Combining Autoencoder with Similarity Measurement for Remaining Useful Life Estimation of Aircraft Engines

Mengni Wang
Shanghai Jiao Tong University
Shanghai, China
moniwong@sjtu.edu.cn

Yuanxiang Li*
Shanghai Jiao Tong University
Shanghai, China
yuanxli@sjtu.edu.cn

Honghua Zhao
Eastern Airlines Technic Co., Ltd
Shanghai, China
hhzhao@ceair.com

Yuxuan Zhang
Shanghai Jiao Tong University
Shanghai, China
yuxuanzhang@sjtu.edu.cn

Abstract—Remaining useful life (RUL) estimation is very important for the maintenance of aircraft engines. We can evaluate the current condition of an aircraft engine and predict its RUL by constructing its degradation curve. However, the degradation curve is often difficult to obtain due to the unobserved degradation patterns. Currently many researchers estimated RUL by setting a fixed RUL target function or making assumptions about how an engine degrades. But in the real world, degradation models of aircraft engines are generally individualized. To obtain personalized degradation curves and predict RUL accurately, we propose a method based on autoencoder and similarity measurement. First, an autoencoder trained with normal data is adopted to extract degradation curves of aircraft engines and build a degradation model template library. Then, we measure each test object with all template curves to get similarities and corresponding RULs based on a sliding window and complexity-invariant distance. At last, the estimated RUL can be obtained by calculating the weighted average of highly relevant corresponding RULs. We conduct the proposed method on the aircraft engine dataset provided by NASA. The experimental results demonstrate that our method can utilize the information of multi-sensor data to generate personalized degradation curves effectively and estimate RUL more accurately.

Keywords—Aircraft Engine, Remaining Useful Life, Autoencoder, Degradation Curve, Similarity Measurement

1 Introduction

Remaining useful life (RUL) estimation is an important part of Prognostics and Health Management. With the fast development of industry and manufacturing, sensors are widely used to collect running data from complex systems, which are very beneficial to evaluate their states and estimate their RULs. This promotes the popularity of condition-based maintenance rather than time-based maintenance, for it can not only ensure the safety of equipment but also lead to significant financial savings. Such goal can be reached by predicting the RUL of target object based on establishing its degradation model with collected multi-sensor data.

RUL estimation methods mainly consist of model-based methods, data-driven methods and experience-based methods [1]. Model-based methods evaluate the system state by building physical model. A physics-based model for bearing prognostics is proposed in [2], by which we can get spall growth trajectory and calculate time to failure according to operating conditions, and then

reduce prediction uncertainty based on self-adjusting. Data-driven methods usually learn degradation model by running data to estimate RUL, where Hidden Markov Model [3], Gaussian process regression [4] and neural networks [5] are frequently used. Experience-based approaches predict the system RUL based on historical failure cases, building hidden relationship among current system states, current lives and recorded failure models. In [6], RUL is obtained by evaluating the instance similarity, which is related to the usage and maintenance history. Since the difficulty in modeling assumptions and capturing features that directly reflect system change, model-based methods are limited, while due to the available of large amount of running data, data-driven methods and experience-based methods have been greatly developed.

In the past few years, artificial neural network (ANN) has become noticeable. Many researchers tend to estimate RUL based on ANN such as convolutional neural network (CNN), long short-term memory (LSTM) [7] and recurrent neural network (RNN). Most of them predict RUL by setting a fixed RUL target function or making assumptions about how an engine degrades. The piece-wise model is commonly employed [8, 9, 10], in which a set of instances are assumed to begin with the same RUL and then start linear degradation after a certain cycle. In [8], a piece-wise RUL target function is adopted to obtain assumed trajectory of RUL, and then LSTM model is obtained by the training data to find optimal parameters. A RUL estimation method based on CNN is applied in [10]. This method first sets a piece-wise linear RUL target function to get target RUL, then CNN is applied on sensor data split by a sliding window.

However, these fixed RUL target functions or degradation models are always too restricted and unreasonable for complex systems. The reasons are as follows: 1) The lifetime of each engine is different. 2) Each engine begins to work with unknown wear which is basically different from each other. 3) With diverse operating conditions and faults, degradation curves do not follow a fixed shape. 4) The accuracy of RUL estimation is greatly affected by the selection of degradation point.

Thus, this paper uses the autoencoder to obtain individualized degradation mode of each engine. We first transform reconstruction errors to state variations of the complex system by autoencoder, then filter them to get degradation curves. Furthermore, with many degradation curves of run-to-failure systems, we can set up a degradation model template library to match degradation models of test engines. To get RUL, we compute the similarities and corresponding RULs between test engine and all instances in the library by a sliding window, based on the main idea of case-based learning [11]. Our experiment is conducted on public C-MAPSS dataset provided by NASA and the result shows that our method can improve the accuracy of RUL estimation in comparison with other fixed-RUL methods.

The contribution of this paper can be summarized as follows:

•Extract the personalized degradation curve by autoencoder with recorded multi-sensor data.

•Analyze the characteristics of degradation curve and propose an effective framework to calculate RUL based on similarity measurement without any degradation assumptions.

The rest of the paper is organized as follows. Section 2 introduces the related work, including time series anomaly detection based on autoencoder and complexity-invariant distance [12]. Section 3 presents our method, describing how to extract degradation curve and measure similarity between test curve and model curves. Section 4 shows the experimental results on C-MAPSS dataset. The last section summaries the paper and discusses the future work.

2 Related Work

2.1 Time Series Anomaly Detection

Autoencoder is proposed for the process of high dimensional data [13], which is widely used in the field of data compression, representative learning and signal denoising. The main idea of autoencoder is to reconstruct original data by some sparse high-level features or in other words to learn an equation to make output same as input. The structure of autoencoder is shown in Fig.1. It consists of an encoder and a decoder. The encoder maps original high-dimensional data to a hidden low-dimensional vector while the decoder aims to convert this vector to a same series as input. For N -dimensional data $x = \{x^i | i = 1,2, \dots N\}$, the reconstruction error can be obtained by (1):

$$e = \sum_{i=1}^{N} \left\| x^i - g\left(f\left(x^i\right)\right) \right\| \quad (1)$$

where $g_\theta()$ is the decoder function, $f_\theta()$ is the encoder function.

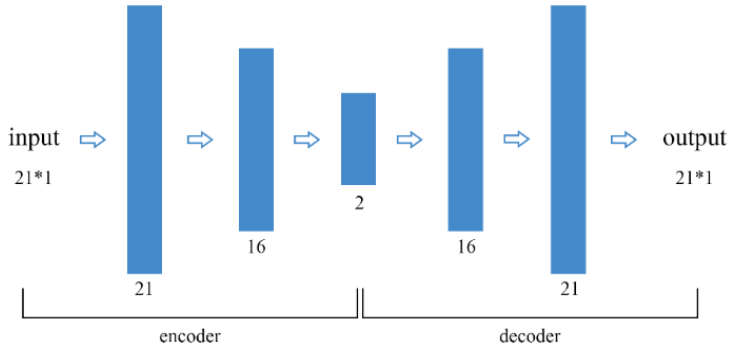

Fig. 1. The structure of autoencoder

Autoencoder based approaches for time series anomaly detection have been proposed in [14,15]. Due to the unrecorded factors or variables, it is difficult to detect anomalies by mathematical models or prediction models. So researchers adopt autoencoder-based architecture to reconstruct normal data behavior and denote anomaly as large reconstruction error, for the autoencoder only knows the representation of normal data.

2.2 Complexity-invariant Distance

For case-based learning, the method of similarity measurement or in other words the way to calculate distance between cases is particularly important. Common distance calculation methods include Euclidean Distance and Dynamic Time Warping (DTW), but they are not very applicable to current situation. Euclidean Distance is often effective but it tends to assign a complex object to a simpler class [12]. DTW is usually used to compare two sequences of different lengths, in which one series may be warped by stretching or shrinking its time axis [16]. To evaluate the similarity between target curve and model curve, here complexity-invariant distance (CID) is adopted. CID improves the accuracy of classification and clustering by taking complexity differences between two sequences into consideration [12]. For physical intuition, the complexity of a sequence can be defined by its length, since complex series is always longer than a simpler series after stretched. On the basis of Euclidean Distance, CID just adds a correction factor compared with it. Assume that Q and C are two series with n steps, then main calculation formulas of CID can be listed as follows:

$$\mathrm{CID}(\mathrm{Q},\mathrm{C}) = \mathrm{ED}(\mathrm{Q},\mathrm{C}) \times CF(Q,C) \tag{2}$$

$$CF(Q,C) = \frac{\max(CE(Q),\ CE(C))}{\min(CE(Q),\ CE(C))} \tag{3}$$

$$CE(Q) = \sqrt{\sum_{i=1}^{n-1}(q_i - q_{i+1})^2} \tag{4}$$

where $\mathrm{CID}(\mathrm{Q},\mathrm{C})$ calculates the CID between Q and C, complexity correction factor of Q and C is defined as $CF(Q,C)$, $CE(Q)$ represents the complexity estimation of time series Q.

3 Method

3.1 Degradation Curve Extraction

Since autoencoder can fuse high-dimensional data and reconstruct them effectively by representative learning, we utilize an autoencoder trained with normal data to extract degradation curve, so that it can reconstruct normal cycles with small errors while high error is generated at the occurrence of anomalous subsequence. Because each engine start running normally and some faults occur at an unknown point during operating, we take the sensor data collected from several initial cycles as normal data. Assume that matrix $X^i = \{x_1^i, x_2^i, \dots, x_n^i\}$ represents all recorded multi-sensor data of engine i with n running cycles. Then autoencoder is trained with initial data $x_1^i \sim x_m^i$ to minimize the loss, and reconstruct X^i to get reconstruction error $\mathrm{e}^i = \{e_1^i, e_2^i, \dots e_n^i\}$, which can be considered as the deviation from normal state. Since test engines are not run-to-failure, normalization to 0-1 on e^i will change the relative shape between test curve and model curve. So here we just filter it to get health indicator $HI^i = \{h_1^i, h_2^i, \dots, h_n^i\}$ as well as degradation curve, where 0 represents normal and

increase in amplitude means degradation in state. After extracting all run-to-failure degradation curves of engines in training subsets, a template library will be established.

3.2 Similarity Measurement

According to the main idea of case-based learning, we need to calculate the CID between two series to get the similarity of them. A sliding window of the same length as test series is used to calculate CID between test series and all segments of each degradation curve, with a step length of 1. Then the similarity $\mathrm{sim}(Q^i, C^j)$ between test engine i and model engine j can be obtained by (5).

$$\mathrm{sim}(Q^i, C^j) = max\left\{\frac{1}{CID(Q^i, C_1^j)}, \frac{1}{CID(Q^i, C_2^j)}, \ldots, \frac{1}{CID(Q^i, C_n^j)}\right\} \quad (5)$$

where $C_k^j, k = 1, 2, \ldots, n$ are segments of the same length as Q^i, split from model curve C^j.

3.3 The Proposed Architecture

The flowchart of our method is shown in Fig. 2. First, we preprocess the original multi-sensor data to satisfy the latter experiment requirements. Second, we use an autoencoder trained with normal data to obtain the state variations of engines as degradation curves which are same as their HIs, and build a degradation model template library. Then we measure each test object with all degradation curves in template library to get similarities and corresponding RULs based on CID. At last we can get the estimated RUL by calculating the weighted average of highly relevant corresponding RULs.

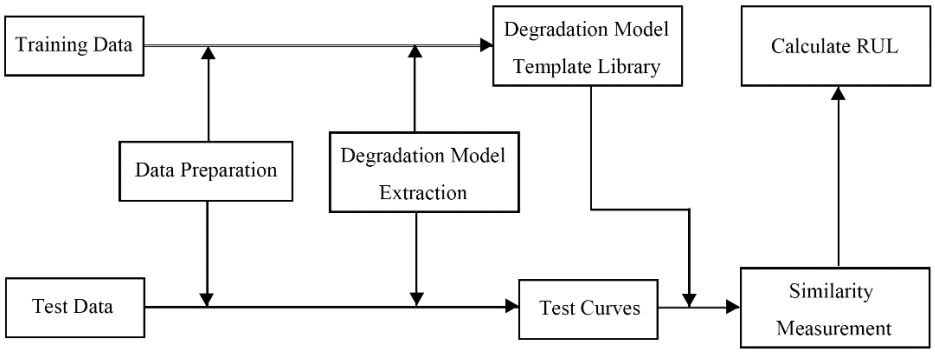

Fig. 2. The flowchart of our method.

4 Experiments

4.1 C-MAPSS Turbofan Engine Dataset and Performance Evaluation

Commercial Modular Aero-Propulsion System Simulation is a tool for simulating a realistic turbofan engine and Saxena et al. [17] modeled a series of turbofan engines as well as recorded their run-to-failure data as C-MAPSS dataset. As shown in Table I, the dataset contains four subsets covering different operating conditions and fault modes, and then they are further divided into training and test subsets. Each engine operates normally at the beginning with unknown different initial wear, and then develops faults at some different points

during running. The faults keep growing until system fails. Engines in training subset fail at the last cycle while in test subset recording data end at some cycle which is prior to final failure. Each subset includes 26 columns, containing engine id, running time (in cycles), 3-dimensional operating condition settings and 21-dimensional sensor data. The number of remaining cycles to failure of each test engine is given in another txt file.

Table I. C-MAPSS DATASET

Sub dataset	FD001	FD002	FD003	FD004
Number of training engines	100	260	100	249
Number of test engines	100	259	100	248
Operating conditions	1	6	1	6
Fault modes	1	1	2	2

In some cases, predicting failure early is better than late. Late prediction may lead to accidents because condition-based maintenance is too late to perform, while early failure warning will not pose life threatening. So an unbalanced evaluation indicator score is employed to penalize late prediction. Its definition is denoted as (6).

$$\mathrm{s} = \begin{cases} \sum_{i=1}^{n} e^{-\left(\frac{d^i}{a_1}\right)} - 1 \ \ for\ d^i < 0 \\ \sum_{i=1}^{n} e^{-\left(\frac{d^i}{a_2}\right)} - 1 \ \ for\ d^i \geq 0 \end{cases} \tag{6}$$

where s is the computed score, n is the number of test engines, $d^i = RUL^i_{est} - RUL^i_{true}$ (Estimated RUL^i – True RUL^i) is the estimation error of engine i, $a_1 = 13$ and $a_2 = 10$.

We can know from (6) that if there are some instances whose d is large enough, then the score will grow exponentially. Another evaluation index is Root Mean Square Error (RMSE), which is given by (7). Compared with score it penalizes estimation errors equally.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n} d_i^2} \tag{7}$$

where $d^i = RUL^i_{est} - RUL^i_{true}$ (Estimated RUL^i – True RUL^i) is the estimation error of engine i, n is the number of test engines.

4.2 Data Normalization

The raw data are collected from many different sensors with various value scales, and it is not beneficial to use these data to make estimation directly. To convert the raw data to desirable form, data preparation is necessary. In order to compare our results with previous studies, here we choose the same z-score normalization:

$$\mathrm{z} = \frac{x-\mu}{\sigma} \tag{8}$$

where μ is the mean and σ is the corresponding standard deviation.

However, engines in subsets FD002 and FD004 run in six different operating conditions, and it should be taken into consideration that the sensor

parameters can be affected by various operating conditions. This problem can be solved by clustering all running data into six species based on operating conditions and conducting z-score normalization on cycles belong to same condition separately. The clustering result of operating conditions based on K-means is shown in Fig. 4.

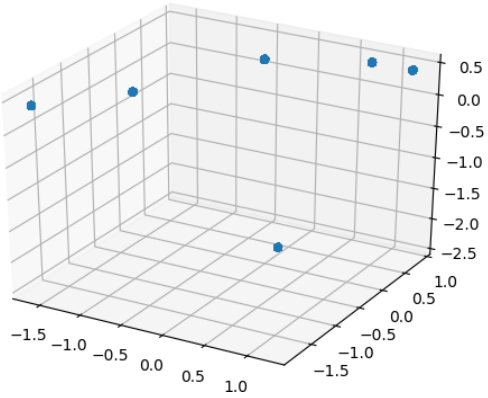

Fig.4 Clustering result of operating conditions based on K-means.

4.3 RUL Estimation

After reconstructing all engines in dataset, we can see that engines degrade variously and their lifetimes are individualized, which verify fixed RUL target function or degradation model is too restricted for complex systems. Fig.5 shows reconstruction errors and degradation curves of engines in training set and test set. Degradation curve is obtained by filtering the corresponding reconstruction error. We can see that fast degradation begins around 150th cycle in this training engine, and degradation curve of test engine starts at about 0.5, which indicates a slight deviation from the healthy state.

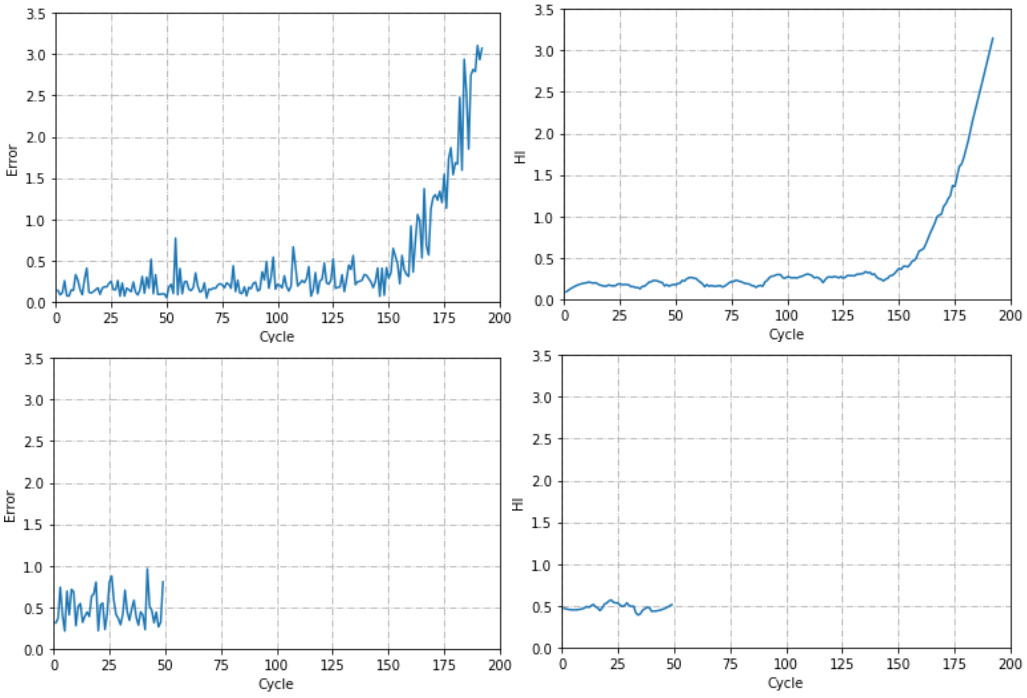

Fig.5. Reconstruction error (Top left) and degradation curve (Top right) of training engine, reconstruction error (Bottom left) and degradation curve (Bottom right) of test engine

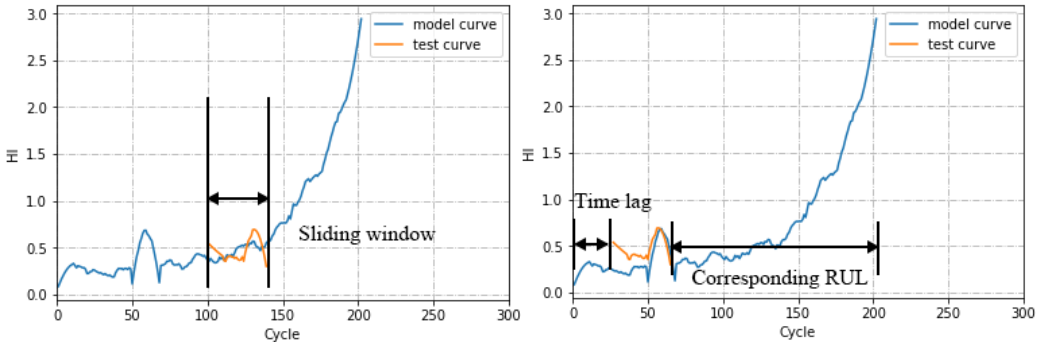

Fig.6. Segments in sliding window are combined to calculate CID with a step length of 1 and RUL_j^i is calculated after finding the most similar segment.

Fig.6 shows the process of getting similarity and corresponding RUL between test curve and model curve. A sliding window of the same length as test series is used to calculate CID with a step length of 1. Two sequences with smallest distance are considered to be in the same degradation state and the reciprocal of this smallest CID is denoted as their similarity. We can see there exist a time lag between two sequences with smallest CID, which means the test engine begin to run with initial wear and degrade similarly to the template engine at that point. Based on the lengths of two curves and the time lag, corresponding RUL_j^i can be calculated by (9).

$$RUL_j^i = \mathrm{N}_{model}^j - N_{test}^i - N_t \tag{9}$$

where RUL_j^i is the corresponding RUL between i-th test curve and j-th model curve, N_{model}^j is the number of running cycles of model curve j, N_{test}^i is the number of running cycles of test engine i and N_t is the length of time lag.

After getting all $\mathrm{sim}(Q^i, C^j)$ and RUL_j^i, the estimated RUL can be calculated as (10), where RUL_{est}^i is the estimated RUL of engine i, m is the number of useful corresponding RULs.

$$RUL_{est}^i = \frac{\mathrm{sim}(Q^i,C^1)}{\sum_{j=1}^m \mathrm{sim}(Q^i,C^j)} \times RUL_1^i + \frac{\mathrm{sim}(Q^i,C^2)}{\sum_{j=1}^m \mathrm{sim}(Q^i,C^j)} \times RUL_2^i + \cdots + \frac{\mathrm{sim}(Q^i,C^m)}{\sum_{j=1}^m \mathrm{sim}(Q^i,C^j)} \times RUL_m^i \tag{10}$$

In order to achieve results with high confidence, only those corresponding RULs whose similarities are not less than 70% of the maximum similarity are used. An example of one test engine's similarity distribution with all model curves is shown in Fig.7.

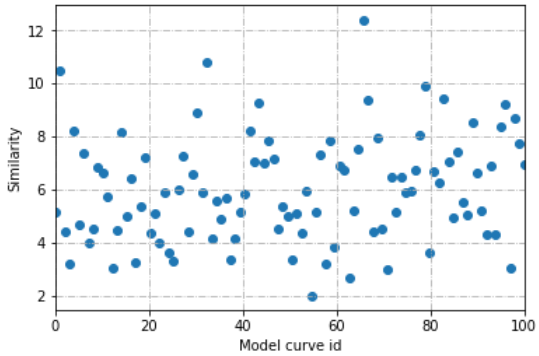

Fig.7. An example of one test engine's similarity distribution with all model curves. It is obvious that some model curves are far from the test curve.

4.4 Performance Comparison

As a public dataset, there are many studies on C-MAPSS dataset. Here we compare our results with some researches which are conducted with the fixed RUL target function and same normalization method as well as evaluation indexes. As shown in Table II and III, our proposed method makes progress in both RMSE and score. This is because the unsupervised learning autoencoder can capture degradation information adequately and similarity measurement with CID can make full use of historical information.

TABLE II. SCORE COMPARISON ON C-MAPSS DATASET

Sub dataset	FD001	FD002	FD003	FD004
SVR [9]	1.38×10^3	5.90×10^5	1.60×10^3	3.71×10^5
RVR [9]	1.50×10^3	1.74×10^4	1.43×10^3	2.65×10^4
CNN [9]	1.29×10^3	1.36×10^4	1.60×10^3	7.89×10^3
Deep LSTM [8]	3.38×10^2	4.45×10^3	8.52×10^2	5.55×10^3
Our method	3.24×10^2	3.25×10^3	6.73×10^2	4.12×10^3

TABLE III. RMSE COMPARISON ON C-MAPSS DATASET

Sub dataset	FD001	FD002	FD003	FD004
SVR [9]	20.96	42.00	21.05	45.35
RVR [9]	23.80	31.30	22.37	34.34
CNN [9]	18.45	30.29	19.82	29.16
Deep LSTM [8]	16.14	24.49	16.18	28.17
Our method	15.09	21.34	15.23	24.68

5 Discussion and Conclusion

In this paper, we combine autoencoder and similarity measurement to estimate RUL of aircraft engines. An autoencoder is trained to capture hidden correlation of multi-sensor data and extract degradation curve of engines. During estimation, a sliding window and CID are used to calculate the similarity between test curves and each model curve, making full use of historical information. The experiment result proves that our method can obtain individualized degradation models of engines adequately and improve the accuracy of aircraft engine RUL estimation.

Though comparison with other researches shows our approach is effective, it should be noted that it is still limited. Since we need to build a degradation curve template library, training objects need to be run-to-failure, or in other words remaining cycles at certain time is necessarily known.

Further improvements are still available. In the process of degradation model extraction, we just use a basic autoencoder architecture, and more complex structure such as sparse autoencoder and denoising autoencoder can be tried to improve representative learning ability. In addition, different distance calculation methods can also be explored to make estimation more robust.

References

[1] Wang T (2010) Trajectory similarity based prediction for remaining useful life estimation. University of Cincinnati.

[2] Marble S, Morton BP (2006) Predicting the Remaining Useful Life of Propulsion System Bearings. In: Proceedings of the IEEE Aerospace Conference, 2006, pp. 1-8.

[3] Giantomassi A, Ferracuti F, Benini A, Longhi S, Petrucci A (2011) Hidden Markov Model for Health Estimation and Prognosis of Turbofan Engines. In: ASME 2011 International Design Engineering Technical Conferences &Computers and Information in Engineering Conference, 2011, pp. 1-6.

[4] Liu DT, Pang J, Zhou J, Peng Y (2012) Data-driven prognostics for lithium-ion battery based on gaussian process regression. In: Proc. IEEE Conf. PHM, May 2012, pp. 1-5.

[5] Ren L, Sun Y, Wang H, Zhang L (2018) Prediction of bearing remaining useful life with deep convolution neural network. In: IEEE Access, vol. 6, pp. 13041-13049, 2018.

[6] Xue F, Bonissone P, Varma A, Yan W, Eklund N, Goebel K, (2008) An instance-based method for remaining useful life estimation for aircraft engines. Journal of Failure Analysis and Prevention. 8(2), pp. 199-206, 2008.

[7] Hochreiter S, Schmidhuber Jürgen (1997) Long short-term memory. Neural Computation, 9(8), 1735-1780.

[8] 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). IEEE, 2017, pp. 88-95.

[9] Heimes FO (2008) Recurrent neural networks for remaining useful life estimation. Prognostics and Health Management, 2008. PHM 2008. International Conference on. IEEE, 2008, pp. 1–6.

[10] Babu GS, Zhao P, Li XL (2016) Deep convolutional neural network based regression approach for estimation of remaining useful life. In: International Conference on Database Systems for Advanced Applications. Springer, 2016, pp. 214–228.

[11] Aamodt A, Plaza E (1994) Case-based reasoning: foundational issues, methodological variations, and system approaches. Ai Communications, 7(1), 39-59.

[12] Batista GEAPA, Keogh EJ, Tataw OM, Souza VCMA (2014) Cid: an efficient complexity-invariant distance for time series. Data Mining and Knowledge Discovery, 28(3), 634-669.

[13] Rumelhart, DE (1986) Learning representations by back-propagating errors. Nature, vol. 323, no. 6088, pp. 533-536, 1986.

[14] Park D, Hoshi Y, Kemp CC (2017) A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. IEEE Robotics and Automation Letters, vol. 3, no. 3, pp. 1544-1551, July 2018.

[15] Malhotra P, Ramakrishnan A, Anand G, Vig L, Agarwal P, Shroff G (2016) Lstm-based encoder-decoder for multi-sensor anomaly detection. Presented at ICML 2016 Anomaly Detection Workshop, Jul. 2016.

[16] Salvador S, Chan P (2007) Toward accurate dynamic time warping in linear time and space. Intelligent Data Analysis, 11(5), 561-580.

[17] Saxena A, Goebel K, Simon D, Eklund N (2008) Damage propagation modeling for aircraft engine run-to-failure simulation. In: 2008 International Conference on Prognostics and Health Management, Denver, CO, 2008, pp. 1-9.