Remaining Useful Life Prediction of Turbofan Engine Based on Temporal Convolutional Networks Optimized by Genetic Algorithm

Zhengkun Chen 1 Baojia Chen 1,* Xueliang Chen 1
1Hubei Key Laboratory of Hydroelectric Machinery Design and Maintenance, College of Mechanical and Power Engineering, China Three Gorges University, Yichang, Hubei 443002, China
*Corresponding author's e-mail: cbjia@163.com

Abstract. It is tough to select the parameters of the deep neural network and enhance the accuracy in the field of remaining useful life (RUL) prediction. To address the problem, one RUL prediction model optimized by a genetic algorithm, based on temporal convolutional networks (TCN), was proposed. Firstly, forward-filling sliding sampling is used to add the samples’ time step. Then the genetic algorithm is used to search the hyperparameters of the residual block of TCN. Finally, the performance of the proposed method is verified by the C-MAPSS dataset. The results show that in the two evaluation metrics of root mean square error (RMSE) and score function (SF), the proposed GA-TCN reduces them by 8.2% ~ 27.56% and 28.24% ~ 79.35%, respectively, when compared with other studies. The RMSE and SF metrics of the proposed method are on average 17.10% and 54.10% lower than that of other methods in four sub-datasets of the turbofan engine.

1. Introduction
The turbofan engine, as the core component of the aerospace industry, has a complex structure and poor operating conditions. Accurate remaining useful life (RUL) prediction of turbofan engines will aid in ensuring the safe and reliable operation of spacecraft, as well as cutting costs on maintenance and enhancing economic benefits. Deep learning, new technology in the field of artificial intelligence in recent years, has excellent depth feature extraction ability and can well establish the mapping relationship between measured data and degradation trends [1]. Wang et al. [2] use sensor data directly as input data to a deep separable convolutional network (DSCN) to extract sensitive features and map them to the RUL of mechanical equipment. Hsu et al.[3] and Zheng et al. [4] both use the depth long-term memory network (LSTM) to estimate the RUL of a turbofan engine. However, the parameters of the same layer of an LSTM cannot be shared, and making LSTM parallel computing difficult. Furthermore, as the layers of network layers increases, so does the number of LSTM parameters, resulting in a very long training time of LSTM. Moreover, in existing RUL prediction methods, deep neural network parameter selection is difficult, and prediction model accuracy is difficult to improve.

Temporal convolutional networks (TCN) [5] is a kind of convolution neural network specially designed for time series processing. Compared with LSTM, it also has the ability of time memory, and its superior performance has been verified in the fields of machine translation, speech synthesis, time series prediction, and so on. Therefore, a TCN model based on a genetic algorithm (GeneticAlgorithm, GA) for parameter search is proposed in this paper, which is used to predict the RUL of the turbofan engine.
engine. On the one hand, this method makes full use of the advantages of convolution layer parameter sharing, parallel computing, and good memory ability in TCN structure; on the other hand, taking advantage of the good global convergence, high optimization efficiency, and a small number of parameters of GA, the parameter optimization of TCN can reduce the workload of manually adjusting parameters, obtain a stable TCN structure, improve the prediction accuracy of the model, and provide a reliable basis for turbofan engine maintenance strategy.

2. Proposed remaining useful life prediction method

2.1. Temporal convolutional networks

Temporal convolutional neural network (TCN) is a kind of convolution neural network, which is specially designed to deal with time series based on a residual network[6] signed by Bai [5] et al. in 2018. Its basic residual block structure is shown in figure 1 For one-dimensional sequence input \( x \in \mathbb{R}^n \) and convolution kernel \( f : \{0, \ldots, k - 1\} \rightarrow \mathbb{R} \), the dilated convolution operation \( F \) of sequence element \( s \) is defined as:

\[
F(s) = (x * f)(s) = \sum_{i=0}^{k-1} f(i) \cdot x_{s-d-i}
\]  

The basic structure of TCN is shown in figure 2, where each layer of TCN is a residual block (Res-Block), and the dilated rate of each series residual block increases as \( 2^{N-1} \) from shallow to deep, which makes it have a better ability to remember historical information. The biggest advantage of TCN over RNN lies in the weight sharing and local perception of the convolution layer. The sharing of weight parameters can effectively reduce the number of parameters that need to be trained in the network, and the local perception characteristics can harmoniously reflect the local structure information of the input data covered by the current convolution kernel. Therefore, TCN can not only learn the long historical time dependence of input time series but also perform parallel computing like CNN. Although RNN can theoretically capture an infinitely long history, TCN has been shown to be more suitable for areas that require long-term historical dependence[5].

2.2. RUL Prediction Framework of GA-TCN

The improved residual block is used to build the deep TCN structure system in this paper, and the genetic algorithm (GA) is used to optimize the key network parameters. The preprocessed time series data is first used as the input of TCN, then the output of the last layer TCN is directly connected to a full connection layer with one neuron number, and the output of the full connection layer is directly used as the RUL prediction result. Figure 3 depicts the remaining service life prediction process, which is divided into three modules: data processing, parameter Optimization of TCN, and RUL prediction.
2.2.1. Data processing

The original sensor data of the turbofan engine are different in dimension and order of magnitude. In order to improve the accuracy of RUL prediction, this paper uses the maximum and minimum normalization method to limit the range of original data to [0, 1].

In order to reduce the interference of the engine normal running state characteristics to the degraded state characteristics, references [2-4] and [7-9] all adopt the assumption of "Piecewise linear degradation" to deal with RUL labels. When the actual remaining life $Y$ is greater than a certain upper boundary value $T$, the engine life is considered to be in the normal running state, and its life is always constant $T$; when $Y$ is less than $T$, the engine life begins to degenerate and the engine life is the original actual value $Y$.

$$Y_{label} = \begin{cases} T, & Y \geq T \\ Y, & Y < T \end{cases}$$

Where: $Y_{label}$ is the segmented RUL labels. In addition, the selection of $T$ value is closely related to the degradation state of the engine. According to the research in reference [10], the $T$ value is usually between 120 and 130 cycles.

2.2.2. Parameter Optimization of TCN

In this subsection, GA uses floating-point coding [11], and its specific search parameters and ranges are shown in Table 1, corresponding to 1) activation functions $A$: ReLU, ELU, LeakyReLU, and SELU; 2) optimizer $O$; 3) the number of layers of the network, that is, the number of residual blocks $L$: to ensure that TCN can learn the time-dependent relationship of input data and facilitate training, it is stipulated that the network has a minimum of three layers and a maximum of seven layers, and its range corresponds to the range of the number of neurons in this layer; 4) Dropout rate $D$; 5) the parameter $\alpha$ of LeakyReLU: the value that will be further randomly generated when an individual network activation function in the initialized population uses LeakyReLU; 6) convolution kernel size $K$; 7) convolution way $P$: $\varphi_1$ means causal convolution, and $\varphi_2$ represents non-causal convolution; 8) batch size $B$; 9) initial learning rate $\lambda$.
Table 1. The parameters to be optimized for TCN and its range

| Variable | Scope of search | Variable | Scope of search | Variable | Scope of search |
|----------|-----------------|----------|-----------------|----------|-----------------|
| $A$      | [ReLU, ELU, LeakyReLU, SELU] | $L_4$    | [0, 4, 8, 16, 32, 64] | $\alpha$ | [0.05, 0.10, 0.15, 0.20, 0.30] |
|          | [Adam, RMSprop,] |          |                 |          |                 |
| $O$      | Adamax, Adagrad, Nadam, Adadelta | $L_5$    | [0, 4, 8, 16, 32, 64] | $K$      | [2, 3, 4, 5] |
| $L_1$    | [4, 8, 16, 32, 64, 128] | $L_6$    | [0, 4, 8, 16, 32, 64] | $P$      | [\varphi_1, \varphi_2] |
| $L_2$    | [4, 8, 16, 32, 64] | $L_7$    | [0, 4, 8, 16, 32, 64] | $B$      | [32, 64, 128, 256] |
| $L_3$    | [4, 8, 16, 32, 64] | $D$      | [0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50] | $\lambda$ | [0.01, 0.005, 0.01, 0.05, 0.1] |

The expansion rate of residual blocks in each layer of TCN is that $2^{N-1}$, $N$ is the layer value of residual blocks in each TCN network individual. In the process of training, when the prediction effect of the verification set which is randomly selected 10% of the training set, is not improved for 150 epochs, the learning rate of the individual network is reduced to 0.1 times of the original. The patience of early stop during training is 200 epochs. the maximum epochs of training are 300. the loss function is mean square error (MSE). The specific steps for optimizing TCN parameters are as follows:

Step 1 (Initialization population): initialize the TCN models with a number of 10, and keep the population constant in each generation after that;

Step 2 (Calculate the fitness of the population): the fitness of the population in GA is the MSE between the predicted value of RUL and the label value, and the smaller the value is, the better the fitness is.

Step 3 (The selection operator): the deterministic selection operator is used, that is, after the individuals of each generation are sorted according to the fitness from large to small, the first 50% is retained, and then 10% is selected from the remaining poor individuals, to prevent the algorithm from maturing prematurely.

Step 4 (The crossover operator): the uniform crossover operator is used, that is, the gene values on each gene position of the selected parents are exchanged with the same crossover probability to form two new individuals. The number of new individuals generated after crossover is determined by both the number of individuals retained after the operation of the previous selection operator and the number of initial populations.

Step 5 (The mutation operator): using the one basic mutation operator, randomly specify a parameter in table 1, mutate within the given range of parameter values, and set the mutation rate to 0.1.

Step 6 (Termination condition): when the algorithm completes 50 iterations, the operation is terminated.

The additional contrast method is the particle swarm optimization (PSO) algorithm [12]. According to the experience in reference [10], the learning factor of PSO is 1.2, the minimum inertia weight is 0.4, the maximum inertia weight is 0.9. other experimental conditions remain unchanged.

3. Test Results and Discussions

In this section, based on the framework of CUDA10.0 and Tensorflow 2.0 GPU, the algorithm is constructed by Python 3.7, and the calculation of the experiment is completed on a computer with configuration information of Intel Core™ i5-9300H CPU, Nvidia GeForce GTX 1660Ti GPU, and 16G RAM.

3.1. Dataset description

The verification data set of this paper adopts the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) data set which is widely used in RUL prediction [13]. The dataset consists of
four sub-datasets. These sub-datasets are simulated under different operating conditions and fault modes, and the monitoring data of 21 sensors and 3 operating conditions are collected to reflect the degradation process of turbofan engine from start to failure. The dataset description is shown in table 2, where FD001 and FD003 subdatasets have only one operating condition, so they are relatively simple and predictable, while FD002 and FD004 are difficult to predict because they have six operating conditions, especially FD004 and two failure modes. Therefore, the performance of the TCN algorithm can be well verified by using this data set.

Table 2. The introduction of C-MAPSS dataset

| Data set     | FD001 | FD002 | FD003 | FD004 |
|--------------|-------|-------|-------|-------|
| Training set     | 100   | 260   | 100   | 248   |
| Testing set      | 100   | 259   | 100   | 249   |
| Operating conditions | 1   | 6   | 1   | 6   |
| Failure mode     | 1   | 1   | 2   | 2   |

In the above 24-dimensional data, FD001 and FD003 are single operating conditions, so the data set with three fixed operating conditions will cause great interference to the extraction of TCN degenerate features. Because of the abnormal data of sensors 1, 5, 6, 10, 16, 18, 9, in the RUL prediction of FD001 and FD003, only the data of the remaining 14 sensors are used as the input of TCN; in the RUL prediction of FD002 and FD004, in addition to the 14 sensor data mentioned above, the data set by 3 operating conditions are also used. In addition, in order to facilitate the comparison, this paper carries out experiments on the upper boundary values of RUL tags, respectively.

3.2. Evaluation indicators

In this paper, the prediction performance of TCN is quantitatively evaluated by scoring function (Scoring Functions, SF) [13], root mean square error (Root Mean Square Error, RMSE) and accuracy (Accuracy) [7].

\[
E_{error}^{ij} = C_{prevRUL}^{ij} - C_{actRUL}^{ij} \quad (3)
\]

\[
RMSE = \sqrt{\frac{1}{Q} \sum_{i=1}^{Q} E_{error}^{ij}} \quad (4)
\]

\[
SF = \begin{cases} 
\sum_{j=1}^{Q} \left( e^{-\frac{E_{error}^{ij}}{13}} - 1 \right), & E_{error}^{ij} < 0 \\
\sum_{j=1}^{Q} \left( e^{-\frac{E_{error}^{ij}}{19}} - 1 \right), & E_{error}^{ij} \geq 0 
\end{cases} \quad (5)
\]

Where, \( E_{error}^{ij} \) is the error between the predicted RUL value of the engine and the actual value of RUL; \( Q \) is the number of test samples.

3.3. Analysis of experimental results

The dataset used in the TCN parameter search is the FD001 subdataset, with the upper boundary value \( T =125 \) and the time step is 36. Therefore, the shape of TCN input data is (None,36,14).The convergence effect of GA and PSO is shown in figure 4. From the figure, we can see that the convergence value of the best individual of GA is about 135,while the convergence value of the best individual of PSO algorithm is about 150 GA. The optimization effect of GA is 11% higher than that of PSO algorithm. Finally, the parameters of the best individual are found in table 3, and then use
these parameters to construct the TCN model structure of this paper, while continuing to use the same learning rate attenuation and early stop strategy mentioned above, only increase the maximum number of training to 1000.

Table 3. The optimal parameters of TCN searched by genetic algorithm

| Variable | Optimal solution | Variable | Optimal solution | Variable | Optimal solution |
|----------|------------------|----------|------------------|----------|------------------|
| A        | LeakyReLU        | L₄       | 16               | α        | 0.10             |
| O        | Adam             | L₅       | 0                | K        | 3                |
| L₁       | 64               | L₆       | 8                | P        | φ₁               |
| L₂       | 32               | L₇       | 8                | B        | 128              |
| L₃       | 16               | D        | 0.25             | λ        | 0.01             |

Figure 4. Comparison of convergence effect of TCN parameters optimized by GA and PSO

To prove the effectiveness of the method proposed in this paper, we have carried out RUL prediction experiments in FD001~FD004 and compared the results with some recent literature results. Recent studies are summarized in table 4.

Table 4. Recent approaches proposed for RUL predictions on the C-MAPSS dataset

| Author & Refs. | year | Approach          |
|----------------|------|-------------------|
| Wang et al.[2] | 2019 | DSCN              |
| Hsu et al.[3]  | 2018 | LSTM              |
| Zheng et al. [4]| 2017 | LSTM + FNN        |
| Ellefsen et al.[7]| 2019 | RBM + LSTM        |
| Li et al. [8]  | 2018 | CNN + FNN         |
| Liu et al.[9]  | 2020 | AGCNN             |

Table 5. Evaluation and comparison of RUL Predicted results

| data set | Evaluation indicators | Proposed GA-TCN | DSCN [2] | LSTM [3] | LSTM+ FNN[4] | RBM+ LSTM[7] | CNN+ FNN[8] | AGCNN[9] |
|----------|-----------------------|-----------------|----------|----------|--------------|--------------|------------|----------|
| FD001    | RMSE SF               | 11.82 205.37    | 10.95 260.07 | 16.74 388.68 | 16.14 338 12.56 231 | 12.56 274 | 12.61 225 12.42 |
| FD002    | RMSE SF               | 16.77 1195.53   | 10.62 4367.56 | 29.43 10654 | 24.49 4450 | 22.73 3366 | 22.36 10412 | 19.43 |
| FD003    | RMSE SF               | 12.70 246.55    | 19.62 822.19 | 18.07 852 16.18 251 | 12.10 284 | 12.64 1492 | 13.39 |
| FD004    | RMSE SF               | 19.62 1653.12   | 22.64 5168.45 | 28.40 6370.60 | 28.17 5550 | 22.66 2840 | 23.31 12466 | 21.5 3392 |
The comparison of the final RUL prediction results is shown in table 5. It can be seen from the table that five of the eight evaluation results are the best. The RMSE and SF indexes of this method on the FD001 data set are on average 12.90% and 28.24% lower than those of the six methods, respectively; the RMSE and SF indexes on the FD002 data set are on average 27.56% and 79.35% lower than the RMSE and SF of the six methods, respectively; and the RMSE and SF indexes on the FD003 data set are on average 8.20% and 36.54% lower than the RMSE and SF of the six methods, respectively. The RMSE and SF indicators on the FD004 dataset are 19.74% and 72.28% lower on average than the RMSE and SF of the six methods, respectively. Therefore, these comparison results strongly prove the effectiveness of this method.

4. Conclusion
A genetic algorithm is used to optimize a TCN based RUL prediction model proposed in this paper. Utilizing the benefits of genetic algorithm's good global convergence and high optimization efficiency, the optimization of TCN parameters can make the TCN model adaptively select model parameters based on specific prediction problems, reduce the workload of manual debugging parameters, improve the model's prediction accuracy, and realize the end-to-end RUL prediction model. By testing the C-MAPSS data set and comparing it with the existing prediction methods, the following conclusions are drawn: 1) GA can quickly find the best parameters of TCN and effectively improve the prediction accuracy of TCN. 2) the RUL prediction and evaluation indexes RMSE and SF of the GA-TCN model on the four sub-test sets are on average reduced by 17.10% and 54.10% respectively compared with six methods, which proves the superiority of this method and can provide reliable guidance for the safe maintenance of the equipment.

Acknowledgments
This program is jointly supported by the projects of National Nature Science Foundation of China (51975324, 52075292), Open Foundation of State Key Laboratory of Mechanical Transmission, Chongqing University, China (SKLMT-MSFKKT-202020), Open Foundation of Hubei Key Laboratory of Hydroelectric Machinery Design & Maintenance (No.2021KJX02 & No.2020KJX02).

References
[1] Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. 2018. Machinery health prognostics: A systematic review from data acquisition to RUL prediction. Mechanical systems and signal processing, 104, 799-834.
[2] Wang, B., Lei, Y., Li, N., & Yan, T. 2019. Deep separable convolutional network for remaining useful life prediction of machinery. Mechanical Systems and Signal Processing, 134, 106330.
[3] Hsu, C. S., & Jiang, J. R. 2018. Remaining useful life estimation using long short-term memory deep learning. In 2018 ieee international conference on applied system invention (icas) (pp. 58-61). IEEE.
[4] Zheng, S., Ristovski, K., Farahat, A., & Gupta, C. 2017. Long short-term memory network for remaining useful life estimation. In 2017 IEEE international conference on prognostics and health management (ICPHM) (pp. 88-95). IEEE.
[5] Bai, S., Kolter, J. Z., & Koltun, V. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271.
[6] He, K., Zhang, X., Ren, S., & Sun, J. 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[7] Ellefsen, A. L., Bjørlykhaug, E., Æsøy, V., Ushakov, S., & Zhang, H. 2019. Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. Reliability Engineering & System Safety, 183, 240-251.
[8] Li, X., Ding, Q., & Sun, J. Q. 2018. Remaining useful life estimation in prognostics using deep convolution neural networks. Reliability Engineering & System Safety, 172, 1-11.
[9] Liu, H., Liu, Z., Jia, W., & Lin, X. 2020. Remaining useful life prediction using a novel feature-attention-based end-to-end approach. *IEEE Transactions on Industrial Informatics*, **17**(2), 1197-1207.

[10] Heimes, F. O. 2008. Recurrent neural networks for remaining useful life estimation. In *2008 international conference on prognostics and health management* (pp. 1-6). IEEE.

[11] Wang, Z., Li, F., Shi, G., Xie, X., & Wang, F. 2020. Network pruning using sparse learning and genetic algorithm. *Neurocomputing*, **404**, 247-256.

[12] Dong, W., Kang, L., & Zhang, W. 2017. Opposition-based particle swarm optimization with adaptive mutation strategy. *Soft Computing*, **21**(17), 5081-5090.

[13] Saxena, A., Goebel, K., Simon, D., & Eklund, N. 2008, October. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 international conference on prognostics and health management* (pp. 1-9). IEEE.