mathematics

MDPI

Article

Improved Conditional Domain Adversarial Networks for Intelligent Transfer Fault Diagnosis

Haihua Qin [1,2], Jiafang Pan [1,2,*], Jian Li [1,2] and Faguo Huang [1,2]

[1] Key Laboratory of Advanced Manufacturing and Automation Technology (Guilin University of Technology), Education Department of Guangxi Zhuang Autonomous Region, Guilin 541006, China
[2] Guangxi Engineering Research Center of Intelligent Rubber Equipment (Guilin University of Technology), Guilin 541006, China
* Correspondence: panjiafang@glut.edu.cn

Abstract: Intelligent fault diagnosis encounters the challenges of varying working conditions and sample class imbalance individually, but very few approaches address both challenges simultaneously. This article proposes an improvement network model named ICDAN-F, which can deal with fault diagnosis scenarios with class imbalance and working condition variations in an integrated way. First, Focal Loss, which was originally designed for target detection, is introduced to alleviate the sample class imbalance problem of fault diagnosis and emphasize the key features. Second, the domain discriminator is improved by the default ReLU activation function being replaced with Tanh so that useful negative value information can help extract transferable fault features. Extensive transfer experiments dealing with varying working conditions are conducted on two bearing fault datasets with the effect of class imbalance. The results show that the fault diagnosis performance of ICDAN-F outperforms several other widely used domain adaptation methods, achieving 99.76% and 96.76% fault diagnosis accuracies in Case 1 and Case 2, respectively, which predicts that ICDAN-F can handle both challenges in a cohesive manner.

Keywords: fault diagnosis; transfer learning; domain adaptation; conditional domain adversarial network; class imbalance

MSC: 68T01

Citation: Qin, H.; Pan, J.; Li, J.; Huang, F. Improved Conditional Domain Adversarial Networks for Intelligent Transfer Fault Diagnosis. *Mathematics* **2024**, *12*, 481. https://doi.org/10.3390/math12030481

Academic Editors: Heung Soo Kim, Salman Khalid, Ananda Shankar and Prashant Kumar

Received: 9 January 2024
Revised: 28 January 2024
Accepted: 31 January 2024
Published: 2 February 2024

1. Introduction

Monitoring the operational status of rotating machines has been highly emphasized in the field of engineering, and one of the key issues is the need for timely and effective diagnosis and feedback of faults occurring in the machines [1] to reduce economic losses and potential personal hazards. Deep learning (DL) and measurement technology (MT) have made remarkable achievements in the recent past; therefore, many scholars have combined these techniques to conduct in-depth research on fault diagnosis (FD) as a practical engineering problem [2–4]. The DL-based FD, which allows one to learn complex input-to-output functions directly from data, is considered a promising solution in today's industrial data volume [5]. It reduces human intervention compared to traditional diagnostic methods that need to be combined with signal processing techniques.

However, a machine in reality often operates under conditions with differences (e.g., speed, temperature, load, etc.), and the application of DL-based FD methods will inevitably produce domain shift phenomena due to these differences, i.e., there is an inconsistent distribution that leads to significant diagnosis performance deterioration between the data that were originally learned and the data that are about to be diagnosed [6], as shown in Figure 1.

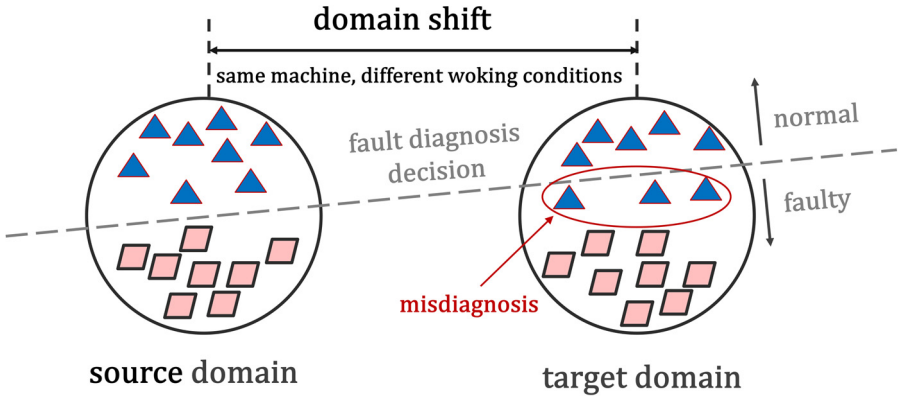


Figure 1. Adverse effect of domain shift.

Moreover, in most similar studies, the DL-based FD methods would assume that the probability of occurrence is balanced and the same between different data samples, but this is contrary to the situation where the normal state is more in real industrial operation. In other words, as shown in Figure 2, class imbalance can also discredit DL-based FD because the model is inclined to be over-trained by the classes with more data samples, while classes with fewer data samples will be squeezed by the decision boundary [7]. Thus, there is an urgent need to study FD methods that are closer to real-world scenarios where there are working differences and data class imbalance.

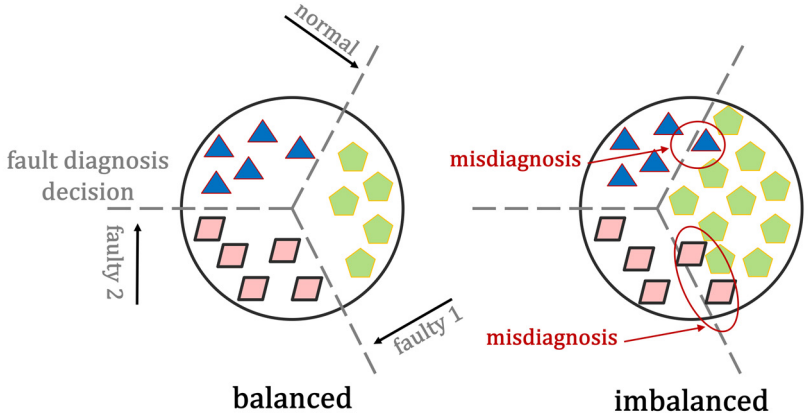


Figure 2. Decision squeeze from class imbalance.

It has been widely reported that many scholars in the field of FD have also realized the limitations of the above DL and achieved exciting research results using the transfer learning (TL) approach. In TL-based FD methods, data with distributional differences are divided into source and target domains and attempt to eliminate the differences between the two domains as much as possible during model training to accomplish domain adaptation (DA). DA aims to map the data to a common feature space and then extracts the transferable features that follow similar distributions across the domains [8] and ultimately diagnoses the brand-new data in the target domains. DA is a much-studied TL subtopic, and in the recent past, its research focus has been centered on two mechanisms: statistical metric-based versus adversarial-based [9].

For the former, representative techniques such as adaptive batch normalization (AdaBN) [10], maximum mean difference (MMD) [11], and correlation alignment (CORAL) [12] have been applied to FD. Zhang [13] et al. obtained statistical information in the target domain to be fed back to the global with the help of AdaBN, which narrowed the distributional differences between the rolling bearing data domains and enhanced the effectiveness of FD in different working conditions. Qian [14] et al. relied on MMD and CORAL to transfer high-level representations of different domains to the same region in the subspace

to design new paradigm metrics and constructed parameter-sharing CNNs to optimize the difference in distributions obtained by the metrics to cope with the problem of bearing data domain shift in different machines FD. The limitation of these methods, however, is that they require human design or improvement in the measure of distribution distance, but human design can bias the effect of DA between data. In fact, when the distribution characteristics in DA cannot be calculated or are difficult to obtain, model networks that rely on data interpretation cannot be led to learn [15].

The second mechanism, which seems to have gained more attention recently, is inspired by the unique idea of adversarial games derived from generative adversarial networks (GAN) [16] and was initially proposed to be applied in computer vision for generating fake-to-fake images. The adversarial domain adaptation (ADA) strategy focuses more on robustly and adaptively extracting features that are invariant between two domains by constructing a domain adversarial neural network (DANN) [17] rather than just correcting the distribution. Li [18] et al. used DANN to optimize FD under altered operating conditions, without manually designing distance metrics and also achieved good FD results. Furthermore, Long [19] et al. pointed out that the ADA strategy lacks the bundling relationship between search features and classes, which is not conducive to capturing the multimodal structure behind data distribution. Specifically, the marginal distribution is the absolute sole protagonist in ADA, but there is still room for improvement in the conditional distribution in practice. With this in mind, Yu [20] et al. introduced the conditional adversarial domain adaptation (CADA) strategy to their FD task, which uses conditional domain adversarial networks (CDAN) to force the adversarial process to acquire features with intra-class compactness and inter-class separability, and experiments proved that the optimization effect is robust.

Nevertheless, there are two unexplored issues with these excellent works. One is that in the FD process of domain-adaptive DA, the effects of the importance of the class and the number of samples within the class are ignored, and the samples are considered equally important when they enter the network, implying that the fault features are as valuable as the normal features and that the class with a large number of samples will receive more attention. Secondly, in ADA or CADA, domain discriminators often have to be designed more fragile, as a strong discriminator is not conducive to feature extractor (generator) optimization [21]. This will derive the need to rationalize and improve the fragile domain discriminators to obtain more robust and extensive information to guide the DA process.

Responding to the two issues mentioned above, this article proposes a new FD method that improves the CDAN and can be used under different working conditions with class imbalance data, which is referred to as ICDAN-F. Its main contributions are as follows:

1. An improved conditional domain adversarial network model for FD, ICDAN-F, is proposed. Unlike previous methods that focus only on a single challenge, it is able to address both challenges of class imbalance and working condition variation in FD.
2. The innovation is to introduce Focal Loss as a constraint in the training phase of ICDAN-F. This loss function can effectively deal with the problems of class imbalance and feature focus inconsistency, and with low computational cost, it can be easily applied to the network to optimize the FD model.
3. Replacing the ReLU activation function of the domain discriminator in conditional domain adversarial networks with Tanh is another improvement. This modification enables the negative value information to participate in the domain adaptation process and better extracts those metastable fault features that are not affected by changes in working conditions.
4. The feasibility of ICDAN-F and the effectiveness of the improvement are verified by a large number of bearing fault experiments. Compared with similar methods that address similar single challenges, ICDAN-F has an advantage in the accuracy of bearing fault diagnosis. The experimental results show that ICDAN-F can accomplish FD under the intertwining of class imbalance and working condition changes.

For the remaining sections, Section 2 presents the necessary assumptions of this research and the relevant theorems for building our proposed method. Experimental details and analysis of results will be placed in Section 3. Section 4 is a discussion and Section 5 will summarize the whole article.

2. Materials and Methods

2.1. Problem Definition and Assumptions

It is hoped that enough knowledge for FD can be learned from the existing working condition data (source domain) to be transferred to the data that are available and generates the working condition variation (target domain). To facilitate understanding of the core of this work, some basic assumptions are presented here:

(1) The source data $D_s = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$ and target domain data $D_t = \{(x_j^t)\}_{j=1}^{N_t}$ come from the same device. These two domains are label-consistent, meaning the fault types in the source domain are fully present in the target domain. Additionally, there is an imbalance as fault class data are lower in quantity compared to the normal class.

(2) The source domain samples x^s, sample size of N_s, are very sufficient subject to learning, i.e., the fault types (labels) y_i^s can be labeled, and obey a probability distribution $C_p = P(x_i^s)$.

(3) The target domain samples x^t and sample size of N_t, are recently obtained, and their labels are not yet known, as they are on another distribution $C_q = Q\left(x_j^t\right)$, $(C_p \neq C_q)$.

(4) Expect that there is a method to build a neural network model $M(\cdot)$ capable of accomplishing the FD task $\hat{y} = M(x)$ of the target domain data without trying to access the labels of the target domain and only via the source domain knowledge already learned.

2.2. Domain Adversarial Neural Network

As previously summarized, DANN has become a popular and powerful method to minimize domain differences by means of adversarial training [22]; it aims to locate a representation space where the two domains are indistinguishable but where samples from the source domain can still be correctly classified. This method is inspired by generative adversarial networks (GANs), which internally generate an adversarial process in which the generator learns to keep confusing the discriminator while the discriminator strives not to be duped and ultimately seeks a Nash equilibrium [23]. DANN is two components simulating a min–max game, where the domain discriminator D is trained to distinguish whether a feature belongs to the source or the target domain, while the other component, the feature generator (extractor) with shared parameters, F, is trained to maximize the loss aiming to fool D. To emphasize, a source classifier C, is required, and the loss of C is minimized in order to guarantee a small classification error.

The structure of DANN is shown in Figure 3. During runtime, the source domain samples are first utilized to calculate L_C via cross-entropy (CE) loss as follows:

$$L_{\mathrm{C}} = -\mathbb{E}_{(x_i^s, y_i^s)\sim \mathcal{D}_s} \sum_{c=1}^{n_c} 1_{[y_i^s = c]} \log[C_c(F(x_i^s))] \tag{1}$$

where n_c is the number of classes, $C_c(\cdot)$ is the likelihood that the classifier C think input is predicted to be class c, $F(x_i^s)$ is the output of the feature extractor F, $\mathbb{E}$ is the mathematical expectation, x_i^s is the i-th sample from source domain, y_i^s is the label corresponding to it, and **1** is the indicator function. Next, using the limited information about the target domain, the stimulus D identifies the domain labels of the current features as well as directs F to extract domain-invariant features.

The domain label 0 or 1 corresponds to source or target domains, and L_D generated by D can be computed according to the binary cross-entropy (BCE) loss as F is continuously directed:

$$L_D = -\mathbb{E}_{x_i^s \sim \mathcal{D}_s} \log[D(F(x_i^s))] - \mathbb{E}_{x_j^t \sim \mathcal{D}_t} \log\left[1 - D\left(F(x_j^t)\right)\right] \tag{2}$$

where $D(\cdot)$ is the likelihood that the computation will be predicted to have domain label 0.

Then, the optimization objective of the whole structure is to expect the total feedback L_{DANN} to be minimal. Specifically, F and C are updated to shrink L_C for better classifications while expanding to constrain F to find domain-invariant features that can lie to D:

$$L_{DANN} = L_C - \lambda_{DANN} L_D \tag{3}$$

where λ_{DANN} is a parameter for adjusting weights.

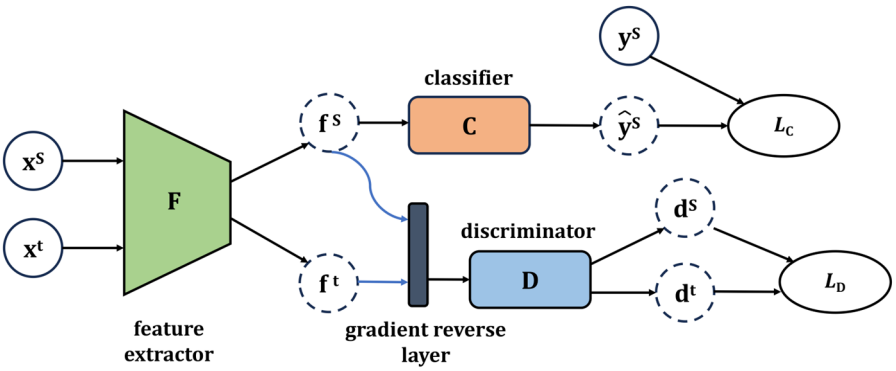


Figure 3. Structure of the DANN.

2.3. Conditional Domain Adversarial Network

CADA sought to pull different in-class features closer together separately, reducing conditional distribution differences explicitly, which drove the creation of conditional domain adversarial networks (CDAN) [19]. It aims to ameliorate the problem that exists with the use of DANN: the alignment of feature distributions within the source or target domains may not be secured even when the discriminator has successfully lied.

The structures of the two networks are much the same and have the same overall goal. What is different is that CDAN conditionalizes the classifier prediction during the optimization of the domain discriminator. That is, the inputs to D are now modified according to Equation (4) to receive information containing categories to better align features of the same class, regardless of the domain they come from.

$$h = F(x) \otimes C(F(x)) \tag{4}$$

where $\otimes$ is a multilinear mapping operation that captures the multiplicative interaction between the two as well as multimodal information about the distribution. In addition, D will now be guided by Equations (5)–(7) to focus on how to ignore hard-to-identify samples and improve the transferability.

$$g = C(F(x)) \tag{5}$$

$$H(g) = -\sum_{c=1}^{n_c} g_c \log g_c \tag{6}$$

$$\omega(H(g)) = 1 + e^{-H(g)} \tag{7}$$

The $L_{D'}$ loss needs to be calculated during training as follows:

$$L_{D'} = -\mathbb{E}_{x_i^s \sim D_s} \omega(H(g_i^s)) \log[D(h_i^s)] - \mathbb{E}_{x_j^t \sim D_t} \omega(H(g_j^t)) \log\left[1 - D(h_j^t)\right] \tag{8}$$

There is no change in the optimization of the overall objective:

$$L_{CDAN} = L_C - \lambda_{CDAN} L_{D'} \tag{9}$$

2.4. ICDAN-F for Fault Diagnosis

The structure of the proposed ICDAN-F, which will be used for FD, is demonstrated here, as shown in Figure 4. Completely consistent with the CDAN's structure, it contains the three components necessary to accomplish unsupervised CADA: the feature generator (extractor) F is served by a convolutional neural network (CNN); behind F is connected the fully connected layer (FC or Linear) acting as the classifier C, and the domain discriminator D is the simplest multilayer perceptron (MLP).

Overall, the optimization computation process is the same as CDAN, with D being used to encourage F to extract domain-invariant features that can be transferred between the two domains. However, improvements are planned to use a new $L_{C'} = L_{Focal}$ instead of the universal Equation (1), and after exploration, it is determined that the performance of D is enhanced by applying the Tanh activation function. Next, some effort is devoted to elaborating the details of this. Now, the optimization of the overall objective is

$$L_{ICDAN-F} = L_{C'} - \lambda_{CDAN} L_{D'} \tag{10}$$

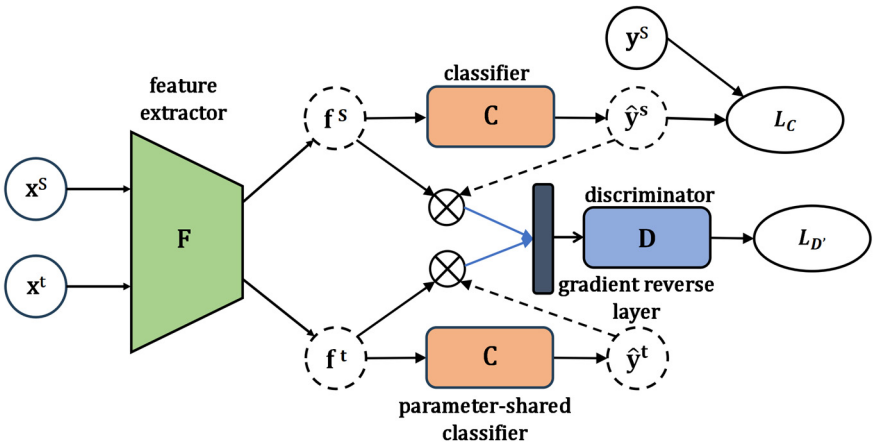


Figure 4. Structure of the CDAN or the ICDAN-F.

2.4.1. Focal Loss

Both class imbalance and sample feature emphasis are known to negatively affect the convergence of the network and the final FD performance [24,25]. In one-stage detectors, a similar situation occurs when the potentially more important foreground portion of the detected image is interfered with by a large amount of background, and the candidate proposal contains too many useless and easily classified background samples [26]; this leads to a class sample imbalance that is detrimental to training. Surprisingly, this phenomenon is optimized by Focal Loss [27], where in a bid to guide the training of network model for a detection task, samples are defined as negative and positive due to large or small data, respectively, and try to reduce the proportion of easy example loss, making the model pay more attention to the learning of hard ones [28]. Following the original derivation, the BCE loss function is expanded as in Equation (11):

$$L_{BCE} = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \tag{11}$$

where y and $\hat{y}$ are genuine label values and predicted values, respectively. For a better understanding, the BCE loss will be recast as follows in Equations (12)–(14):

$$CE(\hat{y}, y) = \begin{Bmatrix} -\log(\hat{y}), & y = 1 \\ -\log(1-\hat{y}), & otherwise \end{Bmatrix} \tag{12}$$

Define a new variable p_t:

$$p_t = \begin{Bmatrix} \hat{y}, & y = 1 \\ 1-\hat{y}, & otherwise \end{Bmatrix} \tag{13}$$

Now, BCE loss is rewritten to be more concise and clearer:

$$L_{BCE} = CE(p_t) = -\log(p_t) \tag{14}$$

It is easy to promote multi-classification; the CE loss function expansion for multi-classification is derived as shown in Equation (15):

$$L_{CCE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} 1_{[y_i=c]} \log(p_{i,c}) \tag{15}$$

where $p_{i,c}$ is the probability matrix that the i-th sample is predicted to be in class c, C is the number of classes, and N is the number of samples.

The CE loss does not distinguish between the samples in terms of difficulty or focus on the presence of imbalance; therefore, focal loss is then tailored to design two parameters, γ and α, to control the impact to be faced:

$$L_{Focal} = \alpha \cdot (1-p_t)^{\gamma} \cdot L_{BCE} \tag{16}$$

Similarly, when defining a new variable, p_c, Equation (18) is obtained for multi-classification,; the difference is that α is now a vector instead of a number. As previously described, the focal loss will be applied to our method, i.e., $L_{C'} = L_{Focal}$.

$$p_c = \begin{Bmatrix} \hat{y}_1, & y = 1 \\ \cdots\cdots & \cdots\cdots \\ \hat{y}_{n_c-1}, & y = n_c - 1 \\ 1-(\hat{y}_1 + \hat{y}_2 + \ldots + \hat{y}_{n_c-1}), & y = n_c \end{Bmatrix} \tag{17}$$

$$L_{Focal} = \alpha \cdot (1-p_c)^{\gamma} \cdot L_{CCE} \tag{18}$$

where γ is the focal parameter that narrows the weights of easily classified samples, and α is the weight parameter that mitigates the imbalance problem. The setting suggested by the designer is $\gamma = 2$, and α is set freely at a ratio of 3 to 1 for positive and negative samples.

For inaccurately classified samples, the loss is unchanged, and for accurately classified samples, the loss becomes smaller. The network model will focus more on how to classify the samples that harm the network performance better rather than just how to classify by using the information provided by the majority of the samples [29]. Doing so overcomes the bad effects of class imbalance on model training while also learning hard-to-classify samples during training, ultimately boosting the classification of the model.

2.4.2. Feature Extractor

In the proposed method, two CNNs with shared parameters are deemed to constitute the feature extractor as a whole. The popularity of CNNs is largely attributed to their hierarchical feature extraction ability [30], and the core convolutional kernel operation is computed as in Equation (19):

$$Z_k = W_k * x + b_k \tag{19}$$

where Z_k is the k-th nonactivated feature map, W_k is the k-th convolution kernel, x is the input feature map of a certain convolution kernel, and b_k is the bias.

The details of the CNN paradigm have not been uniformly specified in most studies, and here, we first referred to an excellent comparative survey [31] to gather insights, and then, based on the findings from that study, we conducted a design of experiments to determine the structure and parameters of our feature extractor, F. This ensures that our design aligns with validated research results and exhibits a high level of reliability and utility. As a result, our feature extractor, F, incorporates the optimal structure and parameters derived from the study and our tuning tests. As part of our classification pipeline, we employed the fully connected layer (FC) and Softmax activation function as the classifier, C, to predict the final classification results.

Using larger convolutional kernels as shown in Table 1 (e.g., 15 kernel size) in the first layer helps the network to globally perceive larger local features. Complex and abstract features can be extracted gradually by stacking meticulous convolutional layers (e.g., 3 kernel size). Batch normalization helps to speed up training and stabilize the model, while max-pooling reduces dimensionality and improves translation invariance. A culling layer mitigates overfitting, and a fully connected layer maps features to specific classes using Softmax activation for multi-class classification. These design choices improve performance and generalization.

Table 1. Parameters of the feature extractor and the classifier.

Layer	Parameter Setting [1]	Output Size	Activation Function
Conv_1_F	k = 15, s = 1, p = 0	16 × 1010	-
BatchNormal_1_F	-	16 × 1010	ReLU
Conv_2_F	k = 3, s = 1, p = 0	32 × 1008	-
BatchNormal_2_F	-	32 × 1008	ReLU
MaxPooling_1_F	k = s = 2	32 × 504	-
Conv_3_F	k = 3, s = 1, p = 0	64 × 502	-
BatchNormal_3_F	-	64 × 502	ReLU
Conv_4_F	k = 3, s = 1, p = 0	128 × 500	-
BatchNormal_4_F	-	128 × 500	ReLU
AdaptiveMaxPooling_1_F	out = 4	128 × 4	-
Linear Layer_1_F	in = 512, out = 256	256	ReLU
Dropout_1_F	o = 0.5	256	-
Linear Layer_2_F	in = 256, out = 256	256	ReLU
Dropout_2_F	o = 0.5	256	-
Linear Layer_2_C	in = 256, out = 10	10	Softmax

[1] In Parameter Setting, "k" is kernel size, "s" is stride, "p" is padding, "in/out" is in/out features size, and "o" is odds of being dropped. In addition, the output size of C depends on the number of fault types being diagnosed.

2.4.3. Domain Discriminator

Domain discriminators D are generally designed to be very simple, which is the consensus feedback from most of the relevant research. A strong discriminator will stop updating the gradient too early, its loss converging quickly to zero; thus, no reliable path for gradient updates to the generator to guide feature generation or feature extraction [32], so the adversarial process will move in a bad direction. Therefore, following the design of some articles [17,19,31,33], a rudimentary MLP is identified initially as the D, and Table 2 has all the details of it but does not use the ReLU activation function as in their design.

Table 2. Parameters of the domain discriminator.

Layer	Parameter Setting [1]	Output Size	Activation Function
Linear Layer_1_D	in = 256, out = 1024	1024	Tanh
Dropout_1_D	o = 0.5	1024	-
Linear Layer_2_D	in = 1024, out = 1024	1024	Tanh
Dropout_2_D	o = 0.5	1024	-
Linear Layer_2_D	in = 1024, out = 1	1	Sigmoid

[1] In Parameter Setting, "in/out" is in/out feature size, and "o" is odds of being dropped.

Meanwhile, in terms of efficiently updating the parameters to guide the F to extract domain-invariant features, the use of a gradient reversal layer (GRL) is mandatory [23], the principle of which is shown in Equations (21) and (22):

$$R_\lambda(x) = x \tag{20}$$

$$\frac{dR_\lambda}{dx} = -\lambda \mathrm{I} \tag{21}$$

$$\lambda = \frac{2}{1 + e^{(-10 \cdot p)}} - 1 \tag{22}$$

where I is the matrix to be inverted, and λ is the previously occurring parameter for adjusting weights. The inversion proceeds dynamically with the ratio p of the current iteration to the total iterations.

2.5. Activation Functions

Activation functions (AF) are so essential in neural networks that they allow them to be utilized in more nonlinear problems [34]. Here, we present the computational principles of five AF used in this research.

ReLU [35] is fast and efficient to compute and is now the preferred choice for activating networks. It will directly ignore and assign zero values to negative values, thus helping to alleviate the problem of gradients that tend to vanish, which is one of the reasons why it is so fast [36].

$$ReLU(x) = \max\{0, x\} \tag{23}$$

However, negative values are sometimes meaningful, so there are other AF that refine the negative semi-axis expression of ReLU, such as LeakyReLU [37] and PReLU [38].

$$LeakyReLU(x) = \begin{cases} \alpha x & x < 0 \\ x & x \geq 0 \end{cases} \tag{24}$$

where α is the slope parameter, typically taken to be 0.01, which allows for a slight leakage of information from the negative axis rather than zeroing out.

$$\begin{aligned} PReLU(x) &= \begin{cases} \alpha_i x_i & x_i < 0 \\ x_i & x_i \geq 0 \end{cases} \\ \Delta a_i &:= \mu \Delta a_i + \epsilon \frac{\partial \mathcal{E}}{\partial a_i} \end{aligned} \tag{25}$$

where i is the present channel and α_i is an adaptive slope parameter that can receive the learning rate ϵ and momentum μ during training, and this information will determine an adaptive update of task function ε to make the activation more suitable for the nonlinear problem at hand.

Tanh [39] has been widely used in neural networks before the advent of ReLU, and due to its ability to provide enough information upon activation, its suitability for some specific tasks is outstanding even now [40].

$$Tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{26}$$

Mish [41] can be viewed as an extension of Tanh, although the former is more sensitive to information that is around the zero point.

$$Mish(x) = x \cdot Tanh(\ln(1 + e^x)) \tag{27}$$

By looking at the details in Figure 5, ReLU appears as a slanted line passing through the origin, growing linearly when the input is greater than zero, and truncating to zero in the negative region. LeakyReLU introduces a small value of negative slope in the negative region, which makes the curve have a certain output in the negative region. PReLU adapts to adjust the slope in the negative region by learning the parameters, which makes the curve more flexible in the negative region [42]. Its slope in the negative region is greater than LeakyReLU, and this green line will oscillate like a pointer in the third quadrant until it stops learning.

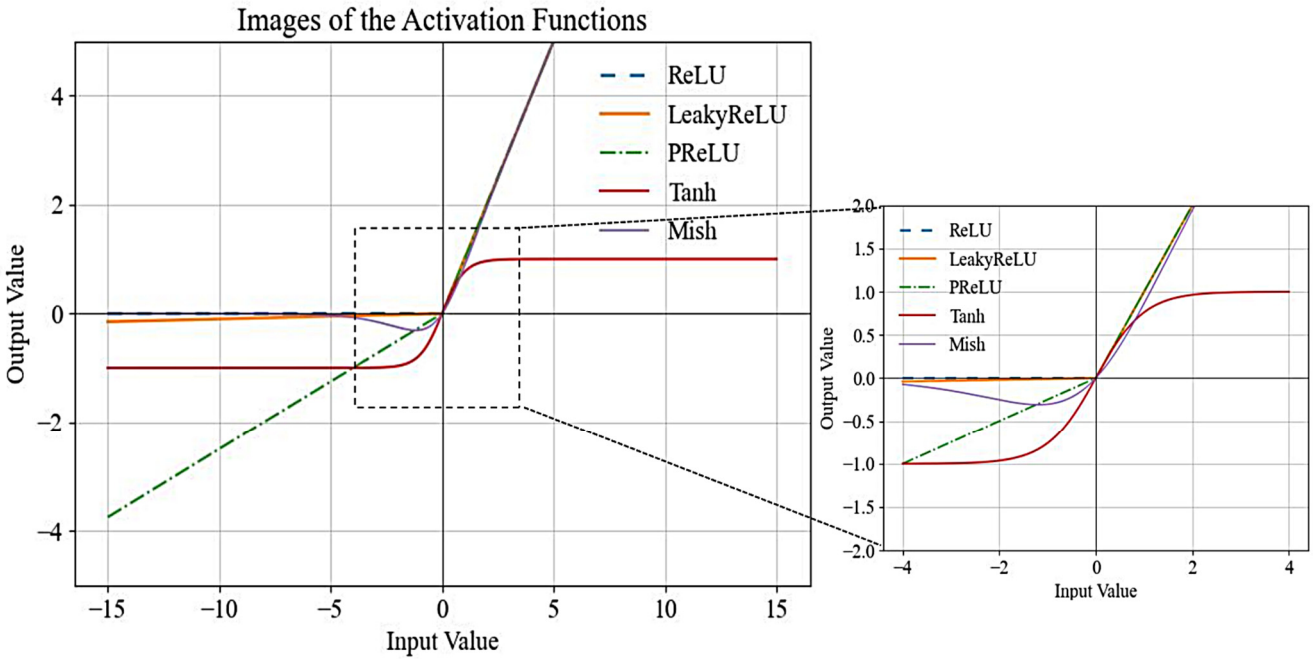


Figure 5. Image of the activation functions.

Tanh's curve shows that it can map values between −1 and 1. It has a higher slope in the region where the input is close to zero, which is good for distinguishing small feature differences, but it will saturate in the region close to the boundary. Mish also has a smooth nonlinearity, with a central localization similar to Tanh's curve, and performs well in the region close to zero, but Mish becomes smoother in the tails on the negative semi-axis and does not saturate on the positive semi-axis.

2.6. Experimental Data

Two bearing fault case datasets will be used for FD experiments. Data are normalized by $\mu - \sigma$ after being divided as samples, and no data enhancement or transformed image operations are applied. The single sample data point length is 1×1024. The details of these two datasets are illustrated below.

2.6.1. Case 1

The CWRU dataset [43,44] is widely used to validate FD methods for bearings, and the equipment used to acquire the bearing's vibration signals is shown in Figure 6. In this case dataset, the faults are artificially synthesized into the bearings employing the EDM technique.

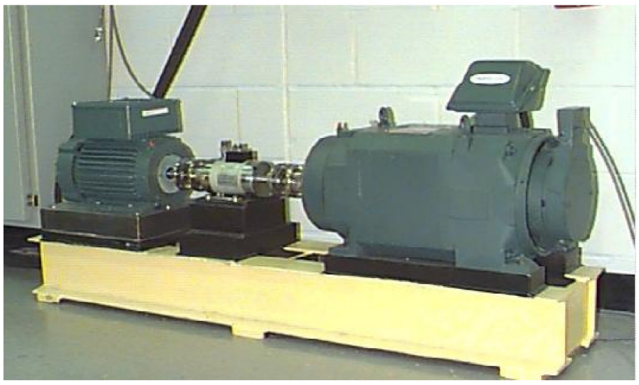


Figure 6. The CWRU equipment for collecting faults data.

Using 12 kHz drive-end bearing data, the SKF 6205 bearings had nine different single fault types. Each of the three critical components (inner ring, rolling balls, and outer ring) had three fault degrees (0.007, 0.014, and 0.021 inches) in that order. There are also normal bearing data with more adequate signal lengths, so 10 fault types in total. Another key piece of information is that the bearings for each of the 10 fault types were run separately and collected from 4 working conditions, as seen in Table 3.

Table 3. Case 1 differences in working conditions.

Working Condition Marker	A	B	C	D
Motor Load and Approx. Speed [1]	0 HP, 1797 r/min	1 HP, 1772 r/min	2 HP, 1750 r/min	3 HP, 1730 r/min

[1] Load zone centered at 6:00.

Table 4 shows the divided Case 1 data samples, which were not overlapped or rotated, and each sample 1×1024 long already contains about 2.5 turns of work data. For anyone working condition, there are many more samples of normal types than of a certain fault type, while there even is a slight decrease for some types. These phenomena are consistent with realistic imbalanced data that we are concerned with in our study.

Table 4. Case 1 experimental data.

Fault Type	Fault Degree (Inches)	Fault Label	Working Condition			
			A	B	C	D
Normal	/	0	235 [1]	470	475	475
IRF	0.007	1	120	120	120	120
	0.014	4	115	115	115	115
	0.021	7	120	120	120	120
BF	0.007	2	120	120	115	115
	0.014	5	120	120	120	120
	0.021	8	120	115	120	120
ORF [2]	0.007	3	120	120	115	120
	0.014	6	115	120	120	120
	0.021	9	120	120	120	120

[1] The value here indicates that the number of normal-type samples labeled 0 in working condition A is 235. [2] The fault location of the ORF is very close to the load center.

An image of the samples with different fault labels over time is shown in Figure 7, utilizing the Case 1 dataset run by A working condition. It enables further observation of the degrees of similarity or difference between samples.

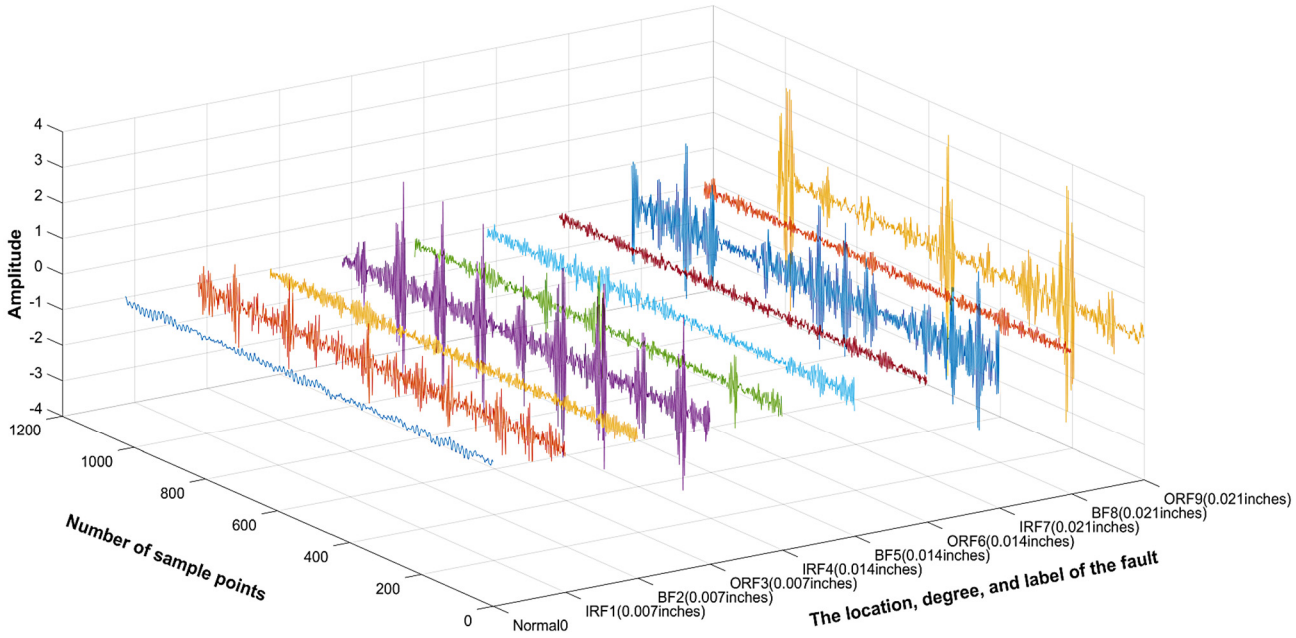


Figure 7. Changes in samples with different fault labels over time (Case 1 A).

2.6.2. Case 2

The JNU dataset [45,46], which comes from Jiangnan University, is another experimental case. The sensor PCB MA352A60 with a 50 kHz sampling frequency collected vibration signals from N205 and NU205 bearings during work. Its bearings faults were simulated inner ring breakage (IB), a certain rolling-body breakage (RB), and outer ring breakage (OB); this breakage is the same and slight, caused by man-made wire-cutting. Table 5 shows the key information for this case, with NB meaning a normal bearing that had no damages.

Table 5. Case 2 key information.

Working Condition Marker	E	F	G
Approx. Speed	600 r/min	800 r/min	1000 r/min
Bearing Type and Faults	N205: NB, OB, RB NU205 [1]: IB	N205: NB, OB, RB NU205: IB	N205: NB, OB, RB NU205: IB

[1] N or NU does not affect their identical properties, only the way the parts are separated.

Table 6 shows the data sample division. In this bearing case, the data in the normal class also far exceed the data in any one fault class and are close to the sum of all fault classes data. Figure 8 presents different data samples from the G operating condition run in Case 2.

Table 6. Case 2 experimental data.

Fault Type	Fault Degree (mm)	Fault Label	Working Condition		
			E	F	G
NB	/	10	1465 [1]	1465	1465
IB	0.3 × 0.25	11	485	485	485
RB	0.5 × 0.15	12	490	490	490
OB	0.3 × 0.25	13	490	490	490

[1] The value here indicates that the number of NB data samples labeled 10 in working condition E is 1465.

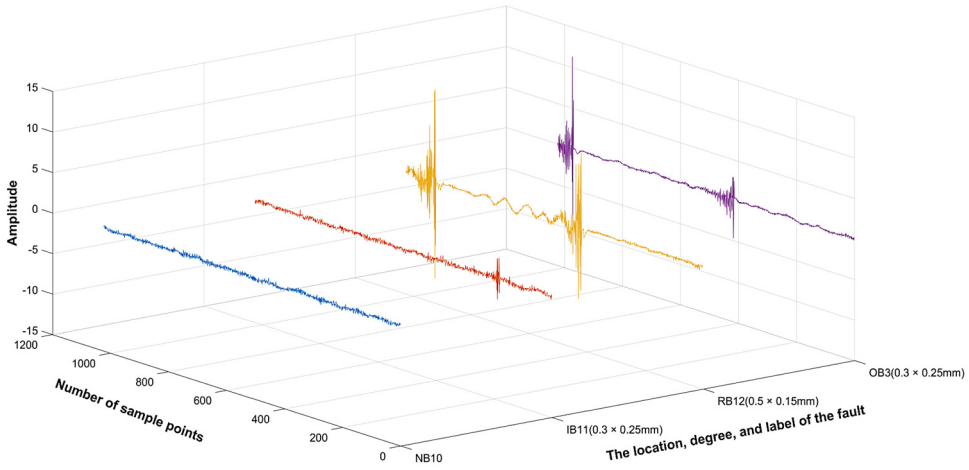


Figure 8. Changes in samples with different fault labels over time (Case 2 G).

3. Experiments and Results

In this section, the experiments require some computer hardware with deep learning tools to aid in its achievement. This necessary information is shown in Table 7.

Table 7. Necessary information for experiments.

Computer Hardware	Deep Learning Tools	Training Settings
CPU—Intel(R) Core(^{TM}) i9-10900K	System—Windows 10	Initial Learning Rate—0.001
	Language—Python 3.8	Total Iterative Epochs—300
GPU—NVIDIA Quadro P2200	Framework—PyTorch [47] 2.0.1	Batch Size—64
		Optimizer—Adam

With existing hardware, the right choice of initial learning rate, total iterations, batch size, and optimizer is crucial for optimizing model training convergence speed. After parameter debugging, the initial learning rate is set to 0.001, striking a balance between stability and speed. Generally, 300 iterations are sufficient to reduce overfitting risk and shorten convergence time. A smaller batch size provides more stochastic training and frequent gradient updates, while a larger batch size is limited by hardware memory and slows down convergence. We chose a batch size of 64, considering hardware limitations. The Adam optimizer, combining momentum and adaptive learning rate, is commonly used to achieve faster convergence during training.

The experimental flows of ICDAN-F to accomplish FD are shown in Figure 9.

- Step 1: Only the samples divided from the source domain data are input into F by batch, and the classified results are predicted by C. $L_{C'} = L_{Focal}$ is computed to guide F training.
- Step 2: No samples are input from the target domain until 50 iterations of Step 1. The reason for this is to obtain a pre-trained model that has fully learned the features of source domain samples.
- Step 3: At the 51st iteration start DA, target domain data during domain adversarial are also fed as samples to the F, which shared parameters.
- Step 4: Features output by F will be sent to D, identifying whether they belong to the source or target domain and computing $L_{D'}$ to feedback GRL. At this point, $L_{C'}$ and $L_{D'}$ jointly guide the training of domain adversarial, and the continuous updates favor F in extracting domain-invariant and transferable features and D in discriminating domains more robustly.

- Step 5: After the 150th and 250th iterations, the learning rate is adjusted to 10% and 1% of the initial value, respectively, in an attempt to seek more stable parameter updates.
- Step 6: Stop parameter updating after the iteration number is 300 and load the never-used target domain samples for testing FD results.
- Step 7: Output the FD accuracy results for this experiment.

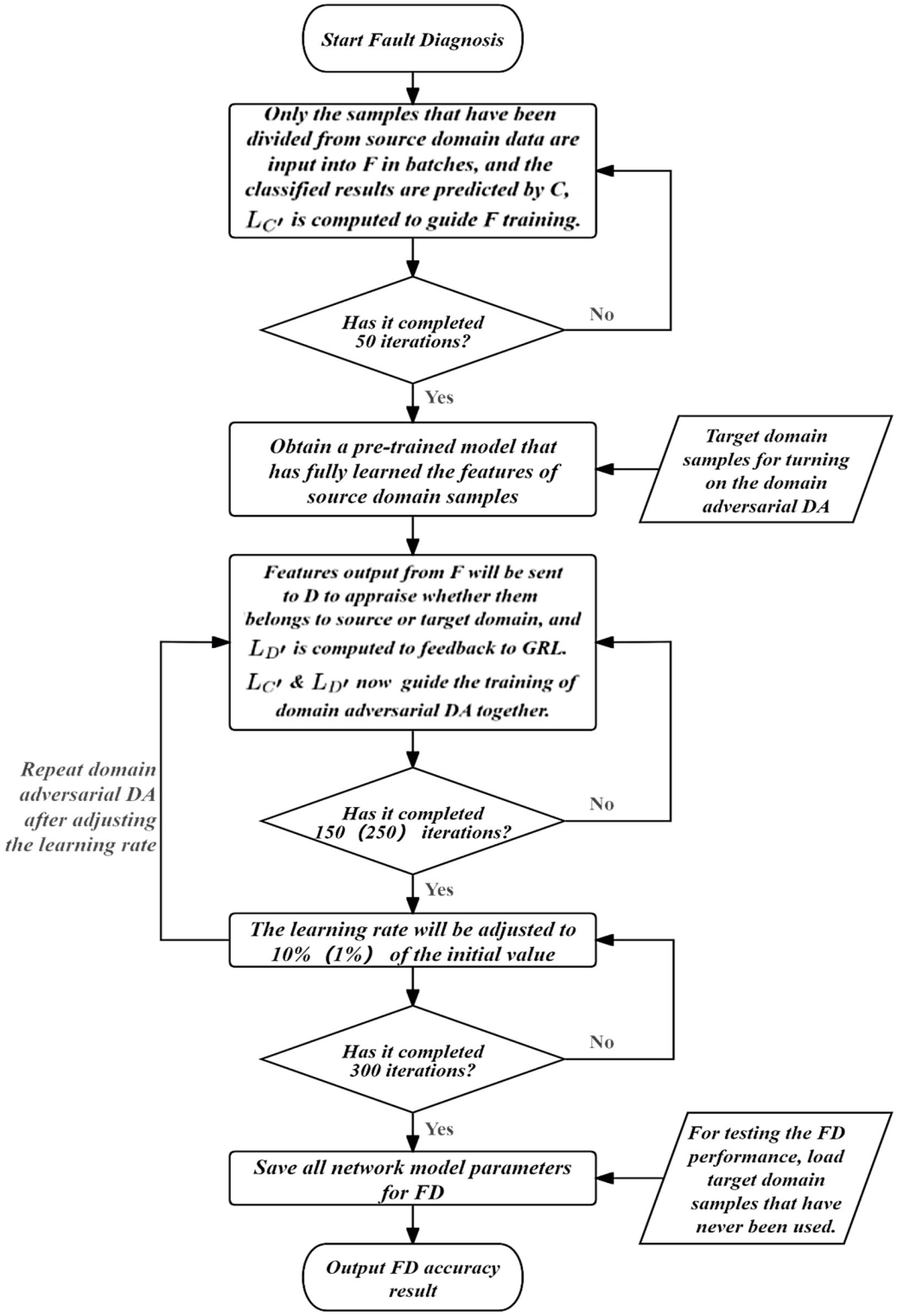


Figure 9. The experimental flows.

3.1. Experimental Verification for Handling Class Imbalance

Before conducting DA, it is necessary to verify the performance of the proposed ICDAN-F method in handling class imbalance-bearing data. Specifically, the target domain in this section represents the same working conditions as the source domain and is affected by class imbalance. The data are divided according to Table 4, only using samples from working condition C of Case 1. Overall, 70% of the samples are allocated for training the model, while the remaining part is reserved for testing the method's performance.

After 100 iterations, the accuracy, precision, recall, and F1 score metrics allow for a detailed analysis of the performance of ICDAN-F in different fault types.

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{28}$$

$$precision = \frac{TP}{TP + FP} \tag{29}$$

$$recall = \frac{TP}{TP + FN} \tag{30}$$

$$F1_{\_score} = 2 \times \frac{precision \times recall}{precision + recall} \tag{31}$$

where TP represents the count of true positive predictions, and FN represents the count of False Negative predictions.

As observed in the detailed Table 8 of one specific experiment, ICDAN-F demonstrates favorable performance in handling class imbalance fault data within the same domain. However, it incorrectly diagnoses faults 4 and 9 while accurately identifying other fault types.

Table 8. Detailed analysis of the proposed ICDAN-F.

Fault Type Label	Precision	Recall	F1_Score
0	1.00	1.00	1.00
1	1.00	1.00	1.00
2	1.00	1.00	1.00
3	1.00	1.00	1.00
4	0.97	1.00	0.99
5	1.00	1.00	1.00
6	1.00	1.00	1.00
7	1.00	1.00	1.00
8	1.00	1.00	1.00
9	1.00	0.97	0.99

Figure 10 clearly reveals the abundance of normal samples. Corresponding to the previous table, one actual fault 9 sample is erroneously diagnosed as fault 4. This observation indicates that while Focal Loss helps alleviate the impact of class imbalance, it is not flawless in its performance in this experiment.

Furthermore, the diagnostic results of ICDAN-F are being analyzed in comparison with existing state-of-the-art methods, with a particular emphasis on mitigating the impact of class imbalance. They utilize the same number of samples and iterations, and the diagnostic results of the methods are shown in Table 9.

The baseline CNN parameters are consistent with Table 1, and there is still room to improve its FD performance in the presence of class imbalance. In contrast to CNN and ICDAN-F, WGANML [48] and MFGAN [49] focus on mitigating the effects of class imbalance via generative training, balancing the number of fault samples before proceeding to FD. The average accuracy results suggest that using a generative approach to the class imbalance problem may lead to more promising results. However, this approach is computationally, time, and resource-intensive. Most importantly, ICDAN-F improves

upon the baseline by showing some degree of class imbalance mitigation and only mildly lags behind the top-performing methods. It can be concluded that ICDAN-F is feasible in dealing with class imbalance in FD.

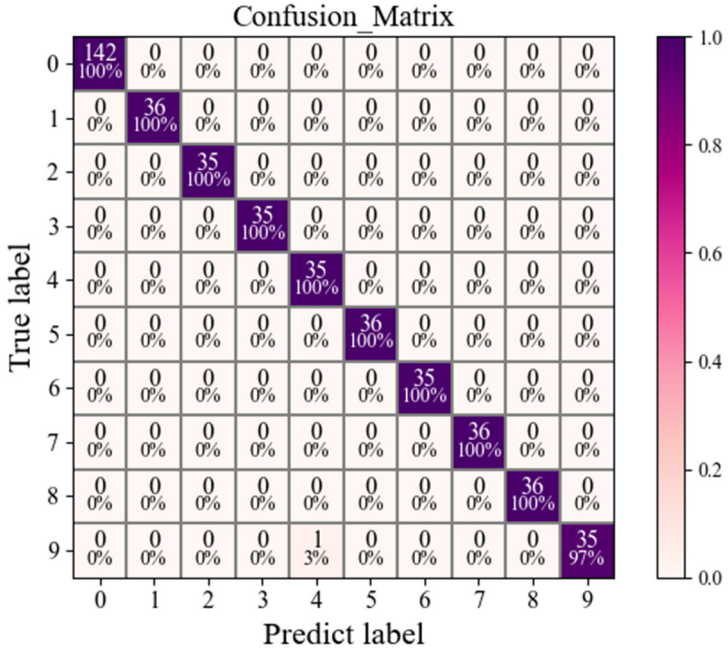


Figure 10. The confusion matrix of ICDAN-F.

Table 9. The diagnostic results of the SOTA methods under class imbalance.

Reference	Whether to Generate Fault Samples	Average Accuracies (%) [1]
CNN (Baseline)	No	98.95
WGANML [48]	Yes	99.89
MFGAN [49]	Yes	100
ICDAN-F (Ours)	No	99.74

[1] The accuracies of five replicate experiments are considered as one average accuracy.

3.2. Experimental Comparison Methods for Domain Adaptation

A brief description of the comparison method we set up to validate the sophistication of our proposed ICDAN-F. Here, it focuses on the process of DA methods, so the parameters previously formulated will not be changed. For all samples, 80% is used for training, and the rest can only be used for testing.

1. Method without DA (NoDA, baseline): Turn off D of DANN, input only N_s to train F and C, and load N_t tests directly to complete the fault diagnosis on N_t data.
2. AdaBN [10]: Similar to NoDA but will use N_t to fine-tune the statistics within the batch normalized (BN) layer of the current network according to Equation (32). The fine-tuning is updated as often as every three training iterations.

$$\begin{aligned} d &= \mu - \mu_j \\ \mu_j &\leftarrow \mu_j + \frac{dk}{n_j} \\ \sigma_j^2 &\leftarrow \frac{\sigma_j^2 n_j}{n_j+k} + \frac{\sigma^2 k}{n_j+k} + \frac{d^2 n_j k}{(n_j+k)^2} \\ n_j &\leftarrow n_j + k \end{aligned} \tag{32}$$

where j is the serial number of the neuron inside BN, n is the total number of samples counted by that neuron, and k, μ, and σ^2 are the sample quantity, mean, and variance

of this one batch, respectively. μ_j and σ_j^2 used for the first update will be hypothesized to standard Gaussian distributions. The schematic is shown in Figure 11.

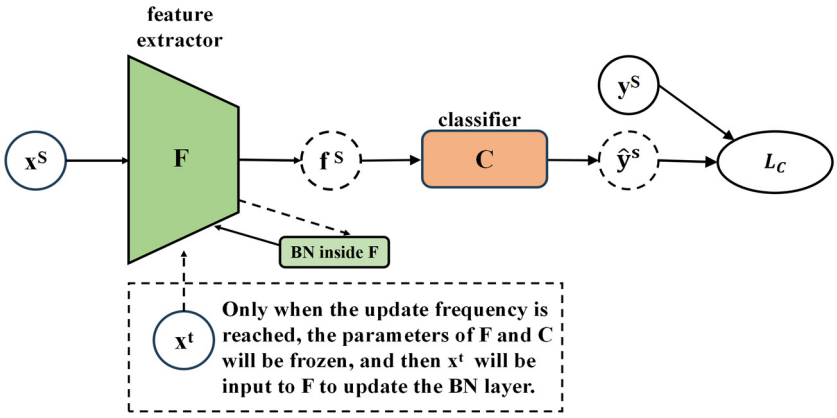


Figure 11. Schematic of the AdaBN.

3. CORAL [12]: one of the outstanding metrics used for DA, it specifically turns off the D of CDAN and computes the CORAL loss using $\hat{y}^s$ and $\hat{y}^t$ according to Equations (33)–(35) in an attempt to align the second-order statistics of the distributions C_p and C_q.

$$L_{CORAL}(x_s, x_t) = \frac{1}{4d} \parallel C^s - C^t \parallel_F^2 \tag{33}$$

$$C^s = \frac{1}{N_s - 1}[x_s^T x_s - \frac{1}{N_s}(1^T x_s)^T (1^T x_s)] \tag{34}$$

$$C^t = \frac{1}{N_t - 1}[x_t^T x_t - \frac{1}{N_t}(1^T x_t)^T (1^T x_t)] \tag{35}$$

where $||\cdot||_F$ are the F-paradigms, N_s is the sample amount in this domain, T is the transposition computation, x is the samples in this domain, and d is the absolute value of the output size of C. C^s and C^t are the covariance matrices of the D_s and D_t. Expect both L_c and L_{CORAL} to be optimized to be as small as possible. The schematic is shown in Figure 12.

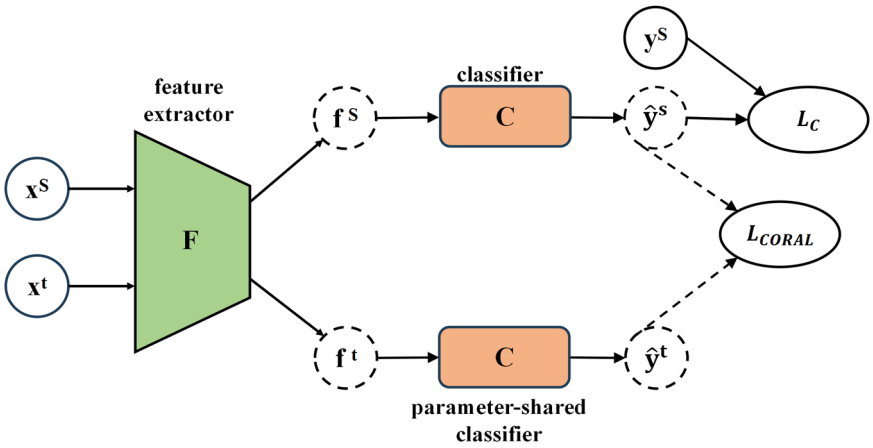


Figure 12. Schematic of the CORAL.

4. DANN [17]: following the previous Section 2.2, a domain adversarial neural network is constructed, where the structure and parameters of both F and D are consistent with the settings within the corresponding tables, but it should be noted that the commonly used ReLU is applied in D instead of Tanh.

5. CDAN [19]: The conditional domain adversarial network constructed here follows the previous Section 2.3. It has the same structure and parameter settings for F and D compared to our proposed ICDAN-F. However, the optimization constraint of CDAN is the cross-entropy loss instead of focal loss, and D internally uses the commonly used ReLU instead of Tanh.

3.3. Experimental Transfer Tasks

All samples divided according to Tables 4 and 6 will be used for the experiments. By displacing all the working conditions of a certain dataset, 18 transfer tasks are constructed, as shown in Table 10, which are designed to accomplish cross-domain fault diagnosis with fault data plagued by class imbalance.

Table 10. Experimental transfer tasks.

Task Markers	Source Domain	Target Domain	Data
$T1_{AB}$	A (0HP, 1797 r/min)	B (1HP, 1772 r/min)	*Case 1*
$T2_{AC}$	A	C (2HP, 1750 r/min)	*Case 1*
$T3_{AD}$	A	D (3HP, 1730 r/min)	*Case 1*
$T4_{BA}$	B	A	*Case 1*
$T5_{BC}$	B	C	*Case 1*
$T6_{BD}$	B	D	*Case 1*
$T7_{CA}$	C	A	*Case 1*
$T8_{CB}$	C	B	*Case 1*
$T9_{CD}$	C	D	*Case 1*
$T10_{DA}$	D	A	*Case 1*
$T11_{DB}$	D	B	*Case 1*
$T12_{DC}$	D	C	*Case 1*
$T13_{EF}$	E (600 r/min)	F (800 r/min)	*Case 2*
$T14_{EG}$	E	G (1000 r/min)	*Case 2*
$T15_{FE}$	F	E	*Case 2*
$T16_{FG}$	F	G	*Case 2*
$T17_{GE}$	G	E	*Case 2*
$T18_{GF}$	G	F	*Case 2*

The first column is the marker for the transfer task, the 1st marker through the 12th marker are associated with the Case 1 dataset, and the rest are for the Case 2 dataset. The second column is the working condition of the source domain data, the third column is the working condition of the target domain data, and the fourth column is the dataset that will be used for this task. Their designations are regular; for example, $T1_{AB}$ is the first task where the source and target data are collected under working condition A (0 HP, 1797 r/min) and working condition B (1 HP, 1772 r/min), respectively.

3.4. Experimental Results

The full results recorded are presented in Tables 11 and 12. The necessary disclaimer is that to minimize uncertainty to obtain robust results, the final diagnostic accuracies of five identical experiments are averaged and considered to be the experimental result for a task.

For Case 1's 12 tasks, observation of the results reveals that NoDA shows an average result of 93.18 as the baseline. This indicates that FD across different working conditions faces difficulties when DA is not employed and there is still room for improvement. Overall, the results achieved in the experiment by the five different DA methods used are acceptable as they do improve the domain displacement problem faced, yielding a higher overall average accuracy than the baseline. We included two statistical metric-based DAs, AdaBN and CORAL, but they only bring a slight improvement; they achieved 93.41 and 95.14 average results, respectively. The adversarial-based DA are stunning, with 99.64 and 99.19 average results for the DANN and CDAN methods, respectively, and our ICDAN-F method takes it a step further to achieve the highest of 99.76. ICDAN-F completes FD within Case 1's eight

tasks with perfect performance, which is not possible with other comparison methods. It validates that our improved method is both correct and effective.

Table 11. Experimental results of Case 1.

Task \ Method	NoDA	AdaBN [10]	CORAL [12]	DANN [17]	CDAN [19]	ICDAN-F (Ours)
$T1_{AB}$	98.85 [1]	99.77	95.65	99.91	100.00	100.00
$T2_{AC}$	94.37	99.23	95.54	100.00	100.00	100.00
$T3_{AD}$	86.97	92.18	91.19	100.00	96.17	98.45
$T4_{BA}$	99.17	94.07	95.33	99.81	100.00	100.00
$T5_{BC}$	99.48	95.54	99.68	100.00	100.00	100.00
$T6_{BD}$	94.33	90.93	93.63	99.94	99.77	100.00
$T7_{CA}$	96.25	85.13	94.95	99.23	99.12	99.39
$T8_{CB}$	96.57	97.04	98.12	99.35	99.48	100.00
$T9_{CD}$	97.78	96.43	98.83	100.00	100.00	100.00
$T10_{DA}$	80.61	84.60	85.75	98.39	98.70	99.31
$T11_{DB}$	82.83	89.66	95.52	99.13	97.40	99.92
$T12_{DC}$	90.51	96.32	97.53	99.86	99.62	100.00
Case 1 AVG	***93.18***	***93.41***	***95.14***	***99.64***	***99.19***	***99.76***

[1] The value of 98.85 in the table represents the average accuracy result of five repetitions of the experiments is 98.85 percent.

Table 12. Experimental results of Case 2.

Task \ Method	NoDA	AdaBN [10]	CORAL [12]	DANN [17]	CDAN [19]	ICDAN-F (Ours)
$T13_{EF}$	97.37	96.37	57.68	97.68	97.98	98.46
$T14_{EG}$	92.01	94.45	53.69	95.56	95.70	96.66
$T15_{FE}$	63.76	82.70	61.91	94.68	90.75	95.46
$T16_{FG}$	98.09	96.16	63.18	98.29	98.33	98.36
$T17_{GE}$	87.20	80.91	53.58	90.65	89.62	92.18
$T18_{GF}$	*98.19*	95.76	60.20	98.74	98.67	99.42
Case 2 AVG	***89.43***	***91.06***	***58.37***	***95.93***	***95.18***	***96.76***

For Case 2's six tasks, all DA methods performed less well than that of Case 1. This is because the differences in working conditions between RPMs are more significant in Case 2 than in Case 1, which makes the effect of statistical metric-based DA unacceptable. The average result of CORAL is 58.37, which is more than 40 less than that of NoDA, and it can be said that CORAL is difficult to metricize and completely fails to meet the DA goal in Case 2. The other four comparison methods worked in general, and similar to Case 1, this time, the adversarial-based DA is also much better. Our method also gives the highest result, 96.76, which is 0.83 and 1.58 higher than DANN and CADN, respectively. It shows that when dealing with FD with large differences in working conditions, ICDAN-F, as an improvement in CDAN, can accomplish FD better and surpass DANN.

The case average results of the two cases after employing different comparison methods for them are shown in Figure 13; there is a significant difference in the accuracy of the case average results between the two cases, which may be due to the different characteristics and task complexity of the two datasets. On the Case 1 dataset, the case results of almost all methods show a high accuracy of more than 90%, but on the Case 2 dataset, the case results of the various methods vary more. This confirms that adversarial-based domain adaptation may be more stable and adaptive.

Figure 14a shows the results for the full range of experimental tasks within Case1. Specifically for some interesting tasks, such as $T3_{AD}$ and $T10_{DA}$, large differences in working conditions (0 HP, 1797 r/min and 3 HP, 1730 r/min) caused the DA process to be carried out under hard-to-statistical metrics, and AdaBN with CORAL obtained much worse results than DANN and CDAN. However, the latter was formed by our introduction of Focal Loss and improvement in D (discriminator) to create the proposed ICDAN-F. It achieves

a performance boost of 2.54 over CDAN in $T11_{DB}$ and a similar improvement of 0.79 over DANN. Furthermore, ICDAN-F demonstrates its advanced nature via consistent enhancements in every task of Case 1. Notably, it achieves the only fully correct FD result in $T6_{BD}$, $T8_{CB}$, and $T12_{DC}$, further highlighting its superiority. Perhaps the performance of the adversarial-based DA methods is closer in the FD of Case 2, but in the three difficult tasks, $T14_{EG}$, $T15_{FE}$, and $T17_{GE}$, the results obtained with ICDAN-F are greatly improved, which can attest to our advantage.

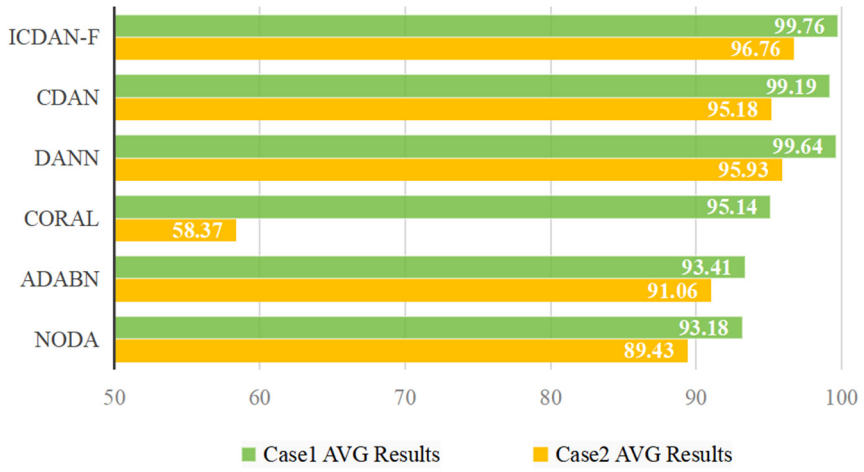
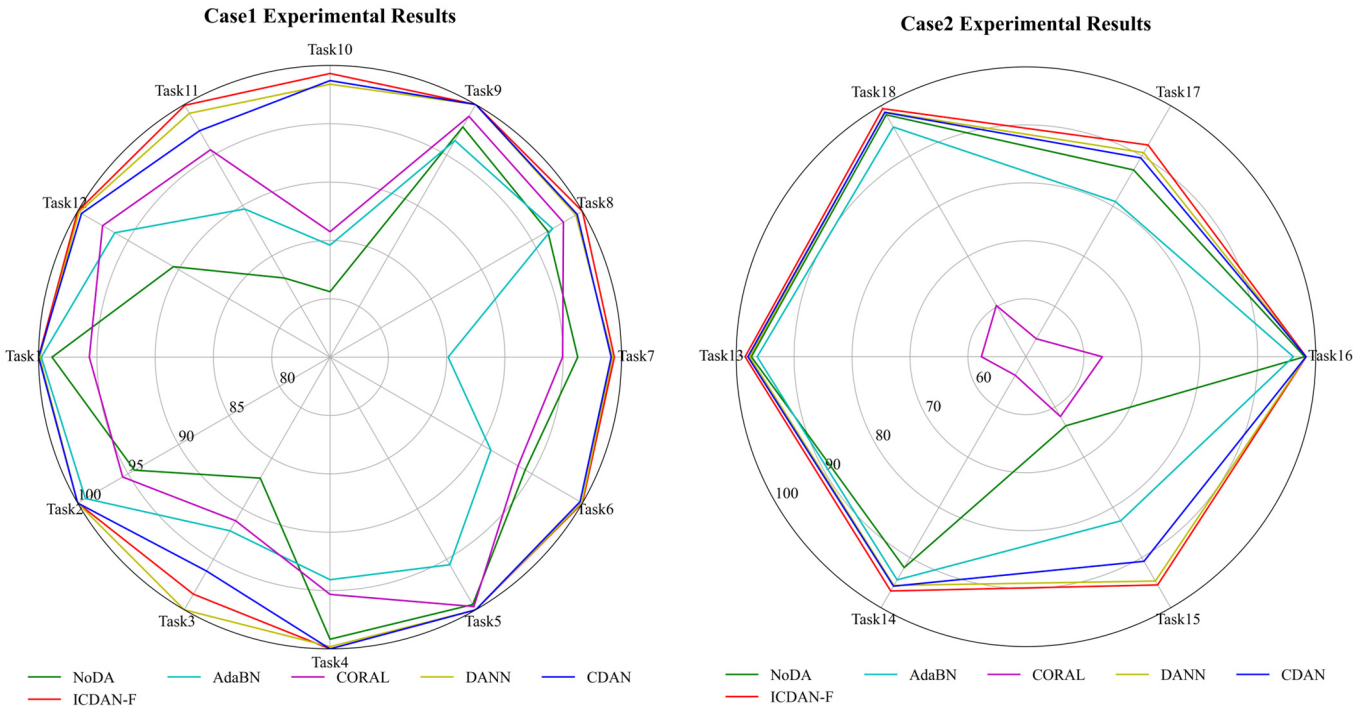


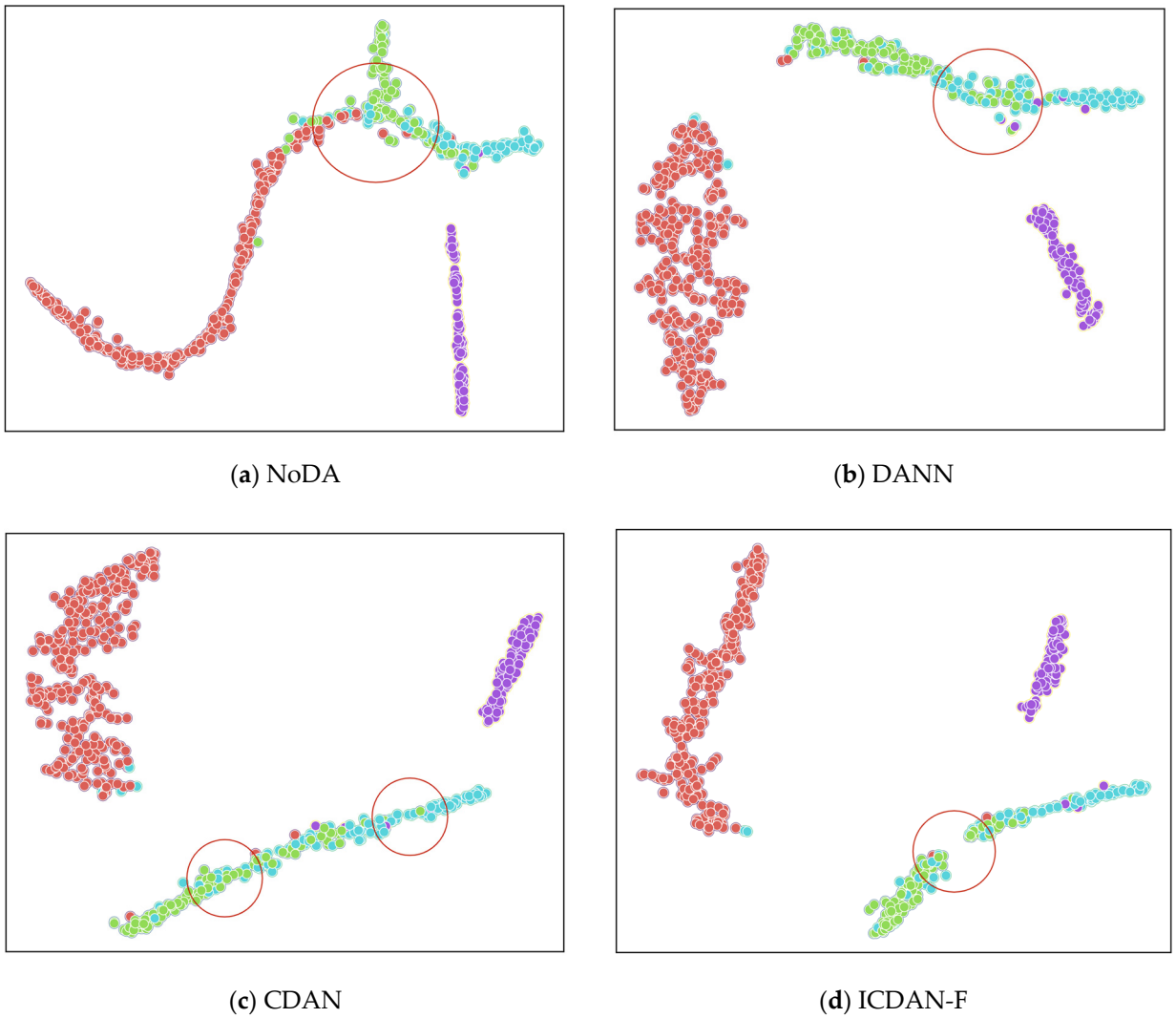
Figure 13. Case average results.



(**a**) Case 1 (**b**) Case 2

Figure 14. FD results on all tasks: (**a**) tasks attributed to Case 1; (**b**) tasks attributed to Case 2.

For the $T17_{GE}$ task, as shown in Figure 15, T-SNE reduces the dimension of the output feature vector to achieve visual results [50]. It is used to demonstrate competitive methods and the sparsity of the final diagnostic features in experiments [51].



(**a**) NoDA (**b**) DANN

(**c**) CDAN (**d**) ICDAN-F

Figure 15. T-SNE visualized the separation of diagnostic features of different methods: (**a**) NoDA (baseline); (**b**) DANN; (**c**) CDAN; (**d**) ICDAN-F.

Correspondingly, a red point in the figures represents features extracted for one sample with fault type NB and fault label 10, while similar corresponds to IB and 11 in green, RB and 12 in blue, and OB and 13 in purple. We use red circles to mark noteworthy places within the figure where, without using any DA method, as shown in Figure 15a, the three features belonging to NB10, IB11, and RB12, after inputting the network, overlap heavily and are indistinguishable. It implies that the FD of these three types of samples is very terrible.

After employing a more advanced adversarial-based DA method, DANN succeeds in keeping NB10 clustered and away from IB11 and RB12, and there is only a small amount of overlap in the visualization features of IB11 and RB12. CDAN behaves similarly to DANN, but it brings the intra-class features closer together, which is why the clustering centers of IB11 or RB12 in Figure 15c are farther apart than in Figure 15b.

The proposed ICDAN-F drives the different intra-class features all closer together, and by looking at Figure 10d can see that NB10 appears closer together compared to the previous two figures. In addition, IB11 and RB12 begin to appear perfectly separated rather than partially overlapping connections, thanks to our addition of Focal Loss, which can focus on class imbalance and emphasize the effects of the different sample features.

4. Discussion

As previously described, a proper improvement in the domain discriminator (D) may have a positive impact on the DA process, which in turn enhances the FD results. Here, we keep the structural parameters and make improvements to D using some different activation functions (AF). However, the AF of its output layer is not changed, and it will always be sigmoid since the output of D is formulated between 0 and 1. We quantitatively experimented with the five specific tasks of interest in the previous subsection for ICDAN-F. The discussion here reviews some of the exploratory processes that were carried out before determining the final ICDAN-F, and Table 13 shows the results of the FD experiments it yielded after improving D using different AF.

Table 13. Experimental results by using different AF to improve D.

AAF \ Task	$T3_{AD}$ *Case 1*	$T11_{DB}$ *Case 1*	$T14_{EG}$ *Case 2*	$T15_{FE}$ *Case 2*	$T17_{GE}$ *Case 2*	*AVG*
ReLU	98.25	98.63	95.97	93.38	90.48	***95.34***
LeakyReLU	96.89	97.53	94.99	92.25	89.11	***94.41***
PReLU	95.34	98.83	94.53	91.88	89.66	***94.05***
Mish	98.12	98.44	95.29	90.96	90.58	***94.67***
Tanh (Ours)	98.45	99.92	96.66	95.46	92.18	***96.54***

A more visual line graphical approach is used to illustrate the effectiveness of the improvement in D by using Tanh instead of ReLU, see Figure 16. Nowadays, in prevalent neural networks in which ReLU will be used as the default AF, it is necessary to consider ReLU as a baseline for the discussion. Remarkably, in these five specific tasks, CDAN obtained an AVG result of 93.92 (see Tables 11 and 12), and after comparing it with the baseline, we can tell that Focal Loss has single-handedly improved the FD performance to 95.34.

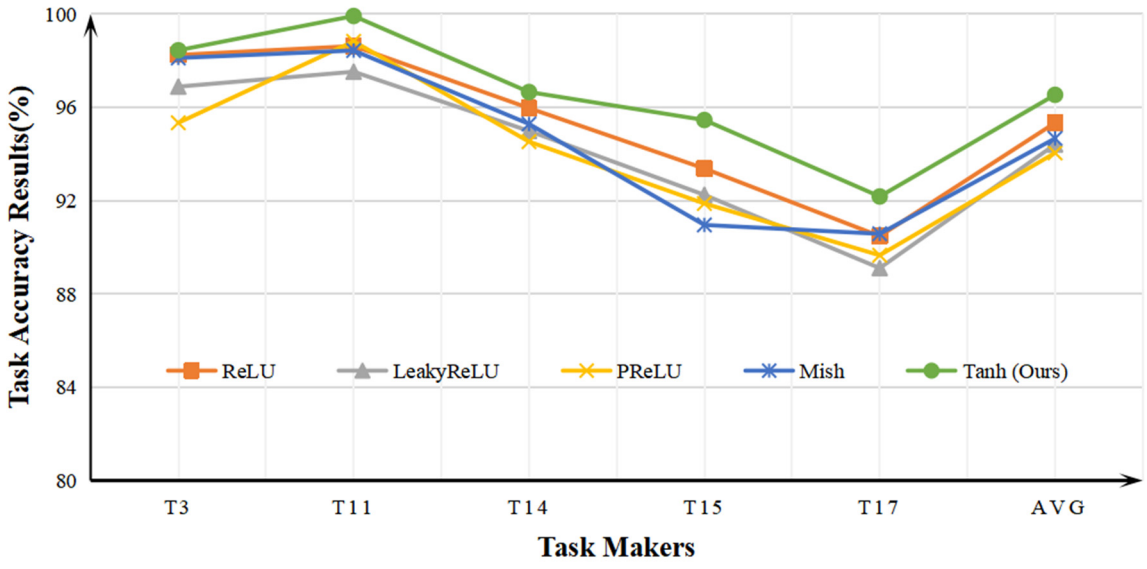


Figure 16. Results of different AFs on five specific tasks.

When ReLU was used as the AF for D, good results were presented for these five specific tasks, with an AVG result of 95.34. When LeakyReLU was used for improvement, the AVG result was 94.41, at which point we conjectured that perhaps D should not activate weak negative-axial information in FD tasks involving domain adversarial DA, which would make the FD result worse. Further, another AF that can activate negative semi-axial information by learning gradient orientation was used in D, that is, PReLU, and a result of

94.05 was obtained after the experiment. It seems that our conjecture was correct: the FD result was again weakly decreased after using the more expressive PReLU.

However, the turning point was when we employed Mish as the AF for improved D and obtained an experimental result of 94.67 that was superior to LeakyReLU and PReLU. At this point, we turned to the idea that activated negative-axis information does not inevitably harm the FD results, e.g., Mish still activated the negative-axis information, and it obtained better results than the last two AFs. It is well known that Mish's activation expression on the negative axis is smoother rather than linear, and perhaps this is its advantage.

Inspired by this, we then additionally chose Tanh to improve D, which is another smooth AF. We obtained a result of 96.54, which is an improvement of 1.20 compared to ReLU. We established that improving D by using Tanh is an approach that can outperform the baseline results, and it is certainly not the only way to improve D. But the experiments we have discussed verify that some of the negative information that flows into D is useful, and the use of a similar smoother AF may be able to improve the performance of D, which in turn leads to more satisfying FD results.

5. Conclusions

This article presents the ICDAN-F as a novel improvement to the conditional domain adversarial network for effective fault diagnosis in the presence of class imbalances and variations in working conditions. The proposed ICDAN-F incorporates Focal Loss as one of the guiding optimization constraints and leverages the Tanh activation function to improve the domain discriminator. In addressing the class imbalance, ICDAN-F exhibits a slightly inferior performance compared to two other exceptional generative solutions for fault diagnosis. Experimental results on two bearing fault datasets demonstrate the superiority of ICDAN-F over five other widely adopted domain adaptation methods, achieving fault diagnosis accuracies of 99.76 and 96.76 in Case 1 and Case 2, respectively. Furthermore, experiments verify that employing smoother activation functions, such as Tanh, enhances the discriminator's efficacy in achieving favorable fault diagnosis outcomes. A comparison between the pre-improvement (CDAN) and post-improvement (ICDAN-F) shows a substantial enhancement in fault diagnosis performance; the latter is able to address both challenges of class imbalance and working condition variation. However, it should be noted that the assessment of ICDAN-F is conducted under the assumption of label-consistent scenarios, where the source and target domains possess the same fault types. Further investigations are required to explore label-inconsistent fault diagnosis methods that can accommodate scenarios where the target domain exhibits different fault types compared to the source domain.

Author Contributions: Conceptualization, H.Q. and J.P.; methodology, H.Q.; software and formal analysis, H.Q. and J.L.; resources, J.P. and F.H.; writing—original draft preparation, H.Q.; writing—review and editing, J.P. and F.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The sources of the data used in this article are publicly available and linked in References.

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

Abbreviations

The following abbreviations are used in this manuscript:

FD	Fault diagnosis
DL	Deep learning
TL	Transfer learning
CNN	Convolutional neural network

DA	Domain adaptation
GAN	Generative adversarial network
ADA	Adversarial domain adaptation
DANN	Domain adversarial neural network
CADA	Conditional adversarial domain adaptation
CDAN	Conditional domain adversarial network
F	Feature extractor in neural network
C	Classifier in neural network
D	Domain discriminator in neural network
FC	Fully connected layer
MLP	Multilayer perceptron
ReLU	Rectified linear unit
LeakyReLU	Leaky rectified linear unit
PReLU	Parametric rectified linear unit
Tanh	Hyperbolic tangent function
Mish	A self-regularized non-monotonic neural activation function called Mish
SKF	Svenska Kullager-Fabriken
EDM	Electrical discharge machining
AVG	The average
T-SNE	T-distributed stochastic neighbor embedding

The list of symbols to be emphasized in this manuscript:

D_s	Data from the source domain	D_t	Data from the target domain
x_i^s	The i-th sample in the D_s	x_j^t	The j-th sample in the D_t
y_i^s	The label of the i-th sample x_i^s	N_t	The total number of samples from the D_t
N_s	The total number of samples from the D_s	$Q\left(x_j^t\right)$	The probability distribution followed by the target domain
$P\left(x_i^s\right)$	The probability distribution followed by the source domain	L_C	The classifier loss according to cross-entropy loss
L_D	The domain discriminator loss according to binary cross-entropy loss in DANN	$L_{D'}$	The domain discriminator loss in CDAN
$L_{C'}$	The classifier loss according to focal loss		

References

1. Li, X.W.; Lei, Y.G.; Xu, M.Z.; Li, N.P.; Qiang, D.K.; Ren, Q.B.; Li, X. A spectral self-focusing fault diagnosis method for automotive transmissions under gear-shifting conditions. *Mech. Syst. Signal Process.* **2023**, *200*, 110499. [CrossRef]
2. Zhao, M.H.; Zhong, S.S.; Fu, X.Y.; Tang, B.P.; Pecht, M. Deep Residual Shrinkage Networks for Fault Diagnosis. *IEEE Trans. Ind. Inform.* **2020**, *16*, 4681–4690. [CrossRef]
3. Sharma, S.; Tiwari, S.K. A novel feature extraction method based on weighted multi-scale fluctuation based dispersion entropy and its application to the condition of machines. *Mech. Syst. Signal Process.* **2022**, *171*, 108909. [CrossRef]
4. Habbouche, H.; Amirat, Y.; Benkedjouh, T.; Benbouzid, M. Bearing Fault Event-Triggered Diagnosis Using a Variational Mode Decomposition-Based Machine Learning Approach. *IEEE Trans. Energy Convers.* **2022**, *37*, 466–474. [CrossRef]
5. Liu, R.N.; Yang, B.Y.; Zio, E.; Chen, X.F. Artificial intelligence for fault diagnosis of rotating machinery: A review. *Mech. Syst. Signal Process.* **2018**, *108*, 33–47. [CrossRef]
6. Li, X.; Zhang, W.; Ding, Q. Cross-Domain Fault Diagnosis of Rolling Element Bearings Using Deep Generative Neural Networks. *IEEE Trans. Ind. Electron.* **2019**, *66*, 5525–5534. [CrossRef]
7. Zhang, W.; Li, X.; Jia, X.D.; Ma, H.; Luo, Z.; Li, X. Machinery fault diagnosis with imbalanced data using deep generative adversarial networks. *Measurement* **2020**, *152*, 107377. [CrossRef]
8. Li, X.; Jia, X.D.; Zhang, W.; Ma, H.; Luo, Z.; Li, X. Intelligent cross-machine fault diagnosis approach with deep auto-encoder and domain adaptation. *Neurocomputing* **2020**, *383*, 235–247. [CrossRef]
9. Li, W.H.; Huang, R.Y.; Li, J.P.; Liao, Y.X.; Chen, Z.Y.; He, G.L.; Yan, R.Q.; Gryllias, K. A perspective survey on deep transfer learning for fault diagnosis in industrial scenarios: Theories, applications and challenges. *Mech. Syst. Signal Process.* **2022**, *167*, 108487. [CrossRef]
10. Li, Y.H.; Wang, N.Y.; Shi, J.P.; Hou, X.D.; Liu, J.Y. Adaptive Batch Normalization for practical domain adaptation. *Pattern Recogn.* **2018**, *80*, 109–117. [CrossRef]

11. Gretton, A.; Borgwardt, K.M.; Rasch, M.J.; Scholkopf, B.; Smola, A. A Kernel Two-Sample Test. *J. Mach. Learn. Res.* **2012**, *13*, 723–773.
12. Sun, B.C.; Saenko, K. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *Computer Vision—Proceedings of the ECCV 2016 Workshops: Amsterdam, The Netherlands, 8–10 and 15–16 October 2016*; Part III 14; Springer International Publishing: Cham, Switzerland, 2016; pp. 443–450. [CrossRef]
13. Zhang, W.; Li, C.H.; Peng, G.L.; Chen, Y.H.; Zhang, Z.J. A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load. *Mech. Syst. Signal Process.* **2018**, *100*, 439–453. [CrossRef]
14. Qian, Q.; Qin, Y.; Luo, J.; Wang, Y.; Wu, F. Deep discriminative transfer learning network for cross-machine fault diagnosis. *Mech. Syst. Signal Process.* **2023**, *186*, 109884. [CrossRef]
15. Zhang, S.Y.; Su, L.; Gu, J.F.; Li, K.; Zhou, L.; Pecht, M. Rotating machinery fault detection and diagnosis based on deep domain adaptation: A survey. *Chin. J. Aeronaut.* **2023**, *36*, 45–74. [CrossRef]
16. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Networks. *arXiv* **2014**. [CrossRef]
17. Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; Lempitsky, V. Domain-Adversarial Training of Neural Networks. *J. Mach. Learn. Res.* **2016**, *17*, 1–35. Available online: https://dl.acm.org/doi/abs/10.5555/2946645.2946704 (accessed on 4 December 2023).
18. Li, X.; Zhang, W.; Ding, Q.; Li, X. Diagnosing Rotating Machines with Weakly Supervised Data Using Deep Transfer Learning. *IEEE Trans. Ind. Inform.* **2020**, *16*, 1688–1697. [CrossRef]
19. Long, M.; Cao, Z.; Wang, J.; Jordan, M.I. Conditional Adversarial Domain Adaptation. In Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montreal, QC, Canada, 3–8 December 2018.
20. Yu, X.L.; Zhao, Z.B.; Zhang, X.W.; Sun, C.; Gong, B.G.; Yan, R.Q.; Chen, X.F. Conditional Adversarial Domain Adaptation with Discrimination Embedding for Locomotive Fault Diagnosis. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–12. [CrossRef]
21. Roth, K.; Lucchi, A.; Nowozin, S.; Hofmann, T. Stabilizing Training of Generative Adversarial Networks through Regularization. *NeurIPS* **2017**. [CrossRef]
22. Zhu, R.H.; Jiang, X.D.; Lu, J.S.; Li, S. Cross-Domain Graph Convolutions for Adversarial Unsupervised Domain Adaptation. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 3847–3858. [CrossRef]
23. Gui, J.; Sun, Z.A.; Wen, Y.G.; Tao, D.C.; Ye, J.P. A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications. *IEEE Trans. Knowl. Data Eng.* **2023**, *35*, 3313–3332. [CrossRef]
24. Zhang, T.C.; Chen, J.L.; Li, F.D.; Zhang, K.Y.; Lv, H.X.; He, S.L.; Xu, E.Y. Intelligent fault diagnosis of machines with small & imbalanced data: A state-of-the-art review and possible extensions. *ISA Trans.* **2022**, *119*, 152–171. [CrossRef] [PubMed]
25. Yang, B.; Lee, C.G.; Lei, Y.G.; Li, N.P.; Lu, N. Deep partial transfer learning network: A method to selectively transfer diagnostic knowledge across related machines. *Mech. Syst. Signal Process.* **2021**, *156*, 107618. [CrossRef]
26. Zhao, K.; Zhu, X.Y.; Jiang, H.J.; Zhang, C.; Wang, Z.H.; Fu, B.W. Dynamic loss for one-stage object detectors in computer vision. *Electron. Lett.* **2018**, *54*, 1433–1434. [CrossRef]
27. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.M.; Dollar, P. Focal Loss for Dense Object Detection. *IEEE Trans. Pattern Anal.* **2020**, *42*, 318–327. [CrossRef]
28. Wang, Z.Y.; Xie, X.M.; Yang, J.X.; Shi, G.M. Soft focal loss: Evaluating sample quality for dense object detection. *Neurocomputing* **2022**, *480*, 271–280. [CrossRef]
29. Wang, Q.; Tian, Y.; Liu, D. Adaptive FH-SVM for Imbalanced Classification. *IEEE Access* **2019**, *7*, 130410–130422. [CrossRef]
30. Khan, A.; Sohail, A.; Zahoora, U.; Qureshi, A.S. A survey of the recent architectures of deep convolutional neural networks. *Artif. Intell. Rev.* **2020**, *53*, 5455–5516. [CrossRef]
31. Zhao, Z.B.; Zhang, Q.Y.; Yu, X.L.; Sun, C.; Wang, S.B.; Yan, R.Q.; Chen, X.F. Applications of Unsupervised Deep Transfer Learning to Intelligent Fault Diagnosis: A Survey and Comparative Study. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–28. [CrossRef]
32. Creswell, A.; White, T.; Dumoulin, V.; Arulkumaran, K.; Sengupta, B.; Bharath, A.A. Generative Adversarial Networks An overview. *IEEE Signal Proc. Mag.* **2018**, *35*, 53–65. [CrossRef]
33. Fan, J.G.; Yuan, X.F.; Miao, Z.M.; Sun, Z.H.; Mei, X.X.; Zhou, F.Y. Full Attention Wasserstein GAN with Gradient Normalization for Fault Diagnosis Under Imbalanced Data. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1–16. [CrossRef]
34. Andrea, A.; Francesco, D.; Francesco, I.; Roberto, P. A survey on modern trainable activation functions. *Neural Netw.* **2021**, *138*, 14–32. [CrossRef]
35. Nair, V.; Hinton, G.E. Rectified Linear Units Improve Restricted Boltzmann Machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; Available online: https://api.semanticscholar.org/CorpusID:15539264 (accessed on 5 December 2023).
36. Qian, S.; Liu, H.; Liu, C.; Wu, S.; Wong, H.S. Adaptive activation functions in convolutional neural networks. *Neurocomputing* **2018**, *272*, 204–212. [CrossRef]
37. Maas, A.L.; Hannun, A.Y.; Ng, A.Y. Rectifier nonlinearities improve neural network acoustic models. In Proceedings of the 30th International Conference on Machine Learning (ICML 2013), Atlanta, GA, USA, 16–21 June 2013; Available online: https://api.semanticscholar.org/CorpusID:16489696 (accessed on 5 December 2023).

38. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; Available online: https://ieeexplore.ieee.org/document/7410480 (accessed on 5 December 2023).
39. Chen, F.C. Back-propagation neural networks for nonlinear self-tuning adaptive control. *IEEE Control. Syst. Mag.* **1990**, *10*, 44–48. [CrossRef]
40. Tim, D.R.; Samuel, L.; Siddhartha, M. On the approximation of functions by tanh neural networks. *Neural Netw.* **2021**, *143*, 732–750. [CrossRef]
41. Misra, D. Mish: A self regularized non-monotonic activation function. In Proceedings of the 31st British Machine Vision Virtual Conference, Virtual, 7–10 September 2020. [CrossRef]
42. Thakur, R.S.; Yadav, R.N.; Gupta, L. Prelu and edge-aware filter-based image denoiser using convolutional neural network. *IET Image Process.* **2020**, *14*, 3869–3879. [CrossRef]
43. Smith, W.A.; Randall, R.B. Rolling element bearing diagnostics using the Case Western Reserve University data: A benchmark study. *Mech. Syst. Signal Pr.* **2015**, *64–65*, 100–131. [CrossRef]
44. Case Western Reserve University. Available online: https://engineering.case.edu/bearingdatacenter (accessed on 5 December 2023).
45. Li, K.; Ping, X.L.; Wang, H.Q.; Chen, P.; Cao, Y. Sequential Fuzzy Diagnosis Method for Motor Roller Bearing in Variable Operating Conditions Based on Vibration Analysis. *Sensors* **2013**, *13*, 8013–8041. [CrossRef]
46. Prof. Li Ke's Scopus. Available online: https://www.scopus.com/authid/detail.uri?authorId=55336879500 (accessed on 10 December 2023).
47. Installing PyTorch. Available online: https://pytorch.org/get-started/previous-versions/ (accessed on 8 December 2023).
48. Luo, P.; Yin, Z.; Yuan, D.; Gao, F.; Liu, J. An Intelligent Method for Early Motor Bearing Fault Diagnosis Based on Wasserstein Distance Generative Adversarial Networks Meta Learning. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 1–11. [CrossRef]
49. Hao, C.; Du, J.; Liang, H. Imbalanced Fault Diagnosis of Rolling Bearing Using Data Synthesis Based on Multi-Resolution Fusion Generative Adversarial Networks. *Machines* **2022**, *10*, 295. [CrossRef]
50. Xiao, Y.M.; Shao, H.D.; Han, S.Y.; Huo, Z.Q.; Wan, J.F. Novel Joint Transfer Network for Unsupervised Bearing Fault Diagnosis from Simulation Domain to Experimental Domain. *IEEE/ASME Trans. Mech.* **2022**, *27*, 5254–5263. [CrossRef]
51. Zhang, X.; He, C.; Lu, Y.; Chen, B.; Zhu, L.; Zhang, L. Fault diagnosis for small samples based on attention mechanism. *Measurement* **2022**, *187*, 110242. [CrossRef]