# Hierarchical Attention Graph Convolutional Network to Fuse Multi-sensor Signals for Remaining Useful Life Prediction

Tianfu Li , Zhibin Zhao , Chuang Sun , Ruqiang Yan , Xuefeng Chen

PII: S0951-8320(21)00397-5
DOI: https://doi.org/10.1016/j.ress.2021.107878
Reference: RESS 107878

To appear in: *Reliability Engineering and System Safety*

Received date: 8 March 2021
Revised date: 29 May 2021
Accepted date: 14 June 2021

# Hierarchical Attention Graph Convolutional Network to Fuse Multi-sensor Signals for Remaining Useful Life Prediction

Tianfu Li, Zhibin Zhao, Chuang Sun*, Ruqiang Yan, Xuefeng Chen

School of Mechanical Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi 710049, China.

*Corresponding author. E-mail address: ch.sun@xjtu.edu.cn.

**Highlights**

- • A novel multi-sensor features fusion approach for remaining useful life prediction is proposed;
- • Multiple sensors are constructed to a sensor network to generate spatial-temporal graphs;
- • Hierarchical attention graph convolutional network is constructed to model the generated spatial-temporal graphs;
- • Regularized self-attention graph pooling is proposed to obtain more informative graph representations;

## Abstract:

Deep learning-based prognostic methods have achieved great success in remaining useful life (RUL) prediction, since degradation information of machine can be adequately mined by deep learning techniques. However, these methods suffer from following weaknesses, that is, 1) interactions among multiple sensors are not explicitly considered; 2) they are more inclined to model temporal dependencies while ignoring spatial dependencies of sensors. To address those weaknesses, the multiple sensors are constructed to a sensor network and hierarchical attention graph convolutional network (HAGCN) is proposed in this paper for modeling the sensor network. In HAGCN, the hierarchical graph representation layer is proposed for modeling spatial dependencies of sensors and bi-directional long short-term memory network is used for modeling temporal dependencies of sensor measurements. Moreover, a regularized self-attention graph pooling is designed in HAGCN to achieve effective information fusion of the sensors. To realize prognostics, the spatial-temporal graphs are firstly generated based on the sensor network. Then, HAGCN is applied to model the spatial and temporal dependencies of the graphs simultaneously. The experimental results of two case studies show the superiority of HAGCN over state-of-the-art methods for RUL prediction.

**Keywords:** RUL prediction, multi-sensor information fusion, sensor network, spatial-temporal graphs, graph convolutional network.

## 1 Introduction

Prognostics and Health Management (PHM), with the goal to cut down the machine downtime and increase the reliability of system, has attracted more and more attention in industrial applications [1-3]. Typically, PHM uses domain knowledge to analyze monitoring data of sensors and find the valuable features to evaluate health condition of equipment [4], [5]. One important task in PHM is remaining useful life (RUL) prediction. With an accurate RUL prediction, that how long the equipment can operate continuously can be known without a system failure.

Benefited from the progress of sensor technologies, computer science, and internet of things, it is now possible to collect more comprehensive telemetry data. Such explosion has brought new challenges and opportunities to the RUL prediction of machine [6]. Therefore, how to establish a powerful RUL prediction model to mine degradation information from enormous telemetry data has become a research hotspot in PHM.

Approaches for RUL prediction can be mainly divided into two categories, i.e., model-driven approaches, and data-driven approaches. Model-driven approaches aim to model the dynamics of machine system accurately, which needs a lot of domain knowledge. However, owing to the complexity of machine system, it is hard to build an accurate mathematical model or physical model to estimate the RUL [7]. In addition, because the mechanism of different mechanical systems is quite distinctive, the model-driven approaches perform poorly in model transfer.

In recent years, data-driven approaches have swept the field of RUL prediction and achieved successful results. It can be further divided into conventional machine learning based approaches and deep learning (DL) based approaches. Conventional machine learning based methods, such as support vector machine [8], and hidden Markova model [9] can model more complex data, but needs extensive efforts for feature engineering. In contrast, DL-based approaches, which extracts features automatically without human intervention, have received a lot of attention. Many DL-based methods, e.g., auto-encoder (AE) [10], long-short-term memory networks (LSTM) [11], and convolutional neural network (CNN) [12], have also been developed for RUL prediction. For example, Ren [13] proposed an AE-based approach to extract features in time series for RUL estimation. Babu [14] proposed a DCNN to capture temporal features over multi-channel sensors, and used the extracted features for RUL estimation. Li [15] proposed a 1D-DCNN for multivariate temporal data feature extraction and applied the features for RUL prediction. Miao [16] proposed a dual-task LSTM for joint learning of degradation assessment and RUL prediction of machines, where the LSTM was used to capture the temporal features of time series. Huang [17] proposed a bi-directional LSTM to integrate multiple sensor data and operational condition signals for RUL prediction.

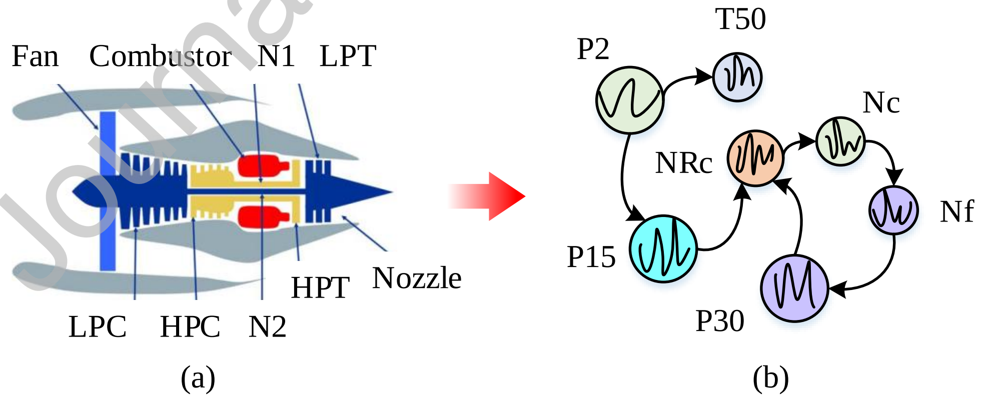


Fig. 1. The sensor network formed by sensors continuously generates spatial-temporal graphs. (a) Engine physical structure [18]; (b) Sensor network.

From the literature review, it can be observed that many existing DL-based models for RUL prediction focus more on modeling temporal dependencies of input sensor measurements, while ignoring spatial and geographic information of sensors. In industrial scenarios, the sensors mounted on the machine can naturally form a sensor network, and the sensor network constantly generate spatial-temporal graphs of health states of the machine, as shown in Fig. 1. The basic assumption to model such a spatial-temporal graph is that feature information of a node depends on historical information of itself and its neighbors. Only modeling the temporal dependencies of time series will not take the influence of other sensors into account. On the contrary, only modeling the spatial dependencies between sensors will not consider the influence of the current moment in the future. Therefore, it is necessary to consider how to model the spatial-temporal features and dynamic correlations of sensor measurements when doing RUL prediction of machine.

In order to make full use of spatial features, some researchers try to use CNN to capture the spatial dependencies between sensors, and then apply LSTM on the time axis to capture the temporal dependencies [18-21]. In addition, in [22], it proposed to combine the temporal prediction methods with a spatial analysis method to implement RUL prediction of equipment. However, it is difficult for them to model such graph data with dynamic node-level inputs in non-Euclidean space. Fortunately, with the prosperity and development of graph convolutional networks (GCNs) [23], it has been widely applied to data that represented in the form of graphs. For example, in drug discovery process, the molecules in the drug are modeled as graphs, and its chemical stability can be discovered with the help of GCNs [24]. In traffic flow forecasting, traffic data is construed as graphs, and GCNs are utilized to model the graphs and implement traffic flow forecasting [25]. Recently, in fault diagnosis, the signal samples are transformed into graphs, and GCNs are leveraged to learn the node representations and achieve fault diagnosis [26]. Therefore, it is possible to use GCNs to model the dynamic node-level inputs and learn a meaningful node or graph representation. Besides, as pointed out in [22], it is important to simultaneously capture the spatial and temporal information of monitored signal in PHM. Thus, in order to capture the spatial and temporal information of graphs for RUL prediction, a novel prognostic method named hierarchical attention graph convolutional network (HAGCN) is proposed with the goal to model the spatial-temporal graphs and achieve more accurate RUL predictions for machinery.

In HAGCN, the bi-directional LTSM (BiLSTM) layer is used to model the temporal dependencies of node features. The hierarchical graph representation layer (HGRL), formed by stacking graph isomorphism convolution layer (GIN) and the proposed regularized self-attention graph pooling layer (RSAGPool), is used to fuse the multi-sensor information and model the spatial dependencies of graphs. Extensive experiments are implemented to validate effectiveness of HAGCN for RUL prediction, and thirteen existing methods are selected for comparison. The experimental results show that HAGCN can achieve state-of-the-art results among all methods. The major contributions of this work are as follows:

- A novel multi-sensor features fusion approach for RUL prediction is proposed, which boosts model performance by simultaneously modeling the spatial and temporal dependencies of sensors.
- Multiple sensors are constructed to a sensor network to generate spatial-temporal graphs, and HAGCN consists of BiLSTM layer and HGRL layer is proposed to model the generated graphs.
- RSAGPool layer in HGRL is proposed to learn a more promising graph representations from spatial-temporal graphs, and the sensor network is simplified based on the learned attention scores.

The remainder of this paper is organized as follows. The preliminary is shown in Section 2, and application of HAGCN for RUL prediction is described in Section 3. After that, the effectiveness of HAGCN on RUL prediction is validated by case studies in Section 4, and model analysis is shown in Section 5. Finally, summary of this paper is shown in Section 6.

## 2 Preliminary

### 2.1 Problem definition of RUL prediction with GCN

Consider a graph $G = (V, E, X)$, where $E$ is the edge set, and $|V| = n$ represents the vertex set. Each node $v_i \in V$ with $d$-dimensional features is denoted by $x_i$. $X \in \mathbb{R}^{n\times d}$ is the node feature matrix, and $A \in \mathbb{R}^{n\times n}$

represents the adjacency matrix.

For RUL prediction, the feature matrix $X$ of graph $G$ changes in real-time with time step $t$. Therefore, we can construct the spatial-temporal graph dataset from the multiple sensor measurements $\{x_1, x_2, \ldots, x_n\}$, and assign the RUL at time step $t$ as the label for each graph. With the constructed dataset $D=\{(G_1, y_1),(G_2, y_2),\ldots\}$, the task of RUL prediction is to learn a mapping $f: G \to y$.

## 2.2 Temporal dependency modeling

The BiLSTM has been broadly utilized to model temporal dependencies of time series. BiLSTM consists of the forward LSTM and the backward LSTM [27], where LSTM is developed from the traditional RNN with the goal to solve long-term dependence problem. The LSTM cell, as shown in Fig. 2, is composed of an input gate, a forget gate, a memory cell and an output gate. By passing through the four gates in turn, the temporal dependencies of time series can be captured.

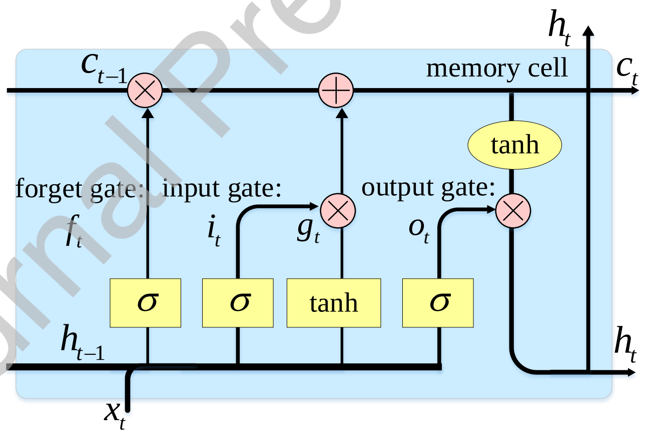


Fig. 2. LSTM cell [27].

## 2.3 Spatial dependency modeling

By stacking multiple graph convolutional (GConv) layers and graph pooling layers, it is able to realize the hierarchical representations and model the spatial dependencies of graphs.

### 2.3.1 Graph isomorphism convolutional layer

The general node aggregation based GCNs [28] share three key operations: 1) $\text{AGGREGATE}(\cdot)$ operation, which is used for node feature aggregation; 2) $\text{COMBINE}(\cdot)$ operation, which is used for node feature update; 3) $\mathrm{READOUT}(\cdot)$ operation, which is used to obtain the graph representation for the graph classification task.

For example, for a graph *G*, its feature representations after *k* iterations of aggregation are:

$$h_v^{(k)} = \mathrm{COMBINE}^{(k)}\left(\mathrm{AGGREGATE}^{(k)}\left(\{h_u^{(k-1)} : u \in \mathbb{N}(v)\}\right), a_v^{(k)}\right) \tag{1}$$

$$h_G = \mathrm{READOUT}(h_v^{(k)} \mid v \in G) \tag{2}$$

where $a_v^{(k)}$ is the aggregated adjacent node features of node *v*. $h_v^{(k)}$ is the updated features representations of graph *G* after *k*-th iteration. $h_v^{(0)} = X_v$, and $\mathbb{N}(v)$ represents the adjacent nodes of the node *v.* The mean or max operator is usually used as the aggregation function, and $h_G$ denotes the graph representations.

However, the node aggregation based GCNs fail to learn graph structure information. In order to address this issue, graph isomorphism network (GIN) [29] uses multi-layer perceptron (MLP) to model and learn the node representation of the node *v*. The graph isomorphism convolution (GINConv) can be denoted as:

$$h_v^{(k+1)} = \mathrm{MLP}^{(k)}\left(\left(1+\vartheta^{(k)}\right)\cdot h_v^{(k-1)} + \sum\nolimits_{u\in \mathbb{N}(v)} h_u^{(k-1)}\right) \tag{3}$$

where $\vartheta > 0$ is the hyper-parameter of GINConv layer.

### 2.3.2 Self-attention graph pooling

In order to find out the meaningful node assignment from the graph topology, the Ref. [30] proposed a self-attention graph pooling (SAGPool). In SAGPool, the graph convolution is used to calculate the self-attention score for each node and obtain the attention mask, which can be denoted as:

$$\begin{cases} Q = \mathrm{GConv}(X, \Theta_{att}) \\ idx \ = \text{top-rank}(Q, kn) \\ Q_{mask} = Q_{idx} \end{cases} \tag{4}$$

where $Q \in \mathbb{R}^{n\times 1}$ is the calculated self-attention score. $\mathrm{GConv}(\cdot)$ represents the graph convolution operation. $X \in \mathbb{R}^{n\times d}$ represents the aforementioned node features, and $\Theta_{att} \in \mathbb{R}^{d\times 1}$ is the only parameter of the SAGPool layer. where $k \in (0,1]$ is the pooling ratio, $\text{top-rank}(\cdot)$ is the function returns the indices of the top $[kn]$ values, and *idx* is the returned node index. $Q_{idx}$ with *idx* returns the feature attention mask

$Q_{mask}$. Afterwards, the input graph is processed with the obtained feature attention mask, which can be denoted as:

$$\begin{cases} X' = X_{idx,:} \\ X_{out} = X' \odot Q_{mask} \\ A_{out} = A_{idx,idx} \end{cases} \tag{5}$$

where $X_{idx,:}$ represents the row-wise indexed feature matrix, $\odot$ is the elementwise product. $A_{idx,\ idx}$ is the adjacency matrix indexed by row and column. $X_{out}$ and $A_{out}$ are the new feature matrix and the corresponding adjacency matrix, respectively.

## 3 The proposed HAGCN for RUL prediction

### 3.1 Regularized self-attention graph pooling layer

Nevertheless, as the complexity of the input graphs increases, it is hard for the vallina SAGPool to learn a proper attention score for each node, which leads to the decrement of model performance and model stability. To overcome these issues, we propose a regularized self-attention graph pooling (RSAGPool) layer, as shown in Fig. 3. In RSAGPool, the vallina SAPool is regularized by applying self-attention mechanism for each input graph to obtain the prior self-attention scores, then calculate the KL-divergence error between the prior scores and the learned attention scores as a regular term. This regular term makes the model focus more on the nodes with larger initial scores in the learning process, thereby reducing the impact of the increase in graph complexity and improving model stability.

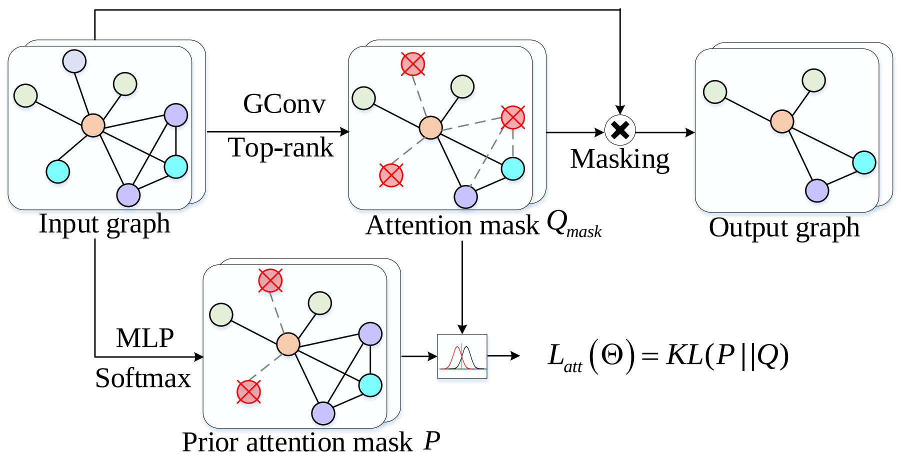


Fig. 3. The process of RSAGPool.

In RSAGPool, the prior self-attention scores are calculated by:

$$\begin{cases} \kappa_j = \mathrm{MLP}\left(X_j\right) \\ P_j = \mathrm{soft\,max}\left(\kappa_j\right) = \dfrac{\exp(\kappa_j)}{\sum_j \exp(\kappa_j)} \end{cases} \tag{6}$$

where $P_j$ is the prior distribution of the self-attention scores of all nodes in graph $G_j$, and $X_j$ is the node feature matrix of $G_j$. $\kappa_j$ is the node-wise feature vector.

After that, the self-attention error $L_{att}(\Theta)$ between the self-attention scores learned by SAGPool and prior self-attention scores are measured by KL-divergence, which can be denoted as:

$$L_{att}\left(\Theta\right) = KL(P\,||\,Q) = \sum_{j=1}^{m} P_j \log \frac{P_j}{Q_j} \tag{7}$$

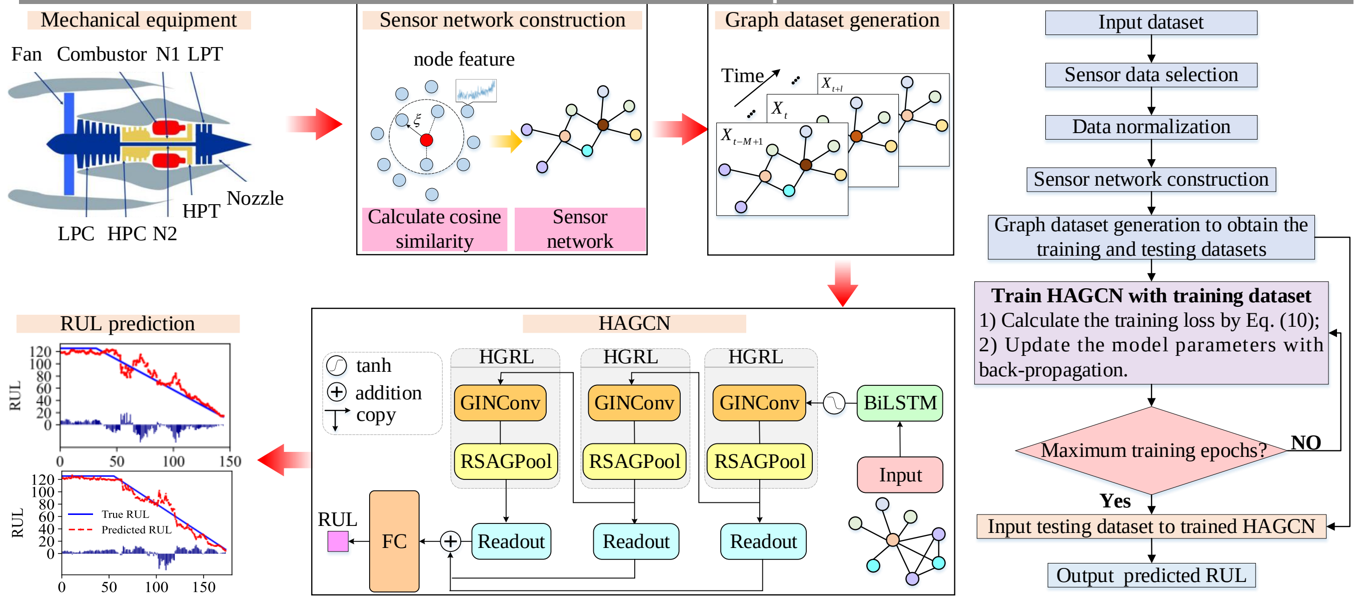
where $Q_j$ is the learned self-attention scores of all nodes in graph $G_j$.

Therefore, the input graph processed by RSAGPool can be denoted as:

$$\begin{cases} X' = X_{idx,:} \\ X_{out} = X' \odot Q_{mask}\left(\Theta\right) \\ A_{out} = A_{idx,idx} \end{cases} \\ \text{s.t. } L_{att}\left(\Theta\right) = KL(P\,||\,Q) \tag{8}$$

### 3.2 Framework of HAGCN for RUL prediction

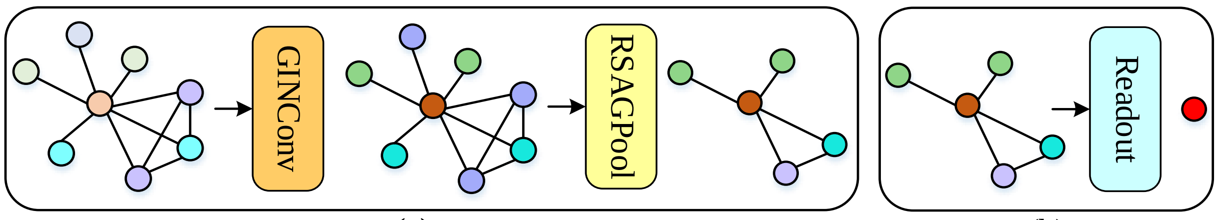
Framework of proposed hierarchical attention graph convolutional network (HAGCN) and the corresponding flow chart for realizing RUL prediction is shown in Fig. 4. As can be seen from Fig. 4(a), the proposed HAGCN contains three main components, including the BiLSTM layer, hierarchical graph representation layer (HGRL) and graph readout operation. The input graphs are first inputted into the BiLSTM to capture the temporal characteristics hidden in the node features. Then the processed graphs are transferred to the stacked three HGRLs, which are formed by GINConv layer and the proposed RSAGPool, to capture the spatial features from the graph structure.


Mechanical equipment
Fan Combustor N1 LPT
LPC HPC N2
HPT Nozzle
Sensor network construction
node feature
Calculate cosine similarity
Sensor network
Graph dataset generation
Time
RUL prediction
True RUL
Predicted RUL
HAGCN
HGRL
GINConv
RSAGPool
Readout
tanh
addition
copy
BiLSTM
Input
RUL
FC
Input dataset
Sensor data selection
Data normalization
Sensor network construction
Graph dataset generation to obtain the training and testing datasets
Train HAGCN with training dataset
1) Calculate the training loss by Eq. (10);
2) Update the model parameters with back-propagation.
Maximum training epochs?
NO
Yes
Input testing dataset to trained HAGCN
Output predicted RUL

(a) Framework of HAGCN (b) Flow chart

Fig. 4. The framework of HAGCN and the corresponding flow chart for realizing RUL prediction.

In addition, based on the idea of skip connection, every HGRL is followed by a graph readout layer to achieve the graph representations at different stages, and the learned three graph representations are fused as the final graph representation. Finally, the learned graph representations are passed to the two stacked fully connected (FC) layers for RUL prediction. Besides, the detailed process of HGRL and graph readout operation is shown in Fig. 5. In HGRL, as shown in Fig. 5(a), the input graph first passes through GINConv layer for message passing and information aggregation, and then the learned graph passes via RSAGPool layer for graph pooling. The result of RSAGPool is then passed to readout layer to obtain representation of the entire graph, as shown in Fig. 5(b).



(a) (b)

Fig. 5. The detailed process of HGRL and graph readout operation. (a) HGRL; (b) graph readout operation.

In model optimization process, our loss function contains two parts, where the mean square error (MSE) loss is utilized to estimate the error between the predicted RUL with the ground-truth RUL and the self-attention error $L_{att}(\Theta)$ is used as the regular term. The MSE loss can be denoted as:

$$L_{MSE}(\omega)=\frac{1}{m}\sum_{i=1}^{m}\left\|\hat{y}_i-y_i\right\|^2 \tag{9}$$

where $\omega$ is model parameter, and *m* is number of graphs. $y_i$ is the ground-truth RUL, and $\hat{y}_i$ is the predicted RUL.

Therefore, the final objective function is to minimize:

$$L_{total}=L_{MSE}(\omega)+\alpha L_{att}(\Theta) \tag{10}$$

where $\alpha$ is the hyper-parameter and is used to balance the value of MSE loss and self-attention error.

The algorithm for applying HAGCN to RUL prediction is summarized in Algorithm 1 and the flow chart is shown in Fig. 4(b). As shown in the flow chart, the sensor network is firstly constructed by calculating the cosine similarity between selected sensors. After that, the spatial-temporal graph dataset is generated by assigning sensor measurements at different time stamps to the sensor network. Then, the proposed HACGN is trained with the training dataset and the model parameters are updated through back-propagation. Finally, the predicted RUL of the testing dataset is obtained by inputting the testing dataset into the trained HAGCN.

```
Algorithm 1: HAGCN for RUL prediction
Input: Spatial-temporal graph dataset $D_{train}$, $D_{test}$.
Output: The RUL of testing graph dataset.
1.  Model training:
2.      for $X$ in $D_{train}$ do
3.          $H^1 \leftarrow \text{LSTM}(X)$;
4.          $H^2 \leftarrow \text{HGRL}_1(H^1)$, $Z^1 \leftarrow \text{Readout}_1(H^1)$;
5.          $H^3 \leftarrow \text{HGRL}_2(H^2)$, $Z^2 \leftarrow \text{Readout}_2(H^2)$;
6.          $H^4 \leftarrow \text{HGRL}_3(H^3)$, $Z^3 \leftarrow \text{Readout}_3(H^3)$;
7.          $\text{RUL}_{train} \leftarrow \text{FC}(Z^1+Z^2+Z^3)$;
8.          Update HAGCN parameter based on $L_{total}$.
9.      end for
```

```
10. Model testing:
11.         $\mathrm{RUL}_{test} \leftarrow \mathrm{HAGCN}(X)$.
```

## 4 Case study

In order to verify the validity of the proposed HAGCN, case studies are carried out in this section. The HAGCN is written in Python 3.7 with Pytorch 1.5 machine learning framework. Besides, the case studies are conducted on a computer with an Intel i7-9700K CPU and a GTX2080Ti GPU.

### 4.1 Case study I: RUL prediction of CMAPSS dataset

#### 4.1.1 Data description of CMAPSS dataset

CMAPSS (Commercial Modular Aero-Propulsion System Simulation) dataset provided by NASA data-repository [18] is utilized for experimental validation. This dataset is composed of four sub-datasets with different working conditions, as shown in Table 1. Each sub-dataset consists of a training set, and a testing set labeled with the real RUL. The engine units (EUs) contained in the training dataset start with a different initial state. 21 sensors, as shown in Table 2, are used to collect the run-to-failure multivariate temporal data of each engine unit, which indicates the last record time cycle is the time that the EUs reach system failure. The testing dataset only contains sensor records for a certain number of engines in a specific operating cycle.

Table 1
The details of CMAPSS dataset

| Dataset | FD001 | FD002 | FD003 | FD004 |
|---|---|---|---|---|
| Training EU | 100 | 260 | 100 | 249 |
| Testing EU | 100 | 259 | 100 | 248 |
| Operating conditions | 1 | 6 | 1 | 6 |
| Fault types | 1 | 1 | 2 | 2 |
| Maximum life cycle | 362 | 378 | 525 | 543 |
| Minimum life cycle | 128 | 128 | 145 | 128 |

#### 4.1.2 Data preprocessing

There are measurements from 21 sensors in each dataset, however, the recorded values of some sensors are constant over time, which do not provide meaningful information for RUL prediction. Hence, 14 sensors are selected, and their measurements are utilized as the raw features [15], and the indices of the selected sensors are 2, 3, 4, 7, 8, 9, 11, 12, 13, 14, 15, 17, 20 and 21.

For each sub-dataset, the recorded multivariate temporal data are normalized to the range of [0, 1]. Since FD002 and FD004 contain multiple operating conditions, the above two datasets are first clustered according to the working condition by K-means [16] and then the corresponding normalization is performed to each cluster. Therefore, the Z-score normalization for this dataset can be unified as the following equation:

$$x_{norm}^{i} = \frac{x^{i,c} - x_{\text{mean}}^{i,c}}{\delta^{i,c}} \tag{11}$$

where $x_{norm}^{i}$ stands for the normalized value of *i*-th sensor. $x^{i,c}$ represents the data of *i*-th sensor collected at *c*-th working condition. $x_{\text{mean}}^{i,c}$ and $\delta^{i,c}$ are the mean and variance of the original measurements of *i*-th sensor, respectively.

Table 2
Detailed description of sensor measurements

| Number | Symbol | Description | Units |
|---|---|---|---|
| 1 | T2 | Total temperature at fan inlet | °R |
| 2 | T24 | Total temperature at LPC outlet | °R |
| 3 | T30 | Total temperature at HPC outlet | °R |
| 4 | T50 | Total temperature at LPT outlet | °R |
| 5 | P2 | Pressure at fan inlet | psia |
| 6 | P15 | Total pressure in bypass-duct | psia |
| 7 | P30 | Total pressure at HPC outlet | psia |
| 8 | Nf | Physical fan speed | rpm |
| 9 | Nc | Physical core speed | rpm |
| 10 | epr | Engine pressure ratio (P50/P2) | -- |
| 11 | Ps30 | Static pressure at HPC outlet | psia |
| 12 | phi | Ratio of fuel flow to Ps30 | pps/psi |
| 13 | NRf | Corrected fan speed | rpm |
| 14 | NRc | Corrected core speed | rpm |
| 15 | BPR | Bypass Ratio | -- |
| 16 | farB | Burner fuel-air ratio | -- |
| 17 | htBleed | Bleed Enthalpy | -- |
| 18 | Nf_dmd | Demanded fan speed | rpm |
| 19 | PCNfR_dmd | Demanded corrected fan speed | rpm |
| 20 | W31 | HPT coolant bleed | lbm/s |
| 21 | W32 | LPT coolant bleed | lbm/s |

### 4.1.3 Constructing spatial-temporal graph datasets

To construct spatial-temporal graphs from the multivariate temporal data, we first need to find the nearest sensors of each sensor and thus construct the sensor network. For multivariate temporal dataset $\{x_1, x_2, \ldots, x_i\}$, each sensor is considered as a node, and to find the neighbors of each sensor, we can first select a sensor and then calculate the Euclidean distance with other sensors, which can be defined as:

$$dis(x_i, x_j) = \|x_i - x_j\|^2 = \|x_i\|^2 + \|x_j\|^2 - 2\|x_i\|\|x_j\|\cos(\theta) \tag{12}$$

where $dis(x_i, x_j)$ means the distance between node $x_i$ and $x_j$, and $\cos(\theta)$ is angle between the two node feature vectors. As mentioned before, the Z-score normalization is used, so the square of the norm of feature vector equal to 1, that is, $\|x_i\|^2 = \|x_j\|^2 = 1$. Thus, the distance can be simplified to

$$dis(x, y) = \|x_i\|^2 + \|x_j\|^2 - 2\|x_i\|\|x_j\|\cos(\theta) = 2\text{-}2\cos(\theta) \tag{13}$$

As can be observed in Eq. (13), the distance between two nodes only determines by the value of $\cos(\theta)$, and $\cos(\theta)$ is the cosine similarity. Therefore, by only calculating the cosine similarity between each sensor, the neighbors of node $x_i$ can be found, and the returned neighbors of node $x_i$ can be denoted as:

$$\mathbb{N}(x_i) = \begin{cases} adj(x_i, x_j), \text{if } \cos(\theta) > 0 \\ 0, \textit{otherwise} \end{cases} \tag{14}$$

where $\mathbb{N}(x_i)$ is the adjacent nodes of node $x_i$, and $adj(x_i, x_j)$ returns the set $\{x_j \mid \cos(\theta) > 0\}, j \le i$. In neighborhood construction, we try to find the nodes with smaller distance to $x_i$ as its neighbors. To make sure the nearest neighbors of a node are found, we let $\cos(\theta) > 0$. The constructed sensor networks of four sub-datasets and their corresponding graph densities [31] are shown in Fig. 6.

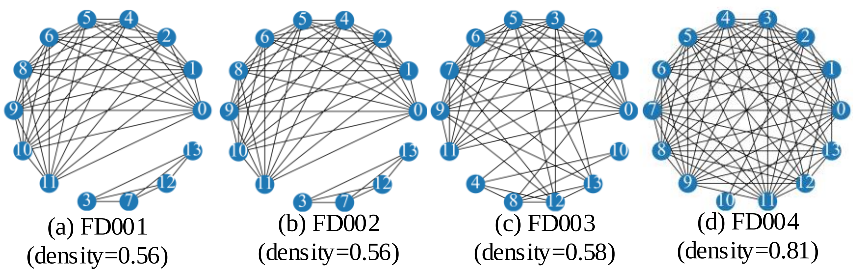


Fig. 6. The constructed sensor network of each dataset.

As shown in Fig. 6, the node label of each graph is the order of the selected sensors after renumbering. It can also be observed that FD001 has the minimal graph density and FD004 has the maximal graph density, which means the graph complexity of FD004 is the largest and the simplest one is FD001.

After obtaining the sensor network, we can generate the spatial-temporal graph datasets by assigning time series collected at different time stamps to the sensor network as the node features. The corresponding RUL label can also be acquired by subtracting the current time stamp from the total working cycles of the engines, as demonstrated in Fig. 7.

In order to obtain the sub-time series collected at different time stamps, we then split the overall running cycles. During the process of data splitting, the collected sensor measurements are truncated by using a sliding window without overlap. In this paper, the sensor measurements are truncated by a sliding window of length 30, and we adopt the piece-wise linear degradation model [32] to calculate the RUL labels, where the maximum RUL of each engine is 130.

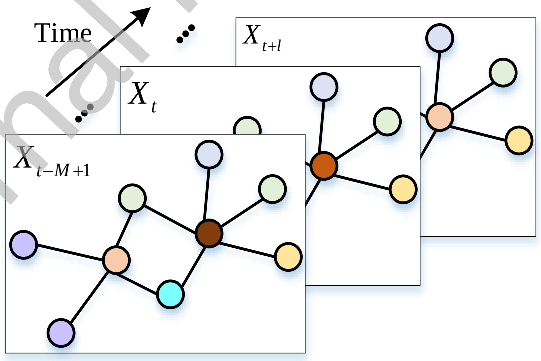


Fig. 7. Example of generating spatial-temporal graph datasets.

### 4.1.4 Evaluation Metrics

To make fair comparisons with other previous models, the same two evaluations metrics, namely root mean square error (RMSE) and scoring function (SF), are used in this experiment to estimate the efficacy of the proposed method. The RMSE, as defined in Eq. (15), is widely used in regression task, and is defined as the error between the predicted RUL and the ground-truth RUL. SF is utilized to measure the ability of late prediction (e.g., the predicted RUL is longer than the actual RUL) of networks, which cannot be revealed in RMSE.

$$\text{RMSE=}\sqrt{\frac{1}{M}\sum_{i=1}^{M}(y_i\text{-}y_i)^2} \tag{15}$$

where $M$ is the total number of graphs.

The SF is defined as follows:

$$\text{SF}=\sum_{i=1}^{M}\text{SF}_i,\ \text{with}\ \text{SF}_i=\begin{cases} e^{-\frac{\delta_i}{13}}-1, \delta_i<0 \\ e^{\frac{\delta_i}{13}}-1, \delta_i\geq 0 \end{cases} \tag{16}$$

where SF is the calculated scores. $\delta_i=\hat{y}_i\text{-}y_i$ is the error between predicted RUL and the ground-truth RUL.

### 4.1.5 Experimental results

In order to show the efficacy of the proposed methods, comparison experiments are carried out by comparing HAGCN with other 13 published studies. These methods can be divided into five categories, including Bayesian based models [33], CNN based methods [14], [34], LTSM based methods [16], [17], [35-37], DBN based methods [38], and hybrid methods [19, 20, 21, 39].

Table 3
The detailed hyperparameters

| Hyperparameter | Value |
|---|---|
| Initial learning rate | 0.01 |
| Batch size | 256 |
| Training epoch | 180 |
| Threshold of Top-rank | 0.001 |
| Balance value $\alpha$ | 100 |

Table 4
The detailed structure of HAGCN

| Number of Components | HAGCN | Output feature size |
|---|---|---|
| 1 | Input | 14×30×256 |
| 2 | BiLSTM | 14×60×256 |
| 3 | HGRL_1 | 14×64×256 |
| 4 | Readout | 64×256 |
| 5 | HGRL_2 | 14×64×256 |
| 6 | Readout | 64×256 |
| 7 | HGRL_3 | 14×64×256 |

| | | |
|---|---|---|
| 8 | Readout | 64×256 |
| 9 | FC | 32×256 |
| 10 | FC | 1x256 |

In the process of model training, the hyperparameters of the model are set as shown in Table 3, where the Adam is used as the optimizer with initial learning rate 0.001, the batch size is 256, the threshold of Top-rank set to 0.001 and the balance value $\alpha$ of is set to 100. The model is trained for 180 epochs, and learning rate decay is adopted, which decreases the learning rate by multiplying 0.1 in epochs 120 and 160. According to the data preprocessing and hyperparameter settings, we can know that the size of input is $14\times30\times256$ (i.e., number of nodes×feature length×batch size). Based on this, we have listed the detailed structure of HAGCN and its output feature size of each layer in Table 4. In the experiments, we test the model at per epoch, and to avoid randomness of the results, the final values of the two evaluation metrics are the average of the last 10 epochs. The experimental results of HAGCN and its standard deviation (STD) are shown in Table 5.

Table 5
The experimental results of CMAPSS testing dataset

| Method | RMSE | | | | SF | | | |
|---|---|---|---|---|---|---|---|---|
| | FD001 | FD002 | FD003 | FD004 | FD001 | FD002 | FD003 | FD004 |
| BDNN [33] | 12.19 | 18.49 | 12.07 | 19.41 | 267.2 | 2007.8 | 409.4 | 2415.7 |
| DCNN-1 [14] | 18.45 | 30.29 | 19.82 | 29.16 | 1286.7 | 13570 | 1596.2 | 7886.4 |
| DCNN-2 [34] | 12.61 | 28.51 | 12.62 | 30.73 | N/A | N/A | N/A | N/A |
| LTSM-1 [16] | 12.29 | 17.87 | 14.34 | 21.81 | N/A | N/A | N/A | N/A |
| LSTM-2 [17] | N/A | 25.11 | N/A | 26.61 | N/A | 4793 | N/A | 4971 |
| LSTM-3 [35] | 14.53 | N/A | N/A | 27.08 | 322.4 | N/A | N/A | 5649.1 |
| LSTM-4 [36] | 16.74 | 29.43 | 18.07 | 28.40 | 388.7 | 10654 | 822.19 | 6370.6 |
| LSTM-5 [37] | 14.89 | 26.86 | 15.11 | 27.11 | 481 | 7982 | 493 | 5200 |
| MODBNE [38] | 15.04 | 25.05 | 12.51 | 28.66 | 334.2 | 5585.3 | 421.9 | 6557.6 |
| HDNN [19] | 13.02 | 15.24 | 12.22 | 18.16 | 245 | 1282.4 | 287.7 | 1527.4 |
| CNN-LSTM [20] | 14.40 | 27.23 | 14.32 | 26.69 | 290 | 9869 | 316 | 6594 |
| BLCNN [21] | 13.18 | 19.09 | 13.75 | 20.97 | 302.3 | 1557.6 | 381.4 | 3858.8 |
| RBM-LSTM [39] | 12.56 | 22.73 | 12.10 | 22.66 | 231 | 3366 | 251 | 2840 |
| **HAGCN (STD)** | **11.93 (0.09)** | **15.05 (0.03)** | **11.53 (0.04)** | **15.74 (0.02)** | **222.3 (7.61)** | **1144.1 (105)** | **240.3 (7.18)** | **1218.6 (27.2)** |
| **IMP** | **2.13%** | **15.78%** | **4.47%** | **13.33%** | **3.77%** | **10.78%** | **4.26%** | **20.22%** |

* N/A means the values is not provided.

From Table 5, we can see that HAGCN can achieve the sate-of-the-art results among all other methods. To show how much improvement that HAGCN can achieve on each dataset, the improvement indicator

(IMP) is calculated by Eq. (17).

$$\mathrm{IMP} = 1 - \frac{\mathrm{HAGCN}}{\mathrm{SOTA}} \tag{17}$$

The values of IMP indicator are shown in the last raw of Table 5. From these results, we can see that HAGCN makes improvements on all datasets. Especially, compared with the existing SOTA on FD004, HAGCN has showed significant improvements of 13.33% and 20.22% on RMSE and SF respectively. The achievements on FD004 show the superiority of HAGCN over other competing methods and demonstrate that HAGCN has the ability to learn temporal and spatial features from the complex network, which helps to handle the problem of late prediction.

Furthermore, four examples of the predicted RUL of each dataset are shown in Fig. 8, and comparison of ground truth RUL and the predicted RUL at the failure point of the entire test EUs is shown in Fig. 9. As it can be seen from Fig. 8, at early stage, the estimated RUL values are very close to the preset maximum RUL. After that, the closer the testing sample is to the failure point, the closer the estimated values are to linear degradation over time. Despite there are remarkable error between the estimate RUL and the ground-truth RUL, the proposed HAGCN can still achieve a high prognostic accuracy especially when the test EUs are close to failure. Besides, we can also observe in Fig. 9 that the predicted RUL values at the failure point of the entire test EUs match the ground-truth RUL very well, which is consistent with the results shown in Table V and indicates the feasibility of HAGCN on RUL prediction.

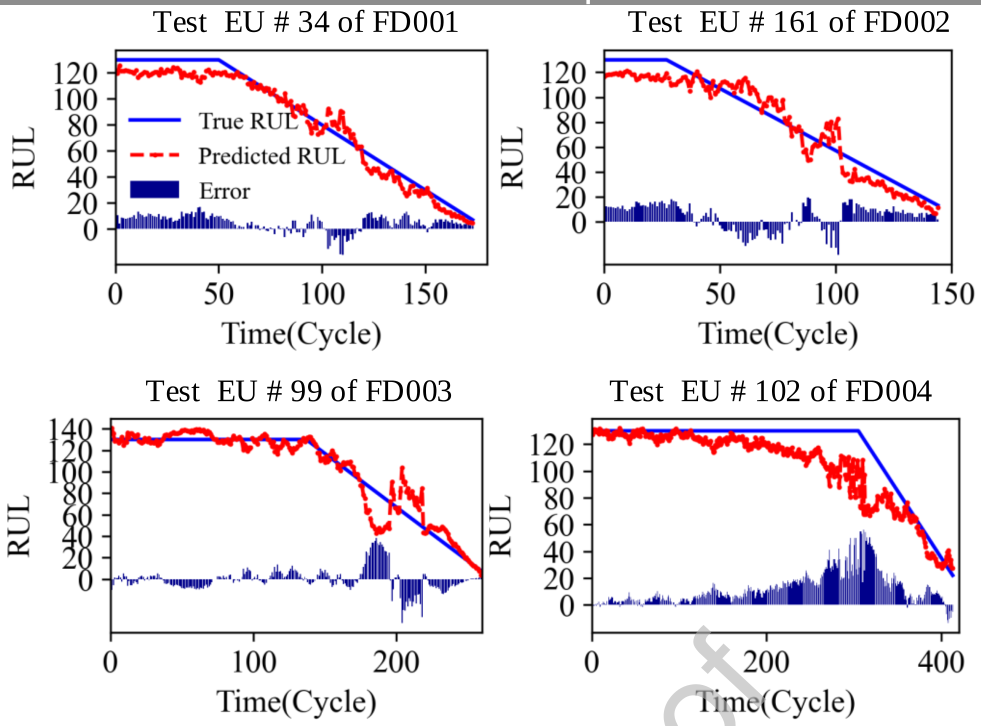


Fig. 8. Example for RUL prediction for the test EU on the four datasets.

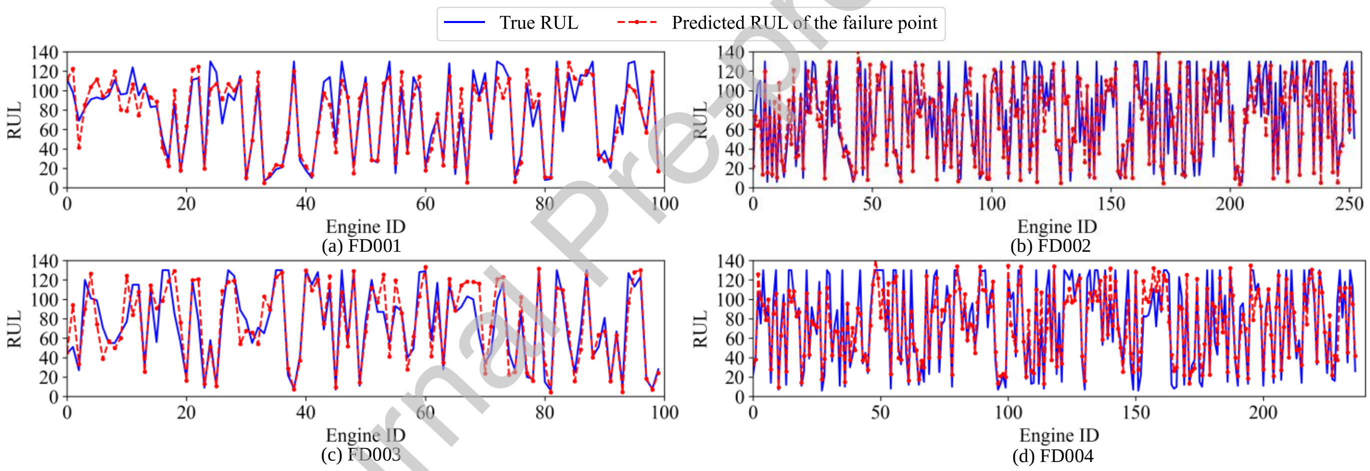


Fig. 9. The predicted RUL and ground-truth RUL of the failure point of the entire test EU.

## 4.2 Case study II: RUL prediction of milling cutters

### 4.2.1 Data Description

The tool wear dataset of milling cutters is collected during the operation of a high-speed CNC machine, which is presented by the PHM Society [40]. It is always used as a baseline dataset for tool wear prediction [41]. As shown in Fig. 10, the CNC machine with spindle speeds up to 42, 000 rpm is used as the experiment platform. During the experiments, six cutters are utilized independently to cut over the stainless-steel workpieces, and the cutters' flank wear are recorded as the label. Therefore, the tool wear dataset contains six different sub-datasets, i.e., ($C_1$, $C_2$, …, $C_6$), each sub-dataset records 315 samples, corresponding to 315 tool wear, and each sub-dataset includes data collected by 7 sensors, which are the milling force and vibration signals in the X, Y, and Z directions, and the RMS value of the acoustic emission signal respectively. Here, three sub-datasets, i.e., $C_1$, $C_4$, and $C_6$, are used in this case study.

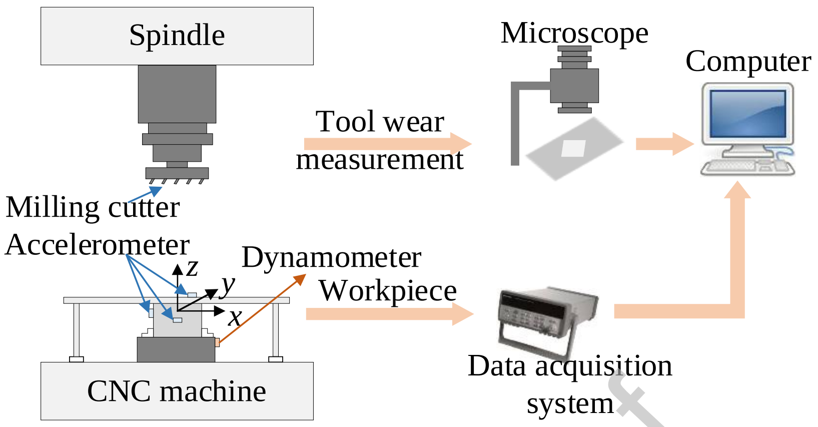


Fig. 10. The CNC machine and tool condition monitoring process.

### 4.2.2 Data preprocessing and sensor network construction

For a sample in the sub-dataset, we calculate an RMS value every 4245 points, and extract a total of 30 RMS values as its statistical features. When each sub-sample in the sub-dataset is calculated, the maximum-minimum normalization is used to normalize the calculated RMS value. After that, the sensor network of each sub-dataset can be constructed according to Eq. (14), and the constructed sensor networks are shown in Fig. 11. It is found that the sensor networks of the three sub-datasets are the same, and this may be caused by the same operating environment of the milling cutters.

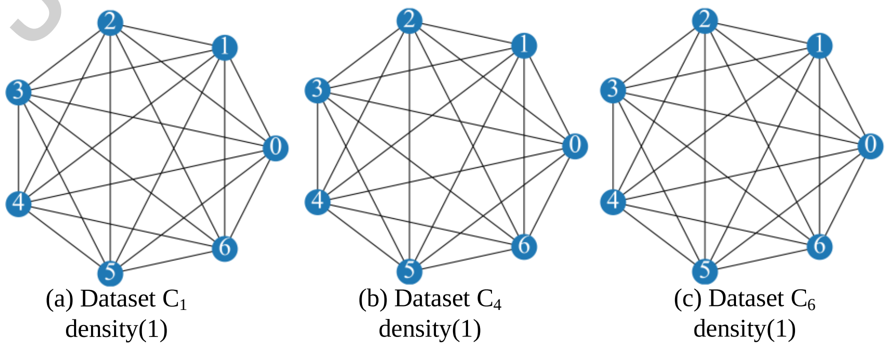


Fig. 11. The constructed sensor network of three sub-datasets.

Since the given label of the sub-dataset is the amount of cutters flank wear in the X, Y, and Z directions. In order to perform a reliable and accurate RUL prediction, the average value of the cutters flank wear of the three flutes is considered to define the failure threshold. Therefore, the RUL of the milling cutters can be calculated by the following equation:

$$\mathrm{RUL}(t) = t_f - t \tag{18}$$

where $t$ is the current time and $\mathrm{RUL}(t)$ is the remaining life at this time. $t_f$ indicates the time when the tool wear reaches the failure threshold, and in this case study the failure threshold is set to 0.16 mm.

### 4.2.3 Experimental Results

The model setting and evaluation metrics are same with the above experiment. To make full use of the dataset, as shown in Table 6, three-fold strategy is taken, which means two of the three sub-datasets are utilized for model training and the other is used for model testing. The comparison experiments with five aforementioned approaches, i.e., DCNN-1, DCNN-2, LSTM-2, LSTM-4, and CNN-LSTM, are implemented to demonstrate the effectiveness of our method. The experimental results are shown in Table 7, and the predicted RUL of each cutter is shown in Fig. 12.

Table 6
Details of the three-fold strategy

| Training set | Testing set | Samples for training | Samples for testing |
|---|---|---|---|
| $C_1$+$C_4$ | $C_6$ | 306+278 | 238 |
| $C_1$+$C_6$ | $C_4$ | 306+238 | 278 |
| $C_4$+$C_6$ | $C_1$ | 278+238 | 306 |

Table 7
Experimental results of RUL prediction of milling cutters

| Method | RMSE | | | SF | | |
|---|---|---|---|---|---|---|
| | $C_6$ | $C_4$ | $C_1$ | $C_6$ | $C_4$ | $C_1$ |
| DCNN-1 | 41.1 | 24.8 | 37.4 | 361118.7 | 6147.2 | 11391.2 |
| DCNN-2 | 33.5 | 36.8 | 37.3 | 266151.1 | 33614.4 | 9673.2 |
| LSTM-1 | 30.2 | 14.5 | 25.9 | 12897.7 | 814.9 | 3064.7 |
| LSTM-2 | 33.7 | 12.8 | 27.7 | 19965.7 | 623.4 | 2980.7 |
| CNN-LSTM | 31.6 | 15.1 | 44.7 | 9673.1 | 829.7 | 46929.1 |
| **HAGCN (STD)** | **15.6 (0.11)** | **11.2 (0.03)** | **23.2 (0.14)** | **1162.8 (5.28)** | **589.6 (2.68)** | **2278.3 (44.5)** |

It can be observed from these results that the proposed HAGCN can achieve the smallest RMSE and SF among the five compared approaches. Besides, it can be found that the calculated RMSE and SF of dataset $C_1$ is greater than dataset $C_4$ and dataset $C_6$, and the predicted results of dataset $C_1$ are also worse than the other two datasets as shown in Fig. 12. This may be caused by fewer training samples are used to train the corresponding model when testing dataset $C_1$.

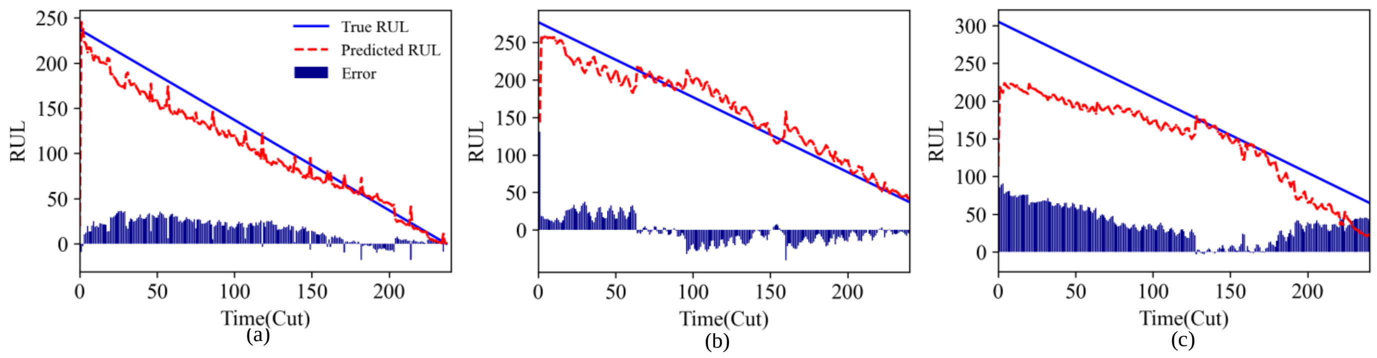


Fig. 12. The predicted cutters RUL of the three sub-datasets. (a) Dataset $C_6$; (b) Dataset $C_4$; (c) Dataset $C_1$.

It can be observed from the results of the above two experiments that the proposed HAGCN can achieve promising RUL prediction results. In practical applications, the appropriate maintenance timing and maintenance strategies can be arranged according to the predicted RUL. Consequently, the safety and reliability can be improved during the operation of machinery.

## 5 Model analysis

### 5.1 The influence of prediction interval

In order to explore the performance of the proposed model under different prediction intervals, five additional sets of experiments are added with sliding window lengths of 10, 20, 40, 50, and 60 on the four sub-datasets. The experimental results are shown in Fig. 13.

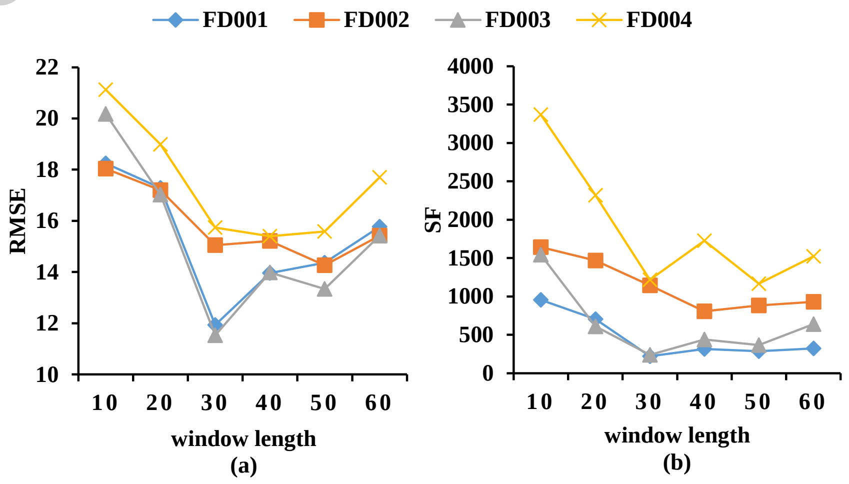


Fig. 13. The experimental results of HAGCN under different window lengths.

From these results, we can see the RMSE and SF of HAGCN on all datasets decline at the beginning and rising up along with the window length increases, due to more information is included in the node features for RUL prediction. However, when the window length is over 30, the performance of HAGCN on most datasets begin to degrade. This may be due to the increase of computational complexity caused by excessive data, which exceeds computational power in turn reduces the model performance.

## 5.2 Ablation study and the importance analysis of RSAGPool

In order to find out which part of HAGCN contributes most to the model results and verify the importance of RSAGPool, four variants are further derived for discussion, as shown in Table 8. Model I, II, and III are obtained by removing BiLSTM, GIN and RSAGPool from HAGCN, respectively. Model IV is obtained by changing RSAGPool to vallina SAGPool in HAGCN. It is worth noting that there are three readout operations in all these models and position of the operations are the same with HAGCN. The experimental set up is the same with the aforementioned experiments, and the results are shown in Fig. 14.

From these results, we can see that Model I without the ability to model the temporal dependencies of the sensor measurements achieves the worst performance in these four models. Model II without the ability to model the spatial dependencies of the multiple sensors performs worse than Model III without ability to achieve hierarchical learning. Among them, the proposed HAGCN with ability to model the spatial-temporal features achieves the best results in the four datasets, showing the superiority of the proposed HAGCN for RUL prediction. This also demonstrates that it is helpful to improve the predictive power by modeling temporal dependencies of node features and spatial dependencies between sensors at the same time.

Table 8
Model descriptions

| Model | Description |
|---|---|
| Model I | HAGCN without BiLSTM |
| Model II | HAGCN without GIN |

| | |
|---|---|
| Model III | HAGCN without RSAGPool |
| Model IV | HAGCN with vallina SAGPool |

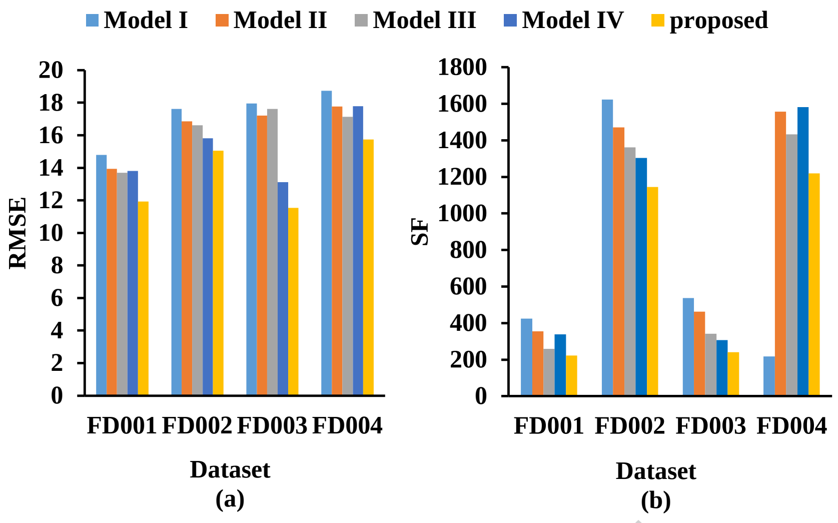


Fig. 14. The experimental results of ablation study.

Furthermore, it can be seen from the comparison of Model IV and HAGCN that the models with the vallina SAGPool achieve a bigger RMSE and SF than models with RSAGPool, which demonstrates that the proposed RSAGPool is conducive to increase prediction performance of the model.

## 5.3 The influence of the attention loss of RSAGPool

It has been demonstrated from the above experimental results that RSAGPool can improve the model performance. In order to find out the influence of the percentage of attention loss on the results of HAGCN, the value of hyperparameter $\alpha$ is increased from 0 to 1000. The experimental results are shown in Fig. 15.

From Fig. 15, we can see that the RMSE and SF of HAGCN on each dataset fluctuate sharply with the increase of alpha, especially when the training graphs come from a difficult dataset. This may be because the more complex the graph, the more difficult it is for the model to simultaneously focus on the importance of each node and learn the graph representation to make predictions. However, we can still observe that when the value of alpha is 100, HAGCN can achieve the best results on all datasets.

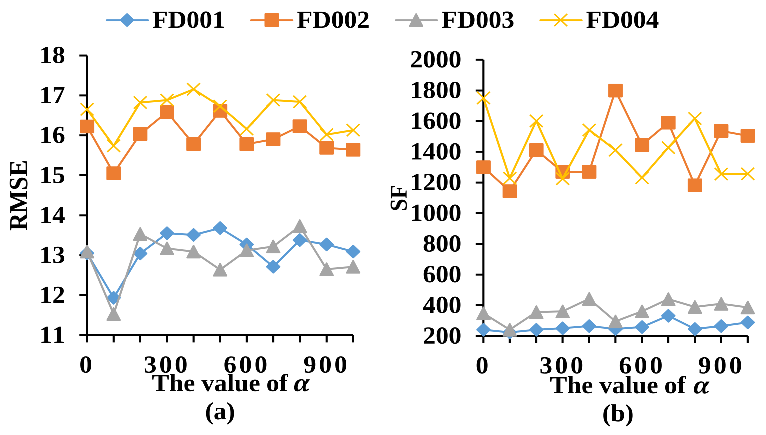
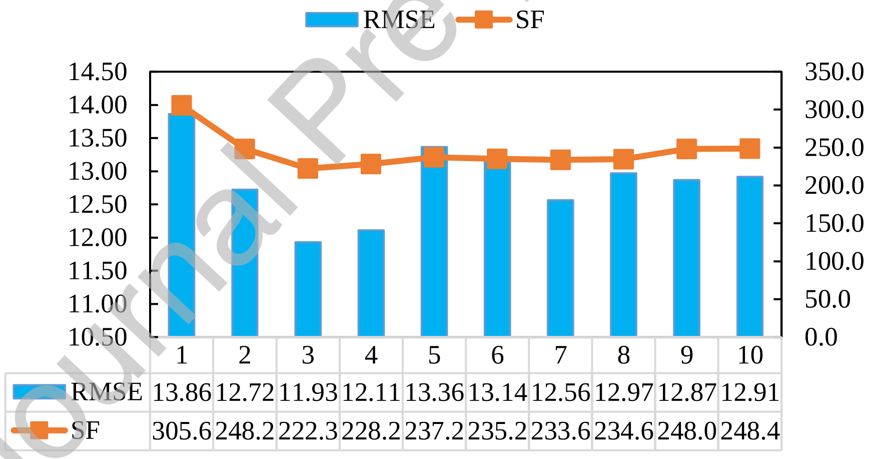


Fig. 15. The influence of percentage of attention loss to experimental results.

## 5.4 The influence of number of HGRLs

To explore the influence of number of HGRLs on model results, we increased the number of HGRLs in HAGCN from 1 to 10, and it is worth noting that each HGRL has the same number of parameters. The experiments are carried out on FD001 of CMPASS dataset, and its results are shown in Fig. 16.



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| RMSE | 13.86 | 12.72 | 11.93 | 12.11 | 13.36 | 13.14 | 12.56 | 12.97 | 12.87 | 12.91 |
| SF | 305.6 | 248.2 | 222.3 | 228.2 | 237.2 | 235.2 | 233.6 | 234.6 | 248.0 | 248.4 |

Fig. 16. The experimental results of FD001 under different number of HGRLs.

From Fig. 16, we can see with the increase of HGRLs, the RMSE and SF decrease first and then slightly increases, and the best performance of HAGCN can be achieved while the number of HGRLs take 3 or 4. This phenomenon indicates that HGRL does help to improve the experimental results, however, if the number of HGRL layers exceeds a certain number, the quality of the learned graph representation will not increase. Therefore, the number of HGRLs take 3 in this paper.

## 5.5 The graph topology learned by HAGCN

To illustrate which graph topologies have learned by HAGCN, a sample of EU#1 of the four test datasets is inputted into the trained model and the graph structures learned by the last layer of RSAGPool are visualized, and the finally learned graph structures are shown in Fig. 17.

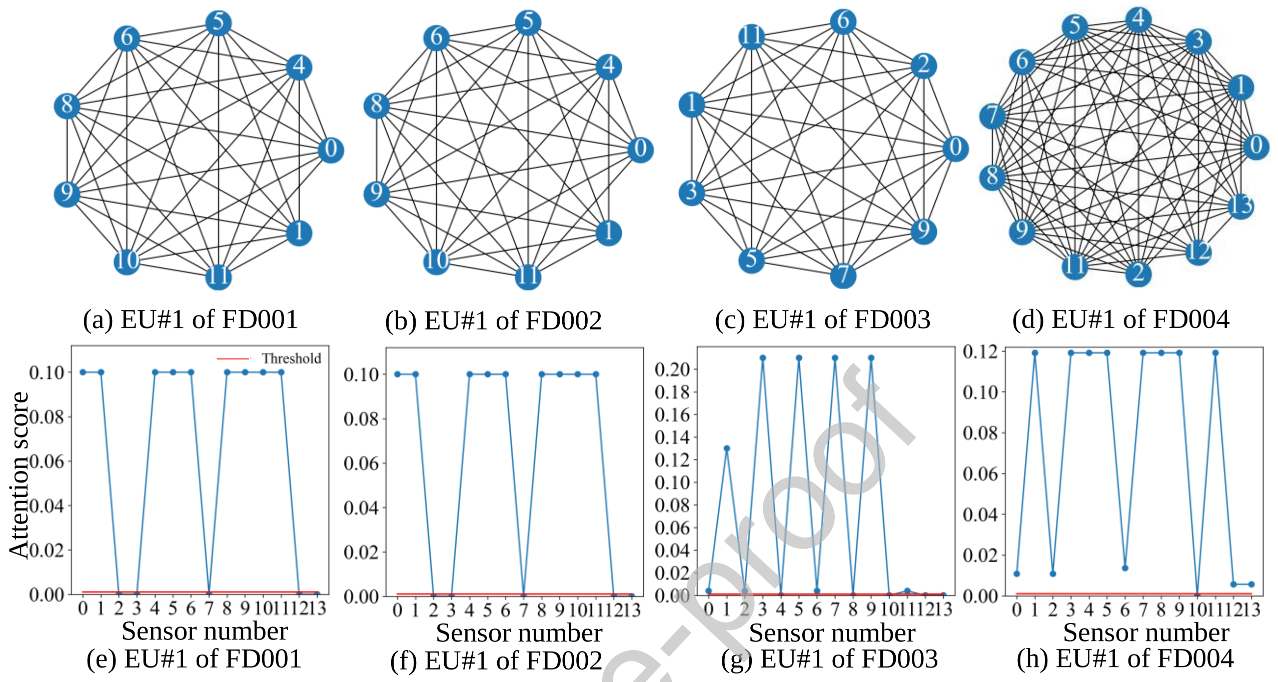


Fig. 17. The learned graph topologies and its attention scores. (a)~(d) The learned sensor network. (e)~(h) The attention score of each node in each sensor network.

It can be observed from Fig. 17(a)~(d) that the sensor network of dataset FD001, FD002, FD003 and FD004 retain 9, 9, 9, and 13 nodes respectively after hierarchical graph representation. This is because the nodes whose attention scores are less than the Top-rank threshold are removed, as shown in Fig. 17(e)~(h). The experimental results indicate that the sensor attention score obtained by RSAGPool can be used as an indicator to evaluate the importance of sensor, so that HAGCN has the ability to simplify the sensor network.

## 6 Conclusion

In this paper, we proposed a hierarchical attention graph convolutional network (HAGCN) based framework for accurately RUL prediction of machinery. Multiple sensors are constructed into a sensor network and spatial-temporal graphs are generated on this sensor network. Then, HAGCN is composed of BiLSTM layer and hierarchical graph representation layer (HGRL) is proposed to model the generated graphs. Moreover, the proposed regularized self-attention graph pooling layer (RSAGPool) in HGRL is utilized to learn a promising graph representation for RUL prediction.

Two case studies are carried out to verify the effectiveness of the proposed approach, and the experimental results show that the proposed method achieves the state-of-the-arts performance among the comparison methods. Since the prediction interval is important for RUL prediction, the influence of different window lengths on prediction results is explored. Then the ablation study is implemented to investigate the influence of each part of HAGCN and the effectiveness of the proposed RSAGPool layer. In addition, the influence of attention loss and the number of HGRLs on model performance are also discussed. Finally, the learned graph topologies are visualized, the results show that HAGCN has ability to simplify the sensor network by the learned attention scores, and this is useful for sensor position optimization in industrial scenarios.

Currently, the prior knowledge is not considered in the process of constructing sensor network, which will make the constructed network unable to accurately reflect the actual situation and lead to negative effects on the model performance. In the future work, we will design more suitable standards for judging whether there is a relationship between sensors, and calibrate the constructed network according to the actual situation, so as to embed the prior knowledge into the designed sensor network.

conflicts of interests

None

## Acknowledgments

This work was supported in part by the National Key Research and Development Program of China (No. 2018YFB1702400), Natural Science Foundation of China (No. 51835009), Shaanxi province 2020 natural science basic research plan (No. 2020JQ-042), National Key Science and Technology Infrastructure Opening Project Fund for Research and Evaluation facilities for Service Safety of Major Engineering

Materials, and Aeronautical Science Foundation (No. 2019ZB070001).

## Reference

[1] T. Li, Z. Zhao, C. Sun, et al. Domain Adversarial Graph Convolutional Network for Fault Diagnosis Under Variable Working Conditions. IEEE Transactions on Instrumentation and Measurement 2021; 70: 3515010.

[2] Z. He, H. Shao, X. Zhong, et al. Ensemble transfer CNNs driven by multi-channel signals for fault diagnosis of rotating machinery cross working conditions. Knowledge-Based Systems, 2020; 207: 106396.

[3] B. Zhang, S. Zhang, and W. Li. Bearing performance degradation assessment using long short-term memory recurrent network. Computers in Industry 2019; 106: 14-29.

[4] W. Zhang, X. Li, Z. Zhu, C. Shen, Q. He. Transfer learning using deep representation regularization in remaining useful life prediction across operating conditions. Reliability Engineering and System Safety 2021; 211: 107556.

[5] T. Li, Z. Zhao, C. Sun et al. Waveletkernelnet: An interpretable deep neural network for industrial intelligent diagnosis. IEEE Transactions on Systems, Man, and Cybernetics: Systems 2021. DOI: 10.1109/TSMC.2020.3048950.

[6] N. Li, G. Nagi, Y. Lei, et al. Remaining useful life prediction based on a multi-sensor data fusion model. Reliability Engineering and System Safety 2021; 208: 107249.

[7] J. Wang, Z. Li, G. Bai and M. J. Zuo. An improved model for dependent competing risks considering continuous degradation and random shocks. Reliability Engineering and System Safety 2020; 193: 106641.

[8] C. Bezerra Souto Maior, M. das Chagas Moura, and I. Didier Lins, et al. Remaining Useful Life Estimation by Empirical Mode Decomposition and Support Vector Machine. IEEE Latin America Transactions 2016; 14(11): 4603-4610.

[9] Z. Chen, Y. Li, T. Xia, and E. Pan. Hidden Markov model with auto-correlated observations for remaining useful life prediction and optimal maintenance policy. Reliability Engineering and System Safety 2019; 184: 123-136.

[10] C. Sun, M. Ma, and Z. Zhao, et al. Deep Transfer Learning Based on Sparse Auto-encoder for Remaining Useful Life Prediction of Tool in Manufacturing. IEEE Transactions on Industrial Informatics 2019; 15(4): 2416-2425.

[11] Z. Shi, A. Chehade. A dual-LSTM framework combining change point detection and remaining useful life prediction. Reliability Engineering and System Safety 2021; 205: 107257.

[12] M. Ma and Z. Mao. Deep Convolution-based LSTM Network for Remaining Useful Life Prediction", IEEE Transactions on Industrial Informatics 2020. DOI: 10.1109/TII.2020.2991796.

[13] L. Ren, Y. Sun, and J. Cui, et al. Bearing remaining useful life prediction based on deep autoencoder and deep neural networks. Journal of Manufacturing Systems 2018; 48: 71-77.

[14] G. Babu, P. Zhao, and X. Li. Deep Convolutional Neural Network Based Regression Approach for Estimation of Remaining Useful Life. International Conference on Database Systems for Advanced Applications 2016; 9642: 214-228.

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

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

[17] C. Huang, H. Huang and Y. Li. A Bidirectional LSTM Prognostics Method Under Multiple Operational Conditions. IEEE Transactions on Industrial Electronics 2019; 66(11): 8792-8802.

[18] A. Saxena, K. Goebel, D. Simon, and N. Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. International conference on prognostics and health management 2008: 1–9.

[19] A. Al-Dulaimi, S. Zabihi, A. Asif, et al. A multimodal and hybrid deep neural network model for Remaining Useful Life estimation. Computers in Industry 2019; 108: 186-196.

[20] Z. Wu, S. Yu, and X. Zhu, et al. A Weighted Deep Domain Adaptation Method for Industrial Fault Prognostics According to Prior Distribution of Complex Working Conditions. IEEE Access 2019; 7: pp. 139802-139814.

[21] H. Liu, Z. Liu, W. Jia, and X. Lin. A novel deep learning-based encoder-decoder model for remaining useful life prediction," International Joint Conference on Neural Networks (IJCNN) 2019. DOI: 10.1109/IJCNN.2019.8852129.

[22] S. Basak, S. Sengupta, S. J. Wen, et al. Spatio-temporal AI inference engine for estimating hard disk reliability. Pervasive and Mobile Computing, 2021, 70: 101283.

[23] Z. Wu, S. Pan, and F. Chen, et al. A Comprehensive Survey on Graph Neural Networks. IEEE Transactions on Neural Networks and Learning Systems 2020. DOI: 10.1109/TNNLS.2020.2978386.

[24] X. Li, X. Yan, Q. Gu, et al. DeepChemStable: Chemical stability prediction with an attention-based graph convolution network. Journal of chemical information and modeling. 2019, 59(3): 1044-1049.

[25] S. Guo, Y. Lin, N. Feng, et al. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. Proceedings of the AAAI Conference on Artificial Intelligence. 2019, 33(1): 922-929.

[26] T. Li, Z. Zhao, and C. Sun, et al. Multi-receptive Field Graph Convolutional Networks for Machine Fault Diagnosis. IEEE Transactions on Industrial Electronics 2020. DOI: 10.1109/TIE.2020.3040669.

[27] K. Greff, R. K. Srivastava, and J. Koutník, et al. LSTM: A Search Space Odyssey. IEEE Transactions on Neural Networks and Learning Systems 2017; 28(10): 2222-2232.

[28] W. L. Hamilton, R. Ying, and J. Leskovec. Inductive Representation Learning on Large Graphs. International Conference on Neural Information Processing Systems (NIPS) 2018.

[29] K. Xu, W. Hu, and J. Leskovec, et al. How Powerful are Graph Neural Networks. International Conference on Learning Representations (ICLR) 2019.

[30] J. Lee, I. Lee, and J. Kang. Self-Attention Graph Pooling. International Conference on Machine Learning (ICML), 2019.

[31] E. L. Lawler. Combinatorial optimization: networks and matroids. Courier Corporation, 2001.

[32] F. O. Heimes, Recurrent neural networks for remaining useful life estimation. International Conference on Prognostics and Health Management 2008: 1-6.

[33] M. Kim, and K. Liu. A Bayesian Deep Learning Framework for Interval Estimation of Remaining Useful Life in Complex Systems by Incorporating General Degradation Characteristics. IISE Transactions 2020: 1–23.

[34] X. u and Q. Wu, X. Li and B. Huang. Dilated Convolution Neural Network for Remaining Useful Life Prediction. Journal of Computing and Information Science in Engineering 2020; 20: 1-14.

[35] Z. Chen, M. Wu, and R. Zhao, et al. “Machine Remaining Useful Life Prediction via an Attention Based Deep Learning Approach. IEEE Transactions on Industrial Electronics 2020. DOI: 10.1109/TIE.2020.2972443.

[36] C. Hsu and J. Jiang. Remaining useful life estimation using long short-term memory deep learning. IEEE International Conference on Applied System Invention (ICASI) 2018: 58-61.

[37] Y. Liao, L. Zhang and C. Liu. Uncertainty Prediction of Remaining Useful Life Using Long Short-Term Memory Network Based on Bootstrap Method. IEEE International Conference on Prognostics and Health Management (ICPHM) 2018: 1-8.

[38] C. Zhang, P. Lim, A. K. Qin, et al. Multiobjective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics. IEEE Transactions on Neural Networks and Learning Systems 2017; 28(10): 2306-2318.

[39] A. Ellefsen, E. Bjørlykhaug, V. Æsøy, et al. Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. Reliability Engineering & System Safety 2018; 172: 1-11.

[40] The Prognostics and Health Management Society (PHM Society), https://www.phmsociety.org/competition/phm/10, 2010.1.

[41] J. Wang, P. Wang and R. Gao. Enhanced particle filter for tool wear prediction. Journal of Manufacturing Systems 2015; 36: 35-45.

**Tianfu Li,** received the B.S. degree in Mechanical Engineering from Chongqing University, China in 2018. He is currently working towards the Ph.D. degree in mechanical engineering in the Department of Mechanical Engineering, Xi'an Jiaotong University, Xi'an, China.

His current research is focused on deep learning, mechanical fault diagnosis and prognosis.

**Zhibin Zhao,** received the Ph.D. degree in Mechanical Engineering from Xi'an Jiaotong University, China in 2020. He is now an Assistant Professor in School of Mechanical Engineering at Xi'an Jiaotong University.

His current research is focused on sparse signal approximation, dictionary learning for machinery health monitoring.

**Chuang Sun**, received the Ph.D. degree in mechanical engineering from Xi'an Jiaotong University, Xi'an, China, in 2014. From Mar. 2015 to Mar. 2016, he was a postdoc at Case Western Reserve University, USA. He is now an Associate Professor in School of Mechanical Engineering at Xi'an Jiaotong University.

His research is focused on manifold learning, deep learning, sparse representation, mechanical fault diagnosis and prognosis, remaining useful life prediction.

**Ruqiang Yan** (M'07, SM'11) received the M.S. degree in precision instrument and machinery from the University of Science and Technology of China, Hefei, China, in 2002, and the Ph.D. degree in mechanical engineering from the University of Massachusetts at Amherst, Amherst, MA, USA, in 2007.

From 2009 to 2018, he was a Professor with the School of Instrument Science and Engineering, Southeast University, Nanjing, China. He joined the School of Mechanical Engineering, Xi'an Jiaotong University, Xi'an, China, in 2018. His research interests include data analytics, machine learning, and energy-efficient sensing and sensor networks for the condition monitoring and health diagnosis of large-scale, complex, dynamical systems.

Dr. Yan is a Fellow of ASME (2019). His honors and awards include the IEEE Instrumentation and Measurement Society Technical Award in 2019, the New Century Excellent Talents in University Award from the Ministry of Education in China in 2009, and multiple best paper awards. He is also the Associate Editor-in-Chief of the IEEE Transactions on Instrumentation and Measurement and an Associate Editor of the IEEE Systems Journal and the IEEE Sensors Journal.

**Xuefeng Chen,** (M'12) received the Ph.D. degree from Xi'an Jiaotong University, Xi'an, China, in 2004. He is currently a Professor of Mechanical Engineering with Xi'an Jiaotong University.

His current research interests include finite-element method, mechanical system and signal processing, diagnosis and prognosis for complicated industrial systems, smart structures, aero-engine fault diagnosis, and wind turbine system monitoring.

Dr. Chen was a recipient of the National Excellent Doctoral Dissertation of China in 2007, the Second Award of Technology Invention of China in 2009, the National Science Fund for Distinguished Young Scholars in 2012, and a Chief Scientist of the National Key Basic Research Program of China (973 Program) in 2015. He is the Chapter Chairman of the IEEE Xi'an and Chengdu Joint Section Instrumentation and Measurement Society.