Applied Soft Computing Journal 89 (2020) 106113

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Remaining useful life prediction using multi-scale deep convolutional neural network

Han Li [a], Wei Zhao [a], Yuxi Zhang [a,*], Enrico Zio [b]

[a] School of Electronic and Information Engineering, Beihang University, Group 203, Beijing 100191, China
[b] Department of Energy, Polytechnic of Milan, Via Ponzio 34/3, Milan 20133, Italy

ARTICLE INFO

Article history:
Received 15 January 2019
Received in revised form 17 December 2019
Accepted 18 January 2020
Available online 23 January 2020

Keywords:
Remaining useful life
Convolutional neural network
Multi-scale
Deep learning

ABSTRACT

Accurate and reliable remaining useful life (RUL) assessment result provides decision-makers valuable information to take suitable maintenance strategy to maximize the equipment usage and avoid costly failure. The conventional RUL prediction methods include model-based and data-driven. However, with the rapid development of modern industries, the physical model is becoming less capable of describing sophisticated systems, and the traditional data-driven methods have limited ability to learn sophisticated features. To overcome these problems, a multi-scale deep convolutional neural network (MS-DCNN) which have powerful feature extraction capability due to its multi-scale structure is proposed in this paper. This network constructs a direct relationship between Condition Monitoring (CM) data and ground-RUL without using any prior information. The MS-DCNN has three multi-scale blocks (MS-BLOCKs), where three different sizes of convolution operations are put on each block in parallel. This structure improves the network's ability to learn complex features by extracting features of different scales. The developed algorithm includes three stages: data pre-processing, model training, and RUL prediction. After the min–max normalization pre-processing, the data is sent to the MS-DCNN network for parameter training directly, and the associated RUL value can be estimated base on the learned representations. Regularization helps to improve prediction accuracy and alleviate the overfitting problem. We evaluate the method on the available modular aero-propulsion system simulation data (C-MAPSS dataset) from NASA. The results show that the proposed method achieves good prognostics performance compared with other network architectures and state-of-the-art methods. RUL prediction result is obtained precisely without increasing the calculation burden.

1. Introduction

Prognostics and health management (PHM) has been applied in diverse fields such as nuclear power facilities [1,2], engine systems [3,4], turbo machines [5], finance [6] and medicine [7,8], since first proposed in the United States Joint Strike Fighter (JSF) program in 1998 [9]. With the CM information collected from sensors, PHM technology can perform the detection and diagnosis of faults, the forecasting of the future state of health (SOH) and the estimation of the RUL of equipment. By using PHM technology can enable maximizing equipment utilization, reducing maintenance costs and avoiding accidents.

Current prognostic approaches can be classified into three main categories: model-based, data-driven and hybrid approaches [10]. Model-based approaches effectively utilize the prior information to develop a physical model, like the Paris–Erdogan (PE) model to describe fracture growth [11]. Jouin, Gouriveau, Hissel, Péra and Zerhouni [12] proposed a model-based method for Proton Exchange Membrane Fuel Cell (PEMFC)'s health assessment and prognostics. Marble et al. [13] developed a spall propagation model for bearing prognostics. Kacprzynski, Roemer, Modgil, Palladino and Maynard [14] proposed an approach for gear health prediction based on a physics-of-failure model. A three-loop Monte Carlo (MC) simulation scheme is proposed to operationalize the Multi-State Physics Modeling (MSPM) approach and system reliability assessment in Wang, Maio and Zio [15] work. Model-based approaches can perform well but their use is often restricted in practice because of insufficient understanding of the degradation mechanisms for building the model and estimating its parameters.

Data-driven approaches have developed with the improvement of sensor technology and signal transmission technology. Gebraeel, Lawley, Liu and Parmeshwaran [16] used neural network on bearing signals to estimate the bearing's failure times. Benkedjouh, Medjaher, Zerhouni and Rechak [17] estimated the

* Corresponding author.
E-mail address: zhangyuxi@buaa.edu.cn (Y. Zhang).

https://doi.org/10.1016/j.asoc.2020.106113
1568-4946/

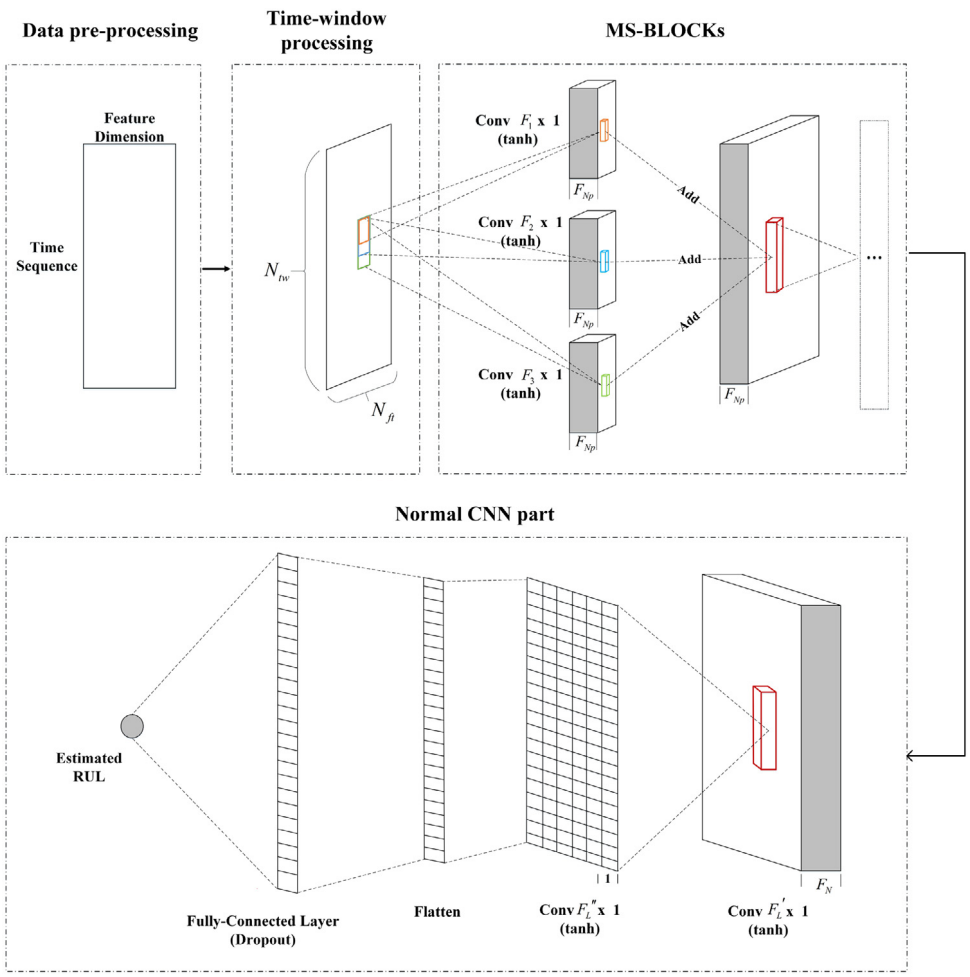

Fig. 1. Architecture of the MS-DCNN network.

RUL of bearing by support vector regression (SVR). A similarity-based method using pointwise fuzzy similarity analysis was proposed by Zio and Di Maio [18]. Liu and Zio [19] developed a weighted-feature and cost-sensitive regression model for component continuous degradation assessment. Malhi, Yan and Gao [20] used continuous wavelet transform (CWT) to preprocess the rolling bearing data, as inputs to recurrent neural networks (RNNs). A long short term memory (LSTM) neural network model was developed in Yuan, Wu and Lin [21] for aircraft turbofan engines' RUL prediction and fault diagnosis. These data-driven approaches mainly rely on historical data collected by sensors to establish a statistical model. Hybrid approaches [22] combine the model-based and data-driven approaches. A hybrid approach, which fuses the outputs from model-based approach and data-driven approach, was proposed by Hansen et al. [23], Baraldi, Compare, Sauco and Zio [24] proposed an original method which extended PF by combining to a data-driven approach.

Within the three approaches mentioned above, data-driven have become the most popular one in recent years and machine learning methods have shown good performance in reliability prognostics. However, for conventional machine learning approaches, superior learning performance (high generalization ability in most cases) depends on effective feature representation. Generally, to describe the degradation process of equipment accurately, the dimension of acquired features are inevitably high, which leads the computational efficiency to be very poor. Therefore, the dimensionality reduction methods are indispensable to transform the high-dimensional features into comparatively low-dimensional. Researchers have to spend much time on feature extraction and dimensionality reduction, which are also very complicated and heavily dependent on personal understanding of the concerned task.

Recently, the deep learning neural network is emerging as a universal network structure with an effective ability to automatically learn high dimensional features from input data. The advantages of this deep learning method mostly derive from two aspects. Firstly, it no longer requires complicated and time-consuming feature extraction and dimensionality reduction work but only need raw data with simple preprocessing as the inputs. Secondly, it can generate features with high-level abstractions of data through the deep hidden layers structure. Thus, CNN can lead to more efficient characteristics extraction compared with traditional machine learning methods such as SVM, logistic regression (LR), and shallow network structures, such as back propagation neural network (BPNN), multi-layer perception (MLP) and radical basis function neural network (RBFNN). Therefore, since proposed, deep learning has raised a lot of research attention and achieved great progress in many areas, especially in the domain of pattern recognition such as image, speech and video recognition [25,26].

In recent years, many researchers have applied deep learning to the PHM field and obtained high prediction accuracy. Ren, Cui, Sun and Cheng [27] proposed a fusion-based deep learning method to predict the RUL of multi-bearing systems. Liao, Jin and Pavel [28] proposed a Restricted Boltzmann Machine (RBM), which uses a new regular term and an unsupervised self-organizing graph algorithm for the RUL prediction of mechanical

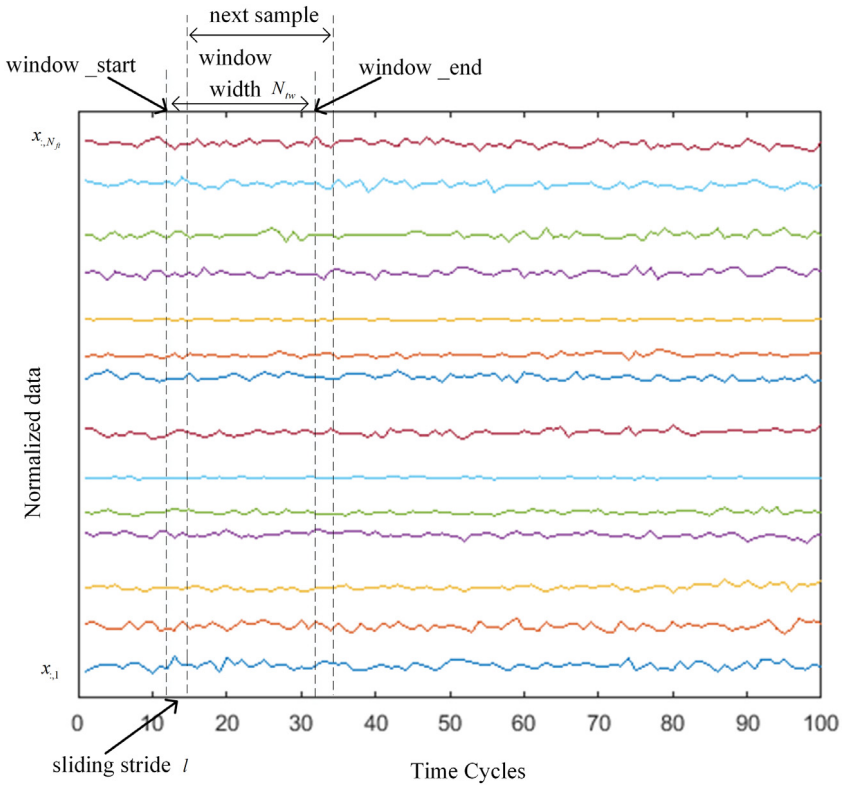

Fig. 2. Training samples with N_{ft} selected features.

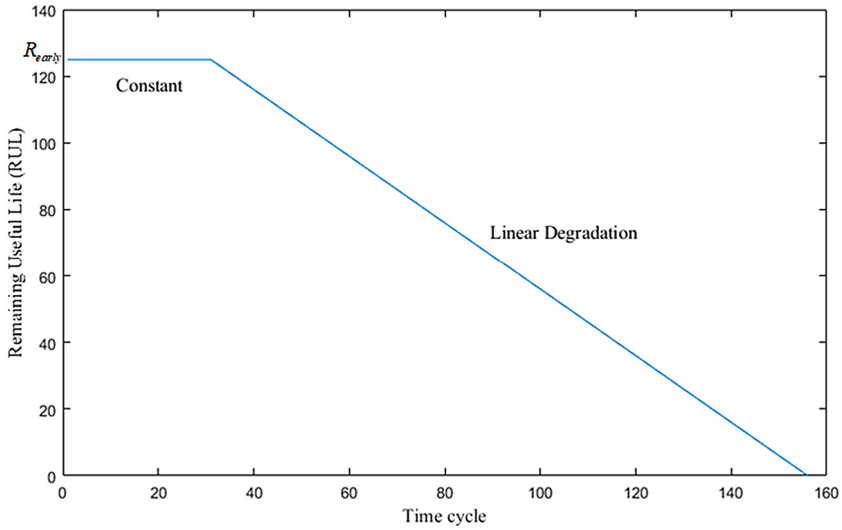

Fig. 3. Piece wise linear RUL function.

equipment. Deutsch and He [29] developed a method based on deep learning for rotating device RUL prediction. Within the deep learning networks, convolution neural networks (CNN) are specifically designed for complex signals to maintain the local spatial correlation regardless of scale, shift and distortion invariance. They are especially suitable to extract fault features from the CM data. The network architecture developed in this paper is based on CNN.

Although deep convolutional neural network (DCNN) has shown good prognostic performance, there are still three shortages:

(1) The learning ability of the network is not strong enough to extract information on different scales. Although it learns degradation features well in prognostics, it has a fixed convolution kernel size at each layer and always employs equal weights for the same inputs because of the weight sharing characteristic. Thus, the choice of the convolution kernel size becomes crucial for CNN network performance. Bigger kernel sizes can extract larger scale features which reflect the overall degradation trend of components and systems, whereas small kernel sizes learn specific detailed degradation characteristics. However, a fixed-size convolution kernel cannot learn diverse information on different scales at the same time.

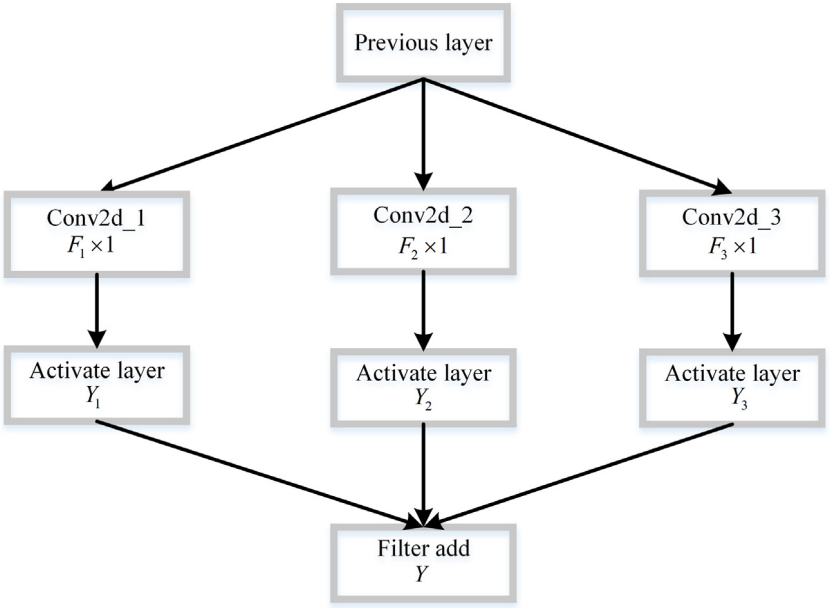

Fig. 4. Architecture of the MS-BLOCK.

(2) Too deep network structures are not suitable for prognostic prediction. Research [30] has shown that the process of deepening the network cannot further effectively improve network performance and the computing time for the training process increases almost linearly with the hidden layer number. Especially with small datasets, a too deep network structure is prone to overfitting.
(3) The robustness of deep learning methods is weak compared to model-based and traditional machine learning methods, such as SVR, decision tree, etc. as the training process has a certain level of randomness.

To address these problems, a two-dimensional Multi-Scale Deep Convolution Neural Network (MS-DCNN) for RUL prediction is proposed in this paper. CM signals generally contain features of different scales, which can be combined to reflect the overall degradation trend of the device. The proposed method utilizes an architecture named MS-BLOCK to extract multi-scale information of CM signals. The MS-BLOCK uses several filters of different sizes to capture more detail features in each layer, which significantly improves the network performance and RUL prognostics accuracy. The addition of regularization improves the prediction accuracy of the test dataset and effectively alleviates the problem of overfitting. In this paper, an available aero-engine dataset-C-MAPSS dataset is used for the validation of the proposed diagnostic algorithms superiority. The C-MAPSS dataset is widely used by scholars and proved to be reasonable and reliable [31]. Experimental results on C-MAPSS dataset show that the proposed approach achieves good prediction accuracy compared with other state-of-the-art methods. Moreover, MS-DCNN obtains lowest STD compared with other network structures, which shows strong robustness.

The rest of the paper is organized as follows. Section 2 presents the MS-DCNN structure and the flowchart of the proposed method. Section 3 discusses the development of the proposed method on the aircraft turbofan engine degradation data. Comparison experiments of the MS-DCNN network with other network architectures and state-of-the-art methods are displayed, and the impacts of network parameters are discussed in this section. In the end, conclusions and further work are given in Section 4.

2. Multi-scale convolutional neural network (MS-DCNN)

In this section, a new prediction architecture for RUL estimation, named MS-DCNN, is proposed. The main architecture of MS-DCNN network is shown in Fig. 1. The whole algorithm can be divided into five parts: data pre-processing, time-window processing, MS-BLOCKs, normal CNN part and RUL evaluation.

2.1. Data pre-processing

For the convenience of the subsequent data processing and fast training convergence, it is quite common to normalize the raw input data. We denote the selected sensor number as N_{ft} and the data length as L. The raw input data can be expressed as $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_L}]^{\mathbf{T}}$, $\mathbf{x_i} = \left[x_{i,1}, x_{i,2}, \ldots, x_{i,j}, \ldots, x_{i,N_{ft}}\right]$, where $x_{i,j}$ represents the jth sensor data at the ith cycle. In this paper, min–max normalization [32] is used for data pre-processing: each element $x_{i,j}$ of the input data is normalized to the range of $[-1, 1]$as Eq. (1).

$$\hat{x}_{i,j} = \frac{2\left(x_{i,j} - x^j_{\min}\right)}{\left(x^j_{\max} - x^j_{\min}\right)} - 1 \forall i, j \tag{1}$$

where $\hat{x}_{i,j}$ represents the normalized value of $x_{i,j}$, $x^j_{\max}$ and $x^j_{\min}$ are the maximum and minimum values of the raw input data in the jth sensor, respectively.

2.2. Time-window processing

After normalization, time-window processing is used to generate the network inputs by sliding the window through the raw data, as shown in Fig. 2. The window width is denoted as N_{tw} and the sliding stride is expressed as l. Input pairs can be expressed as $\mathbf{I} = [\mathbf{x_{window\_start}}, \ldots, \mathbf{x_{window\_end}}]$, $window\_end = window\_start + N_{tw}$. After a sliding operation with length l, the next sample can be expressed as $\mathbf{I'} = [\mathbf{x_{window\_start+l}}, \ldots, \mathbf{x_{window\_end+l}}]$. The input size of MS-DCNN is $N_{\text{tw}} \times N_{ft}$.

Considering that short sliding strides can increase the number of training samples, which may reduce the risk of overfitting, it is reasonable to set $l = 1$.

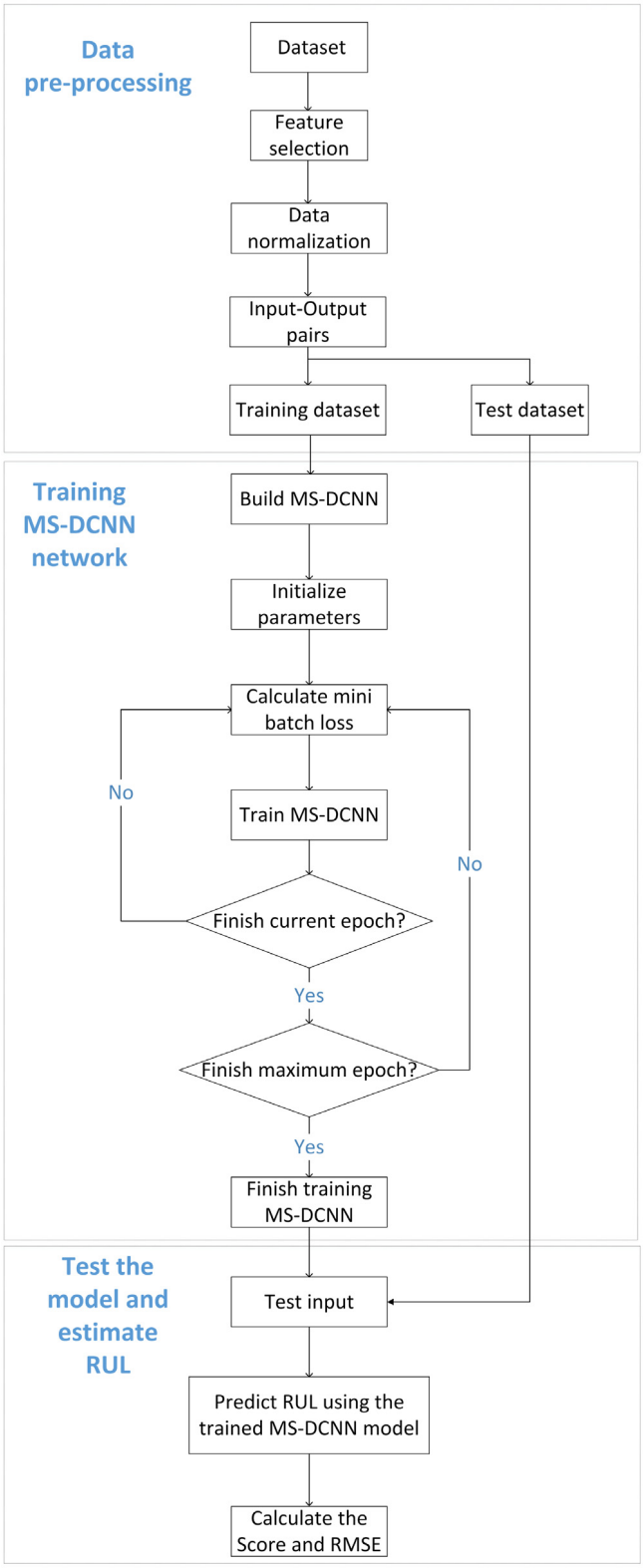

Fig. 5. Flowchart of the MS-DCNN architecture.

The training label is ground-RUL. A piece wise linear function [33] is used to model the RUL label for each input pair, as displayed in Fig. 3. It can be divided into two parts: a constant part and a linearly degrading part. RUL denoted as R_{early} is a constant value as long as the machine is considered healthy, then, the RUL exhibits a linear degradation trend until failure. Each pair of input values corresponds to an RUL value at the window_end.

2.3. Multi-scale block (MS-BLOCK)

The scheme of the MS-BLOCK is shown in Fig. 4. The output of the previous layer is used as the input of the MS-BLOCK. The inputs are sent to three channels with different convolution kernel sizes $F_1 \times 1, F_2 \times 1, F_3 \times 1$ for feature extraction. Different scale features can be extracted by setting these three parameters' size. In each channel, the input will be convolved first, then an activation function is added for the nonlinearity of the network model. Finally, the features extracted from each channel are added together as the output of the MS-BLOCK. The input patch in each convolutional layer is assumed to be $\mathbf{m} = (m_{ij})_{L' \times W'}, i = 1, 2, \ldots, L', j = 1, 2, \ldots, W'$, where L' and W' represent the length and width of the input, respectively and m_{ij} denotes the extracted feature values. The convolution operation in each convolutional layer can be defined as a multiplication operation between a filter kernel $\mathbf{w_{p,q}}$, $\mathbf{w_{p,q}} \in R^{F_p}$ $(p = 1, 2, 3)$, and the input patch $\mathbf{m}$, as shown in Eq. (2).

$$y_{p,q} = activate\left(\mathbf{w_{p,q}^T m} + b_{p,q}\right), p = 1, 2, 3; 0 < q \leq F_{Np}, q \in Z \tag{2}$$

where $\mathbf{w_{p,q}^T}$ represents the qth filter weight matrix's transpose of the pth channel. F_{Np} represents the filter number of the pth channel. $b_{p,q}$ and $activate$represent the qth filter's bias term of the pth channel and non-linear activation function [34], respectively. The output $y_{p,q}$can be considered as the learned feature of the filter kernel $\mathbf{w_{p,q}}$on the input patch $\mathbf{m}$. By performing the same operation, the feature map of the pth channel can be obtained, which is denoted as Eq. (3)

$$Y_p = \left\{y_{p,1}, y_{p,2}, \ldots, y_{p,q}, \ldots, y_{p,F_{Np}}\right\}, p = 1, 2, 3 \tag{3}$$

Then, the feature output learned from the three different size scales can be expressed as Eq. (4)

$$Y = \sum_{p=1}^{3} Y_p \tag{4}$$

The output feature map will continue to be learned as the input of the next layer. Note that raw input data have been sent into the MS-BLOCK after simple pre-processing. There are three convolutional layers with the same MS-BLOCK architecture stacked in the network for feature extraction. In order to facilitate the addition of features at different scales, it is necessary to keep the dimensions the same, so zero-padding [35] is used in this architecture and the number of convolution kernels at different scales is set to F_{Np}. The multiple fault features extracted from the different size filters are added as the input of the next convolutional layer. On the one hand, using multiple convolution layers to extract fault features in parallel can learn the mapping relationship more accurately, on the other hand, the addition of different features does not increase the dimension of the features used.

2.4. CNN part

A typical CNN mainly includes a convolutional layer, an activation function, and a pooling layer, in addition to the input and output layers. Each convolution layer has several convolution kernel cores. The main characteristics of the convolution layer are the local connection and weight sharing, which enable CNN translation-variation [34].

Table 1
Introduction of C-MAPSS dataset.

Sub-datasets	FD001	FD002	FD003	FD004
Engine units in the training dataset	100	260	100	249
Engine units in the test dataset	100	259	100	248
Fault modes	ONE (HPC Degradation)	ONE (HPC Degradation)	TWO (HPC Degradation, Fan Degradation)	TWO (HPC Degradation, Fan Degradation)
Conditions	ONE (Sea Level)	Six	ONE (Sea Level)	Six
Training samples	17731	48819	21820	57522
Test samples	100	259	100	248

Table 2
21 sensor outputs for degradation modeling.

Index	Description	Symbol
1	Total temperature at fan inlet	°R
2	Total temperature at LPC outlet	°R
3	Total temperature at HPC outlet	°R
4	Total temperature at LPT outlet	°R
5	Pressure at fan inlet	psia
6	Total pressure in bypass-duct	psia
7	Total pressure at HPC outlet	psia
8	Physical fan speed	rpm
9	Physical core speed	rpm
10	Engine pressure ratio (P50/P2)	–
11	Static Pressure at HPC outlet	psia
12	Ratio of fuel flow to Ps30	pps/psi
13	Corrected fan speed	rpm
14	Corrected core speed	rpm
15	Bypass Ratio	–
16	Burner fuel–air ratio	–
17	Bleed Enthalpy	–
18	Demanded fan speed	rpm
19	Demanded corrected fan speed	rpm
20	HPT coolant bleed	lbm/s
21	LPT coolant bleed	lbm/s

After the multi-scale convolution operation, the output feature maps of the three MS-BLOCK layers are sent to a normal CNN part. A general convolutional layer with F_N filters whose kernel size is $F_L' \times 1$ is constructed and one filter with $F_L'' \times 1$ filter size is set up in the last convolutional layer for reducing dimensionality by integrating the features learned previously. During the convolutional operations, zero-padding is employed to keep consistent the input and output dimensions of the different layers.

Next, the two-dimensional feature maps obtained from the previous layers are flattened and a fully-connected layer is connected with the one-dimension features. Afterward, dropout operation is applied and one neuron is linked at the end of the MS-DCNN network to provide the final output value, which in our case represents the predicted RUL.

Each network layer uses the hyperbolic tangent as activation function and Xavier normal initializer for network initializations [30]. Besides the dropout technique, also L2 regularization is adopted to prevent overfitting of the proposed model [36]. Adam algorithm [37], which can adaptively adjust the learning rate, is used to iteratively update the neural network parameters.

2.5. Regularization

Many literatures [38–40] have discussed the role of regularization in network optimization and literature [34,41] give people another way to think deeply about regularization in recent years. In this paper, regularization helps to improve the accuracy of the test dataset. Moreover, it is used to alleviate the overfitting problem. A common problem of deep learning is the need for a large amount of data because there are many parameters in the model that need to be obtained through training. A small amount of data may cause overfitting problems, which may lead to high accuracy on the training dataset but poor performance on the test dataset. Considering that the training data are typically limited in the practice of PHM applications, it is necessary to avoid overfitting. The regularization methods used in this paper include dropout and L2 regularization. More discussions about these methods have shown in Section 3.6.5.

Dropout modifies the neural network structure, as it "deletes" some hidden layer units at random, then keeps the input and output layers unchanged, and updates the weights of the network according to the steepest descent algorithm [42]. The neural network hereby obtained has only a few hidden layer elements and a small number of parameters, which can adapt to different datasets and avoid overfitting phenomenon [43].

Regularization also includes L1 and L2 regularization, the latter being also called weight decay. The L1 regularization can make the weight parameter w sparse while the L2 regularization minimizes the regularization term $\|w\|_2$ by modifying the loss function for the training. L2 regularization can avoid overfitting more effectively. L2 regularization can be defined as Eq. (5),

$$C = C_0 + \frac{\lambda}{2n}\sum_{w} w^2 \tag{5}$$

where C_0 denotes the initial loss function, which indicates RMSE in this paper. $\frac{\lambda}{2n}\sum_w w^2$ is the regularization term which weighs the proportion of the regularization term and the C_0 term. λ represents the regularization term coefficient and n is the number of weight parameters.

2.6. MS-DCNN architecture

The flowchart of the proposed MS-DCNN architecture is illustrated in Fig. 5. The steps description is given as follows:

i. First, the CM data have been pre-processed by a [-1,1] normalization. No other operation is applied to the raw data.
ii. The two-dimensional features, with a size of $N_{tw} \times N_{ft}$, generated by time-window processing constitute the training datasets and test datasets of the MS-DCNN network. Then, a suitable R_{early} and corresponding RUL labels are set up. After these operations, the Input–Output pairs of the MS-DCNN network are successfully established.
iii. Build MS-DCNN network according to Fig. 1.
iv. Initialize network parameters by Xavier initialization and train MS-DCNN network model on the training datasets using the deepest descent algorithm. The training datasets have been randomly divided into several mini-batches. The batch size is displayed in Table 2. L2 regularization is applied to constrain weights in loss function and dropout operation is used after the fully-connected layers. Early Stopping is used for preventing from overfitting during training process [44].
v. Predict the RUL from the test dataset with the trained MS-DCNN model, and evaluate the prediction performance using the scoring function and RMSE metrics.

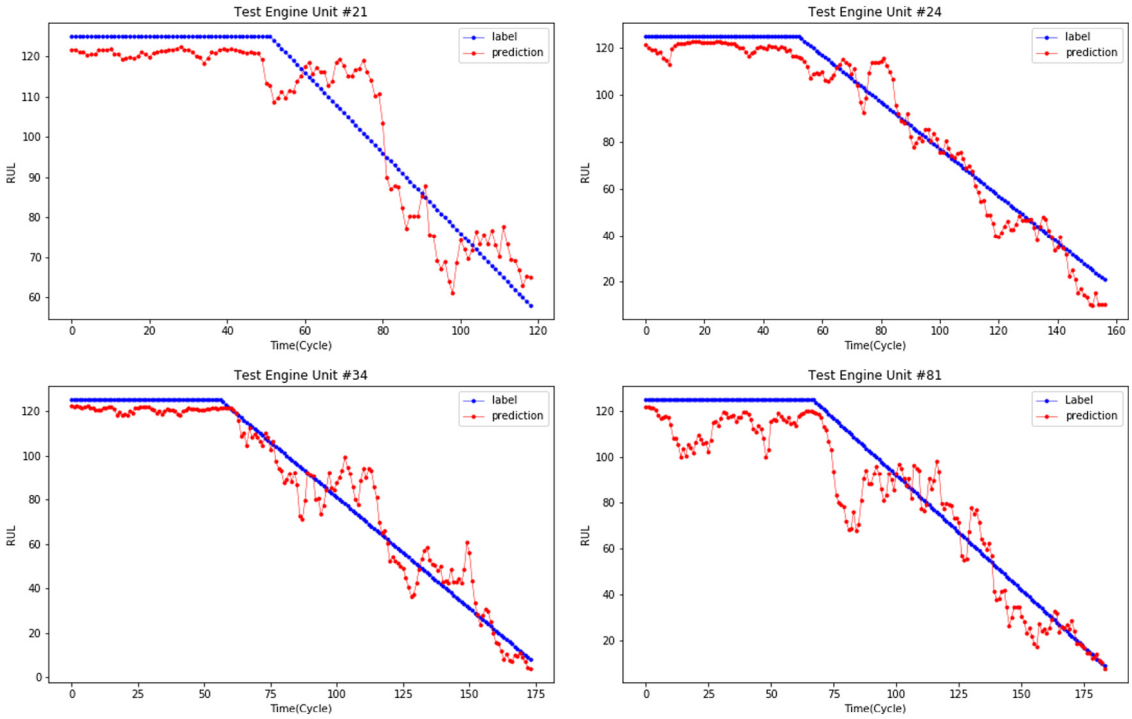

Fig. 6. Four engines life-time RUL prediction results on FD001.

Table 3
Parameters of the proposed approach.

Parameter	Value	Parameter	Value
F_N	10	N_{tw} for FD001 to FD004	30/20/30/15
$F_1/F_2/F_3$	10/15/20	Neurons in FC	64
F_L'	10	Dropout rate	0.5
F_L''	3	R_{early}	125
N_{ft}	14	Numbers of MS-BLOCK	3
Batch size	512	λ	1
L2 normalization	0.0001	Learning rate	0.0001
F_{Np}	10	Max epoch for FD001 to FD004	180/160/100/190
padding	same	Activate function	hyperbolic tangent (tanh)

3. Experimental study

In this section, the proposed approach MS-DCNN is assessed and compared with other methods on a benchmarking prognostic dataset, the NASA C-MAPSS dataset [45]. The other methods considered are deep neural network (DNN), recurrent neural network (RNN), long short-term memory (LSTM), and deep convolutional neural network (DCNN). Experiment results with different network structures are discussed in Section 3.4. Other state-of-the-art methods that have reported results on the C-MAPSS datasets have been considered for comparison, and a detailed description is given in Section 3.5. Section 3.6 explores the impact of different network parameters on the prediction performance. All the experiments are processed on a PC with Intel Core i7-7700K CPU, 8-GB RAM and GeForce GTX 1080 Ti GPU with Keras 2.1.5 (on the Tensorflow 1.1 backend). All the experimental results are averaged by 20 trials to reduce the effect of randomness.

3.1. Data description

The turbofan engine degradation dataset was simulated using commercial modular aero-propulsion system simulation (C-MAPSS), a simulation module developed in Matlab and Simulink environment [49]. The C-MAPSS datasets are collected from different parts of turbofan engine systems by NASA's Prognostic Center of Excellence and consist of four sub-datasets classified by different failure modes and operation settings, named FD001, FD002, FD003, FD004, respectively [32]. The characteristic of the C-MAPSS dataset is displayed in Table 1. Each sub-dataset is divided into one training set, one test set, and one ground-RUL set. These datasets consist of multivariate time series which record the state of the turbofan engines from health to failure. Each engine was monitored using 21 sensors, which are listed in Table 2. In order to compare with other methods, we selected 14 sensor measurements as the raw input data, as done in the literature [32,33], whose indices are 2, 3, 4, 7, 8, 9, 11, 12, 13, 14, 15, 17, 20 and 21. Generally, the state is considered healthy at the beginning of the time strides. Then, the engine begins to degenerate until it fails. Therefore, it is reasonable to use a piece-wise linear degradation model [33] to fit the engine's degradation process. The training datasets record all health information from

Table 4
The RMSE indicators of four engines whole life-time cycles, the early 80% life-time cycles and the later 20% life-time cycles.

	Whole life-time	The early 80% life-time	The later 20% life-time
#21	6.59	6.85	**5.43**
#24	6.99	6.92	**7.29**
#34	9.48	9.91	**7.51**
#81	14.99	16.14	**9.12**

Table 5
Evaluation metrics of different network architectures on C-MAPSS datasets.

		DNN [30]		RNN [30]		LSTM [30]		DCNN [30]		MS-DCNN	
		Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
FD001	RMSE	13.56	0.21	13.34	0.43	13.52	0.61	12.61	0.19	**11.44**	**0.07**
	Score	348.3	17.5	339.2	29.0	431.7	42.4	273.7	24.1	**196. 22**	**4.99**
FD002	RMSE	24.61	0.33	24.03	0.26	24.42	0.45	22.36	0.32	**19.35**	**0.08**
	Score	15622	872	14245	622	14459	815	10412	544	**3747**	**228**
FD003	RMSE	13.93	0.34	13.36	0.38	13.54	0.29	12.64	0.14	**11.67**	**0.06**
	Score	364.3	19.3	315.7	24.2	347.3	28.0	284.1	26.5	**241.89**	**6.95**
FD004	RMSE	24.31	0.24	24.02	0.41	24.21	0.36	23.31	0.39	**22.22**	**0.14**
	Score	16223	895	13931	1102	14322	1043	12466	853	**4844**	**365**

Table 6
Average training time with MS-DCNN and DCNN network.

	FD001		FD002		FD003		FD004	
	epoch	Time(s)	epoch	Time(s)	epoch	Time(s)	epoch	Time(s)
DCNN	250	135	250	278	250	156	250	294
MS-DCNN	180	132	160	272	100	90	190	312

Table 7
Performance of various approaches on C-MAPSS datasets.

Method	FD001		FD002		FD003		FD004	
	RMSE	Score	RMSE	Score	RMSE	Score	RMSE	Score
MS-DCNN	**11.44**	**196.22**	**19.35**	3747	**11.67**	**241.89**	**22.22**	4844
Semi-supervised setup [46]	12.56	231	22.73	**3366**	12.1	251	22.66	**2840**
DCNN [30]	12.61	273.7	22.36	10412	12.64	284.1	23.31	12466
RULCLIPPER [47]	13.27	216	22.89	2796	16	317	24.33	3132
MODBNE [32]	15.04	334.23	25.05	5585.34	12.51	421.91	28.66	6557.62
DBN [32]	15.21	417.59	27.12	9031.64	14.71	442.43	29.88	7954.51
LSTM [48]	16.14	338	24.49	4450	16.18	852	28.17	5550
MLP [32]	16.78	560.59	28.78	14026.72	18.47	479.85	30.96	10444.35
SVM [32]	40.72	7703.33	52.99	316483.31	46.32	22541.58	59.96	141122.19

Table 8
Performance with different numbers of MS-BLOCK.

Number of MS-BLOCK	1		2		3		4		5	
	Mean	STD	Mean	STD	Mean	STD	Mean	STD	Mean	STD
RMSE	12.57	0.21	11.85	0.12	**11.44**	0.09	11.97	0.1	12.37	0.13
Score	263.32	11.26	215.34	4.37	**196.22**	4.65	218.47	5.21	265.05	5.33

the beginning to the time of failure. During training, all data representing the health information are sent to the network after time-window processing. Then, the model obtained after network training is used to predict the engine's remaining useful life in the test datasets.

3.2. Evaluation indicator

In this paper, two metrics are used to evaluate the performance of the proposed MS-DCNN modeling approach, i.e. the scoring function and root mean squared error (RMSE).

Scoring function: The scoring function has been used in several works of literature [33,50–52] and adopted by the International Conference on Prognostics and Health Management (PHM08) Data Challenge. This scoring function emphasizes the late prediction and gives them much stronger penalty than early predictions, that because late prediction usually leads to more severe consequences.

(1) The function formula is displayed in Eq. (6).

$$s = \sum_{i=1}^{N} s_i, s_i = \begin{cases} e^{-\frac{d_i}{13}} - 1, \text{ for } d_i < 0 \\ e^{\frac{d_i}{10}} - 1, \text{ for } d_i \geq 0 \end{cases} \tag{6}$$

(2) where N is the total number of data samples and s denotes the score result. $d_i = RUL'_i - RUL_i$ represents the difference between the predicted result RUL'_i and the ground truth value RUL_i. RMSE: The function formula is displayed as Eq. (7)

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

where N is the total number of the data sample.

3.3. RUL prediction on C-MAPSS datasets

The detailed parameters of the proposed approach are displayed in Table 3. The life-time prediction result of the testing engine units on FD001 is obtained in Fig. 6. Four examples out of one hundred testing engine units, whose unit numbers are 21, 24, 34 and 81 respectively, are presented for demonstration. It can be seen that the prediction RUL in the early period is close to the constant R_{early} and afterward, the RUL prediction results perform almost a linear degradation trend until the end of the available testing samples. The RMSE indicators of four engines whole life-time cycles, the initial 80% life-time cycles, and the later 20% life-time cycles are shown in Table 4. One can see for the four engines, RMSE indicators of the later 20% life-time are lower than the early 80% life-time's RMSE indicators. Although there is some error existing between the prediction results and the ground-RUL values, the prognostic accuracy is high, especially

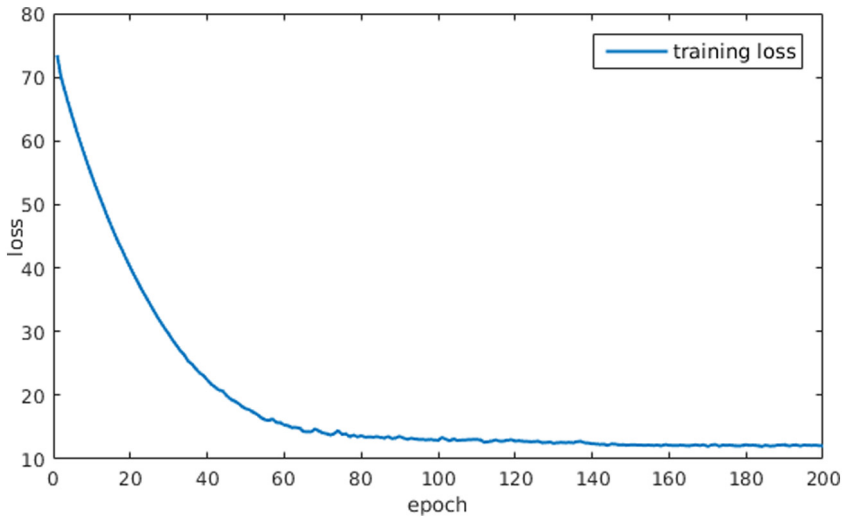

Fig. 7. Training loss curve of FD001.

when the engine units are close to failure because of the fault features are more prominent. The training loss curve is presented in Fig. 7. The training loss dropped 80% in the first 70 epochs and then kept a steady decline trend until 170th epoch. In the end, the training loss becomes stable at about the 180th epoch. From the experiment result, we can see that the proposed method achieves a good prediction performance.

3.4. Comparing with other network architectures

In this part of the study, different network structures are used for comparative experiments with the MS-BLOCK approach. The selected network and network structure settings are based on the literature [30]. In order to eliminate the impact of network depth, all networks are set to the same number of layers. The input and output layers are the same for all the compared methods and only the network structure is changed. The experimental results are shown in Table 5.

It can be observed that generally, the MS-DCNN has achieved the highest prediction accuracy and lowest STD in two evaluation indicators in all cases. The DCNN structure is the second-best using stacked convolutional layers. Comparing with DCNN network structure, we can conclude that MS-BLOCK contributes to the learning capability of the network as only the multi-scale structure is different from DCNN network. Score evaluation indicator has a higher penalty for late predication because late predictions impose a severe threat to reliability and safety in real-life PHM applications as the maintenance procedure would be scheduled too late. The accuracy of Score prediction is greatly improved, especially on the FD002 and FD004 datasets. The increasing number of operation conditions and fault modes make these two datasets more complicated and these sophisticated features contain more multi-scale degradation features. MS-BLOCK dramatically improves the prediction accuracy with its good feature extraction capability facing these complex fault prediction problems. Moreover, the lowest STD reflects that the network has strong robustness.

We also reproduced the DCNN method in literature [30] and compared the average training time of the MS-DCNN and DCNN networks for the four sub-datasets. The results are given in Table 6. The epoch in the Table denotes maximum epoch number when the trained network performs stably. From Table 6, we can see that both networks reach steady-state without too much time difference, but the proposed MS-DCNN achieves higher prediction accuracy than DCNN. That is because the MS-DCNN can learn enough feature information with fewer epoch times benefiting from the strong learning ability, although the training parameters are more than DCNN.

3.5. Comparison with state-of-the-art methods

Several studies have reported results on the four considered sub-datasets in the C-MAPSS datasets. Here, Semi-supervised setup [46], DCNN [30], RULCLIPPER [47], MODBNE [32], DBN [32], LSTM [48], MLP [32] and SVM [32] have been taken for comparison. The comparison results are summarized in Table 7.

From Table 7 we can see the proposed MS-DCNN method obtained substantially improved RMSE prediction accuracy on all subsets although the RULCLIPPER approach [47] and the semi-supervised learning approach [53] achieved slightly higher Score prediction accuracy on subset FD002 and FD004, respectively. More specifically, RMSE indicators on four sub-datasets have reduced 8.92%, 14.87%, 3.55%, 1.94%, respectively, compared with the latest state-of-the-art method [53] owing to the network's ability to extract multi-scale detail features. Therefore, the proposed method is promising in prognostic tasks.

3.6. Effects of parameters

In this section, the effect of some parameters in the proposed network is discussed on the FD001 dataset.

3.6.1. Numbers of MS-BLOCK

This experiment investigates the effect of the number of MS-BLOCK on the prognostic performance, at the same network depth. Four different network structures are shown in Fig. 11. Table 8 describes the result of the experiments.

From Table 8, we can see the prognostics performance is consequently improved as the number of MS-BLOCK increases. However, the marginal growth of prognostics performance would sharply decline once the MS-BLOCK number exceeds some specific value. The raw input data contains limited feature information. Unlimited increasing the number of MS-BLOCK may cause over-fitting problems, rather than extracting more useful information, which would result in a high accuracy result in the training dataset and a small accuracy result in the test dataset. On the other side, the computational amount would increase seriously when the number of MS-BLOCK is becoming larger since the number of parameters remaining to be trained in the convolution layer increases. In this paper, the default is set to 3.

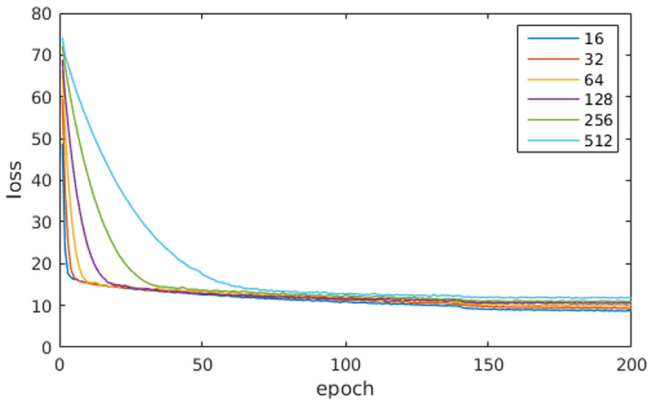

Fig. 8. Comparison of the training loss evolution with different batch size.

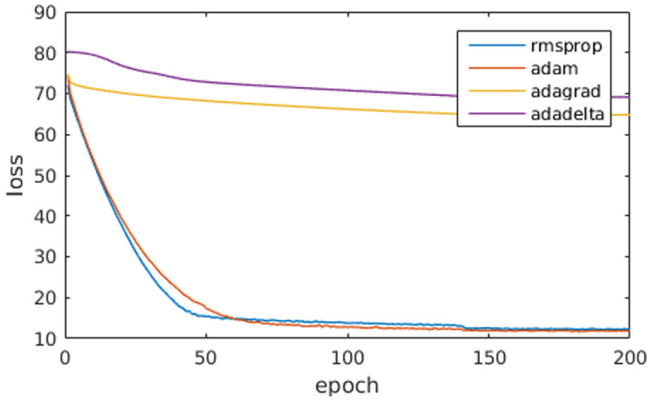

Fig. 9. Comparison of the training loss evolution with various optimizers.

3.6.2. Number of neurons in the fully-connected layer

Table 9 presents the prediction result of the proposed model with different numbers of fully-connected layer neurons. The number of fully-connected layer neurons has a significant influence on the prediction result. The fully-connected layer holds composite and aggregated information from all the convolutional layers that matter the most. The increasing number of neurons can improve the model's ability to fusing features with the cost of a complexed model and slow running speed. However, the risk of overfitting also increases. Conversely, fewer neurons may reduce the complexity of the model, make the network convergence faster and the ability to fuse features is limited, thereby getting poor performance. The selection of the neurons number in the fully connected layer is generally based on the balance of the model size and running speed. In general, the number of neurons is selected in terms of the amount of dataset. Usually, a large and complex dataset need a larger neuron number, and a smaller neuron number is more suitable for a small dataset with low dimension features. The determination of specific parameters is empirical without a specific formulation. From Table 9, we can conclude that 64 is the best parameter for the fully-connected neuron number, in our case study.

Table 9
Performance with different numbers of fully-connected layer neurons.

Neuron numbers	32		64		100		128	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
RMSE	12.13	0.13	**11.44**	0.09	12.66	0.23	12.33	0.19
Score	205.76	4.66	**196.22**	4.65	266.79	9.36	243.56	9.89

Table 10
Performance with different batch size.

	16	32	64	128	256	512
RMSE	14.37	14.21	13.58	12.26	11.90	**11.44**
Score	359.72	361.71	345.26	228.39	221.42	**196.22**

3.6.3. Impact of the batch size

In this case, we test the impact of the batch size. The batch size determines the direction of loss declining, which has an essential influence on the performance of the algorithm. Fig. 8 shows the loss values obtained when training with different batch size, while Table 8 displays the predictive performance of those different sets. From Fig. 8 and Table 10, we can conclude that a larger batch size has a smaller oscillating and performs better as it is commonly found in the literature [53]. However, large batch size can be limited by GPU memory. The batch size of 512 samples is appropriate based on the experiments and it is used in all the case studies in this paper.

3.6.4. Impact of the network optimizer

We test four network optimizers: AdaGrad, AdaDelta, Adam, and RMSProp in this experiment. The comparison of the training loss evolution with various optimizers is displayed in Fig. 9. Table 11 displays prediction results of various network optimizers.

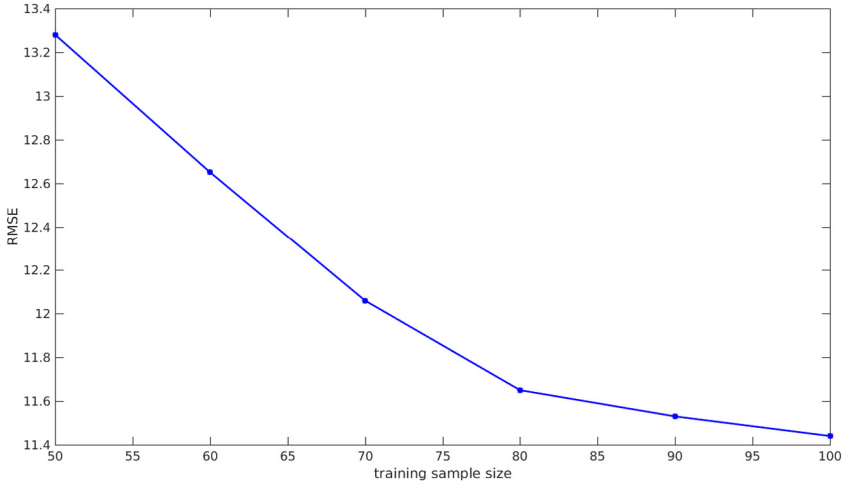

Fig. 10. RMSE with different training sample sizes.

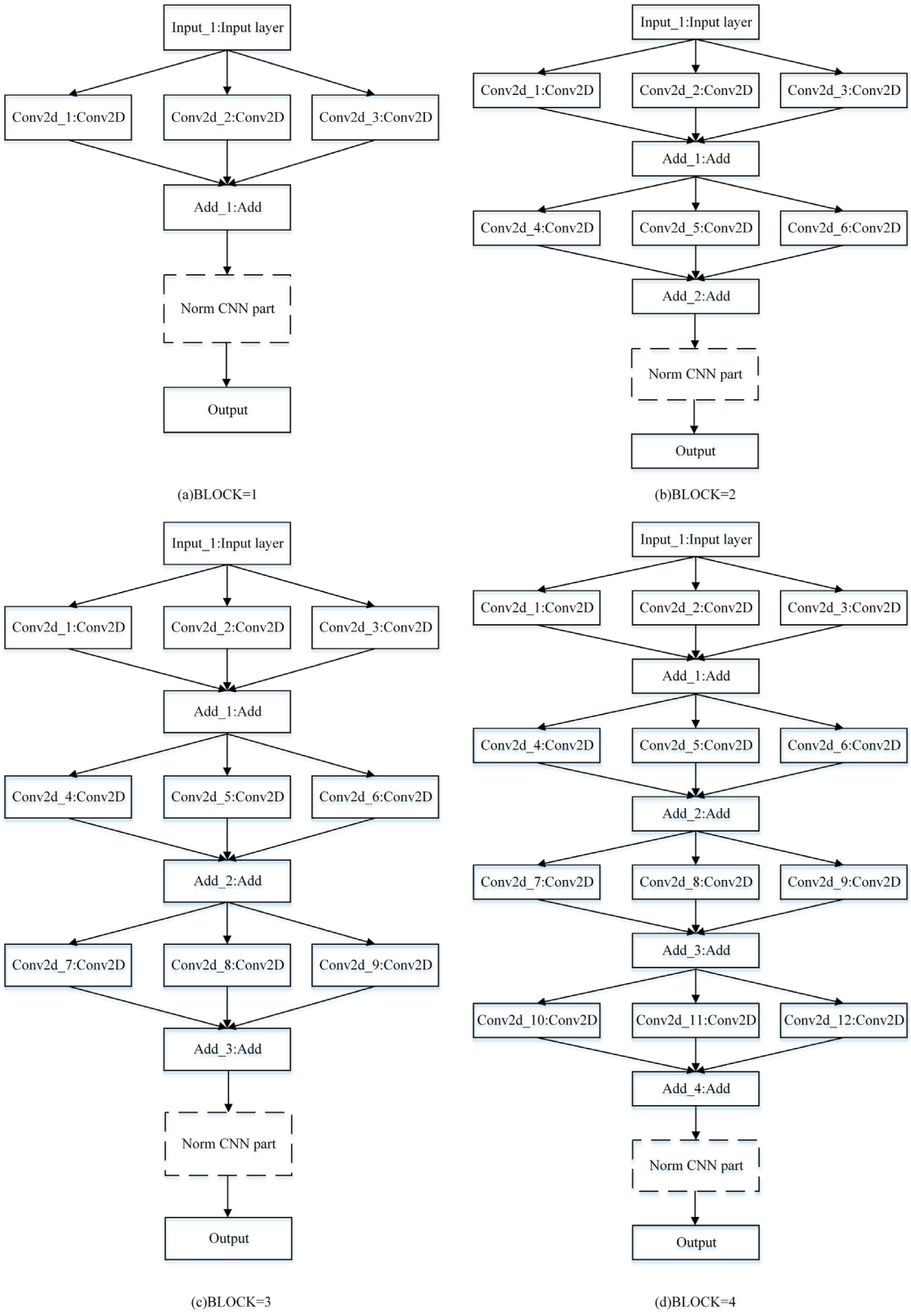

Fig. 11. Illustration of different numbers of MS-BLOCK in the MS-DCNN network.

The AdaDeleta and AdaGrad optimizers do not converge during the training process while Adam and RMSProp perform fast convergence in the first fifty epoch and keep smooth declining during the subsequent training process. However, Adam optimizer achieves the lowest RMSE and Score in this experiment. The choice of optimizer depends on the architecture and the problem at hand [54].

Table 11
Performance with different network optimizer.

	RMSProp	Adam	AdaGrad	AdaDelta
RMSE	12.07	**11.44**	70.05	73.94
Score	218.58	**196.22**	92303.58	132029.03

Table 12
Performance with different regularization methods.

	00	01	10	11
Training loss	3.66	4.51	3.92	12.16
RMSE (test loss)	13.22	12.20	12.64	**11.44**
Score	280.57	241.64	279.38	**196.22**

3.6.5. Impact of regularization

The regularization method could automatically cut back the unimportant feature variables and extract the vital feature variables from many multiple features. This operation can reduce the magnitude of the feature variable make a great significance for the RUL prediction of increasingly complex feature dimensions of failure features. We have given verification from several experiments. We set up four experiments on FD001: no dropout and L2 regularization (denoted as 00), dropout only (01), L2 regularization only (10), and dropout and L2 regularization (11). These four sets of experiments are in the same setting except for the regularization method. Table 12 shows the experimental results.

In light of these results, the 00 strategy is significantly worse than the other strategies. The 11 strategy achieves the highest prediction accuracy, which indicates that regularization is necessary for this work. It should be noted there has an impact on prediction accuracy whether using dropout or L2 regularization. Moreover, when the regularization is turned off (00, 01, 10), training loss is significantly smaller than test loss, which means overfitting problem occurs. However, in 11 strategy, the training loss and test loss are almost equal, indicating regularization alleviate overfitting problem obviously. In summary, regularization helps to improve the accuracy of the test dataset, and it is encouraged to use both dropout and L2 regularization together.

3.6.6. Experiment on the training sample size

We have set up several experiments on FD001 to analyze the generalization performance of the network. FD001 contains a series of 2D multi-state data of 100 engines. We use 50, 60, 70, 80, 90, and 100 engines as training datasets, respectively, and use the same network structure for training. The prediction results are shown in Fig. 10. We can see the experiment with the larger training sample achieves the higher prediction accuracy as it contains more feature information. When the number of engines in the training sample is 80, 90 or 100, RMSE curve changes gently which indicates the network is insensitive to data and the generalizing ability is good.

4. Conclusions

In this paper, a new deep learning network model named MS-DCNN, which can learn multi-scale information of input data has been proposed for prognostics. The method adopted a new MS-BLOCK architecture. This architecture can extract the information of different scale features by setting different convolution kernel size in parallel at the same layer. The addition and fusion of these features can improve the learning ability of the network. L2 regularization and dropout help to improve the prediction accuracy and alleviate the overfitting problem. Experiments are carried out on the accessible C-MAPSS dataset. The task aims to predict the RUL of aero-engines units accurately. Small RMSEs and scores between raw data and ground-RUL have been obtained on testing datasets.

In the experimental study, the proposed method is compared with other deep learning network architectures such as DNN, RNN, LSTM, and DCNN. The proposed MS-DCNN method has achieved high RUL prediction accuracy without increasing computational load and training time, which proves the superiority of MS-BLOCK architecture. Smallest STDs compared with other methods, reflect the strong robustness of the algorithm, which is of considerable significance to the industrial applications. Comparing with other state-of-the-art methods such as Semi-supervised setup, MODBNE, DBN, and LSTM, it can be seen that the RMSE indicators dropped 8.92%, 14.87%, 3.55%, 1.94%, respectively, on the four datasets. Hence, it is promising in real-life PHM applications subjected to multiple operating conditions and fault modes, as the complexity of the features is increasing, and MS-BLOCK has the powerful ability to extract multi-scale features.

In this article, we have explored the impact of network parameters and optimization methods on algorithm performance. However, the size of the multi-scale convolution kernel is obtained empirically. In future research, we will explore a way to automatically determine the sizes and numbers of the convolution kernels in the network during the training process. Furthermore, the amount of calculation and the numbers of parameters are also two parameters which can be tuned in the future.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.asoc.2020.106113.

CRediT authorship contribution statement

Han Li: Methodology. **Wei Zhao:** Project administration. **Yuxi Zhang:** Conceptualization. **Enrico Zio:** Writing - review & editing.

References

[1] K. Jamali, Achieving reasonable conservatism in nuclear safety analyses, Reliab. Eng. Syst. Saf. 137 (2015) 112–119.
[2] J. Park, W. Jung, A systematic framework to investigate the coverage of abnormal operating procedures in nuclear power plants, Reliab. Eng. Syst. Saf. 138 (2015) 21–30.
[3] J. Xu, Y. Wang, L. Xu, PHM-Oriented integrated fusion prognostics for aircraft engines based on sensor data, IEEE Sens. J. 14 (2014) 1124–1132.
[4] X. Wu, Y. Chang, J. Mao, Z. Du, Predicting reliability and failures of engine systems by single multiplicative neuron model with iterated nonlinear filters, Reliab. Eng. Syst. Saf. 119 (2013) 244–250.
[5] K.-Y. Chen, Forecasting systems reliability based on support vector regression with genetic algorithms, Reliab. Eng. Syst. Saf. 92 (2007) 423–432.
[6] L.B. Andersen, D. Häger, S. Maberg, M.B. Næss, M. Tungland, The financial crisis in an operational risk management context—A review of causes and influencing factors, Reliab. Eng. Syst. Saf. 105 (2012) 3–12.
[7] T. Kontogiannis, S. Malakis, A systemic analysis of patterns of organizational breakdowns in accidents: A case from Helicopter Emergency Medical Service (HEMS) operations, Reliab. Eng. Syst. Saf. 99 (2012) 193–208.
[8] M.d.C. Moura, J.M. Santana, E.L. Droguett, I.D. Lins, B.N. Guedes, Analysis of extended warranties for medical equipment: A stackelberg game model using priority queues, Reliab. Eng. Syst. Saf. 168 (2017) 338–354.
[9] K.G. Blemel, Dynamic autonomous test systems for prognostic health management, 1998, Joint strike fighter program office arlington va.
[10] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, D. Siegel, Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications, Mech. Syst. Signal Process. 42 (2014) 314–334.
[11] P.C. Paris, A critical analysis of crack propagation laws, Trans. ASME D 85 (1963) 528–533.

[12] M. Jouin, R. Gouriveau, D. Hissel, M.-C. Péra, N. Zerhouni, Degradations analysis and aging modeling for health assessment and prognostics of PEMFC, Reliab. Eng. Syst. Saf. 148 (2016) 78–95.
[13] S. Marble, B.P. Morton, Predicting the remaining life of propulsion system bearings, in: 2006 IEEE Aerospace Conference, 2006, p. 8.
[14] G.J. Kacprzynski, M.J. Roemer, G. Modgil, A. Palladino, K. Maynard, Enhancement of physics-of-failure prognostic models with system level features, in: Proceedings, IEEE Aerospace Conference, vol. 2916, 2002, pp. 6-2919-2916-2925.
[15] W. Wang, F.D. Maio, E. Zio, Three-loop Monte Carlo simulation approach to multi-state physics modeling for system reliability assessment, Reliab. Eng. Syst. Saf. 167 (2017) 276–289.
[16] N. Gebraeel, M. Lawley, R. Liu, V. Parmeshwaran, Residual life predictions from vibration-based degradation signals: a neural network approach, IEEE Trans. Ind. Electron. 51 (2004) 694–700.
[17] T. Benkedjouh, K. Medjaher, N. Zerhouni, S. Rechak, Remaining useful life estimation based on nonlinear feature reduction and support vector regression, Eng. Appl. Artif. Intell. 26 (2013) 1751–1760.
[18] E. Zio, F. Di Maio, A data-driven fuzzy approach for predicting the remaining useful life in dynamic failure scenarios of a nuclear system, Reliab. Eng. Syst. Saf. 95 (2010) 49–57.
[19] J. Liu, E. Zio, Weighted-feature and cost-sensitive regression model for component continuous degradation assessment, Reliab. Eng. Syst. Saf. 168 (2017) 210–217.
[20] A. Malhi, R. Yan, R.X. Gao, Prognosis of defect propagation based on recurrent neural networks, IEEE Trans. Instrum. Meas. 60 (2011) 703–711.
[21] M. Yuan, Y. Wu, L. Lin, Fault diagnosis and remaining useful life estimation of aero engine using LSTM neural network, in: 2016 IEEE International Conference on Aircraft Utility Systems (AUS), 2016, pp. 135–140.
[22] A. Heng, S. Zhang, A.C.C. Tan, J. Mathew, Rotating machinery prognostics: State of the art, challenges and opportunities, Mech. Syst. Signal Process. 23 (2009) 724–739.
[23] R.J. Hansen, D.L. Hall, S.K. Kurtz, New approach to the challenge of machinery prognostics, J. Eng. Gas Turbines Power 117 (1995) 320–325.
[24] P. Baraldi, M. Compare, S. Sauco, E. Zio, Ensemble neural network-based particle filtering for prognostics, Mech. Syst. Signal Process. 41 (2013) 288–300.
[25] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A.r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, B. Kingsbury, Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups, IEEE Signal Process. Mag. 29 (2012) 82–97.
[26] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: International Conference on Neural Information Processing Systems, 2012, pp. 1097–1105.
[27] L. Ren, J. Cui, Y. Sun, X. Cheng, Multi-bearing remaining useful life collaborative prediction: A deep learning approach, J. Manuf. Syst. 43 (2017) 248–256.
[28] L. Liao, W. Jin, R. Pavel, Enhanced restricted Boltzmann machine with prognosability regularization for prognostics and health assessment, IEEE Trans. Ind. Electron. 63 (2016) 7076–7083.
[29] J. Deutsch, D. He, Using deep learning-based approach to predict remaining useful life of rotating components, IEEE Trans. Syst. Man Cybern. S 48 (2018) 11–20.
[30] X. Li, Q. Ding, J.-Q. Sun, Remaining useful life estimation in prognostics using deep convolution neural networks, Reliab. Eng. Syst. Saf. 172 (2018) 1–11.
[31] E. Ramasso, A. Saxena, Review and analysis of algorithmic approaches developed for prognostics on CMAPSS dataset, in: Conference of the Prognostics and Health Management Society, 2015.
[32] C. Zhang, P. Lim, A.K. Qin, K.C. Tan, Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics, IEEE Trans. Neural Netw. Learn. Syst. 28 (2017) 2306–2318.
[33] F.O. Heimes, Recurrent neural networks for remaining useful life estimation, in: 2008 International Conference on Prognostics and Health Management, 2008, pp. 1–6.
[34] J. Koushik, Understanding Convolutional Neural Networks, 2016.
[35] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
[36] F. Nie, H. Huang, X. Cai, C. Ding, Efficient and robust feature selection via joint ℓ 2, 1 -norms minimization, in: International Conference on Neural Information Processing Systems, 2010, pp. 1813–1821.
[37] D. Kingma, J. Ba, Adam: A method for stochastic optimization, Comput. Sci. (2014).
[38] D.V. Hieu, L.D. Muu, P.K. Quy, L.V. Vy, Explicit extragradient-like method with regularization for variational inequalities, Results Math. 74 (2019).
[39] I.H. Ikasari, V. Ayumi, M.I. Fanany, S. Mulyono, Multiple regularizations deep learning for paddy growth stages classification from LANDSAT-8, Int. C Adv. Comp. Sci. I (2016) 512–517.
[40] G.C. Cheng, Y.X. Hou, X.Z. Zhao, Q. Yu, Local and Non-Local Regularization for Semi-supervised deep learning, in: International Conference on Control Engineering and Automation, Iccea 2014, 2014, pp. 272–278.
[41] A. Hernandez-Garcia, P. König, Do deep nets really need weight decay and dropout?, 2018.
[42] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, Nature 323 (1986) 533.
[43] M. Jones, T. Poggio, Regularization theory and neural networks architectures, Neural Comput. 7 (1995) 219–269.
[44] Y. Yao, L. Rosasco, A. Caponnetto, On early stopping in gradient descent learning, Const. App. 26 (2) (2007) 289–315.
[45] A. Saxena, K. Goebel, D. Simon, N. Eklund, Damage propagation modeling for aircraft engine run-to-failure simulation, in: 2008 International Conference on Prognostics and Health Management, 2008, pp. 1–9.
[46] A. Listou Ellefsen, E. Bjørlykhaug, V. Æsøy, S. Ushakov, H. Zhang, Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture, Reliab. Eng. Syst. Saf. 183 (2019) 240–251.
[47] E. Ramasso, Investigating computational geometry for failure prognostics, 2014.
[48] S. Zheng, K. Ristovski, A. Farahat, C. Gupta, Long short-term memory network for remaining useful life estimation, in: IEEE International Conference on Prognostics and Health Management, 2017, pp. 88–95.
[49] X. Fang, K. Paynabar, N. Gebraeel, Multistream sensor fusion-based prognostics model for systems with single failure modes, Reliab. Eng. Syst. Saf. 159 (2017) 322–331.
[50] T. Wang, Y. Jianbo, D. Siegel, J. Lee, A similarity-based prognostics approach for Remaining Useful Life estimation of engineered systems, in: 2008 International Conference on Prognostics and Health Management, 2008, pp. 1–6.
[51] J.B. Coble, J.W. Hines, Prognostic algorithm categorization with PHM Challenge application, in: 2008 International Conference on Prognostics and Health Management, 2008, pp. 1–11.
[52] P. Wang, B.D. Youn, C. Hu, A generic probabilistic framework for structural health prognostics and uncertainty management, Mech. Syst. Signal Process. 28 (2012) 622–637.
[53] H. Wu, S. Prasad, Semi-supervised deep learning using pseudo labels for hyperspectral image classification, IEEE Trans. Image Process. 27 (2018) 1259–1270.
[54] S. Lathuiliére, P. Mesejo, X. Alameda-Pineda, R. Horaud, A comprehensive analysis of deep regression, IEEE Trans. Pattern Anal. Mach. Intell. (2019) 1–1.