# Intelligent Fault Diagnosis for Rotary Machinery Using Transferable Convolutional Neural Network

Zhuyun Chen, *Student Member, IEEE*, Konstantinos Gryllias, *Member, IEEE,* and Weihua Li, *Senior Member, IEEE*

***Abstract*—Deep neural networks present very competitive results in mechanical fault diagnosis. However, training deep models require high computing power while the performance of deep architectures in extracting discriminative features for decision making often suffers from the lack of sufficient training data. In this paper, a Transferable Convolutional Neural Network (TCNN) is proposed to improve the learning of target tasks. Firstly, a one-dimension CNN is constructed and pre-trained based on large source task datasets. Then a transfer learning strategy is adopted to train a deep model on target tasks by reusing the pre-trained network. Thus, the proposed method not only utilizes the learning power of deep network but also leverages the prior knowledge from the source task. Four case studies are considered and the effects of transfer layers and training sample size on classification effectiveness are investigated. Results show that the proposed method exhibits better performance compared with other algorithms.**

***Index Terms*—Fault diagnosis, Transfer learning, Convolutional Neural Network, Rotary machinery.**

## I. Introduction

THE effective fault detection and diagnosis techniques are of great importance in ensuring the safe and reliable operation of complex mechanical systems. Gears and rolling-element bearings, being vital components, are often the main sources of failure in rotating machines. In order to detect early, accurately and on-time the generation of faults, a number of diagnosis methods have been proposed including signal processing techniques and data driven methods.

The former techniques e.g. time-domain, frequency-domain and time-frequency analysis provide clear physical interpretations, but require high level diagnostic expertise and may fail when incipient or compound faults are developed in machinery operating under varying conditions [1-2]. The latter methods such as artificial neural network, support vector machines and manifold learning [3-5] may be more suitable for complex diagnosis problems, but their performance relies strongly on the quality of the hand-crafted features [6-8].

Since 2006, deep learning has emerged as a new branch of artificial intelligence. Deep learning methods, such as Deep Belief Network (DBN), Auto-Encoder (AE) and Convolutional Neural Network (CNN) present significant advantages in solving varieties of classification problems. Compared to other intelligent fault diagnosis methods, deep networks containing multiple hidden layers are able to learn useful discriminative features from the raw data itself. Furthermore, hierarchical distributed features learned layer-by-layer from large amounts of mechanical data turn out to be more effective and robust than manually selected or hand-crafted features [9]. Therefore, deep learning presents the potential to overcome the aforementioned deficiencies in the current intelligent diagnosis methods.

In recent years, deep learning methods have been also proposed for mechanical fault diagnosis and prediction [10-12]. Shao [13] used an AE method for electrical locomotive roller bearing fault diagnosis based on raw vibration signals. Sun [14] presented a sparse Deep Stacking Network (DSN) to model the sparsity of output labels achieving improved motor diagnosis accuracy. An improved Local Connect Network (LCN) with Normalized Sparse AE (NSAE) was constructed in [15] to extract dissimilar and meaningful features for bearing and gear diagnosis. Moreover, multiple sensor signals have been fused in [16-18] for improving the mechanical fault classification performance.

In addition to the fully-connected networks like DBN and AE, CNN has been also proposed for fault detection due to its good local representation and invariance. Sun [19] presented a novel convolutional discriminative feature learning method for induction motor fault diagnosis. Guo [20] adopted the 2DCNN hierarchical framework with an adaptive learning rate to recognize bearing fault categories and sizes. Liu [21] developed a dislocated time series CNN (DTS-CNN) to capture the relationship between different fault signals, presenting improved fault classification capability of an induction motor. Furthermore, in [22], a deep residual network with wavelet coefficients has been proposed for fault diagnosis of planetary gearboxes.

In addition, CNN with a 1D convolutional kernel (1DCNN) has been used in [23-24] for different diagnosis tasks, and

This work was supported in part by the National Natural Science Foundation of China under Grant 51875208 and National Key R&D Program of China under Grant 2018YFB1702402. (Corresponding author: Weihua Li).

Zhuyun Chen and Weihua Li are with the School of Mechanical & Automotive Engineering, South China University of Technology, Guangzhou 510640, China (e-mail: mezychen@gmail.com, whlee@scut.edu.cn).

Konstantinos Gryllias is with the Department of Mechanical Engineering, KU Leuven, Celestijnenlaan 300, Leuven, Belgium (e-mail: konstantinos.gryllias @kuleuven.be).

appears to be more suitable for 1D time series data. More specifically, a deep CNN with wide first-layer kernels (WDCNN) has been proposed in [25], achieving a 100% testing accuracy on the publically available CWRU bearing data set.

As mentioned above, different networks and architectures have been explored to conduct machinery fault detection and diagnosis, achieving good performance. However, the advantages of the evolution in deep learning techniques have not been fully exploited yet. Usually, there is the assumption that the training data and the test data have the same or similar distribution when using machine learning based methods for fault diagnosis. For a given fault diagnosis task, if there are sufficient training samples, an effective model for fault classification can be trained. However, in real industry applications, it is hard to collect sufficient fault samples for training models. Additionally, each time the model is applied to a new diagnosis task, it needs to be re-trained. The working conditions and the working environment might diverse significantly, leading to obvious differences between the training data (used for learning the model, usually obtained in experiments) and the real data (which should be monitored and diagnosed). As a result, the learned model might not be as effective at the testing phase as in the training phase.

Fortunately, transfer learning provides a way to deal with such problems. Massive data can be obtained in laboratory experiments by fault simulation, and thus the model can be trained sufficiently. Through the transfer of the knowledge learned from the experimental data (source domain), the learned model can be used for another similar task (target domain).

Moreover, deep learning diagnosis methods, such as CNN and DBN, may suffer a significant loss in performance when applied in a new diagnosis task having available only a small amount of data, even if the new task is similar to the original one. This problem usually occurs as the deep network easily overfits on the small training data and leads to poor performance on the testing data.

In an effort to deal with such diagnosis problems and motivated by the transfer learning, TCNN method is proposed to leverage source domain diagnosis knowledge, trying to save time and to improve the performance of processing new diagnosis issues in the target domain.

Transfer learning aims to transfer knowledge learned from related domains to help improve the learning performance of a target task with a small training data. It practically relaxes the assumption that the source and the target datasets must be in the same feature space and have the same distribution, which provides a useful scheme to reduce the need to re-collect training data of enough size [26].

Transfer learning, as an effective method, has achieved remarkable success based on deep model in a number of vision recognition tasks [27-29]. In the field of fault diagnosis, transfer learning with deep neural networks has been less explored due to the limitation of domain-specific dataset of sufficient size and common deep network model. Lu [30] proposed a deep neural network model combined with Maximum Mean Discrepancy (MMD) for fault diagnosis. Zhang [31] constructed an artificial neural network and used the first-layer weights trained in one operating condition for target task. Sun [32] proposed a Sparse Auto-encoder (SAE), which was trained by historical failure data and then transferred to a new tool for remaining useful life (RUL) prediction. Shao [33] utilized directly a pre-trained CNN model for bearing fault diagnosis. However, expertise knowledge is required to generate time-frequency images and the features obtained from the natural image task are different from the diagnosis information, which could lead to a reduction of performance.

In this paper a transfer learning approach for fault detection in rotating machinery is proposed, inspired by the success of transfer learning in image processing [27], web page categorization [28] and medical disease recognition [29]. Specifically, a transfer learning framework based on TCNN is developed to explore a strong intelligent fault diagnosis scheme, which provides to the model an ability to learn general representations. Those representations can enable a wide variety of tasks and can adapt quickly to new fault diagnosis issues with less human intervention. In the proposed method, raw time-series signals, collected from source domain datasets, are directly used to train the designed CNN without any hand-crafted feature extraction. Then TCNN obtained from the pre-trained model can be adopted to transfer source domain knowledge to new target domain tasks improving the diagnosis performance.

The contributions of the proposed TCNN can be summarized as follows:

(1) The proposed method can be used to deal with fault diagnosis problems lacking training data, for equipment working under different conditions, and even more at different facilities (under the term that the tasks are similar). Moreover, it is promising to be applied in practical industry applications.

(2) The proposed method gives to the target model reasonable parameter initializations by a pre-training strategy. Therefore, it provides a potential tool to train a deep network-based diagnosis system fast and efficiently with less overfitting risk. It can improve the model performance as well as save time.

(3) From the perspective of model transfer, the proposed scheme can be used not only for CNN model, but also can be extended to other deep learning algorithms like DBN, SAE, LSTM, etc.

The rest of the paper is organized as follows. In Section II, the CNN basic theoretical background is given. In Section III, the proposed TCNN intelligent diagnosis method is introduced. The details of the experimental datasets, including gearbox datasets and bearing datasets are described in Section IV. In Section V, the results of the application of the proposed method on the measurements are presented and analyzed. Finally, some key conclusions are made in Section VI.

## II. Convolutional Neural Network

### A. *Convolutional Neural Network (CNN)*

The CNN, usually consisted of an input layer, multiple hidden layers and an output layer, is well known for its shared-weight architecture and some degree of translation-invariant characteristics. It can extract local features at lower layers and then combine them into more abstract features at higher layers. The basic CNN architecture, including a convolutional layer, a pooling layer and a fully-connected layer is further introduced.

*1) Convolutional layer*

The convolutional layer usually is made of a set of learnable kernels and one trainable bias per feature map. The kernel size corresponds to the length of the convolution window and the kernel depth or kernel filter corresponds to the number of the feature map outputs. The output of neurons, which are connected to the input volume can be obtained by computing the dot product between their weights and the small region. Considering a *L*-layers CNN architecture, the *l*-layer convolved feature maps can be expressed as:

$$y_{l,j}^{conv} = \sum\nolimits_i^k w_{i,j}^l * \mathrm{y}_{l-1,i}^{pool} + b_j^l \tag{1}$$

$$y_{l,j}^{\mathrm{Re}LU} = f(y_{l,j}^{conv}) = max[0, y_{l-1,j}^{conv}] \tag{2}$$

where $y_{l,j}^{conv}$ is the output of the *l*-th layer, $w^l$ is the convolutional kernel of the *l*-th layer, *k* is the number of kernels, and $b^l$ is the bias. $f(\cdot)$ is the activation function which transforms the input to the output map in order to increase the nonlinear property. An activation function ReLU (Rectified Linear Unit) expressed as $max(0,\mathrm{x})$ is usually used instead of a sigmoid. ReLU is proved to work far better in most of the classification tasks for its capacity to accelerate the convergence and alleviate the vanishing gradient problem [34].

*2) Pooling layer*

After the convolutional layer, an additive pooling layer is applied to each feature map from the previous layer. Pooling is a form of non-linear down-sampling, which can reduce each map size and the network parameters, achieving spatial invariance. Two common pooling strategy choices are the average and the max pooling. Max pooling is generally favorable as it can lead to faster convergence, select superior invariant features and improve generalization. The max pooling function is given by:

$$\mathrm{y}_{l,j}^{pool} = \underset{M\times N}{max}\left(p(s_1,s_2)\,\mathrm{y}_{l-1,j}^{\mathrm{Re}LU}\right) \tag{3}$$

where $p(s_1,s_2)$ is the window function applied to the input patch, $s_1$ and $s_2$ correspond to the size of the window, which can be of arbitrary size and overlapping and *M*×*N* corresponds to the size of the *l*-layer feature map output.

*3) Fully-connected layer*

The feature map outputs of the last max pooling layer are reshaped into a vector. The neurons in fully-connected layers have fully connection to all activations in the previous layer. Finally, a Softmax classifier is attached to discriminate the different classes. Similar to the fully-connected network learning procedure, CNN can be effectively trained by minimizing the supervised loss function:

$$L(\theta) = \sum_{j=1}^{c} L(\mathrm{y}_j, f_j(\mathrm{x});\theta) \tag{4}$$

where *c* corresponds to the number of classes and y*ⱼ* is the 1-of-*c* code of the training labels. $f_j(\mathrm{x})$ is the network output of the *j*-th label of the sample. The cross entropy is used instead of the squared-error loss function, improving the update speed of model parameters. Then Eq. (4) can be rewritten as:

$$\begin{aligned} L(\theta) &= -\frac{1}{m}\sum_{i=1}^{m}\sum_{j}^{c} 1\{\mathrm{y}_i = j\}\log(\mathrm{p}(\mathrm{y}_i = j \mid \mathrm{x}_i^{L-1};\theta)) \\ &= -\frac{1}{m}\sum_{i=1}^{m}\sum_{j}^{c} 1\{\mathrm{y}_i = j\}\log\frac{e^{\theta_j^T \mathbf{x}_i^{L-1}}}{\sum_{l=1}^{c} e^{\theta_l^T \mathbf{x}_i^{L-1}}} \end{aligned} \tag{5}$$

$$\nabla_{w_j^o} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} \mathbf{x}_i (1\{\mathrm{y}_i = j\} - \mathrm{p}(\mathrm{y}_i = j \mid \mathbf{x}_i;\theta)) \tag{6}$$

where *m* is the number of training samples and the output layer parameters $w_j^o$ can be updated using Eq. (6). Other layer parameters can be successively adjusted according to the rule of back propagation algorithm and stochastic gradient descent.

## III. Proposed Transfer Scheme

CNN has demonstrated powerful modeling capabilities in extracting local features and in obtaining hierarchically discriminative representations at different layers. 1D CNN appears as an efficient tool for processing vibration signals for mechanical fault diagnosis. In this paper a novel TCNN architecture is proposed in order to improve the learning of efficient discriminative features from raw data. TCNN is a modified version of WDCNN [25], where dropout techniques [35], kernel numbers and fully-connected layers are added.

As presented in Fig. 1, the 1D raw vibration signals are firstly input into the first convolutional layer to achieve signal local features. Then Batch Normalization (BN in Fig. 1) is implemented to reduce the distribution of each layer's input by performing normalization for each training mini-batch. Max Pooling is used to down-sample the input and to create position invariance over larger local regions. Dropout is added as a regularization constraint to reduce node interactions and to learn robust features. Then a non-linear map is conducted layer-by-layer by forward propagation. The last Softmax is used to convert the probability output of the categories. The parameters of each layer are updated by using a back propagation algorithm and minimizing the cross entropy error.

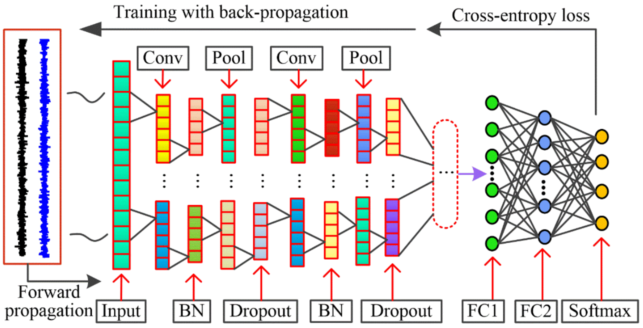


Fig. 1. The architecture of TCNN

In our experiment, Taking the diagnosis accuracy and computational cost into consideration, TCNN with different network layers is explored. Grid-search method in a relatively small ranges of layers is conducted to find a sub-optimal architecture on the source domain dataset. A threshold of testing accuracy (99%) is set as stop criterion, and the optimal architecture of TCNN is listed in Table I.

TABLE I
PARAMETERS OF TCNN

| Block Name | No. | Layer Type | Kernel Size | Kernel Number | Stride | Padding |
|---|---|---|---|---|---|---|
| **B1** | 1 | Convolution | 64×1 | 32 | 16 | Yes |
| | 2 | BN | - | - | - | - |
| | 3 | Max Pooling | 2×1 | 32 | 2 | - |
| | 4 | Dropout | - | - | - | No |
| **B2** | 5 | Convolution | 3×1 | 32 | 1 | Yes |
| | 6 | BN | - | - | - | - |
| | 7 | Max Pooling | 2×1 | 32 | 2 | - |
| | 8 | Dropout | - | - | - | No |
| **B3** | 9 | Convolution | 3×1 | 64 | 1 | Yes |
| | 10 | BN | - | - | - | - |
| | 11 | Max Pooling | 2×1 | 64 | 2 | - |
| | 12 | Dropout | - | - | - | No |
| **B4** | 13 | Convolution | 3×1 | 64 | 1 | Yes |
| | 14 | BN | - | - | - | - |
| | 15 | Max Pooling | 2×1 | 64 | 2 | - |
| | 16 | Dropout | - | - | - | No |
| **B5** | 17 | Convolution | 3×1 | 64 | 1 | Yes |
| | 18 | BN | - | - | - | - |
| | 19 | Max Pooling | 2×1 | 64 | 2 | - |
| | 20 | Dropout | - | - | - | No |
| **B6** | 21 | Fully Connected | 1000 | - | - | - |
| | 22 | BN | - | - | - | - |
| | 23 | Dropout | - | - | - | - |
| | 24 | Fully Connected | 100 | - | - | - |
| | 25 | BN | - | - | - | |
| | 26 | Dropout | - | - | - | - |
| | 27 | Softmax | 20 | - | - | - |

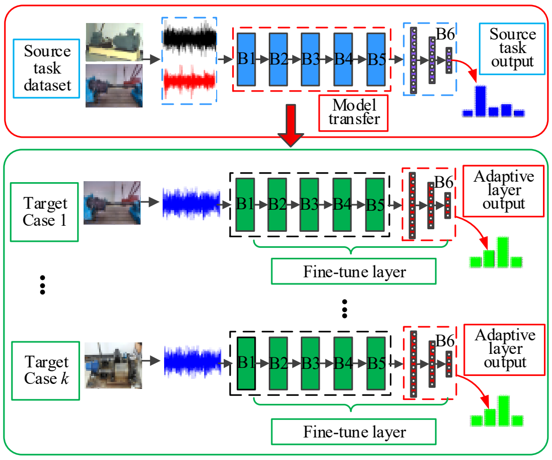


Fig. 2. Transfer learning for fault diagnosis using TCNN

The layers and the parameters of TCNN, which are subdivided into six building blocks (B1-B6), are presented in Table I. Each building block contains specific layers. The first five blocks consist of Convolutional, Max Pooling, BN and Dropout layers. Following the last dropout layer, the output layer, represented by B6, is designed. It includes the fully-connected layers FC1, FC2 and a Softmax classifier which have embedded multiply non-linear layers of BN and dropout. In the case of one-dimensional (1-D) vibration signal, the first convolutional layer extracts the features from the raw input, where a large convolution kernel (specifically, 64*1 with stride 16) is applied. The number of the filter is set to 32. After that, the rest of the four convolutional layers have 32, 64, 64 and 64 kernels, respectively. Each of them have small kernel sizes (specifically, 3*1 with stride 2). The large kernel is adopted with the purpose to suppress high frequency noise while small kernels in the following layers help to enhance the feature learning capability and improve the network performance. In addition, zero-padding is used to keep the same size before and after the convolution operation. For the pooling layer, the number of kernels in each building block is the same as that in the convolutional layers (e.g. 32 kernels in the first pooling layer), but, compared with the previous output, the feature map size is halved by performing down sampling with a pooling size of 2*1 and stride 2. Finally, the network ends with two fully-connected layers (specifically, 1000 and 100 nodes, respectively) and the Softmax for classification.

As CNN is a supervised forward network, fully labeled source domain datasets are used. In this scheme, as it will be analytically presented later, large source task datasets are firstly collected Then the CNN is pre-trained based on the source domain datasets to obtain a Transferable CNN (TCNN). During the transfer stage, as the labels of the source domain dataset are usually not equal to those of the target task, the Softmax output of B6 in TCNN model is replaced with a new one, corresponding to the categories of the different target tasks, as shown in Fig. 2. Finally, the parameters of the different layers are fine-tuned on the target data with a small number of training samples.

The transfer scheme of TCNN for fault diagnosis involves seven steps:

1) Source domain datasets and target domain datasets are collected from different experimental platforms.

2) The source domain and target datasets, are respectively divided into training data and testing data. All feature vectors are normalized into the range [-1, 1].

3) CNN model is constructed with multiple convolutional, max pooling, BN, dropout and fully-connected layers. TCNN model is constructed by pre-training CNN model with the source domain samples.

4) The Softmax output of B6 in TCNN is replaced by an adaptive one, which corresponds to the categories of the target case. The training instances of target domain datasets are used for fine-tuning of the B6 of TCNN, while the other layer parameters are frozen.

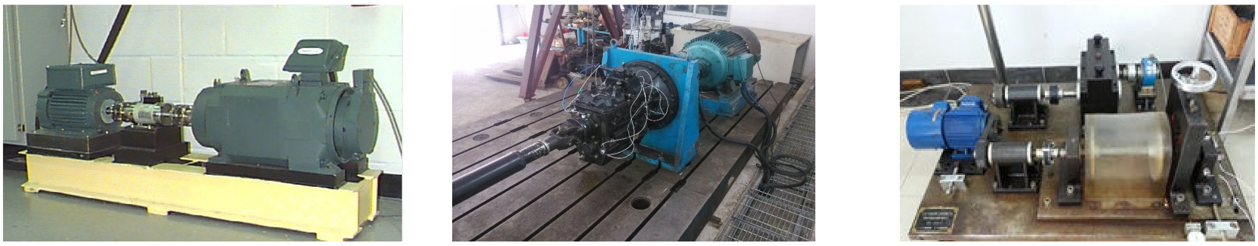
5) TCNN is fine-tuned by fixing the last two blocks B5 and B6, while the rest of the layer's parameters are frozen.

6) The learning process in step 5 is repeated until the classification rate reaches the given value or the given iterations are completed, which means that the optimized transferable layers are determined.

7) Testing samples of target cases are fed into the optimized TCNN and the discriminative classes are obtained.

## IV. DESCRIPTION OF EXPERIMENTAL DATASETS

Transfer learning strategies depend on various factors. Among them, the two most important ones are its similarity to



(a) CWRU bearing test (b) Gearbox test (c) Bearing test

Fig. 3. The experiment test rig

TABLE II
DESCRIPTIONS OF THE SOURCE DOMAIN BEARING DATASET

| **Dataset** | **Speed** | **Fault Types** | **Fault diameters** | **Number** | **Class** |
|---|---|---|---|---|---|
| | **(rpm)** | | **(inch)** | **of Samples** | **Label** |
| $\mathbf{D}_1^s$ | 1772 & 1750 & 1730 | Health | 0 | 800 & 800 & 800 | 1 |
| | 1772 & 1750 & 1730 | BF | 0.007 | 800 & 800 & 800 | 2 |
| | 1772 & 1750 & 1730 | IF | 0.007 | 800 & 800 & 800 | 3 |
| | 1772 & 1750 & 1730 | OF | 0.007 | 800 & 800 & 800 | 4 |
| | 1772 & 1750 & 1730 | BF | 0.014 | 800 & 800 & 800 | 5 |
| | 1772 & 1750 & 1730 | IF | 0.014 | 800 & 800 & 800 | 6 |
| | 1772 & 1750 & 1730 | OF | 0.014 | 800 & 800 & 800 | 7 |
| | 1772 & 1750 & 1730 | BF | 0.021 | 800 & 800 & 800 | 8 |
| | 1772 & 1750 & 1730 | IF | 0.021 | 800 & 800 & 800 | 9 |
| | 1772 & 1750 & 1730 | OF | 0.021 | 800 & 800 & 800 | 10 |

TABLE III
DESCRIPTIONS OF THE SOURCE DOMAIN GEARBOX DATASET

| **Dataset** | **Speed** | **Fault Types** | | **Number** | **Class** |
|---|---|---|---|---|---|
| | **(rpm)** | **The Fifth shift gear** | **Output bearing** | **of Samples** | **Label** |
| $\mathbf{D}_2^s$ | 500 & 750 | Normal | Normal | 900 & 900 | 1 |
| | 500 & 750 | Minor chipped tooth | Normal | 900 & 900 | 2 |
| | 500 & 750 | Half chipped tooth | Normal | 900 & 900 | 3 |
| | 500 & 750 | Missing tooth | Normal | 900 & 900 | 4 |
| | 500 & 750 | Normal | 0.2 mm inner race fault | 900 & 900 | 5 |
| | 500 & 750 | Minor chipped tooth | 0.2 mm inner race fault | 900 & 900 | 6 |
| | 500 & 750 | Half chipped tooth | 0.2 mm inner race fault | 900 & 900 | 7 |
| | 500 & 750 | Missing tooth | 0.2 mm inner race fault | 900 & 900 | 8 |
| | 500 & 750 | Half chipped tooth | 2 mm inner race fault | 900 & 900 | 9 |
| | 500 & 750 | Missing tooth | 2 mm inner race fault | 900 & 900 | 10 |

TABLE IV
DESCRIPTION OF TARGET DOMAIN GEARBOX DATASET

| **Dataset $\mathbf{D}_1^t$** | **Speed** | **Fault Types** | | **Number** | **Class** |
|---|---|---|---|---|---|
| | **(rpm)** | **The Fifth shift gear** | **Output bearing** | **of Samples** | **Label** |
| **C1** | 1250 | Normal | Normal | 300 | 1 |
| | 1250 | Minor chipped tooth | Normal | 300 | 2 |
| | 1250 | Half chipped tooth | Normal | 300 | 3 |
| | 1250 | Missing tooth | Normal | 300 | 4 |
| | 1250 | Normal | 0.2 mm inner race fault | 300 | 5 |
| | 1250 | Minor chipped tooth | 0.2 mm inner race fault | 300 | 6 |
| | 1250 | Half chipped tooth | 0.2 mm inner race fault | 300 | 7 |
| | 1250 | Missing tooth | 0.2 mm inner race fault | 300 | 8 |
| | 1250 | Half chipped tooth | 2 mm inner race fault | 300 | 9 |
| | 1250 | Missing tooth | 2 mm inner race fault | 300 | 10 |
| **C2** | 1000 & 1250 | Normal | Normal | 300 & 300 | 1 |
| | 1000 & 1250 | Minor chipped tooth | Normal | 300 & 300 | 2 |
| | 1000 & 1250 | Half chipped tooth | Normal | 300 & 300 | 3 |
| | 1000 & 1250 | Missing tooth | Normal | 300 & 300 | 4 |
| | 1000 & 1250 | Normal | 0.2 mm inner race fault | 300 & 300 | 5 |
| | 1000 & 1250 | Minor chipped tooth | 0.2 mm inner race fault | 300 & 300 | 6 |
| | 1000 & 1250 | Half chipped tooth | 0.2 mm inner race fault | 300 & 300 | 7 |
| | 1000 & 1250 | Missing tooth | 0.2 mm inner race fault | 300 & 300 | 8 |
| | 1000 & 1250 | Half chipped tooth | 2 mm inner race fault | 300 & 300 | 9 |
| | 1000 & 1250 | Missing tooth | 2 mm inner race fault | 300 & 300 | 10 |

TABLE V
DESCRIPTION OF TARGET DOMAIN BEARING DATASET

| Dataset $D_2^t$ | Speed | Fault Types | | Number of Samples | Class |
|---|---|---|---|---|---|
| | (rpm) | Inner race | Outer race | | Label |
| C3 | 1100 | Normal | Normal | 300 | 1 |
| | 1100 | Normal | 0.5 mm fault | 300 | 2 |
| | 1100 | Normal | 2 mm fault | 300 | 3 |
| | 1100 | 0.5 mm fault | Normal | 300 | 4 |
| | 1100 | 2 mm fault | Normal | 300 | 5 |
| C4 | 800 & 1100 | Normal | Normal | 300 & 300 | 1 |
| | 800 & 1100 | Normal | 0.5 mm fault | 300 & 300 | 2 |
| | 800 & 1100 | Normal | 2 mm fault | 300 & 300 | 3 |
| | 800 & 1100 | 0.5 mm fault | Normal | 300 & 300 | 4 |
| | 800 & 1100 | 2 mm fault | Normal | 300 & 300 | 5 |

the original dataset and the size of the new dataset [27]. To comprehensively evaluate the proposed transfer scheme, source domain datasets are firstly collected. Then target datasets are obtained from different test platforms to simulate the relative similarity with source datasets in the data distribution. The influence of the training sample size on the classification performance is further explored in four target cases.

### A. *Source domain datasets ($D^s$)*

The source domain datasets include a bearing dataset ($D_1^s$) and a gearbox dataset ($D_2^s$), which are used to train the proposed transferable CNN (TCNN).

The bearing dataset ($D_1^s$) is obtained from the Case Western Reserve University (CWRU) Bearing Data Center [36]. The test rig is shown in Fig.3(a) and consists of a 2 HP Reliance Electric motor, a torque transducer and a dynamometer. During the test, the vibration signals were collected under three different working conditions ('1', '2' and '3' HP at speeds ranging from 1772 rpm to 1730 rpm), and the sampling rate is 12 kHz. Three kinds of defects (inner race defect, outer race defect and ball defect) were artificially introduced into the deep-groove ball bearings (Type: 6205-2RS JEM SKF) in different severity levels (0.007, 0.014, and 0.021 inch in diameter, and 0.011 inch in depth for each case). Ten bearings are involved in the experiment, one healthy bearing and nine faulty ones with different defects or different fault severity levels, and each of them is taken as one class. The length of each sample is set as 2000 data points. For every bearing, there are 800 samples for each running condition, thus in total 2400 samples for three working conditions. Among them, 1500 samples were used for training, and the rest for testing. The details are listed in Table II.

The gearbox dataset ($D_2^s$) was collected on a five-speed automobile transmission. The transmission test rig is shown in Fig. 3(b). The main components of the experimental apparatus include a driven motor, a torque transducer, a gearbox, and a loading motor. The gearbox has five forward gear pairs and one backward gear pair. In the experiment, the fifth speed gear is used to conduct the fault test, and the test was performed under two running speeds (500 rpm and 750 rpm), while the sampling rate was 24 kHz. The output shaft load torque was 50 N•m. In order to simulate different fault types, gear faults (minor chipped tooth, half chipped tooth and missing tooth) and bearing faults (inner race defect) with different fault diameters (0.2 mm and 2 mm) have been introduced by electro-discharge machining. 900 samples have been collected under every working condition, and thus 1800 samples for every class. Among them, 1000 samples are used for training and the remaining for testing. Considering different combinations of bearing faults and gear faults, there are in total ten classes of different faults. More details can be found in Table III.

### B. *Target domain datasets ($D^t$)*

The target domain datasets containing a gearbox dataset ($D_1^t$) and a bearing dataset ($D_2^t$), are used to evaluate the proposed method.

The gearbox dataset ($D_1^t$) has been acquired on the same platform of the source domain gearbox dataset with the same fault configuration but operating under different speeds (1000 rpm and 1250 rpm). 300 samples have been collected under each running condition. This dataset is used to evaluate the generalization of the proposed TCNN under different working conditions. More details are listed in Table IV.

The bearing dataset ($D_2^t$) has been obtained from the experiments conducted on a test bench, shown in Fig. 3(c). The shaft is driven by an induction motor through a belt connection. The shaft output end is supported by the testing bearing.

A healthy bearing and two sets of faulty bearings (inner race defects and outer race defects) have been used in the experiments. For each fault, two different defect severity levels (0.5 mm and 2 mm in diameter) have been introduced and totally 5 health conditions have been generated to simulate different fault types and defect severities. An accelerometer has been attached vertically to the bearing housing to acquire the vibration signals under running speeds of 800 rpm and 1100 rpm, respectively. The sampling frequency is 12 kHz and 300 samples have been collected under each running condition. This dataset is used to evaluate the generalization performance of TCNN on different equipment. More details are listed in Table V.

Additionally, in order to assess the performance of the proposed method, the datasets ($D^t$) are grouped into four cases (C1-C4) according to the running operating conditions. The dataset ($D_1^t$) is divided into C1 and C2, containing vibration data captured under the constant rotating speed of 1250 rpm and the combined speeds of 1000 rpm and 1250 rpm, respectively. Dataset ($D_2^t$) was divided into C3 and C4 with constant rotating speed of 1100 rpm and combined speeds of 800 rpm and 1100 rpm respectively. For each case, 200 samples are used for training and the rest 100 samples for testing. For the gear and the bearing dataset, usually, the data combined under variable speed conditions are more difficult to be classified due to the change of the characteristic fault frequencies. Therefore, C2 and C4 were exactly designed to emulate the fault condition with varying speeds, introducing an extra complexity compared to C1 and C3. The details of target cases are illustrated in Table IV and Table V.

## V. Experimental Results Analysis

In this section, the proposed method is applied and a comparison with four state-of-the-art methods is performed based on the source domain datasets with twenty categories of gear and bearing faults where large labeled training data are available. After proving its superiority, transfer learning is applied on the target datasets. The feature learning ability and the transfer effectiveness are analyzed. Finally, a number of comprehensive comparisons with other algorithms are conducted to evaluate the effectiveness of the modified TCNN.

### *A. Evaluation on source domain datasets*

In order to examine the superiority of the modified TCNN, a comparison of TCNN with CNN-Wen [12], 2DCNN [20], 1DCNN [23] and WDCNN [25], all trained on the source domain training set and evaluated on the corresponding testing set, is made. For the CNN-Wen and the 2DCNN, all the samples are converted into 40×50 images from the original 2000 data points, which are suitable for network input. Adam optimization algorithm is employed to update the parameters due to its computational effectiveness. The max training epoch is set as 200 with a batch size of 100. All computations have been performed on a computer with Intel Xeon E5-262v3 (2.4GHz), 64GB RAM, 4 TITAN X graphics cards and Google TensorFlow framework [37].

The classification process is repeated 10 times and the final results are averaged throughout all the experiments. The average accuracy and the standard deviation (STD) of the training and the testing stage are presented in Table VI.

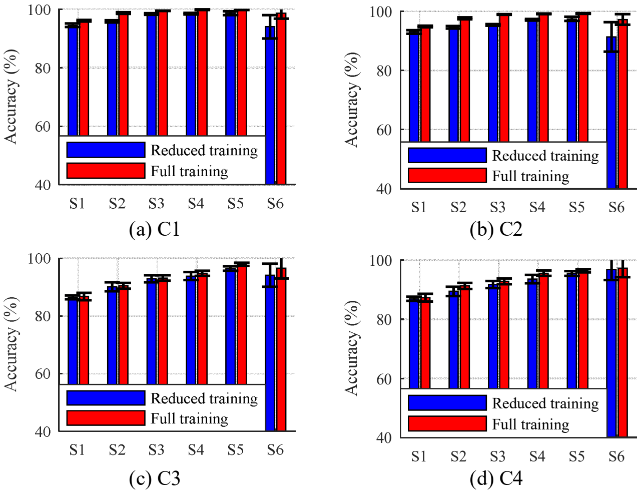
TABLE VI
RESULT COMPARISON WITH DIFFERENT METHODS

| Methods | Training accuracy (%) | Testing accuracy (%) |
|---|---|---|
| **CNN-Wen** | 99.99±0.04 | 91.26±2.63 |
| **2DCNN** | 99.99±0.02 | 87.15±1.25 |
| **1DCNN** | 98.58±1.85 | 94.32±2.92 |
| **WDCNN** | 99.85±0.30 | 96.54±1.61 |
| **TCNN** | 99.99±0.01 | 99.03±0.21 |

From the results, it can be seen that all the other four methods obtain high training accuracies, but the testing accuracies are lower than the proposed method. 1DCNN and WDCNN obtain a testing accuracy of 94.32% and 96.54% with a standard deviation of 2.92% and 1.61%, performing better than CNN-Wen and 2DCNN. It is possible that a CNN with one-dimensional (1D) architecture is more effective than the 2DCNN in capturing discriminative features from raw vibration signal inputs. Which contribute to high classification performance. In contrast, TCNN presents very competitive results in testing accuracy and standard deviation compared with the other four methods. The proposed method achieves a testing accuracy of 99.03% with a standard deviation of 0.21%, showing the superiority of the modified TCNN.

### *B. Model-based transfer learning in TCNN*

In order to study the influence of the training sample size on the classification rate, initially a reduced dataset and the full training dataset are used for fine-tuning of TCNN in the training stage. In the cases of gear data (C1 & C2), the reduced dataset consists of 30% of the training samples, whereas in the case of bearing data (C3 & C4) of 50% of the training. For TCNN fine-tuning, the Stochastic Gradient Descent (SGD) is utilized to adjust the network weights with a batch size equal to 50. Grid search is used to find good learning rate and momentum. Finally, the learning rate is chosen as 0.01 and the momentum is set as 0.97 with 100 epochs, so that a minor change is allowed for each parameter update during the fine-tuning procedure. The classification accuracies with respect to the different fine-tuning layers for all four cases are displayed in Fig. 4. The x-axis (S1-S6) represents the number of the fine-tuned layers, while the y-axis gives the corresponding classification results.



(a) C1 (b) C2

(c) C3 (d) C4

Fig. 4. The relationship between accuracy and fine-tuned layers

It can be noticed that by just fine-tuning the last output layers B6, keeping the parameters of the rest layers (B1-B5) unadjusted, achieves a relative high accuracy in C1 and C2 (corresponding to the results shown in axis S1) but not enough high accuracy in C3 and C4. Intuitively it can be explained that the learnt representations from the high layers of TCNN could be more specific to the task it's trained for. As C3 and C4 are less similar to the source domain dataset, many of the high layer features cannot be directly reused. Therefore, only fine-tuning the output layer B6 results in poor performance.

Furthermore, there is a progressively increase in the classification performance when more layers are fine-tuned. The accuracy reaches at least 95% in C1 and C2 and 90% in C3 and C4 by just fine-tuning the last two blocks using the total of training samples. When the fine-tuned layers increase to four blocks (corresponding to the result of S4), the improvements for the testing accuracies are not so obvious.

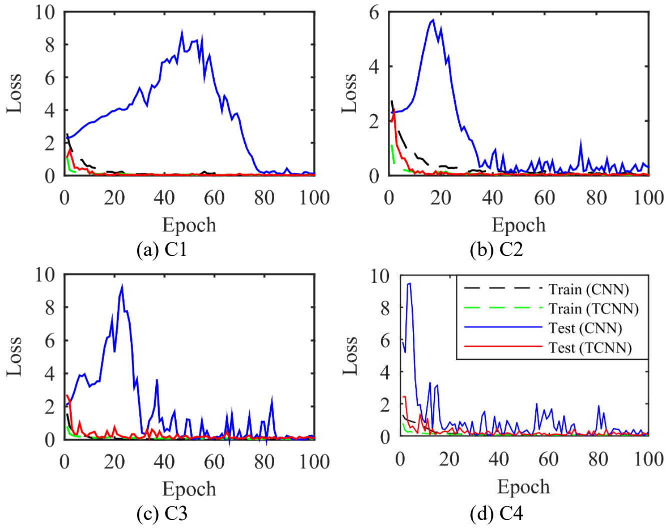
Additionally, for the reduced training samples, a significantly decrease in the classification performance occurs when all the layers are readjusted. It is noted that fine-tuning all layer parameters does not perform as well as with the bottom layer weights fixed. This is mostly due to the fact that in B1, there is a rapid increase in the number of the convolutional kernel size and filters which bring in a high number of parameters to be fine-tuned. When few training instances are used to fine-tune TCNN, although TCNN have enough capacity to process much information, the information contained in the small training data is not enough to train all the neurons of the hidden layers. As a result, overfitting problems occur during the training process, which reduce the ability to predict input data and lead to poor performance in testing instances.

### C. *Comparison with CNN (without transfer)*

In order to verify the reliability and extensibility, an evaluation is conducted to investigate if TCNN model can improve the classification accuracy in the target cases compared with the baseline CNN. TCNN and CNN share the same network structure and TCNN is actually the pre-trained CNN on the source domain datasets. The difference between the two models is that, the weights of the networks are different. For target domain tasks, the CNN was initialized randomly with random weights, whereas TCNN has already been pre-trained to have relatively ‘good’ weights. The comparison is used to evaluate the effectiveness of transfer learning. For a fair comparison, the Adam algorithm is used to train the CNN which presents better classification accuracy and convergence speed than the SGD algorithm in the experiments. As TCNN can nearly achieve the best result in ‘S5’, the parameters of the last five blocks are fine-tuned in the target cases for comparison.

#### 1) *Loss and accuracy comparison*

The learning procedures of TCNN and baseline CNN without transfer are compared. Both models are trained on four cases with a batch size of 50 and 100 epochs. The training and testing losses obtained during the training stage of TCNN and CNN on C1, C2, C3 and C4 are respectively presented in Fig. 5.



(a) C1

(b) C2

(c) C3

(d) C4

Fig. 5. Loss curve of CNN and TCNN

In the case of CNN, the training loss is smooth in all cases but the testing loss presents a large fluctuation even after 80 iterations. On the contrary, TCNN achieves better performance on all four cases. The loss curve converges more quickly. After 20 epochs of training, the training loss and the testing loss of TCNN are gradually close to a fixed value and remain stable. By contrast, the testing loss of CNN diverges until it reaches a certain number of iterations. This is mostly because the pre-training procedure of TCNN establishes an optimization starting point of the fine-tuning procedure inside a region of parameter space, which helps to achieve fast and accurate convergence to a good generalizing minimum. On the other hand, CNN is easy to get stuck in poor local solutions in the training procedure with random initialized weights.

TABLE VII
COMPARISON RESULTS WITH REDUCED TRAINING SAMPLES (%)

| Cases | Testing accuracy±STD (%) | | | Training time (s) | | |
|---|---|---|---|---|---|---|
| | CNN | TCNN | TCNN -20 | CNN | TCNN | TCNN -20 |
| C1 | 63.6±4.9 | 98.7±0.6 | 94.9±3.0 | 120 | 29 | 12 |
| C2 | 67.5±3.1 | 96.4±0.8 | 96.6±1.6 | 236 | 50 | 17 |
| C3 | 90.6±9.2 | 97.2±0.1 | 95.1±1.7 | 33 | 24 | 11 |
| C4 | 90.7±7.2 | 95.5±0.8 | 93.2±4.6 | 52 | 42 | 16 |

TABLE VIII
COMPARISON RESULTS WITH FULL TRAINING SAMPLES (%)

| Cases | Testing accuracy±STD (%) | | | Training time (s) | | |
|---|---|---|---|---|---|---|
| | CNN | TCNN | TCNN -20 | CNN | TCNN | TCNN -20 |
| C1 | 94.9±3.2 | 99.9±0.1 | 99.7±0.3 | 45 | 38 | 15 |
| C2 | 93.2±4.6 | 99.3±0.1 | 98.6±0.6 | 98 | 73 | 22 |
| C3 | 93.6±4.8 | 97.9±0.5 | 96.2±1.2 | 53 | 41 | 14 |
| C4 | 93.8±6.3 | 96.5±0.5 | 96.2±3.5 | 81 | 68 | 20 |

Moreover, the classification accuracy and the computational loads are evaluated and compared between TCNN and baseline CNN. TCNN with just 20 epochs (TCNN-20) is also used for comparison. Table VII and Table VIII list the results obtained using the reduced and the full training samples in the four cases. Compared to TCNN, the classification accuracies of CNN presented in Table VI are respectively only 63.60% and 67.49% using C1 and C2 with reduced training samples and they reach 94.94% and 93.22% when full training samples are used, as presented in Table VIII. In C3 and C4, CNN achieved relatively high accuracies but presented high standard deviations. On the other hand, TCNN presents some advantages over CNN in accuracy and computational cost in all cases.

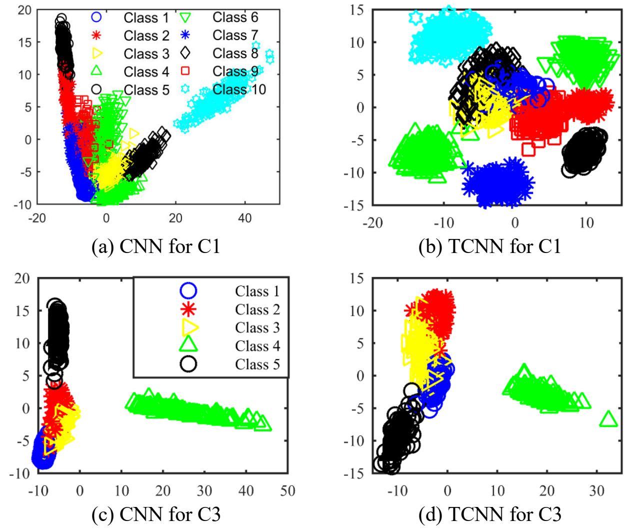
It can also be observed that as the size of training samples increased, the accuracy of TCNN and CNN increased. More especially CNN gives a major improvement. It is possible that CNN with a large number of parameters trained on the reduced training data suffers from the overfitting problem. By adding more training samples, CNN is prone to be resistant to the overfitting and achieves further improvements but still the performance is lower than TCNN’s.

Through all results, TCNN even with only 20 epochs still significantly outperforms CNN in terms of classification accuracy (mean and standard deviation) and computational cost.

The training time of TCNN-20 is at least 3 times reduced in all cases compared to that of CNN. It should be highlighted that the training time of CNN at C1 and C2 cases (Table VII) is much longer compared to other cases. This is due to the fact that the testing loss is still divergent after 100 epochs, so a tiny batch size of 10 has been used that significantly accelerates the convergence but incurs the cost of extended training time.

### *2) High-level feature visualization*

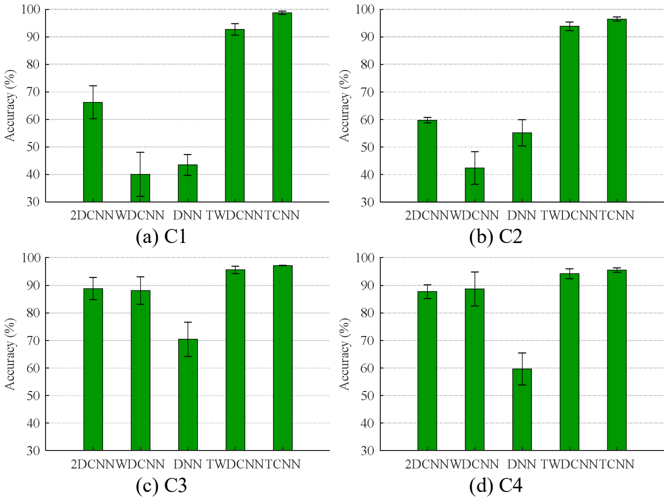
In order to gain some insight into what the network has learned in high-level layers, the activation outputs of layer 20 are used. It would be expected that the features closer to the output layer are more linearly separable. Those features are obtained from TCNN and CNN with testing samples of C1 and C3, respectively. In addition, the Principal Component Analysis (PCA) technique is used for visualization by reducing the data dimension from 192 to 2. The clustering results are presented in Fig. 6. It can be seen that the different categories are heavily overlapped in the CNN case. Especially for C1, most of the points are mixed with each other and only points in class ten (10) are well distinguished. Therefore, it can be expected that the classification performance will be not good enough. This conclusion is consistent with the results obtained in Table VIII. In the cases of C1 and C3, the high-level features of different classes obtained in TCNN are more discriminative in a lower-dimensional space than in CNN. The features learned from TCNN are relatively well clustered and most of the categories are separable presenting less overlap. The results show that TCNN is better to lean meaningful discriminative features from lower layers to high-level layers than CNN, which may help to yield high classification performance in testing stage.



(a) CNN for C1 (b) TCNN for C1

(c) CNN for C3 (d) TCNN for C3

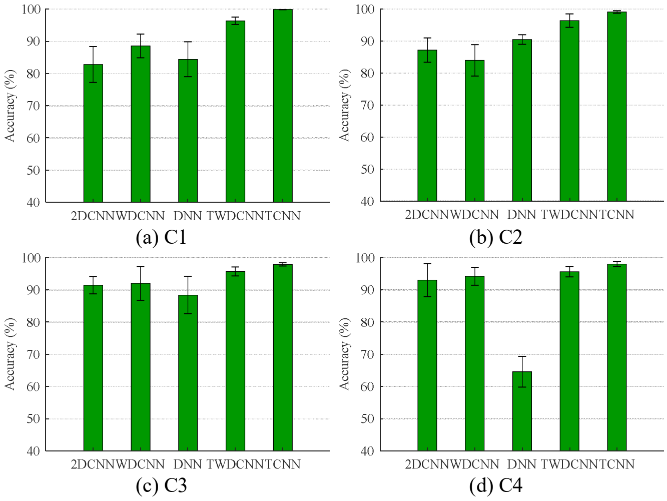
Fig. 6. The testing samples visualization



(a) C1 (b) C2

(c) C3 (d) C4

Fig. 7. Accuracy comparison of five methods on the reduced training samples



(a) C1 (b) C2

(c) C3 (d) C4

Fig. 8. Accuracy comparison of five methods on the full training samples

TABLE IX
COMPARISON RESULTS WITH REDUCED TRAINING SAMPLES (%)

| **Cases** | **Testing accuracy±STD (%)** | | | | |
|---|---|---|---|---|---|
| | **2DCNN** | **WDCNN** | **DNN** | **TWDCNN** | **TCNN** |
| **C1** | 66.2±6.0 | 40.0±8.0 | 43.4±3.8 | 92.7±2.1 | 98.7±0.6 |
| **C2** | 59.7±1.0 | 42.4±5.9 | 55.2±4.8 | 93.8±4.8 | 96.4±4.8 |
| **C3** | 90.7±7.2 | 95.5±0.8 | 93.2±4.6 | 93.2±4.6 | 93.2±4.6 |
| **C4** | 87.7±2.5 | 88.7±6.1 | 59.7±5.8 | 94.2±1.8 | 95.5±0.8 |

TABLE X
COMPARISON RESULTS WITH FULL TRAINING SAMPLES (%)

| **Cases** | **Testing accuracy±STD (%)** | | | | |
|---|---|---|---|---|---|
| | **2DCNN** | **WDCNN** | **DNN** | **TWDCNN** | **TCNN** |
| **C1** | 82.8±5.6 | 88.6±3.7 | 84.4±5.4 | 96.4±1.1 | 99.9±0.1 |
| **C2** | 87.2±3.8 | 84.1±4.9 | 90.6±1.5 | 96.4±2.1 | 99.3±0.1 |
| **C3** | 91.5±2.7 | 92.0±5.2 | 88.4±5.8 | 95.7±1.4 | 97.9±0.5 |
| **C4** | 93.0±5.1 | 94.2±2.8 | 64.6±4.8 | 95.6±1.6 | 96.5±0.5 |

## *D. Comparison with other methods*

Furthermore, a comparison of the proposed method with four different methods is carried out for the reduced and the full training dataset: (1) 2DCNN. All the samples are resized to 40×50 pixels from the original 2000 data points. The architecture used in [20] is considered here. Additionally, a further improvement is made for a fair comparison by adopting the fully-connected layer architecture of TCNN, which helps to learn better discriminate features. (2) WDCNN is employed which achieves state-of-art results in the public CWRU bearing data set in [25]. (3) DNN [10]. The raw vibration signals are firstly transformed into Fourier spectra and then are fed into the DNN for faulty decision. (4) TWDCNN. The proposed transfer learning strategy is used. WDCNN is pre-trained and transferred (Named TWDCNN) for classification in four target cases as it was presented for TCNN before.

The identification accuracy of the proposed method and the other algorithms are shown in Fig. 7 and Fig. 8. The test accuracy of each method is listed in Table IX and Table X. In Fig.7 and Table IX, compared to 2DCNN and DNN, it can be observed that WDCNN achieves similar performance in C3 and C4, obtaining accuracy of 95.5% and 88.7%, respectively. However, in C1 and C2, which contain more fault types, they get worse performance with an accuracy of 40% and 42.5%. It is possible that WDCNN has a larger number of trainable parameters which could lead to more overfitting in the case of the reduced dataset. By adding more training samples as shown in Fig. 8 and Table X, WDCNN and the other traditional deep learning networks all improve the classification performances.

In C3 and C4, though five classes exist, the highest accuracy yielded in TCNN is 95.5% compared with 98.7% obtained in C1 and C2 with ten categories. This can be contributed to its similarity to the source datasets. It is in accordance with [27] that the effectiveness of feature transfer will gradually decline as the source dataset and target dataset become less similar.

TWDCNN, sharing the same network architecture with WDCNN, makes a large improvement using the proposed transfer learning methods. On the other hand, TCNN achieves higher accuracy on all testing cases compared to other methods The improved performance of TWDCNN and TCNN is more obvious in the case of small number of training samples. This is due to the fact that DNN, WDCNN and 2D CNN are based on deep architectures, which need massive amount of samples to enhance the classification, while TWDCNN and TCNN could utilize the prior knowledge from source domain datasets that reduce the dependence on the number of training samples.

## VI. Conclusion

In this paper, a transfer learning framework based on TCNN has been proposed for fault diagnosis of mechanical systems. The key idea of this method is to exploit the knowledge gained from fault diagnosis issues and different machines (historical data) to improve the performance of target task problems. TCNN is a modified version of WDCNN, where dropout techniques, kernel numbers and fully-connected layers are added in order to improve the learning of efficient discriminative features from raw data. Different diagnosis cases as well as different datasets have been used in order to test and validate the performance of the proposed method, presenting good stability and robustness and achieving better results compared to state-of-the-art architectures. The proposed method can be used not only for complex diagnosis cases but also for other data-driven tasks, including condition monitoring, anomaly detection, bearing life prediction, prognostics etc. As a next step, the authors will extend the proposed methodology towards unsupervised or semi-supervised settings.

## References

[1] Y. Hao, L. Song, M. Wang, L.Cui, and H. Wang, "Underdetermined Source Separation of Bearing Faults Based on Optimized Intrinsic Characteristic-Scale Decomposition and Local Non-Negative Matrix Factorization," IEEE Access, vol. 7, pp.11427-11435, Jan. 2019.

[2] H. Wang, P. Wang, L. Song, B. Ren, L. Cui, L. "A Novel Feature Enhancement Method based on Improved Constraint Model of Online Dictionary Learning," IEEE Access, 2019. DOI: 10.1109/ACCESS.2019.2895776

[3] Y Liao, L Zhang, W Li. "Regrouping particle swarm optimization based variable neural network for gearbox fault diagnosis," Journal of Intelligent & Fuzzy Systems, vol. 34, no. 6, pp. 3671-3680, Jun. 2018.

[4] R. H. C. Palácios, I. N. da Silva, A. Goedtel, W. F. Godoy, T. D. Lopes. Diagnosis of stator faults severity in induction motors using two intelligent approaches. IEEE Transactions on Industrial Informatics, vol. 13, no.4, pp.1681-1691, Aug. 2017.

[5] L. Song, H. Wang, and P. Chen. "Vibration-Based Intelligent Fault Diagnosis for Roller Bearings in Low-Speed Rotating Machinery," IEEE Transactions on Instrumentation and measurement, vol. 67, no. 8, pp. 1887-1899, Mar. 2018.

[6] L. Cui, T. Yao, Y. Zhang, "Application of pattern recognition in gear faults based on the matching pursuit of a characteristic waveform," Measurement. Vol. 104, pp. 212-222, Mar. 2017.

[7] W. Li, S. Zhang, S. Rakheja. "Feature denoising and nearest–farthest distance preserving projection for machine fault diagnosis," IEEE Transactions on Industrial Informatics, vol. 12, no.1, pp. 393-404, Feb. 2016.

[8] K. Zhu, X. Lin, K. Li and L. Jiang, "Compressive sensing and sparse decomposition in precision machining process monitoring: From theory to applications," Mechatronics,vol. 31, pp.3-15, Oct. 2015.

[9] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang and R. X. Gao, "Deep Learning and Its Applications to Machine Health Monitoring: A Survey," arXiv preprint arXiv:1612.07640, 2016.

[10] F. Jia, Y. Lei, J. Lin, X. Zhou and N. Lu, "Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data," Mech. Syst. Signal Process., vol.72-73, pp. 303-315, May. 2016.

[11] B. Zhang, S. Zhang, W. Li, "Bearing performance degradation assessment using long short-term memory recurrent network," Computers in industry, vol.106, pp. 14-29, April. 2019.

[12] L. Wen, X. Li, L. Gao, Y. Zhang, Y, "A New Convolutional Neural Network-Based Data-Driven Fault Diagnosis Method.," IEEE Transactions on Industrial Electronics, vol. 65, no. 7, pp.5990-5998, Nov. 2017.

[13] H. Shao, H. Jiang, H. Zhao and F. Wang, "A novel deep autoencoder feature learning method for rotating machinery fault diagnosis," Mech. Syst. Signal Process., vol. 95, pp. 187-204, Oct. 2017.

[14] C. Sun, M. Ma, Z. Zhao, X. Chen. "Sparse Deep Stacking Network for Fault Diagnosis of Motor," IEEE Transactions on Industrial Informatics, vol. 14, no. 7, July. 2018.

[15] F. Jia, Y. Lei, L. Guo, J. Lin and S. Xing, "A neural network constructed by deep learning technique and its application to intelligent fault diagnosis of machines," Neurocomputing, vol. 272, pp. 619-628, Jan. 2018.

[16] M. Ma, C. Sun, X. Chen, "Deep Coupling Autoencoder for Fault Diagnosis With Multimodal Sensory Data," IEEE Transactions on Industrial Informatics, vol. 14, no. 3, pp. 1137-1145, Jan. 2018.

[17] Z. Chen, W. Li, "Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network," IEEE Transactions on Instrumentation and Measurement, vol. 66, no.7, pp. 1693-1702, Jul.2017

[18] H. Wang, S. Li, L. Song and Li. Cui, L. "A novel convolutional neural network based fault recognition method via image fusion of multi-vibration-signals," Computers in Industry, vol. 105, pp. 182-190, Feb. 2019

[19] .W. Sun, R. Zhao, R. Yan, S. Shao and X. Chen, "Convolutional Discriminative Feature Learning for Induction Motor Fault Diagnosis," IEEE Transactions on Industrial Informatics, vol. 13, no. 3, pp. 1350-1359, Jun. 2017.

[20] X. Guo, L. Chen, and C. Shen, "Hierarchical adaptive deep convolution neural network and its application to bearing fault diagnosis," Measurement, vol. 93, pp. 490–502, Nov. 2016.

[21] R. Liu, G. Meng, B. Yang, C. Sun, X. Chen, X, "Dislocated time series convolutional neural architecture: An intelligent fault diagnosis approach for electric machine," IEEE Transactions on Industrial Informatics, vol. 13, no.3, pp. 1310-1320, Jun. 2017.

[22] M. Zhao, M. Kang, B. Tang and M. Pecht, "Deep Residual Networks with Dynamically Weighted Wavelet Coefficients for Fault Diagnosis of

Planetary Gearboxes," IEEE Trans. Ind. Electron., vol. 65, no. 5, pp. 4290-4300, May. 2018.

[23] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-time motor fault detection by 1-d convolutional neural networks," IEEE Trans. Ind. Electron., vol. 63, no. 11, pp. 7067–7075, Nov. 2016.

[24] R. Huang, Y. Liao, S. Zhang and W. Li, "Deep Decoupling Convolutional Neural Network for Intelligent Compound Fault Diagnosis," IEEE Access, Vol.7, pp.1848-1858, 2019.

[25] W. Zhang, G. Peng, C. Li, Y. Chen and Z. Zhang, "A New Deep Learning Model for Fault Diagnosis with Good Anti-Noise and Domain Adaptation Ability on Raw Vibration Signals," Sensors, vol. 17, no. 2, Feb. 2017.

[26] S. J. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on knowledge and data engineering, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[27] D. Soekhoe, P. van der Putten and A. Plaat, "On the impact of data set size in transfer learning using deep neural networks," In International Symposium on Intelligent Data Analysis, Springer, Cham, pp. 50-60, Oct. 2016.

[28] D. López-Sánchez, A. G. Arrieta and J. M. Corchado, "Deep neural networks and transfer learning applied to multimedia web mining," In International Symposium on Distributed Computing and Artificial Intelligence, Springer, Cham pp. 124-131, Jun. 2017.

[29] D. S. Kermany, M. Goldbaum, W. Cai, C. C. Valentim, H. Liang, S. L. Baxter, ... & J. Dong, "Identifying medical diagnoses and treatable diseases by image-based deep learning," Cell, vol.172, no.5, pp. 1122-1131, Feb. 2018.

[30] W. Lu, B. Liang, Y. Cheng, D. Meng, J. Yang, and T. Zhang, "Deep model based domain adaptation for fault diagnosis," IEEE Trans. Ind. Electron., vol. 64, no. 3, pp. 2296-2305, Nov. 2016.

[31] R. Zhang, H. Tao, L. Wu and Y. Guan, "Transfer Learning With Neural Networks for Bearing Fault Diagnosis in Changing Working Conditions," IEEE Access, vol. 5, pp. 14347-14357, Jun. 2017.

[32] C. Sun, M. Ma, Z. Zhao, S. Tian, R. Yan, and X. Chen, "Deep Transfer Learning Based on Sparse Auto-encoder for Remaining Useful Life Prediction of Tool in Manufacturing," IEEE Transactions on Industrial Informatics, DOI: 10.1109/TII.2018.2881543, Nov. 2018.

[33] S. Shao, S. McAleer, R. Yan, R, P. Baldi, "Highly-Accurate Machine Fault Diagnosis Using Deep Transfer Learning," IEEE Transactions on Industrial Informatics. DOI: 10.1109/TII.2018.2864759, 2018.

[34] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," In Advances in neural information processing systems. pp. 1097-1105, 2012.

[35] N. Srivastava, G. Hinton, A. Krizhevsky, I. and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," The Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929-1958, Jan. 2012.

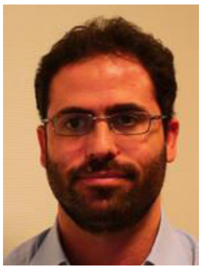
[36] Case Western Reserve University Bearing Data Center Website ⟨http://csegroups.case.edu/bearingdatacenter/home⟩.

[37] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, …, M. Kudlur, "TensorFlow: A System for Large-Scale Machine Learning". In OSDI, vol. 16, pp. 265-283, Nov. 2016.

[38] J. Yosinski, J. Clune, Y. Bengio and H. Lipson, "How transferable are features in deep neural networks?" In Advances in neural information processing systems pp. 3320-3328, Dec. 2014

**Zhuyun Chen** received his B.S. degree in mechanical design, manufacturing, and automation from Nanjing Agricultural University, Nanjing, China, in 2013. From Dec 2017 to Dec 2018, he was a joint PhD at KU Leuven, Belgium.

He is currently pursuing his Ph.D. degree in mechanical engineering with the South China University of Technology, Guangzhou, China. His current research interests include dynamic signal processing and deep learning methods for mechanical fault diagnosis and prognostics.



**Konstantinos Gryllias (M'15)** holds a 5 years engineering diploma degree and a PhD degree in Mechanical Engineering from National Technical University of Athens, Greece.

Since October 2014, he holds an assistant professor position on "Vibro-acoustics of machines and transportation systems" at the Department of Mechanical Engineering of KU Leuven, Belgium. He is also the manager of the University Core Lab Dynamics in Mechanical & Mechatronic Systems DMMS-M of Flanders Make, Belgium. His research interests lie in the fields of condition monitoring, signal processing, prognostics and health management of mech. &mechatronic systems.

**Weihua Li (M'12–SM'18)** received his Ph.D. degree in mechanical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2003.

He is currently a Professor with the School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou, China. His current research interests include nonlinear time series analysis, dynamic signal processing, and machine learning based methods for health monitoring and prognosis of complex dynamical systems.