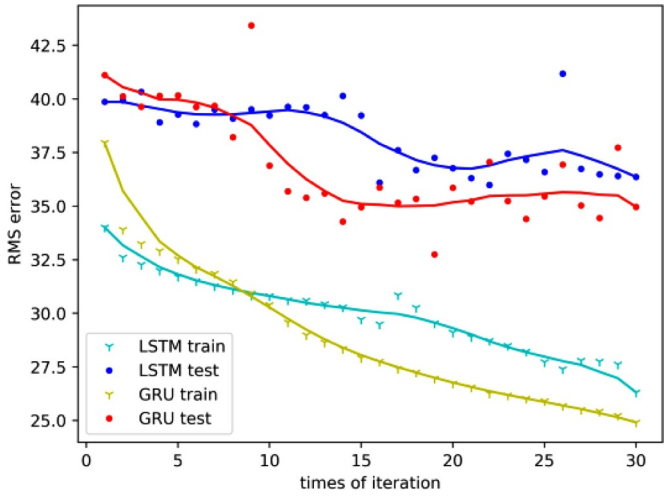



**Fig. 9.** Error curve of training set and testing set with GRU and LSTM.

calculation process.

### 3.2. Phase2: GRU

The objective of GRU is to convert the source data into the RUL label by learning the law of nonlinear degeneration. RNN has many different structures, which are applied to specific tasks. The structures with one input and many outputs are usually applied to automatic image annotation. The structures with many inputs and one output are usually applied to sentence sentiment analysis. The structures with many inputs and many outputs are usually applied to machine translation and video classification.

Here are two solutions in the RUL prediction.

(1) Sequence-to-sequence method

The full life cycle data sequences are chosen as the inputs and the RUL sequences, composed of RUL labels at each time, are chosen as the outputs.

(1) Sequence-to-one method

The full life cycle data sequences are divided into several fixed-length short sequences, which are chosen to be the inputs. This sequence of data is used to predict the RUL at the last cycle number.

In consideration of the varying length of full life cycle data, Sequence-to-sequence method will be more difficult to train. At the same time, the available source sequences are usually not full life cycle data. Using the model trained by full life cycle data for RUL prediction of incomplete life cycle data needs the extensibility verification. Particularly, Sequence-to-sequence method means each full life cycle data is regarded as only a sample, which will make the GRU over–fitting due to the sparse samples.

On the contrary, Sequence-to-one method will be both applicable whether it's full life cycle data or not. The model with fixed-length inputs will be easier to build and train. Each full life cycle data can be divided into hundreds of samples, to avoid the over-fitting problem.

Data preprocessing has been finished, and the results has been shown as Fig. 4. As solution (2) has been chosen, the inputs and outputs of the GRU will be set as Fig. 5. Next, the most important task is how to train the network to ensure it learn the rule of nonlinear degradation process well.

### 3.3. Network training

The common back-propagation algorithm doesn't work here, because RNN contains time sequence. A special method, back-propagation through time (BPTT) will be adapted to train the RNN. The network is trained to decrease the least square error $e_t$of the network output by updating the weight matrices.

The key point of back-propagation is calculating the gradient of the hidden layer's output$\frac{\partial e_t}{\partial h_t}$. Taking LSTM as an example, the back-propagation of $\frac{\partial e_t}{\partial h_t}$ comes from the current output layery$_t$ and the next hidden layer which can replace by the gates$O_{t+1}$, $F_{t+1}$, $I_{t+1}$, $G_{t+1}$. When we calculate the first gradient of $\frac{\partial e_{N-1}}{\partial h_{N-1}}$, the error from the gates function is considered as 0. The specific process is shown as following:

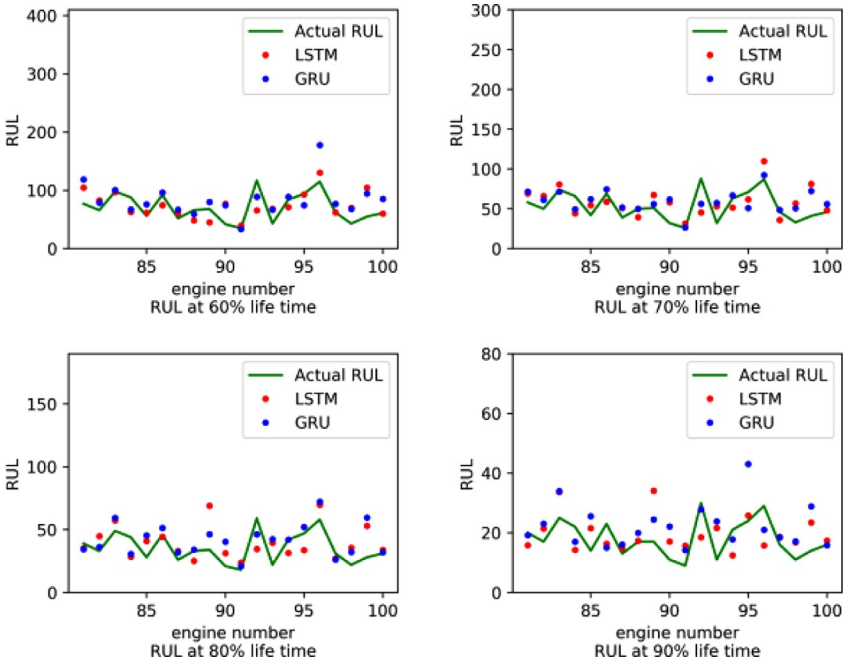



**Fig. 10.** RUL of 81#–100# engines at 60%, 70%, 80% and 90% lifetime.

**Table 3**
model description and comparison of LMD, F-NMD and T-NMD.

| N | LMD | F-NMD | T-NMD |
|---|---|---|---|
| Full name | linear multistate deterioration | fixed-structure nonlinear multistate deterioration | structurally optimized nonlinear multistate deterioration |
| Basic model | PCA + NHSMM | GKPCA + NHSMM | GKPCA + NHSMM + enumeration method |
| Advantages | – | consider nonlinearity | structural parameters are optimized |
| Disadvantages | ignore nonlinearity | structural parameters are not optimal | – |

**Table 4**
RMS error of RUL prediction and training time of 81# − 100# engines with different models.

| Models | LMD | F-NMD | T-NMD | GRU | LSTM |
|---|---|---|---|---|---|
| Error | 53.7 | 47.8 | 42.7 | 35.0(32.6) | 36.2(34.6) |
| Time(h) | 43 | 46.5 | 46.5 | 0.34 | 0.68 |

**Table 5**
prediction error of 81# − 100# engines with different models.

| No. | LMD | F-NMD | T-NMD | GRU | GRU_s | LSTM | LSTM_s |
|---|---|---|---|---|---|---|---|
| 81 | 44.8 | 39.5 | 24.2 | 29.4 | 22 | 32.7 | 31.2 |
| 82 | 51.3 | 28.7 | 10.5 | 20 | 18.7 | 25.7 | 23.9 |
| 83 | 38.3 | 64.8 | 47.7 | 32.1 | 30 | 16.2 | 11 |
| 84 | 92.7 | 57 | 64.3 | 35.4 | 33.3 | 38.5 | 37.8 |
| 85 | 17.5 | 21.5 | 6.4 | 21.6 | 19.4 | 19.5 | 16.8 |
| 86 | 68.6 | 64.1 | 66.1 | 21.1 | 15 | 52.7 | 51.9 |
| 87 | 15.4 | 19.9 | 18.8 | 31.7 | 27.8 | 14.7 | 12.1 |
| 88 | 27.3 | 28.6 | 19.1 | 21.2 | 17.4 | 27.8 | 27 |
| 89 | 25.9 | 26 | 12.6 | 16.2 | 13.1 | 29.8 | 22.8 |
| 90 | 67.2 | 30.4 | 40 | 28.5 | 27.1 | 33.9 | 32.3 |
| 91 | 27 | 35.5 | 23.9 | 8.3 | 7.8 | 7.7 | 5.2 |
| 92 | 86.1 | 98.1 | 83.6 | 74 | 72.7 | 77.7 | 77 |
| 93 | 61 | 26.3 | 42.9 | 26.2 | 22.3 | 23.2 | 20.9 |
| 94 | 83 | 52 | 56 | 22.9 | 22.1 | 27.5 | 27.2 |
| 95 | 35.6 | 61.7 | 42.2 | 34.7 | 34.5 | 19 | 17.6 |
| 96 | 70.3 | 86.8 | 79 | 79.3 | 76.9 | 74.6 | 72.8 |
| 97 | 45.2 | 27.2 | 9.1 | 12.4 | 9.6 | 15.7 | 13.1 |
| 98 | 58.5 | 34.9 | 38.3 | 34 | 31.8 | 40.6 | 39.8 |
| 99 | 28.6 | 18.5 | 17.1 | 32.9 | 27.7 | 30.6 | 26 |
| 100 | 22.6 | 23.1 | 8.21 | 28.6 | 22.6 | 18.3 | 16.8 |
| RMS | 53.7 | 47.8 | 42.7 | 35.0 | 32.6 | 36.2 | 34.6 |

$$\begin{aligned}\frac{\partial e_t}{\partial h_t} &= \frac{\partial y_t}{\partial h_t}\frac{\partial e_t}{\partial y_t} + \frac{\partial I_{t+1}}{\partial h_t}\frac{\partial e_{t+1}}{\partial I_{t+1}} + \frac{\partial O_{t+1}}{\partial h_t}\frac{\partial e_{t+1}}{\partial O_{t+1}} + \frac{\partial F_{t+1}}{\partial h_t}\frac{\partial e_{t+1}}{\partial F_{t+1}} + \frac{\partial G_{t+1}}{\partial h_t}\frac{\partial e_{t+1}}{\partial G_{t+1}} \\ &= V^T\frac{\partial e_t}{\partial y_t} + W_I{}^T\frac{\partial e_{t+1}}{\partial I_{t+1}} + W_O{}^T\frac{\partial e_{t+1}}{\partial O_{t+1}} + W_F{}^T\frac{\partial e_{t+1}}{\partial F_{t+1}} + W_G{}^T\frac{\partial e_{t+1}}{\partial G_{t+1}}\end{aligned} \tag{14}$$

After getting the gradient of $\frac{\partial e_t}{\partial h_t}$, the gradient of current hidden layer's gates function can be easily inferred from forward transfer formulas.

$$\begin{aligned}\frac{\partial e_t}{\partial S_t} &= \frac{\partial e_t}{\partial h_t}{*}O_t{*}\tanh'(S_t^0) + F_{t+1}{*}\frac{\partial e_{t+1}}{\partial S_{t+1}} \\ \frac{\partial e_t}{\partial O_t} &= \frac{\partial e_t}{\partial h_t}{*}\tanh(S_t){*}\sigma'(O_t^0) \\ \frac{\partial e_t}{\partial F_t} &= \frac{\partial e_t}{\partial S_t}{*}S_{t-1}{*}\sigma'(F_t^0) \\ \frac{\partial e_t}{\partial I_t} &= \frac{\partial e_t}{\partial S_t}{*}G_t{*}\sigma'(I_t^0) \\ \frac{\partial e_t}{\partial G_t} &= \frac{\partial e_t}{\partial S_t}{*}I_t{*}\tanh'(G_t^0)\end{aligned} \tag{15}$$

For GRU, the gradient of $\frac{\partial e_t}{\partial h_t}$ is calculated as:

$$\begin{aligned}\frac{\partial e_t}{\partial h_t} &= \frac{\partial y_t}{\partial h_t}\frac{\partial e_t}{\partial y_t} + \frac{\partial h_{t+1}}{\partial h_t}\frac{\partial e_{t+1}}{\partial h_{t+1}} + \frac{\partial \tilde{h}_{t+1}}{\partial h_t}\frac{\partial e_{t+1}}{\partial \tilde{h}_{t+1}} + \frac{\partial z_{t+1}}{\partial h_t}\frac{\partial e_{t+1}}{\partial z_{t+1}} + \frac{\partial r_{t+1}}{\partial h_t}\frac{\partial e_{t+1}}{\partial r_{t+1}} \\ &= V^T\frac{\partial e_t}{\partial y_t} + (1 - z_{t+1}){*}\frac{\partial e_{t+1}}{\partial h_{t+1}} + W^T\frac{\partial e_{t+1}}{\partial \tilde{h}_{t+1}}{*}r_{t+1} + W_z{}^T\frac{\partial e_{t+1}}{\partial z_{t+1}} + W_r{}^T\frac{\partial e_{t+1}}{\partial r_{t+1}},\end{aligned} \tag{16}$$

The gradient of each gate can also be inferred from the forward transfer formulas:

$$\begin{aligned}\frac{\partial e_t}{\partial z_t} &= \frac{\partial e_t}{\partial h_t}{*}(\tilde{h}_t - h_{t-1}){*}\sigma'(z_t^0) \\ \frac{\partial e_t}{\partial \tilde{h}_t} &= \frac{\partial e_t}{\partial h_t}{*}z_t{*}\tanh'(\tilde{h}_t^0) \\ \frac{\partial e_t}{\partial r_t} &= W^T\frac{\partial e_t}{\partial h_t}{*}h_{t-1}{*}\sigma'(r_t^0)\end{aligned} \tag{17}$$

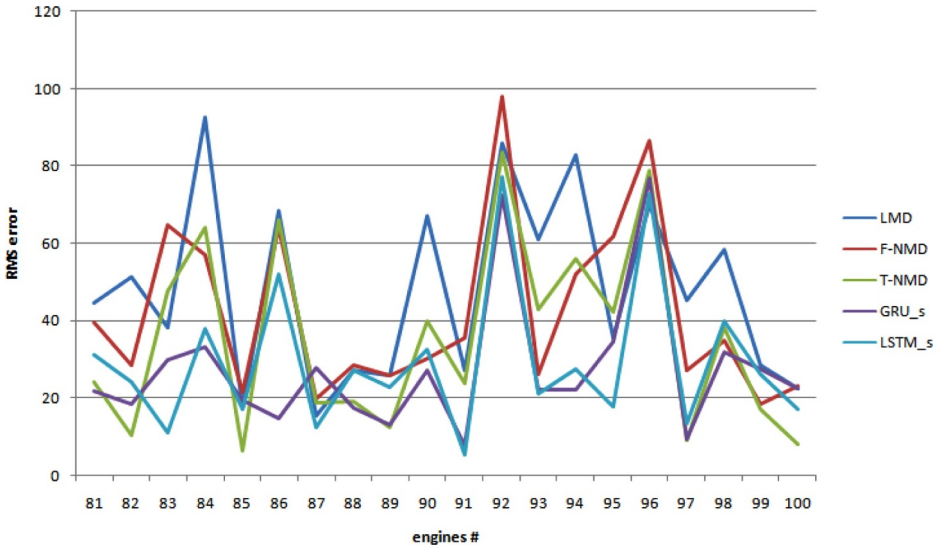



**Fig. 11.** error of 81#–100# engines with different models.

The formula (14)–(17) mainly describe the BPTT process. It can be easily found that the BPTT of GRU is more concise, of course, it costs less time than the BPTT of LSTM. The specific different in time consuming will be shown in Section IV.

In all, this general solution mainly contains two phases: KPCA and GRU. Correspondingly, it is used to solve two key problems respectively: high-dimensional data and deterioration nonlinearity. KPCA is used to reduce dimensionality of source data and extract nonlinear feature. GRU is used to establish the nonlinear deterioration model for RUL prediction. In the end, BPTT algorithm is used for the network training of GRU. The whole process of this general solution has been shown in Fig. 6.

## 4. Case study

### 4.1. Data description

C-MAPSS, called the commercial modular aero-propulsion system simulation, is a flexible turbofan engine simulation environment with easy access to health, control and engine parameters through a graphical user interface, established by US Army Research Laboratory, Glenn Research Center [25]. The diagram of engine simulated in C-MAPSS has been shown as Fig. 7. C-MAPSS can be used for the development and validation of control and diagnostic algorithms and it runs faster than real time. The C-MAPSS-Data is widely used by scholars and proved to be reasonable and reliable which is available through NASA.

Experimental scenario data sets consist of multiple multivariate time series. Each time series is from a different engine, the data can be considered to be from a fleet of engines of the same type. Each engine starts with different degrees of initial wear and manufacturing variation which is unknown to the user. The data are provided as a zip-compressed text file with 26 columns of numbers, separated by spaces. Each row is a snapshot of data taken during a single operational cycle, each column is a different variable. The physical meanings of C-MAPSS data sources have been shown in Table 1.

In order to verify the reliability of GRU, 1# − 80#engines are chosen as training set, and the rest 81# − 100#engines are chosen as testing set. Through the BPTT of GRU with the RUL labels in training set, GRU will learn well. And the testing set is used to verify the reliability of GRU by comparing the outputs of GRU with the actual RUL provided in testing set.

### 4.2. Parameter determination and results analysis

In order to describe the difference between the estimated RUL and the actual RUL quantitatively, root mean square (RMS) is used to represent the prediction accuracy which is one of the most popular methods.

The sequence length N is one of the most important parameters directly relevant to the ultimate accuracy of the results. At the same time, N also decides the depth of RNN. In order to learn how the model performs when the N changes, Several sets of experiments were carried out. Using the single-variable control method, 10 repeated tests were carried out at $N = 5$, 15, 25 and 50 respectively. The results are shown at the Table 2. As N changes from 5 to 50, the prediction error decrease from 40.8 to 34.2, and training time increase from 1.21 to 5.72 minutes. It is easy to find that the prediction error is negatively correlated with $N$. Theoretically, if you want to reduce the model prediction error further, you can continue to increase $N$. However, in practice, $N$ is constrained by RUL of the sample (usually $N$ should be less than half of the total life cycle sequence length), and the larger $N$ is, the longer the data acquisition preparation time is, which is not conducive to engineering application. Therefore, we will choose $N = 50$ in next experiments for simplifying the comparison of different models.

Fig. 8(a) describes the RUL of 85# − 88#engines with the model GRU, while Fig. 8(b) is the RUL of 85# − 88# engines with the model LSTM. Fig. 8(a) and (b) show that the RUL of 85# − 88# engines with GRU is closer to the actual values with a smaller deviation in general than that with LSTM. Because of the result fluctuates greatly, sliding average method is applied to smooth the results, increasing the prediction accuracy effectively.

Fig. 9 describes the error curve of training set and testing set with GRU and LSTM. The error curve of GRU drops faster than LSTM, no matter in training set or in testing set. In the training set, the RMS error decreases monotonically, and through 30 times of iteration, RMS error of GRU decreases from 38 to 25, yet LSTM decreases from 34 to 27. In the testing set, the minimum error of GRU is around 35, appearing at 14 times of iteration, yet the minimum error of LSTM is around 37, appearing at 20 times of iteration. Therefore, GRU has a faster convergence rate than LSTM.

As further estimation of the results with different models, the RUL of 81#–100# engines at 60%, 70%, 80% and 90% lifetime are shown in Fig. 10. It shows the difference results of GRU and LSTM in different engines at the specific life time.

In order to verify the efficiency of GRU more objectively, the LMD, F-NMD and T-NMD models presented by Chen et al. [15] is selected to be the control group. All three are NHSMM based models, and their full names and advantages and disadvantages are listed in Table 3. The RMS error of RUL prediction and training time are shown in Table 4, where the errors of GRU and LSTM in the brackets are the results after the smooth.

Quantitatively, Table 4 shows RMS error of RUL prediction and training time of 81#-100# engines with different models. Compared with RNN and HHM, the significant different is the training time. Although HHM behaves pretty well in the tasks with time sequence, all the HHM models spend much time for training. Even more, if larger state space dimension of HMM was considered, the training time would explode to be very long. RNN models (both GRU and LSTM) reduce the training time from 46 h with HHM to less than one hour. Further more, the training time of RNN models increases approximately in direct proportion to the depth of RNN, thereby avoiding the explosion of training time. In terms of prediction accuracy, both GRU and LSTM behave better than LMD and T-NMD. Particularly, sliding average method is applied to smooth the results, decreasing the error significantly from 35.0 to 32.6 in GRU and from 36.2 to 34.6 in LSTM. Compared with GRU and LSTM, in items of both the prediction accuracy and the training time, GRU performs better than LSTM. To be more specific, the error of each engines are shown in Table 5 and Fig. 11.

When this proposed scheme is applied in practical industry, following items should be noticed. The equipment model in the training set must be the same as the equipment model to be tested. The sample distribution of training set needs to traverse all failure modes as far as possible. The parameters of the network model need to be modified according to the actual training effect. If all these items are concerned, a well-trained KPCA-GRU model will perform excellently in practical industry as well as in simulation dataset.

## 5. Conclusion

In this paper, a general two-step solution for RUL prediction of nonlinear deterioration process is proposed. In the solution, KPCA is applied as the first step for nonlinear feature extraction. The second step is using GRU, a simplified network of LSTM with fewer parameters, to predict RUL. C-MAPSS-Data, a dataset of aero-engines with nonlinear deterioration process, was used to test the proposed method. Results show that GRU performs better than LSTM both in training time and prediction accuracy and is able to provide better RUL prediction for nonlinear deterioration process of complex systems. Comparing with HHM, GRU based solution prominently decreases the training time from more than 40 h to less than one hour. As for prediction accuracy, GRU based solution is also better than HHM. The study in this paper provides a general solution with GRU as a basic framework for RUL

prediction of nonlinear deterioration process.

In the future work, we will study how to extract nonlinear features more effectively to further improve the prediction accuracy.

## Acknowledgments

The authors would like to sincerely thank all the anonymous reviewers for the valuable comments that greatly helped to improve the manuscript.

This work was supported financially in part by the National Natural Science Foundation of China under Grant 51875436 and Grant 61633001, in part by the China Postdoctoral Science Foundation under Grant 2018M631145.

## Appendix

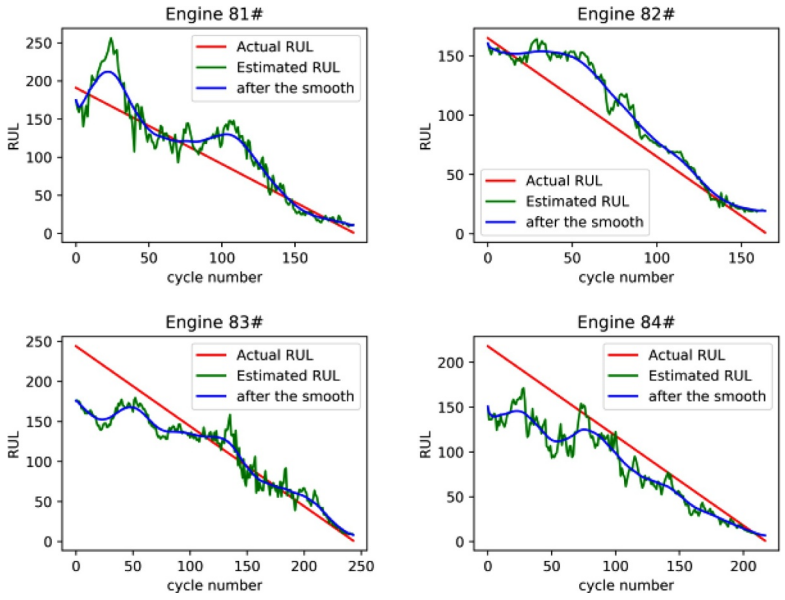



**Fig. A1.** RUL of 81#–84# engines with GRU.

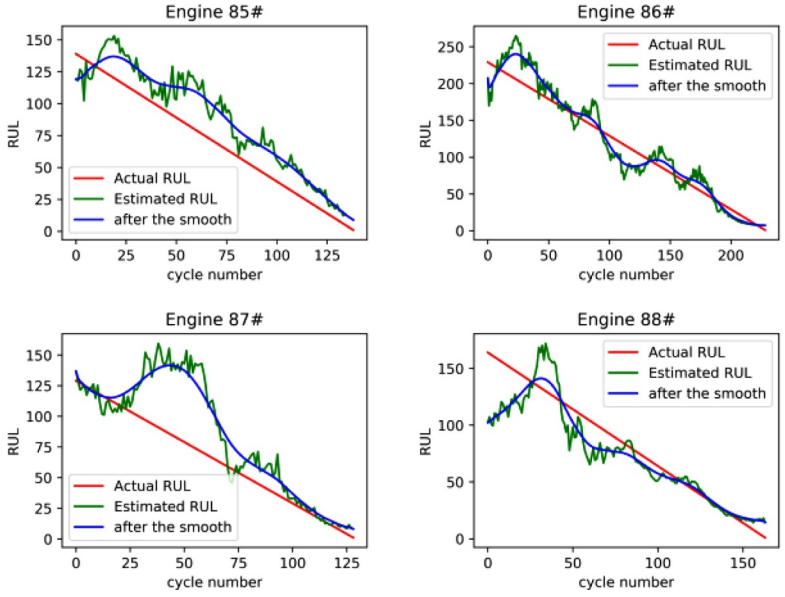



**Fig. A2.** RUL of 85#–88# engines with GRU.

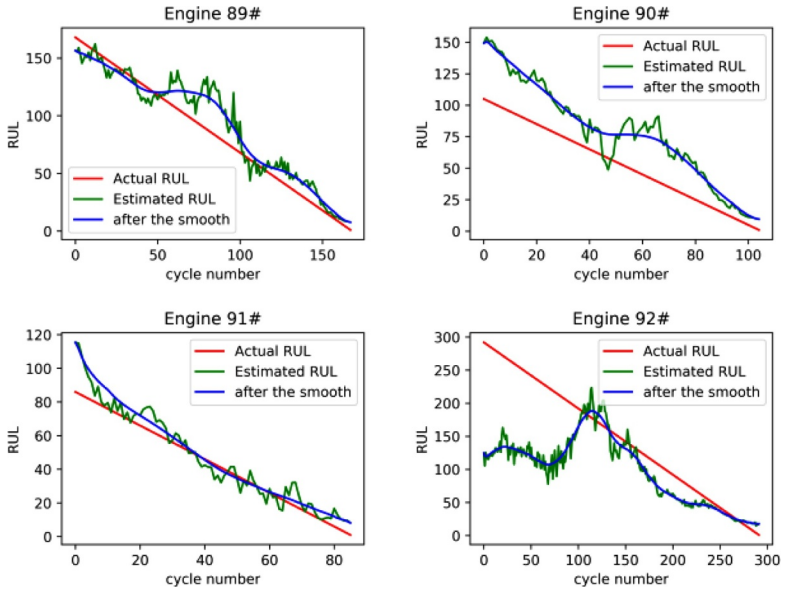



**Fig. A3.** RUL of 89#–92# engines with GRU.

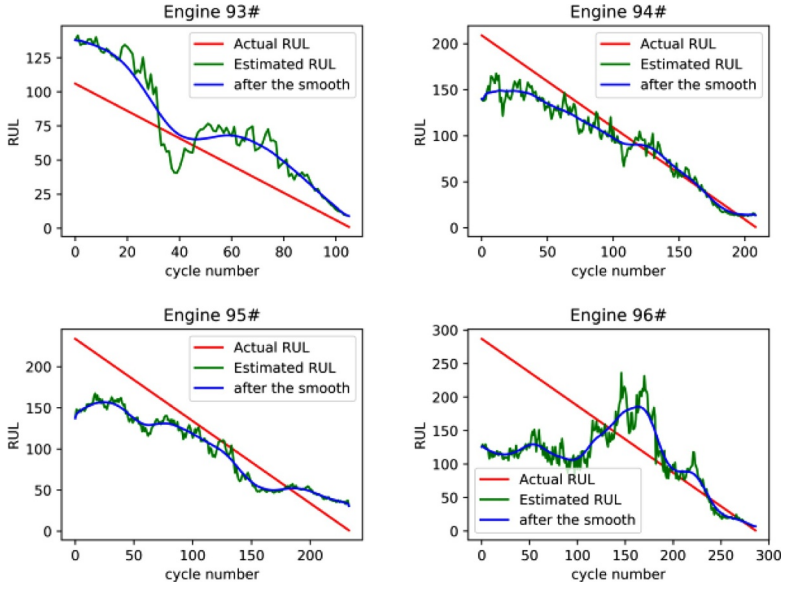



**Fig. A4.** RUL of 93#–96# engines with GRU.

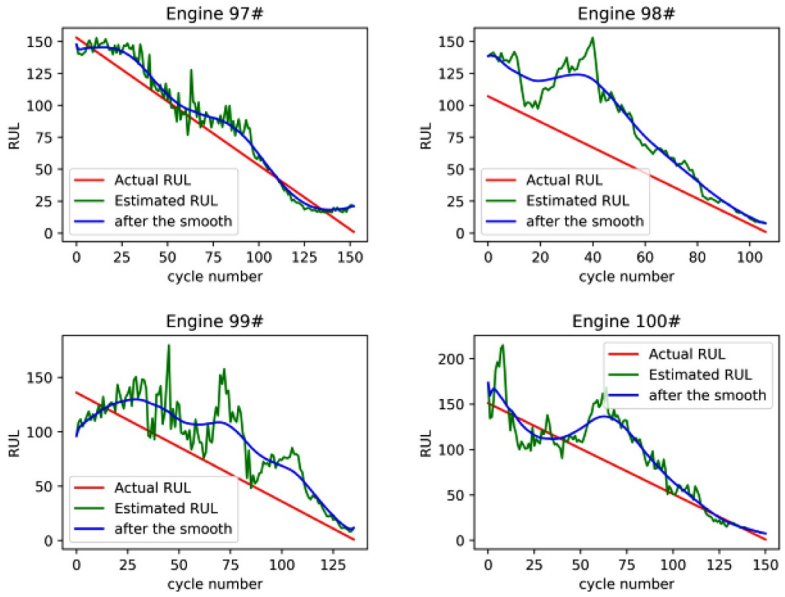



**Fig. A5.** RUL of 97#–100# engines with GRU.

## References


[1] Dong M, Peng Y. Equipment PHM using non-stationary segmental hidden semi-Markov model. Rob Comput Integr Manuf 2011;27(3):581–90.
[2] Si X-S, et al. Remaining useful life estimation – A review on the statistical data driven approaches. Eur J Oper Res 2011;213(1):1–14.
[3] Ahmad W, et al. A reliable technique for remaining useful life estimation of rolling element bearings using dynamic regression models. Reliab Eng Syst Safety 2018.
[4] Hu Y, et al. A prediction method for the real-time remaining useful life of wind turbine bearings based on the Wiener process. Renewable Energy 2018.
[5] Huang Z, et al. Remaining useful life prediction for an adaptive skew-Wiener process model. Mech Syst Sig Process 2017;87:294–306.
[6] Zhang Z, et al. Degradation data analysis and remaining useful life estimation: A review on Wiener-process-based methods. Eur J Oper Res 2018.
[7] Le Son K, Fouladirad M, Barros A. Remaining useful lifetime estimation and noisy gamma deterioration process. Reliab Eng Syst Safety 2016;149:76–87.
[8] Ling MH, Ng HKT, Tsui KL. Bayesian and likelihood inferences on remaining useful life in two-phase degradation models under gamma process. Reliab Eng Syst Safety 2017.
[9] Baptista M, et al. Remaining useful life estimation in aeronautics: combining data-driven and Kalman filtering. Reliab Eng Syst Safety 2018.
[10] Son J, et al. Remaining useful life prediction based on noisy condition monitoring signals using constrained Kalman filter. Reliab Eng Syst Safety 2016;152:38–50.
[11] Duong PLT, Raghavan N. Heuristic Kalman optimized particle filter for remaining useful life prediction of lithium-ion battery. Microelectron Reliab 2018;81:232–43.
[12] Liu Q, et al. A novel method using adaptive hidden semi-Markov model for multi-sensor monitoring equipment health prognosis. Mech Syst Sig Process 2015;64-65:217–32.
[13] Chen Z, et al. Hidden Markov model with auto-correlated observations for remaining useful life prediction and optimal maintenance policy. Reliab Eng Sys Safety 2017.
[14] Li X, et al. Optimal Bayesian control policy for gear shaft fault detection using hidden semi-Markov model. Comput Ind Eng 2018;119:21–35.
[15] Chen G, et al. Hyper-parameter optimization based nonlinear multistate deterioration modeling for deterioration level assessment and remaining useful life prognostics. Reliab Eng Syst Safety 2017;167.
[16] Moghaddass R, Zuo MJ. An integrated framework for online diagnostic and prognostic health monitoring using a multistate deterioration process. Reliab Eng Syst Safety 2014;124:92–104.
[17] Guo L, et al. A recurrent neural network based health indicator for remaining useful life prediction of bearings. Neurocomputing 2017;240:98–109.
[18] Liu H, et al. Fault diagnosis of rolling bearings with recurrent neural network-based autoencoders. ISA Trans 2018.
[19] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9(8):1735–80.
[20] Hinchi AZ, Tkiouat M. Rolling element bearing remaining useful life estimation based on a convolutional long-short-term memory network. Procedia Comput Sci 2018;127:123–32.
[21] Yuan M, Wu Y, Lin L. Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network. IEEE international conference on aircraft utility systems. 2016.
[22] Malhotra, P., et al., Multi-sensor prognostics using an unsupervised health index based on LSTM encoder-decoder. 2016.
[23] Cho K, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. Comput Sci 2014.
[24] Graves A. 2005 special issue: framewise phoneme classification with bidirectional LSTM and other neural network architectures. Elsevier Science Ltd; 2005. p. 602.
[25] Saxena A, et al. Damage propagation modeling for aircraft engine run-to-failure simulation. International conference on prognostics and health management. 2008.