Measurement 168 (2021) 108286

Contents lists available at ScienceDirect

Measurement

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

Transferable convolutional neural network based remaining useful life prediction of bearing under multiple failure behaviors

Han Cheng, Xianguang Kong, Gaige Chen *, Qibin Wang, Rongbo Wang

School of Mechano-Electronic Engineering, Xidian University, Xi'an 710071, PR China

ARTICLE INFO

Keywords:
Remaining useful life prediction
Transferable convolutional neural network
Domain invariance
Multiple failure behaviors

ABSTRACT

Remaining useful life (RUL) prediction has been a hotspot topic, which is useful to avoid unexpected breakdowns and improve reliability. Different bearing failure behaviors caused by multiple failure modes may lead to inconsistent feature distribution, which affects the prediction model performance. To accurately predict the RUL of bearing under different failure behaviors, a transferable convolutional neural network (TCNN) is proposed to learn domain invariant features. In the proposed method, a convolutional neural network is employed to extract the degradation features. Then multiple-kernel maximum mean discrepancies are integrated into optimization objective to reduce distribution discrepancy. The trained TCNN can be used to predict RUL by feeding data. Its effectiveness is verified by a run-to-failure bearing dataset. The comparison results reveal that the proposed method avoids the influence of kernel selection, improves the performance of domain adaptation effectively, and achieves a better RUL prediction performance.

1. Introduction

Bearing is a crucial rotating component in mechanical equipment. Its degradation will directly affect the performance of the mechanical system. Therefore, accurate prediction of the bearing performance degradation could help the staff to make suitable maintenance plans as early as possible and avoid the shutdowns of the mechanical system caused by unexpected bearing failures. To meet these demands, the remaining useful life (RUL) prediction, one of the main tasks in prognostic and health management (PHM) technology, has received increasing attention in recent years [1–4].

In general, the RUL prediction approaches could be classified into three categories [5]: physics-based approaches [6–8], data-driven approaches [9–11], and hybrid approaches [12]. Compared with the physics-based and hybrid approaches, data-driven approaches are widely used in industrial applications. The data-driven approaches could estimate the degradation process from historical run-to-failure data via machine learning or statistical techniques, without failure mechanisms. For data-driven approaches, extracting effective degradation features from the time domain, the frequency domain, and the time–frequency domain is a significant step to complete accurate predictions. Javed et al. [13] extracted monotonicity features by trigonometric functions and cumulative transformation, and the wavelet-extreme learning machine was employed to prognostic the health trend of bearing. Singleton et al. [14] built a bearing RUL prediction model based on the extended Kalman filter (KF). Soualhi et al. [15] developed a data-driven approach for bearing RUL prediction using the Hilbert-Huang transform (HHT) and the support vector machine (SVM). Ahmad et al. [16] proposed a prognostic method using the dynamic regression model, which utilizes the Relative Root Mean Square as the HI indicator. Pan et al. [17] proposed a two-stage RUL prediction method based on the extreme learning machine to make a higher short-term prediction accuracy in the case of limited learning data. While several studies have shown that excellent and convincing performance, these methods suffer from two main deficiencies. (1) The imperfection of expert knowledge may cause the handcrafted feature to fail to effectively reflect the bearing degradation. (2) Feature extraction and model construction are separate. This may make the model unable to adaptively mine degradation features based on targets. All these factors will affect the performance and efficiency of the model subsequently.

With the collection of abundant condition data, deep learning technology is widely used to analyze complex data. The deep neural network can adaptively learn the representative information through multiple non-linear transformations. Therefore, the deep neural network is helpful to overcome the disadvantages of the above traditional data-driven methods. At present, many researchers have focused more on

* Corresponding author.
E-mail address: chengaige163@163.com (G. Chen).

https://doi.org/10.1016/j.measurement.2020.108286
Received 26 April 2020; Received in revised form 18 July 2020; Accepted 22 July 2020
Available online 28 July 2020

the data-driven approaches based on deep learning technology, such as deep autoencoder network (DAE) [18], convolutional neural network (CNN) [19] and recurrent neural network (RNN) [20]. Some researchers have also explored various extensions of these deep neural networks. Ren et al. [21] proposed the multi-scale dense gate recurrent unit network to capture and ensemble different time-scale attention information through multi-scale layers and dense layers. Li et al. [22] used a multi-scale feature extraction method based on CNN to enhance the model learning ability. Yang et al. [23] proposed the double-convolutional neural networks (CNN) to predict bearing RUL. In double-convolutional neural networks, two CNNs were used to identify the incipient fault point and the bearing RUL respectively. Xiang et al. [24] introduced the attention mechanism to the long-short-term memory neural network (LSTM) to improve the prediction accuracy of RUL.

Despite it can be seen that the data-driven approaches based on deep learning have been widely used to solve RUL prediction, less attention is paid to the influence of multiple failure behaviors on the prediction accuracy of RUL. Owing to the failure mode, the bearing degradation shows different behaviors [25]. Multiple failure behaviors may influence the consistency of feature space, which makes it difficult to satisfy the hypotheses that the training data must have the same distribution with the testing data. This problem may lead to the performance degradation of the conventional deep learning model in the face of different failure behaviors. Besides, existing relevant researches related to multiple failure behaviors mainly focus on reliability estimation, prognostics approaches are scarce [25–28]. Transfer learning provides a feasible idea to solve the problem of inconsistent feature space caused by multiple failure behaviors [29]. Transfer learning can minimize the distribution discrepancy from different domains by sample selection, feature transformation, or parameter adaptation [30]. In the field of PHM, the application of transfer learning has mainly focused on fault diagnosis [31,32], but there are fewer on RUL prediction [33–35]. For the application of transfer learning on bearing RUL prediction, Mao et al. [36] employed transfer component analysis (TCA) to improve the domain adaptation of deep features extracted by contractive denoising autoencoder (CDAE). Zhu et al. [37] presented a transfer learning model based on a three-layers multiple layer perceptron (MLP) to solve the distribution discrepancy problem between different bearing working conditions. The previous studies show that transfer learning can be a promising tool for a high-accuracy RUL predict model under different degradation behaviors.

To overcome the performance degradation of the RUL prediction model caused by different failure behaviors, a transferable convolutional neural network (TCNN) is proposed in this paper. In the proposed method, a multi-layers CNN is utilized to extract the degradation features and find the mapping relationship between the degradation features and the corresponding RUL. Furthermore, multiple-kernel maximum mean discrepancies (MK-MMD), a more robust method than single-kernel method [36,37], are adopted as a regularization term to learn domain invariant feature by minimizing the distribution discrepancy of the feature space between different failure behaviors. In the end, the effectiveness of the proposed method is validated by the run-to-failure bearing datasets. The contributions of this paper are summarized as follows: (1) The TCNN is proposed for bearing RUL prediction under different failure behaviors. The domain adaptation technology in TCNN is used to handle the feature distribution shift caused by different failure behaviors. Therefore, the proposed model has higher accuracy compared with other methods. (2) MK-MMD is employed as the regularization term to complete the transfer learning tasks. By leveraging different kernels to comprehensively evaluate the distribution discrepancy of the feature space, it can effectively avoid the influence of kernel selection and improve the model performance.

The rest of the paper is organized as follows. Section 2 describes the proposed approaches and the training process. Section 3 goes through the implementation of the approaches on an actual case and the results are discussed. Finally, the conclusions are given in Section 4.

2. The proposed method

In this section, the flowchart of the proposed method is first described in detail. Then the CNN is briefly introduced. Afterward, the domain adaptation technology is proposed to solve the problem of inconsistent feature space under different failure behaviors. In the end, the architecture and training process of the proposed method is described.

2.1. Overview of the proposed method

In reality industrial applications, the bearing may be run till failure by one or multiple failure modes. Different failure modes cause different behaviors in bearing degradation process, thus resulting in different data distributions. This is one of the reasons that the traditional model is unstable in the RUL prediction under different bearing failure behaviors. To conquer this problem, a TCNN based on MK-MMD is proposed for RUL prediction in this paper. The idea of domain adaptation is utilized to make the distribution of featured learned from the source domain close to the target domain. The proposal contains three steps: feature extraction, domain adaptation, and RUL prediction, as shown in Fig. 1. Both the labeled samples in the source domain (one failure behavior) and the unlabeled samples in the target domain (another failure behavior) are used as the inputs of the proposed TCNN at first. For the training samples in the source domain, the actual RUL values of each sample are normalized to a range of [0, 1] through dividing by bearing lifetime, and these normalized RUL values are taken as the sample labels [20]. Then, in the step of feature extraction, a multiple-layers CNN is used to extract the degradation features from the sample in the source domain and target domains simultaneously. For domain adaptation, the distribution discrepancy of the extracted features between two domains is measured by MK-MMD. The distribution discrepancy between two domains and the prediction error in the source domain are combined to form the optimization objective, and Adam optimizer is employed to train the TCNN. In the end, the samples in the testing datasets are fed into the trained TCNN to correctly predict the RUL. Since the domain invariant features can be extracted by the TCNN, the TCNN with higher robustness and adaptability can deal with the bearing RUL prediction under multiple failure behaviors.

2.2. Convolutional neural network

CNN is one of the most important types of multilayer feed-forward neural network. Owing to the characteristic of local connection, weight sharing, and spatial pooling, CNN is widely used in many fields. Among a variant of the CNN model, 1-D CNN is suitable for processing 1-D signals in the industrial filed [38]. Its topology structure is shown in Fig. 2.

In general, the 1-D CNN architecture mainly includes the convolutional layer, the pooling layer, and the full connection layer, which are described in detail as follows:

(1) Convolutional layer

The main task of the convolutional layer is to obtain the feature maps through a series of kernel convolutional operations. Each convolutional layer is made of a set of learnable kernels K (also called the filter). The parameters of the kernel mainly are the kernel size, which corresponds to the size of the convolution window, and the kernel depth, which corresponds to the number of the feature map channels. Owing to utilized 1-D convolution, the kernel moves along the input vector X during the feed-forward of the network and the convolutional computation is repeated. The convolutional process is shown in Fig. 2(a). The convolved feature is calculated by the dot product between the weight of the filter and the perceived region of input data, which can be expressed as [38]:

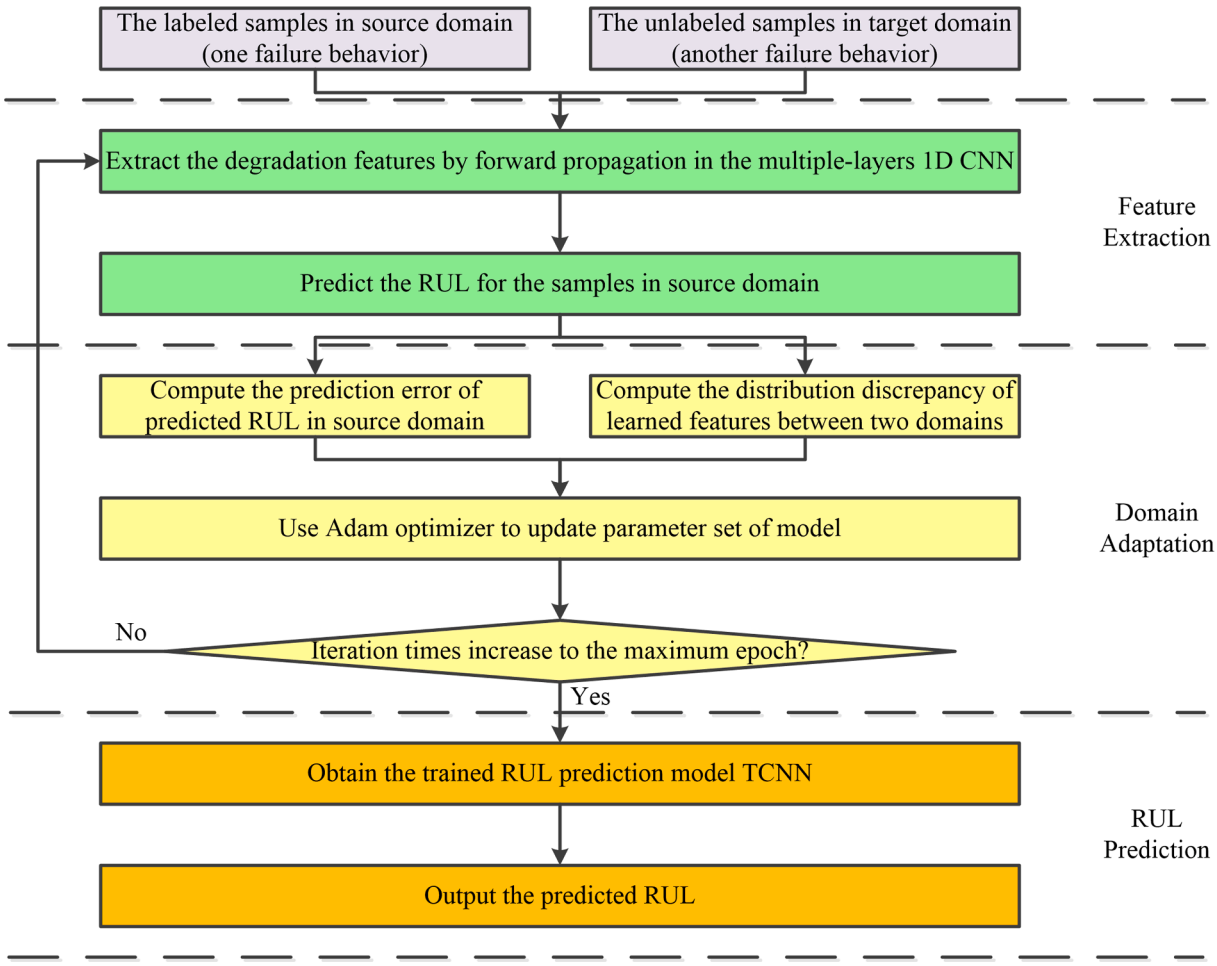

Fig. 1. The procedure of the proposed method.

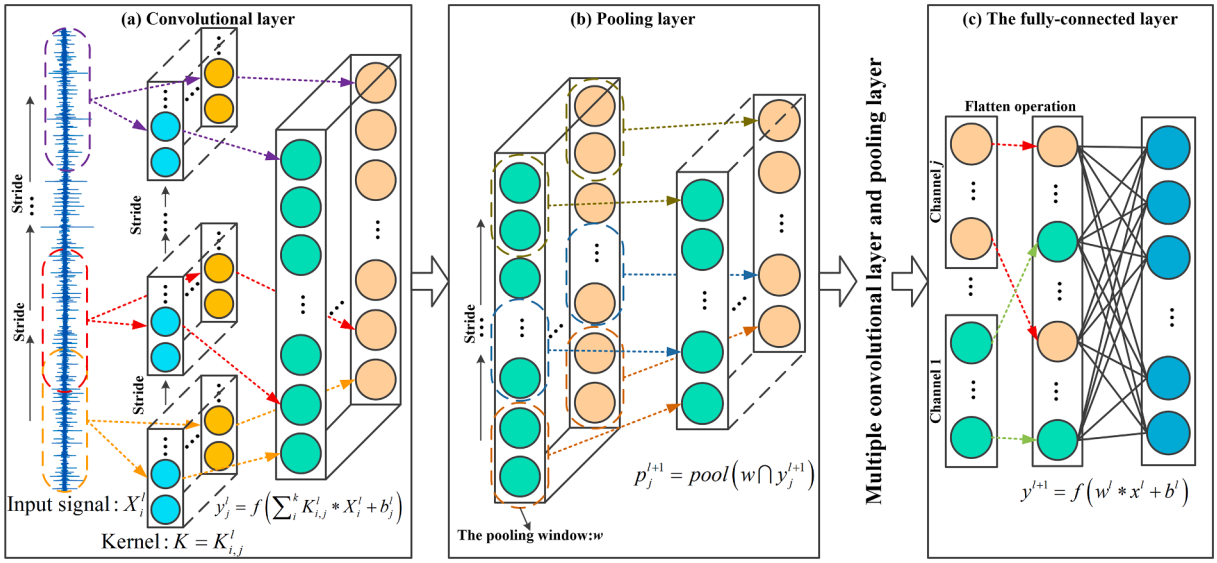

Fig. 2. The topology structure of 1-D CNN model.

$$y_j^l = f\left(\sum_i^k K_{i,j}^l * X_i^l + b_j^l\right) \quad (1)$$

where X_i^l is the input data in layer l. $K_{i,j}^l$ and b_j^l denote the weight and bias vectors respectively. y_j^{l+1} is the output of the jth feature map channel in the layer l. $f(\cdot)$ is the nonlinear activation function that can be used to determine whether the neuron is awakened. The Rectified Linear Unit (ReLU) is used as the activation function in this paper due to its capacities to accelerate the convergence and alleviate the vanishing

gradient problem [39]. The ReLU can be represented as follows:

$$f(x) = max(0, x) \tag{2}$$

(2) Pooling layer

In general, the pooling layer is periodically placed after the convolutional layer. The main task of the pooling layer is to compress the data volume and improve computing efficiency. The pooling operations include the max-pooling and the average-pooling. Since the average-pooling can avoid the loss of the important information with non-maximum, the average-pooling strategy is utilized in this paper, which can be expressed as:

$$p_j^{l+1} = mean\left(w \cap y_j^{l+1}\right) \tag{3}$$

where y_j^{l+1} and p_j^{l+1} represent the input and output of the pooling layer respectively. w represents the pooling window, which can slide with a step. $\cap$ represents the overlap between the pooling window w and the input of the pooling layer y_j^{l+1}. $mean(\cdot)$ represents the operation that takes the mean from the input data.

(3) Fully-connected layer

After multiple the convolutional layers and the pooling layer, the learned feature vectors are flattened into a vector. After the flattened operation, the neurons in fully-connected layers could be fully connected to all neurons in the previous layer. The main task of the fully-connected layer is to further extract the features and connect with the output layer. The calculation of the fully-connected layer could be shown as follows:

$$y^{l+1} = f\left(w^l * x^l + b^l\right) \tag{4}$$

where y^{l+1} and x^l represent the output and input of the full-connected layer l. w^l denote the weight vector, and b^l is the bias vector. $f(\cdot)$ is the nonlinear activation function which is also adopted ReLU in this paper.

2.3. Domain adaptation

In the case of multiple failure behaviors, the probability distribution of different failure behaviors may exhibit significant differences. Since the feature distribution shift, the assumption of the traditional data-driven method is difficult to satisfy. Therefore, the accuracy and the generalization ability of the traditional bearing RUL prediction method under multiple failure behaviors are undesirable. Domain adaptation is an important type of transfer learning. It can automatically apply the model trained in the source domain to different but related target domains [40]. To obtain the desired performance of the RUL prediction method under multiple failure behaviors, domain adaptation could be utilized to learn domain invariant features between the source domain and the target domain. In this situation, the source domain and the target domain represent different failure behaviors respectively. Domain invariant features could make that the learned features subject to the similar distribution no matter from source domain or target domain.

In domain adaptation, the key is to measure the similarity of the distributions between the source domain and the target domain. In this paper, MK-MMD, which is the multiple kernel variant of maximum mean discrepancies (MMD), is used to measure the distribution discrepancy [41,42]. MMD could quantify the difference of the distributions by the squared distance of the instances in the reproducing kernel Hilbert space (RKHS), which is calculated as follows [43,44]:

$$MMD\left(X^S, X^T\right) = \left\| \frac{1}{n^S} \sum_{i=1}^{n^S} \Phi\left(x_i^S\right) - \frac{1}{n^T} \sum_{j=1}^{n^T} \Phi\left(x_j^T\right) \right\|_{\mathscr{H}}^2 \tag{5}$$

where $\mathscr{H}$ denotes the RKHS. $X^S = \left\{x_i^S\right\}_{i=1,\ldots,n^S}$ and $X^T = \left\{x_j^T\right\}_{j=1,\ldots,n^T}$ are source data and target data drawn from different distribution respectively. To simplify the calculation, the feature map $\Phi(\cdot)$ is computed by kernel trick k, which is computed as follows:

$$k\left(x_i^S, x_j^T\right) = \left\langle \Phi\left(x_i^S\right), \Phi\left(x_j^T\right) \right\rangle \tag{6}$$

For MMD, kernel selection affects the performance of MMD. This is because different kernels may embed probability distributions in different RKHSs, which emphasize the different orders of sufficient statistics [41]. Therefore, selecting a suitable kernel can guarantee the model has better performance. MK-MMD could leverage a mixture of multiple kernels to enhance effectiveness, thereby developing a principled method for optimal kernel selection. MK-MMD can be calculated as follows:

$$k\left(x_i^S, x_j^T\right) = \sum_{i=1}^{K} k_i\left(x_i^S, x_j^T\right) \tag{7}$$

where $k_i (i = 1, \ldots, K)$ represents different kernel methods.

2.4. Transferable convolutional neural network (TCNN)

2.4.1. Network architecture

The basic architecture of the proposed TCNN is shown in Fig. 3, which is comprised of three modules: feature extraction module, RUL prediction module, and domain adaptation module. The feature extraction module could automatically learn multiple high-level degradation features from the samples both in the source and target domains. The feature extraction module is formed by three 1-D convolutional layers (denoted as C1, C2, and C3 respectively), three average-pooling layers (denoted as P1, P2, and P3 respectively) and three fully-connected layers (denoted as FC1, FC2, and FC3 respectively). In the feature extraction module, the input data are fed into these three 1-D convolutional layers at first. In each convolutional layer, the zeros-padding operation is implemented to ensure that the boundary information is not lost. Further, each convolutional layer is followed by one average-pooling layer. After three convolutional and pooling operations, the learned features from layer P3 is flattened and fed into three fully-connected layers layer-by-layer. In this way, the high-level degradation features in source and target domain (denoted as FC3S and FC3T respectively) could be further learned. In the end, the learned high-level degradation features from FC3 are fed into the RUL prediction module and domain adaptation module. RUL can be obtained by the fully-connected layer (denoted as FC4) in the RUL prediction module and the distribution difference of extracted features between the source domain and the target domain is measured by domain adaptation module.

2.4.2. Optimization objective

To properly adjust neuron parameters effectively, the proposed TCNN has two optimization objects: (1) the prediction loss L_R between the observed RUL values and the estimated RUL values in the source domain; (2) the distribution discrepancy loss L_D of learned high-level representations between two domains.

The first optimization object L_R of TCNN is to minimize the prediction error in the source domain, which could lead the feature extractor and the RUL estimator to learn the mapping relation between the features and RUL. The mean absolute error (MAE) is used to define the prediction error L_R, which is mainly used in the regression problem. The MAE function can be written as:

$$L_R = \frac{1}{N_S} \sum_{i=1}^{N_S} \left| \tilde{y}_i^S - y_i^S \right| \tag{8}$$

where y_i^S and $\tilde{y}_i^S$ are the observed RUL values and the estimated RUL

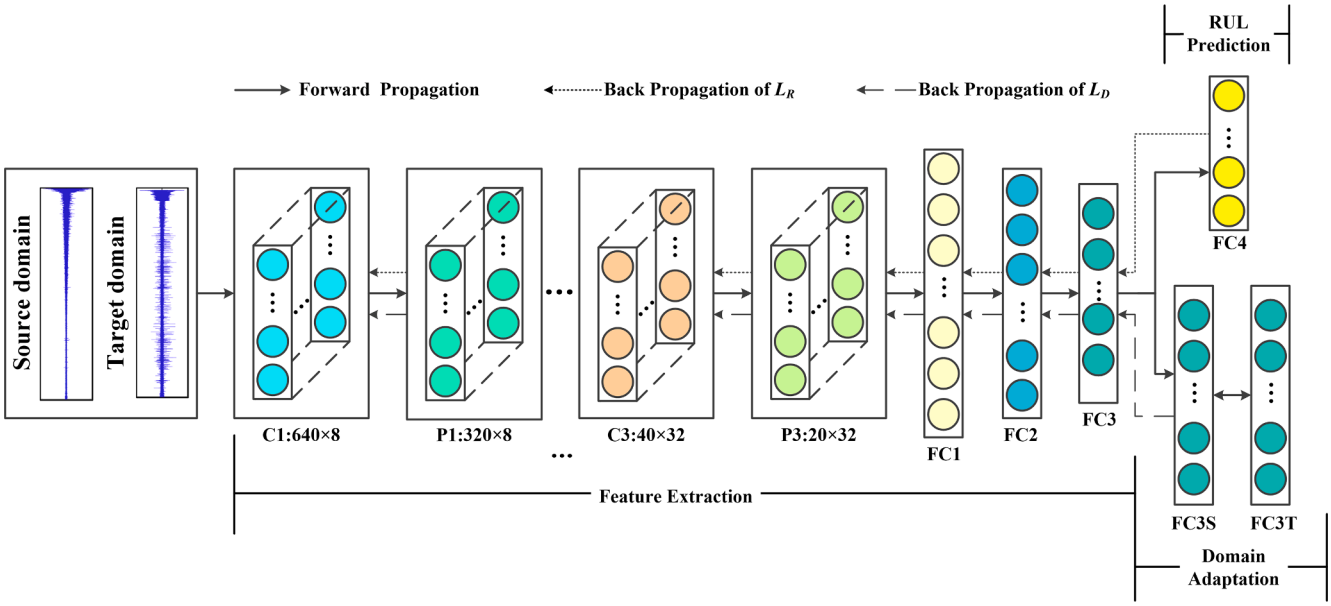

Fig. 3. The architecture of the proposed TCNN.

values both from the source domain, respectively. N_S is the number of samples in the source domain.

The second optimization object L_D is to minimize the distribution discrepancy of learned high-level representations between the source domain and target domain, which could lead the parameters of the feature extractor to learn the domain invariant features simultaneously. According to Section 2.3, MK-MMD is used as the second optimization object. The second optimization object L_D is defined as:

$$L_D = \sum_{k\in K} MMD_k\left(F^S, F^T\right) \tag{9}$$

where F^S and F^T denote the high-level features extracted by feature extractor in the source domain and target domain respectively. K denotes the kernel set.

$$L = L_R + \lambda L_D = \frac{1}{N_S}\sum_{i=1}^{N_S}\left|\tilde{y}_i^S - y_i^S\right| + \lambda\sum_{k\in K} MMD_k\left(F^S, F^T\right) \tag{10}$$

where λ denotes the tradeoff parameter.

Once the loss function L of TCNN is built, the optimal parameters of TCNN can be searched by optimizing the loss function L. So, the loss function Eq. (11) is rewritten as follows:

$$L(W) = \min_W L_R(W) + \lambda L_D(W) \tag{11}$$

where W is the parameter set of TCNN. Then, W is updated as follows:

$$W \leftarrow W - \varepsilon\left(\frac{\partial L_R}{\partial W} + \lambda\frac{\partial L_D}{\partial W}\right) \tag{12}$$

where ε is the learning rate. In this paper, Adam optimizer is employed as the optimization method to update the parameters [45]. This process is repeated until the loss converged to the expected value. Besides, the mini-batch gradient descent is employed to divide the training dataset into multiple mini-batches. The model is trained by the mini-batch in the iteration, which can reduce the calculation cost. When TCNN completes the training process, the domain invariant features of the sample under different failure behaviors can be learned.

3. Experimental results and discussion

3.1. Case introduction

The experimental data, which was shared in the IEEE international conference of PHM 2012, acquired from the accelerated aging PRONOSTIA platform. The PRONOSTIA platform is shown in Fig. 4 and the detail is described in [46]. On the PRONOSTIA platform, two types of measurements can be acquired, i.e. vibration and temperature. For vibration signal, two high-frequency accelerometers 3035B DYTRAN were used to measure the vibration signals in the horizontal and vertical direction, respectively. The sampling frequency is 25.6 kHz, and 2560 data points are recorded every 10 s. In the run-to-failure experiment, all the testing bearings experienced a naturally degraded until failure, i.e., no defects were seeded into these testing bearings at the beginning of the tests, which is common in the realistic industrial scenario. The tests were stopped when the amplitude of the vibration signal exceeded 20 g. The bearing failure may be caused by the failure of balls, rings, or cage or their combinations. These bearing failure modes lead to different degradation trends under the same operational condition. Therefore, the horizontal vibration signals of seven bearings under the operating conditions of 1800 rpm and 4000 N are used to verify the proposed method [22]. The degradation trends of each bearing are shown in Fig. 5. According to the conclusions of the change point detection algorithm (CPDA) in the references [20], seven bearing datasets are divided into two types of failure behavior. Bearings 1, 3, and 4 belong to the same type of failure behavior, denoted as failure behavior 1 (FB1), whereas bearings 2, 5, 6, and 7 belong to another type of failure behavior, denoted as failure behavior 2 (FB2). To verify the proposed method, FB1 and FB2 are selected as the source domain and the target domain respectively. Further, all datasets are split into training datasets and testing datasets, where bearing 1 and 2 are selected as training datasets, and bearing 3, 4, 5, 6, and 7 are selected as testing datasets.

3.2. Approaches implementation

To evaluate the performance of the proposed TCNN, the Fast Fourier transform (FFT) is utilized on each dataset to compute the frequency spectrum at first. Then, to eliminate the large magnitude difference and speed up the learning process, each dataset is normalized individually using:

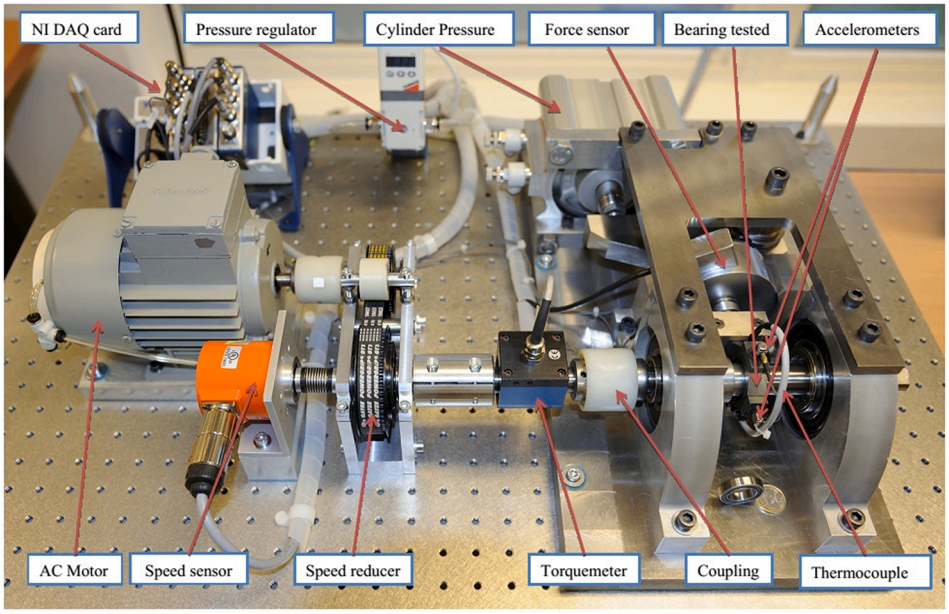

Fig. 4. The PRONOSTIA platform.

$$x_{norm} = \frac{x - x_\mu}{x_\sigma} \tag{13}$$

where x_μ and x_σ are the mean and variance of the data respectively.

After data preprocessing, the proposed TCNN is defined and the structure parameter of the model is listed in Table 1. For training the model, some hyper-parameters are set. In this paper, the maximum training epoch is set to 500 and the initial learning rate of Adam optimizer is 0.0005. In each epoch, mini-batch size is set as 128 to randomly divide the training samples into multiple mini-batches. A mixture of five RBF kernels is selected in MK-MMD and the default bandwidth parameters are selected as 0.25, 0.5, 1, 2, and 4 [42,47]. The tradeoff parameter is set to 0.5. The training experiments are implemented using Pytorch with 7300 CPU and 8G RAM.

To evaluate the performance of the proposed method, four performance metrics are utilized to measure the difference between the actual RUL y^{real} and the corresponding predicted RUL y^{pre}, namely mean absolute error (MAE), root mean square error (RMSE), R-square (R2), and the percent error of prediction results (Er) [20], which are defined as:

$$MAE = \sum_{i=1}^{N} \left| y_i^{real} - y_i^{pre} \right| / N \tag{14}$$

$$RMSE = \sqrt{\sum_{i=1}^{N} \left(y_i^{real} - y_i^{pre} \right)^2 / N} \tag{15}$$

$$R2 = 1 - \frac{\sum_{i=1}^{N} \left(y_i^{real} - y_i^{pre} \right)^2}{\sum_{i=1}^{N} \left(y_i^{real} - \bar{y} \right)^2} \tag{16}$$

$$Er_i = \frac{y_i^{real} - y_i^{pre}}{y_i^{real}} \times 100\% \tag{17}$$

$$\bar{y} = \sum_{i=1}^{N} y_i^{real} / N \tag{18}$$

where *Er* is applicable to evaluate the percent error of prediction results at a certain moment.

3.3. Results and discussion

In this section, the performance of the proposed method on the PHM 2012 dataset is presented. To fully verify the performance of the proposed method, three experiments are used to conduct. The first one is to discuss the effect of the hyper-parameter and the network architecture of the proposed method. The second one is to compare the performance of the proposed method with related models. The third one is to compare the accuracy of the RUL prediction results at a certain moment with related works. For performance metrics, MAE, RMSE, and R2 are used in the first and second experiments and *Er* is used in the third experiment. To reduce the effect of randomness, the training process of the model is repeated five times in each experiment, and five testing results are obtained after each repeated experiment. The average values of the repeated experiment are taken as the final testing result. The RUL prediction results on the five testing datasets by TCNN are shown in Fig. 6. Due to the influence of the local fluctuations, the RUL trends of bearings are highlighted by the downsampling technology. It can be seen from Fig. 6 that the RUL trend of bearings with different failure behaviors can be well displayed.

3.3.1. Hyper-parameters and network architecture discussion

In experiment 1, the hyper-parameter and network architecture are discussed. Evaluated hyper-parameters are the initial learning rate of Adam optimizer and tradeoff, which values are listed in Table 2. Fig. 7 shows boxplots of the results of the initial learning rate and the tradeoff on all testing datasets. To quantize the performance of the proposed method, the mean and standard deviation of the evaluated metrics on all testing datasets for the initial learning rate and tradeoff are provided in Tables 3 and 4 respectively.

For the initial learning rate, it can be seen from Fig. 7 (a), (c), (e) and Table 3 that the performance of the proposed method is improved with the decrease of the initial learning rate. This is because that the small initial learning rate is more conducive to gradual convergence and avoids missing the minimum value. When the learning rate is 0.0001, the performance of the proposed method is worse than the model with the learning rate of 0.0005. This is because the smaller learning rate needs longer training epoch to converge. Besides, the tradeoff is another hyper-parameter that needs to be discussed and the results are shown in Fig. 7 (b), (d), (f) and Table 4. According to the boxplot, it is found that the performance is not significantly affected by the tradeoff when the

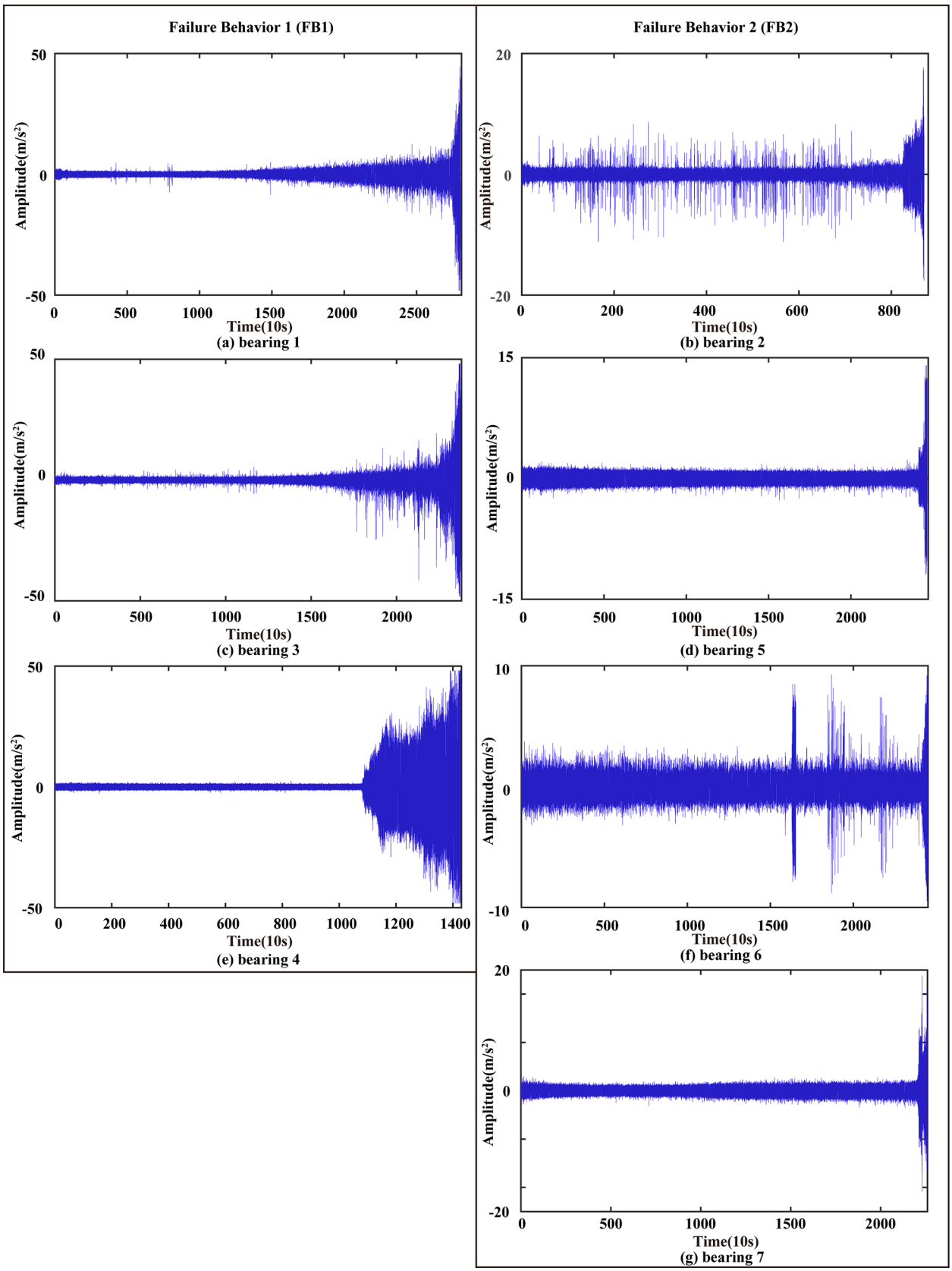

Fig. 5. The vibration signals of each testing bearings under different failure behaviors.

Table 1
Model structure of TCNN.

No.	Layer	Tied Parameters	Activation Function	Output Size
1	Input Layer	\	\	1280 × 1
2	C1	Kernel Number: 8 Kernel Size: 4 × 1 Stride: 2 × 1	ReLU	640 × 8
3	P1	Kernel Size: 2 × 1 Stride: 2 × 1	\	320 × 8
4	C2	Kernel Number: 16 Kernel Size: 4 × 1 Stride: 2 × 1	ReLU	160 × 16
5	P2	Kernel Size: 2 × 1 Stride: 2 × 1	\	80 × 16
6	C3	Kernel Number: 32 Kernel Size: 4 × 1 Stride: 2 × 1	ReLU	40 × 32
7	P3	Kernel Size: 2 × 1 Stride: 2 × 1	\	20 × 32
8	Flatten	\		640
9	FC1	Weights: 640 × 160 Bias: 160 × 1	ReLU	160
10	FC2	Weights: 160 × 40 Bias: 40 × 1	ReLU	40
11	FC3	Weights: 40 × 10 Bias: 10 × 1	ReLU	10
12	FC4	Weights: 10 × 1 Bias: 1 × 1	\	1

tradeoff value in the range [1, 0.1]. That suggests the proposed method can keep stable in a reasonable tradeoff range.

Furthermore, the effect of network architecture should be discussed. In general, domain distribution discrepancy exists in all layers of networks. The position of the layer with MMD minimization may affect the performance of the model. In this test, the performance of the network with a single layer and multiple layers MMD minimization in three fully-connected layers is evaluated, i.e. MMD(FC1), MMD(FC2), MMD(FC3), MMD(FC1,2), MMD(FC1,3), MMD(FC2,3) and MMD(FC1,2,3) (denoted as MMD1, MMD2, MMD3, MMD12, MMD13, MMD23, MMD123 respectively). As stated in the references [42] and [47], the deep features must transition from general to specific when the data is passed from the lower layers to the higher layers. So the lower layers mainly extract generic features from signals. The distribution discrepancy of these generic features is not obvious. The higher layers are responsible for extracting the high-level abstract features which are specific to the target task. Owing to the difference of the target task, the domain shift is more obvious in higher layers. This is the reason that only the fully-connected layer is evaluated.

The average results and boxplots on all testing datasets are shown in Table 5 and Fig. 8. The results of the method with single layer MMD minimization are compared at first. The results of the MMD3 method provides relatively better results than the MMD2 method and the MMD1 method. This is because the last layer with MMD minimization could allow the extracted features to remain high relevance to the target task and improve domain invariance. Furthermore, comparing the MMD13 and MMD23 method with the MMD1, MMD2, and MMD3 method can be found that the results of the MMD13 and MMD23 method are higher than the MMD1, lower than the MMD3 method. This indicates that the performance of the MMD3 method will be reduced by further ensuring the domain invariance of the FC1 layer or the FC2 layer. Besides, the model performance that the FC1 layer and the FC2 layer are combined with the FC3 layer respectively for domain adaptation is higher than that of the FC1 layer or the FC2 alone. For further verification, the MMD12 and MMD123 method with the MMD1, MMD2, and MMD3 method are compared. It is found that the combination of the FC1 layer and the FC2 layer results in the performance degradation of the method with single layer MMD minimization. By comparing the MMD12 method and MMD123 method, it can be found that although minimizing the distribution discrepancy of the FC3 layer can improve the performance of the MMD12 method, but the performance of the MMD123 method is just higher than the MMD1 method and the MMD12 method. Therefore, the proposed method could achieve the best performance by only minimizing the distribution discrepancy of the FC3 layer.

3.3.2. *Comparison with related models*

In experiment 2, three types of RUL prediction methods are used for comparison to verify the effectiveness of the proposed method. Three types of RUL prediction methods are CNN without transfer learning, TCNN with single kernel MMD (TCNN-SK), and other types of transfer learning respectively. The test process based on different comparison purposes and the comparison results are as follows:

(1) **Comparison with CNN without transfer learning:** The purpose of the first comparison method is to verify the improvement of the proposed method in RUL prediction with unlabeled target domain data. The comparison methods are CNN model trained only in source domain data (CNN-S), only in target domain data (CNN-T), and both in source and target domain data (CNN-ST) respectively. Then all methods are applied on all testing datasets, where CNN-T and CNN-ST represent the situation when target labels are available. The hyper-parameters and network architecture of CNN are the same as the TCNN. The training loss curve and the results of various methods on five testing datasets are shown in Fig. 9 and Table 6. As can be seen from Fig. 9, the training loss curves can converge as the training epoch increases. The minimum convergence value of TCNN is higher than that of CNN-S and CNN-T, which is because that the distribution difference between two domains is considered in the TCNN. Although the minimum convergence value and convergence speed of TCNN are not as good as other methods, it can be seen from Table 6 that the TCNN has the best prediction performance on testing datasets. Compared with the CNN-S method and CNN-T method, the proposed method could maintain stable performance in two kinds of failure behavior datasets. The difference between the CNN-ST method and TCNN method is that the target domain datasets of the CNN-ST method have labels in the training process. But the proposed method has higher average performance than the CNN-ST method, especially on the bearing 5 dataset. It indicates that the proposed method is more effective in solving bearing RUL prediction under multiple failure behaviors.

(2) **Comparison with single kernel MMD method:** The second comparison method is designed to illustrate the effects of MK-MMD in optimal kernel selection. Five TCNN with single kernel MMD is used for comparison, where kernels are select as 0.25, 0.5, 1, 2, and 4 respectively. The results of various methods are listed in Table 7. According to the result of all TCNN-SK methods, the optimal kernel can be selected as 0.5 or 1. However, the proposed method could achieve the performance of the TCNN-SK with the optimal kernel without selecting the optimal kernel. Therefore, MK-MMD is helpful to improve the efficiency of transfer learning based on single kernel MMD.

(3) **Comparison with other types of transfer learning:** The third comparison method is employed to compare two transfer learning methods, i.e., Transferable Multiple Layer Perceptron (TMLP) [37] and Domain-adversarial training of neural networks (DANN) [48]. TMLP adds a domain classifier and a domain distribution discrepancy metric in a three layers MLP. The input data of TMLP is 25 handcrafted features which are the time domain features (mean, RMS, kurtosis, skewness, peak-to-peak, variance, entropy, crest factor, wave factor, impulse factor, margin factor, related-similarity), the frequency domain features (related-similarity of four sub-bands frequency spectra) and the time–frequency domain features (energy ratios of eight frequency sub-bands generated by haar wavelet package transform) [20]. DANN is a deep neural network with a domain discriminative

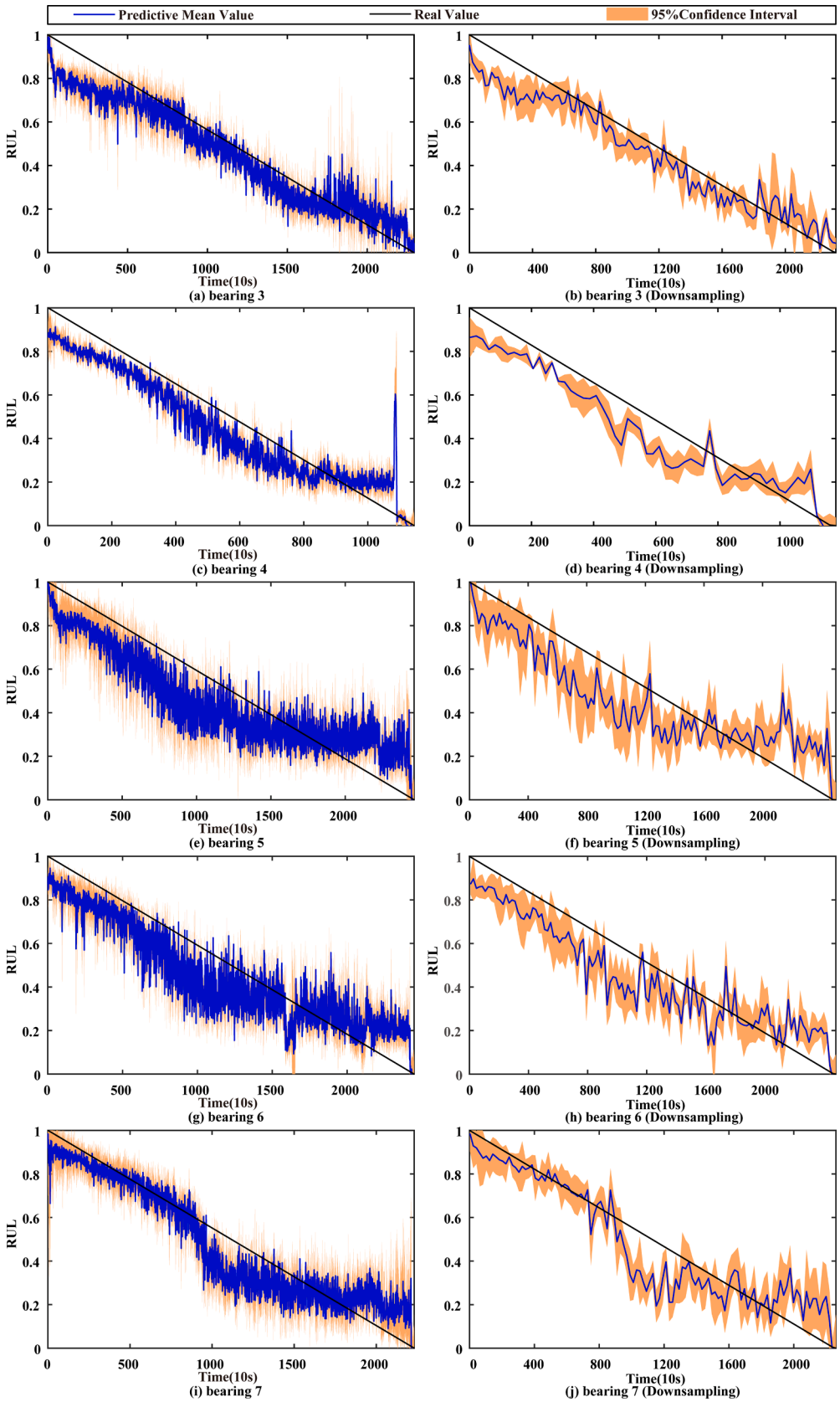

Fig. 6. The RUL prediction results of each testing datasets obtained by TCNN and its downsampling result.

Table 2
The list of evaluated hyper-parameter values.

Hyper-parameters	Ranges
Initial learning rate	{0.01, 0.005, 0.001, 0.0005, 0.0001}
Tradeoff	{0.1, 0.5, 1, 5, 10, 50, 100}

component. In this experiment, the architecture of DANN and the input data are the same as the TCNN. For both methods, two optimal hyper-parameters are also searched from Table 2. The results of TMLP and DANN on testing datasets are listed in Table 8. From the results, it can be found that TCNN outperforms two other transfer learning. This validates that TCNN can extract features with both domain invariance and task relevance more effectively than the other two transfer learning methods. For TMLP, the handcrafted feature set may also have an impact on performance. This is because the extracted features from handcrafted features may discard some representational information related to the learning task.

Table 3
Results of the proposed method with different initial learning rates.

No.	Initial learning rate	MAE	RMSE	R2
1	0.0001	0.10 ± 0.02	0.13 ± 0.02	0.80 ± 0.07
2	0.0005	**0.10 ± 0.02**	**0.12 ± 0.02**	**0.82 ± 0.07**
3	0.001	0.10 ± 0.02	0.12 ± 0.02	0.81 ± 0.07
4	0.005	0.11 ± 0.03	0.13 ± 0.03	0.78 ± 0.11
5	0.01	0.10 ± 0.02	0.13 ± 0.03	0.79 ± 0.08

Table 4
Results of the proposed method with different initial learning rates.

No.	Tradeoff	MAE	RMSE	R2
1	100	0.13 ± 0.03	0.15 ± 0.03	0.71 ± 0.11
2	50	0.12 ± 0.03	0.15 ± 0.03	0.73 ± 0.10
3	10	0.11 ± 0.02	0.14 ± 0.02	0.77 ± 0.07
4	5	0.11 ± 0.02	0.13 ± 0.02	0.79 ± 0.07
5	1	0.10 ± 0.02	0.12 ± 0.02	0.81 ± 0.07
6	0.5	**0.10 ± 0.02**	**0.12 ± 0.02**	**0.82 ± 0.07**
7	0.1	0.10 ± 0.03	0.13 ± 0.03	0.80 ± 0.10

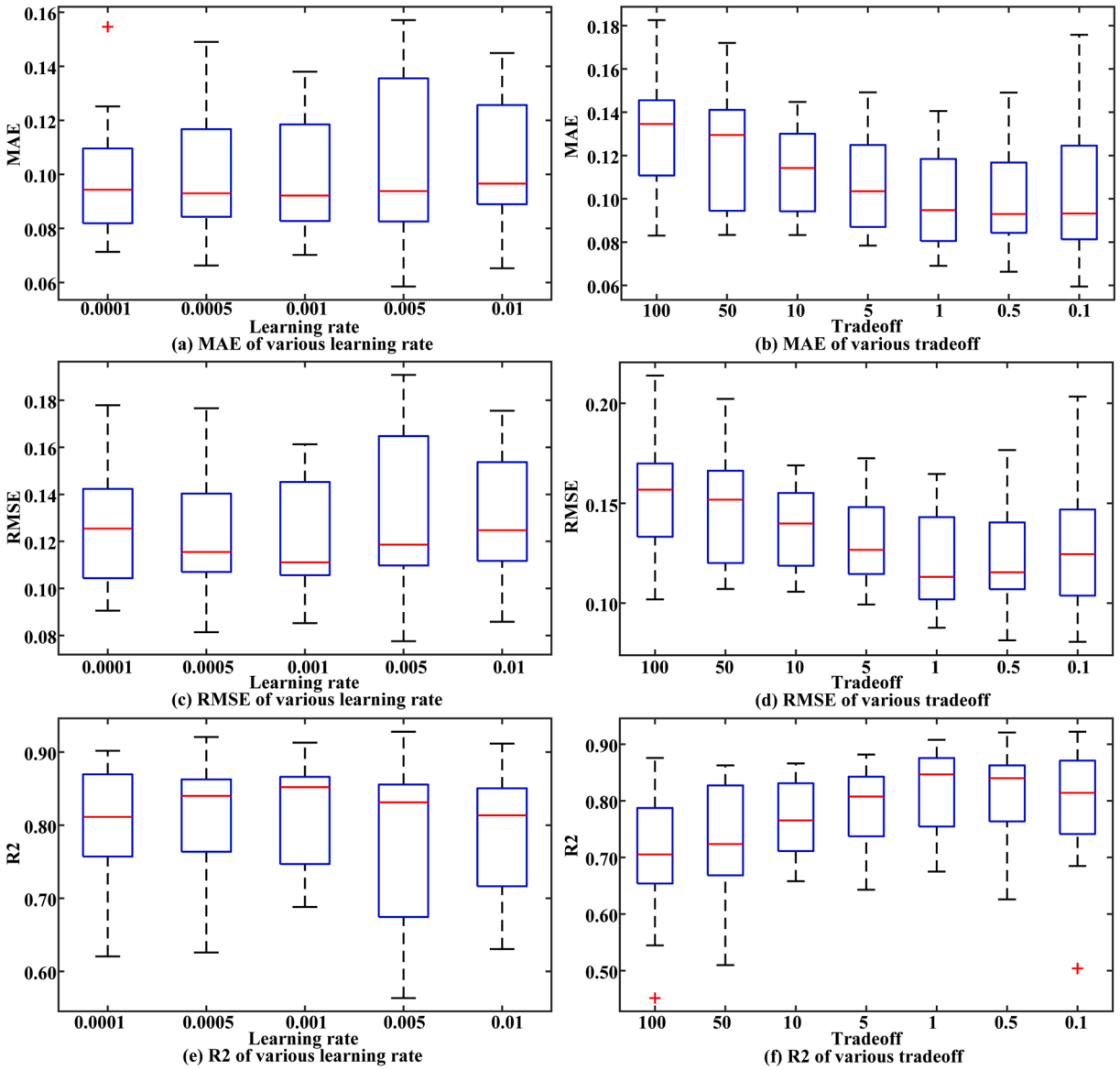

Fig. 7. Results using various initial learning rates and tradeoffs of the proposed method.

Table 5
Results of the proposed method with different layer domain adaptation.

No.	The layer with domain adaptation	MAE	RMSE	R2
1	MMD1	0.12 ± 0.02	0.14 ± 0.02	0.74 ± 0.08
2	MMD2	0.11 ± 0.02	0.13 ± 0.02	0.79 ± 0.08
3	MMD3	**0.10 ± 0.02**	**0.12 ± 0.02**	**0.82 ± 0.07**
4	MMD12	0.11 ± 0.02	0.14 ± 0.02	0.77 ± 0.07
5	MMD13	0.11 ± 0.02	0.13 ± 0.02	0.78 ± 0.08
6	MMD23	0.10 ± 0.02	0.12 ± 0.02	0.81 ± 0.08
7	MMD123	0.11 ± 0.02	0.13 ± 0.02	0.78 ± 0.08

3.3.3. Comparison with results in related works

In experiment 3, the accuracy of the RUL prediction result at a certain moment is compared with related works to further evaluate the performance of the proposed method. According to the reference [20], the evaluation times of each testing bearing are shown in Table 9. To eliminate the effects of the local fluctuations and compare with the references [20,49–51], all predicted RUL before the evaluation time are fitted by the linear equation $y = at + b$ and the least square method at first. Then, the fitted RUL is converted by Eq. (19) [52].

$$ConvRUL_t = \frac{RUL_t}{1 - RUL_t} \times P_t \tag{19}$$

where P_t is the evaluation time, RUL_t and $ConvRUL_t$ are the fitted RUL value and the converted RUL value at P_t. To evaluate the performance of the model on five bearings, $\overline{Er}$ and $\overline{|Er|}$ is also considered in this paper [49]. Such two scores can be obtained as follow:

$$\overline{Er} = \frac{1}{N}\sum_{i=1}^{N} Er_i \tag{20}$$

$$\overline{|Er|} = \frac{1}{N}\sum_{i=1}^{N} |Er_i| \tag{21}$$

The results of this paper and related papers [20,49–51] are listed in Table 9, where the 4th and 5th columns are the results of this paper. As can be seen from Table 9, the proposed method in this paper has the lowest percent error and the second-lowest absolute percent error. This result strengthens the usability of the proposed method for RUL prediction.

In summary, the above results verify the effectiveness of the proposed TCNN model and indicate that the proposed method can effectively extract domain invariant features, improve the accuracy of RUL prediction under multiple failure behaviors, and simplify the optimal kernel selection strategy of MMD. Despite the TCNN has high accuracy, it still suffers two disadvantages. Firstly, the selection of the optimal

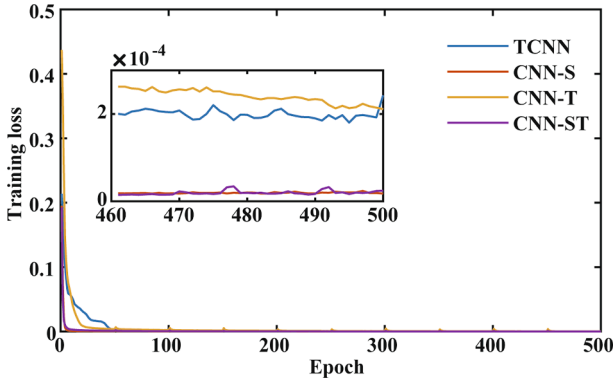

Fig. 9. Training loss curve of TCNN, CNN-S, CNN-T, and CNN-ST.

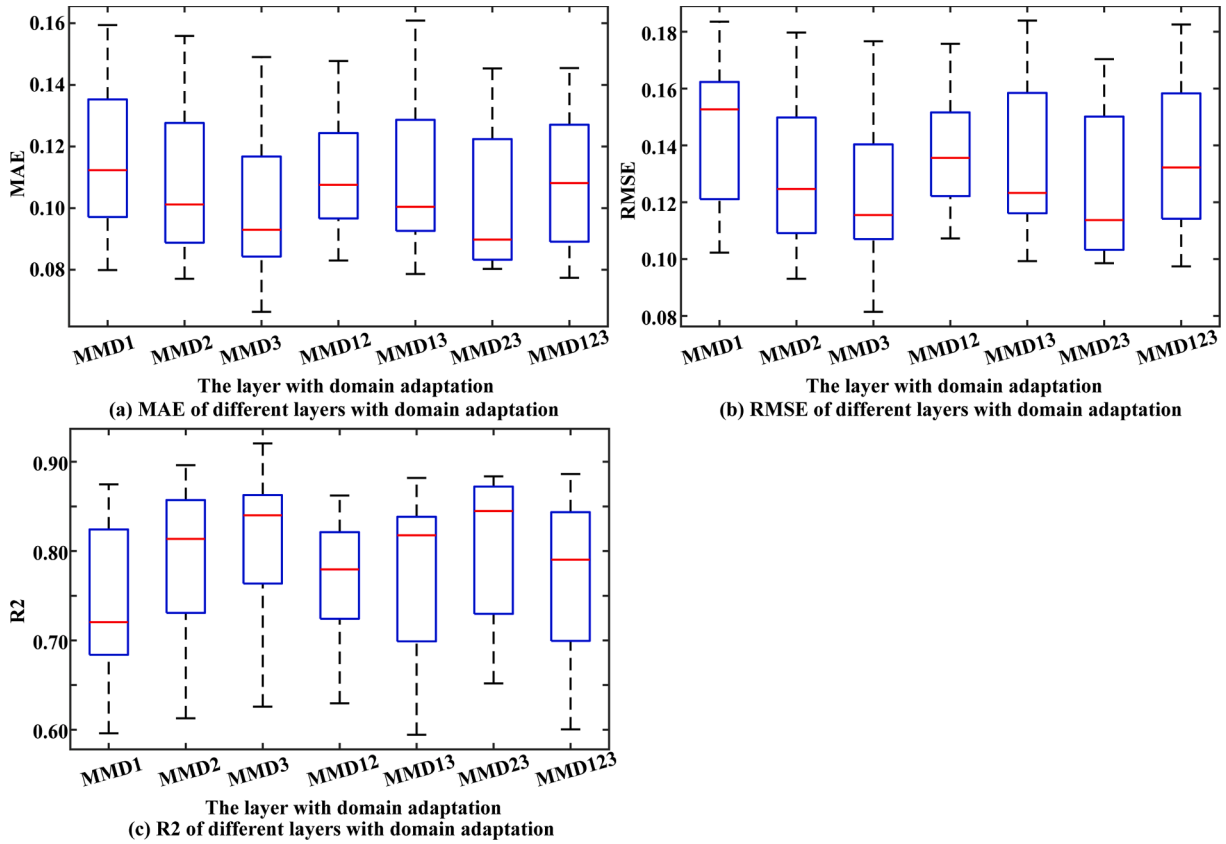

Fig. 8. The evaluated metrics of the proposed method with different layer domain adaptation.

Table 6
Results of the proposed method and CNN without transfer learning method.

Method		Bearing 3	Bearing 4	Bearing 5	Bearing 6	Bearing 7	Average
TCNN	MAE	0.08 ± 0.01	0.09 ± 0.01	**0.13 ± 0.01**	**0.12 ± 0.01**	**0.09 ± 0.01**	**0.10 ± 0.02**
	RMSE	0.10 ± 0.01	0.11 ± 0.01	**0.15 ± 0.02**	**0.14 ± 0.01**	**0.11 ± 0.01**	**0.12 ± 0.02**
	R2	0.89 ± 0.03	0.86 ± 0.01	**0.72 ± 0.06**	**0.76 ± 0.02**	**0.85 ± 0.01**	**0.82 ± 0.07**
CNN-S	MAE	0.07 ± 0.01	0.18 ± 0.01	0.23 ± 0.01	0.20 ± 0.01	0.19 ± 0.01	0.17 ± 0.06
	RMSE	0.09 ± 0.01	0.21 ± 0.01	0.26 ± 0.01	0.24 ± 0.01	0.23 ± 0.02	0.20 ± 0.06
	R2	0.91 ± 0.01	0.48 ± 0.04	0.19 ± 0.06	0.31 ± 0.05	0.39 ± 0.09	0.46 ± 0.26
CNN-T	MAE	0.09 ± 0.01	0.08 ± 0.01	0.12 ± 0.02	0.11 ± 0.01	0.14 ± 0.01	0.11 ± 0.02
	RMSE	0.11 ± 0.02	0.11 ± 0.01	0.16 ± 0.02	0.14 ± 0.01	0.16 ± 0.01	0.14 ± 0.03
	R2	0.85 ± 0.05	0.86 ± 0.02	0.70 ± 0.09	0.76 ± 0.04	0.67 ± 0.03	0.77 ± 0.10
CNN-ST	MAE	**0.06 ± 0.01**	**0.08 ± 0.01**	0.16 ± 0.02	0.12 ± 0.02	0.09 ± 0.01	0.10 ± 0.04
	RMSE	**0.08 ± 0.01**	**0.11 ± 0.01**	0.19 ± 0.02	0.15 ± 0.02	0.12 ± 0.01	0.13 ± 0.04
	R2	**0.93 ± 0.01**	**0.86 ± 0.01**	0.57 ± 0.08	0.74 ± 0.06	0.83 ± 0.03	0.79 ± 0.13

Table 7
Results of the proposed method and TCNN with single kernel MMD.

No.	Method	MAE	RMSE	R2
1	TCNN	**0.10 ± 0.02**	**0.12 ± 0.02**	**0.82 ± 0.07**
2	TCNN-SK (kernel = 0.25)	0.11 ± 0.03	0.13 ± 0.03	0.79 ± 0.11
3	TCNN-SK (kernel = 0.5)	0.10 ± 0.03	0.12 ± 0.03	0.81 ± 0.09
4	TCNN-SK (kernel = 1)	0.10 ± 0.03	0.12 ± 0.03	0.81 ± 0.09
5	TCNN-SK (kernel = 2)	0.13 ± 0.03	0.15 ± 0.03	0.70 ± 0.13
6	TCNN-SK (kernel = 4)	0.14 ± 0.04	0.16 ± 0.04	0.67 ± 0.14

hyper-parameters is difficult owing to the large number of hyper-parameters. Secondly, the training process of TCNN is time-consuming. Therefore, the network structure with high computing efficiency and a small number of parameters will be explored further [53].

4. Conclusions

A TCNN method is proposed for bearing RUL prediction under multiple failure behaviors. The proposed method involves three stages: feature extraction, domain adaptation, and RUL prediction. In the method, the domain adaptation technology is used to solve the inconsistent of the feature distribution between different failure behaviors. The effectiveness of the proposed method is verified by a run-to-failure bearing dataset. Several comparison experiments are also conducted to verify the ability of the proposed method in RUL prediction under two failure behaviors. For bearing RUL prediction under multiple failure behaviors, it can be drawn that: (1) the proposed method obtains higher accuracy than the standard method without transfer learning processing and two transfer learning method (TMLP and DANN); (2) the proposed method can avoid the influence of kernel selection and improve the performance of domain adaptation technology effectively. In further work, the network configurations and hyper-parameter optimizations in high efficiency could be valuable related research points.

CRediT authorship contribution statement

Han Cheng: Methodology, Software, Writing - original draft, Writing - review & editing. **Xianguang Kong:** Resources, Supervision, Funding acquisition. **Gaige Chen:** Conceptualization, Writing - review & editing, Funding acquisition. **Qibin Wang:** Funding acquisition. **Rongbo Wang:** Visualization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table 8
Results of the proposed method and other transfer learning methods.

Method		Bearing 3	Bearing 4	Bearing 5	Bearing 6	Bearing 7	Average
TCNN	MAE	0.08 ± 0.01	**0.09 ± 0.01**	**0.13 ± 0.01**	**0.12 ± 0.01**	**0.09 ± 0.01**	**0.10 ± 0.02**
	RMSE	0.10 ± 0.01	**0.11 ± 0.01**	**0.15 ± 0.02**	**0.14 ± 0.01**	**0.11 ± 0.01**	**0.12 ± 0.02**
	R2	0.89 ± 0.03	**0.86 ± 0.01**	**0.72 ± 0.06**	**0.76 ± 0.02**	**0.85 ± 0.01**	**0.82 ± 0.07**
TMLP	MAE	**0.07 ± 0.01**	0.14 ± 0.02	0.18 ± 0.02	0.18 ± 0.03	0.14 ± 0.02	0.15 ± 0.05
	RMSE	**0.09 ± 0.01**	0.17 ± 0.02	0.22 ± 0.01	0.24 ± 0.02	0.18 ± 0.02	0.18 ± 0.05
	R2	**0.90 ± 0.02**	0.64 ± 0.08	0.43 ± 0.05	0.39 ± 0.11	0.62 ± 0.10	0.60 ± 0.20
DANN	MAE	0.08 ± 0.01	0.19 ± 0.01	0.22 ± 0.02	0.20 ± 0.01	0.18 ± 0.03	0.18 ± 0.06
	RMSE	0.10 ± 0.01	0.23 ± 0.02	0.26 ± 0.02	0.25 ± 0.02	0.22 ± 0.03	0.21 ± 0.06
	R2	0.87 ± 0.01	0.38 ± 0.11	0.18 ± 0.13	0.27 ± 0.13	0.40 ± 0.17	0.42 ± 0.27

Table 9
Results of RUL prediction at evaluated time and comparison.

Testing dataset	Evaluation time (10 s)	Actually RUL (10 s)	Predict RUL (10 s)	*Er*	*Er* [49]	*Er* [20]	*Er* [50]	*Er* [51]
Bearing3	1801	573	447.67	21.87	7.62	43.28	54.73	−1.04
Bearing4	1138	290	55.034	81.02	−157.71	67.55	38.69	−20.94
Bearing5	2301	161	306.98	−90.67	−72.57	−22.98	−99.4	−278.26
Bearing6	2301	146	149.06	−2.10	0.93	21.23	−120.07	19.18
Bearing7	1501	757	446.06	41.08	85.99	17.83	70.65	−7.13
$\overline{Er}$				10.24	−27.14	25.38	−11.08	−57.63
$\overline{\lvert Er \rvert}$				47.35	64.96	34.57	76.70	65.31

Acknowledgements

This research was supported by the Project of National Natural Science Foundation of China (Grant No. 51875432, 51905399, 51975446), Natural Science Foundation of Shaanxi Province (Grant No. 2019JQ-548) and China Postdoctoral Science Foundation (Grant No. 2019M663925XB). Thanks to the rolling dataset supplied by the Franche-Comté Electronics Mechanics Thermal Science and Optics-Science and Technologies (FEMSTO-ST) Institute. Comments and suggestions from the editor and reviewers are very much appreciated.

Declaration of Competing Interest

The authors declare that they have no conflict of interest.

References

[1] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, D. Siegel, Prognostics and health management design for rotary machinery systems—reviews, methodology and applications, Mech. Syst. Signal Pr. 42 (1-2) (2014) 314–334.
[2] O. Fink, E. Zio, U. Weidmann, A classification framework for predicting components' remaining ueful life based on discrete-event diagnostic data, IEEE T Reliab. 64 (3) (2015) 1049–1056.
[3] 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 PR. 104 (2018) 799–834.
[4] J. Coble, J.W. Hines, Applying the general path model to estimation of remaining useful life, Int. J. Prognost. Health Manage. 2 (1) (2011) 71–82.
[5] A. Cubillo, S. Perinpanayagam, M. Esperon-Miguez, A review of physics-based models in prognostics: Application to gears and bearings of rotating machinery, Adv Mech Eng. 8 (8) (2016) 21.
[6] Y.N. Qian, R.Q. Yan, R.X. Gao, A multi-time scale approach to remaining useful life prediction in rolling bearing, Mech Syst Signal Pr. 83 (2017) 549–567.
[7] Y. Lei, N. Li, S. Gontarz, J. Lin, S. Radkowski, J. Dybala, A model-based method for remaining useful life prediction of machinery, IEEE T Reliab. 65 (3) (2016) 1314–1326.
[8] Z. Meng, J. Li, N. Yin, Z. Pan, Remaining useful life prediction of rolling bearing using fractal theory, Measurement 156 (2020), 107572.
[9] Y.N. Qian, R.Q. Yan, S.J. Hu, Bearing degradation evaluation using recurrence quantification analysis and Kalman Filter, IEEE T Instrum Meas. 63 (11) (2014) 2599–2610.
[10] L. Xiao, X.H. Chen, X.H. Zhang, M. Liu, A novel approach for bearing remaining useful life estimation under neither failure nor suspension histories condition, J Intell Manuf. 28 (8) (2017) 1893–1914.
[11] S.J. Dong, T.H. Luo, Bearing degradation process prediction based on the PCA and optimized LS-SVM model, Measurement 46 (9) (2013) 3143–3152.
[12] W. Ahmad, S.A. Khan, J.M. Kim, A hybrid prognostics technique for rolling element bearings using adaptive predictive models, IEEE T Ind Electron. 65 (2) (2017) 1577–1584.
[13] K. Javed, R. Gouriveau, N. Zerhouni, P. Nectoux, Enabling health monitoring approach based on vibration data for accurate prognostics, IEEE T Ind Electron. 62 (1) (2014) 647–656.
[14] R.K. Singleton, E.G. Strangas, S. Aviyente, Extended Kalman filtering for remaining-useful-life estimation of bearings, IEEE T Ind Electron. 62 (3) (2014) 1781–1790.
[15] A. Soualhi, K. Medjaher, N. Zerhouni, Bearing health monitoring based on Hilbert-Huang transform, support vector machine, and regression, IEEE T Instrum. Meas. 64 (1) (2015) 52–62.
[16] W. Ahmad, S.A. Khan, M.M. Islam, J.M. Kim, A reliable technique for remaining useful life estimation of rolling element bearings using dynamic regression models, Reliab Eng. Syst. Safe. 184 (2019) 67–76.
[17] Z. Pan, Z. Meng, Z. Chen, W. Gao, Y. Shi, A two-stage method based on extreme learning machine for predicting the remaining useful life of rolling-element bearings, Mech Syst Signal Pr. 144 (2020), 106899.
[18] F. Xu, Z. Huang, F. Yang, D. Wang, K. Tsui, Constructing a health indicator for roller bearings by using a stacked auto-encoder with an exponential function to eliminate concussion, Appl. Soft. Comput. 89 (2020), 106119.
[19] L. Guo, Y. Lei, N. Li, S. Xing, Deep convolution feature learning for health indicator construction of bearings, in: 2017 prognostics and system health management conference (PHM-Harbin), IEEE (2017) 1-6.
[20] L. Guo, N. Li, F. Jia, Y. Lei, J. Lin, A recurrent neural network based health indicator for remaining useful life prediction of bearings, Neurocomputing. 240 (2017) 98–109.
[21] L. Ren, X. Cheng, X. Wang, J. Cui, L. Zhang, Multi-scale dense gate recurrent unit networks for bearing remaining useful life prediction, Future Gener. Comp Sy. 94 (2019) 601–609.
[22] X. Li, W. Zhang, Q. Ding, Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction, Reliab. Eng. Syst. Safe. 182 (2019) 208–218.
[23] B. Yang, R. Liu, E. Zio, Remaining useful life prediction based on a double-convolutional neural network architecture, IEEE T Ind Electron. 66 (12) (2019) 9521–9530.
[24] S. Xiang, Y. Qin, C. Zhu, Y. Wang, H. Chen, Long short-term memory neural network with weight amplification and its application into gear remaining useful life prediction, Eng. Appl. Artif. Intel. 91 (2020), 103587.
[25] P. Kundu, S. Chopra, B.K. Lad, Multiple failure behaviors identification and remaining useful life prediction of ball bearings, J Intell. Manuf. 30 (4) (2019) 1795–1807.
[26] Y.K. Son, Reliability prediction of engineering systems with competing failure modes due to component degradation, J. Mech Sci. Technol. 25 (7) (2011) 1717.
[27] S. Wang, Reliability model of mechanical components with dependent failure modes, Math Probl. Eng., 2013.
[28] Q. Zhang, C. Hua, G. Xu, A mixture Weibull proportional hazard model for mechanical system failure prediction utilising lifetime and monitoring data, Mech. Syst. Signal. PR. 43 (1–2) (2014) 103–112.
[29] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, C. Liu, A survey on deep transfer learning, in: International Conference on Artificial Neural Networks., 2018, 270–279.
[30] W.M. Kouw, M. Loog, A review of domain adaptation without target labels, IEEE T Pattern Anal. (2019).
[31] K. Xu, S. Li, J. Wang, Z. An, W. Qian, H. Ma, A novel convolutional transfer feature discrimination network for unbalanced fault diagnosis under variable rotational speeds, Meas. Sci. Technol. 30 (10) (2019), 105107.
[32] H. Zheng, R. Wang, Y. Yang, J. Yin, Y. Li, M. Xu, Cross-domain fault diagnosis using knowledge transfer strategy: a review,", IEEE Access 7 (2019) 129260–129290.
[33] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, X. Chen, Deep transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing, IEEE T Ind. Inform. 15 (4) (2019) 2416–2425.
[34] A. Zhang, H. Wang, S. Li, Y. Cui, Z. Liu, G. Yang, J. Hu, Transfer learning with deep recurrent neural networks for remaining useful life estimation, Appl. Sci.-Basel 8 (12) (2018) 2416.
[35] P.R.d.O. da Costa, A. Akçay, Y. Zhang, U. Kaymak, Remaining useful lifetime prediction via deep domain adaptation, Reliab. Eng. Syst. Safe. 195 (2020) 106682.
[36] W. Mao, J. He, M.J. Zuo, Predicting remaining useful life of rolling bearings based on deep feature representation and transfer learning, IEEE T Instrum. Meas. (2019).
[37] J. Zhu, N. Chen, C. Shen, A new data-driven transferable remaining useful life prediction approach for bearing under different working conditions, Mech. Syst. Signal. Pr. 139 (2020), 106602.
[38] B. Zhao, X. Zhang, H. Li, Z. Yang, Intelligent fault diagnosis of rolling bearings based on normalized CNN considering data imbalance and variable working conditions, Knowl.-Based Syst. (2020), 105971.
[39] A. L. Maas, A. Y. Hannun, A. Y. Ng, Rectifier nonlinearities improve neural network acoustic models, in: Proc. icml. 30 (1) (2013) 3.
[40] S.J. Pan, Q.A. Yang, A Survey on Transfer Learning, IEEE T Knowl. Data En. 22 (10) (2010) 1345–1359.
[41] A. Gretton, D. Sejdinovic, H. Strathmann, S. Balakrishnan, M. Pontil, K. Fukumizu, B. K. Sriperumbudur. Optimal kernel choice for large-scale two-sample tests, in: Advances in neural information processing systems. (2012) 1205-1213.
[42] M. Long, Y. Cao, J. Wang, M. I. Jordan, Learning transferable features with deep adaptation networks, in: international conference on machine learning. (2015) 97-105.
[43] K.M. Borgwardt, A. Gretton, M.J. Rasch, H.-P. Kriegel, B. Schölkopf, A.J. Smola, Integrating structured biological data by kernel maximum mean discrepancy, Bioinformatics 22 (14) (2006) e49–e57.
[44] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, T. Darrell, Deep domain confusion: maximizing for domain invariance. arXiv preprint arXiv:1412.3474 (2014).
[45] D. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, in: international conference on learning representations, 2014.
[46] 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 (PHM'12), IEEE (2012) 1-8.
[47] X. Li, W. Zhang, Q. Ding, J.-Q. Sun, Multi-layer domain adaptation method for rolling bearing fault diagnosis, Signal Process. 157 (2019) 180–197.
[48] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, Domain-adversarial training of neural networks, J. Mach. Learn. Res. 17 (1) (2016) 2096–12030.
[49] Y. Chen, G. Peng, Z. Zhu, S. Li, A novel deep learning method based on attention mechanism for bearing remaining useful life prediction, Appl. Soft Comput. 86 (2020), 105919.
[50] 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.
[51] S. Hong, Z. Zhou, E. Zio, K. Hong, Condition assessment for the performance degradation of bearing based on a combinatorial feature extraction method, Digit. Signal. Process. 27 (2014) 159–166.
[52] B. Wang, Y. Lei, N. Li, T. Yan, Deep separable convolutional network for remaining useful life prediction of machinery, Mech. Syst. Signal. Pr. 134 (2019), 106330.
[53] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, K. Keutzer, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. (2016) arXiv preprint arXiv:1602.07360.