sensors

MDPI

Article

Unsupervised Domain Adaptive 1D-CNN for Fault Diagnosis of Bearing

Xiaorui Shao [1] and Chang-Soo Kim [2,*]

[1] Industrial Science Technology Research Center, Pukyong National University, Busan 608737, Korea; shaoxiaorui@pukyong.ac.kr
[2] Information Systems, Pukyong National University, Busan 608737, Korea
* Correspondence: cskim@pknu.ac.kr

Abstract: Fault diagnosis (FD) plays a vital role in building a smart factory regarding system reliability improvement and cost reduction. Recent deep learning-based methods have been applied for FD and have obtained excellent performance. However, most of them require sufficient historical labeled data to train the model which is difficult and sometimes not available. Moreover, the big size model increases the difficulties for real-time FD. Therefore, this article proposed a domain adaptive and lightweight framework for FD based on a one-dimension convolutional neural network (1D-CNN). Particularly, 1D-CNN is designed with a structure of autoencoder to extract the rich, robust hidden features with less noise from source and target data. The extracted features are processed by correlation alignment (CORAL) to minimize domain shifts. Thus, the proposed method could learn robust and domain-invariance features from raw signals without any historical labeled target domain data for FD. We designed, trained, and tested the proposed method on CRWU bearing data sets. The sufficient comparative analysis confirmed its effectiveness for FD.

Keywords: fault diagnosis; vibration signal; 1D-CNN; domain adaption; autoencoder

check for updates

Citation: Shao, X.; Kim, C.-S. Unsupervised Domain Adaptive 1D-CNN for Fault Diagnosis of Bearing. *Sensors* **2022**, *22*, 4156. https://doi.org/10.3390/s22114156

Academic Editor: Hossam A. Gabbar

Received: 14 May 2022
Accepted: 28 May 2022
Published: 30 May 2022

1. Introduction

Bearing is one of the critical elements for the rotating machine in the production industries, the healthy status of which significantly influences production efficiency [1,2], system safety, and reliability [3]. Therefore, designing an advanced approach for fault diagnosis (FD) of bearing is critical, aiming to identify and detect the failure of bearing before it happens.

The current methods for FD mainly include model-based, signal-based, and learning-based methods [4,5]. Among them, model-based and signal-based methods require a priori knowledge of the operating systems, which increases the difficulty and cost for FD. On the contrary, learning-based methods do not require prior system knowledge and have extracted more attention.

Learning-based methods for FD can be divided into traditional machine learning and deep learning methods. Traditional learning methods consist of support vector machine (SVM) [6], decision tree (DT) [7], and random forest (RF) [8]. Primarily, SVM aims at finding one segmentation hyperplane with various kernels, which can correctly classify the data and maximize the spacing among various faults. For example, Diego, et al. [9] proposed an automatic scheme based on SVM for FD of bearing, where only normal vibration signals are used. Lu and Li [10] utilized principal component analysis (PCA) and SVM for FD of bearing. DT is a tree-based method that finds the critical feature to construct the rule for FD. For example, Tan, et al. [11] utilized DT to detect the fault of an induction motor, and Asman, et al. [12] applied it to find the causes in a 275 kV transmission line network. RF integrates multiple DTs to select the key features and make the decision for FD [13,14]. It usually performs better than DT. However, there are some limitations that occur in the traditional machine learning methods. For instance, the performance of SVM highly depends on kernel selection [15]. DT and RF are sensitive to noisy data and still require selecting valuable features by hand [16].

Compared to the traditional machine learning methods, deep learning-based methods extract rich hidden features from raw vibration signals without any handcraft feature selection operations and have been widely adopted. Specifically, convolution neural network (CNN) [17], one of the deep neural networks, is the mainstream choice for FD due to its excellent feature extraction capacity. For example, Chen, et al. [18] applied two-dimensional (2-D) CNN for FD of the gearbox, where vibration signals are processed into statistical measures and frequency domain representations by fast Fourier transform (FFT) as CNN's input. Wen, et al. [19] also adopted 2-D CNN based on LeNet to extract the hidden patterns from 2-D images for FD. Shao, et al. [20] applied a pre-trained 2-D CNN structure VGG-16 [21] for FD, in which raw signals are processed into the 2-D image using continuous wavelet transformation (CWT). However, 2-D CNN requires more time to train the model and give the results of FD due to the usage of extra preprocessing operations. Its performance is affected by different processing methods. In contrast, 1D-CNN can overcome those drawbacks and has been applied for FD with raw signals. For example, Zhang proposed [22] a novel 1D-CNN with wide convolution operation (WDCNN) at the first layer for FD of bearing. The experimental analysis confirmed its good anti-noise and domain adaption capacities when combining adaptive batch normalization (AdaBN) [23]. Jiang, et al. [24] proposed a multi-scale CNN (MSCNN) for FD of the wind turbine gearbox, in which the raw signal is split into multi-course grained features with the average operation, and then multi-scale 1D-CNN is used to learn differ-scale hidden features. The experimental results suggested that three-scale CNN is better for FD, considering the diagnostic performance and required time. Motivated by MSCNN, Shao, et al. [25] proposed a multi-scale feature fusion CNN (MSFFCNN) for 1-D time series classification which can be used for FD. Moreover, Kim, et al. [5] applied multi-domain features, including statistical, raw signals and discrete wavelet transformed (DWT) frequency domain features as the input of 1D-CNN (MDCNN) for FD of bearing. Liu, et al. [26] combined 1D-CNN and multi-task learning for FD of wheelset bearing, in which the signals' loadings and speed identification were measured to make full use of information to improve FD performance.

Although the above 1D-CNN-based methods have achieved good performance within the tolerable processing time, they still have limitations. Firstly, they require sufficient labeled historical data to train the model. However, it is difficult or even not available to collect all kinds of faulty data in practice due to the signals' randomness and irregularity. Secondly, they assume that both training (source domain) and testing (target domain) data have the same distribution while they are quite different, and as a result, the performance is unsatisfactory and still can be improved.

Luckily, deep transfer learning (DTL) can learn and transfer the knowledge from the source domain into the target domain with limited training samples that have been applied to FD. The DTL-based method for FD is relatively new. It mainly consists of supervised pre-trained DTL and unsupervised DTL (also called domain adaption). Supervised pre-trained DTL trains the model on the source domain data while fine-tuning the model on the target domain data. For example, Shao, et al. [20] trained the VGG-16 on the ImageNet set and fine-tuned the model for each faulty data set for FD. Brusa, et al. [27] pre-trained the YAMNet on the sound and music data set and fine-tuned the model on the bearing data set for FD. Yao, et al. [28] pre-trained the deep model on the source domain faulty data and froze some parts to fine-tune one CNN on the target domain data for FD of nuclear power plants. Unlike supervised pre-trained DTL, unsupervised DTL for FD does not require any labeled target domain data. It aims to reduce the domain shift between two domains by calculating their similarities. For example, Lu, et al. [29] utilized a fully connected neural network to extract the hidden features from two domains. Moreover, the single-kernel maximum mean discrepancy (MMD) was used to minimize the domain shift for FD. Wen, et al. [30] combined a three-layer sparse autoencoder and single-kernel MMD for FD of bearing. Zhu, et al. [31] utilized 2-D CNN to extract hidden features and multi-kernel MMD at the last two layers to reduce the domain shifts for FD. However, MMD has serval shortcomings, such as sensitivity to kernel selection and data samples, and low scalability for large applications [32]. To overcome MMD's shortcomings, Wang, et al. [32] utilized correlation alignment (CORAL) [33,34] to calculate the distribution discrepancy between each layer of stacked denoising autoencoder for FD of a power plant thermal system. However, calculating each layer's CORAL is time-consuming, and we argue that CNN can extract more robust features than a fully connected neural network.

As we described above, few references have utilized 1D-CNN to extract hidden features from raw signals considering the domain shift for FD. This manuscript proposed a novel lightweight framework for FD of bearing. A 1D-CNN autoencoder was designed to extract rich features with less noise from raw vibration signals. Furthermore, the domain shifts were minimized by calculating the CORAL distance between extracted source and target features. Sufficient experiments confirmed that the proposed method could detect faults in a complex environment accurately.

The main contribution of this manuscript is summarized as follows:

- One new lightweight domain adaption 1D-CNN autoencoder was proposed. It could automatically learn rich, robust hidden features from raw vibration signals for FD with less time.
- The CORAL was combined with the proposed 1D-CNN autoencoder to minimize the domain shift between source and target domains. Therefore, the proposed method could learn rich, robust, and domain-invariant hidden features to accurately and timely detect the cross-domain faults without labeled target domain samples. Sufficient comparative experiments confirmed its effectiveness.
- Each component's effectiveness was analyzed through one ablation study.
- The effectiveness of the reconstruction ratio is discussed in depth.

The rest of this manuscript is arranged as follows. Section 2 gives some pre-knowledge used in the proposed method, including the problem statement, CNN, and CORAL distance. Section 3 describes the proposed method for FD in detail. The experimental verification is carried out in Section 4. Section 5 discusses the proposed method for FD. At last, we conclude this manuscript in Section 6.

2. Preliminaries

2.1. Problem Statement

Assume that we collected some data in the source domain, denoted as $\mathcal{D}^{\int} = \left\{ \left(x_i^s, y_i^s \right) \right\}_{i=1}^{N}$. Where $x_i^s = \left\{ t_{i,j}^s, t_{i,j+1}^s, \ldots, t_{i,n+j-1}^s \right\} \in X^s \in R^n$ is the vibration signal, which consists of n data points from time step j to $n + j - 1$. Each signal x_i^s corresponds to one faulty type $y_i^s = \{1, 2, 3, \ldots, C\} \in Y^s$. Same for the source domain data, we collected target domain data $\mathcal{D}^t = \left\{ \left(x_i^t \right) \right\}_{i=1}^{N}$ without labels, where $x_i^t = \left\{ t_{i,j}^t, t_{i,j+1}^t, \ldots, t_{i,n+j-1}^t \right\} \in X^t \in R^n$. We assumed that both source and target domain data have the same feature space $\mathcal{X}$ and label space $\mathcal{Y}$ but different distributions, i.e., $X^s = X^t = \mathcal{X}$, $Y^s = Y^t = \mathcal{Y}$ and $P(X^s) \neq P(X^t)$. Our goal was to build one transferable model $f(\cdot)$, which can learn domain-invariant features by using labeled source domain samples and unlabeled target samples that enable: $\mathcal{X} \to \mathcal{Y}$. In this manuscript, we first utilized one 1D-CNN autoencoder to learn hidden features with less noise from source and target data; then, the extracted features were processed by CORAL to minimize the distribution discrepancy. Therefore, the proposed method could learn common domain-invariant features to detect the fault on unseen target domain sets accurately.

2.2. CNN

CNN is a classical feedback neural network which has been widely used in the area of image classification [17,35], medical heathy management and control [36,37], and time series processing [38–40] due to its robust feature extraction capacity. The conventional

CNN mainly is used to process 2-D images, while this manuscript is for 1-D signals. Giving one signal x, the 1D-CNN learns the hidden patterns through three key components. Firstly, convolution operation with various kernels is calculated to extract features in multiple views. The convoluted features have parameter sharing and local connection characteristics, which ensures that the neural network extracts rich hidden features with fewer parameters than a fully connected neural network. Then, one activation function activates the convoluted features to strengthen the non-linear expression. Those two processes are shown in (1), where the k^{th} feature map h_k is calculated by using the convolution operation $*$ between signal x and k^{th} kernel filter f_k with a bias b_k; σ is one activation function such as the sigmoid and rectified linear unit (ReLU) [41]. Thirdly, the pooling operation reduces the feature map's size to speed up the network. The most used pooling operation is the maximum polling operation max, which calculates the maximum value of the feature map within a certain range w, as shown in (2). The final features h are obtained after n-time convolution and maximum pooling operations (called convolution block $ConvBlock$), as shown in (3). The final features h are used for predicting the faults. More details about CNN can be found in [17].

$$h_k = \sigma(x * f_k + b_k) \tag{1}$$

$$h_{pk} = max(h_k, w) \tag{2}$$

$$h = ConvBlock_n(x) = max(\sigma(x * f_k + b_k))_n \tag{3}$$

2.3. CORAL Distance

CORAL [33] calculates the source and target features' second-order statistics (covariances) to align their distributions. It is a parameter-free way to measure the distance between source and target domains and does not require any labeled target domain samples. Compared to MMD, it is not sensitive to kernel selection and data samples, which is suitable for vibration signal data. The calculation of CORAL is denoted in (4), where C_s and C_t are second-order statistics of source and target features, respectively; d is the dimension of source and target features; and $\|\cdot\|_F^2$ denotes the square matrix of Frobenius norm. The covariance matrix is calculated as (5) and (6), where n_s and n_t are the numbers of source and target features, respectively; h_s and h_t denote, respectively, model extracted hidden features for source and target domains; and $\mathbf{1}$ is a column vector with all elements equal to 1.

$$\mathcal{L}_{coral} = \frac{1}{4d^2}\|C_s - C_t\|_F^2 \tag{4}$$

$$C_s = \frac{1}{n_s - 1}\left(h_s^T h_s - \frac{1}{n_s}\left(\mathbf{1}^T h_s\right)^T \left(\mathbf{1}^T h_s\right)\right) \tag{5}$$

$$C_d = \frac{1}{n_t - 1}\left(h_t^T h_t - \frac{1}{n_t}\left(\mathbf{1}^T h_t\right)^T \left(\mathbf{1}^T h_t\right)\right) \tag{6}$$

3. The Proposed Method

The proposed method for FD is shown in Figure 1. It mainly consists of five steps: input construction, feature extraction, reconstruction of the raw signal, domain shift reduction, and fault diagnosis. More details about each part are described in the following subsection.

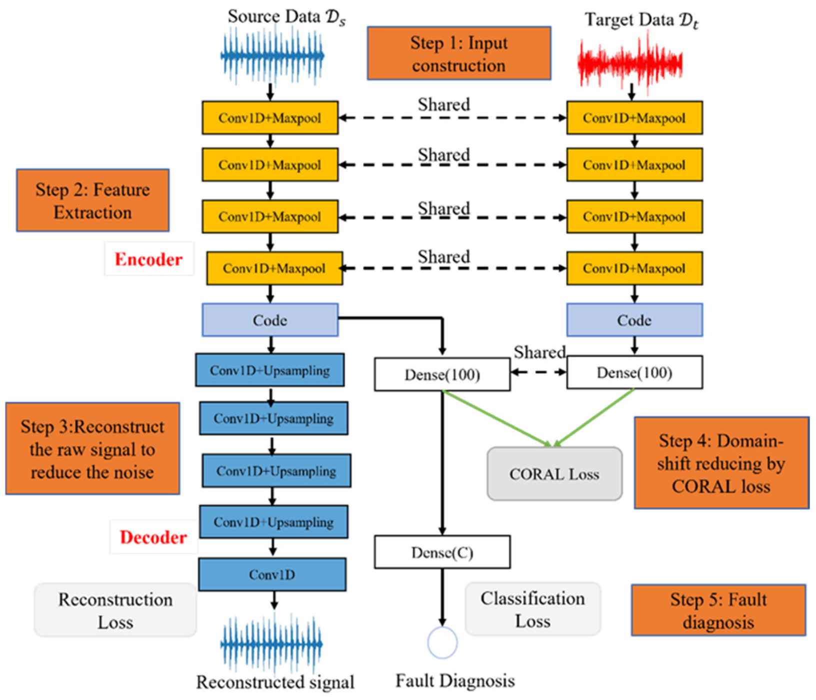

Source Data $\mathcal{D}_s$
Step 1: Input construction
Target Data $\mathcal{D}_t$
Conv1D+Maxpool
Shared
Step 2: Feature Extraction
Encoder
Code
Dense(100)
Conv1D+Upsampling
Step 3:Reconstruct the raw signal to reduce the noise
CORAL Loss
Step 4: Domain-shift reducing by CORAL loss
Decoder
Dense(C)
Conv1D
Reconstruction Loss
Classification Loss
Step 5: Fault diagnosis
Reconstructed signal
Fault Diagnosis

Figure 1. The structure of the proposed method for FD.

3.1. Input Construction

The proposed method for FD utilizes labeled source domain data and unlabeled target domain data as the input of the 1D-CNN for FD. Moreover, to keep the source domain inner character and reduce the noise, the 1D-CNN autoencoder reconstructs the source domain signal. Thus, the proposed method has dual inputs and outputs, as shown in (7) and (8), where x_i^s *and* x_i^t are signals from source $\mathcal{D}^\int$ and target $\mathcal{D}^t$, respectively; N is the number of samples. In practice, we collected one long signal for each kind of faulty data, and the overlap algorithm was applied to construct training samples, as shown in Figure 2. The details of the process for the overlap algorithm are given in Algorithm 1. One long signal Sig with length l could generate $N = \lfloor \frac{l-w}{k} \rfloor + 1, k \leq w$ samples, where k is the overlap step, w is each sample's length, and $\lfloor \cdot \rfloor$ is a round-down operation.

$$Input = (x_i^s, x_i^t)_{i=1}^N \tag{7}$$

$$Output = (x_i^s, y_i^s)_{i=1}^N \tag{8}$$

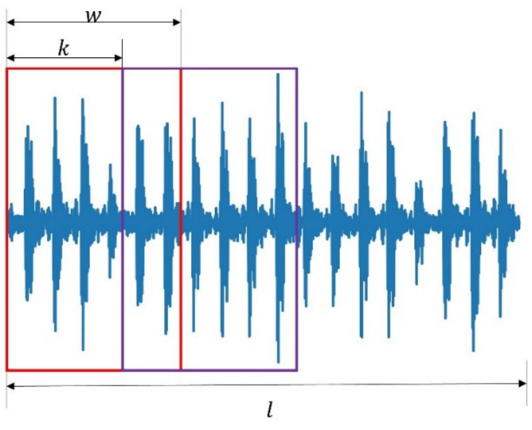


Figure 2. An example of an overlap method to generate training samples.

Algorithm 1: The overlap algorithm to generate training samples.

```
Input: Given one long signal $Sig$ witha length of $l$, whose fault type is $F$; the training sample's number $N$; and the training sample's length $w$.
Output: $N$ training samples $\{x_i, y_i\}$.
1. calculating the overlap step $k = \lfloor \frac{l-w}{N-1} \rfloor$; defined each sample's starting index $s = 0$ and ending index $e = w$; defined $Input$ and $Label$ matrix to save training features $x_i$ and corresponding fault type $y_i$.
2. For $i$ in 1 to $l$:{
3.     $Input[i] = Sig[s : e]$
4.     $Label[i] = F$
5.     $s = s + k$
6.     $e = s + w$}
7. End for
8. Return $Input$ and $Label$
```

3.2. Feature Extraction

The proposed method utilized a 1D-CNN autoencoder to extract hidden features with less noise for source and target data. The encoder was implemented with four stacked convolution blocks (Conv1D + Maxpool) to extract representative features (code), as marked in orange color in Figure 1. The decoder part was used to reduce the noise, as marked in blue color. The term 'Conv1D' is the operation of 1-D convolution, and 'Maxpool' denotes the maximum pooling operation. We set the small output nodes {16,24,24,24} for four 1-D convolution layers. Moreover, the wide kernel size was adopted for convolution operation since it could mine the short-term and long-term relationships in one long vibration signal [25]. Primarily, we set kernel size {12,9,9,9} for four 1D-CNN layers. Moreover, all layers employed ReLU to activate the hidden feature map to strengthen its non-linear expression. The feature extraction process for source and target data are written as (9) and (10), where raw signals x_i^s and x_i^t are fed into four convolution blocks $ConvBlok_4$ to extract the hidden features $Code_s$ and $Code_t$.

$$Code_s = ConvBlok_4(x_i^s) \tag{9}$$

$$Code_t = ConvBlok_4(x_i^t) \tag{10}$$

3.3. Reconstruct the Raw Signal

The above step extracted hidden features. However, we believed it still contained some noise, which may significantly influence the performance of FD. Therefore, the proposed method adopted one decoder to reconstruct the signal to reduce the noise for source domain data. We did not adopt a decoder for target data (1) because only source features are used for FD and the noise in the target features do not influence its effectiveness but adopting a decoder in the target data increases the model's size, and (2) to keep source features' self-domain characters uninfluenced by target features.

We adopted one symmetrical structure with an encoder to reconstruct the source signal $x_i^{s'}$, as marked with blue color in Figure 1, where the term 'Upsampling' is the upsampling operation, and this step could be written as (11). To obtain the near-optimal reconstructed signal $x_i^{s'}$ with less noise, the model calculates the mean square error (MSE) loss between raw signal x_i^s and reconstructed signal $x_i^{s'}$ to update the network with a gradient descent algorithm, denoted as (12).

$$x_i^{s'} = Decoder(Code_s) \tag{11}$$

$$\mathcal{L}_{mse}\left(x_i^s, x_i^{s'}\right) = \frac{1}{N}\sum_{i=1}^{N}\left(x_i^s - x_i^{s'}\right)^2 \tag{12}$$

3.4. Domain Shift Reduction

The previous two steps ensured that the proposed method extracts rich hidden features with less noise while the domain shift between the source and target features still exists. The proposed method calculated the CORAL distance to minimize the domain shift. Before calculating the CORAL distance, one fully connected dense layer with 100 nodes was used to extract the deeper hidden expression h_s and h_t, as shown in (13). They were used to calculate CORAL distance through (4). Significantly, the CORAL distance was designed as one loss function $\mathcal{L}_{coral}$ to update the network in the proposed method. Through $\mathcal{L}_{coral}$, the encoder parts of the source and target domain are connected and shared to learn domain shift knowledge to detect the faults.

$$\begin{cases} h_s = Dense(Code_s) \\ h_t = Dense(Code_t) \end{cases} \tag{13}$$

3.5. Fault Diagnosis

After domain shift reduction, the model learned that domain-invariant features h_s would be used to detect the faults. The proposed method for fault diagnosis was trained by minimizing the classification loss $\mathcal{L}_c$, which we defined as cross-entropy loss, denoted as:

$$\mathcal{L}_c = -\frac{1}{N}\sum_{i=0}^{N-1}\sum_{c=0}^{C} y^s_{i,c} p^s_{i,c} \tag{14}$$

where N denotes the number of samples, C is the faulty types, $y^s_{i,c}$ is the i^{th} source sample within label c, and $p^s_{i,c}$ is the probability of $y^s_{i,c}$. The softmax function maps the final output into one probability vector; the biggest probability is selected as the predicted faulty type.

The final loss for training the model was the combination of CORAL loss $\mathcal{L}_{coral}$, reconstruction loss $\mathcal{L}_{mse}$, and fault diagnosis classification loss $\mathcal{L}_c$, as shown in (15), where $\mathcal{L}_{coral}$ is to minimize the domain shift, $\mathcal{L}_{mse}$ is for the 1D-CNN autoencoder to reduce the noise, and $\mathcal{L}_c$ is for fault diagnosis. Moreover, α is the domain adaption ratio, and β is the reconstruction ratio. They are two tradeoff parameters to balance FD and others. Particularly, we set α as 0.1 since the main task of the proposed model is FD, which was motivated by [32]; a grid search was designed to find the best parameter β, which was set as 10 and is discussed later. Moreover, the proposed model was trained with the optimizer of Adam.

$$\mathcal{L}_{proposed} = \alpha\mathcal{L}_{coral} + \beta\mathcal{L}_{mse} + \mathcal{L}_c \tag{15}$$

4. Experimental Verification

To verify the effectiveness of the proposed method, we implemented the proposed method based on the operating system of ubuntu 16.04 with intel(R) i7 700 CPU. The programming language was python 3.5, and the deep learning platform was Keras.

4.1. Data

Case Western Reserve University (CRWU) bearing data center provides some public faulty bearing data sets, which were utilized to test the proposed method's performance. We adopted data collected at 12,000 samples per second for drive end bearing and normal experiments. It consisted of four faulty types according to faulty diameter, including 7, 14, 21, and 28 mils under four different loads: 0 power horse (PH), 1 PH, 2 PH, and 3 PH. Each faulty type was caused by three different components of bearing: inner ring (IR), ball, and outer ring (OR) except for 28 mils only caused by IR and ball. Thus, each load of faulty data consisted of 11 = 3 $\times$ 3 + 2 faulty types: IR7, Ball7, OR7, IR14, Ball14, OR14, IR21, Ball21, OR21, IR28, Ball28, and one normal type. Therefore, the FD problem was one 12 classification problem.

For simplicity, we integrated the data into four subsets: A, B, C, and D corresponding to 0 PH, 1 PH, 2 PH, and 3 PH for analysis, respectively. Each subset was a long time series processed by the overlap Algorithm 1 to generate training samples, as shown in Figure 2. Primarily, we generated 685 samples for each fault with a length of 2048, which is identical to [5]. That is, each subset consisted of 8220 = 685 × 12 samples. Moreover, one summary subset E that combines A, B, C, and D was built to also analyze. The details of each subset are given in Table 1. Here, rpm is revolutions per minute.

Table 1. Data description.

Subset	Samples	Faulty Types	Conditions (Load, Speed)
A	8220	IR7, Ball7, OR7, IR14, Ball14, OR14, IR21, Ball21, OR21, IR28, Ball28, and normal	0 PH, 1797 rpm
B	8220	IR7, Ball7, OR7, IR14, Ball14, OR14, IR21, Ball21, OR21, IR28, Ball28, and normal	1 PH, 1772 rpm
C	8220	IR7, Ball7, OR7, IR14, Ball14, OR14, IR21, Ball21, OR21, IR28, Ball28, and normal	2 PH, 1750 rpm
D	8220	IR7, Ball7, OR7, IR14, Ball14, OR14, IR21, Ball21, OR21, IR28, Ball28, and normal	3 PH, 1730 rpm
E	32,880	IR7, Ball7, OR7, IR14, Ball14, OR14, IR21, Ball21, OR21, IR28, Ball28, and normal	0,1,2,3 PH 1797, 1772, 1750, 1730 rpm

We give an example for each fault under different loads to illustrate the difficulties for FD, as shown in Figure 3. Although in the same load, each faulty signal shocked fluently and randomly, which increased the difficulty for FD. Primarily, it was difficult to identify (IR7, IR28), (Ball14, OR14), and (OR7, IR14, OR21, OR). Moreover, the same faults displayed differently under different loads, e.g., Ball21 in subset A was far from subset B. In contrast, some faults had high similarities, e.g., Ball28 in subset C was similar to IR14 in subset D.

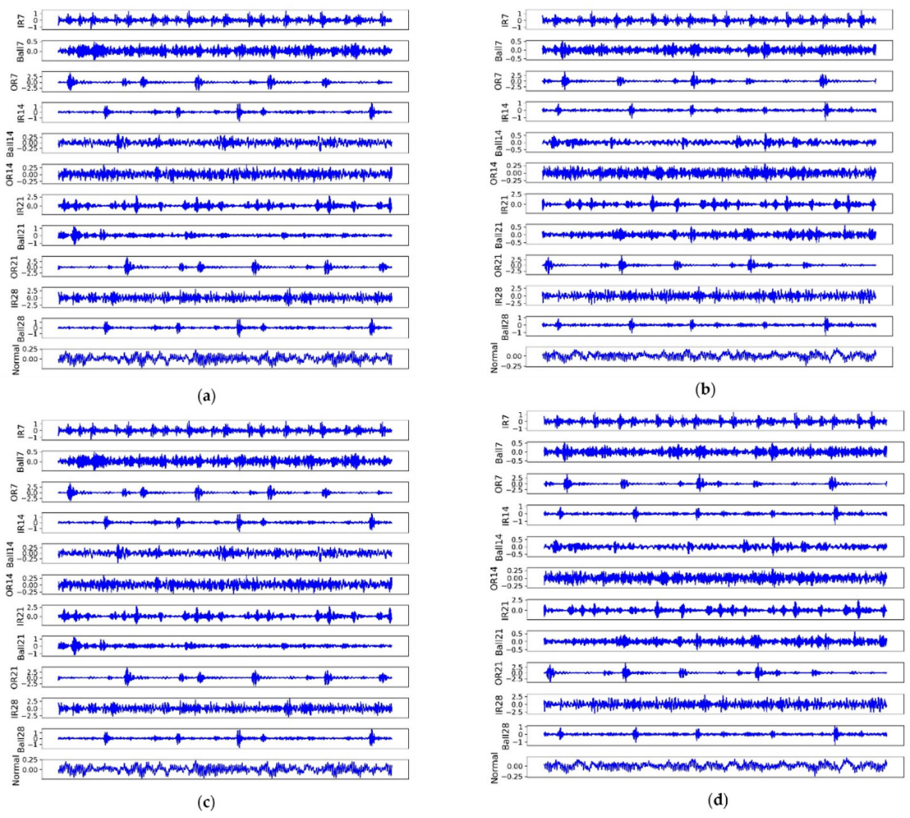
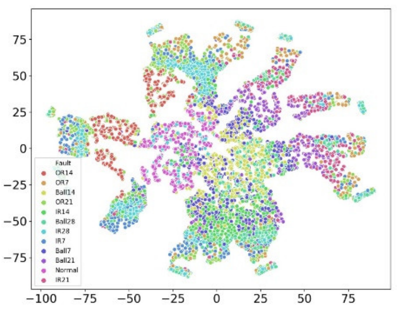
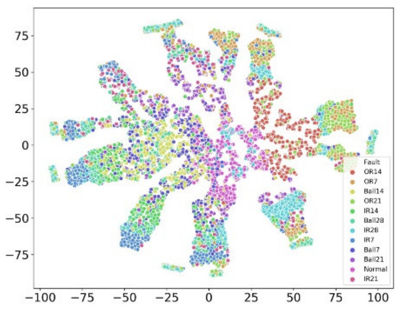


Figure 3. The raw signal for each fault in four subsets (loads). (**a**) Is subset A; (**b**) is subset B; (**c**) is subset C, and (**d**) is subset D.

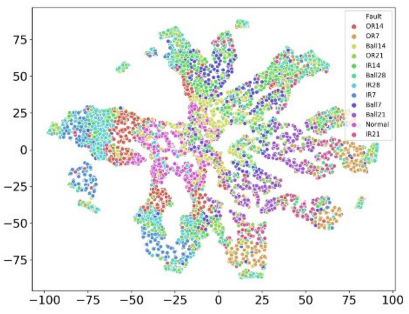
Moreover, we utilized t-distributed stochastic neighbor embedding (t-SNE) technology to see the inner distribution of each fault under different loads, as shown in Figure 4. The results showed that each fault had a different distribution under different loads. For instance, the distribution OR14 for subsets A and B were relatively concentrated while others were not; Ball 7 was distributed concentratedly for subset C and others were distributed sparsely. Moreover, different faults were mixed, inseparable, non-linear, and difficult to identify. Similar to IR28 and Ball 28 in subset A were IR7 and IR28 in subset B, OR14 and normal in Subset C, and IR7 and Ball 28 in subset D. One common finding was that all faults were distributed in a transmitting ring for different loads, which led us to apply deep transfer learning to detect the faults under a complex environment.



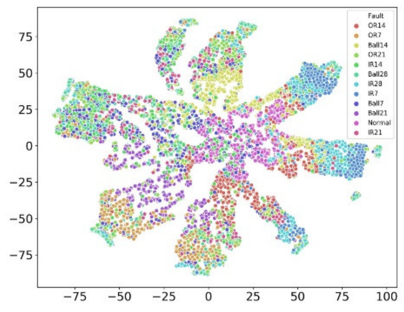
(**a**)



(**b**)



(**c**)



(**d**)

Figure 4. Each fault inner distribution visualization under different loads via t-SNE. (**a**) Is subset A; (**b**) is subset B; (**c**) is subset C, and (**d**) is subset D.

Above mentioned characteristics make FD more challenging. To overcome those issues, the proposed model must learn rich distinguishable features and have an excellent transferable capacity as we cannot collect all kinds of faulty data under a complex production environment. Therefore, this manuscript combined a 1D-CNN autoencoder and CORAL to extract rich distinguishable, domain-invariant features with less noise for FD, in which 1D-CNN autoencoder was to extract rich distinguishable features while CORAL was to minimize the domain shift between source and target data.

4.2. One Domain Fault Diagnosis

To verify the proposed method's performance for FD, we first analyzed its performance for one-domain FD, i.e., training and testing the model in the same load. Primarily, we compared the proposed method with both traditional machine learning methods: SVM [6] and RF [8], and current state-of-the-art deep learning-based methods: WDCNN [22], MSFFCNN [25], MDCNN [5], and MSCNN [24]. Particularly, we implemented SVM and RF based on the library of 'sklearn' with default settings and strictly reproduced deep learning models according to the given parameters in the papers. For the proposed method, we adopted the 1D-CNN autoencoder for one-domain FD, in which the parameter β was set as 10 to reconstruct the signal, and its influence is discussed later.

Moreover, each method was fairly verified through a five-fold cross-validation approach, the workflow of which is given in Figure 5. Firstly, the collected vibration signals were processed with Algorithm 1 to generate input matrices (7) and (8). Then, the input and corresponding labels were randomly split into five equal parts. Four of them were used as a training set to train the model, while the rest was for testing. Thirdly, 80% of the training set was used to train the model, while 20% was used as validation data to find the best convergence path with an early stop strategy. Primarily, we set training epochs as 100, and patience was five, i.e., if the validation accuracy was not increased in five continuous epochs, the training process stopped. Then, the model with the highest validation accuracy was saved for evaluating the model on the testing subset. Otherwise, the training process executed until 100 epochs.

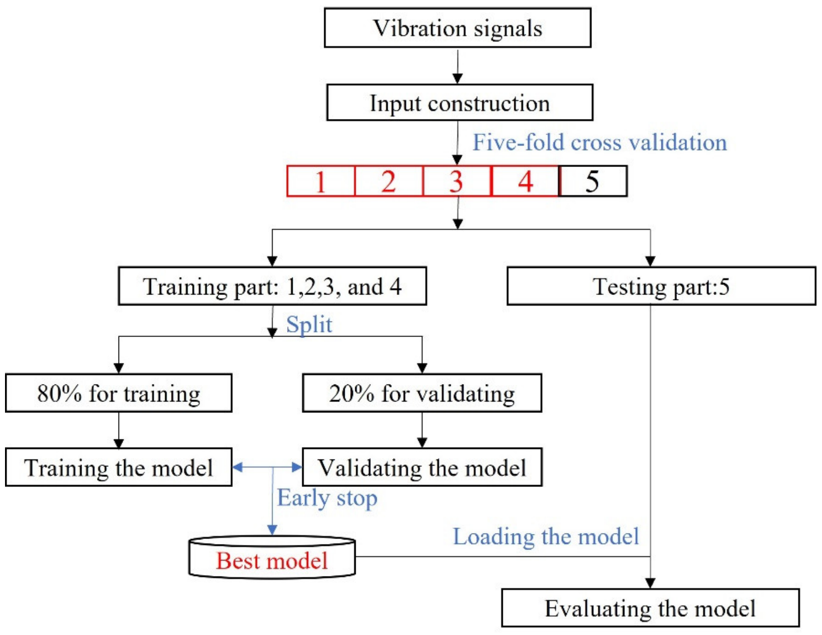


Figure 5. The workflow for one-domain fault diagnosis testing used a five-fold cross-validation approach.

We give one training loss curve example of the proposed method on subset A, as shown in Figure 6. The results showed that the training process ended at epoch 37, and the training loss mainly was controlled by classification loss and CORAL loss as the reconstruction loss was much less than them.

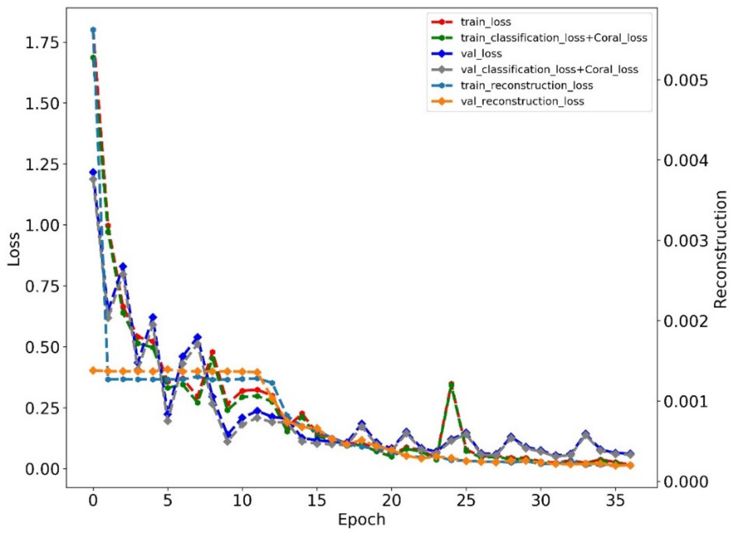


Figure 6. The training loss of the proposed method on subset A.

We adopted the average accuracy of five-fold cross-validation to evaluate each method, as shown in Table 2. The results showed that the proposed method received the highest average accuracy of 99.93% for five subsets with the lowest standard error of 0.07%. It performed the best on subsets B, C, and D. Significantly, the proposed method received an accuracy of 100% on subsets C and D. Moreover, the proposed method did not require any additional inputs. In contrast, MSCNN requires calculating three-scale mean values, and MDCNN requires calculating six statistic indexes and DWT transformed coefficients, and therefore, we saved much time for training the model. Although WDCNN did not require any additional inputs, its accuracy was lower than the proposed method. Another finding was that the traditional machine learning method performed worst without the feature selection operation; their accuracies were lower than 75%. In contrast, deep learning-based methods performed much better, whose accuracies were near 100%. Those methods could be ranked as follows: the proposed method > MSCNN > MDCNN > MSFFCNN > WDCNN > RF > SVM according to the average accuracy of five subsets and standard errors.

Table 2. The comparative results for one-domain FD (%).

Method	A	B	C	D	E	Average
SVM [6]	66.40 ± 0.39	71.09 ± 0.97	67.81 ± 1.05	70.82 ± 0.92	65.93 ± 0.27	68.41 ± 2.17
RF [8]	71.56 ± 3.72	75.21 ± 1.28	74.31 ± 0.91	76.50 ± 0.39	74.26 ± 0.48	74.37 ± 1.62
WDCNN [22]	99.34 ± 0.40	99.04 ± 0.14	99.88 ± 0.10	99.91 ± 0.08	99.70 ± 0.15	99.57 ± 0.34
MSFFCNN [25]	99.66 ± 0.30	99.57 ± 0.22	**100 ± 0.0**	98.99 ± 0.02	99.80 ± 0.29	99.60 ± 0.34
MDCNN [5]	99.81 ± 0.30	**99.94 ± 0.0**	99.96 ± 0.05	**100.0 ± 0.0**	99.90 ± 0.05	99.92 ± 0.06
MSCNN [24]	**99.94 ± 0.01**	99.76 ± 0.06	**100 ± 0.0**	**100.0 ± 0.0**	**99.96 ± 0.12**	**99.93 ± 0.09**
1D-CNN autoencoder	99.82 ± 0.19	**99.94 ± 0.11**	**100 ± 0.0**	**100.0 ± 0.0**	99.90 ± 0.05	**99.93 ± 0.07**

4.3. Cross-Domain Fault Diagnosis

The previous subsection confirmed the effectiveness of the proposed method for one-domain FD. In practice, collecting all kinds of data is difficult and even not available. Therefore, building one transferable model that trains the model on the source data and performs well on unseen target data is critical and necessary. In this manuscript, we defined twelve transfer learning tasks: A→B, A→C, A→D, B→A, B→C, B→D, C→A, C→B, C→D, D→A, D→B, and D→C to verify the proposed method's effectiveness, where A→B means training the model on the subset A while testing it on subset B, and so forth.

We compared the proposed method with several machine learning methods and deep learning-based methods, identical to one-domain FD, to illustrate its priority for cross-domain FD. Moreover, we compared it with two domain adaptive methods: WDCNN + AdaBN [23] and the deep adaptive model with MMD (DaMMD) [29]. Notice that we combined the structure of 1D-CNN in the proposed method and MMD for DaMMD to compare fairly.

We trained and tested each comparative method for each task ten times. Each time adopted an early stop strategy to find the best model. The results as the ten-time average accuracy are shown in Table 3. The results indicated that the proposed method outperformed others in the average accuracy of twelve tasks, up to 96.40%. Traditional machine learning methods performed worse for cross-domain FD. Particularly, SVM and RF only obtained 89.58% and 47.47%, respectively, for twelve tasks. The deep learning-based methods could be divided into two groups according to their performance. One includes WDCNN and MDCNN as their accuracy was higher than 90%; another consists of MSFCNN and MSCNN as their accuracy was lower than 90%. For the domain adaptive deep model, WDCNN + AdaBN obtained 94.92% average accuracy for twelve tasks, and DaMMD was 92.68%. Those methods could be ranked as follows: the proposed method > WDCNN + AdaBN > MDCNN > WDCNN > DaMMD > MSFFCNN > MSCNN > SVM > RF. Not supervised, the proposed method showed a significant improvement compared to others. Significantly, it improved by 1.56% compared to WDCNN + AdaBN; 2.01% compared to MDCNN; 4.01% compared to DaMMD; and 103.08% compared to RF.

Table 3. The comparative results for cross-domain FD (%).

Method	A→B	A→C	A→D	B→A	B→C	B→D	C→A	C→B	C→D	D→A	D→B	D→C	Average
SVM [6]	55.80	53.89	54.45	56.14	59.83	67.06	52.12	66.81	68.64	52.23	66.62	61.35	59.58 ± 6.04
RF [8]	48.81 ± 0.52	46.70 ± 0.67	45.82 ± 0.52	48.13 ± 0.41	50.34 ± 0.61	48.59 ± 0.32	44.96 ± 0.56	49.35 ± 0.34	50.05 ± 0.61	42.34 ± 0.54	45.96 ± 0.71	48.55 ± 0.73	47.47 ± 2.26
WDCNN [22]	**97.35 ± 3.03**	96.12 ± 4.32	92.86 ± 7.01	97.16 ± 1.21	99.25 ± 2.03	97.38 ± 2.13	92.45 ± 3.15	97.46 ± 1.35	97.11 ± 3.12	82.89 ± 7.45	85.52 ± 6.32	93.09 ± 7.14	94.05 ± 4.88
MSFFCNN [25]	91.98 ± 9.12	93.25 ± 7.13	85.07 ± 8.32	92.94 ± 5.00	95.29 ± 5.01	90.24 ± 6.12	81.02 ± 12.11	86.80 ± 12.35	85.12 ± 14.02	85.35 ± 5.01	86.50 ± 7.12	92.08 ± 8.31	88.80 ± 4.70
MDCNN [5]	96.76 ± 2.01	94.91 ± 2.13	94.32 ± 4.12	96.22 ± 2.03	**99.84 ± 1.01**	98.90 ± 1.03	93.79 ± 1.23	97.02 ± 2.23	98.22 ± 3.00	86.05 ± 6.08	84.28 ± 8.75	93.56 ± 6.44	94.50 ± 4.59
MSCNN [24]	86.85 ± 5.18	81.85 ± 4.67	80.75 ± 9.26	85.52 ± 5.24	97.87 ± 2.29	92.12 ± 3.24	81.13 ± 5.78	93.60 ± 0.59	**99.36 ± 0.59**	72.65 ± 5.94	77.92 ± 4.00	86.80 ± 5.31	86.18 ± 7.83
WDCNN + AdaBN [23]	91.97 ± 2.81	**98.02 ± 0.98**	89.37 ± 4.43	94.98 ± 2.85	99.66 ± 1.03	97.93 ± 0.90	94.30 ± 1.38	97.71 ± 0.36	98.14 ± 1.19	87.01 ± 3.32	91.97 ± 2.81	98.02 ± 0.98	94.92 ± 3.87
DaMMD [29]	90.03 ± 1.42	87.66 ± 1.43	77.41 ± 2.05	97.17 ± 0.18	95.09 ± 0.74	90.02 ± 2.35	**95.73 ± 0.48**	**97.78 ± 0.16**	92.35 ± 0.80	**93.28 ± 0.33**	96.91 ± 0.22	**98.69 ± 0.33**	92.68 ± 5.69
Proposed	96.88 ± 0.53	96.06 ± 0.69	**95.07 ± 1.21**	**99.41 ± 0.22**	99.47 ± 0.27	**97.10 ± 0.80**	93.00 ± 1.20	96.40 ± 0.43	97.51 ± 0.24	91.95 ± 0.75	95.68 ± 0.79	98.25 ± 0.59	**96.40 ± 2.19**

Moreover, only the proposed method performed better than 90% for all transfer tasks. It received the three highest accuracies for tasks A→D, B→A, and B→D, with accuracies of 95.07%, 99.41%, and 97.10%, respectively. Even though DaMMD obtained the four best accuracies, it was not stable. Primarily, it only received an accuracy of 77.41% for A→D. On the contrary, the proposed method received the lowest standard error of 2.19%, and it was much more robust for cross-domain FD.

To quantify the difference among those methods, we calculated the *p*-value of the *t*-test, as shown in Figure 7. Notice that we only kept two decimal places. The findings indicated that the proposed method was significantly different from SVM, RF, MSFFCNN, and MSCNN due to their *p*-values being much less than 0.05, and it was a little different from DaMMD due to the *p*-value being 0.06. Moreover, there was no significant difference between the proposed method and WDCNN, MDCNN, and WDCNN + AdaBN. However, the proposed method was more accurate and stable. In addition, SVM and RF were significantly different from others.

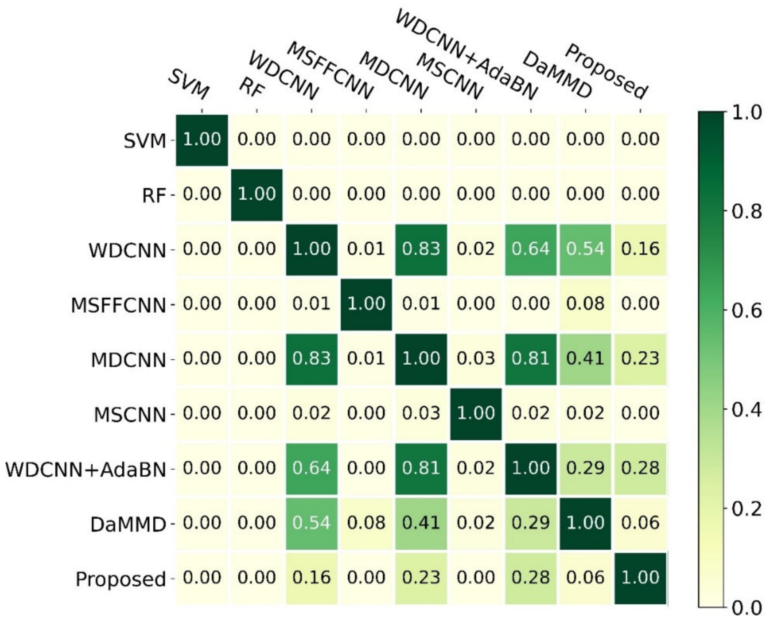


Figure 7. The *p*-values of the *t*-test for different methods.

The above analysis confirmed that the proposed method could detect the fault accurately and robustly under a complex environment using the raw signals without any labeled target domain samples.

4.4. The Effectiveness of Each Component

We designed an ablation study to explore the effectiveness of each component in the proposed method. Specifically, we designed a 1D-CNN and 1D-CNN autoencoder to validate the effectiveness of the autoencoder. We designed 1D-CNN+CORAL to validate the effectiveness of CORAL; designed the proposed method to see the effectiveness of the combination of 1D-CNN, autoencoder, and CORAL; and designed the proposed method with two decoders (Proposed$_{\text{two}}$) to verify their effectiveness. The results of the ablation study are shown in Table 4. The results showed that 1D-CNN obtained 92.77% average accuracy for twelve tasks, higher than DaMMD. The application of the autoencoder improved the accuracy from 92.77% to 92.79. Moreover, the usage of CORAL improved the accuracy by 0.05%, which could be calculated by comparing the 1D-CNN and 1D-CNN + CORAL. By combining 1D-CNN, autoencoder, and CORAL, the average accuracy was 96.40%, which improved by 3.63% compared to 1D-CNN. Although the Proposed$_{\text{two}}$ won serval cases such as A→C, A → D, and B → D**,** its average accuracy for twelve tasks was lower than the proposed method while its standard error was higher. Particularly, the utilization of the decoder part in the target data decreased the average accuracy by 0.19%. In addition, it increased the model's size, which was the reason we adopted the decoder part in the source data for the proposed method.

Table 4. The results of the ablation study.

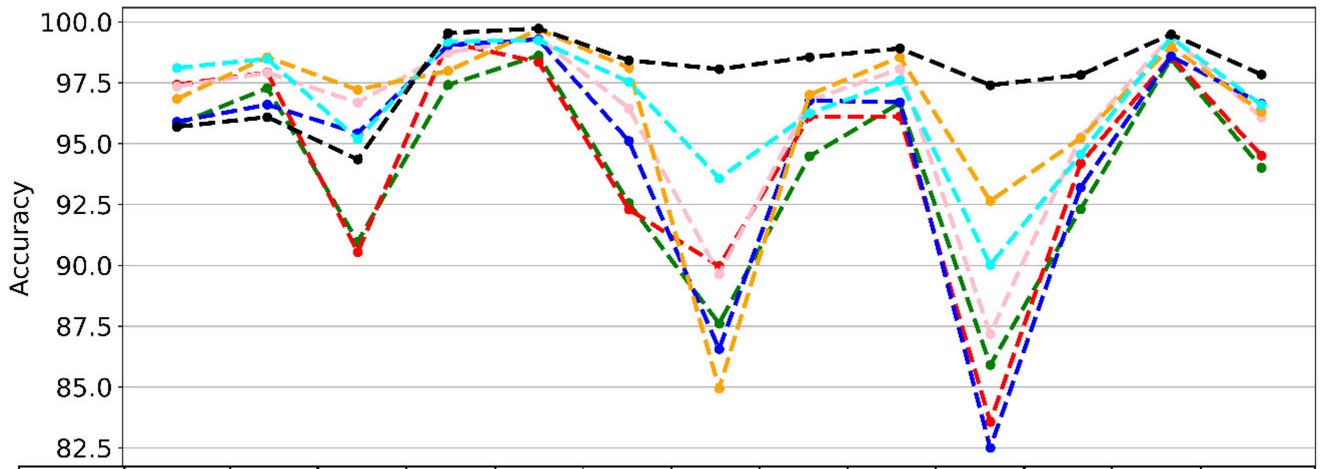
Method	**A→B**	**A→C**	**A→D**	**B→A**	**B→C**	**B→D**	**C→A**	**C→B**	**C→D**	**D→A**	**D→B**	**D→C**	**Average**
1D-CNN	94.49 ± 4.29	95.27 ± 3.27	87.53 ± 6.02	95.29 ± 4.14	99.94 ± 0.08	92.61 ± 3.08	92.08 ± 2.25	96.56 ± 1.72	93.82 ± 6.52	86.14 ± 4.22	86.25 ± 4.48	93.27 ± 6.44	92.77 ± 4.05
1D-CNN autoencoder	94.91 ± 0.04	95.47 ± 0.04	88.80 ± 0.05	95.53 ± 0.04	**99.81 ± 0.00**	93.64 ± 0.05	93.15 ± 0.04	97.40 ± 0.01	94.42 ± 0.04	87.64 ± 0.06	87.01 ± 0.04	93.70 ± 0.07	92.79 ± 4.98
1D-CNN + CORAL	92.52 ± 1.77	83.81 ± 2.48	81.15 ± 3.31	96.74 ± 0.74	94.64 ± 1.16	87.76 ± 1.41	**93.31 ± 0.52**	**98.29 ± 0.15**	94.95 ± 0.57	**93.34 ± 0.61**	**96.96 ± 0.12**	**98.60 ± 0.21**	92.83 ± 5.43
Proposed$_{\text{two}}$	96.53 ± 0.46	**96.64 ± 1.17**	**97.23 ± 0.40**	**99.60 ± 0.22**	**99.64 ± 0.07**	**98.78 ± 0.34**	91.28 ± 2.50	93.71 ± 2.41	97.50 ± 0.37	89.21 ± 0.74	96.48 ± 0.48	97.99 ± 1.11	96.21 ± 3.11
Proposed	**96.88 ± 0.53**	96.06 ± 0.69	95.07 ± 1.21	99.41 ± 0.22	99.47 ± 0.27	97.10 ± 0.80	93.00 ± 1.20	96.40 ± 0.43	**97.51 ± 0.24**	91.95 ± 0.75	95.68 ± 0.79	98.25 ± 0.59	**96.40 ± 2.19**

We calculated the contribution ratio $C_{ration} = \frac{Ip_c}{Ip_{total}}$ for each component, where Ip_c denotes the improvement of each component and Ip_{total} is the actual improvement of the proposed method. The results showed that autoencoder contributed to a 0.56% improvement while CORAL contributed 1.38%. The combination of the autoencoder and CORAL contributed 98.07%.

4.5. Anti-Noise Testing

The data collected from the natural production environment convolves some noises, which increases the difficulty for FD. It requires the proposed model to have an excellent anti-noise capacity. We tested the proposed method's anti-noise capacity on twelve transfer tasks with simulated white noise. Particularly, different intensity white noises were added into raw signals to train and test the model. The intensity of white noise was measured with the signal-noise ratio (SNR), defined as (16), where p_s and p_w are the power of the signal and white noise, respectively, whose unit is the decibel (dB). The testing results with different SNR ranged from -4 dB to 8 dB, as shown in Figure 8. The results showed that the proposed method has an excellent anti-noise capacity. Significantly, the proposed method's average accuracy for twelve transfer tasks was higher than 94%. The lowest accuracy was 94.01% under the noise of −4 dB, while the highest was 97.84% under an 8 dB noisy environment.

$$\text{SNR} = 10log\frac{p_s}{p_w} \tag{16}$$



SNR	A→B	A→C	A→D	B→A	B→C	B→D	C→A	C→B	C→D	D→A	D→B	D→C	Average
-4dB —●—	95.78 ±0.61	97.28 ±0.64	90.97 ±4.33	97.41 ±0.52	98.62 ±0.82	92.56 ±1.99	87.61 ±6.25	94.48 ±2.24	96.76 ±2.36	85.91 ±2.24	92.31 ±3.70	98.48 ±0. 72	94.01 ±4.03
-2dB —●—	97.42 ±0.19	97.92 ±0.44	90.54 ±2.34	99.16 ±0.27	98.35 ±0.57	92.30 ±0.86	89.97 ±4.35	96.11 ±0.46	96.12 ±0.73	83.57 ±1.19	94.18 ±0.85	98.57 ±0.79	94.51 ±4.46
0dB —●—	97.35 ±0.48	97.91 ±0.46	96.69 ±1.40	98.74 ±0.49	99.35 ±0.22	96.54 ±3.34	89.64 ±1.81	96.77 ±0.52	98.07 ±0.34	87.18 ±2.37	95.27 ±1.15	99.42 ±0.36	96.07 ±3.65
2dB —●—	95.90 ±0.56	96.60 ±0.89	95.43 ±0.98	99.05 ±0.77	99.28 ±0.23	95.11 ±1.17	86.56 ±3.80	96.77 ±0.40	96.70 ±1.21	82.50 ±3.57	93.20 ±1.47	98.59 ±0.91	96.64 ±4.89
4dB —●—	96.84 ±0.60	98.55 ±0.44	97.21 ±0.68	97.99 ±1.00	99.69 ±0.10	98.12 ±1.27	84.94 ±1.27	97.01 ±0.85	98.55 ±0.44	92.64 ±2.42	95.23 ±2.71	98.99 ±0.20	96.31 ±3.88
6dB —●—	98.11 ±0.26	98.48 ±0.84	95.18 ±1.27	99.20 ±0.32	99.24 ±0.14	97.54 ±0.56	93.57 ±1.05	96.24 ±0.59	97.60 ±0.44	90.02 ±0.88	94.55 ±1.19	99.36 ±0.13	96.59 ±2.71
8dB —●—	95.69 ±1.36	96.09 ±0.91	94.35 ±2.10	99.54 ±0.15	99.73 ±0.15	98.42 ±0.39	98.06 ±0.53	98.55 ±0.26	98.91 ±0.26	97.40 ±0.42	97.82 ±0.32	99.49 ±1.19	97.84 ±1.61

Figure 8. The results of antinoise testing.

Moreover, the model's performance increased with the SNR, which means that the model performs better for FD with less noise. Moreover, all cases showed the same trend. Mainly, the proposed method performed worse for A→D, C→A, and D→A, while it performed better for B→A, B→C, C→D, and D→C. Another finding was that the proposed method with a bit of noise, especially at 6 dB (96.59%) and 8 dB (97.84%), performed better than without noise (96.40%). It indicated that adding a little noise could help the model learn more distinguishable features.

To quantify the difference between the proposed method under noisy and non-noisy environments, we calculated the p-value of the t-test, as shown in Figure 9. The results indicated no significant difference between non-noisy and noisy environments as their p-values were higher than 0.05 (the red dotted line). Moreover, the proposed method with noise performed better than all comparative methods, except for WDCNN + AdaBN which performed a little better than the proposed method under the noise of −4 dB and −2 dB; WDCNN and MDCNN performed a little better than the proposed method at −4 dB.

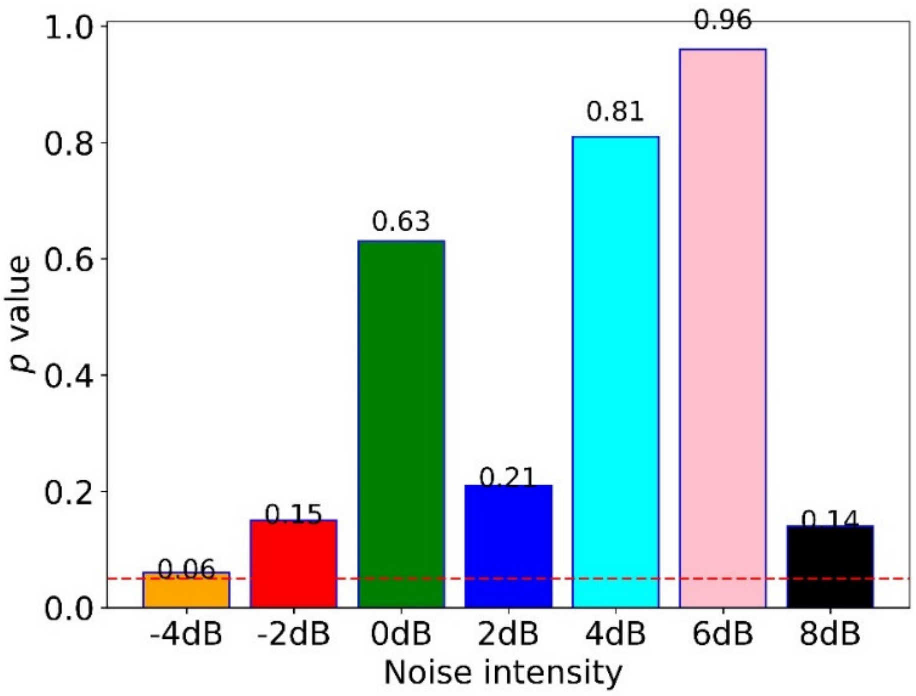


Figure 9. The *p*-value of the *t*-test for the proposed method under noisy and non-noisy environments.

4.6. The Effectiveness of Tradeoff Parameter β

To explore the influence of the reconstruction ratio β, we designed five sub-experiments with different α for cross-domain FD, where α was from 0.1 to 20. The results for twelve tasks are shown in Table 5. The results showed no apparent patterns in different β, but the proposed method obtained the highest accuracy when we set it as 10. Moreover, all five sub-experiments' accuracies were higher than 90%, better than MSCNN and MSFFCNN. To quantify the difference between 10 and others, we calculated the *p*-value of the *t*-test, as shown in Figure 10. The findings indicated that only the reconstruction ratio of 0.1 had a significant difference from 10. In contrast, others did not, which could be calculated by comparing their *p*-values with 0.05 (dotted line in Figure 10). This finding suggested that setting a slightly big reconstruction ratio is better, especially setting ten as the reconstruction ratio.

Table 5. The effectiveness of reconstruction ratio β.

β	**A→B**	**A→C**	**A→D**	**B→A**	**B→C**	**B→D**	**C→A**	**C→B**	**C→D**	**D→A**	**D→B**	**D→C**	**Average**
0.1	94.42 ± 0.57	93.82 ± 0.18	85.98 ± 2.07	96.85 ± 0.67	96.49 ± 0.87	88.76 ± 1.99	87.35 ± 2.37	96.20 ± 0.80	96.74 ± 0.18	77.58 ± 1.58	91.34 ± 1.54	98.05 ± 0.52	91.97 ± 5.82
0.5	94.48 ± 0.50	**96.26 ± 0.29**	94.19 ± 1.46	98.12 ± 0.34	98.00 ± 0.44	94.86 ± 0.70	91.96 ± 1.11	95.76 ± 0.63	97.13 ± 0.52	89.56 ± 1.87	94.27 ± 0.72	**98.73 ± 0.21**	95.28 ± 2.56
1	93.91 ± 3.27	95.55 ± 3.87	94.96 ± 1.57	95.63 ± 0.88	97.04 ± 0.51	93.69 ± 1.60	90.07 ± 4.06	95.03 ± 0.24	96.71 ± 0.60	84.72 ± 1.62	94.05 ± 0.41	98.25 ± 0.31	94.13 ± 3.45
5	94.01 ± 0.91	94.70 ± 0.50	95.64 ± 2.14	98.03 ± 0.53	96.72 ± 0.43	93.85 ± 1.84	91.36 ± 5.70	92.57 ± 0.80	91.09 ± 2.45	95.09 ± 4.64	94.13 ± 2.05	97.47 ± 0.17	93.86 ± 2.17
10	**96.88 ± 0.53**	96.06 ± 0.69	**95.07 ± 1.21**	**99.41 ± 0.22**	**99.47 ± 0.27**	**97.10 ± 0.80**	**93.00 ± 1.20**	**96.40 ± 0.43**	**97.51 ± 0.24**	**91.95 ± 0.75**	**95.68 ± 0.79**	98.25 ± 0.59	**96.39 ± 2.19**
20	95.31 ± 0.79	95.93 ± 0.65	87.59 ± 3.11	99.18 ± 4.58	97.76 ± 1.08	88.80 ± 1.99	92.93 ± 1.25	96.37 ± 0.31	94.02 ± 1.71	83.10 ± 1.97	93.98 ± 0.88	98.01 ± 0.53	93.58 ± 4.60

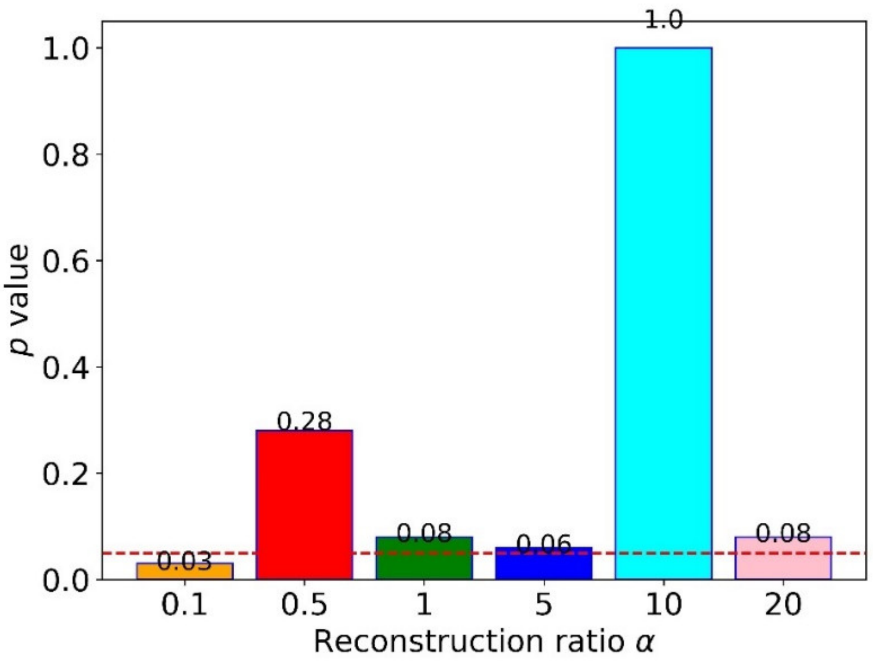


Figure 10. The *p*-values of the *t*-test between a reconstruction ratio of ten and others.

5. Discussion

FD plays a critical role in building a smart factory, which can help the manager find the fault timely to avoid accidents and improve the system's efficiency. Learning-based methods that can extract the feature automatically from raw signals have been widely applied for FD. However, most of them assume that we can collect sufficient historical data to train the model, which is not easy in practice. Moreover, they assume that the source and target data have the same distribution, decreasing the FD's performance.

This manuscript proposed a novel, domain adaptive, and effective deep model based on 1D-CNN for cross-domain FD to solve the above issues, as shown in Figure 1. The 1D-CNN autoencoder was developed to extract rich, less-noisy hidden features from raw signals. The CORAL processed the extracted features from the source and target data to minimize domain shift. Therefore, the proposed method could accurately detect the faults on unseen target domain data using a model trained on source domain data.

To validate the effectiveness of the proposed method, we compared the proposed method with some learning-based methods on CRWU bearing data sets. Significantly, we verified the one-domain FD's capacity using a five-fold cross-validation approach, as shown in Figure 5. The comparative analysis confirmed its effectiveness for one-domain FD, as shown in Table 2. Significantly, the proposed method won three times for five subsets and received the highest average accuracy.

To validate the effectiveness of the proposed method for cross-domain FD, we compared the proposed method with other leading methods on twelve transfer tasks. The experimental results showed that the proposed method outperformed others, as shown in Table 3. Significantly, the proposed method obtained 96.40% average accuracy. Moreover, only the proposed method's accuracy was higher than 90% for all tasks; it obtained the three highest accuracies for tasks A→D, B→A, and B→D, with accuracies of 95.07%, 99.41%, and 97.10%, respectively. From the view of standard error, we could conclude that the proposed method has good robustness, with a standard error of 2.19%. The *t*-test results indicated that the proposed method was significantly different from SVM, RF, MSFFCNN, and MSCNN due to their *p*-values being much less than 0.05, and it was slightly different from DaMMD due to the *p*-value being 0.06, as shown in Figure 7. Moreover, there was no significant difference between the proposed method and WDCNN, MDCNN, and WDCNN + AdaBN.

An ablation study was designed to validate each component's effectiveness in the proposed method. The results showed that 1D-CNN obtained 92.77% average accuracy for twelve tasks, which was higher than DaMMD; the application of the autoencoder improved the accuracy by 0.02%, and the CORAL improved the accuracy by 0.05%. By combining 1D-CNN, autoencoder, and CORAL, the average accuracy increased up to 96.40%, as shown in Table 4.

The anti-noise testing results showed that the proposed method is not sensitive to noise, as shown in Figure 8. Significantly, the average accuracy of the proposed method was higher than 94%. It obtained the lowest accuracy of 94.01%, and the highest was 97.84%. Additionally, the model's performance increased with the SNR, which means that the model performs better for FD with less noise. Moreover, the results showed that adding a little noise could increase the performance of the proposed method. In addition, there was no difference between non-noisy and noisy environments (-4 dB to 8 dB) for the proposed method, which is shown in Figure 9.

To explore the effect of the reconstruction ratio β in the proposed method, we designed six sub-experiments with different β from 0.1 to 20, as shown in Table 5. The results showed that the proposed method performed the best when setting β as 10. Moreover, the proposed method was not sensitive to the reconstruction ratio when setting it a little big, which is shown in Figure 10.

The proposed method has 357,161 parameters and takes up to 4.4 megabytes (MB), and each training step takes three milliseconds (ms). It is easy to deploy to all kinds of clients for real-time FD, and it takes around 0.7 ms to detect one sample.

As discussed above, different reconstruction ratios influence the FD's performance. This manuscript selected the best parameter manually, which was time-consuming. In the future, we will develop an automatic structure to search for a better reconstruction ratio from a wide range. Moreover, the authors will test the proposed method's generalizability on other time series data sets.

6. Conclusions

This manuscript proposed a novel, lightweight, and domain adaptive framework based on 1D-CNN for accurately detecting the faults under complex environments. Particularly, 1D-CNN with an autoencoder structure was designed to extract the rich hidden features with less noise from raw signals; CORAL processed the extracted features to minimize the domain shift. Therefore, the proposed could extract rich, domain-invariant features with less noise for FD. The massive comparative experiments indicated that the proposed method could accurately detect the faults under a complex environment only using raw signals without labeled target samples. Significantly, the proposed method obtained near 100% accuracy for one-domain FD and 96% for cross-domain FD. Moreover, the proposed method has a good anti-noise capacity, and the noise almost does not influence the proposed method. On the contrary, adding little noise can improve its performance.

In the future, we will design an automatic structure based on deep reinforcement learning (DRL) to search for the best reconstruction ratio under the proposed framework for FD more accurately. Additionally, we will verify its generalizability on other time series data sets.

Author Contributions: Conceptualization, X.S. and C.-S.K.; methodology, X.S.; algorithm, X.S.; validation, X.S. and C.-S.K.; experimental analysis, X.S.; writing—original draft preparation, X.S.; writing—review and editing, X.S.; supervision, C.-S.K.; project administration, C.-S.K.; funding acquisition, C.-S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Pukyong National University 2022.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data are from CWRU bearing data center.

Acknowledgments: We would like to thank CWRU bearing data center for providing the data set.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Randall, R.B.; Antoni, J. Rolling element bearing diagnostics—A tutorial. *Mech. Syst. Signal Processing* **2011**, *25*, 485–520. [CrossRef]
2. Tao, J.; Liu, Y.; Yang, D. Bearing Fault Diagnosis Based on Deep Belief Network and Multisensor Information Fusion. *Shock. Vib.* **2016**, *2016*, 9306205. [CrossRef]
3. Guo, S.; Zhang, B.; Yang, T.; Lyu, D.; Gao, W. Multitask Convolutional Neural Network with Information Fusion for Bearing Fault Diagnosis and Localization. *IEEE Trans. Ind. Electron.* **2020**, *67*, 8005–8015. [CrossRef]
4. Gao, Z.; Cecati, C.; Ding, S.X. A survey of fault diagnosis and fault-tolerant techniques-part II: Fault diagnosis with model-based and signal-based approaches. *IEEE Trans. Ind. Electron.* **2015**, *62*, 3757–3767. [CrossRef]
5. Shao, X.; Wang, L.; Kim, C.S.; Ra, I. Fault Diagnosis of Bearing Based on Convolutional Neural Network Using Multi-Domain Features. *KSII Trans. Internet Inf. Syst.* **2021**, *15*, 1610–1629. [CrossRef]
6. Hearst, M.A.; Dumais, S.T.; Osuna, E.; Platt, J.; Scholkopf, B. Support vector machines. *IEEE Intell. Syst. Appl.* **1998**, *13*, 18–28. [CrossRef]
7. Quinlan, J.R. Induction of decision trees. *Mach. Learn.* **1986**, *1*, 81–106. [CrossRef]
8. Boehmke, B.; Greenwell, B. Random Forests. In *Hands-On Machine Learning with R*; CRC Press: Boca Raton, MA, USA, 2019; pp. 203–219. [CrossRef]
9. Fernández-Francos, D.; Marténez-Rego, D.; Fontenla-Romero, O.; Alonso-Betanzos, A. Automatic bearing fault diagnosis based on one-class m-SVM. *Comput. Ind. Eng.* **2013**, *64*, 357–365. [CrossRef]
10. Liu, Q.; Cheng, Y. Bearing fault diagnosis based on PCA and SVM. *Vibroengineering Procedia* **2014**, *4*, 206–210.
11. Tran, V.T.; Yang, B.S.; Oh, M.S.; Tan, A.C.C. Fault diagnosis of induction motor based on decision trees and adaptive neuro-fuzzy inference. *Expert Syst. Appl.* **2009**, *36*, 1840–1849. [CrossRef]
12. Asman, S.H.; Aziz, N.F.A.; Amirulddin, U.A.U.; Kadir, M.Z.A.A. Decision tree method for fault causes classification based on rms-dwt analysis in 275 kv transmission lines network. *Appl. Sci.* **2021**, *11*, 4031. [CrossRef]
13. Chen, Z.; Han, F.; Wu, L.; Yu, J.; Cheng, S.; Lin, P.; Chen, H. Random forest based intelligent fault diagnosis for PV arrays using array voltage and string currents. *Energy Convers. Manag.* **2018**, *178*, 250–264. [CrossRef]
14. Aldrich, C.; Auret, L. Fault detection and diagnosis with random forest feature extraction and variable importance methods. In Proceedings of the 13th Symposium on Automation in Mining, Mineral and Metal Processing, IFAC, Cape Town, South Africa, 2–4 August 2010.
15. Xie, J.; Zhang, L.; Duan, L.; Wang, J. On cross-domain feature fusion in gearbox fault diagnosis under various operating conditions based on Transfer Component Analysis. In Proceedings of the 2016 IEEE International Conference on Prognostics and Health Management, Ottawa, ON, Canada, 20–22 June 2016.
16. Shao, X.; PU, C.; ZHANG, Y.; KIM, C.S. Domain Fusion CNN-LSTM for Short-Term Power Consumption Forecasting. *IEEE Access* **2020**, *8*, 188352–188362. [CrossRef]
17. Lecun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
18. Chen, Z.; Li, C.; Sanchez, R. Gearbox Fault Identification and Classification with Convolutional Neural Networks. *Shock Vib.* **2015**, *2015*, 390134. [CrossRef]
19. Wen, L.; Li, X.; Gao, L.; Zhang, Y. A New Convolutional Neural Network-Based Data-Driven Fault Diagnosis Method. *IEEE Trans. Ind. Electron.* **2018**, *65*, 5990–5998. [CrossRef]
20. Shao, S.; McAleer, S.; Yan, R.; Baldi, P. Highly Accurate Machine Fault Diagnosis Using Deep Transfer Learning. *IEEE Trans. Ind. Inform.* **2019**, *15*, 2446–2455. [CrossRef]
21. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015-Conference Track Proceedings, San Diego, CA, USA, 6–9 May 2015; pp. 1–14.
22. Zhang, W.; Peng, G.; Li, C.; Chen, Y.; Zhang, Z. A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals. *Sensors* **2017**, *17*, 425. [CrossRef]
23. Li, Y.; Wang, N.; Shi, J.; Liu, J.; Hou, X. Revisiting Batch Normalization for Practical Domain Adaptation. In Proceedings of the International Conference on Learning Representations, Palais des Congrès Neptune, Toulon, France, 24–26 April 2017; pp. 441–446.
24. Jiang, G.; He, H.; Yan, J.; Xie, P. Multiscale Convolutional Neural Networks for Fault Diagnosis of Wind Turbine Gearbox. *IEEE Trans. Ind. Electron.* **2019**, *66*, 3196–3207. [CrossRef]
25. Shao, X.; Soo Kim, C.; Geun Kim, D. Accurate Multi-Scale Feature Fusion CNN for Time Series Classification in Smart Factory. *Comput. Mater. Contin.* **2020**, *65*, 543–561. [CrossRef]
26. Liu, Z.; Wang, H.; Liu, J.; Qin, Y.; Peng, D. Multitask Learning Based on Lightweight 1DCNN for Fault Diagnosis of Wheelset Bearings. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 3501711. [CrossRef]
27. Brusa, E.; Delprete, C.; Di Maggio, L.G. Deep transfer learning for machine diagnosis: From sound and music recognition to bearing fault detection. *Appl. Sci.* **2021**, *11*, 1663. [CrossRef]
28. Yao, Y.; Ge, D.; Yu, J.; Xie, M. Model-Based Deep Transfer Learning Method to Fault Detection and Diagnosis in Nuclear Power Plants. *Front. Energy Res.* **2022**, *10*, 823395. [CrossRef]

29. Lu, W.; Liang, B.; Cheng, Y.; Meng, D.; Member, S.; Yang, J. Deep Model Based Domain Adaptation for Fault Diagnosis. *IEEE Trans. Ind. Electron.* **2017**, *64*, 2296–2305. [CrossRef]
30. Wen, L.; Gao, L.; Li, X. A New Deep Transfer Learning Based on Sparse Auto-Encoder for Fault Diagnosis. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *49*, 136–144. [CrossRef]
31. Zhu, J.; Chen, N.; Shen, C. A New Deep Transfer Learning Method for Bearing Fault Diagnosis Under Different Working Conditions. *IEEE Sens. J.* **2020**, *20*, 8394–8402. [CrossRef]
32. Wang, X.; He, H.; Li, L. A Hierarchical Deep Domain Adaptation Approach for Fault Diagnosis of Power Plant Thermal System. *IEEE Trans. Ind. Inform.* **2019**, *15*, 5139–5148. [CrossRef]
33. Sun, B.; Feng, J.; Saenko, K. Return of frustratingly easy domain adaptation. In Proceedings of the 30th AAAI Conference on Artificial Intelligence, AAAI 2016, Phoenix, AZ, USA, 12–17 February 2016; pp. 2058–2065.
34. Sun, B.; Saenko, K. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In Proceedings of the Computer Vision–ECCV 2016 Workshops, Amsterdam, The Netherlands, 8–16 October 2016; pp. 443–450. [CrossRef]
35. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [CrossRef]
36. Munir, K.; Elahi, H.; Ayub, A.; Frezza, F.; Rizzi, A. Cancer diagnosis using deep learning: A bibliographic review. *Cancers* **2019**, *11*, 1235. [CrossRef]
37. Shorfuzzaman, M.; Masud, M. On the Detection of COVID-19 from Chest X-Ray Images Using CNN-based Transfer Learning. *Comput. Mater. Contin.* **2020**, *64*, 1359–1381. [CrossRef]
38. Le Guennec, A.; Malinowski, S.; Tavenard, R. Data Augmentation for Time Series Classification using Convolutional Neural Networks. In Proceedings of the ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data, Riva Del Garda, Italy, 12–23 September 2016.
39. Liu, S.; Ji, H.; Wang, M.C. Nonpooling convolutional neural network forecasting for seasonal time series with trends. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 2879–2888. [CrossRef]
40. Shao, X.; Kim, C.-S.; Sontakke, P. Accurate Deep Model for Electricity Consumption Forecasting Using Multi-Channel and Multi-Scale Feature Fusion CNN–LSTM. *Energies* **2020**, *13*, 1881. [CrossRef]
41. Agarap, A.F. Deep Learning using Rectified Linear Units (ReLU). *arXiv* **2018**, arXiv:1803.08375.