Neurocomputing 379 (2020) 117–129

Contents lists available at ScienceDirect

Neurocomputing

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

Recurrent convolutional neural network: A new framework for remaining useful life prediction of machinery

Biao Wang[a], Yaguo Lei[a,*], Tao Yan[a], Naipeng Li[a], Liang Guo[b]

[a] Key Laboratory of Education Ministry for Modern Design & Rotor-Bearing System, Xi'an Jiaotong University, Xi'an 710049, China
[b] School of Mechanical Engineering, Southwest Jiaotong University, Chengdu 610031, China

ARTICLE INFO

Article history:
Received 25 June 2019
Revised 14 September 2019
Accepted 25 October 2019
Available online 31 October 2019

Communicated by Hongli Dong

Keywords:
Deep learning
Convolutional neural network
Recurrent connection
Remaining useful life prediction
Uncertainty quantification

ABSTRACT

Deep learning is becoming more appealing in remaining useful life (RUL) prediction of machines, because it is able to automatically build the mapping relationship between the raw data and the corresponding RUL by representation learning. Among deep learning models, convolutional neural networks (CNNs) are gaining special attention because of its powerful ability in dealing with time-series signals, and have achieved promising results in current studies. These studies, however, suffer from the two limitations: (1) The temporal dependencies of different degradation states are not considered during network construction; and (2) The uncertainty of RUL prediction results cannot be obtained. To overcome the above-mentioned limitations, a new framework named recurrent convolutional neural network (RCNN) is proposed in this paper for RUL prediction of machinery. In RCNN, recurrent convolutional layers are first constructed to model the temporal dependencies of different degradation states. Then, variational inference is used to quantify the uncertainty of RCNN in RUL prediction. The proposed RCNN is evaluated using vibration data from accelerated degradation tests of rolling element bearings and sensor data from life testing of milling cutters, and compared with some state-of-the-art prognostics approaches. Experimental results demonstrate the effectiveness and superiority of RCNN in improving the accuracy and convergence of RUL prediction. More importantly, RCNN is able to provide a probabilistic RUL prediction result, which breaks the inherent limitation of CNNs and facilitates maintenance decision making.

1. Introduction

In modern industries, remaining useful life (RUL) prediction is being emerged as a critical technique for ensuring the safety, availability and efficiency of machinery and reducing maintenance costs [1–4]. There are two major categories for existing RUL prediction methods, i.e., model-based methods and data-driven methods [5]. Model-based methods use first principles and failure mechanisms to model the degradation processes of machines, and then employ statistical estimation techniques, such as linear least squares, maximum likelihood estimation and sequential Monte Carlo, to identify model parameters and predict the RUL. Model-based methods may work well when the developed models are able to accurately describe the degradation processes of machines. However, it is quite difficult to build an accurate degradation model in practice, because the failure mechanisms of machines are generally various or not straightforward. Data-driven methods, on the contrary, do not need to know the explicit failure mechanisms of machines. They use available sensor data coupled with machine learning (ML) techniques [6], e.g., support vector machine (SVM), fuzzy neural network and Gaussian process regression (GPR), to learn the degradation characteristics of machines and then employ these learned ML models to estimate the RUL. As a result, data-driven methods are more suitable and easier to be deployed in industrial applications, and many related studies have been carried out over the past few years. Soualhi et al. [7] extracted sensitive features from vibration signals using Hilbert-Huang transform, and these features were fed into SVM to predict RUL of rolling element bearings. Ali et al. [8] proposed a prognostics methodology based on artificial neural network (ANN) for RUL prediction of pump bearings, in which Weibull distribution was used to fit the extracted features, and ANN was employed to perform RUL estimation tasks. Lim et al. [9] presented a Time Series-Histogram of Features method to extract features exhibiting local degradation trends from sensor data of turbofan engines, and utilized multi-layer perceptron to predict the RUL. Wu et al. [10] developed a random forest-based prognostics method for cutting tools, where 28 statistical features were extracted from cutting force, vibration and acoustic emission signals, and random forest was trained using these extracted features to

* Corresponding author.
E-mail address: yaguolei@mail.xjtu.edu.cn (Y. Lei).

https://doi.org/10.1016/j.neucom.2019.10.064
0925-2312/

estimate the tool wear depth. Aye et al. [11] constructed a health indicator by combining principal component analysis with Gaussian mixture model, and developed an integrated GPR model for predicting RUL of slow speed bearings.

Recently, deep learning is becoming more appealing in data-driven RUL prediction. Deep learning is a class of ML techniques that is composed of multiple layers of nonlinear processing units [12]. Compared with traditional ML techniques, deep learning has more powerful representation learning ability and is able to automatically learn multi-level representations from raw data [13,14]. Therefore, with the aid of deep learning techniques, prognostics models can be directly established based on raw sensor data, which gets rid of the complex processes of manual feature extraction [15]. Among deep learning techniques, convolutional neural networks (CNNs) [16] are gaining special attention in RUL prediction of machines because of its ability in dealing with time-series signals, and have achieved promising results in several recent studies. Babu et al. [17] presented a CNN-based prognostics method for RUL prediction of turbofan engines and demonstrated its superiority compared with the traditional ML methods. Ren et al. [18] applied CNN into RUL prediction of bearings, in which a 13-layer CNN was built and then trained using 64 frequency-domain features. Wen. et al. [19] proposed a residual CNN ensemble and utilized it to estimate RUL of turbofan engines. Zhu et al. [20] combined wavelet transform and CNN to predict the bearing RUL, where wavelet transform was first used to extract the time-frequency features and then a multi-scale CNN was employed to estimate the RUL. Similarly, Li et al. [21] used short-time Fourier transform to obtain time-frequency features of vibration signals, and these features were input to CNN to predict the RUL of bearings. Through analyzing these studies, it can be found that current CNN-based RUL prediction methods suffer from the following two limitations.

(1) The temporal dependencies of different degradation states are not considered. During operations, machines degrade over time from normal operating states to total failures, and the development of faults is also a gradual evolution process [1]. Correspondingly, the degradation states of a machine at different time instances are correlative on the time scale. Thus, it is important to model contextual dependencies of different degradation states based on time-series sensor data for accurate RUL prediction. Existing CNN-based studies, however, neglect this dependency during network construction, thereby affecting the accuracy of the prognostics models and limiting their generalization.
(2) The uncertainty of RUL prediction results cannot be obtained. The uncertainty quantification is necessary for prognostics, because it is able to know the confidence of RUL prediction results and then facilitate maintenance decision making [22]. However, current CNN-based prognostics models are built based on classic CNN architectures, which only provide a point estimation rather than a probabilistic distribution in RUL prediction of machines. Consequently, these methods are unable to quantify the prediction uncertainty, increasing the risks in maintenance decision making.

To address the above-mentioned limitations, in this paper, we propose a new framework named recurrent convolutional neural network (RCNN) for RUL prediction of machinery. In RCNN, recurrent convolutional layers are first constructed to model the temporal dependencies of different degradation states. Then, variational inference is used to quantify the uncertainty of RCNN in RUL prediction. The proposed RCNN is evaluated on accelerated degradation datasets of rolling element bearings and life testing datasets of milling cutters, and also compared with some existing prognostics approaches. Experimental results demonstrate the effectiveness and superiority of the proposed RCNN in improving RUL prediction accuracy and quantifying uncertainty in prognostics. The main contributions of this paper are summarized as follows: (1) A new core building block, i.e., recurrent convolutional layer, is built to learn the temporal dependencies from time-series sensor data, which enables the prognostics network to effectively memorize useful degradation information over time and thus enhances its representation ability; and (2) A probabilistic RUL prediction result is obtained based on variational inference, which overcomes the inherent limitation of CNNs and is beneficial to maintenance decision making.

The rest of this paper is organized as follows. Section 2 describes the proposed RCNN for RUL prediction of machinery in detail. In Section 3 and Section 4, the effectiveness and superiority of the proposed RCNN are demonstrated using vibration data from accelerated degradation tests of rolling element bearings and sensor data from life testing of milling cutters, respectively. Finally, Section 5 concludes this paper.

2. RCNN for RUL prediction of machinery

In this section, we first detailedly describe the proposed RCNN, where a new core building block, called recurrent convolutional layer, is constructed to model the temporal dependencies of different degradation states of a machine. Then, the prediction uncertainty of RCNN is quantified based on variational inference.

2.1. Proposed RCNN

2.1.1. Recurrent convolutional layer

In CNNs, convolutional layers are the core building blocks and are able to automatically extract discriminative features from the input time-series sensor data. However, in a convolutional layer, there is no cycle or loop that is able to feed the output of a layer back into the input. This means that information only flows in the forward direction in CNNs. Correspondingly, at each time step, CNNs only consider the current input and neglects the previous degradation information. Therefore, CNNs are unable to model the contextual dependencies of different machine degradation states. In particular, this weakness in CNNs is not considered and addressed in existing CNN-based prognostics approaches, which affects their prediction accuracy and generalization. To solve such an issue and boost the prediction performance of networks, a new core building block, i.e., recurrent convolutional layer, is built in this paper.

Different from the convolutional layer that passes on the information in only one direction, the recurrent convolutional layer attaches a recurrent connection between the output and the input, forming the looping of information. Through the recurrent connection, the output of the recurrent convolutional layer is fed back into its input, and thus the recurrent convolutional layer is able to memorize information over time. Accordingly, the output of the recurrent convolutional layer not only depends on the current input, but also depends on the previous stored state that maintains a memory about history of all past inputs. This temporal dynamic behavior enables the recurrent convolutional layer to make full use of the information coming from the input time-series sensor data and to model the temporal dependencies of different degradation states. Formally, for the ith recurrent convolutional layer, its state $\boldsymbol{x}_t^i$ at time step t can be written as

$$\boldsymbol{x}_t^i = f\left(\boldsymbol{x}_t^{i-1}, \boldsymbol{h}_{t-1}^i\right) \tag{1}$$

where $f(\cdot)$ is the nonlinear activation function, such as sigmoid, tanh and rectified linear unit (ReLU), $\boldsymbol{x}_t^{i-1}$ is the input volume, i.e., the input time-series sensor data or the feature maps in the previous layer $i-1$, and $\boldsymbol{h}_{t-1}^i = \boldsymbol{x}_{t-1}^i$ is the stored state at time step t-1 that is fed back by the recurrent connection.

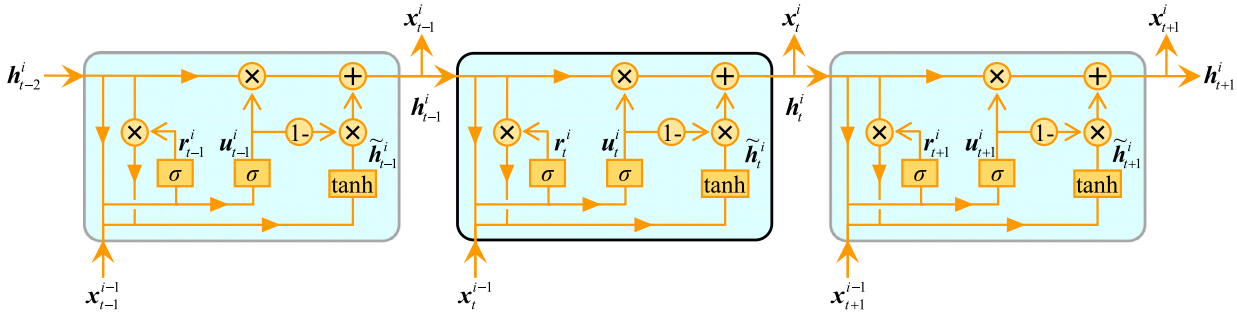


Fig. 1. An illustration of the gating mechanism in the recurrent convolutional layer.

Theoretically, recurrent connections are able to make the recurrent convolutional layer learn arbitrary long-term dependencies from the input sensor data. However, in practice, the recurrent convolutional layer is limited to looking back only a few time steps, because it often suffers from the problem of vanishing or exploding gradients during training. Therefore, to alleviate the effect of vanishing and exploding gradients and capture long-term dependencies, a gating mechanism [23] is introduced into the recurrent convolutional layer. As shown in Fig. 1, there are two gates in the gated recurrent convolutional layer, i.e., reset gate $\boldsymbol{r}_t^i$ and update gate $\boldsymbol{u}_t^i$, and they can be respectively calculated by

$$\boldsymbol{r}_t^i = \sigma\left(\boldsymbol{K}_r^i * \boldsymbol{x}_t^{i-1} + \boldsymbol{W}_r^i * \boldsymbol{h}_{t-1}^i + \boldsymbol{b}_r^i\right) \tag{2}$$

$$\boldsymbol{u}_t^i = \sigma\left(\boldsymbol{K}_u^i * \boldsymbol{x}_t^{i-1} + \boldsymbol{W}_u^i * \boldsymbol{h}_{t-1}^i + \boldsymbol{b}_u^i\right) \tag{3}$$

where $\sigma(\cdot)$ is the logistic sigmoid function, $*$ denotes the convolution operator, $\boldsymbol{K}_r^i$, $\boldsymbol{W}_r^i$ $\boldsymbol{K}_u^i$ and $\boldsymbol{W}_u^i$ are the convolution kernels, and $\boldsymbol{b}_r^i$ and $\boldsymbol{b}_u^i$ are the bias terms. At each time step t, the state $\boldsymbol{x}_t^i$ of the gated recurrent convolutional layer can be obtained by

$$\boldsymbol{x}_t^i = \boldsymbol{u}_t^i \circ \boldsymbol{h}_{t-1}^i + \left(1 - \boldsymbol{u}_t^i\right) \circ \tilde{\boldsymbol{h}}_t^i \tag{4}$$

$$\tilde{\boldsymbol{h}}_t^i = \tanh\left(\boldsymbol{K}_h^i * \boldsymbol{x}_t^{i-1} + \boldsymbol{W}_h^i * \left(\boldsymbol{r}_t^i \circ \boldsymbol{h}_{t-1}^i\right) + \boldsymbol{b}_h^i\right) \tag{5}$$

where $\circ$ denotes the Hadamard product, $\tilde{\boldsymbol{h}}_t^i$ is the newly generated state, $\tanh(\cdot)$ is the tanh activation function, $\boldsymbol{K}_h^i$ and $\boldsymbol{W}_h^i$ are the convolution kernels, and $\boldsymbol{b}_h^i$ is the bias term. From Eqs. (4) and (5), it can be seen that the state $\boldsymbol{x}_t^i$ at time step t is a linear combination of the previous state $\boldsymbol{h}_{t-1}^i$ and the current candidate state $\tilde{\boldsymbol{h}}_t^i$, and is controlled by the reset and update gates.

By introducing the gating mechanism, the recurrent convolutional layer has the ability to forget or emphasize the previous and current information. On the one hand, the reset gate $\boldsymbol{r}_t^i$ is able to decide how much the past information will be forgot. For example, in Eq. (5), if the reset gate $\boldsymbol{r}_t^i$ is close to 0, the current candidate state $\tilde{\boldsymbol{h}}_t^i$ will be forced to ignore the previous state $\boldsymbol{h}_{t-1}^i$ and be reset with the current input $\boldsymbol{x}_t^{i-1}$ only. This will effectively allow the network to forget some previous irrelevant information and to learn a more compact representation. On the other hand, the update gate $\boldsymbol{u}_t^i$ controls how much information from the previous state will pass to the current state. This is able to help the network to remember long-term information and to eliminate the issue of vanishing gradients. Moreover, it should be noted that since each feature map in the recurrent convolutional layer has separate reset and update gates, it is able to adaptively capture the dependencies over different time scales [23]. If the reset gates are frequently active, the corresponding feature maps will learn to capture the short-term dependencies or focus on the current input only. On the contrary, if the update gates of feature maps are often active, they will capture the long-term dependencies.

2.1.2. Pooling layer and fully-connected layer

Except for the recurrent convolutional layers, pooling layers and fully-connected layers are also employed in the proposed RCNN. The use of pooling layers is to reduce the dimensionality of representations and make the extracted features compact. In RCNN, a pooling layer is placed after a recurrent convolutional layer, and pooling operation is independently conducted on every input feature map and outputs a summary statistic of a local patch of the previous feature map. Especially, this operation creates an invariance to small shifts and distortions, which greatly improves the statistical efficiency of the network. In the ith pooling layer, the state $\boldsymbol{y}_t^{ij}$ of the j-th feature map at time step t can be obtained by

$$\boldsymbol{y}_t^{ij} = \text{pool}\left(\boldsymbol{x}_t^{ij}, p, s\right) \tag{6}$$

where pool$(\cdot)$ is the down-sampling function, such as max pooling, average pooling and L^2-norm pooling, p is the pooling size, and s is the stride size.

The use of fully-connected layers is to perform high-level reasoning and regression analysis. The fully-connected layer is placed at the end of RCNN and is used as the output layer to estimate the RUL. In a fully-connected layer, each neuron is fully connected to all neurons in the previous layer, which is the same as the traditional multi-layer perceptron in principle. For the ith fully-connected layer, the output $\boldsymbol{z}_t^i$ at time step t can be computed by

$$\boldsymbol{z}_t^i = f\left(\boldsymbol{z}_t^{i-1}\right) = f\left(\boldsymbol{W}_f^i \boldsymbol{z}_t^{i-1} + \boldsymbol{b}_f^i\right) \tag{7}$$

where $\boldsymbol{z}_t^{i-1}$ is the input of the ith fully-connected layer, i.e., the output of the previous layer $i-1$, $\boldsymbol{W}_f^i$ is the weight matrix, and $\boldsymbol{b}_f^i$ is the bias vector.

2.1.3. Architecture of RCNN

The architecture of the proposed RCNN is shown in Fig. 2, which is formed by a stack of recurrent convolutional layers (denoted as RCL), pooling layers (denoted as PL) and fully-connected layers (denoted as FCL). In the proposed RCNN, to integrate the degradation information from different sensors, multi-channel time-series sensor data of size $H \times 1 \times C$ are used as the inputs of the prognostics network, where H is the length of each sensor sequence, and C is the number of sensors. Then, N recurrent convolutional layers and N pooling layers are used to automatically learn multiple levels of representations from the input sensor data and model the temporal dependencies of different degradation states. For the i-the recurrent convolutional layer, where $i = 1, 2, \ldots, N$, all of the convolution kernels have the same parameter settings, i.e., the number of kernels is set to be $2^{(i-1)}M$ and the kernel size is $k \times 1$. For the first $N-1$ pooling layers, the max pooling is used as the down-sampling function, and the pooling operation is performed using a non-overlapping window, i.e., $p = s$. The final pooling layer, i.e., the Nth pooling layer, utilizes a global max

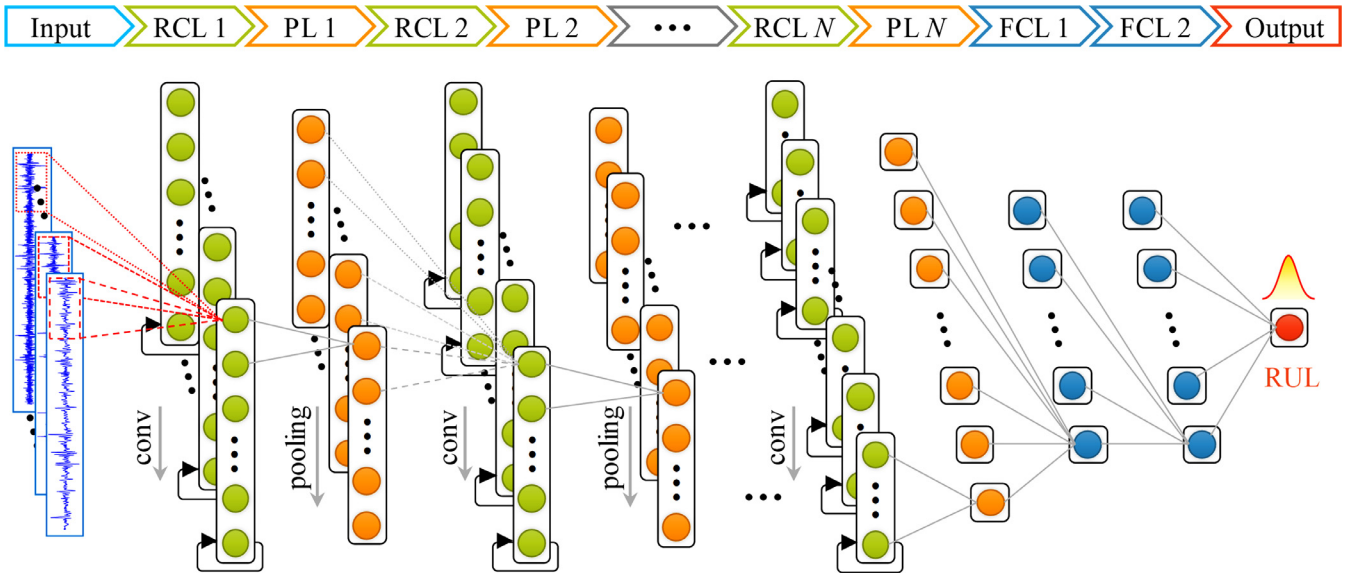


Fig. 2. Architecture of the proposed RCNN.

pooling to conduct the down-sampling, and correspondingly the high-level representations from the N-th recurrent convolutional layer are transformed into a vector of size $2^{(N-1)}M$. After that, this vector is fed into subsequent L fully-connected layers to perform RUL estimation. In this paper, L is set to be 3. That is, there are 3 fully-connected layers in RCNN. Each of the first two fully-connected layers has F neurons, and both of them adopt ReLU to implement nonlinear activation. The third fully-connected layer has only one neuron and is used as the output layer of RCNN to predict the RUL.

2.2. Uncertainty quantification for RCNN

The uncertainty quantification of deep prognostics networks plays an important role in maintenance decision making. In this paper, we employ variational inference to quantify the uncertainty of RCNN in prognostics. The proposed RCNN can be viewed as a probabilistic model with random variables $\boldsymbol{\omega}$ following Gaussian prior distribution. The random variables $\boldsymbol{\omega}$ are composed of all learnable network parameters, including the convolution kernels and biases in the recurrent convolutional layers $\boldsymbol{\omega}_{\mathrm{RCL}} = \{\boldsymbol{K}_r^i, \boldsymbol{W}_r^i, \boldsymbol{b}_r^i, \boldsymbol{K}_u^i, \boldsymbol{W}_u^i, \boldsymbol{b}_u^i, \boldsymbol{K}_h^i, \boldsymbol{W}_h^i, \boldsymbol{b}_h^i\}_{i=1}^N$ and the weights and biases in the fully-connected layers $\boldsymbol{\omega}_{\mathrm{FCL}} = \{\boldsymbol{W}_f^i, \boldsymbol{b}_f^i\}_{i=1}^L$. Given a training dataset consisting of T inputs $\boldsymbol{X} = \{\boldsymbol{x}_t\}_{t=1}^T$ and their corresponding outputs $\boldsymbol{O} = \{o_t\}_{t=1}^T$, the posterior distribution of $\boldsymbol{\omega}$ can be obtained by Bayes' theorem, i.e.,

$$p(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{O}) = \frac{p(\boldsymbol{O}|\boldsymbol{X}, \boldsymbol{\omega})p(\boldsymbol{\omega})}{p(\boldsymbol{O}|\boldsymbol{X})} \tag{8}$$

Based on the above equation, a predictive distribution for a new input $\boldsymbol{x}^*$ can be got by

$$p(o^*|\boldsymbol{x}^*, \boldsymbol{X}, \boldsymbol{O}) = \int p(o^*|\boldsymbol{x}^*, \boldsymbol{\omega})p(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{O})\mathrm{d}\boldsymbol{\omega} \tag{9}$$

However, the posterior distribution $p(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{O})$ in Eq. (9) cannot be got in a closed form, because it is difficult to analytically marginalize the likelihood over $\boldsymbol{\omega}$. Accordingly, variational inference is used to approximate the posterior distribution $p(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{O})$ in this paper. First of all, we define an approximating variational distribution $q(\boldsymbol{\omega})$ to factorize over the weights and biases [24], which is written as

$$\begin{aligned} q(\boldsymbol{\omega}) &= q(\boldsymbol{\omega}_{\mathrm{RCL}})q(\boldsymbol{\omega}_{\mathrm{FCL}}) \\ &= \prod_{i=1}^{N} q(\boldsymbol{K}_r^i)q(\boldsymbol{W}_r^i)q(\boldsymbol{b}_r^i)q(\boldsymbol{K}_u^i)q(\boldsymbol{W}_u^i)q(\boldsymbol{b}_u^i)q(\boldsymbol{K}_h^i) \\ &\quad \times q(\boldsymbol{W}_h^i)q(\boldsymbol{b}_h^i)\prod_{i=1}^{L} q(\boldsymbol{W}_f^i)q(\boldsymbol{b}_f^i) \end{aligned} \tag{10}$$

In Eq. (10), the variational distribution of each weight is defined to be a Gaussian mixture distribution with two components, and each bias follows a simple Gaussian distribution. For simplicity of notation, $\boldsymbol{W}^l$ and $\boldsymbol{b}^l$ are respectively used to represent the weight and bias in the weight layer l, where $l = 1, 2, \ldots, N+L$. Correspondingly, $q(\boldsymbol{W}^l)$ and $q(\boldsymbol{b}^l)$ can be expressed as

$$q(\boldsymbol{W}^l) = \pi^l \mathcal{N}(\boldsymbol{\mu}_W^l, \tau^{-1}\boldsymbol{I}) + (1-\pi^l)\mathcal{N}(0, \tau^{-1}\boldsymbol{I}) \tag{11}$$

$$q(\boldsymbol{b}^l) = \mathcal{N}(\boldsymbol{\mu}_b^l, \tau^{-1}\boldsymbol{I}) \tag{12}$$

where $\pi^l \in [0, 1]$ is the probability that is given in advance, $\boldsymbol{\mu}_W^l$ and $\boldsymbol{\mu}_b^l$ are the variational parameters of weight and bias, respectively, and τ is the model precision.

Then, we minimize the Kullback–Leibler (KL) divergence between the approximating variational distribution and the posterior distribution: $\mathrm{KL}(q(\boldsymbol{\omega})\|p(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{O}))$, resulting in the approximate predictive distribution

$$p(o^*|\boldsymbol{x}^*, \boldsymbol{X}, \boldsymbol{O}) \approx \int p(o^*|\boldsymbol{x}^*, \boldsymbol{\omega})q^*(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \tag{13}$$

where $q^*(\boldsymbol{\omega})$ is the value that is able to minimize the KL divergence. As a result, the optimization objective of RCNN is to minimize the KL divergence, which is equivalent to maximizing the evidence lower bound [25]. Therefore, the objective function of RCNN can be written as

$$\begin{aligned} \mathcal{L} &= \mathrm{KL}(q(\boldsymbol{\omega})\|p(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{O})) \\ &= \int q(\boldsymbol{\omega}) \log \frac{q(\boldsymbol{\omega})}{p(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{O})}\mathrm{d}\boldsymbol{\omega} \\ &\propto -\int q(\boldsymbol{\omega}) \log p(\boldsymbol{O}|\boldsymbol{X}, \boldsymbol{\omega})\ \mathrm{d}\boldsymbol{\omega} + \mathrm{KL}(q(\boldsymbol{\omega})\|p(\boldsymbol{\omega})) \\ &= -\sum_{t=1}^{T}\int q(\boldsymbol{\omega}) \log p(o_t|x_t, \boldsymbol{\omega})\ \mathrm{d}\boldsymbol{\omega} + \mathrm{KL}(q(\boldsymbol{\omega})\|p(\boldsymbol{\omega})) \end{aligned} \tag{14}$$

Table 1
Operating conditions of FEMTO-ST bearing datasets.

Operating condition	Radial force	Rotating speed	Bearing dataset	
			Training dataset	Testing dataset
Condition 1	4000 N	1800 rpm	Ber 1_1 Ber 1_2 Ber 1_3 Ber 1_4 Ber 1_5	Ber 1_6 Ber 1_7
Condition 2	4200 N	1650 rpm	Ber 2_1 Ber 2_2 Ber 2_3 Ber 2_4 Ber 2_5	Ber 2_6 Ber 2_7
Condition 3	5000 N	1500 rpm	Ber 3_1 Ber 3_2 Ber 3_3	–

In Eq. (14), the first term can be evaluated by Monte Carlo integration, and the detailed procedure is described as follows. Firstly, we re-parameterize the integrands in each sum term using the standard normal distribution $q(\boldsymbol{\alpha}) = \mathcal{N}(0, \boldsymbol{I})$ and the Bernoulli distribution $q(\boldsymbol{\beta}) = \text{Bernoulli}(\pi)$. According to Eqs. (11) and (12), $\boldsymbol{W}^l$ and $\boldsymbol{b}^l$ can be respectively re-written as

$$\boldsymbol{W}^l = \boldsymbol{\beta}^l\left(\boldsymbol{\mu}_W^l + \tau^{-\frac{1}{2}}\boldsymbol{\alpha}\right) + \left(1-\boldsymbol{\beta}^l\right)\tau^{-\frac{1}{2}}\boldsymbol{\alpha} \tag{15}$$

and

$$\boldsymbol{b}^l = \boldsymbol{\mu}_b^l + \tau^{-\frac{1}{2}}\boldsymbol{\alpha} \tag{16}$$

Then, based on the above two stochastic estimators, i.e., Eqs. (15) and (16), the objective function can be reformulated as

$$\mathcal{L} = -\sum_{t=1}^{T}\int q\left(\boldsymbol{\alpha},\boldsymbol{\beta}\right)\log p\left(o_t\left|\boldsymbol{x}_t,\boldsymbol{\omega}\left(\boldsymbol{\alpha},\boldsymbol{\beta}\right)\right.\right)\mathrm{d}\boldsymbol{\alpha}\mathrm{d}\boldsymbol{\beta} + \mathrm{KL}(q(\boldsymbol{\omega})\|p(\boldsymbol{\omega}))\tag{17}$$

where each integrand in the first term does depend on the weights and biases. After that, Monte Carlo integration with a single sample $\hat{\boldsymbol{\omega}}_t \sim q(\boldsymbol{\omega})$ is used to estimate each integral in the first term of Eq. (17), resulting in an unbiased estimator $\log p(o_t|\boldsymbol{x}_t, \hat{\boldsymbol{\omega}}_t)$. Finally, the objective function can be obtained by

$$\mathcal{L} = -\sum_{t=1}^{T}\log p\left(o_t\left|\boldsymbol{x}_t,\hat{\boldsymbol{\omega}}_t\right.\right) + \mathrm{KL}(q(\boldsymbol{\omega})\|p(\boldsymbol{\omega}))\tag{18}$$

Further, the second term in Eq. (14) can be approximated as L^2 regularization based on [26], i.e.,

$$\mathrm{KL}(q(\boldsymbol{\omega})\|p(\boldsymbol{\omega})) = \sum_{l=1}^{M+L}\left(\frac{\pi^l c^2}{2}\left\|\boldsymbol{\mu}_W^l\right\|_2^2 + \frac{c^2}{2}\left\|\boldsymbol{\mu}_b^l\right\|_2^2\right)\tag{19}$$

where c is the prior length-scale. Hence, the objective function in Eq. (18) can be written as

$$\begin{aligned}\mathscr{L} &= -\sum_{t=1}^{T}\log p(o_t|\boldsymbol{x}_t,\hat{\boldsymbol{\omega}}_t) + \sum_{l=1}^{M+L}\left(\frac{\pi^l c^2}{2}\parallel\boldsymbol{\mu}_W^l\parallel_2^2 + \frac{c^2}{2}\parallel\boldsymbol{\mu}_b^l\parallel_2^2\right)\\ &\propto \frac{1}{T}\sum_{t=1}^{T}\frac{-\log p(o_t|\boldsymbol{x}_t,\hat{\boldsymbol{\omega}}_t)}{\tau}\\ &\quad + \sum_{l=1}^{M+L}\left(\frac{\pi^l c^2}{2\tau T}\parallel\boldsymbol{\mu}_W^l\parallel_2^2 + \frac{c^2}{2\tau T}\parallel\boldsymbol{\mu}_b^l\parallel_2^2\right)\\ &= \frac{1}{T}\sum_{t=1}^{T}E\left(o_t,\hat{o}(\boldsymbol{x}_t,\hat{\boldsymbol{\omega}}_t)\right) + \sum_{l=1}^{M+L}\left(\frac{\pi^l c^2}{2\tau T}\parallel\boldsymbol{\mu}_W^l\parallel_2^2 + \frac{c^2}{2\tau T}\parallel\boldsymbol{\mu}_b^l\parallel_2^2\right)\end{aligned}\tag{20}$$

where $E(\cdot,\cdot)$ is the loss function, such as mean square error, mean absolute error and Huber loss.

Through analyzing Eqs. (15), (16) and (20), it can be seen that for the probabilistic RCNN, the estimation process of the network output $\hat{o}(\cdot)$ with a sample $\hat{\boldsymbol{\omega}}$ is to randomly mask rows in each weight matrix during the forward pass, which is identical to performing dropout [27] in each weight layer of RCNN. In addition, the second term in Eq. (20) is equivalent to adding a L^2 regularization term to each weight and bias during model optimization. Consequently, the uncertainty of RCNN can be quantified by applying dropout with probability π and L^2 regularization with weight decay coefficient λ into each recurrent convolutional layer and each fully-connected layer. Correspondingly, the objective function in Eq. (20) is proportional to the following objective function:

$$\mathcal{L} \propto \mathcal{L}_{\text{dropout}} = \frac{1}{T}\sum_{t=1}^{T}E\left(o_t,\hat{o}_t\right) + \lambda\sum_{l=1}^{M+L}\left(\left\|\boldsymbol{W}^l\right\|_2^2 + \left\|\boldsymbol{b}^l\right\|_2^2\right)\tag{21}$$

The above objective function $\mathcal{L}_{\text{dropout}}$ can be minimized using some optimizers, such as stochastic gradient descent, RMSprop and Adam. Then, predictions can be achieved by Monte Carlo estimation [26], which is also referred to as Monte Carlo dropout. To be specific, for a new input $\boldsymbol{x}^*$, the Monte Carlo dropout performs V stochastic forward passes through the network to obtain the predictive mean and variance, which is formulated as

$$p(o^*|\boldsymbol{x}^*,\boldsymbol{X},\boldsymbol{O}) \approx \int p(o^*|\boldsymbol{x}^*,\boldsymbol{\omega})q(\boldsymbol{\omega})\mathrm{d}\boldsymbol{\omega} \approx \frac{1}{V}\sum_{v=1}^{V}p\left(o^*\left|\boldsymbol{x}^*,\hat{\boldsymbol{\omega}}_v\right.\right)\tag{22}$$

with $\hat{\boldsymbol{\omega}}_v \sim q(\boldsymbol{\omega})$.

3. Case study I: RUL prediction of rolling element bearings

In this section, we use vibration signals collected from accelerated degradation tests of rolling element bearings to validate the effectiveness of the proposed RCNN. Moreover, to illustrate the superiority of RCNN in RUL prediction, the prediction results are also compared with those of four state-of-the-art prognostics approaches.

3.1. FEMTO-ST bearing datasets

The vibration data used in this paper are the public FEMTO-ST bearing datasets, which are provided by the Franche-Comté Electronics Mechanics Thermal Science and Optics-Science and Technologies Institute and are available on the website of NASA Prognostics Data Repository [28]. The FEMTO-ST bearing datasets are acquired by conducting accelerated degradation tests of rolling element bearings on the PRONOSTIA experimental platform. As tabulated in Table 1, a total of seventeen bearings are tested under three different operating conditions. To monitor the failure progression of bearings and acquire their run-to-failure data, two accelerometers are positioned at 90° on the outer race of the tested bearings. The sampling frequency is 25.6 kHz, and 2560 data points are recorded every 10 s. Fig. 3 shows the horizontal and vertical vibration signals of Ber 1_1 during the whole operating life. In this paper, we use the last two bearing datasets of Condition 1 and Condition 2 as the testing datasets, and the remaining ones are used as the training datasets. During training, both horizontal and vertical vibration signals are used as the inputs of RCNN, and correspondingly the size of an input volume is $2560 \times 1 \times 2$.

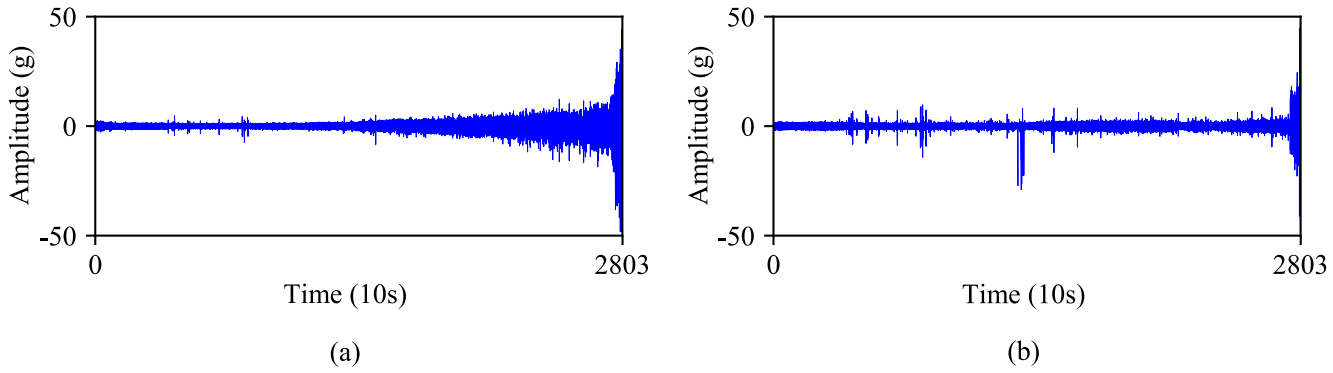


Fig. 3. Vibration signals of Ber 1_1 during the whole operating life. (a) Horizontal direction. (b) Vertical direction.

3.2. Prognostics metrics

To quantitatively evaluate the prediction performance of RCNN, two commonly used prognostics metrics, i.e., cumulative relative accuracy (CRA) and convergence (C_{PE}) [29], are employed in this paper. CRA is able to evaluate the comprehensive prediction accuracy of a prognostics algorithm at multiple time instances. Given RUL prediction results, CRA value can be computed by

$$CRA = \frac{1}{S}\sum_{s=1}^{S}\left(1 - \frac{|ActRUL_s - PreRUL_s|}{ActRUL_s}\right) \tag{23}$$

where S is the number of testing samples, and $ActRUL_s$ and $PreRUL_s$ are the actual RUL and the predictive RUL values corresponding to the s-th testing sample, respectively. The closer the CRA value is to 1, the more accurate the RUL prediction results are.

C_{PE} is a measure of convergence rate, which is defined as the Euclidean distance between the origin and the centroid of the area under the prediction error curve [29]. Mathematically, C_{PE} is able to be calculated by

$$C_{PE} = \sqrt{(C_x - t_p)^2 + C_y^2} \tag{24}$$

where t_p is the first prediction time instance, and (C_x, C_y) is the centroid of the area under the prediction error curve. The lower the C_{PE} value is, the faster the predictive RUL converges the actual RUL. Particularly, since predictions at time instances that are close to the end of lifetime are more meaningful than those at the early stage for maintenance decision making and spare parts management [30], we calculate CRA and C_{PE} values from the half to the end of lifetime for each bearing in the testing datasets.

3.3. Configuration of RCNN

In RUL prediction of bearings, we perform 5-fold cross-validation on the training datasets to determine the structural parameters of RCNN, including the number of convolution kernels M, the kernel size $k \times 1$, the number of recurrent convolutional layers N, the pooling size p, and the number of neurons F. Meanwhile, we apply dropout and L^2 regularization to each recurrent convolutional layer and each fully-connected layer, and perform V stochastic forward passes to obtain the predictive mean and variance. Furthermore, we use mean square error as the loss function of RCNN, and employ Adam optimizer [31] with a mini-batch size of 128 to optimize the objective function $\mathcal{L}_{dropout}$ in Eq. (21) by iteratively updating the network weights and biases. The RCNN model is trained for 200 epochs from scratch, and its detailed configuration is summarized in Table 2.

Table 2
Configuration of RCNN in RUL prediction of bearings.

Hyperparameter	Size	Hyperparameter	Size
Number of kernels M	16	Kernel size $k \times 1$	8×1
Number of layers N	4	Pooling size p	8
Number of neurons F	100	Weight decay coefficient λ	10^{-5}
Number of forward passes V	1000	Dropout probability π	0.15
Mini-batch size	128	Number of epochs	200

3.4. RUL prediction for bearings

The prediction performance of RCNN is investigated and discussed by performing the RUL estimation tasks of bearings in this section. Firstly, we analyze the effect of network depth and dropout probability on the prediction accuracy and uncertainty of RCNN, respectively. Then, the benefits from recurrent convolutional layers are demonstrated and discussed. Finally, the proposed RCNN is compared with four state-of-the-art prognostics approaches so as to illustrate its superiority.

3.4.1. Effect of network depth on prediction performance

For deep prognostics models, network depth plays an important role in boosting representation learning ability. Thus, we first investigate the effect of network depth on the prediction performance of RCNN. In the proposed RCNN, the network depth can be increased by periodically stacking more recurrent convolutional layers and pooling layers. As listed in Table 3, we employ RCNN with five different network depths to perform RUL prediction of bearings, and evaluate the prediction results of each bearing based on CRA and C_{PE}. From Table 3, it can be observed that with the increment of network depth, the CRA values are higher and the C_{PE} values are lower. This implies that by increasing the network depth, RCNN has more powerful representation learning ability and so achieves higher accuracy and faster convergence in RUL prediction of bearings. However, it should be noted that the deeper the network depth is, the larger the model size is, which will lead to a heavier computational burden. Meanwhile, since the available training samples are limited, large network depth may result in accuracy saturation or even over-fitting problem. As a result, the number of recurrent convolutional layers N is set to be 4 in this paper.

3.4.2. Effect of dropout probability on uncertainty quantification

Based on the results of variational inference in Section 2.2, the uncertainty of RCNN can be quantified by Monte Carlo dropout. During cross-validation, we note that the results of uncertainty quantification of RCNN are largely affected by dropout probability. To analyze this influence, we set the value range of π to be [0.05,

Table 3
The effect of the network depth on the prediction performance of RCNN.

Number of layers N	Total model parameters	CRA				C_{PE}			
		Ber 1_6	Ber 1_7	Ber 2_6	Ber 2_7	Ber 1_6	Ber 1_7	Ber 2_6	Ber 2_7
2	57,421	0.5623	0.4102	0.4696	0.6829	6307.1472	5886.0009	1796.8519	588.1422
3	208,269	0.6981	0.6372	0.7395	0.7967	6214.6742	5681.2825	1790.0149	573.4991
4	804,877	0.7781	0.7419	0.8116	0.8453	6164.8550	5619.0031	1758.4386	564.5396
5	3,177,741	0.7717	0.7533	0.8250	0.8637	6187.1288	5613.3652	1760.6395	564.9190
6	12,642,061	0.7320	0.7795	0.8320	0.8439	6180.8083	5617.5423	1765.5693	568.7934

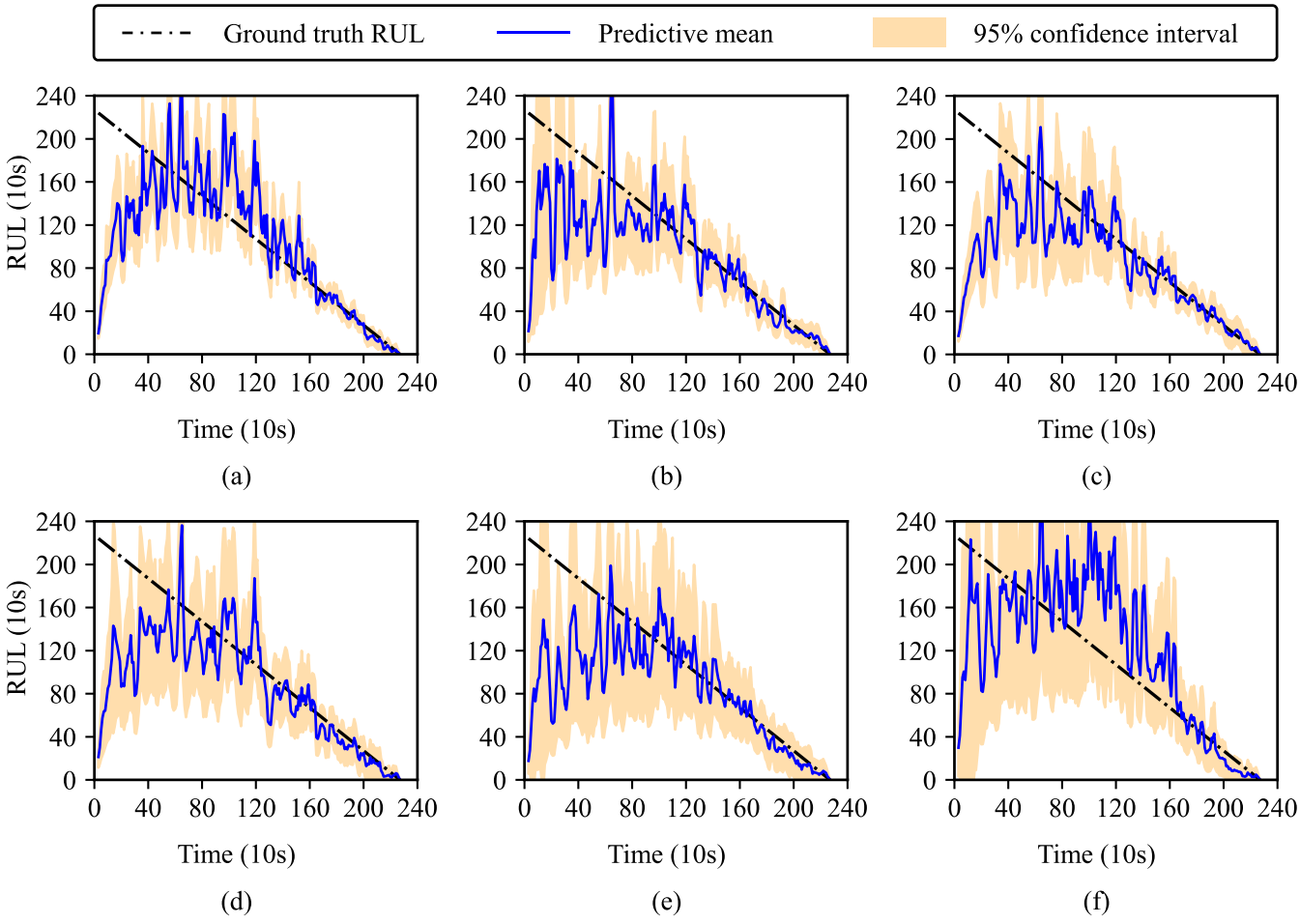


Fig. 4. RUL prediction results of Ber 2_7 using different dropout probability. (a) $\pi = 0.05$. (b) $\pi = 0.1$. (c) $\pi = 0.15$. (d) $\pi = 0.2$. (e) $\pi = 0.25$. (f) $\pi = 0.3$.

0.3] with an interval of 0.05, and the corresponding RUL prediction results of Ber 2_7 are shown in Fig. 4. In particular, Fig. 5 also plots the RUL distributions at time instance of 1700s (three quarters of lifetime) and gives the corresponding mean and standard deviation (SD). In prognostics, the sources of uncertainty include modeling errors, measurement noise, future loading, operating, environmental and usage conditions and so on [32]. Since bearings are tested under constant operating conditions, we mainly focus on the uncertainty from modeling errors and measurement noise in this case study. Specifically, we hope to select an appropriate π value to reduce the uncertainty from modeling errors while capturing the uncertainty from measurement noise. From Figs. 4 and 5, it can be clearly seen that RCNN with a larger dropout probability results in a wider uncertainty interval. This is because that the larger dropout probability will mask more units in the weight layers during the forward pass, thereby increasing the modeling uncertainty and leading to a larger SD of RUL (see Fig. 5). The high uncertainty is detrimental to maintenance decision making and thus it is desired to select a small π value. However, it should be noted that if the dropout probability is too small, the uncertainty from measurement noise will not be captured effectively. Meanwhile, a small π value is also unfavourable to prevent the over-fitting problem and improve the generalization of the proposed RCNN. According to the above analyses, the dropout probability π is set to be 0.15 in RUL prediction of bearings.

3.4.3. Benefits from recurrent convolutional layers

Different from a standard CNN, the proposed RCNN constructs a new core building block, i.e., recurrent convolutional layer, to learn short-term or long-term dependencies from time-series sensor data. To illustrate the benefits from recurrent convolutional layers, we apply the other two convolutional networks into RUL prediction of bearings, including a standard CNN (denoted as S-CNN) and a CNN with recurrent connections (denoted as R-CNN). The architectures of these two prognostics networks are similar to that of RCNN. The difference is that S-CNN uses standard convolutional

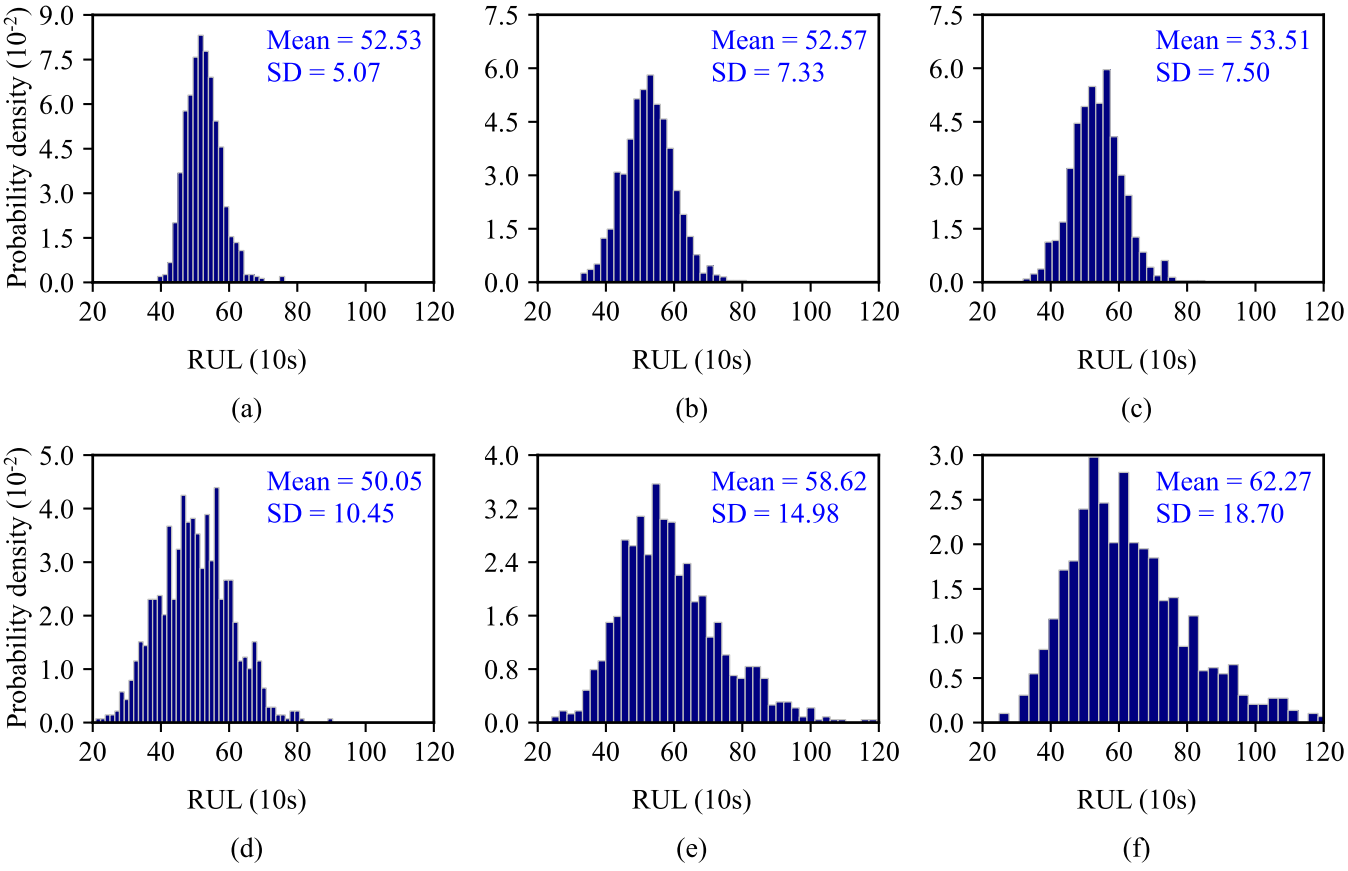
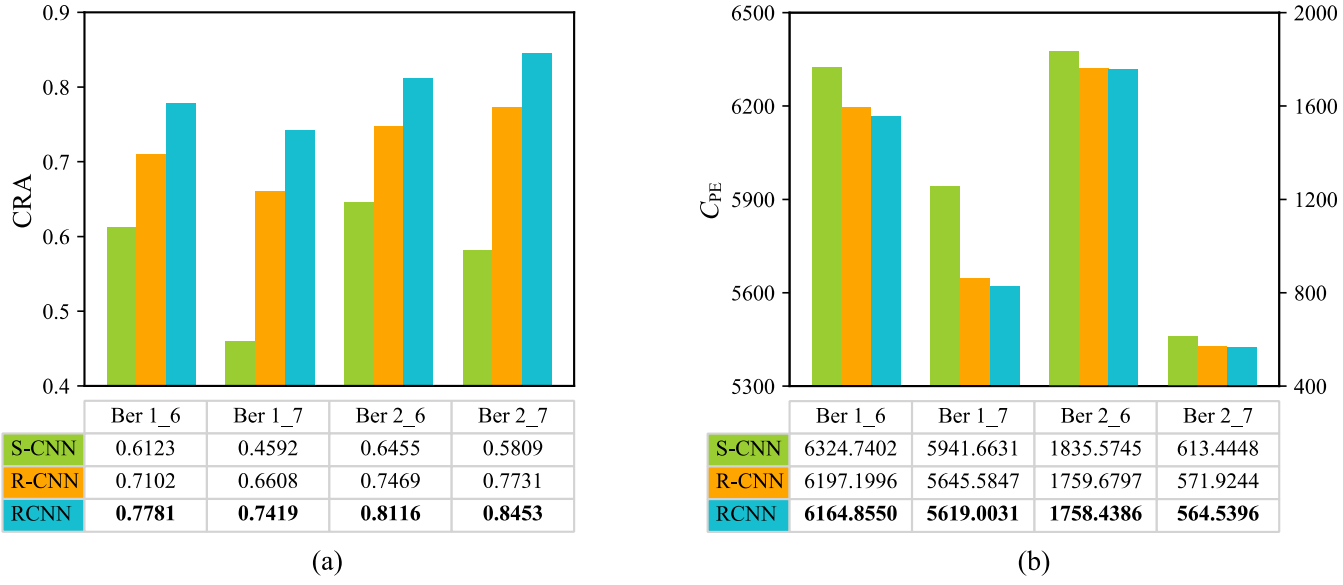


Fig. 5. Predictions for Ber 2_7 at time instance of 1700s using different dropout probability. (a) $\pi = 0.05$. (b) $\pi = 0.1$. (c) $\pi = 0.15$. (d) $\pi = 0.2$. (e) $\pi = 0.25$. (f) $\pi = 0.3$.



	Ber 1_6	Ber 1_7	Ber 2_6	Ber 2_7
S-CNN	0.6123	0.4592	0.6455	0.5809
R-CNN	0.7102	0.6608	0.7469	0.7731
RCNN	**0.7781**	**0.7419**	**0.8116**	**0.8453**

	Ber 1_6	Ber 1_7	Ber 2_6	Ber 2_7
S-CNN	6324.7402	5941.6631	1835.5745	613.4448
R-CNN	6197.1996	5645.5847	1759.6797	571.9244
RCNN	**6164.8550**	**5619.0031**	**1758.4386**	**564.5396**

Fig. 6. Performance estimation results of three different prognostics networks. (a) CRA values. (b) C_{PE} values.

layers to replace recurrent convolutional layers while R-CNN employs recurrent convolutional layers without a gating mechanism. Moreover, S-CNN and R-CNN have the same hyperparameter settings as the proposed RCNN, and their performance estimation results are shown in Fig. 6. It can be observed that by adding recurrent connections to each convolutional layer, R-CNN obtains a higher CRA value and a lower C_{PE} value than S-CNN for each bearing. Further, the introduction of the gating mechanism described in Section 2.1.1 makes the proposed RCNN achieve higher accuracy and faster convergence compared with R-CNN. The above analysis and comparison results indicate that through building recurrent convolutional layers to model the temporal dependencies of different degradation states, the proposed RCNN effectively improves the prediction performance of convolutional networks.

Table 4
Performance comparison of five different prognostics approaches on FEMTO-ST bearing datasets.

Bearing dataset		M1 [12]	M2 [14]	M3 [11]	M4 [27]	Proposed RCNN
Ber 1_6	CRA	0.5550	0.7227	0.6123	0.7410	**0.7781**
	C_{PE}	6313.0213	6214.1844	6324.7402	6167.4558	**6164.8550**
Ber 1_7	CRA	0.4789	0.7231	0.4592	0.6721	**0.7419**
	C_{PE}	5910.2072	5647.1975	5941.6631	5654.8450	**5619.0031**
Ber 2_6	CRA	0.6731	0.7590	0.6455	0.7206	**0.8116**
	C_{PE}	1758.2144	1759.7486	1835.5745	1772.8769	**1758.4386**
Ber 2_7	CRA	0.4305	0.7889	0.5809	0.8183	**0.8453**
	C_{PE}	666.6638	583.9817	613.4448	571.1441	**564.5396**

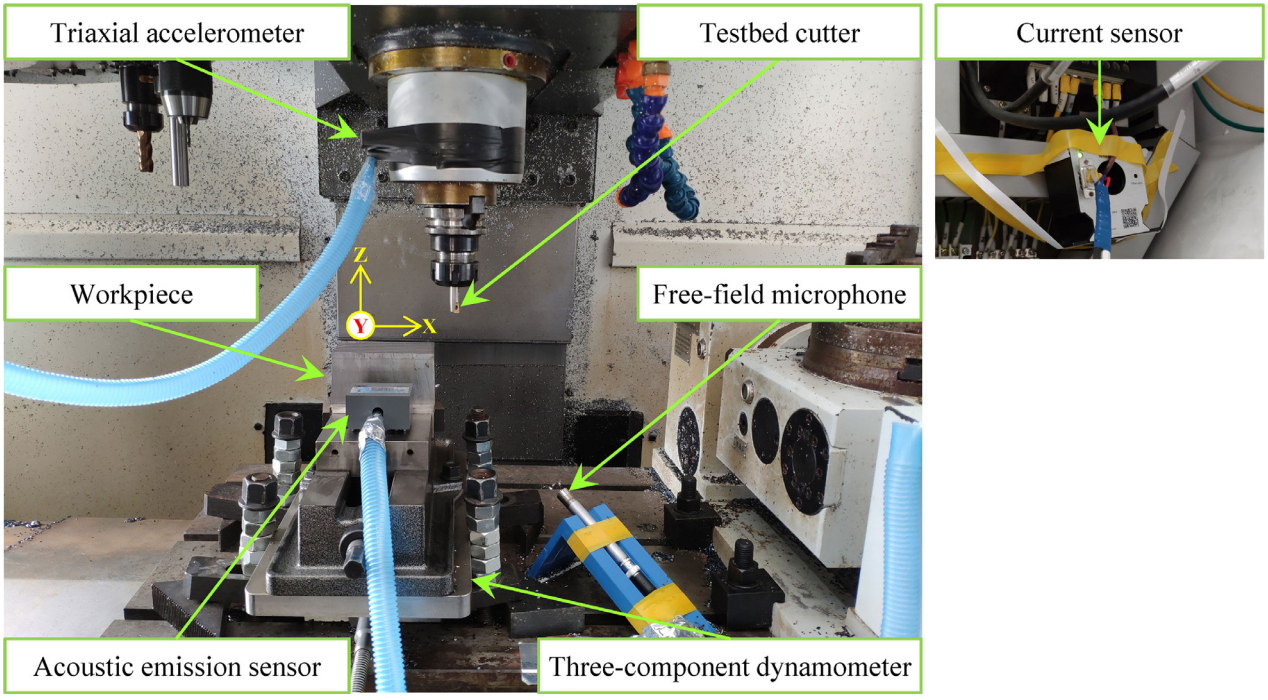


Fig. 7. CNC milling machine and sensor placement.

3.4.4. Comparison with state-of-the-art prognostics approaches

To demonstrate the superiority of the proposed RCNN, four state-of-the-art prognostics approaches presented in [17,18,20] and [33] are implemented to predict the RUL of four tested bearings, and they are denoted as M1, M2, M3 and M4, respectively. All of the first three prognostics approaches are developed based on standard CNNs, and the last one is a combination of CNN and long short-term memory (LSTM). To be specific, M1 first uses fast Fourier transform to extract 64 frequency-domain features and then constructs a CNN containing 3 convolution layers, 3 average pooling layers and 7 fully-connected layers to predict RUL of bearings. M2 employs wavelet transform to obtain time-frequency features and predicts the RUL using an 8-layer CNN. M3 builds a CNN by a stack of 2 convolutional layers, 2 average pooling layers and 1 fully-connected layer and utilizes raw vibration signals as its inputs for training and testing. M4 first uses a convolutional layer and a global average pooling layer to extract local features from vibration signals and then employs an LSTM layer and a fully-connected layer to predict the RUL. Table 4 summarizes the performance evaluation results of these five different prognostics approaches on FEMTO-ST bearing datasets. It can be seen from Table 4 that the proposed RCNN gets the highest CRA values and the lowest C_{PE} values among all of these five prognostics approaches, which signifies that RCNN provides the most accurate RUL estimation results and achieves the fastest convergence. More importantly, instead of a point estimation, RCNN is able to offer a probabilistic distribution by quantifying uncertainty in prognostics, which overcomes the inherent limitation of CNN and is beneficial to maintenance decision making. Consequently, the proposed RCNN is superior to the other four prognostics approaches in RUL prediction of rolling element bearings.

4. Case study II: RUL prediction of milling cutters

To further demonstrate the effectiveness and superiority of the proposed approach in RUL prediction of machinery, RCNN is employed to predict RUL of milling cutters in this section, and is also compared with five published prognostics approaches focusing on RUL estimation of cutting tools.

4.1. Experiment introduction

As shown in Fig. 7, life testing of milling cutters is conducted in a computer numerical control (CNC) milling machine (DAHENG VMC850). The workpiece has size $100 \times 60 \times 60\,mm^3$ (length × width × height) and its material is C45 grade steel. During machining, the table feeds workpiece from left to right along X-axis direction, which is referred to as a pass. In particular, for each new pass, the tested cutter is returned to the starting point, ensuring that all passes are made in the same feed direction. In

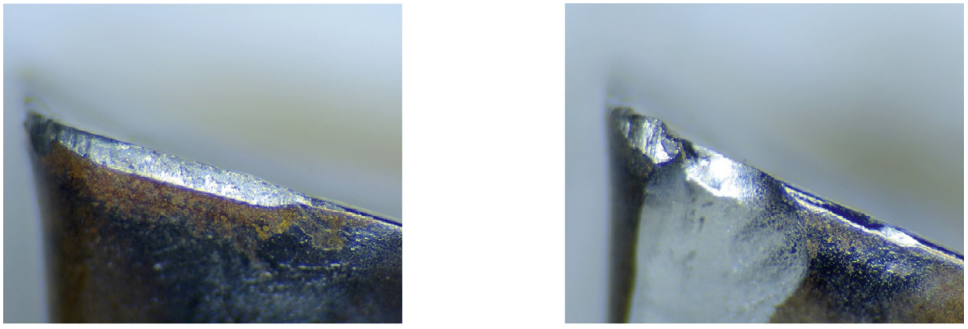


Fig. 8. Cutter deterioration photographs. (a) Flank wear. (b) Brittle fracture.

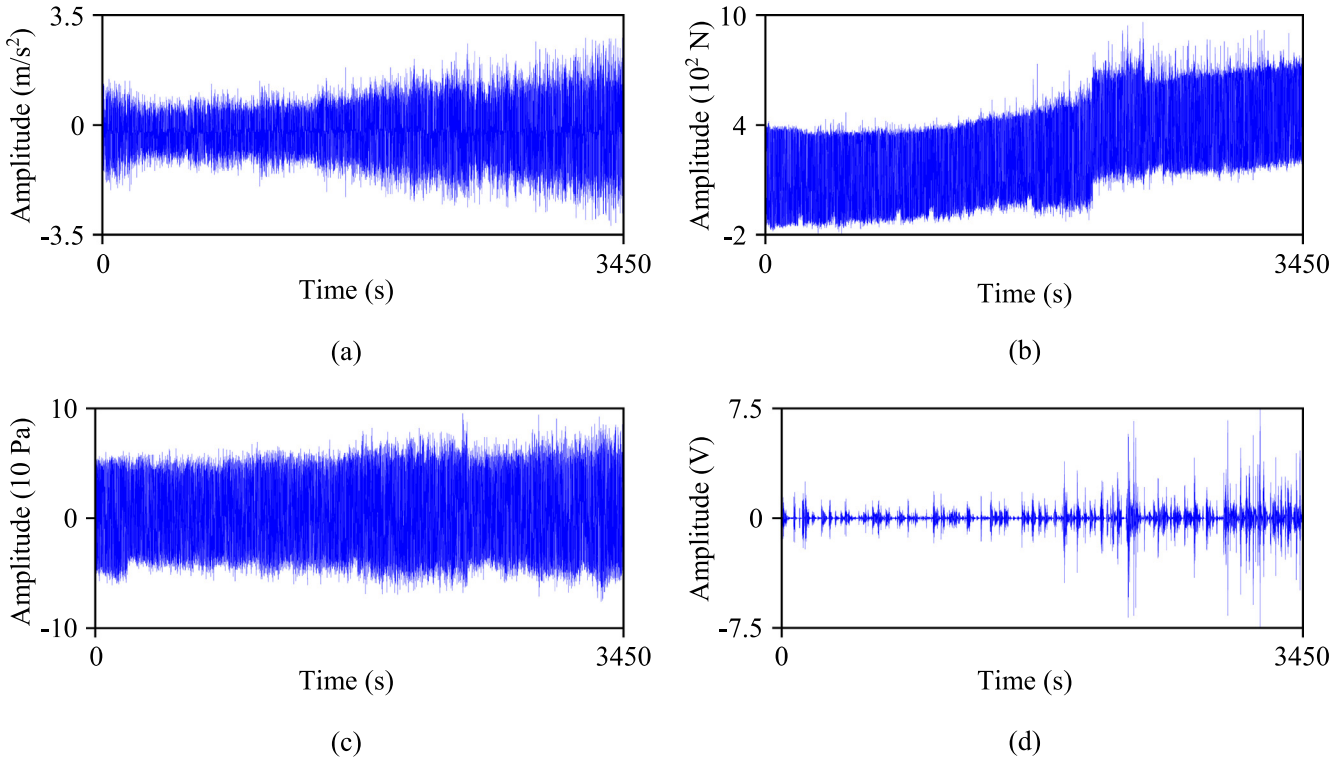


Fig. 9. Monitoring signals of Cutter 1 during the whole operating life. (a) Vibration signals in Z-axis direction. (b) Force signals in Z-axis direction. (c) Sound signals. (d) Acoustic emission signals.

our experiments, six milling cutters whose type is APMT 1135 are tested under dry milling condition, i.e., without the application of a cutting fluid. The machining parameters are set as follows: the spindle speed is 2500 rpm; the feed rate is 200 mm/min; and the depth of cut in Z-axis direction is 2 mm. To monitor failure progressions of milling cutters, as shown in Fig. 7, five types of sensors are used in our experiments, including triaxial accelerometer (PCB 356A15), three-component dynamometer (Kistler 9257B), free-field microphone (Brüel & Kjær 4966), current sensor (Anyway CSA201-P030T01) and acoustic emission sensor (PAC WD). For the first four sensors, the sampling frequency is set to be 10 kHz and 1×10^4 data points are recorded every pass. For the last sensor, i.e., acoustic emission sensor, the sampling frequency is 1 MHz and 1×10^6 data points are recorded every pass.

Fig. 8 shows cutter deterioration photographs after reaching the end of life. It can be seen that the deterioration of milling cutters contains two major classes: wear and brittle fracture. For wear, we define the width of the flank wear band as cutter-life criteria based on international standard ISO 8688–1, and use a GP-300C industrial microscope to measure it after each pass. When the width of the flank wear band is greater than 0.3 mm, the tested cutter attains the deterioration limit and the experiment is stopped. Fig. 9 plots the monitoring signals of Cutter 1 during the whole operating life. In this case study, Cutter 1, Cutter 2, Cutter 3 and Cutter 4 are used as training datasets, and Cutter 5 and Cutter 6 are testing datasets. Moreover, all types of monitoring signals are used as the inputs of RCNN, and the size of an input volume is $10000 \times 1 \times 9$.

4.2. RUL prediction results, comparison and discussion

In RUL prediction of milling cutters, we use the same hyper-parameter settings listed in Table 2. Meanwhile, CRA and C_{PE} are also employed to evaluate the prediction performance quantitatively. Fig. 10 shows the RUL prediction results of Cutter 5 and Cutter 6. From Fig. 10, it can be observed that at the initial prediction stage, there is a large deviation between the estimated RUL and the actual RUL, but as time goes on, the deviation reduces gradually and the predictive RUL converges to the actual RUL faster.

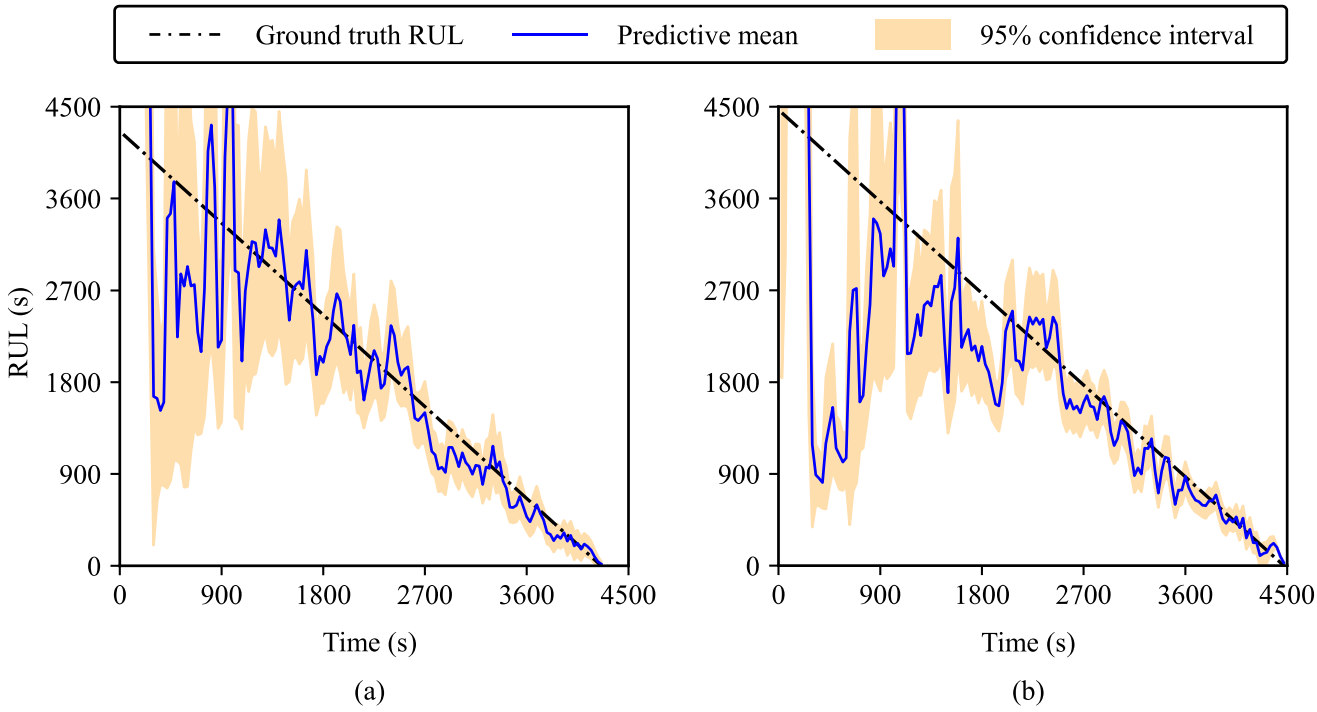


Fig. 10. RUL prediction results of milling cutters. (a) Cutter 5. (b) Cutter 6.

Table 5
Performance comparison of six different prognostics approaches on milling cutter datasets.

Milling cutter dataset		SVM [28]	FNN [29]	DBN [30]	CNN [31]	CBLSTM [32]	Proposed RCNN
Cutter 5	CRA	0.6320	0.6104	0.6622	0.7060	0.8171	**0.8578**
	C_{PE}	1152.3089	1156.0747	1122.2809	1109.5670	1067.3637	**1053.1341**
Cutter 6	CRA	0.4736	0.6219	0.7687	0.7483	0.7986	**0.8601**
	C_{PE}	1313.8384	1201.2950	1133.8981	1150.9644	1136.0671	**1112.9371**

This is because at the beginning stage, cutters are in a break-in period and only have slight wear. Correspondingly, it is difficult for RCNN to capture the degradation characteristics and build the accurate mapping between the monitoring signals and the RUL at this stage. Whereas with the increase of cutter wear, more degradation information can be captured by monitoring signals. Accordingly, the proposed RCNN obtains more and more accurate RUL prediction results. In addition, it can be also seen from Fig. 10 that because of the lack of degradation information, the modeling errors are large at the beginning. Meanwhile, monitoring signals also contain considerable measurement noise at this stage, leading to a large fluctuation of the predictive mean and a wide 95% confidence interval. As more degradation information is available, the uncertainty from modeling errors reduces gradually and the proposed RCNN achieves a narrow uncertainty interval. This indicates that the proposed RCNN is able to handle the uncertainty in prognostics fairly well. Further, we also employ five existing prognostics approaches focusing on RUL prediction of cutting tools in this case study for comparison, including SVM [34], fuzzy neural network (FNN) [35], deep belief network (DBN) [36], CNN [37] and convolutional bi-directional LSTM (CBLSTM) [38]. For the first three approaches, i.e., SVM, FNN and DBN, we first extract and select features from the acquired monitoring signals following [34, 35] and [36]. Then, these features are input into the corresponding models for training and testing. Whereas for CNN and CLSTM, raw monitoring signals are used as the inputs of models for training and testing. Especially, for each prediction model, we perform 4-fold cross-validation to tune the hyperparameters, aiming to obtain better RUL prediction results. Table 5 reports the performance evaluation results of these six different prognostics approaches on milling cutter datasets. It can be clearly seen that compared with the other five existing approaches, the proposed RCNN has higher CRA value and lower C_{PE} value for each milling cutter, which means the proposed RCNN outperforms the other prognostics approaches in RUL prediction of milling cutters. This performance improvement validates again the benefits from the constructed recurrent convolutional layers.

5. Conclusions

In this paper, we propose a new prognostics framework named RCNN for RUL prediction of machinery. The proposed RCNN uses time-series data acquired by different sensors as its inputs. Then, a new core building block, called recurrent convolutional layer, is constructed to model the temporal dependencies of different degradation states, and a max pooling layer is used to reduce the dimensionality of representations and make the extracted features compact. Through periodically stacking multiple recurrent convolutional layers and max pooling layers, the high-level representations can be automatically learned from the input sensor data. Finally, the RUL is estimated by feeding these learned representations into the subsequent fully-connected layers. Further, we employ variational inference to quantify the uncertainty of RCNN in prognostics, and a probabilistic RUL prediction result is obtained by Monte Carlo dropout. We evaluate the prediction performance of RCNN using vibration data collected from accelerated degradation tests of rolling element bearings and multi-sensor data acquired by life testing of milling cutters, and also compare the proposed RCNN

with some state-of-the-art prognostics approaches. Experimental results show that the proposed RCNN has a clear advantage in accuracy and convergence compared with some existing CNN-based prognostics approaches and other typical machine learning models. More importantly, RCNN is able to provide probabilistic RUL prediction results, which breaks the inherent limitation of CNN and facilitates maintenance decision making.

Declaration of Competing Interest

None

Acknowledgments

This research was supported by National Key R&D Program of China (2018YFB1306100), NSFC-Zhejiang Joint Fund for the Integration of Industrialization and Informatization (U1709208) and National Natural Science Foundation of China (61673311).

References

[1] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, J. Lin, Machinery health prognostics: a systematic review from data acquisition to rul prediction, Mech. Syst. Signal Process. 104 (2018) 799–834.
[2] M. Rigamonti, P. Baraldi, E. Zio, I. Roychoudhury, K. Goebel, S. Poll, Ensemble of optimized echo state networks for remaining useful life prediction, Neurocomputing 281 (2018) 121–138.
[3] H. Dong, N. Hou, Z. Wang, H. Liu, Finite-horizon fault estimation under imperfect measurements and stochastic communication protocol: dealing with finite-time boundedness, Int. J. Robust Nonlinear Control 29 (2019) 117–134.
[4] X. Li, H. Dong, Z. Wang, F. Han, Set-membership filtering for state-saturated systems with mixed time-delays under weighted try-once-discard protocol, IEEE Trans. Circuits Syst. II: Express Briefs 66 (2019) 312–316.
[5] B. Wang, Y. Lei, N. Li, N. Li, A hybrid prognostics approach for estimating remaining useful life of rolling element bearings, IEEE Trans. Reliab. (2018) 1–12 In press, doi:10.1109/TR.2018.2882682.
[6] J. Li, H. Dong, Z. Wang, N. Hou, F.E. Alsaadi, On passivity and robust passivity for discrete-time stochastic neural networks with randomly occurring mixed time delays, Neural Comput. Appl. 31 (2017) 65–78.
[7] A. Soualhi, K. Medjaher, N. Zerhouni, Bearing health monitoring based on hilbert–huang transform, support vector machine, and regression, IEEE Trans. Instrum. Meas. 64 (2015) 52–62.
[8] J.B. Ali, B. Chebel-Morello, L. Saidi, S. Malinowski, F. Fnaiech, Accurate bearing remaining useful life prediction based on weibull distribution and artificial neural network, Mech. Syst. Signal Process. 56-57 (2015) 150–172.
[9] P. Lim, C.K. Goh, K.C. Tan, A novel time series-histogram of features (TS-HoF) method for prognostic applications, IEEE Trans. Emerging Top. Comput. Intell. 2 (2018) 204–213.
[10] D. Wu, C. Jennings, J. Terpenny, R.X. Gao, S. Kumara, A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests, J. Manuf. Sci. Eng. 139 (2017) 1–9.
[11] S.A. Aye, P.S. Heyns, An integrated gaussian process regression for prediction of remaining useful life of slow speed bearings based on acoustic emission, Mech. Syst. Signal Process. 84 (2017) 485–498.
[12] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (2015) 436–444.
[13] N. Hou, H. Dong, Z. Wang, W. Ren, F.E. Alsaadi, Non-fragile state estimation for discrete markovian jumping neural networks, Neurocomputing 179 (2016) 238–245.
[14] F. Yang, H. Dong, Z. Wang, W. Ren, F.E. Alsaadi, A new approach to non-fragile state estimation for continuous neural networks with time-delays, Neurocomputing 197 (2016) 205–211.
[15] B. Wang, Y. Lei, N. Li, T. Yan, Deep separable convolutional network for remaining useful life prediction of machinery, Mech. Syst. Signal Process. 134 (2019) 1–18.
[16] L. Guo, Y. Lei, N. Li, T. Yan, N. Li, Machinery health indicator construction based on convolutional neural networks considering trend burr, Neurocomputing 292 (2018) 142–150.
[17] G.S. Babu, P. Zhao, X. Li, Deep convolutional neural network based regression approach for estimation of remaining useful life, in: 21st International Conference on Database Systems for Advanced Applications, Springer, 2016, pp. 214–228.
[18] L. Ren, Y. Sun, H. Wang, L. Zhang, Prediction of bearing remaining useful life with deep convolution neural network, IEEE Access 6 (2018) 13041–13049.
[19] L. Wen, Y. Dong, L. Gao, A new ensemble residual convolutional neural network for remaining useful life estimation, Math. Biosci. Eng. 16 (2019) 862–880.
[20] J. Zhu, N. Chen, W. Peng, Estimation of bearing remaining useful life based on multiscale convolutional neural network, IEEE Trans. Ind. Electron. 66 (2018) 3208–3216.
[21] X. Li, W. Zhang, Q. Ding, Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction, Reliab. Eng. Syst. Saf. 182 (2019) 208–218.
[22] P. Baraldi, F. Mangili, E. Zio, Investigation of uncertainty treatment capability of model-based and data-driven prognostic methods using simulated data, Reliab. Eng. Syst. Saf. 112 (2013) 94–108.
[23] K. Cho, B.V. Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2014, pp. 1724–1734.
[24] G. Hinton, D.V. Camp, Keeping neural networks simple by minimizing the description length of the weights, in: The 6th Annual ACM Conference on Computational Learning Theory, ACM, 1993, pp. 5–13.
[25] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[26] Y. Gal, Z. Ghahramani, Dropout as a bayesian approximation representing model uncertainty in deep learning, in: The 33rd International Conference on Machine Learning, 2016, pp. 1050–1059.
[27] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R.R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, 2012, arXiv:1207.0580.
[28] P. Nectoux, R. Gouriveau, K. Medjaher, E. Ramasso, B. Chebel-Morello, N. Zerhouni, C. Varnier, PRONOSTIA: an experimental platform for bearings accelerated degradation tests, in: IEEE International Conference on Prognostics and Health Management, IEEE, 2012, pp. 1–8.
[29] A. Saxena, J. Celaya, B. Saha, S. Saha, K. Goebel, Metrics for offline evaluation of prognostic performance, Int. J. Prognost. Health Manag. 1 (2010) 4–23.
[30] R.K. Singleton, E.G. Strangas, S. Aviyente, Extended kalman filtering for remaining-useful-life estimation of bearings, IEEE Trans. Ind. Electron. 62 (2015) 1781–1790.
[31] D.P. Kingma, J.L. Ba, Adam: a method for stochastic optimization, 2015, arXiv:1412.6980.
[32] S. Sankararaman, K. Goebel, Uncertainty in prognostics and systems health management, Int. J. Prognost. Health Manag. 6 (2015) 1–14.
[33] A.Z. Hinchi, M. Tkiouat, Rolling element bearing remaining useful life estimation based on a convolutional long-short-term memory network, Proc. Comput. Sci. 127 (2018) 123–132.
[34] T. Benkedjouh, K. Medjaher, N. Zerhouni, S. Rechak, Health assessment and life prediction of cutting tools based on support vector regression, J. Intell. Manuf. 26 (2015) 213–223.
[35] C. Zhang, X. Yao, J. Zhang, H. Jin, Tool condition monitoring and remaining useful life prognostic based on a wireless sensor in dry milling operations, Sensors 16 (2016) 795.
[36] Y. Chen, Y. Jin, G. Jiri, Predicting tool wear with multi-sensor data using deep belief networks, Int. J. Adv. Manuf. Technol. 99 (2018) 1917–1926.
[37] Y. Wang, W. Dai, J. Xiao, Detection for cutting tool wear based on convolution neural networks, in: 2018 12th International Conference on Reliability, Maintainability, and Safety, IEEE, 2018, pp. 297–300.
[38] R. Zhao, R. Yan, J. Wang, K. Mao, Learning to monitor machine health with convolutional bi-directional lstm networks, Sensors 17 (2017) 273.

Biao Wang is currently working toward the Ph.D. degree in mechanical engineering at the Key Laboratory of Education Ministry for Modern Design and Rotor-Bearing System, Xi'an Jiaotong University, P. R. China. He received the B.S. and M.S. degrees in aeronautical and astronautical science and technology from Xi'an Jiaotong University, Xi'an, P. R. China, in 2014 and 2016, respectively.

His-research interests include machinery condition monitoring, intelligent fault diagnostics and remaining useful life prediction of rotating machinery.

Yaguo Lei received the B.S. and Ph.D. degrees in mechanical engineering from Xi'an Jiaotong University, Xi'an, P. R. China, in 2002 and 2007, respectively. He is currently a Full Professor of mechanical engineering at Xi'an Jiaotong University. Prior to joining Xi'an Jiaotong University in 2010, he was a Postdoctoral Research Fellow with the University of Alberta, Edmonton, AB, Canada. He was also an Alexander von Humboldt Fellow with the University of Duisburg-Essen, Duisburg, Germany.

His-research interests include machinery condition monitoring and intelligent maintenance, dynamic modeling, signal processing, intelligent fault diagnosis, and remaining useful life prediction. He has pioneered many fault diagnosis and prognosis methods, and applied them to industrial areas, such as wind turbines, trains and industrial robots, etc.

Dr. Lei is a fellow of IET, a senior member of IEEE, a member of ASME, and a senior member of CMES, ORSC and CAA, respectively. He is currently an associate editor or editorial board member of more than ten journals, including Mechanical Systems and Signal Processing and Neural Computing & Applications.

Tao Yan is currently working toward the Ph.D. degree in mechanical engineering at the Key Laboratory of Education Ministry for Modern Design and Rotor-Bearing System, Xi'an Jiaotong University, P. R. China. He received the B.S. degree in mechanical engineering from Central South University, P. R. China, in 2016.

His-research interests are machinery condition monitoring, intelligent fault diagnostics and remaining useful life prediction of rotating machinery.

Naipeng Li received the B.S. degree in mechanical engineering from Shandong Agricultural University, Shandong, China, in 2012, and the Ph.D. degree in mechanical engineering from Xi'an Jiaotong University, Xi'an, China, in 2019.

He is currently a Lecturer at Xi'an Jiaotong University. His-research interests include machinery condition monitoring, intelligent fault diagnosis and prognostics, and remaining useful life prediction of rotating machinery.

Liang Guo received the B.S. and Ph.D. degrees in mechanical engineering from Southwest Jiaotong University, Chengdu, P. R. China, in 2011 and 2016, respectively. His-research interests focus on machinery intelligent fault diagnostics and remaining useful life prediction.

He is currently an Assistant Professor at Southwest Jiaotong University. Prior to joining Southwest Jiaotong University in 2018, he was a Postdoctoral Research Fellow with the Xi'an Jiaotong University, Xi'an, P. R. China. He was also a visiting Ph.D. with the Clemson University, Clemson, U.S.A.