Journal of Computers Vol. 32 No. 3, 2021, pp. 69-85
doi:10.3966/199115992021063203006

Temporal Convolutional Networks Applied to Remaining Useful Life Prediction of Turbofan Engine

Yue-feng Liu*, Gong Zhang, Jia-qi Li

School of Information Engineering, Inner Mongolia University of Science and Technology, Ardin Street 007, Baotou, China
liuyuefeng01035@163.com,18435205149@163.com,jiaqi0724@live.com

Received 9 August 2020; Revised 15 December 2020; Accepted 21 January 2021

Abstract. In recent years, turbofan engine failures have frequently occurred, traditional breakdown maintenance has been difficult to meet the demand. Remaining useful life (RUL) prediction technology has become one of the effective ways to solve the above-mentioned problems. To accurately obtain the RUL of the turbofan engine, an RUL prediction method based on Temporal Convolutional Networks (TCN) is proposed in this paper. The overall network can be divided as follows: Firstly, combining the advantages of LSTM and autoencoder to complete the feature extraction of sequence data. Secondly, TCN is used in the RUL prediction part. TCN does not disclose future sequence information and it has a larger and more flexible receptive field. TCN also features the residual structure to make full use of the original input information and to avoid the disappearance of gradients. The effectiveness of the proposed method is verified in CMAPSS datasets. Finally, compared with other excellent RUL prediction methods, the proposed method improves the prediction accuracy on complex datasets.

Keywords: turbofan engine, Remaining Useful Life (RUL), Autoencoder, Long Short-Term Memory (LSTM), Temporal Convolutional Networks (TCN)

1 Introduction

In contemporary society, both civil and military aircraft played an important role in all aspects of people's life. In recent years, the continuous occurrence of aircraft failures led to heavy losses of human property. Traditional breakdown maintenance was replaced by Condition-based maintenance (CBM), the most important thing in CBM is to determine the health status of the turbofan engine. For the health monitoring of the turbofan engine, one of the important methods is to predict RUL through the data obtained from the relevant sensors [1-2]. Therefore, the research on RUL prediction methods is particularly important. According to the related work [3], the RUL prediction methods were generally divided into three categories: model-based method, AI-based method, and hybrid method.

The model-based method is to establish specific physical models and statistical models for the research objects. In terms of physical models, Paris-Erdogan (PE) model was the most widely used [4-5]. In terms of statistical models, Wiener Process and Gaussian Process regression [6-12] also played a key role. Although model-based methods have shown good results, it is difficult to obtain accurate physical and statistical models of the research object, and a large amount of domain knowledge is required.

With the development of artificial intelligence and sensor technology, the amount and availability of data is increasing. Traditional model-based methods are difficult to process such a large amount of data. AI-based methods have unique advantages in processing big data. Therefore, AI-based methods have become a hot research field. Deep Belief Network (DBN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), LSTM, Gate Recurrent Unit (GRU), and other methods had gradually stepped onto the stage of RUL prediction. Zhang et al. proposed a united multi-objective deep belief network (MODBNE) for RUL prediction, [13]. Compared with DBN, CNN has stronger feature

* Corresponding Author

extraction capabilities. Babu et al. used CNN to predict RUL for the first time [14]. CNN works well as feature extraction, but its application in RUL prediction needs to be improved. Later work used it as a feature extraction tool, For example, Li et al. took advantage of the local feature extraction of CNN, CNN was used to extract multi-dimensional features, the processed data was input into DNN to predict the mechanical bearing RUL [15]. In their work, they used deep convolution neural networks (DCNN) to predict RUL [16]. Although CNN showed good performance in data feature extraction and RUL prediction, CNN was not sensitive to past data and could not make full use of the information of sequence data. LSTM and GRU are specially designed to process time series data. LSTM and GRU used the gate mechanism to solve gradient disappearance and gradient explosion problems in RNN. In the following work, the advantages of LSTM are also proved from an experimental perspective. For example, Zheng et al. applied LSTM to RUL prediction, the final result of RUL prediction was better than the previous method [17]. Wang et al. applied bidirectional LSTM (Bi-LSTM) to the RUL prediction of the turbofan engine, which was more accurate than the traditional LSTM prediction [18]. Miao et al. used the Dual-Task Deep LSTM joint learning turbofan engine degradation assessment and RUL. It improved the accuracy of RUL prediction compared with the previous method [19]. LSTM and its improved methods were currently one of the most utilized methods in the industry and their performance were also at the forefront of the industry. At the same time, the GRU obtained by simplifying LSTM was also applied because of its simple structure and simpler model training. Chen et al. used Kernel Principal Component Analysis (KPCA) for data feature extraction and finally used GRU for RUL prediction [20]. The method based on AI could make full use of a large amount of data brought by the big data era and had higher accuracy than the traditional methods based on model, but it also made the RUL prediction model more complex, time consuming and costly.

The above methods are combined to obtain a hybrid method. Because it can combine the advantages of various methods, it has become the mainstream method. The more widely used ones are the combination of AI-based methods. For example, Li et al. called the combination of CNN and LSTM as directed acyclic graph (DAG). The combination of CNN and LSTM was used for feature extraction. Finally, the processing results were input into LSTM to get the prediction results of RUL [21]. Liu et al. used CNN and Bi-LSTM encoders for RUL prediction, CNN and Bi-LSTM were used for feature extraction in the encoder and the full connection layer in the decoder was used for RUL prediction [22]. Some scholars combined model-based methods with AI-based methods. For example, Wu et al. combined Autoregressive moving average model (ARMA) and radial basis function neural network (RBFNN) to predict RUL. Experiments showed that the proposed method had a good performance on different datasets [23]. Zhang et al. combined PF with the exponential model for RUL prediction. Compared with the combination of other models, good results had been achieved [24].

The hybrid method is still a hot direction in this field. However, because there are not enough neural networks for time series in this field in recent years, the network model used for RUL prediction is relatively fixed. The TCN network was first proposed in 2018, which caused a great sensation in the academic field [25]. The biggest feature of TCN lied in the use of causal convolution and dilation convolution. The advantage of the former was that it would not disclose future information to the prediction of the past period time. The advantage of the latter was that it could get a larger receptive field in a simple network structure. Because of the design of the residual block structure in the TCN, original input and data processed by the residual block could be integrated through the activation function. The existence of one-dimensional convolution made the same shape of the input and output. It also avoided gradient disappearance. In order to solve the problem of fixed models and the disadvantages of LSTM in the field of RUL prediction, this paper proposes an LSTM-based autoencoder as a feature extraction tool and TCN as an RUL prediction tool. The main technical contributions of this paper are as follows:

1. This article uses an LSTM-based autoencoder for feature extraction. Compared with Principal component analysis (PCA), KPCA, and autoencoder, the adopted method can better extract the hidden information in time series data.

2. This paper uses the TCN network for the first time in the field of RUL prediction of the turbofan engine. Compared with classical methods such as LSTM, the results of RUL prediction are excellent.

The arrangement of this paper is as follows: in the second section, related theories and calculation autoencoder based on LSTM and TCN were introduced. In the third section, the experiment in detail was introduced, including the introduction of the dataset, the method of data preprocessing, the specific parameter setting of the model, and the setting of the experimental environment. In the fourth section is

the analysis of the experimental results and the comparison with the current excellent methods. The final section is the conclusion of this paper and the next work.

2 Methodology

A generalized solution to the RUL prediction problem, each part of the proposed overall network, and the corresponding principles and mathematical calculations will be introduced.

2.1 The Proposal and Solution of the Problem

In order to predict RUL of the turbofan engine, data can be utilized are the values of three operating conditions and 21 sensors. In offline training, *Maxcycles* represents the maximum life of the current engine, *Cycle* represents the number of cycles that have been performed, the corresponding RUL value y_{true} can be obtained through a simple calculation by equation (1).

$$y_{true} = Maxcycles - Cycle . \tag{1}$$

So, the overall prediction problem can be expressed through equation (2).

$$y_{pred} = f(X,\theta) . \tag{2}$$

The method can be simply regarded as the function f, X represents input data, θ represents relative parameters. Input data can be calculated through f to get the corresponding predicted value y_{pred} of RUL. Objective function can be minimized by equation (3), in which N stands for training size.

$$Min\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_{ture} - y_{pred})} . \tag{3}$$

2.2 The Overall Process of the Network

Input sequence is given by equation (4) to equation (7).

$$X = [x_1, x_2, ..., x_{t+1}, ..., x_T] \in R^{T*S} . \tag{4}$$

$$x_t = [x_t^1, x_t^2, ..., x_t^S] \in R^{1*S}, t \in [1, T] . \tag{5}$$

$$x^s = [x_1^s, x_2^s, ..., x_T^s] \in R^{T*1}, s \in [1, S] . \tag{6}$$

$$X_t = [x_t, x_{t+1}, ..., x_{t+n-1}] \in R^{n*S}, t \in [0, T+1-n] . \tag{7}$$

Where X represents the dataset with T length and S features, and x_t represents a piece of data with S features at time t. x^s represents all the data containing the s-th feature, and X_t represents a piece of data intercepted through Time windows (TW) processing. The difference is that the sequence length of x_t is 1 and the sequence length of X_t is n. In order to extract the hidden information of the long-term dependent sequence data. Finally, X_t can be superimposed and intercepted to get the final sequence data X_T.

Overall network structure is shown in Fig. 1. It broadly includes the process from data acquisition to RUL prediction.

2.3 Data Preprocessing

First of all, the selected data features have different ranges and the range among different features is very large. If original data is directly input into the model, it will slow down the model learning and fitting speed. Therefore, the method of min-max normalization to process original data by equation (8).

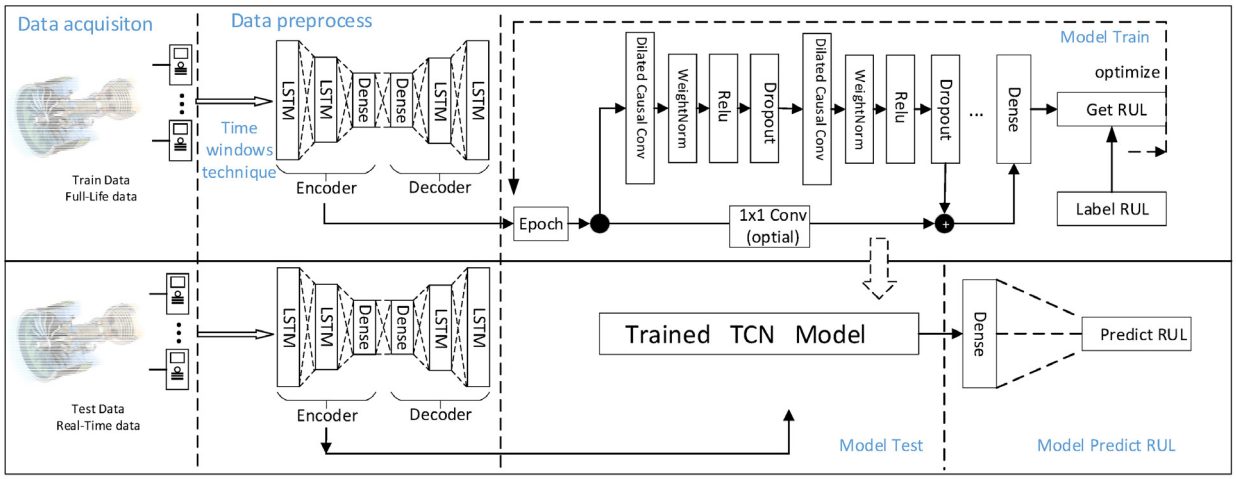


Fig. 1. The process of data acquisition to RUL predict

$$x^{s'} = \frac{x^s - x^s_{\min}}{x^s_{\max} - x^s_{\min}}. \tag{8}$$

Where $x^s_{\min}$ and $x^s_{\max}$ represent minimum and maximum values of the s-th feature. It is worth noting that only normalize 21 sensors and three operating conditions. As shown in Fig. 2, the data normalized by min-max is processed by the TW technology.

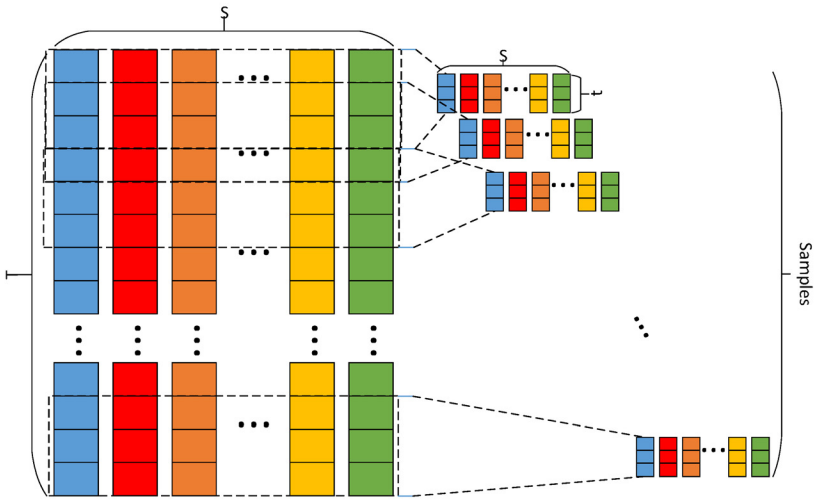


Fig. 2. Time Windows Technology

The format of our original data is (T, S), T represents the total length and S represents the number of features. After TW processing, Samples (t, S) can be obtained, where t represents the selected time step, (Samples, t, S) is the result obtained by superimposing X_t, where t belongs to (0, T).

Autoencoder. In Image Processing, Natural Language Processing, and other fields, autoencoder is a sharp tool for data dimensionality reduction. It is different from linear dimensionality reduction methods such as PCA. As a tool of nonlinear dimensionality reduction, autoencoder has achieved good results on many targets.

Data from previous processing were input into the autoencoder based on LSTM. Its structure is shown in the preprocessing part of the data, Fig. 1. Through model training, the input and output are approximately the same. Therefore encoder was selected as an important part of our data processing.

LSTM. LSTM is generally divided into input gate, output gate, forget gate and cell state, as shown in Fig. 3.

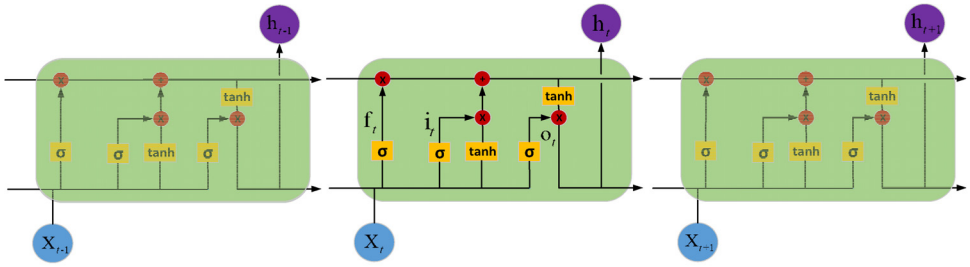


Fig. 3. The structure of LSTM

The specific calculation process is as follows. In order not to lose generality, X_t was set as input:

$$\mathrm{i}^t = \sigma(W_{ix}X_t + W_{ih}h^{t-1} + W_{ic}c^{t-1} + b_i)\,. \tag{9}$$

$$f^t = \sigma(W_{fx}X_t + W_{fh}h^{t-1} + W_{fc}c^{t-1} + b_f)\,. \tag{10}$$

$$c^t = f^t \otimes c^{t-1} \oplus i^t \otimes \tanh(W_{cx}X_t + W_{ch}h^{t-1} + b_c)\,. \tag{11}$$

$$o^t = \sigma(W_{ox}X_t + W_{oh}h^{t-1} + W_{oc}c^t + b_o)\,. \tag{12}$$

$$h^t = o^t \otimes \tanh(c^t)\,. \tag{13}$$

i^t, f^t, c^t, o^t stands for values of input gate, forget gate, cell activation, output gate at t time. σ represents logistic sigmoid function. $\otimes$ represents element-wise multiplication, $\oplus$ represent Element-wise Summation. $W_{ix}, W_{ih}, W_{ic}, W_{fx}, W_{fh}, W_{fc}, W_{cx}, W_{ch}, W_{ox}, W_{oh}, W_{oc}$ represents the corresponding weight which i^t, f^t, c^t, o^t is conformity with $X_t, h^{t-1}, c^{t-1}, c^t$. b_i, b_f, b_c, b_o symbolizes bias corresponded to i^t, f^t, c^t, o^t.

Encoder. Next, the specific process of the encoder in the autoencoder will be introduced.

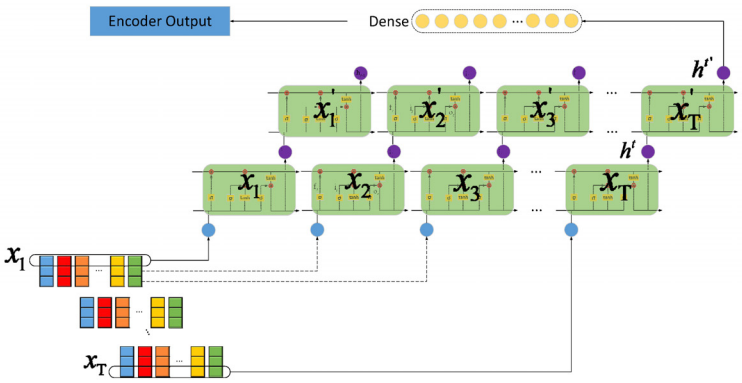


Fig. 4. The structure of Encoder

X after TW processing was input into encoder. Output h^t can be obtained by equation (9) to equation (13). Then h^t was input into LSTM. $h^{t'}$ can be obtained by above calculation. After a full connection layer, in equation (14), w represents weight, b represents bias. Preprocessed data can be obtained.

$$x = wh^{t'} + b. \tag{14}$$

2.4 The Proposed RUL Prediction Model

After completing the data preprocessing part, the data processed by the autoencoder based on LSTM in the previous step was input into the TCN for RUL prediction.

Causal Convolutions. To put it simply, TCN = 1D FCN + casual convolutions, the idea of one-dimensional full convolution is to make the length of the output sequence of each hidden layer consistent with the length of the input sequence. The main idea of causal convolution is that the output at t time is only related to the input before t time, which avoids disclosing future information. So the predicted results are not affected by future information. But there is a problem: if the length of the sequence is too long, the receptive field should be expanded by increasing the number of layers of the network. In order to reduce the complexity of the network, dilation convolution was introduced.

Dilated Convolutions. There is now a given input sequence X_t, given filter {0, ….., k-1}, the dilation convolution F is defined in the x_t^s input as:

$$F(x_t^s) = \sum_{i=0}^{k-1} f(i) \cdot x_{t-d \cdot i}^s . \tag{15}$$

d stands for dilation factor and k is the filter size. t-d*i represents the direction of the past. In a broad sense, when d is 1, the dilation convolution is the most common convolution. Besides, the receptive field can be increased by choosing a larger k and a larger d. As shown in Fig. 5, the effective receptive field of each layer that uses dilation convolution is (k-1) d. d = 2, k = 2, dilation = [1, 2, 4].

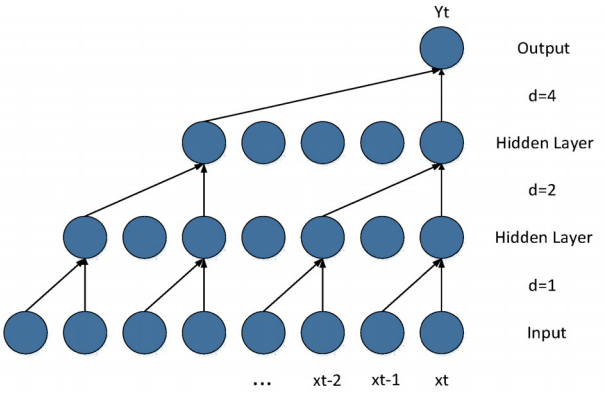


Fig. 5. The structure of Dilated Convolutions

Residual Connections. The concept of residual network is first put forward by [26], which effectively solves the problem of gradient disappearance. Its calculation equation (16) is as follows.

$$o = Activation(X + Q(X)) . \tag{16}$$

X is the input, $Q(X)$ represents the output after passing through the last hidden layer of the residual network, and is the activation function used by X and $Q(X)$. In order to combine them, one-dimensional convolution was used to make their shape the same, which can be explained in Fig. 6 and Fig. 7.

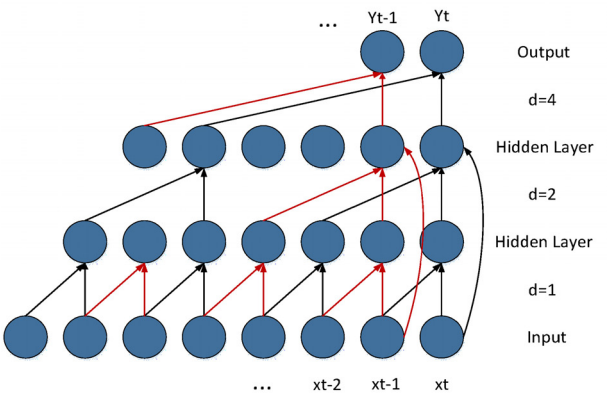


Fig. 6. The structure of Dilated Convolutions with Residual Connections

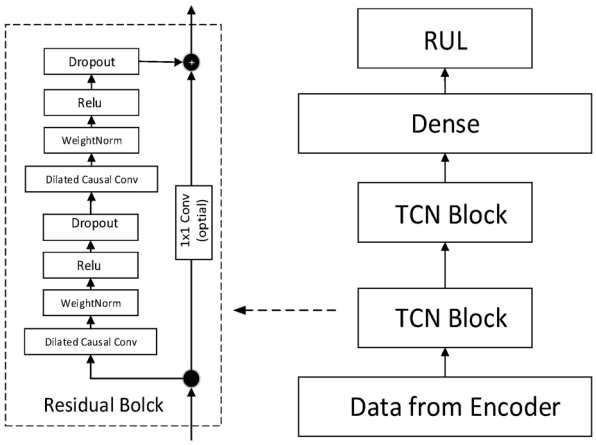


Fig. 7. The structure of RUL prediction

In the specific experiment, receptive field can be calculated by equation (17).

$$receptive\ field = K * nb\_stacks * dilation[-1]. \quad \textbf{(17)}$$

Where K stands for filter size. nb_stacks represents the number of residual blocks and $dilation[-1]$ represents the last number in dilation.

The specific calculation process is as follows: As shown in Fig. 7, In TCN Block, after equation (15), $F(x_t^s)$ will through an activation function in equation (18), its output is $p(x_t^s)$. Repeat the above process, output $Q(X)$ can be obtained through TCN Block, final TCN Block output o can be obtained by equation (16). After the TCN Block calculation, the data finally passes through a full connection layer to get the final predicted RUL.

$$p(x_t^s) = Relu(F(x_t^s)). \quad \textbf{(18)}$$

3 Experiment

Next, the source of the dataset will be introduced. Finally, this section will show the way of data preprocessing, RUL target function, evaluation indicators as well as experimental setting.

3.1 Introduction of Dataset

Turbofan engine degradation datasets are generated by C-MAPSS [27]. It presents the main architecture of the turbofan aircraft engine simulated in C-MAPSS, Fig. 8. The main components include a fan, a combustor, a low-pressure turbine (LPT), a low-pressure compressor (LPC), a high-pressure compressor (HPC), a high-pressure turbine (HPT), and a nozzle.

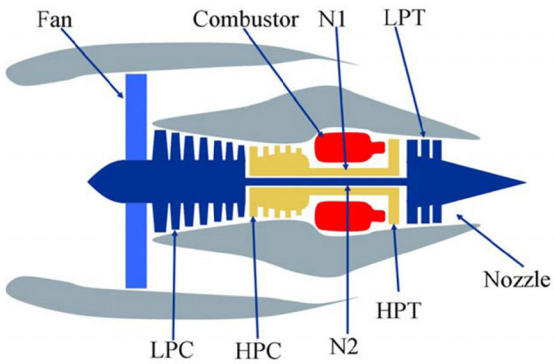


Fig. 8. A simplified diagram of the simulation engine in C-MAPSS [27]

CMAPSS datasets are divided into four sub-datasets, namely FD001 to FD004, as shown in Table 1. FD001 and FD003 are generally considered to be simple datasets because of the fewer operation conditions and failures. On the contrary, FD002 and FD004 are generally considered to be complex datasets. Time step for prediction was set as the smallest sequence -1 in the test. Because if the selected time step is too short, it can't accurately record the downward trend of RUL. If the length of the selected time step is too long, the shortest sequence in the test can't be predicted. All these will lead to errors in the prediction.

Table 1. Description of the C-MAPSS dataset

Dataset	FD001	FD002	FD003	FD004
Engines' number of Train	100	260	100	249
Engines' number of Test	100	259	100	248
Operating conditions	1	6	1	6
Fault conditions	1	1	2	2
Min cycle of Test	31	21	38	19
Length of Train	20631	53759	24720	61249
Length of Test	13096	33991	16596	41214

Each sub dataset has 27 columns, which are Unit, Cycle, three operating conditions, and 21 sensors. Each sub dataset contains a training set and a test set, where the training set contains data for the whole life cycle that is from the highest RUL to failure (RUL is 0). However, the data composition of the test set starts to fail at any point in time.

3.2 Data Preprocessing

In this experiment, the data preprocessing is divided into the following steps:

For simple datasets (FD001 and FD003), first of all, the values of 3 operating conditions and 21 kinds of sensors were drawn and observed, which the third operating condition and the features such as 1th sensor, 5th sensor, 6th sensor, 10th sensor, 16th sensor, 18th sensor, 19th sensor do not change with cycle, so it can be seen that they are not effective in predicting RUL. In the correlation analysis of sensor values, values of 9th sensor and 14th sensor are highly correlated. Finally, the used features in the simple datasets are 2th sensor, 3th sensor, 4th sensor, 7th sensor, 8th sensor, 11th sensor, 12th sensor, 13th sensor, 14th sensor, 15th sensor, 20th sensor, 21th sensor, and 1th operating condition, 2th operating condition. In total of 15 features. For complex datasets, after the same graphic observation and correlation analysis, it's difficult to dig their internal relationship, so 21 sensors and 3 operating conditions were retained.

Original dataset (excluding Unit and Cycle) were processed through min-max normalize method. It takes FD001 data as an example to show the difference before and after normalization, as shown in Fig. 9.

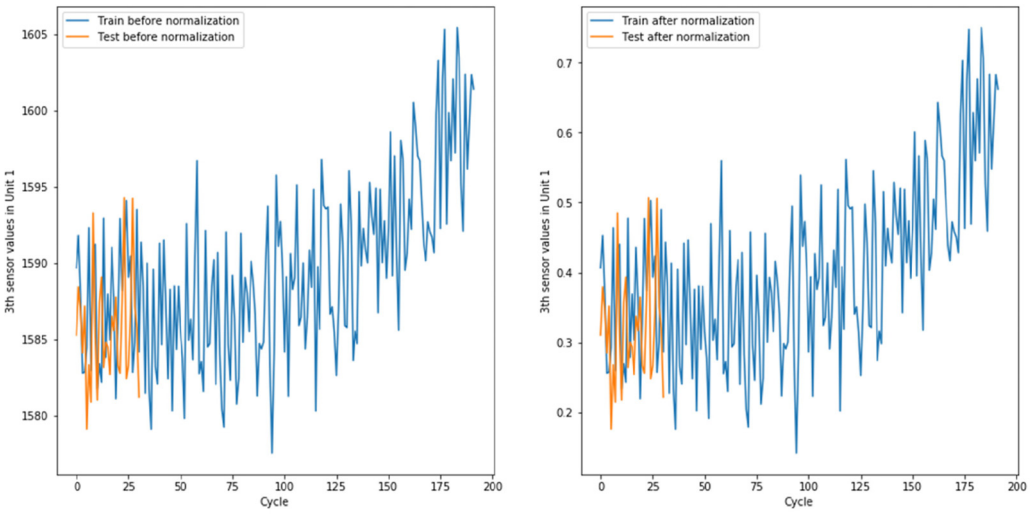


Fig. 9. The comparison of 3th sensor in Unit 1 before and after normalization

Using the LSTM-based autoencoder for dimensionality reduction. Our goal is to maximize the information of the original data while reducing the dimension. So the encoder in the autoencoder based on LSTM was used to reduce the dimensionality of the data. For FD001 and FD003, the above 15 selected features were input into the LSTM-based autoencoder. For FD002 and FD004, 24 features were input into the LSTM-based autoencoder. Obtained data from Encoder are the input of RUL prediction.

3.3 RUL Target Function

In this paper, the piecewise linear degradation model was used as our RUL target function. Compared with linear degradation, it's more suitable for RUL prediction.

The maximum value of RUL was set to 125, which is to prevent the model from over predicting and to improve the data fitting ability of the model, as shown in Fig. 10. Before the cycle = 81, the RUL is in a healthy state. After the cycle is greater than 81, RUL begins to decrease linearly and finally fails at cycle=206.

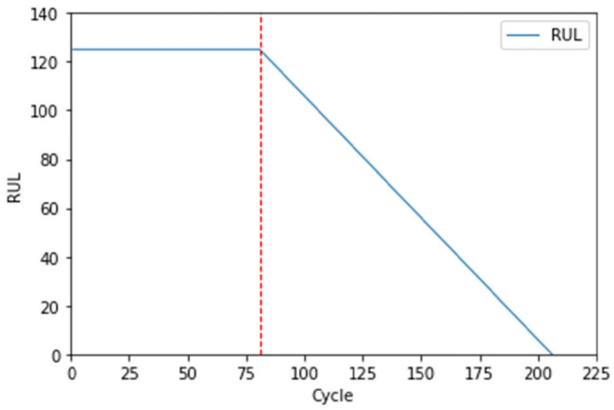


Fig. 10. Piece-wise linear RUL function

3.4 Evaluation of the Model

In this paper, the piecewise linear degradation model was used as our RUL target function. Compared with linear degradation, it's more suitable for RUL prediction.

In equation (19) to equation (21), RUL_{pred} represents predicted RUL, RUL_{true} represents true RUL, E_n represents prediction error, N represents training size, $RMSE$ represents root mean square error, S represents the result obtained by evaluation function Score.

$$E_n = RUL_{pred} - RUL_{true} \quad n \in [1, N]. \tag{19}$$

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} E_i^2}. \tag{20}$$

$$S = \begin{cases} \sum_{i=1}^{N} (e^{-\frac{E_i}{13}}) - 1 & E_i < 0 \\ \sum_{i=1}^{N} (e^{-\frac{E_i}{10}}) - 1 & E_i \geq 0 \end{cases}. \tag{21}$$

The images of the Score and RMSE functions are as follows.

There is no difference between the positive and negative errors of the prediction in RMSE function, as shown in Fig. 11. For example, the values of RMSE are the same when the predicted value is less than 20 and the predicted value is greater than 20. But in the Score function, if the prediction error is negative, then the penalty score value is relatively small, but if the prediction error is positive, then the greater the error, the greater the penalty Score value. For example, if the predicted value is less than the true value 50, the score is 48, and if the predicted value is greater than the true value 50, then the Score is 138.

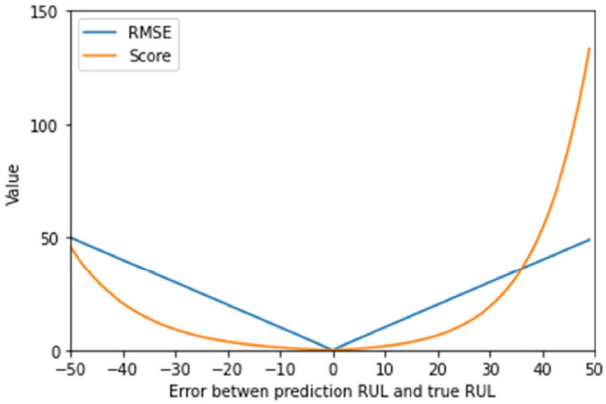


Fig. 11. The comparison of RMSE and Score

3.5 Experimental Setting

In this experiment, we used the Google Colab platform, where the GPU is NVIDIA Tesla P100 (16 GB memory), and the deep learning framework used in the experiment is Keras, which uses a Tensorflow backend.

Next, the parameters of the model will be introduced, as shown in Table 2 and Table 3.

Table 2. Autoencoder parameters

Layer	Unit	Activation Function
LSTM	64	Tanh
LSTM	32	Tanh
Dense	10	Relu
Dense	10	Relu
LSTM	32	Tanh
LSTM	64	Tanh

Table 3. TCN parameters

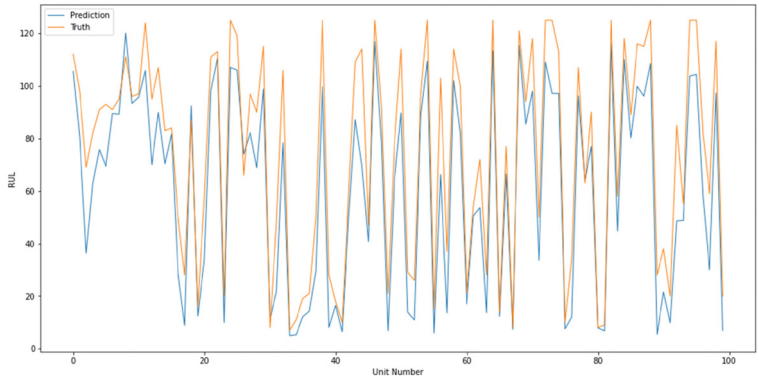
Dataset	Nb_stack	Filters	Kernel_size	Dilations	Batch_size	Epoch
FD001	2	32	2	[1, 2, 4, 8]	128	25
FD002	2	64	3	[1, 2, 4]	256	30
FD003	2	32	3	[1, 2, 4, 8]	128	25
FD004	2	64	3	[1, 2, 4]	256	30

4 Result Analysis

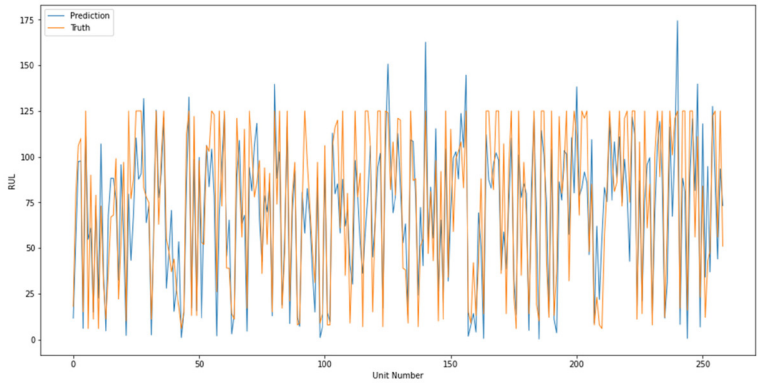
Comparison between predicted RUL and real RUL on the corresponding 100, 259, 100, 248 engines of the FD001 to FD004 test datasets are shown in Fig. 12.

The red curve represents predicted RUL and the blue curve represents real RUL. The following bar chart describes the error distribution between experimental prediction results and real RUL prediction results, as shown in Fig. 13.

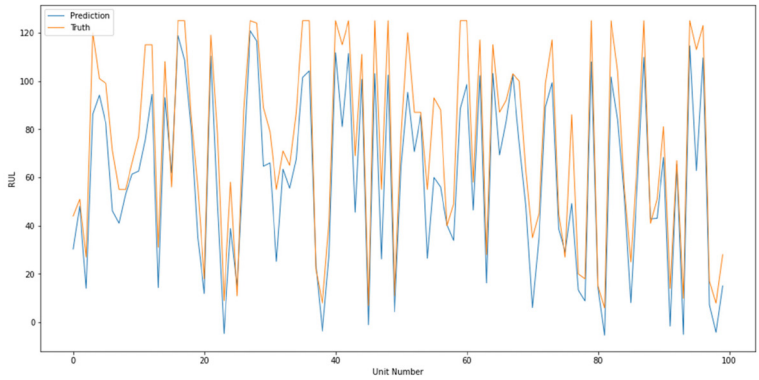
To sum up, on FD001, the number of predicted results lower than, equal to, and higher than the real results is 80, 2, 18. On FD002, it is 146, 9, 104. On FD003, it is 88, 4, 8. On FD004, it is 151, 6, 91. It can be seen that FD001 and FD003 have a smaller range of errors than FD002 and FD004, so they show lower RMSE and Score.



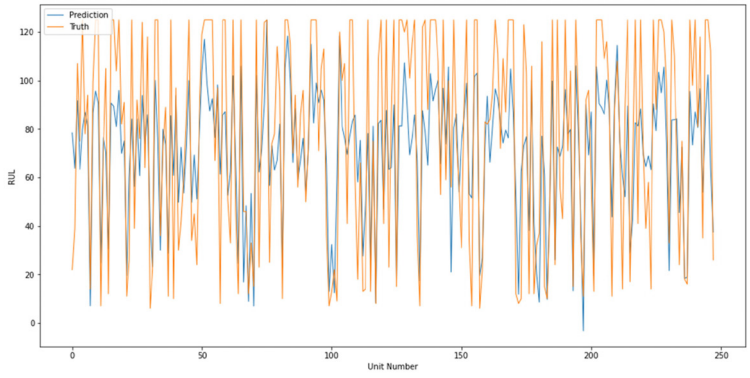
(a) FD001



(b) FD002



(c) FD003



(d) FD004

Fig 12. RUL prediction of FD001(a), FD002(b), FD003(c), FD004(d)

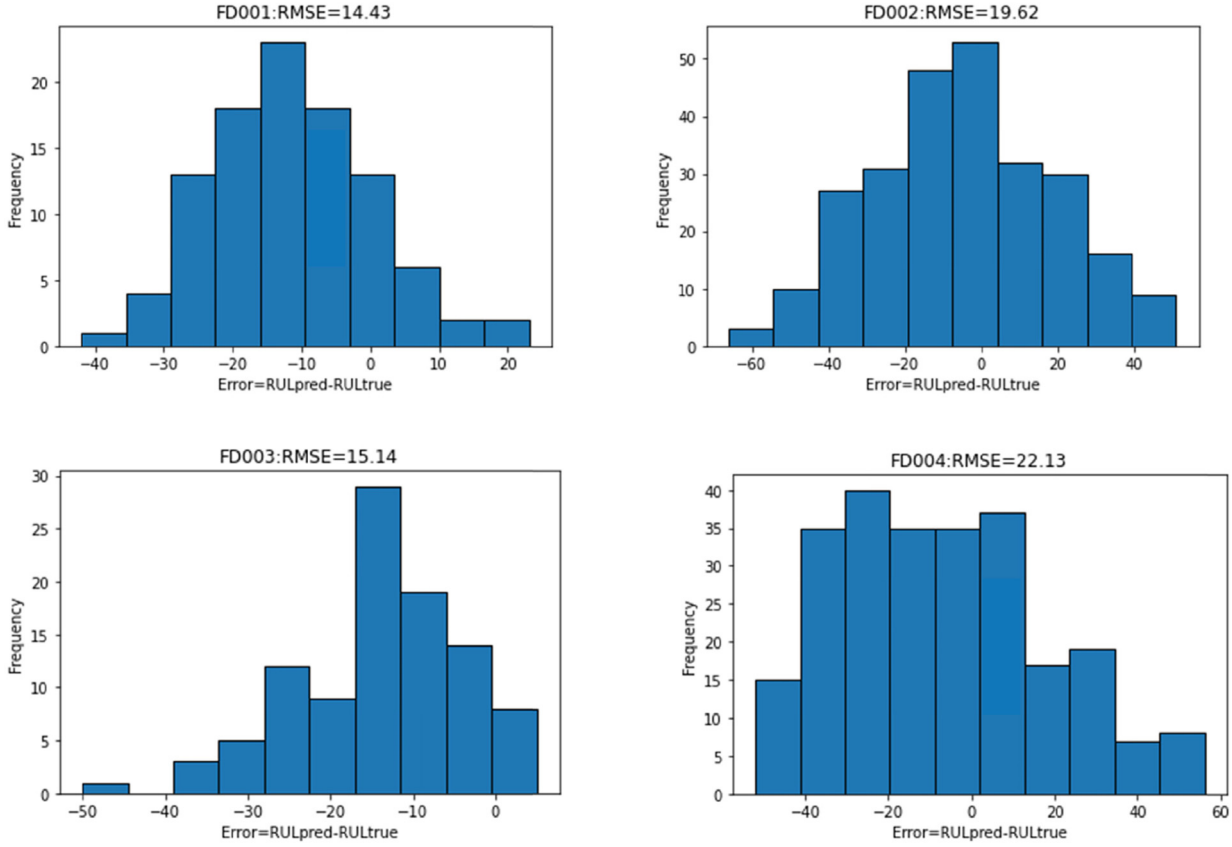


Fig. 13. RUL prediction error of FD001, FD002, FD003, FD004

FD001 and FD003: Excluding individual parts, the predicted curve covers the real curve, as shown in Fig. 10(a). If applied to practice, there will be the following situation: warning time of RUL will be delayed, which will lead to problems with the corresponding maintenance of most engines. Although the early warning is implemented in the case of high RUL, the same situation occurs in the case of low RUL, as shown in Fig. 10(c).

FD002 and FD004: FD002 and FD004 have improved a little, but due to the emergence of some singular values, the prediction is quite different from the real situation. On the whole, it performs better than simple datasets. Compared with FD002 and FD004, RMSE and Score of FD004 are higher than FD002. FD004 has fewer singular points and performs better than FD002 in the case of low RUL, as shown in Fig. 10(b) and Fig. 10(d). So predicted curve is closer to the actual needs, which also shows that the proposed method has better performance in a multi-operating environment and multi-fault mode. The ideal situation is to achieve early warning for RUL prediction, which is shown on the line chart as a real curve covering the prediction curve.

Below the RUL predictions of four randomly selected engines will be introduced in each sub datasets.

The blue lines represent real RUL and the yellow lines represent predicted RUL, Fig. 14 to Fig. 17. It shows the engine sequence number and the corresponding RMSE and Score values. It can be seen that because time step data were used to predict the next data. The following situation will appear. For example, in FD004, 18 data were used to predict the next data. So the first 18 cycles of the predicted RUL curve are not depicted, as shown in Fig. 17. Because the engine is randomly selected, the prediction effect may be good or bad. FD003 does not perform well compared to other sub datasets, partly because of random selection and partly because results may not perform well in predicting a single engine on FD003, as shown in Fig. 16. Finally, it can be found that with the increasing complexity of sub datasets, our performance is getting better and the degree of the fitting is getting higher. Final RUL prediction of an engine is becoming more and more accurate, which also shows that our proposed method has a better performance on complex datasets.

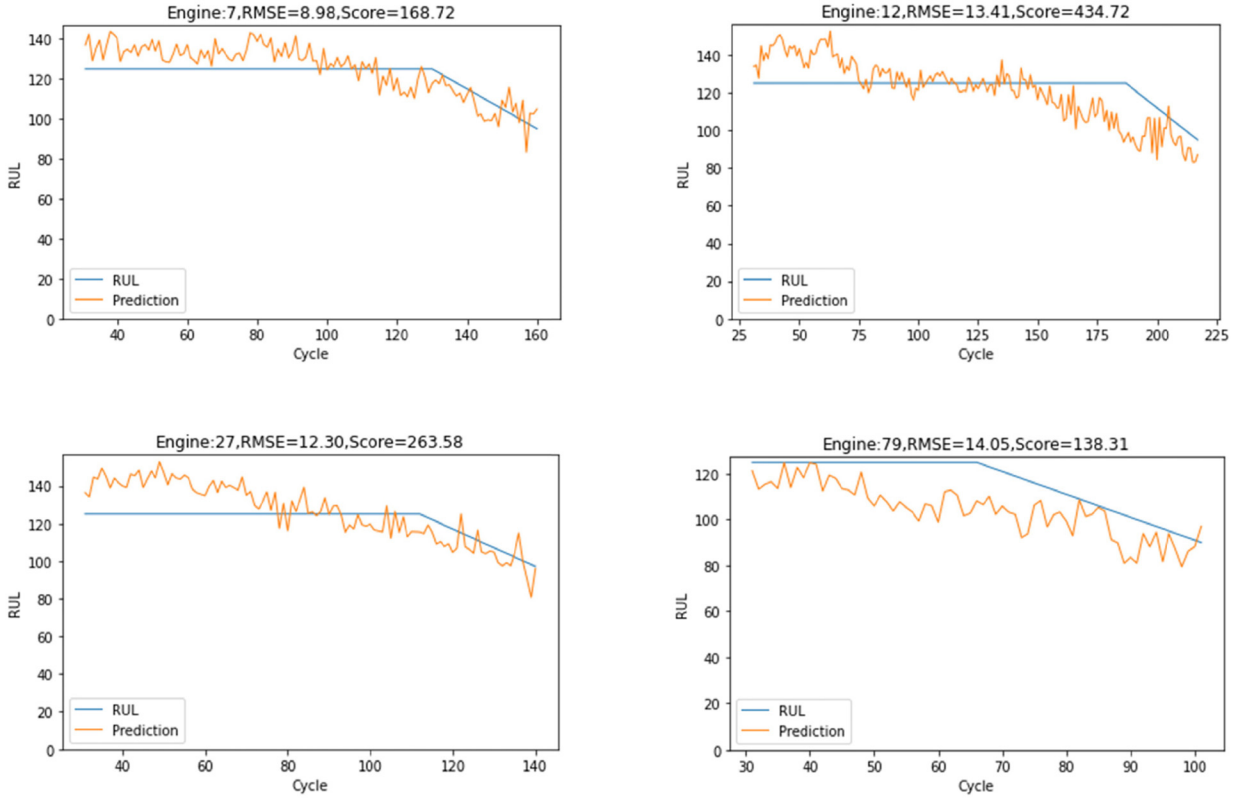


Fig. 14. Predicted RUL of 4 engines in FD001 dataset

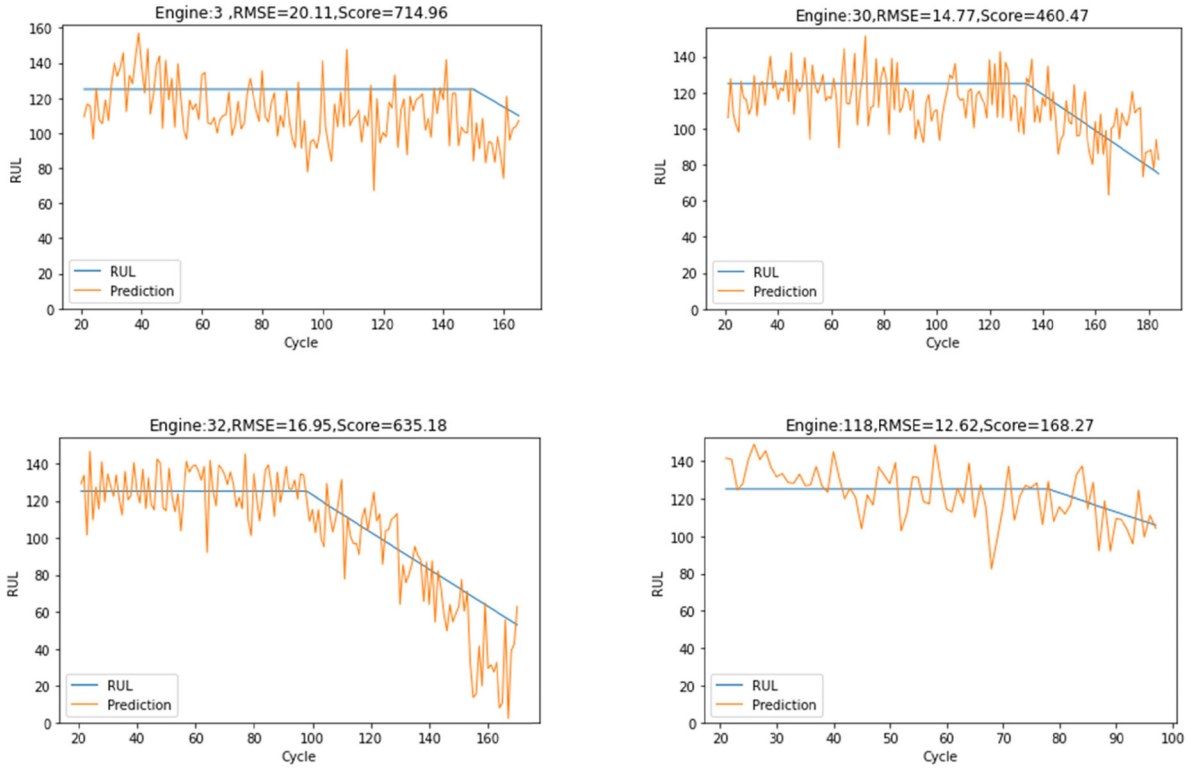


Fig. 15. Predicted RUL of 4 engines in FD002 dataset

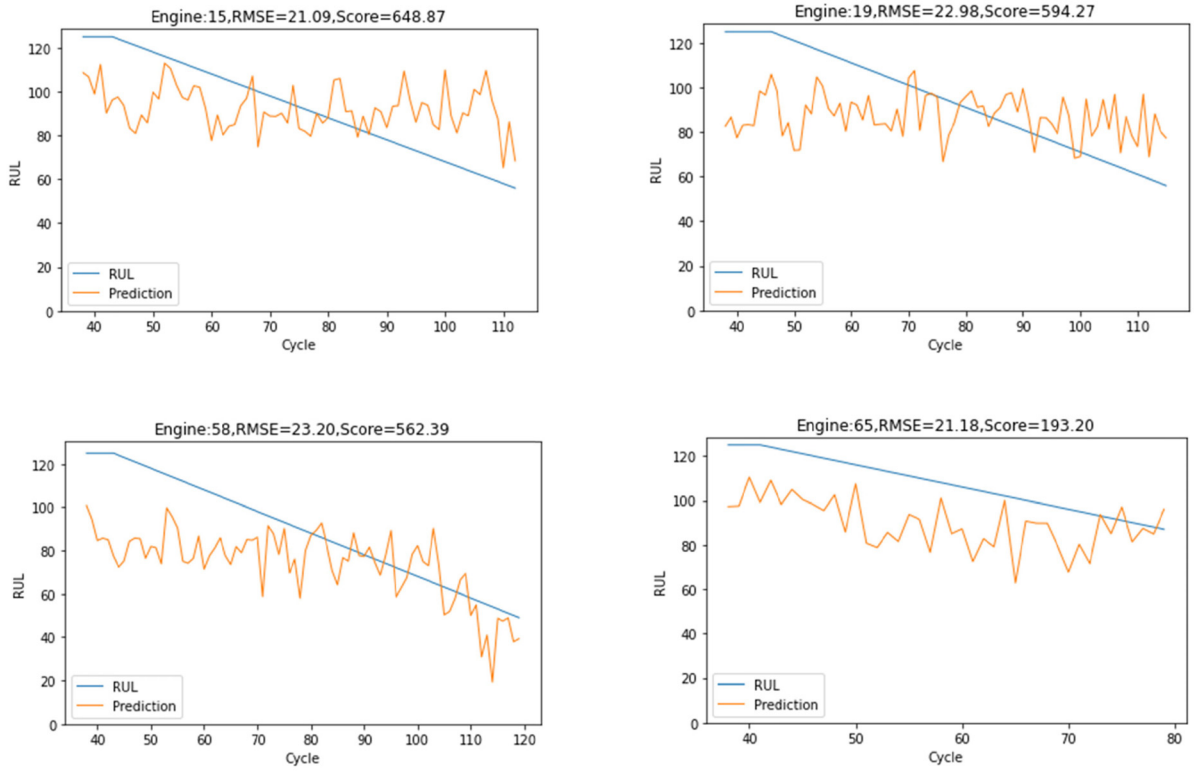


Fig. 16. Predicted RUL of 4 engines in FD003 dataset

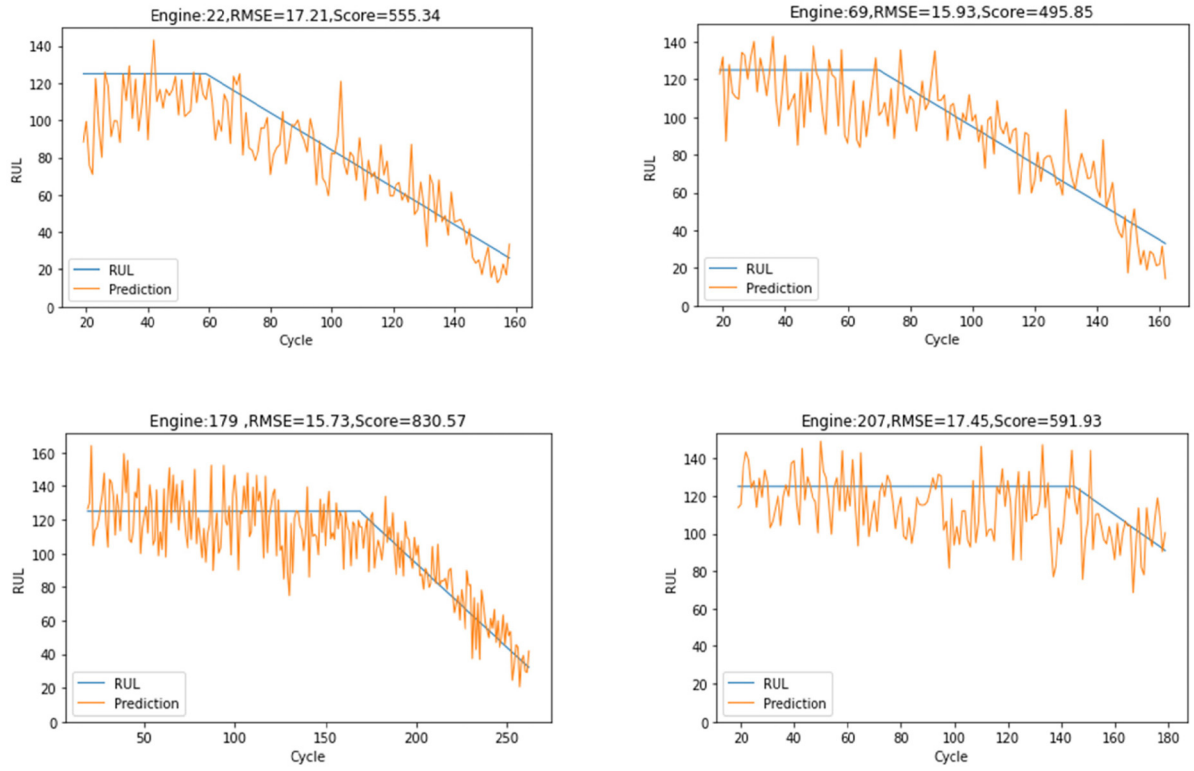


Fig. 17. Predicted RUL of 4 engines in FD004 dataset

It can be seen that the performance of the model in this paper is slightly worse on relatively simple datasets such as FD001 and FD003, as shown in Table 4, Fig. 18, and Fig. 19. RMSE and Score are slightly higher than the best results of the listed methods, but not much different from the best results. It

performs well on complex datasets, such as FD002 and FD004. RMSE and Score are the best among the listed methods, which shows that the model in this paper plays an important role in complex data sets and shows good performance. The following conclusion can be obtained, as shown in Table 5. In the preprocessing part of the data, LSTM-based autoencoder has unique advantages in feature extraction and dimensionality reduction, it performs better than selecting all features directly. In the prediction part of RUL, in order to verify the effect of TCN in RUL prediction, TCN was replaced by LSTM for RUL prediction, the results show that TCN performs better than LSTM. In summary, the method plays a significant role in each part of the entire RUL prediction process.

Table 4. Compare the prediction RMSE and Score with other methods

Dataset	FD001		FD002		FD003		FD004	
Matric	Score	RMSE	Score	RMSE	Score	RMSE	Score	RMSE
MLP [14]	1.80×10^{4}	37.56	7.80×10^{6}	80.03	1.74×10^{4}	37.39	5.62×10^{6}	77.37
SVR [14]	1.38×10^{3}	20.96	5.90×10^{5}	42.00	1.60×10^{3}	21.05	3.71×10^{5}	45.35
RVR [14]	1.50×10^{3}	23.80	1.74×10^{4}	31.30	1.43×10^{3}	22.37	2.65×10^{4}	34.34
DBN [13]	4.18×10^{3}	15.21	9.03×10^{3}	27.12	4.42×10^{2}	14.71	7.95×10^{3}	29.88
CNN [14]	1.29×10^{3}	18.45	1.36×10^{4}	30.29	1.60×10^{3}	19.82	5.55×10^{3}	29.16
DCNN [16]	**2.74×10^{2}**	**12.61**	1.04×10^{4}	22.36	**2.84×10^{2}**	**12.64**	1.25×10^{4}	23.31
RNN [16]	3.39×10^{2}	13.44	1.43×10^{4}	24.03	3.47×10^{2}	13.36	1.43×10^{4}	24.02
LSTM [17]	3.38×10^{2}	16.14	4.45×10^{3}	24.49	8.52×10^{2}	16.18	5.55×10^{3}	28.17
BiLSTM [18]	2.95×10^{2}	13.65	4.13×10^{3}	23.18	3.17×10^{2}	13.74	5.43×10^{3}	24.86
TCN	3.07×10^{2}	14.43	**3.15×10^{3}**	**19.62**	3.57×10^{2}	15.14	**3.68×10^{3}**	**22.13**

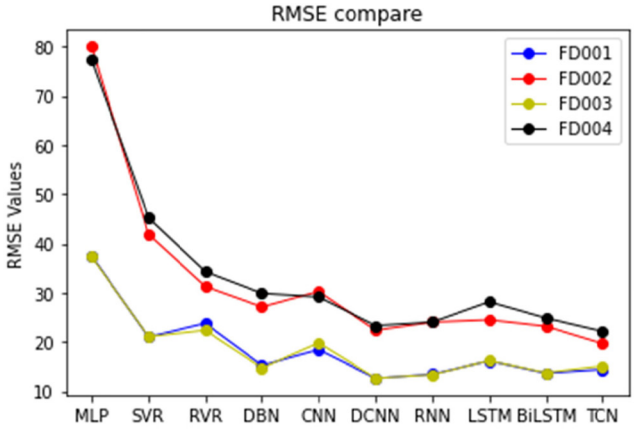


Fig. 18. Comparison the RMSE of different methods

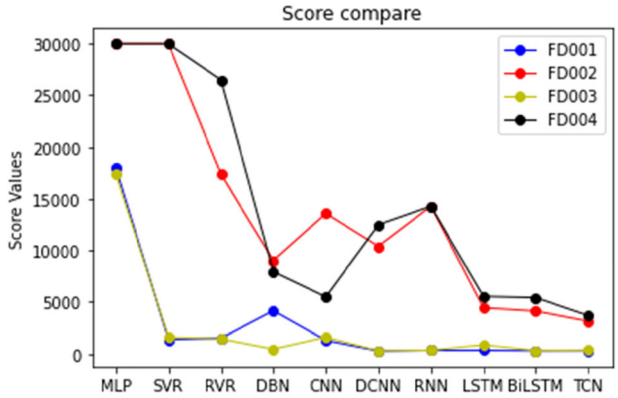


Fig. 19. Comparison the Score of different methods

Table 5. Performances analysis of each part for experiment

Dataset	FD001		FD002		FD003		FD004	
Matric	Score	RMSE	Score	RMSE	Score	RMSE	Score	RMSE
Exclude Encoder	4.07×10^2	15.43	3.14×10^3	19.14	7.36×10^2	16.21	4.50×10^3	21.54
Exclude TCN	4.50×10^2	16.46	4.80×10^3	22.15	9.12×10^2	18.18	5.73×10^3	23.45
Our method	3.07×10^2	14.43	3.15×10^3	19.62	3.57×10^2	15.14	3.68×10^3	22.13

5 Conclusion

This paper proposes a network structure that combines LSTM-based autoencoder and TCN. The original data is preprocessed by using the nonlinear dimensionality reduction of autoencoder and the advantage of extracting sequence information by LSTM. RUL prediction is carried out by using the advantages of causal convolution in TCN that does not disclose future information, dilation convolution in TCN has a large receptive field and residual structure to retain the original information of data. Compared with classical methods and methods that have performed well in the field at present, the effectiveness of the proposed network structure is proved. It shows excellent performance on the relatively complex datasets FD002 and FD004. At the same time, there are still some problems to be solved, such as dealing with the numerical points of anomaly prediction, improving the performance on simple datasets like FD001 and FD003 and optimizing the network structure, etc. Efforts will be made to solve the above problems in future work.

Acknowledgements

This work was supported by Natural Science Foundation of Inner Mongolia (Grant No. 2018MS06019).

References

[1] D. Wang, K.-L. Tsui, Q. Miao, Prognostics and Health Management: A Review of Vibration Based Bearing and Gear Health Indicators, IEEE Access 6(99)(2018) 665-676.

[2] R. Zhao, R.-Q. Yan, Z.-H. Chen, K.-Z. Mao, P. Wang, R.-X. Gao, Deep learning and its applications to machine health monitoring, Mechanical Systems and Signal Processing 115(2019) 213-237.

[3] Y.-G. Lei, N.-P. Li, L. Guo, N.-B. Li, T. Yan, J. Lin, Machinery health prognostics: A systematic review from data acquisition to RUL prediction, Mechanical Systems and Signal Processing 104(2018) 799-834.

[4] Y.-G. Lei, N.-P. Li, S. Gontarz, J. Lin, S. Radkowski, J. Dybala, A model-based method for remaining useful life prediction of machinery, IEEE Transactions on Reliability 65(3)(2016) 1314-1326.

[5] J.-Z. Sun, H.-F. Zuo, W.-B. Wang, M.-G. Pecht, Prognostics uncertainty reduction by fusing on-line monitoring data based on a state-space-based degradation model, Mechanical Systems and Signal Processing 45(2)(2014) 396-407.

[6] Y.-N. Qian, R.-Q. Yan, Remaining Useful Life Prediction of Rolling Bearings Using an Enhanced Particle Filter, IEEE Transactions on Instrumentation and Measurement 64(10)(2015) 2696-2707.

[7] Y.-X. Wen, J.-G. Wu, D. Das, T.-L.-B. Tseng, Degradation modeling and RUL prediction using Wiener process subject to multiple change points and unit heterogeneity, Reliability Engineering & System Safety 176(2018) 113-124.

[8] X.-S. Si, Z.-Q. Ren, A Data-Fusion Based Prognostic Method for Complex Degrading System, in: Proc. 2019 Prognostics and System Health Management Conference, 2019.

[9] X.-D. Xu, C.-Q. Yu, S.-J. Tang, X.-Y. Sun, X.-S. Si, L.-F. Wu, Remaining useful life prediction of lithium-ion batteries based on Wiener processes with considering the relaxation effect, Energies 12(9)(2019) 1685.

[10] J.-F. Jia, J.-Y. Liang, Y.-H. Shi, J. Wen, X.-Q. Pang, J.-C. Zeng, SOH and RUL Prediction of Lithium-Ion Batteries Based on Gaussian Process Regression with Indirect Health Indicators, Energies 13(2)(2020) 375.

[11] Y. Peng, Y.-D. Hou, Y.-C. Song, J.-Y. Pang, D.-T. Liu, Lithium-ion battery prognostics with hybrid Gaussian process function regression, Energies 11(6)(2018) 1420.

[12] S. Ahsan, T.-A. Lemma, M. Muhammad, Prognosis of gas turbine remaining useful life using particle filter approach, Materialwissenschaft und Werkstofftechnik 50(3)(2019) 336-345.

[13] C. Zhang, P. Lim, A.-K. Qin, K.-C. Tan, Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics, IEEE Transactions on Neural Networks and Learning 28(10)(2016) 2306-2318.

[14] G.-S. Babu, P. Zhao, X.-L. Li, Deep convolutional neural network based regression approach for estimation of remaining useful life, in: Proc. Appl International conference on database systems for advanced applications, 2016.

[15] X. Li, W. Zhang, Q. Ding, Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction, Reliability Engineering & System Safety 182(2019) 208-218.

[16] X. Li, Q. Ding, J.-Q. Sun, Remaining useful life estimation in prognostics using deep convolution neural networks, Reliability Engineering & System Safety 172(2018) 1-11.

[17] S. Zheng, K. Ristovski, A. Farahat, C. Gupta, Long short-term memory network for remaining useful life estimation, in: Proc. 2017 IEEE International Conference on Prognostics and Health Management, 2017.

[18] J.-J. Wang, G. Wen, S.-P. Yang, Y.-Q. Liu, Remaining useful life estimation in prognostics using deep bidirectional lstm neural network, in: Proc. 2018 Prognostics and System Health Management Conference, 2018.

[19] C.-G. Huang, H.-Z. Huang, Y.-F. Li, A Bidirectional LSTM Prognostics Method Under Multiple Operational Conditions, IEEE Transactions on Industrial Electronics 66(11)(2019) 8792-8802.

[20] H.-H. Miao, B. Li, C. Sun, J. Liu, Joint Learning of Degradation Assessment and RUL Prediction for Aeroengines via Dual-Task Deep LSTM Networks, IEEE Transactions on Industrial Informatics 15(9)(2019) 5023-5032.

[21] X.-Q. Pang, R. Huang, J. Wen, Y.-H. Shi, J.-F. Jia, J.-C. Zeng, A Lithium-ion Battery RUL Prediction Method Considering the Capacity Regeneration Phenomenon, Energies 12(12)(2019) 2247.

[22] J.-T. Qu, F. Liu, Y.-X. Ma, J.-M Fan, A neural-network-based method for RUL prediction and SOH monitoring of lithium-ion battery, IEEE Access 7(2019) 87178-87191.

[23] J.-L. Li, X.-Y. Li, D. He, A directed acyclic graph network combined with cnn and lstm for remaining useful life prediction, IEEE Access 7(2019) 75464-75475.

[24] L.-J. Zhang, Z.-Q. Mu, C.-Y. Sun, Remaining useful life prediction for lithium-ion batteries based on exponential model and particle filter, IEEE Access 6(2018) 17729-17740.

[25] S.-J. Bai, J.-Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, arXiv Preprint arXiv:1803.01271 (2018).

[26] K.-M. He, X.-Y. Zhang, S.-Q. Ren, J. Sun, Deep residual learning for image recognition, in: Proc. IEEE Conference on Computer Vision & Pattern Recognition, 2016.

[27] A. Saxena, K. Goebel, D. Simon, N. Eklund, Damage propagation modeling for aircraft engine run-to-failure simulation, in: Proc. 2008 International Conference on Prognostics and Health Management, 2008.