



Spatio-temporal graph convolutional neural network for remaining useful life estimation of aircraft engines

Mengni Wang¹ · Yuanxiang Li¹ · Yuxuan Zhang¹ · Lei Jia¹

Received: 26 March 2020 / Revised: 27 July 2020 / Accepted: 17 October 2020 / Published online: 17 November 2020
© The Author(s) 2021

Abstract

Accurate remaining useful life (RUL) estimation is crucial for the maintenance of complex systems, e.g. aircraft engines. Thanks to the popularity of sensors, data-driven methods are widely used to evaluate RULs of systems especially deep learning approaches. Though remarkably capable at non-linear modeling, deep learning-based prognostics techniques lack powerful spatio-temporal learning ability. For instance, convolutional neural networks are restricted to only process grid structures rather than general domains, recurrent neural networks neglect spatial relations between sensors and suffer from long-term dependency learning. To solve these problems, we construct a graph structure on sensor network with Pearson Correlation Coefficients among sensors and propose a method for combining the power of graph convolutional network on spatial learning and sequence learning success of temporal convolutional networks. We conduct the proposed method on aircraft engine dataset provided by NASA. The experimental results demonstrate that the established graph structure is appropriate and the proposed approach can model spatio-temporal dependency accurately as well as improve the performance of RUL estimation.

Keywords Aircraft engine · Remaining useful life estimation · Graph structure · Spatio-temporal modeling

1 Introduction

In view of the major consequences and huge expense are usually relevant to system failures, prognostics and health management (PHM) has received increasing attention in academia and industry. As a challenging task in PHM, prognostics aims to estimate how much time remains before a likely failure or in other words remaining useful life (RUL) to promote reliability and safety. Methods to evaluate RUL can be mainly grouped into three major categories: model-based methods, experience-based methods and data-driven methods [1]. The former requires extensive expert knowledge about systems to build a physical model [2–4],

which is usually unavailable in practice, e.g. aircraft engines. Experience-based methods leverage historical failure cases to build hidden relationship among current system states, current lives and recorded failure instances. However, methods included in this category, e.g. instance-based methods [5, 6], require more calculations as reference cases increase and depend too much on run-to-failure cases.

Data-driven methods, especially deep learning techniques, have emerged as a research hotspot with the availability of sensor data from numerous systems. The clipped sensor readings and their corresponding RULs are used as inputs to train a deep neural network, and the network can discover intricate relations between them with powerful non-linear modeling capabilities. For example, Babu et al. [7] treated segmented sensor readings as 2-dimensional (2D) inputs and utilized 2D convolution to extract more informative feature. Li et al. [8] stacked 1-dimensional (1D) convolution layers for feature extraction of each sensor signals and predicted RUL by a fully-connected layer. Malhi et al. [9] proposed a learning-based method for long-term prognostics of rolling bearing with recurrent neural network (RNN). Wu et al. [10] combined a long short-term memory network (LSTM) based model with a dynamic difference method to get rid of operating condition influence.

✉ Yuanxiang Li
yuanxli@sjtu.edu.cn

Mengni Wang
moniwong@sjtu.edu.cn

Yuxuan Zhang
yuxuanzhang@sjtu.edu.cn

Lei Jia
westjia@sjtu.edu.cn

¹ School of Aeronautics and Astronautics, Shanghai Jiao Tong University, Shanghai 200240, China

However, methods mentioned above neglect spatial relationships between different sensors and RNN-based structures are possible to accumulate error by steps and difficult to capture long-term dependence. In this work, we adopt two crucial strategies to solve these two problems. First, we compute the weighted adjacency matrix of sensor network with Pearson correlation coefficients (PCCs) among sensors. Second, graph convolutional networks (GCNs) and temporal convolutional networks (TCNs) constituted by stacked dilated casual convolutions work together to capture spatio-temporal dependencies followed by gating mechanism and skip connections.

The rest of the paper is organized as follows. Section 2 introduces the related work, including convolution on graph structure and spatio-temporal graph neural networks. Section 3 presents how to build graph structure of sensor network via PCCs and introduces the proposed framework. Section 4 shows the experimental results on C-MAPSS dataset. The last section draws the conclusion and discusses the future work.

2 Related work

2.1 Convolution on graph structure

Convolutions in the graph domain can be categorized as spatial approaches and spectral approaches [11]. The former defines convolutions directly on graph which is challenging to deal with differently sized neighborhoods [12–14]. Li et al. [12] model the spatial dependency based on diffusion process as:

$$g_\theta * x = \sum_{k=0}^{K-1} (\theta_{k,1} A_1^k + \theta_{k,2} A_2^k) x, \quad (1)$$

where $x \in \mathbb{R}^N$ is the input; g_θ is a filter parameterized by $\theta \in \mathbb{R}^{K \times 2}$; A_1 and A_2 are the transition matrices of the diffusion process and the reverse one; k is the diffusion step and in [12] a finite K -step truncation of the diffusion process is adopted.

The latter operated convolutions based on Fourier domain [15], which is defined as:

$$g_\theta * x = g_\theta(L)x = g_\theta(U \Lambda U^T)x = U g_\theta(\Lambda) U^T x, \quad (2)$$

where g_θ is a filter parameterized by $\theta \in \mathbb{R}^N$; $x \in \mathbb{R}^N$ is the signal; $L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ is the normalized graph Laplacian (D is the degree matrix and A is the adjacency matrix of the graph); Λ is the diagonal matrix of eigenvalues of L ; U is the matrix of eigenvectors of L .

Further to reduce the high computing consume of Eq. (2), many researchers proposed different methods. Hammond

et al. [16] approximate the $g_\theta(\Lambda)$ with a truncated expansion in terms of Chebyshev polynomials up to K th order:

$$g_\theta * x \approx \sum_{k=0}^{K-1} \theta_k T_k(\tilde{L}) x, \quad (3)$$

where $\tilde{L} = \frac{2}{\lambda_{\max}} L - I_N$, λ_{\max} is the largest eigenvalue of L ; $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ with $T_0(x) = 1$ and $T_1(x) = x$; $\theta \in \mathbb{R}^K$ is a vector of Chebyshev coefficients.

More efforts are made to overcome the problem of overfitting and numerical instabilities in [17], where a first-order approximation of Chebyshev polynomials and some normalization tricks are introduced:

$$\begin{aligned} g_\theta * x &\approx \theta'_0 x + \theta'_1 (L - I_N) x = \theta'_0 x - \theta'_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} x \\ &= \theta \left(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \right) x = \theta \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} x, \end{aligned} \quad (4)$$

where $\theta = \theta'_0 = -\theta'_1$, θ'_0 and θ'_1 are two free parameters; $\tilde{A} = A + I_N$ is the adjacency matrix with self-connections, I_N is the identity matrix; $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$.

2.2 Spatio-temporal graph neural networks

Spatio-temporal graph neural networks have a wide range of applications, e.g. traffic forecasting, action forecasting and wind speed forecasting [18–22]. In these tasks, the key is to determine the optimal combinations of spatial information and temporal dynamics under specific settings. Li et al. [12] replaced the matrix multiplications in gated recurrent units (GRU) with diffusion convolutions to extract spatio-temporal features of traffic flow, which was modeled as a directed graph. Seo et al. [19] combined convolutional neural network (CNN) on graphs to identify spatial structures and RNN to find dynamic patterns for predicting structured sequences of data, e.g. frames in videos.

However, RNN-based approaches are inefficient for long sequences and more likely to explode when they are combined with GCN [20]. Yu et al. [21] computed the distances among stations as the adjacency matrix of road graph, and then utilized complete convolutional structures to extract spatio-temporal features of traffic network. Yan et al. [22] designed a generic representation of skeleton sequences and extended graph neural networks with 1-D CNN to model dynamic skeletons effectively. Zhang et al. [23] used adaptive graph convolution to learn spatio-temporal features for RUL prediction. Different from their work, we explore a different graph convolution process and a different network structure.

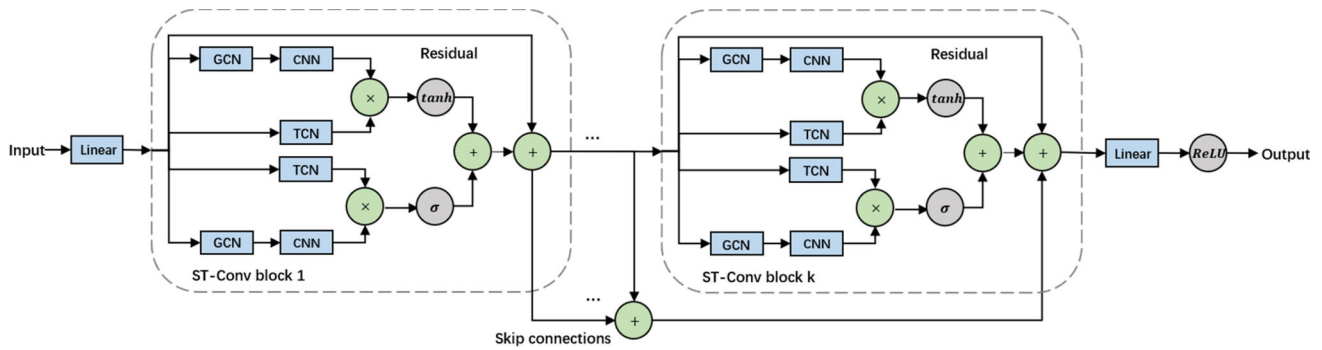


Fig. 1 Structure of the proposed method

3 RUL estimation based on spatio-temporal graph convolutional network

In this work, the graph structure of sensor network is constructed with PCCs and a method combining GCNs and TCNs is proposed to estimate RULs. Figure 1 shows the structure of our model where spatio-temporal convolutional blocks (ST-Conv blocks) are crucial components. Each ST-Conv block consists of two GCNs and two TCNs followed by gating mechanism to fuse both spatial and temporal features. Residual connections are applied in blocks and skip connections are utilized throughout the network to speed up convergence. The fully connected layer is utilized to get the final RUL result.

3.1 Problem definition

Generally, RUL estimation is a typical time-series prediction problem that predicting the system will fail after \hat{y}^t time steps based on given T running observations:

$$[x^{t-T+1}, \dots, x^t] \rightarrow^{h(\cdot)} \hat{y}^t. \tag{5}$$

where \hat{y}^t is the estimated RUL; $x^i \in \mathbb{R}^N$ ($i = t - T + 1, \dots, t$) is recorded sensor readings at time i ; $h(\cdot)$ is a function that maps T running observations to the RUL at time t .

Figure 2 shows a simplified diagram of engine. We can see various subroutines are assembled together and we can monitor its status by collecting data from different parts, such as pressure at fan inlet, HPC stall margin, temperature at LPT outlet and so on. It is undoubtful that each part will be affected by other parts even operating environments rather than independent of each other. Considering the external and internal influences, graph structure is very suitable for this situation since it can aggregate information from neighboring nodes. In this work, we define the sensor network as a graph where sensors are nodes, their relations are edges, impact magnitude and its positive or negative represent weighted adjacency matrix. The schematic diagram is shown in Fig. 3.

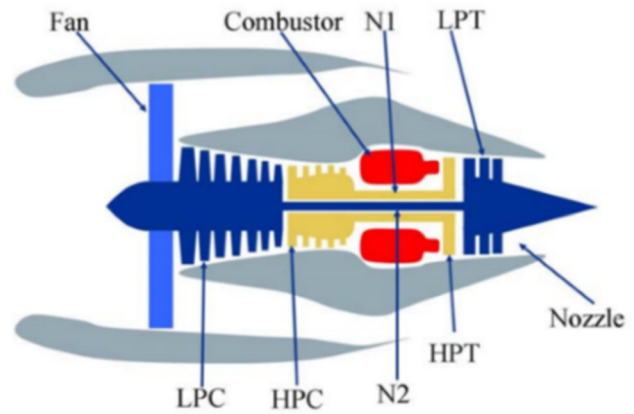


Fig. 2 Simplified diagram of engine [24]

Combined with the graph structure, the original Eq. (5) can be expanded to:

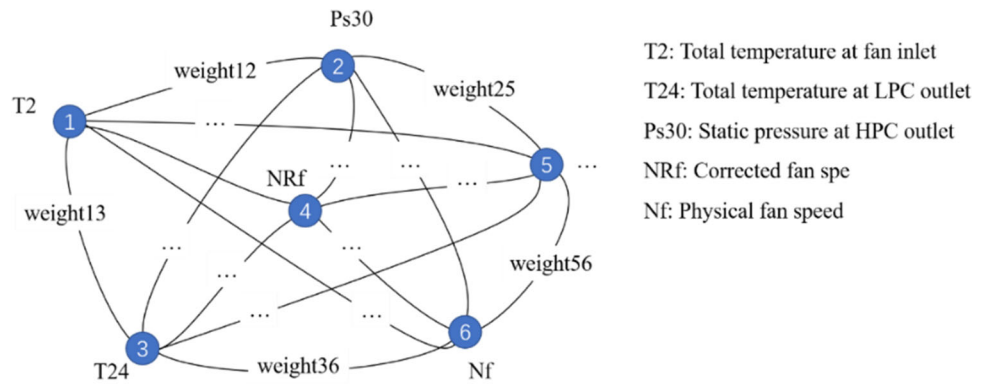
$$[x^{t-T+1}, \dots, x^t; \mathcal{G}] \rightarrow^{h(\cdot)} \hat{y}^t, \tag{6}$$

where $\mathcal{G} = (\mathcal{V}, \mathcal{E}, A)$ is the graph structure of sensor network; \mathcal{V} is the set of nodes or in other words sensors with a total of N ; \mathcal{E} is the set of edges; $A \in \mathbb{R}^{N \times N}$ is a weighted adjacency matrix derived from other nodes.

3.2 Graph convolution for spatial information

Different from traffic forecasting or action recognition where the adjacency matrix can be computed based on Euclidean distance among stations in traffic network or the intra-body connections of joints can be represented by distance according to partitioning strategies, relationships between sensors are not provided explicitly in RUL estimation tasks. Since the proximity of two sensors in Euclidean domain doesn't mean they are closely related, Euclidean distance is not the optimal metric for relationship modeling. Hence, we construct the weighted adjacency matrix based on PCCs among sensors. The formulation of PCCs for sequence X and Y is:

Fig. 3 Graph structure of sensor network



$$\rho_{X,Y} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X \sigma_Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}\sqrt{E(Y^2) - E^2(Y)}}, \tag{7}$$

where $\text{cov}(X, Y)$ is the covariance of X and Y ; σ_X and σ_Y are the standard deviations of X and Y ; μ_X and μ_Y are the means of X and Y ; $\rho_{X,Y}$ is a number between -1 and 1 that indicates the extent to which two variables are related. Then the weighted adjacency matrix is defined as:

$$A_{ij} = e^{\rho_{X_i} X_j}. \tag{8}$$

Further based on the research of Li et al. [12] and Eq. (4), GCN on sensor readings can be represented as:

$$Z = \sum_{k=0}^{K-1} P^k X W_k, \tag{9}$$

where $P = \frac{A}{\sum_j A_{ij}}$ is the transition matrix; W_k are trainable weights.

3.3 Spatio-temporal convolutional block

The ST-Conv block takes advantages of CNNs and RNNs as well as prevents their disadvantages. Due to the defects of RNN-based structure, here we choose dilated causal convolution [25] as temporal convolutional layer to extract temporal dynamics. Dilated causal convolution can learn long-term dependency properly in a non-recursive manner and increase the receptive field without greatly increasing computational cost or stacking many layers [26]. The formulation of dilated causal convolution is:

$$x * f(t) = \sum_{s=0}^{K-1} f(s)x(t - d * s), \tag{10}$$

where $x \in \mathbb{R}^T$ is a 1D sequence input; $f(\cdot) \in \mathbb{R}^K$ is a filter; d is the dilation factor which controls the skipping distance.

LSTM outperforms than basic RNN in sequences modeling thanks to the gating mechanism which can effectively control the flow of information. Learning from the success of LSTM and to enrich convolution operations with

nonlinearity, 1-D dilated causal convolution and gating mechanism work together to extract features on time axis [27]. Furthermore, to fuse information from both spatial and temporal domains simultaneously, we embed GCN into ST-Conv block. So, the formulation of ST-Conv block can be defined as follows:

$$Y = \tanh(\theta_{1,1} * X + \theta_{1,2} * Z) \odot \sigma(\theta_{2,1} * X + \theta_{2,2} * Z), \tag{11}$$

where \odot is the Hadamard product; $\sigma(\cdot)$ is the sigmoid activation function; $\theta_{1,1}$, $\theta_{1,2}$, $\theta_{2,1}$ and $\theta_{2,2}$ are convolution parameters; Z is the output of GCN; X is the output of TCN.

4 Experiment

4.1 Dataset description

Saxena et al. [24] modeled a series of turbofan engines by Commercial Modular Aero-Propulsion System Simulation and recorded their run-to-failure data as C-MAPSS dataset. As shown in Table 1, the dataset contains four subsets covering different operating conditions and fault modes and they are further divided into training and test subsets. Each record includes 26 channels, involving engine id, running time (in cycles), 3-dimensional operating condition settings and 21-dimensional sensor readings. During training, all available data points from all engines are used as training samples. And for testing, only one data point corresponding to the last recorded operational cycle for each engine is used as the test sample [8].

4.2 Data preprocess

In practical application, degradation of a system tends to be negligible in the initial stage and increases as it approaches run-to-failure. Hence, we utilize a piece-wise linear degradation model to obtain RUL labels with respect to each sample, whose maximum RUL is set as 130 according to the research of Zheng et al. [28].

Though each observation is 26-dimensional, some channels do not provide valuable information about degradation, i.e. records on sensor #1, #5, #10, #16, #18, #19 and setting3. Each selected channel is normalized by z-score normalization:

$$z = \frac{x-\mu}{\sigma}, \quad (12)$$

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

4.3 Experimental setting and evaluation metrics

The experiments are conducted on a 64-bit windows server with one Intel Core i7-7800 × CPU, a 64G RAM and one Nvidia GTX 1080Ti GPU. The default length of sliding window is 30. The selected observations are setting1, setting2, sensor#2, #3, #4, #6, #7, #8, #9, #11, #12, #13, #14, #15, #17, #20, #21 and cycle in turn.

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 average score (AS) is employed to penalize late prediction. Its definition is denoted as:

$$AS = \frac{1}{n} \sum_{i=1}^n s_i, \quad (13)$$

where n is the number of test samples; s_i is the computed score of i th sample:

$$s_i = \begin{cases} e^{-\left(\frac{d_i}{a_1}\right)} - 1 & \text{for } d_i < 0 \\ e^{\frac{d_i}{a_1}} - 1 & \text{for } d_i \geq 0 \end{cases}, \quad (14)$$

where $d_i = RUL_i^{\text{true}} - RUL_i^{\text{pre}}$ is the RUL estimation error of the i th engine; $a_1 = 13$ and $a_2 = 10$ are default factors in two cases.

We can know from Eq. (14) that if there are some instances whose d is large enough, then the score will grow exponentially. Furthermore, we also adopt another fair evaluation index called Root Mean Square Error (RMSE), it is:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n d_i^2}, \quad (15)$$

where d_i is the estimation error of the i th engine; n is the number of test samples.

4.4 Experiment results

To show the effectiveness of our approach, we compare its performance with other methods which are conducted under the same data division way and same evaluation metrics.

Table 1 Information of 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

The Adam optimization algorithm and RMSE loss function are used during training. The comparison results are shown in Table 2. We can know from Table 2 that our proposed method has competitive performance in both RMSE and AS especially for those operating condition is more complex. Compared with previous approaches whose AS of FD002 and FD004 is fifteen times more than it in a single environment, our method reaches a more similar level in different environments. This proves GCN is effective for aggregating spatial information from different orders of neighboring nodes. Figure 4 shows RUL prediction samples of four subsets.

4.5 Graph structure analysis

4.5.1 Analysis of weighted adjacency matrix

Figure 5 shows weighted adjacency matrixes of FD001 and FD002 based on PCCs. Since data in FD001 is collected under single operating condition, 0th and 1st observations (i.e. setting1 and setting2) are unrelated with other sensor readings, while in FD002 sensors are influenced by them. So, the weighted adjacency matrixes can not only identify relations of running data but also model the influence of operating conditions.

4.6 Analysis the effectiveness of GCN

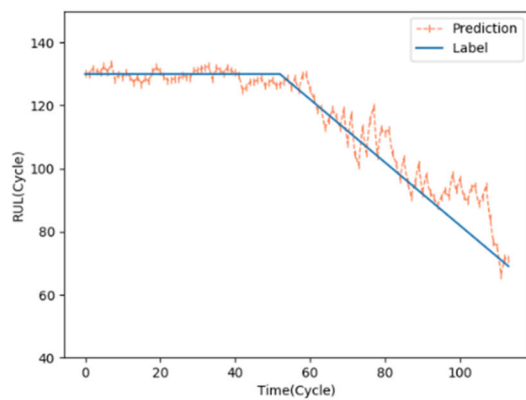
To further prove the effectiveness of our approach, we compare our method with its non-GCN version. Figure 6 shows their performance differences with respect to RMSE and AS. We can see non-GCN method performs slightly worse on RMSE and AS in FD001 and FD003 but much worse in FD002 and FD004 where environments are more complex. This demonstrates that our method can cope with multiple operational conditions and RUL estimation tasks can benefit from spatial information.

4.7 Effects of sequence length on performance

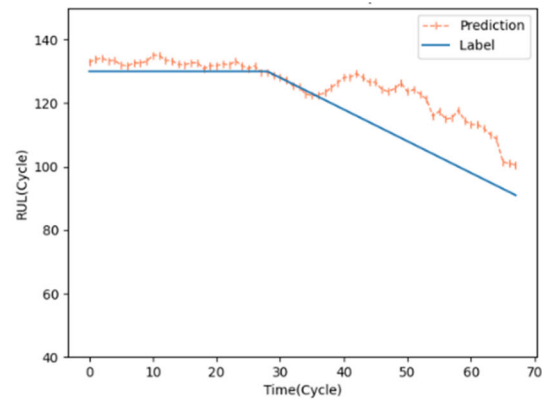
To explore the effect of sequence length on model performance, here we take FD001 dataset as an example. Figure 7 shows the effect of sequence length on model performance. The histogram represents RMSE and the line is average training time. It can be observed that with the growth of sequence length, average training time tends to increase while

Table 2 Performance comparisons of different methods on C-MAPSS dataset

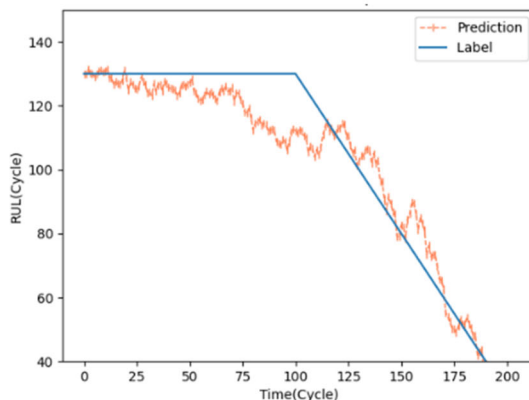
Method	FD001		FD002		FD003		FD004	
	RMSE	AS	RMSE	AS	RMSE	AS	RMSE	AS
NN [8]	14.80	4.96	25.64	70.48	15.22	5.22	25.80	82.35
DNN [8]	13.56	3.48	24.61	60.32	13.93	3.64	24.31	65.42
RNN [8]	13.44	3.39	24.03	55.00	13.36	3.16	24.02	56.17
LSTM [8]	13.52	4.32	24.42	55.83	13.54	3.47	24.21	57.75
MODBNE [29]	15.04	3.34	25.05	21.57	12.51	4.22	28.66	26.44
DCNN1 [8]	18.45	12.87	30.29	53.00	19.82	15.96	29.16	31.80
DCNN2 [8]	12.61	2.74	22.36	40.20	12.64	2.84	23.31	50.27
Our method	12.76	2.66	17.74	9.98	12.07	2.78	18.08	10.38



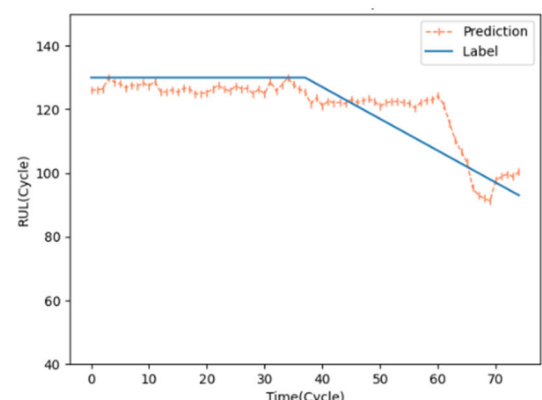
(a) FD001



(b) FD002



(c) FD003



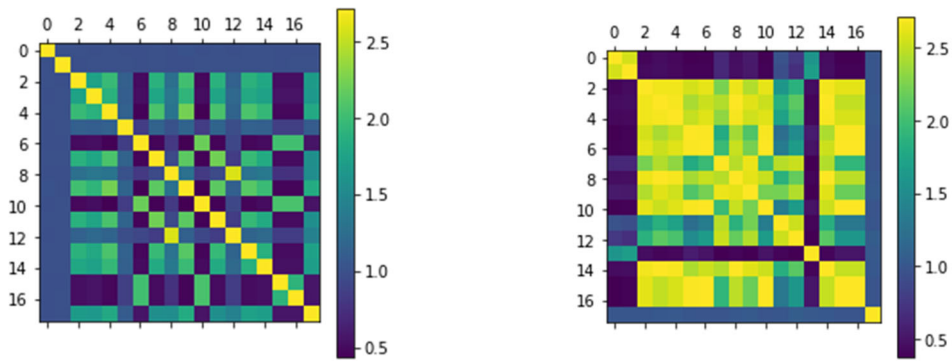
(d) FD004

Fig. 4 RUL prediction samples of four subsets

the RMSE continues to decrease. Though the RMSE keeps going down, average training time grows rapidly when the sequence length is greater than 30. So, we choose 30 as our default sequence length in experiments.

5 Conclusion

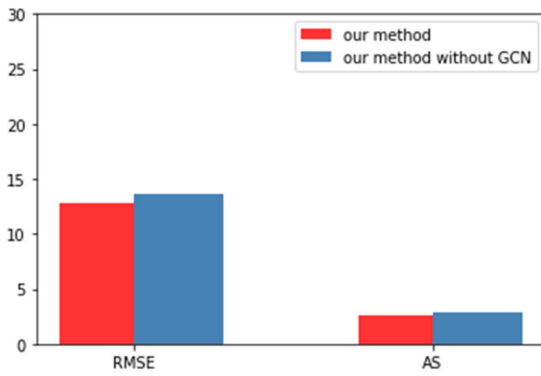
This paper constructs the graph structure of the sensor network properly and proposes an effective method to estimate RUL, which combines GCN and Gated-TCN to substantially fuse spatial and temporal features. The experiments on the C-MAPSS dataset prove that our method can extract spatio-temporal



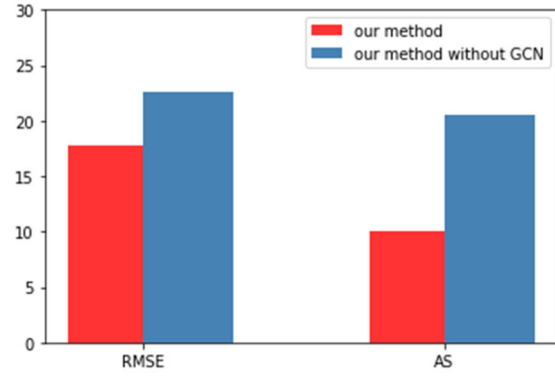
(a) The weighted adjacency matrix of FD001

(b) The weighted adjacency matrix of FD002

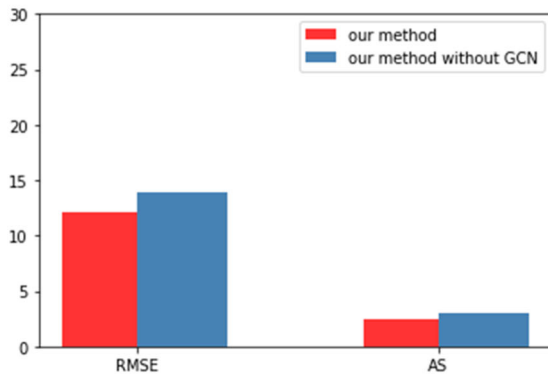
Fig. 5 Weighted adjacency matrixes of FD001 and FD002



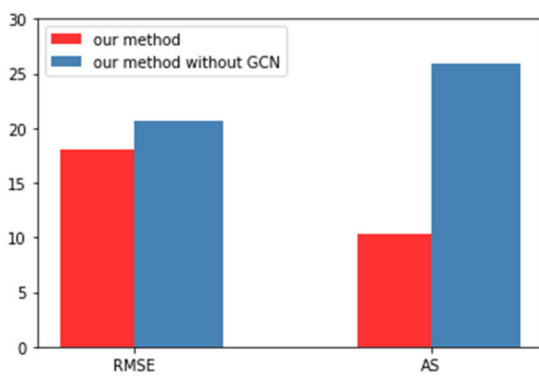
(a) FD001



(b) FD002



(c) FD003



(d) FD004

Fig. 6 Performance differences with respect to RMSE and AS

features adequately and improve the accuracy of RUL estimation. Analysis about weighted adjacency matrix indicates that the constructed graph is qualified for multi-environment tasks. Furthermore, the effects of sequence length on prognostic performance are investigated.

Further improvements are still available. For instance, more precise adjacency matrix and more effective spatio-temporal structure deserves more exploration.

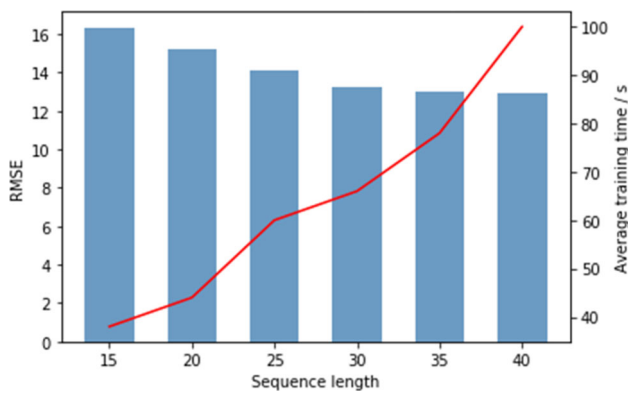


Fig. 7 The effect of sequence length on model performance. The histogram represents RMSE and the line is average training time

Acknowledgements The authors thanks to NASA for providing the C-MAPSS dataset.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Wang T (2010) Trajectory similarity based prediction for remaining useful life estimation. University of Cincinnati, Cincinnati
- Jouin M, Gouriveau R, Hissel D, Péra M, Zerhouni N (2016a) Particle filter-based prognostics: review, discussion and perspectives. *Mech Syst Signal Process* 72:2–31
- Jouin M, Gouriveau R, Hissel D, Péra MC, Zerhouni N (2016b) Degradations analysis and aging modeling for health assessment and prognostics of PEMFC. *Reliability Eng Syst Safety* 148:78–95
- Ali JB, Chebelmorello B, Saidi L, Malinowski S, Fnaiech F (2015) Accurate bearing remaining useful life prediction based on Weibull distribution and artificial neural network. *Mech Syst Signal Process* 56:150–172
- 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. *J Fail Anal Prev* 8(2):199–206
- Wang M, Li Y, Zhao H, Zhang Y (2020) Combining autoencoder with similarity measurement for aircraft engine remaining useful life estimation. In: *Proceedings of the international conference on aerospace system science and engineering 2019*. Springer, pp 197–208
- Babu G S, Zhao P, Li X (2016) Deep convolutional neural network based regression approach for estimation of remaining useful life. In: *International conference on database systems for advanced applications*. Springer, pp 214–228
- Li X, Ding Q, Sun J (2018) Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Eng Syst Safety* 172:1–11
- Malhi A, Yan R, Gao RX (2011) Prognosis of defect propagation based on recurrent neural networks. *IEEE Trans Instrum Meas* 60(3):703–711
- Wu Y, Yuan M, Dong S, Lin L, Liu Y (2018) Remaining useful life estimation of engineered systems using vanilla LSTM neural networks. *Neurocomputing* 275:167–179
- Zhou J, Cui G, Zhang Z, Yang C, Liu Z, Wang L, et al (2018) Graph neural networks: a review of methods and applications. *ArXiv Preprint arXiv:1812.08434* [cs.LG]
- Li Y, Yu R, Shahabi C, Liu Y (2018) Diffusion convolutional recurrent neural network: data-driven traffic forecasting. *ArXiv Preprint arXiv:1707.01926* [cs.LG]
- Duvenaud D, Maclaurin D, Aguilera-Iparraguirre J, Gomez-bombarelli R, Hirzel T, Aspurguzik A, Adams R (2015) Convolutional networks on graphs for learning molecular fingerprints. In: *International conference on neural information processing (NIPS)*, pp 2224–2232
- Gao H, Wang Z, Ji S (2018) Large-scale learnable graph convolutional networks. In: *Proceedings of SIGKDD*. ACM, pp 1416–1424
- Bruna J, Zaremba W, Szlam A, LeCun Y (2013) Spectral networks and locally connected networks on Graphs. *ArXiv Preprint arXiv:1312.6203* [cs.LG]
- Hammond D, Vandergheynst P, Gribonval R (2011) Wavelets on graphs via spectral graph theory. *Appl Comput Harmonic Anal* 30(2):129–150
- Kipf T, Welling M (2016) Semi-supervised classification with graph convolutional networks. *ArXiv Preprint arXiv:1609.02907* [cs.LG]
- Jain A, Zamir AR, Savarese S, Saxena A (2016) Structural-RNN: deep learning on spatio-temporal graphs. In: *2016 IEEE conference on computer vision and pattern recognition (CVPR)*, pp 5308–5317
- Seo Y, Defferrard M, Vandergheynst P, Bresson X (2017) Structured sequence modeling with graph convolutional recurrent networks. In: *International conference on neural information processing*, pp 362–373
- Wu Z, Pan S, Long G, Jiang J, Zhang C (2019) Graph WaveNet for Deep spatial-temporal graph modeling. In: *Proceedings of the twenty-eighth international joint conference on artificial intelligence*, pp 1907–1913
- Yu B, Yin H, Zhu Z (2018) Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: *27th international joint conference on artificial intelligence*, pp 3634–3640
- Yan S, Xiong Y, Lin D (2018) Spatial temporal graph convolutional networks for skeleton-based action recognition. In: *AAAI-18 AAAI Conference on Artificial Intelligence*, pp 7444–7452
- Zhang Y, Li Y, Wei X, Jia L (2020) Adaptive spatio-temporal graph convolutional neural network for remaining useful life estimation. *International Joint conference on Neural Networks (IJCNN)*. Accepted
- 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., pp 1–9
- Kalchbrenner N, Espeholt L, Simonyan K, Den Oord A V, Graves A, Kavukcuoglu K (2016) Neural Machine Translation in Linear Time. *ArXiv Preprint arXiv:1610.10099* [cs.CL]
- Yu F, Koltun V (2016) Multi-scale context aggregation by dilated convolutions. *ArXiv preprint arXiv: 1511.07122* [cs.CV]
- Den Oord A V, Kalchbrenner N, Vinyals O, Espeholt L, Graves A, Kavukcuoglu K (2016) Conditional image generation with PixelCNN decoders. *ArXiv preprint arXiv:1606.05328* [cs.CV]
- Zheng S, Ristovski K, Farahat A K, Gupta C (2017) Long short-term memory network for remaining useful life estimation. *2017 IEEE international conference on prognostics and health management (ICPHM)*. IEEE, pp 88–95
- Zhang C, Lim P, Qin AK, Tan KC (2017) Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. *IEEE Trans Neural Networks* 28(10):2306–2318