Cloud-Edge based Lightweight Temporal Convolutional Networks for Remaining Useful Life Prediction in IIoT

Lei Ren, Yuxin Liu, Xiaokang Wang, Jinhu Lü, and M. Jamal Deen

***Abstract*—Industrial Internet of Things (IIoT), as an important industrial branch of Internet of Things (IoT), has an essential purpose to improve intelligent industrial production. For this purpose, IIoT big data should be efficiently processed to mine valuable information. In handing the IIoT big data, cloud-edge computing is getting more attention to reduce the interaction latency to meet the real-time requirement, especially in the field of prognostic and health management (PHM). It is expected that, Artificial intelligence (AI) technologies will significantly change the manner of processing IIoT big data. Therefore, new methods about PHM, combining cloud-edge computing with AI technologies, are required to process the IIoT big data for intelligent industrial manufacturing. As an essential element of PHM, predicting the remaining useful life (RUL) of industrial equipment plays an increasingly crucial role, especially for the industrial intelligence. However, traditional methods pay much attention on prediction accuracy and neglect the influence of computing time. In this paper, by combining cloud-edge computing with AI technology, a new data-driven method, namely cloud-edge based lightweight temporal convolutional networks, for RUL prediction is proposed. First, to meet the real-time requirement, a cloud-edge computing and AI based framework for RUL prediction is presented. Second, a new model structure named lightweight temporal convolutional network (LTCN) is proposed and applied in the framework. Real-time prediction results will be obtained in edge plane and higher accuracy prediction results will be obtained through historical information in the cloud plane. Third, an incremental learning approach based on updating partial parameters of LTCN is discussed to improve the accuracy of prediction models with newly collected data. Experiments show that our method can improve the prediction accuracy and computing time of RUL.**

***Index Terms*—Industrial Internet of Things, Cloud-Edge Computing, RUL Prediction, Lightweight Temporal Convolutional Network, Incremental Learning.**

I. Introduction

INTERNET of Things (IoT) is a network of interconnected devices and things and uses intelligent software to autonomously collect, analyze and act on data or information for the benefit of human society[1, 2]. The Internet of Things can also be considered as the integration of the cyber space and physical space, where the cyber space mainly contains digital, logical and discrete information, and the physical space is the natural and human-made space in which we live [3].

As an important branch of IoT, Industrial Internet of Things (IIoT), by wireless and wired communication, integrates industrial production equipment, monitoring equipment, control systems and intelligent analysis technologies into all aspects of industrial production. IIoT allows for greatly improved efficiency, and enhanced product quality, finally leading to industrial intelligence [4, 5].

Large-scale IIoT big data are generated and collected from every aspect of industrial manufacturing, including the production, transportation and after-sales, and it has significant value for industrial intelligence. Therefore, to enhance the level of industrial intelligence, we need to efficiently process and analyze the IIoT big data [6–8]. However, IIoT big data are large-scale and continuously generated, which brings enormous challenges for its processing and application. Also, IIoT big data contains large-scale historical data and small-scale local data, and both of them should be processed efficiently and quickly [1].

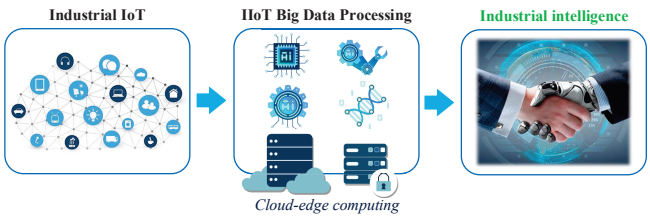

Fig. 1. Schematic representation of how IIoT generates big data that to be processed in cloud-edge computing for industrial intelligence.

Cloud computing, as a scalable computing platform usually used to process and analyze the large-scale historical data, will provide strong support for processing large-scale IIoT big data [1]. Meanwhile, with the popularity of smart devices and the development of computational and storage capacities, edge computing, as an emerging paradigm of computing, has attracted widespread attention from industry and academia. Edge computing is used to process the small-scale local data

L. Ren, Y. Liu and J. Lü are with School of Automation Science and Electrical Engineering, Beihang University, Beijing, 100191, China, and Beijing Advanced Innovation Center for Big Data-based Precision Medicine, Beihang University, Beijing, 100191, China.

X. Wang is with Department of Computer Science, St. Francis Xavier University, Antigonish, B2G 2W5, Canada.

M.J. Deen is with Department of Electrical and Computer Engineering, McMaster University, Hamilton, L8S 4K1, Canada.

L. Ren is the corresponding author, E-mail: renlei@buaa.edu.cn.

as an edge device of IIoT [1, 9, 10]. Therefore, cloud-edge computing in IIoT provides new opportunities and challenges in processing and mining important information in IIoT, especially in the field of prognostic and health management (PHM) [11].

Artificial intelligence (AI) with its rapid and expansive development brings new opportunities for the growth of the Internet of Things (IoT), and even the Industrial Internet of Things (IIoT). As shown in Figure 1, AI technologies in the cloud-edge computing environment is needed to process and analyze the IIoT big data promptly and accurately for industrial intelligence [4, 5].

As an essential part of industrial intelligence, prognostic and health management (PHM), including fault diagnosis and remaining useful life (RUL) prediction, has an important role in improving production efficiency and avoiding failure in industrial equipment. The remaining useful life (RUL) prediction of industrial component, such as rolling bearings, lithium-ion batteries and gearbox, is an essential element of PHM. For example, rolling bearings, as core components of industrial rotating equipment, play an important role in most industrial machinery. Any bearing failure may lead to reducing production and increasing downtime and safety risks. Many model-driven methods and data-driven methods were proposed to predict RUL of industrial equipment. Model-driven methods describe the equipment degradation by building mathematical or physical models. However, it is difficult to establish an accurate model for a complex non-linear system. Based on industry big data, data-driven methods have stronger abilities for solving non-linear problems. The capacity to analyze big data is improved greatly with the development of deep learning technology, and many methods based on deep learning are used in RUL predictions. In recent years, Convolution Neural Network [12] has shown greater capability in processing long sequence data than models using Recurrent Neural Network.

At present, there are still several challenges to be addressed for current data-driven prediction methods. First, prediction methods in the cloud server may cause delays of equipment health monitoring due to the interaction latency and huge data transmission. Second, RUL prediction models usually have complex architectures and a large number of parameters to ensure high prediction accuracy, which may lead to poor time-consumption performance. Third, the data constantly generated and collected from IIoT can be further used to improve the prediction accuracy.

To address these three challenges, the contributions in this paper can be summarized as follows. (1) A cloud-edge computing and artificial intelligence based framework for remaining useful life prediction in IIoT is presented to meet the real-time requirement in practical applications. (2) Based on the temporal convolutional network, a new model structure named lightweight temporal convolutional network (LTCN) is proposed. In the edge and cloud planes, a single sampling time lightweight temporal convolutional network (ST-LTCN) and a multiple sampling time lightweight temporal convolutional network (MT-LTCN) are presented, respectively. (3) Because of data being constantly generated and collected from IIoT, an incremental learning method based on updating partial parameters of LTCN is proposed to avoid having to recompute historical data.

This paper is organized as follows. The relevant work about Internet of Things, cloud-edge computing and RUL prediction methods are presented in section II. In section III, the cloud-edge based framework with data preprocessing, lightweight temporal convolutional networks and incremental learning module is proposed. Then, in section IV and section V, the structure of lightweight temporal convolutional network and RUL prediction of equipment method are respectively presented. In the section VI, the experiment results are presented and analyzed based on an open dataset. Finally, conclusions are given in section VII.

II. Related work

In this section, the related work about Internet of Things, cloud-edge computing and remaining useful life prediction methods of industrial equipment are briefly reviewed.

Internet of Things. Internet of Things (IoT) collects information about objects and processes from sensing devices. Recently, several typical IoT applications were discussed, including Industrial Internet of Things, mobile crowdsensing and intelligent network management [13, 14]. In addition, some path planning and scheduling schemes are proposed to improve the performance of IoT [15, 16]. IIoT is the industrial branch of IoT using intelligent analysis technologies to achieve industrial intelligence. In [17], several challenges for IIoT, such as real-time performance, energy efficiency, coexistence, interoperability, and security and privacy were discussed. In [11], a survey was presented to show that predictive maintenance (PdM) with IIoT can stimulate the development of PdM and new business opportunities.

Cloud-edge computing. Cloud computing provides strong support for processing big data in IIoT. Edge computing is used for data processing at the edge of the network to handle real-time requirements. For example, in [18], an experiment about mobile gaming was carried out to show that edge computing is necessary to meet the latency requirements. In [9], several typical IoT and IIoT applications benefiting from cloud and edge computing were discussed. A cloud-edge computing framework was proposed in [1] to match the user needs in a local Cyber-physical-social system. And in [19], a new scheme that integrates edge-centric computing and content-centric to improve the service capability of 5G mobile networks was proposed.

Remaining useful life prediction of industrial equipment. The main remaining useful life prediction methods are data preprocessing and model prediction. Data preprocessing contains data normalization, noise removing and feature extraction. Prediction models include machine learning models and deep learning models. Most machine learning methods, such as support vector machine (SVM) [20], take low-dimension features extracted from the original long sequence data as input, which may lead to the loss of useful information in original signals. If the input feature dimension is high, then the machine learning methods will perform poorly.

Approaches based on deep learning can deal with high-dimensional features. The Recurrent Neural Network (e.g.

Long Short-Term Memory [21, 22] and Gate Recurrent Unit [23]) is applied in many methods to predict remaining useful life. However, these methods also try to extract low-dimension features from the original long sequence data and predict RUL with short-term historical information, since the Recurrent Neural Network performs poorly when the input sequence length is long. In recent years, Convolution Neural Network [12, 24] has shown greater capability in processing long sequence data, and it has fewer parameters since it uses the local connectivity and parameter sharing.

A new model named temporal convolutional network (TCN) was proposed to deal with sequence modeling tasks [12]. The causal convolution and dilated convolution used in TCN make it more suitable on sequence data prediction than traditional convolutional neural networks (CNN). However, methods based on TCN for remaining useful life prediction are rarely proposed. Also, the computational time of TCN needs to be further considered. If the input sequence length is long, then TCN will have many more calculations to perform.

In this paper, we propose a new model structure named LTCN (lightweight TCN) based on TCN, and apply two specific LTCNs to the edge and cloud plane in the prediction framework, respectively. In the edge plane, the prediction model will take the preprocessing sequence at a single sampling time as input and obtain a rapid prediction result. In the cloud plane, the model will take the compression features extracted from the edge plane at multiple sampling times as input, and obtain a more accurate and smooth prediction result. To make full use of the newly collected data and avoid the recomputing historical data, we introduce an incremental learning approach in the cloud to improve the accuracy of prediction models.

III. A Cloud-Edge based Framework for Bearing Remaining Useful Life Prediction

In this section, as shown in Figure 2, a cloud-edge based framework for remaining useful life prediction of industrial equipment is proposed. This framework consists of three parts: equipment plane, edge plane and cloud plane. The industrial equipment data will be generated and collected constantly in the equipment plane. The first and second prediction results will be obtained in the edge and cloud plane respectively. The prediction models will be updated through the incremental learning module in the cloud. Next, the processing in the edge and cloud planes will be discussed in detail.

A. Edge plane

The real-time industrial equipment data collected from sensors in the equipment plane will be sent to the edge plane for preprocessing. The prediction model named ST-LTCN will take the preprocessing data at each sampling time as input, and obtain compression features and rapid prediction results. All data and results generated in the edge plane will be sent to the database in the cloud.

B. Cloud plane

The prediction model in the cloud, named MT-LTCN, will take the compression features from the database as input, and obtain a more accurate and smooth prediction result through historical information to improve the prediction accuracy in the edge plane. With the newly collected and existing data in the database, the incremental learning module will continue training the existing models in both edge and cloud planes. Then, the prediction models will be updated if the prediction accuracy improves.

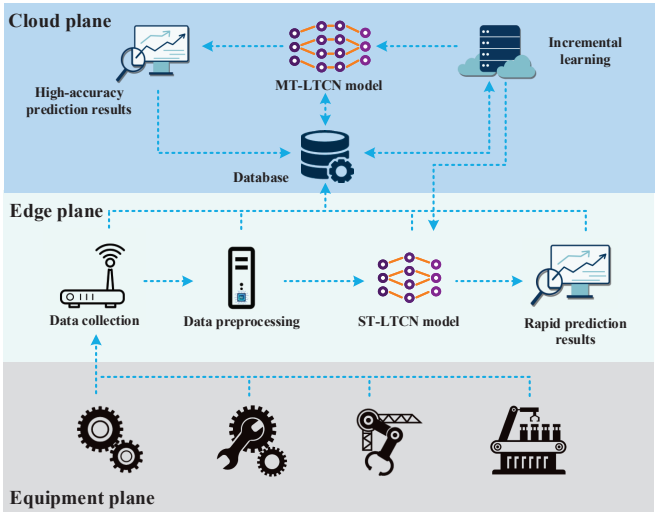

Fig. 2. Cloud-edge computing framework for industrial equipment remaining useful life prediction.

Because of the importance of prediction models in the cloud-edge based framework, the structure of lightweight temporal convolutional network will be discussed first in section IV.

IV. Lightweight Temporal Convolutional Network

In this section, the new model structure named lightweight temporal convolutional network (LTCN) will be presented. The traditional temporal convolutional network (TCN) is used for sequence modeling. Given an input sequence x_0,..., x_T, TCN tries to predict the corresponding outputs y_0,..., y_T at each time. It is constrained to predict the output y_t for some time t by only using the inputs that have been observed: x_0,..., x_t. TCN contains three parts: causal convolutions, dilated convolutions and residual connections. Besides, TCN applies $1D$ fully-convolutional network (FCN) architecture, where each hidden layer is the same length as the input layer. These components not only make the output at time t be convolved only with the elements from time t and earlier in the previous layer, but also make TCN extract features from an exponentially large receptive field.

For the regression task of sequence modeling, only the predicting output $\hat{y_T}$ with filters is needed and applied in TCN, as shown in Figure 3(a). Other predicting outputs $\hat{y_0}$,..., $\hat{y_{T-1}}$ are not used, which causes wasted computations. When the input sequence length is short, this computational waste can be ignored. But when the input sequence length is long, the computational waste will be huge.

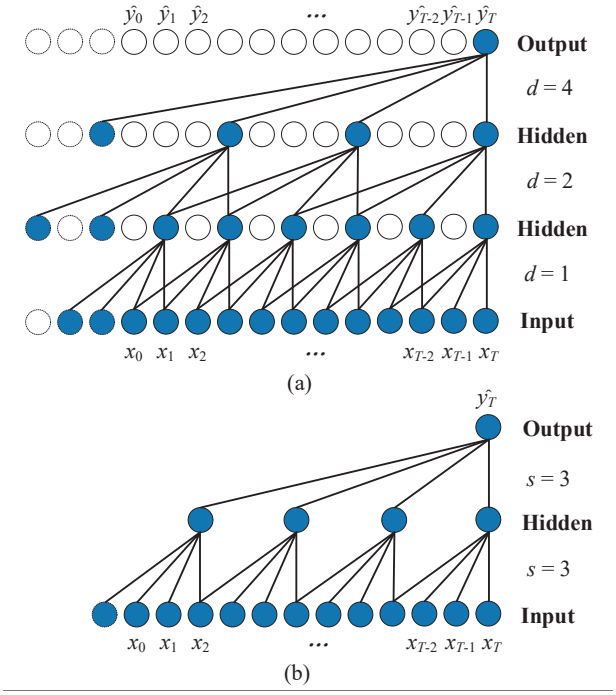

Fig. 3. (a) Dilated causal convolution layer. (b) Stride convolution layer.

Based on TCN, we propose a network structure named lightweight temporal convolutional network (LTCN) with fewer parameters and higher processing speed for regression tasks. LTCN is a function $f : X^{T+1} \rightarrow Y^1$ that produces the mapping

$$\hat{y_T} = f(x_0, ..., x_T). \tag{1}$$

As shown in Figure 3(b), the $1D$ dilated convolutions are replaced with the $1D$ stride convolutions, and the global fully-convolutional network architecture is removed to reduce the amount of computation. All the elements in LTCN are applied in both forward and back propagations. Using a larger stride factor s and kernel size k enables the models to represent a wider range of inputs with less convolution layers.

The frameworks of LTCN and stride convolution blocks are shown in the Figure 4, in which LTCN contains multiple stride convolutional blocks and fully connected layers. Compression features will be extracted through multiple stride convolution blocks and used by the fully connected layers to carry out regression predictions. Within a stride convolution block, LTCN has a layer of $1D$ fully convolution with zero padding of length $(k-1)$ (i.e. causal convolution), a layer of stride convolution with suitable zero padding and ReLU activation functions. The formula of ReLU is shown in Equation 2. Each convolution layer consists of multiple filters. Weight normalization is applied to convolution filters for normalization and dropout for regularization. The ReLU activation function is also used in fully connected layers. Besides, because of the range setting of equipment RUL value in section V, the Sigmoid activation function is applied in the last fully connected layer. The formula of Sigmoid is shown in Equation 3.

$$f(x) = \begin{cases} x & , \text{if } x > 0 \\ 0 & , \text{if } x \leq 0 \end{cases} \tag{2}$$

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

The length of the output through a stride convolution block will be $1/s$ of the length of the input. If the input length of the last stride convolution block is close to the global kernel size, the kernel size in the last stride convolution layer can be set as the length of the input without padding. In this case, the prediction $\hat{y_T}$ could depend on a history of size $k_n \cdot s^{n-1}$, where n is the number of stride convolution blocks and k_n is the kernel size in the last stride convolution block.

Based on lightweight temporal convolutional networks, the RUL prediction method taking bearings as an example will be discussed in detail in the next section.

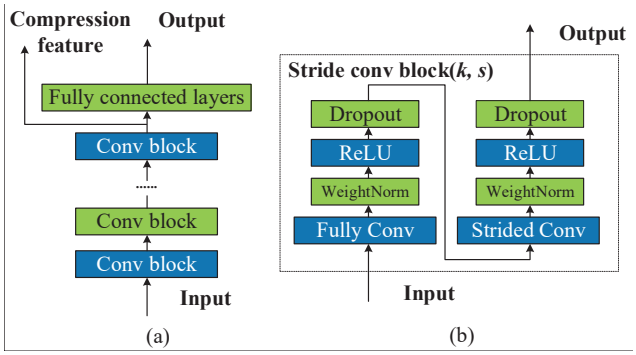

Fig. 4. (a) Framework of lightweight temporal convolutional network. (b) Stride convolution block.

V. RUL PREDICTION METHOD AND INCREMENTAL LEARNING METHOD

In this section, the RUL prediction method with bearings as an example and incremental learning method will be discussed. The target function for bearing RUL prediction is first determined. Then, data preprocessing and model predictions in the edge and cloud plane are discussed, respectively. Next, the parameters of the prediction models are presented. Finally, the incremental learning method based on updating partial parameters are introduced.

A. Bearing RUL prediction method

The detailed process of bearing RUL prediction is shown in Figure 5. After determining the target function, bearing RUL prediction method mainly includes data preprocessing, predictions in the edge plane and predictions in the cloud plane. The five corresponding processes will be discussed next.

1) Target function for bearing RUL prediction: In general, from the signal of full-life bearings in time domain, the early vibration signals are stable and have no obvious vibration changes. The features of early vibration signals are not useful to characterize the degradation performance. Therefore, it is more appropriate to use a piece-wise method to characterize the degradation.

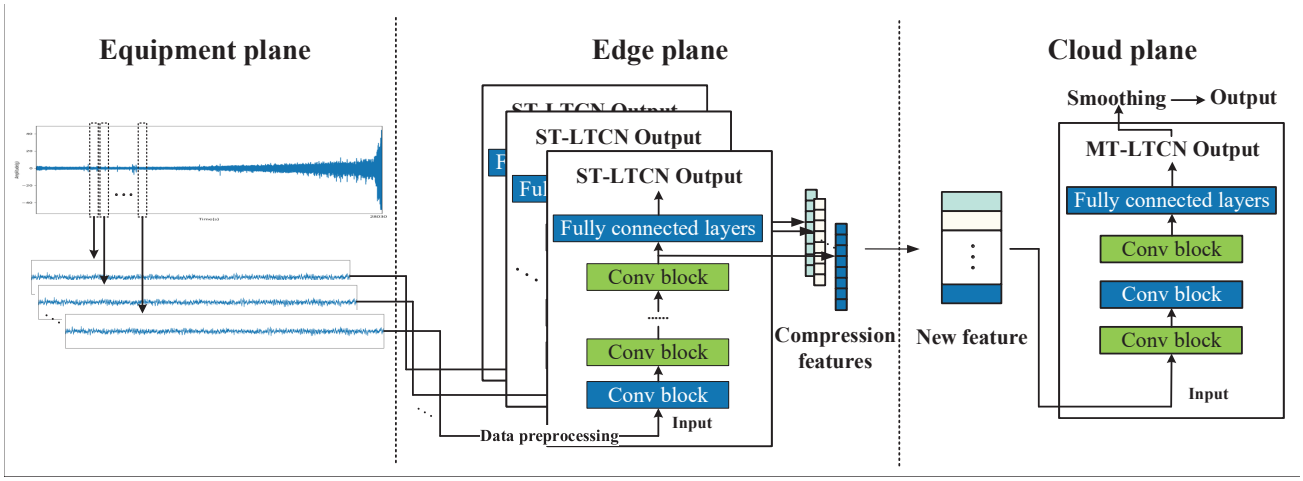

Fig. 5. The detailed process of bearing remaining useful life prediction.

As shown in Figure 6, the piece-wise linear degradation model is applied as the target function rather than linear degradation model in bearing remaining useful life prediction. The degradation of bearings typically begins after a certain time of being used. The value of the RUL is limited from 0 to 1. The segmentation point between the stable phase and linear degradation phase is defined as the first predicting time (FPT). In this paper, the initial time when the vibration signal of the bearing is significantly intensified and maintained is determined to be the FPT of the bearing. If the segmentation point mentioned above is not obvious, the Ahmad method [25] will be used to determine the FPT.

2) Data preprocessing in the edge plane: Since n_s samples are collected for each sampling time t, the original feature of the vibration signal is considered as a length n_s sequence of depth 1 in this paper.

Gaussian noise is often mixed in vibration signals. The wavelet denoising method can be used to reduce the noise[26]. To reduce the influence of Gaussian noise, the wavelet denoising method with Daubechies wavelet of order 4 (db4) is applied during data preprocessing. Level 4 decomposition is employed, and the minimax threshold and soft mode is used in the processing. The vibration signal after denosing is still a length n_s sequence of depth 1.

Besides, due to the approximate symmetry of the bearing data in time domain, the absolute value vibration signals after denoising will be taken to make the features more easily distinguished. After data preprocessing, the length n_s sequences of depth 1 will be obtained.

3) Prediction at single sampling time in the edge plane: At this step, the preprocessing sequence at a single sampling time is taken as the input of LTCN. A single sampling time lightweight temporal convolutional network (ST-LTCN) is trained, which requires (1, n_s) dimension input features. Then, 8-dimension compression features will be extracted through stride convolution blocks. Next, the compression features will be used by fully connected layers of ST-LTCN to carry out regression predictions. In addition, the compression features will be sent to the cloud database and further used by MT-

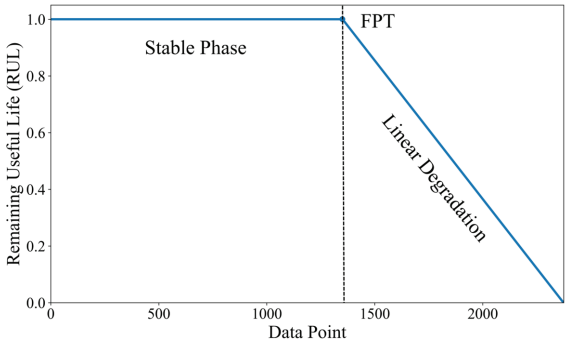

Fig. 6. Piece-wise linear remaining useful life target function. FPT is the first predicting time.

LTCN. For ST-LTCN, the function of multiple convolutional blocks $f_s(\cdot)$ is given in Equation 4, and the fully connected layers function $g_s(\cdot)$ is given in Equation 5.

$$z_t = f_s(x_t^1, ..., x_t^{n_s}), \tag{4}$$

$$\hat{y_t^s} = g_s(z_t), \tag{5}$$

where x_t^i is the ith 1-dimension sample at the time t, z_t is the 8-dimension compression features extracted from multiple convolutional blocks of ST-LTCN at the time t, and $\hat{y_t^s}$ is the prediction RUL of ST-LTCN at the time t.

Due to the input at a single sampling time, the prediction result of ST-LTCN is obtained using no historical information, so the prediction accuracy is not very high. However, the prediction results will be obtained rapidly since each convolutional layer of ST-LTCN has only a small number of filters.

4) Prediction at multiple sampling time in the cloud plane: The 8-dimension compression features extracted by ST-LTCN can represent the degree of bearing degradation. After determining the time-window size T_w, the current 8-dimension compression feature will be combined with (T_w-1) features from the previous (T_w-1) time points, and a new feature that has the shape of (8, T_w) will be obtained. Using the new features as input, the multiple sampling time lightweight temporal convolutional network (MT-LTCN) will get the prediction

result through historical information. The function $f_m(\cdot)$ of MT-LTCN is given in Equation 6.

$$\hat{y_t^m} = f_m(z_t, z_{t-1}..., z_{t-T_w+1}), \tag{6}$$

where z_t is the 8-dimension compression feature extracted from ST-LTCN at the time t and $\hat{y_t^m}$ is the RUL prediction of MT-LTCN at the time t. Since each convolutional layer of MT-LTCN has multiple filters, the prediction results will be obtained slower than using ST-LTCN.

The RUL prediction results is often discontinuous while the actual RUL is continuous. To reduce the influence of discontinuous results on predictions, the forward predictions are used to linearly smooth the current result. The RUL $\hat{y_t^m}$ predicted by MT-LTCN at the time t, will be set as the value regressed linearly by the RUL values at eleven time-points, $t-10$, $t-9$,..., t, as shown in Equation 7.

$$\hat{y_t^l} = f_l(\hat{y_{t-10}^m}, \hat{y_{t-9}^m}..., \hat{y_t^m}). \tag{7}$$

The smoothing will help reduce the fluctuations in the result and make the predictions closer to the real situation.

5) Parameters of the proposed models: The parameters of ST-LTCN and MT-LTCN used in this paper are shown in Table I. Here, n is the number of stride convolution blocks in LTCN. Within one stride convolution block, k represents the kernel size, s is the stride factor, $hidden$ represents the number of filters in each convolutional layer, $dropout$ is the dropout factor. For the fully connected layers, $hidden$ represents the number of neurons in each layer, and $dropout$ is the dropout factor. Each parameters of the stride convolution blocks in ST-LTCN is constant, but they are different in MT-LTCN.

TABLE I
PARAMETERS OF THE PROPOSED METHODS

	Parm	ST-LTCN	MT-LTCN
Stride conv block	n	5	3
	k	10	8,5,5
	s	4	5,2,1
	$hidden$	8	32
	$dropout$	0.05	0.05
Fully connected layer	$hidden$	8,4,1	32,16,8,1
	$dropout$	0.05	0.05

B. Incremental learning in the cloud plane

In general, the existing prediction models which are already trained with sufficient training set can give good obtain bearing RUL prediction results. When the original training set is sufficient, retraining from the begining will take much more time to obtain the optimal models. At the same time, continuous training may lead to overfitting. Both approaches updates all parameters of the models. Because of the high similarity between the newly collected data in practical applications and the existing data in the database, an incremental learning method is proposed. This incremental method is based on partial parameters updating to improve the accuracy in a short time and avoid overfitting.

Suppose that an original sufficient dataset ψ is stored in the cloud database and existing models (i.e. ST-LTCN and MT-LTCN) applied in the edge and cloud planes, respectively, were trained with a sufficient dataset. Then, using the original dataset ψ and newly collected data ϕ, the new training set ψ' and the updated parameters P_f' of prediction models will be obtained through the incremental learning method. The main steps are described in the Algorithm below.

Algorithm Incremental learning method

Input: Original training set ψ, newly collected data ϕ, parameters of stride convolution blocks in prediction model P_s, parameters of fully connected layers in prediction model P_f, learning rate l_r, training epoch e

Output: Training set ψ', parameters of fully connected layers in prediction model P_f'

1: $\psi' \leftarrow \psi + \phi$
2: $P_f' \leftarrow P_f$
3: **for** $e = 1 \rightarrow e$ **do**
4: $\quad P_f' \leftarrow BPalgorithm(\psi, P_s, P_f', l_r)$
5: **return** ψ', P_f'

As shown in the Algorithm and Figure 7, after labeling, the newly collected data will be combined with the original training set to form a new training set. Then, the parameters of stride convolution blocks in ST-LTCN and MT-LTCN will be frozen and not updated in the training process. Through the new training set, the fully connected layers of ST-LTCN and MT-LTCN will be trained with a small learning rate. Parameters of fully connected layers of ST-LTCN and MT-LTCN P_f' will be updated. This process of updating partial parameters is represented as $BPalgorithm(\psi, P_s, P_f', l_r)$ in Algorithm. After the training process, the new training set will be stored in the cloud database for use in future. The parameters of the prediction models in both edge and cloud plane will be updated accordingly.

The incremental learning method is used to obtain the updated parameters of the prediction models and new training set without having to recompute that of the original models. The existing models will be trained continuously with a small learning rate to avoid overfitting. After the training process, the selected subset of existing model parameters will be updated. Due to the sufficiency of the training set and the updating of a subset of parameters, the training time of an epoch and the possibility of overfitting will be greatly reduced.

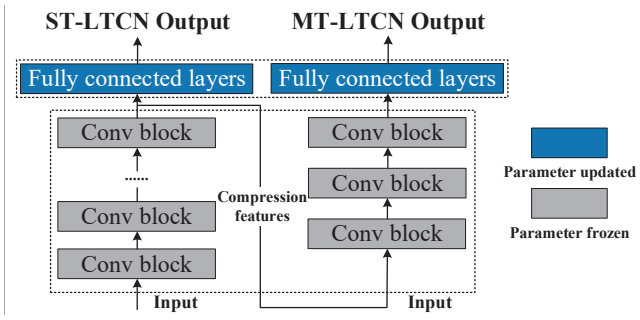

Fig. 7. Incremental learning method based on updating a subset of all model parameters.

VI. EXPERIMENT AND ANALYSIS

In this section, experiments are carried out to measure the performance of the proposed method. The experimental platform, dataset, evaluation metrics, experiment results and analysis will be presented. First, the time-window size of MT-LTCN is determined. Second, the experimental results of ST-LTCN and MT-LTCN for bearing RUL prediction are presented and compared with other models in bearing RUL prediction. Finally, the experimental results of incremental learning method are presented and compared with other methods.

A. Experimental platform and dataset

The training process and testing process both use PyTorch on CPU i5-6300HQ, which is more similar to the actual computing system in practical applications than a GPU. The Adam optimizer is applied and the batch size is 64.

The dataset used in this paper is provided by the PRONOSTIA platform [27]. The real-time vibration signals in both horizontal and vertical directions, which can characterize the degradation performance of bearings, are collected by the experimental equipment. The time of collecting samples is 0.1s, and sampling interval is 10s. The sampling frequency is 25.6kHz, which means that 2560 samples in the horizontal direction and 2560 samples in the vertical direction are collected at each sampling.

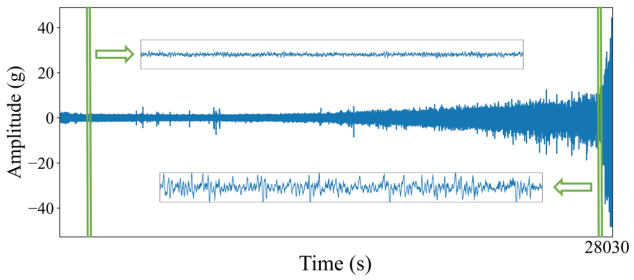

Fig. 8. The full life vibration signal of a bearing.

The dataset contains 17 bearings data under three different operating conditions. A vibration signal of bearing along the whole life is shown in Figure 8. In this paper, the horizontal direction data of 7 bearings under condition 1 (1800rpm and 4000N) is used to carry out the RUL prediction experiment. The FPTs (first predicting times) of bearings are selected as 1297, 825, 1351, 1083, 2411, 2403 and 2207, respectively.

B. Evaluation metrics

In this part, two evaluation metrics used in this paper, will be introduced. First, the root mean square error (RMSE) value between the true RUL and the prediction is used as the evaluation metric for prediction accuracy. The RMSE metric has equal weight for both early and late predictions, and the formula of this metric is shown in Equation 8.

$$RMSE = \sqrt{\frac{1}{N}\Sigma_{i=1}^{N}(y_i - \hat{y}_i)^2}, \tag{8}$$

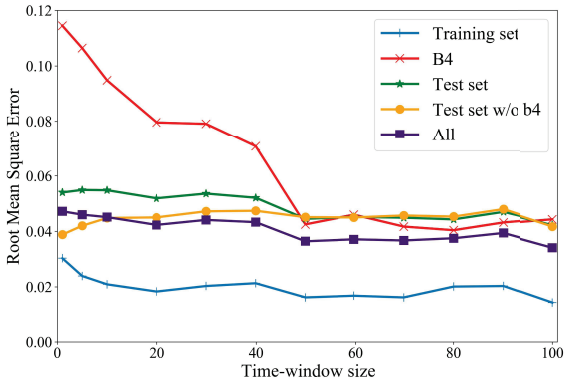

Fig. 9. Time-window size selection.

where y_i and $\hat{y}_i$ are the true and predicted RUL values for sample i respectively, and N is the number of testing samples. Then, the time for processing one sample is selected as the evaluation metric for prediction speed.

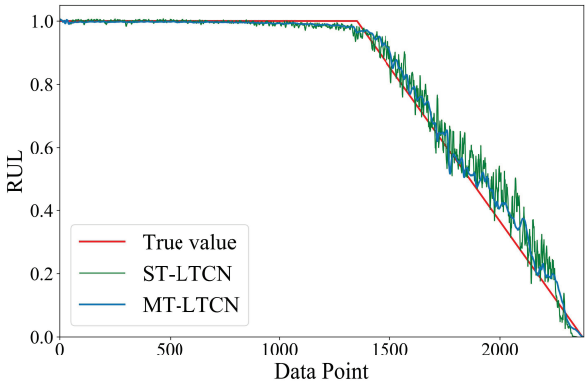

Fig. 10. RUL prediction results of LTCN along the bearing full life.

C. Results and analysis

1) Results of MT-LTCN with different time-window sizes: Since the bearing deteriorates over time, it is essential to select an appropriate time-window size for MT-LTCN. This selected time-window size T_w will make MT-LTCN obtain the optimal prediction results.

In this experiment, the data of bearings 1 and 2 are taken as the training set, and the data of the remaining 5 bearings is taken as the test set. Since the prediction accuracy of bearing 4 is quite different from that of the other bearings in the test set, then the predicted RMSEs for the whole test set and the test set without bearing 4 are both calculated. Figure 9 illustrates the experimental results when different time-window sizes are applied for MT-LTCN. The optimal overall performance for MT-LTCN is obtained when T_w equals to 50. Therefore, the results of MT-LTCN shown in the following experiments are implemented with the optimal T_w.

2) Performance of ST-LTCN and MT-LTCN on bearing RUL prediction: In this experiment, 90 percent data of bearings 1 and 2 are taken as the training set, ten percent of the data as the validation set, and the remaining 5 bearing data as the test

set. ST-LTCN and MT-LTCN are designed, and the prediction accuracy and computing time are determined. The prediction results on bearing 3 is shown in Figure 10.

To validate the effectiveness of the lightweight models, two TCNs [12] are trained. The experimental results are shown in Figure 11(a) and (b) and Table II. The RMSE in Table II is calculated based on the average of all bearing prediction results in the test set, and the cpu time for processing one sample is calculated based on the average of every samples in the test set. It can be seen from the results that MT-LTCN performs better than other models in prediction accuracy, and ST-LTCN performs much better in terms of the model size and prediction speed.

Compared with TCN, LTCN not only can obtain similar prediction accuracy, but also require fewer parameters and shorter prediction time. Considering both the prediction accuracy and computing time, ST-LTCN is suitable for deployment on the edge and MT-LTCN on the cloud.

TABLE II
RESULT COMPARISONS BETWEEN TCN AND LTCN

Methods	RMSE	Model size	Time per sample
ST-TCN	0.0517	131K	10.74ms
ST-LTCN	**0.0540**	**74K**	**0.48ms**
MT-TCN	0.0505	632K	2.57ms
MT-LTCN	**0.0433**	**325K**	**0.93ms**

3) Performance comparisons with different methods: The following experiments compare the performance of LTCN with LSTM [22] and GRU [28]. Since the performances of ST-LSTM and ST-GRU are poor, the 8-dimension compression features extracted by ST-LTCN are applied in the prediction process of MT-LSTM and MT-GRU. The comparison results are shown in Figure 11(c) and (d) and Table III. As shown, it is difficult for ST-LSTM and ST-GRU to deal with length-2560 sequence. ST-LTCN performs much better than ST-LSTM and ST-GRU in terms of prediction accuracy and speed, although the model size is larger. Compared with MT-LSTM and MT-GRU, MT-LTCN has a higher prediction accuracy.

TABLE III
PERFORMANCE COMPARISONS BETWEEN LTCN AND LSTM, GRU

Methods	RMSE	Model size	Time per sample
ST-LSTM	0.1218	21K	3.47ms
ST-GRU	0.0822	19K	2.43ms
ST-LTCN	**0.0540**	**74K**	**0.48ms**
MT-LSTM	0.0539	73K	0.77ms
MT-GRU	0.0566	59K	0.69ms
MT-LTCN	**0.0433**	**325K**	**0.93ms**

A more intuitive comparison of the prediction accuracy and processing time is shown in Figure 12. Considering the metrics of prediction accuracy and processing time, we note that ST-LTCN and MT-LTCN are more suitable for bearing RUL prediction based on the cloud-edge computing framework.

4) The validity of the incremental learning method: In this part, the incremental learning method is compared with retraining from the beginning and continuous training methods to verify the speed and accuracy performance. The experimental condition is first introduced. Next, the training loss and time are presented. Finally, the prediction accuracy using different training approachs is presented.

In this group of experiments, four bearing data are randomly selected as the original training set and two bearing data as the newly collected data in practical application. Ninety percent of the six bearings data is used as the new training set, 10 percent as the validation set, and the remaining bearing data is taken as the test set. The retraining and continuous training methods update all parameters of the predicting models, while our method updates partial parameters. All models are trained for 20 epochs. Each epoch contains 10934 data. The learning rate in incremental learning is 0.0002 while it is 0.001 in the retraining and continuous training methods. The mean square error (MSE) is taken as the training loss function to better show the training trend.

As shown in Figure 13, the training loss of ST-LTCN can only reach about 0.05 while the training loss of MT-LTCN can reach nearly 0, which shows that MT-LTCN can better predict the RUL. The training loss of retraining ST-LTCN is larger than other methods, which means more time is needed for retraining. The difference between the training loss of our method and the continuous training method is very small. However, the training time of an epoch using our method for ST-LTCN and MT-LTCN is 8.45s and 4.67s respectively, while the training time of an epoch updating all parameters is 19.78s and 7.86s. That is, the training time using our incremental learning method is much shorter.

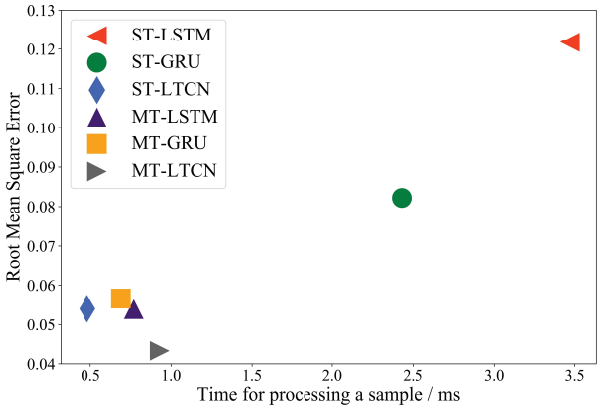

Fig. 12. Performance comparisons of RMSE and time-consumption.

For showing the global prediction results of the dataset, the RMSEs based on the average of training set, test set and all data are calculated. The prediction accuracy of ST-LTCN and MT-LTCN using different training methods now presented. As shown in Table IV and Figure 14, the performance of the retraining method on both training and test sets is very poor. The continuous training method updating all parameters performs well on the training set, but it is easy to cause overfitting, which can be seen in the performance on test set. However, the incremental learning method proposed in this paper performs well on both training and test sets. The optimal overall performance is obtained when ST-LTCN and MT-LTCN are trained with our method.

In general, the incremental learning method can not only reduce the training time and the possibility of overfitting,

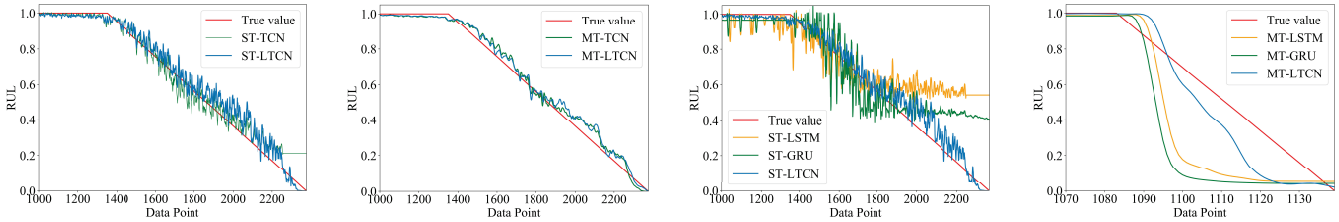

Fig. 11. Prediction result comparisons between (a) ST-TCN and ST-LTCN; (b) MT-TCN and MT-LTCN; (c) ST-LSTM, ST-GRU and ST-LTCN, and (d) MT-LSTM, MT-GRU and MT-LTCN.

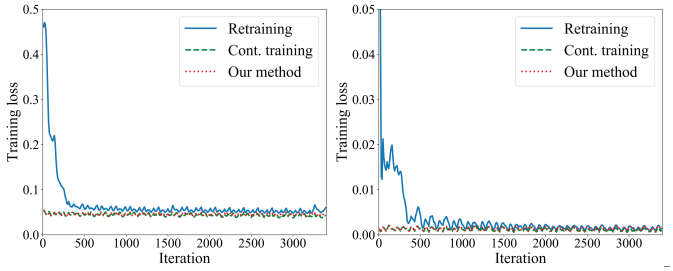

Fig. 13. (a) Training loss comparisons of ST-LTCN between retraining, continuous training and our method. (b) Training loss comparisons of MT-LTCN between retraining, continuous training and our method.

but also give the optimal overall performance to improve the prediction accuracy. Compared with other methods, our method has an advantage in the industrial equipment RUL prediction framework that requires high real-time performance.

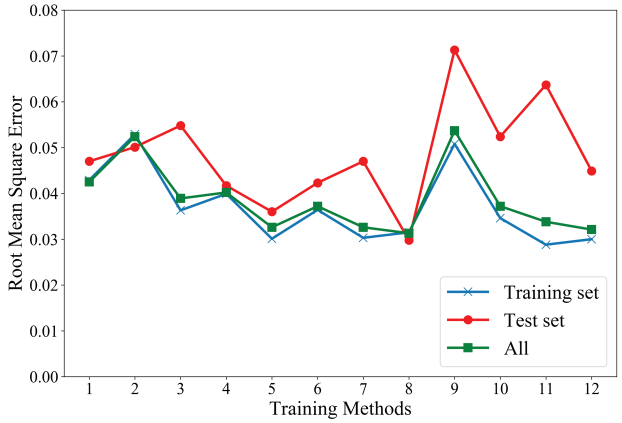

Fig. 14. Performance comparisons of different training methods.

VII. CONCLUSION

In this paper, we propose a cloud-edge computing and artificial intelligence based framework, which can support the rapid and accurate RUL prediction of industrial equipment. Considering the high latency of cloud server and the limited computation resources of edge devices, a new model structure named lightweight temporal convolutional network (LTCN) is proposed, and two specific LTCNs (i.e. single sampling time LTCN and multiple sampling time LTCN) are applied in the edge and cloud planes respectively. Besides, to avoid recomputing of historical data, an incremental learning approach based on updating a subset of all parameters of LTCN is presented to improve the accuracy of prediction models in a short time. In future, we plan to compare the performance of our models using different feature extraction methods.

TABLE IV
PERFORMANCE COMPARISONS BETWEEN INCREMENTAL LEARNING AND RETRAINING

	Methods		RMSE		
Idx	ST-LTCN	MT-LTCN	Training set	Test set	All
1	Origin	-	0.0429	0.0470	0.0425
2	Retraining	-	0.0529	0.0501	0.0524
3	Cont. training	-	0.0363	0.0548	0.0389
4	**Our method**	**-**	**0.0399**	**0.0417**	**0.0402**
5	Our method	Origin	0.0301	0.0360	0.0326
6	Our method	Retraining	0.0364	0.0423	0.0372
7	Our method	cont. training	0.0303	0.0470	0.0326
8	**Our method**	**Our method**	**0.0315**	**0.0298**	**0.0313**
9	Cont. training	Origin	0.0508	0.0713	0.0537
10	Cont. training	Retraining	0.0346	0.0524	0.0372
11	Cont. training	Cont. training	0.0288	0.0637	0.0338
12	**Cont. training**	**Our method**	**0.0300**	**0.0449**	**0.0321**

VIII. ACKNOWLEDGMENT

The research is supported by The National Key Research and Development Program of China No. 2019YFB1705903, and the NSFC (National Science Foundation of China) project No. 61572057 and 61836001.

REFERENCES

[1] X. Wang, L. T. Yang, X. Xie, J. Jin, and M. J. Deen, "A cloud-edge computing framework for cyber-physical-social services," *IEEE Communications Magazine*, vol. 55, no. 11, pp. 80–85, 2017.

[2] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2018.

[3] X. Wang, W. Wang, L. T. Yang, S. Liao, D. Yin, and M. J. Deen, "A distributed hosvd method with its incremental computation for big data in cyber-physical-social systems," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 2, pp. 481–492, 2018.

[4] X. Wang, L. T. Yang, L. Song, H. Wang, L. Ren, and M. J. Deen, "A tensor-based multi-attributes visual feature recognition method for industrial intelligence," *IEEE Transactions on Industrial Informatics*, DOI: 10.1109/TII.2020.2999901, 2020.

[5] L. Ren, Z. Meng, X. Wang, L. Zhang, and L. T. Yang, "A data-driven approach of product quality prediction for complex production systems," *IEEE Transactions on Industrial Informatics*, DOI: 10.1109/TII.2020.3001054, 2020.

[6] J. A. Stankovic, "Research directions for the internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 3–9, 2014.
[7] S. Chen, H. Xu, D. Liu, B. Hu, and H. Wang, "A vision of iot: Applications, challenges, and opportunities with china perspective," *IEEE Internet of Things Journal*, vol. 1, no. 4, pp. 349–359, 2014.
[8] L. Ren, L. Zhang, L. Wang, F. Tao, and X. Chai, "Cloud manufacturing: key characteristics and applications," *International Journal of Computer Integrated Manufacturing*, vol. 30, no. 6, pp. 501–515, 2017.
[9] J. Pan and J. McElhannon, "Future edge cloud and edge computing for internet of things applications," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 439–449, 2017.
[10] L. Ren, Y. Laili, X. Li, and X. Wang, "Coding-based large-scale task assignment for industrial edge intelligence," *IEEE Transactions on Network Science and Engineering*, 2019.
[11] M. Compare, P. Baraldi, and E. Zio, "Challenges to iot-enabled predictive maintenance for industry 4.0," *IEEE Internet of Things Journal*, 2019.
[12] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.
[13] H. Li, K. Ota, and M. Dong, "Deep reinforcement scheduling for mobile crowdsensing in fog computing," *ACM Transactions on Internet Technology (TOIT)*, vol. 19, no. 2, pp. 1–18, 2019.
[14] Y. Zuo, Y. Wu, G. Min, C. Huang, and K. Pei, "An intelligent anomaly detection scheme for micro-services architectures with temporal and spatial data analysis," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 2, pp. 548–561, 2020.
[15] L. Wang, Y. Meng, H. Zhu, M. Tang, and K. Ota, "Edge-assisted stream scheduling scheme for the green-communication-based iot," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 7282–7292, 2019.
[16] Y. Zuo, Y. Wu, G. Min, and L. Cui, "Learning-based network path planning for traffic engineering," *Future Generation Computer Systems*, vol. 92, pp. 59–67, 2019.
[17] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, 2018.
[18] G. Premsankar, M. Di Francesco, and T. Taleb, "Edge computing for the internet of things: A case study," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1275–1284, 2018.
[19] H. Li, K. Ota, and M. Dong, "Eccn: Orchestration of edge-centric computing and content-centric networking in the 5g radio access network," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 88–93, 2018.
[20] T. H. Loutas, D. Roulias, and G. Georgoulas, "Remaining useful life estimation in rolling bearings utilizing data-driven probabilistic e-support vectors regression," *IEEE Transactions on Reliability*, vol. 62, no. 4, pp. 821–832, 2013.
[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[22] S. Zheng, K. Ristovski, A. Farahat, and C. Gupta, "Long short-term memory network for remaining useful life estimation," in *2017 IEEE International Conference on Prognostics and Health Management*, Ottawa, Canada, Jun 2017, pp. 88–95.
[23] L. Ren, X. Cheng, X. Wang, J. Cui, and L. Zhang, "Multi-scale dense gate recurrent unit networks for bearing remaining useful life prediction," *Future Generation Computer Systems*, vol. 94, pp. 601–609, 2019.
[24] X. Li, Q. Ding, and J.-Q. Sun, "Remaining useful life estimation in prognostics using deep convolution neural networks," *Reliability Engineering & System Safety*, vol. 172, pp. 1–11, 2018.
[25] W. Ahmad, S. A. Khan, and J.-M. Kim, "A hybrid prognostics technique for rolling element bearings using adaptive predictive models," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1577–1584, 2017.
[26] H. Bendjama, S. Bouhouche, and M. S. Boucherit, "Application of wavelet transform for fault diagnosis in rotating machinery," *International Journal of Machine Learning and Computing*, vol. 2, no. 1, pp. 82–87, 2012.
[27] N. Patrick, G. Rafael, M. Kamal *et al.*, "Pronostia: an experimental platform for bearings accelerated life test," in *IEEE International Conference on Prognostics and Health Management*, Denver, Colorado, USA, Jun 2012, pp. 1–8.
[28] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

Lei Ren (M'17-) is an associate professor and the deputy head of Cloud Manufacturing Research Center at School of Automation Science and Electrical Engineering, Beihang University, and a senior research scientist at Engineering Researching Center of Complex Product Advanced Manufacturing Systems, Ministry of Education, China. His research interests include Industrial Big Data, Industrial AI and Cloud Manufacturing.

Yuxin Liu received the B.Eng. Degree in automation engineering from Beihang University, China, in 2019, where he is currently pursuing the M.S. degree with the School of Automation Science and Electrical Engineering. His current research interests include Deep Learning, Big Data and Edge Computing.

Xiaokang Wang (M'18-) received the Ph.D degree in Computer System Architecture in Huazhong University of Science and Technology, Wuhan, China, in 2017. Currently, he is a Post-Doctoral Fellow with Department of Computer Science, St. Francis Xavier University, Canada. His research interests are Cyber-Physical-Social Systems, Big Data, Parallel and Distributed Computing, and Cloud-Edge Computing.

Jinhu Lü (M'03-SM'06-F'13) is currently the Dean of the School of Automation Science and Electrical Engineering, Beihang University, Beijing, China. He was an Editor in various ranks for 15 SCI journals, including seven IEEE Transactions journals. His research interests include Complex Networks, Multiagent Systems, and Big Data.

M. Jamal Deen is currently Distinguished University Professor, Senior Canada Research Chair in Information Technology, and President of the Academy of Science, Royal Society of Canada (RSC). His research records include more than 500 peer-reviewed articles and two textbooks "Silicon Photonics-Fundamentals and Devices", and "Fiber Optic Communications-Fundamentals and Applications". His current research interests include Nano-/opto-electronics, Nanotechnology, and their emerging applications in health and environment. He was elected to Fellow status in ten national academies and professional societies, including RSC, IEEE, APS, ECS, and AAAS.