

## Article

# Intelligent Fault Diagnosis Across Varying Working Conditions Using Triplex Transfer LSTM for Enhanced Generalization

Misbah Iqbal <sup>1</sup>, Carman K. M. Lee <sup>1,2,\*</sup> , Kin Lok Keung <sup>1</sup>  and Zhonghao Zhao <sup>1</sup> <sup>1</sup> Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong SAR, China; misbah.iqbal@connect.polyu.hk (M.I.)<sup>2</sup> Centre for Advances in Reliability and Safety Limited (CAiRS), Hong Kong SAR, China

\* Correspondence: ckm.lee@polyu.edu.hk

**Abstract:** Fault diagnosis plays a pivotal role in ensuring the reliability and efficiency of industrial machinery. While various machine/deep learning algorithms have been employed extensively for diagnosing faults in bearings and gears, the scarcity of data and the limited availability of labels have become a major bottleneck in developing data-driven diagnosis approaches, restricting the accuracy of deep networks. To overcome the limitations of insufficient labeled data and domain shift problems, an intelligent, data-driven approach based on the Triplex Transfer Long Short-Term Memory (TTLSTM) network is presented, which leverages transfer learning and fine-tuning strategies. Our proposed methodology uses empirical mode decomposition (EMD) to extract pertinent features from raw vibrational signals and utilizes Pearson correlation coefficients (PCC) for feature selection. L2 regularization transfer learning is utilized to mitigate the overfitting problem and to improve the model's adaptability in diverse working conditions, especially in scenarios with limited labeled data. Compared with traditional transfer learning approaches, such as TCA, BDA, and JDA, which demonstrate accuracies in the range of 40–50%, our proposed model excels in identifying machinery faults with minimal labeled data by achieving 99.09% accuracy. Moreover, it performs significantly better than classical methods like SVM, RF, and CNN-based networks found in the literature, demonstrating the improved performance of our approach in fault diagnosis under varying working conditions and proving its applicability in real-world applications.



**Citation:** Iqbal, M.; Lee, C.K.M.; Keung, K.L.; Zhao, Z. Intelligent Fault Diagnosis Across Varying Working Conditions Using Triplex Transfer LSTM for Enhanced Generalization. *Mathematics* **2024**, *12*, 3698. <https://doi.org/10.3390/math12233698>

Academic Editor: Yolanda Vidal

Received: 3 October 2024

Revised: 16 November 2024

Accepted: 22 November 2024

Published: 26 November 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

**Keywords:** fault diagnosis; transfer learning; fine-tuning; deep LSTM; empirical mode decomposition**MSC:** 68TXX; 65-11

## 1. Introduction

Rotating machines are employed in various applications, from manufacturing to wind turbines, airplane engines, electric motors, marine propellants, and mining equipment [1–5]. Critical components of rotating machines often experience impact damage due to the complicated architecture and harsh working environment, which may risk personal safety and create enormous economic losses. Critical components of rotating machines often experience impact damage due to the complicated architecture and harsh working environment, which may risk personal safety and create enormous economic losses. Still, intelligent fault detection is a challenging task in both academic research and industry applications, regardless of technological development. These challenges stem from a variety of factors, including diverse operating conditions in real-world environments, varying sensor configurations (positions) across different rotating machines, and a lack of labeled faulty data for the training of diagnostic models. The complicated architecture of rotating machines and their severe operating conditions make it difficult to diagnose faults. Consequently, researchers and industry professionals are always looking for new ways to make fault diagnosis systems for rotating machinery more accurate and reliable [6,7].

Data collection, signal processing, feature extraction, reduction, and classification are the typical building blocks of defect detection. In recent years, various data-collecting methods have emerged, such as vibration analysis, acoustic emission monitoring, motor current analysis, and oil analysis. These techniques are critical for determining the machine's health and finding faults before they cause breakdowns. Because vibrations accurately represent machines' health conditions, they are frequently the first choice for data collection [3]. Sensors, for instance, accelerometers can collect vibration data as time series signal data to identify any irregularities in machine operation, such as unbalances or misalignments. Nevertheless, suitable signal processing methods are necessary for the analysis of time series vibration data due to their non-stationary and non-linear nature. Different methods are commonly employed, including time-domain analysis (statistical and autocorrelation), frequency-domain analysis (fast Fourier transform), and time-frequency-domain analysis (wavelet transform, short-time Fourier transform, Hilbert Huang transform) [8,9]. Among them, FFT is widely used but cannot effectively capture non-stationary signals. WT requires determining a suitable basis function, and using an inappropriate wavelet mother function can result in errors during diagnosis [10,11]. To address these limitations and improve fault diagnostic accuracy, this paper introduces self-adaptive empirical mode decomposition to extract useful information from non-stationary and nonlinear vibration signals.

Traditional data-driven fault diagnosis approaches such as artificial neural networks (ANNs), K-nearest neighbor (KNN), random forests (RFs), and support vector machines (SVMs) are commonly used in defect recognition [12–19]. However, these methods often lack the ability to effectively learn complex nonlinear fault features, often necessitating manual feature extraction [20]. This manual feature extraction process is time-consuming and requires in-depth expertise. The limited generalizability of handcrafted features has driven the need for novel and sensitive feature extraction methods. Deep learning (DL) has emerged as a promising solution, negating the need for manual feature extraction and the associated difficulties in learning the sequential data [1,21–23]. Most popular DL networks, including convolutional neural networks (CNNs), graph neural networks (GNNs), autoencoders (AEs), deep belief networks (DBNs), and recurrent neural networks (RNNs), capture complex patterns within the data [24–29]. To improve the efficiency of fault diagnosis, these shallow and deep architectures are modified according to specific types of data and tasks. In the realm of industrial machinery condition monitoring, LSTMs have become prominent for their use [30,31]. Their capabilities to evaluate vibration signals and other time series data from sensors have led to the successful detection of anomalies and prediction of failures. Despite their potential for defect diagnosis on rotating machines, deep learning models still face challenges on their path to full reliability.

- (1) Deep learning models generally necessitate a substantial amount of labeled data to generalize effectively. The primary problem with fault diagnosis is the challenge of gathering sufficient data on various fault conditions, as machines rarely exhibit faults during normal operation. The scarcity of data makes it difficult to train models effectively, potentially leading to overfitting (a phenomenon where models perform well on training data but underperform on new, unseen data) [32].
- (2) The expense associated with expert labeling of the acquired data poses a significant financial and resource burden. Because it may be impossible to detect every possible fault scenario, especially in industrial complex machinery, accurate fault class labeling based on the machine's operating conditions typically requires skilled personnel [33].
- (3) Variations in loads, speeds, and temperatures are common operating conditions for machines in industrial settings. Domain shift, in which data distribution changes between two domains, is the result of this unpredictability and variability. Due to this shift, DL models trained on data from one domain may struggle to generalize effectively under new conditions, which makes them unreliable for defect diagnosis in the real-world scenarios [34].

Transfer learning (TL) has the potential to mitigate data scarcity, reduce labeling costs, and tackle the domain shift problem by leveraging knowledge from one domain to another on a related task. The key aspect of TL lies in applying what has been learned in one domain to address one problem to a related but distinct domain. Figure 1 illustrates the fundamental difference between traditional machine learning architecture and transfer learning architecture. The TL approaches can be briefly categorized into four groups: parameter-based TL, feature-based TL, instance-based TL, and relevance-based TL [34]. Each of these groups tailors themselves to handle a distinct set of circumstances and difficulties. Among these approaches, parameter-based TL is one of the most popular TL methods, particularly suitable for classification tasks due to its simplicity and ease of implementation [35]. An integral strategy in the parameter-based TL framework is fine-tuning, which involves modifying the parameters of a pre-trained model to adapt it to a new but related task [36,37]. It efficiently employs learned features from a model trained on a large, labeled dataset, minimizing the amount of data and time required to train on the new task. As a rule, when fine-tuning a model, it is necessary to retrain the higher layers that are task-specific while freezing the bottom layers that capture general information. This speeds up training and improves the model's performance on the new task.

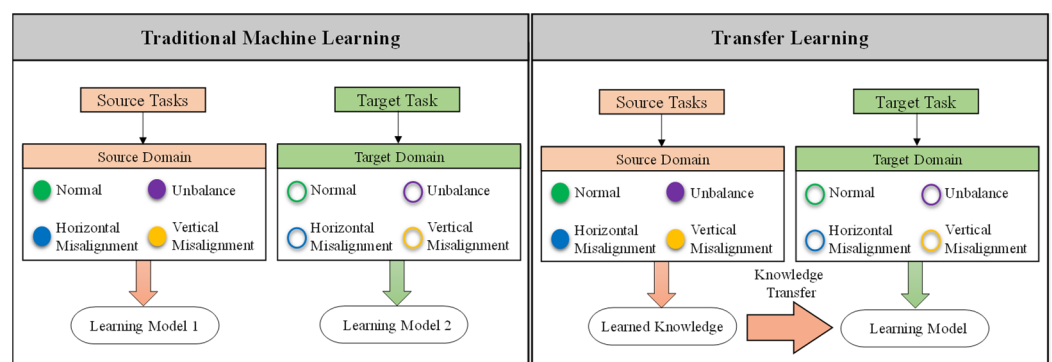


**Figure 1.** The difference between traditional machine learning architecture and transfer learning architecture.

Many studies have investigated the potential applications of fine-tuning methods for failure predictions using time series data. For example, Shao et al. [37] proposed a novel approach based on TL and fine-tuning. Su et al. [38] converted the time series fault signals into images and fine-tuned the pre-trained VGG-16 model. They also examined how image size influences a fine-tuned model. The authors [39] used two models, VGG-16 with and without an attention mechanism, to evaluate the applicability of models on different gear datasets. They did not consider other environmental conditions in which the model would be applicable. In a study on CNN-SA module fault detection using images, Zhong et al. [40] emphasized the importance of the CNN model's last layers. On the other hand, LeNet-5 [41] and MobileNet V2 [42] are alternative built-in models for fault identification that use fine-tuning techniques. To improve fault diagnosis with infrared thermal images under different WCs, Shao et al. [43] suggested a fine-tuning strategy based on enhanced CNN and stacked convolutional autoencoders. The use of built-in pre-trained models for fault identification, which convert time series vibration data into images, presents several drawbacks. (1) There is a potential incompatibility between the pre-trained model's architecture and the specific characteristics of the vibration data [44]. (2) The process of converting time series data into images can result in a loss of temporal information that is crucial for accurately diagnosing faults in machinery [32,45]. (3) A small or diverse dataset can result in overfitting due to the large number of parameters. The intricacy and inherent bias from training on previous data may make these models difficult to understand and unreliable in new scenarios [46,47].

Time series data and simpler models, such as one-dimensional CNN networks [48,49], stacked autoencoders [50], and LSTM networks [51,52], can improve fault classification accuracy and interpretability. LSTMs are specifically designed to capture temporal dependencies in sequential data, enabling more effective analysis of vibration signals without losing critical information during image conversion. Some investigators used LSTM networks to detect machine faults via fine tuning. For instance, Tang et al. [51] pre-trained the LSTM model using labeled samples from bearing and gear source datasets and then fine-tuned it using a small collection of labeled target data. However, the authors manually extracted the features from raw vibrational signals before inputting them into the model. Furthermore, to diagnose faults in wind turbine SCADA datasets, Zhang et al. [53] used a two-layered LSTM model. According to [54], the addition of the L1 regularization technique resolves the overfitting issue that arises during fine-tuning with small, labeled target data. For bearing fault diagnosis, An et al. [52] proposed an intelligent fault diagnosis method using LSTM with RNN architecture under varying WCs.

### 1.1. Research Gap

It is noteworthy that the aforementioned work has achieved significant improvements in recent years. However, a notable gap exists in the literature regarding the application of fine-tuning methods to other types of machine faults. This limitation highlights the need for further research to explore the versatility of fine-tuning techniques, potentially leading to more generalized models for condition monitoring. The following shortcomings in earlier studies need to be addressed:

1. The requirement of uniform data distribution for training and testing datasets should be addressed. However, in practical situations, variations in machine operating conditions may modify vibrational patterns, thereby affecting the data obtained from the actual operational platform. Additionally, training and testing datasets have unequal data samples and distributions.
2. Currently, models are typically trained independently for each task, limiting their applicability to new working conditions (e.g., varying loads and speeds) and highlighting the need for significant improvements in fault identification accuracy.
3. Acquiring data labels in the context of big data can be tedious because of the inherent limitations, including labor-intensive and time-consuming procedures.
4. When working with sparse labeled data, TL performs fault diagnosis more efficiently. Overfitting is a significant problem when using pre-trained models, especially with small datasets, due to the advanced nature of these models and their typically numerous parameters.

### 1.2. Key Contributions

This paper introduces a Triplex Transfer LSTM (TTLSTM), a fine-tuning strategy-based transfer learning method to address the observed limitations for intelligent fault diagnosis in rotating machines. This approach is designed to enhance fault diagnostic performance, especially when the target domain has limited labeled data. The key contributions of the proposed methodology are summarized as follows:

1. This paper proposes a Triplex Transfer Long Short-Term Memory (TTLSTM) method, which uses a deep LSTM network to learn long-term dependencies and intricate non-linear correlations in the data.
2. This method uses the empirical mode decomposition (EMD) technique to extract features from non-stationary and non-linear vibrational signals, as well as the Pearson correlation coefficient (PCC) feature selection method, which improves the model's diagnostic performance.
3. With small, labeled target domain data, a fine-tuning strategy-based TL method is designed for effective fault diagnosis. The proposed method overcomes the limitations of insufficient labeled data and domain shift problems by leveraging transfer learning

and fine-tuning strategies to improve the model's adaptability across diverse working conditions (from low motor speed to high motor speed and vice versa).

4. To alleviate the overfitting problem, L2 regularization TL is utilized, particularly in scenarios with limited labeled target data. The ablation experiments are also conducted to validate the positive impact of L2 regularization in the TTLSTM model. The developed model is then compared with state-of-the-art fault diagnosis methods to ensure its robust performance across WCs and its ability to generalize well to new unknown data.

The remaining sections of the paper are organized as follows: Section 2 presents a problem description and introduces the fundamental concepts of fine-tuning strategy-based transfer learning along with LSTM. Building upon this foundation, Section 3 discusses the proposed methodology (TTLSTM) centered around deep LSTM. This section outlines the key steps, starting with data pre-processing involving normalization, followed by the detailed implementation of EMD. Section 4 discusses experimental results in depth. Finally, Section 5 concludes the paper.

## 2. Problem Description and Preliminary

This section offers a comprehensive description of the fine-tuning strategy-based transfer learning problem and explores its intricacies. It provides a detailed explanation of the core concept of transfer learning and elucidates the architecture of the LSTM network.

### 2.1. Concept of Transfer Learning

It is common practice in traditional ML and DL methods to train and validate models using labeled data from a specific domain. Transfer learning, on the other hand, used training and testing data from distinct but related domains. Parameter-based TL has been the most widely used transfer learning approach because of its intuitive underlying principle and simple implementation, making it especially effective for classification problems that use labeled training data, particularly with the help of fine-tuning algorithms [35,55].

A domain is defined as  $\mathcal{D} = \{\mathbb{X}, P(X)\}$  which includes the feature space  $\mathbb{X}$  and marginal probability distribution  $P(X)$ . Labeled source domain data in one domain can be expressed as  $\mathcal{D}_S = \{\mathbb{X}_S, P(X_S)\}_{i=1}^{n_S} = \{(x_{S_1}, y_{S_1}), (x_{S_2}, y_{S_2}), \dots, (x_{S_i}, y_{S_i})\}_{i=1}^{n_S}$ , where  $x_{S_i} \in \mathbb{X}_S$  and  $y_{S_i} \in \mathcal{Y}_S$  are samples and labels of source domain data and  $n_S$  are number of samples in the source domain data. Target domain data can be denoted as  $\mathcal{D}_T = \{\mathbb{X}_T, P(X_T)\}_{i=1}^{n_T} = \{(x_{T_1}, y_{T_1}), (x_{T_2}, y_{T_2}), \dots, (x_{T_i}, y_{T_i})\}_{i=1}^{n_T}$ , where  $x_{T_i} \in \mathbb{X}_T$  and  $y_{T_i} \in \mathcal{Y}_T$  are samples and labels of target domain data and  $n_T$  are number of samples in the target domain. The existence of different distributions implies that there are two potential scenarios: either the feature spaces are distinct, or the marginal probabilities differ. In its simplest form, transfer learning is a process of transferring knowledge from one related but distinct domain (the source domain  $\mathcal{D}_S$ ) to another (the target domain  $\mathcal{D}_T$ ).

In this work, we assume that the source domain and the target domain share the same set of features by having identical feature spaces ( $\mathbb{X}_S = \mathbb{X}_T$ ). Despite sharing the same label spaces ( $\mathcal{Y}_S = \mathcal{Y}_T$ ), the two domains' data distributions are quite different from one another. Because of this attribute, the method of transfer learning utilized in the research was considered an example of homogeneous transfer learning [55].

### 2.2. Long Short-Term Memory Network

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNNs) developed to handle sequential input and capture long-term dependencies. The development of LSTM addressed the limitations of conventional RNNs in capturing and memorizing information from prolonged sequences. The core key of LSTM is the idea of memory cells that can be used to temporarily store data and then update it with the latest information. The input gate, forget gate, and output gate are the three primary elements of these memory cells. Every gate in a memory cell regulates how data enters and leaves the device [56].

Mathematically, an LSTM unit can be represented as follows:

$$i_t = \sigma(W_{xi} * x_t + W_{hi} * h_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma(W_{xf} * x_t + W_{hf} * h_{t-1} + b_f) \quad (2)$$

$$g_t = \tanh(W_{xg} * x_t + W_{hg} * h_{t-1} + b_g) \quad (3)$$

$$C_t = f_t \otimes C_{t-1} + i_t \otimes g_t \quad (4)$$

$$o_t = \sigma(W_{xo} * x_t + W_{ho} * h_{t-1} + b_o) \quad (5)$$

$$h_t = o_t \otimes \tanh(C_t) \quad (6)$$

where  $i_t$ ,  $f_t$ ,  $g_t$ , and  $o_t$  are input gate, forget gate, candidate cell and output gate of LSTM network.  $W_{xi}$ ,  $W_{hi}$ ,  $W_{xf}$ ,  $W_{hf}$ ,  $W_{xg}$ ,  $W_{hg}$ ,  $W_{xo}$ , and  $W_{ho}$  are weights matrices, and  $b_i$ ,  $b_f$ ,  $b_g$ , and  $b_o$  are bias vectors.  $x_t$ ,  $h_t$ , and  $C_t$  are current input vector, current hidden state, and current state of the cell. The Hadamard product, denoted by  $\otimes$ , signifies the element-wise multiplication between two vectors. The parameters described above undergo continuous updates throughout the entire network training process until they successfully establish the desired mapping relationship between the input and output.

This paper proposes a Triplex Transfer LSTM (TTLSTM) network, which uses a deep LSTM network, to improve the model's adaptability across diverse WCs with limited labeled target data. This architecture enhances the model's capacity to adapt to differing data distributions between the source and target domains. Because memory cells have gating processes that enable them to learn and retain long-term dependencies in the data, LSTM is the optimal choice for diagnosing faults in rotating machines. In an LSTM network, the sigmoid and hyperbolic tangent functions play crucial roles. These functions, along with the gating mechanisms, make it easier to control the information flow and memory retention, enabling the identification of complex patterns and correlations within sequential data. This approach enables the model to effectively transfer knowledge from the source to the target domain, ultimately contributing to improved performance and generalization across diverse domains. Section 3 will explain the details of the proposed TTLSTM network.

### 2.3. Sliding Window

In the realm of processing sequential data, such as time series vibrational data from rotating machines, the sliding window technique serves as a crucial method for generating input sequences suitable for models like LSTM [57]. Both the source and target domain data typically exist as signals, whereas LSTM necessitates a 2D or 3D matrix as input. A 2D matrix typically comprises the dimensions of time and features [timestep, features], while a 3D matrix adds an extra dimension of batches, i.e., [batch\_size, timestep, features]. Consequently, signal data must be transformed into a 3D matrix format for direct input into the LSTM.

Let  $X$  represent the input data time series signal from machinery vibration. The sliding window technique applies a specified window width ( $w$ ) to the time series data. Each window encapsulates the data as a data matrix unit. This approach splits the sequential data into either overlapping or non-overlapping segments of a fixed length. These segments (windows) move through the data one step at a time, assisting in the conversion of individual data points into organized input sequences suitable for LSTM processing. Mathematically, at each time step  $t$ , the sliding window extracts a segment of  $X$ , denoted as  $X_{t:t+w-1}$ , which represents the data from time  $t$  to  $t + w - 1$ . This process continues iteratively, creating a series of sequential data segments suitable for further analysis or processing, as shown in Figure 2.

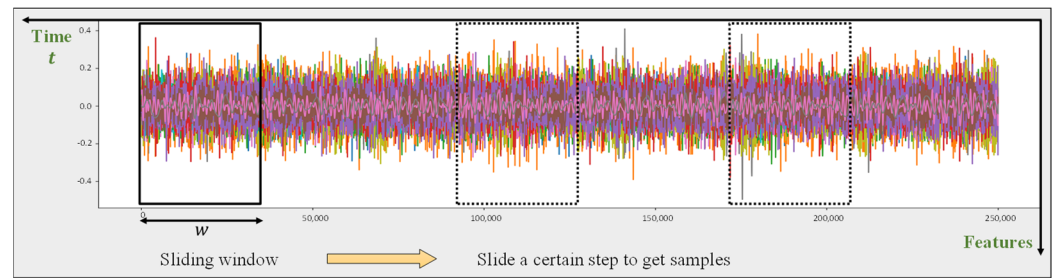


**Figure 2.** Data segmentation: utilizing sliding windows.

### 3. Proposed Methodology

The following section will discuss a Triplex Transfer LSTM (TTLSTM), a fine-tuning strategy-based transfer learning method for intelligent fault diagnosis in rotating machines. It will cover its basic architecture, proposed methodology, essential implementation steps, and model pre-training and fine-tuning approach.

#### 3.1. Pre-Processing Overview

##### 3.1.1. Normalization

Normalization of the data is a crucial step in data pre-processing to ensure that all features have a consistent scale and distribution. One commonly used technique for data normalization is Min–Max scaling, as explained in Equation (7).

$$X_{norm} = \frac{(X - X_{min})}{(X_{max} - X_{min})}, \quad (7)$$

where  $X$  is the original value and  $X_{norm}$  is the normalized value. The normalized data has a range of  $[0, 1]$ . Min–Max scaling offers several benefits, including mitigating the impact of outliers, facilitating the convergence of optimization algorithms, and thus improving the performance of deep learning models.

##### 3.1.2. Empirical Mode Decomposition

Empirical mode decomposition (EMD), an adaptive noise mitigation method, is a powerful tool for pre-processing non-linear and non-stationary time series signals. In other words, it decomposes the non-stationary time series signals into their constituent, finite components, known as intrinsic mode functions (IMFs). Since EMD adaptively decomposes the signal based on its intrinsic features, it is ideally suited for fault identification in rotating machinery.

The envelopes of an IMF are symmetric with zero, and there are the same number of extrema as zero crossings [58,59]. Sifting is the term used to describe the steps taken to remove IMFs, and they are as follows:

1. To create the upper and lower envelopes, locate all the local extrema, and then use a cubic spline to connect the local maxima and minima. There should be no information lost between the envelopes.
2. The first component is the difference between the mean of the upper and lower envelope ( $m_1$ ) and the signal  $x(t)$ .

$$d_1 = m_1 - x(t), m_1 = [e_{upper}(t) + e_{lower}(t)] / 2 \quad (8)$$

To create IMFs, the following conditions should be fulfilled:

- A maximum of one (1) difference is allowed between the number of extrema and zero crossings.
- The midpoint between the upper and lower bounds is always zero.

3. If these above conditions are fulfilled, then  $d_1$  will be the first IMF. If this condition is not satisfied,  $d_1$  will become the original signal, and again, other IMF will be generated  $d_2$ . This process repeats  $n$ -times, as indicated by

$$d_{1(n-1)} - m_{1n} = d_{1n} \quad (9)$$

This logic can be regarded as  $d_{1n} = S_1$ .

4. Differentiate  $S_1$  from  $x(t)$

$$x(t) - S_1 = u_1 \quad (10)$$

This  $u_1$  is treated as the original signal, and all the above steps are repeated  $k$  times until  $u_k$  becomes a monotonic function from which no more IMFs can be extracted.

The original signal can be described as

$$x(t) = \sum_{j=1}^k S_j + r_k \quad (11)$$

$r_k$  are called residuals.

### 3.1.3. Feature Selection via Pearson Correlation Coefficient

The linear relationship between two quantitative variables ( $X$  and  $Y$ ) can be measured by an indicator called the Pearson correlation coefficient (PCC). In this study, PCC assesses the similarity between the high-frequency intrinsic mode function (IMF) and the original signal to gauge its usefulness. PCC can be expressed as

$$PCC = \frac{E[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}, \quad (12)$$

where  $\mu_X$  and  $\mu_Y$  are means of signals  $X$  and  $Y$ , and  $\sigma_X$  and  $\sigma_Y$  are the standard deviations of signals  $X$  and  $Y$ , respectively.  $E$  is the expectation. The correlation coefficient, ranging from  $-1$  to  $+1$ , provides insight into the strength and direction of a linear relationship between  $X$  and  $Y$ . It defines four levels of correlation strength: strong, moderate, weak, and no correlation, along with the corresponding ranges of PCC values. A positive or negative sign indicates a positive or negative correlation between the variables, respectively [60].

### 3.2. The Architecture of TTLSTM Model

The primary objective of this study is to use the concept of feature extraction to build a domain-adaptive intelligent fault diagnostic model. One potential approach for extracting domain-invariant characteristics from vibrational signals is the application of deep learning techniques. This article suggests a TTLSTM model that uses a deep LSTM network to train on a large enough labeled source domain dataset and then fine-tune on a small, labeled target domain dataset. This approach involves more than just relying on the features extracted from pre-trained models and replacing the final layers; it also entails retraining the layers.

#### 3.2.1. Pre-Training Network

Pre-training, which involves training a model with massive, labeled data, is identical to supervised learning. The most noticeable distinction is that this labeled data typically comes from a different domain than the target domain, while the diagnostic tasks remain consistent across domains, focusing on fault diagnosis under varied WCs. The pre-training phase aims to learn suitable representations and patterns from the labeled source domain data [37,61,62]. Different LSTM layers learn progressively more abstract representations of the input data. The first LSTM layer learns lower-level temporal patterns, while the second layer learns higher-level patterns, and so on. This logic makes the model well-suited for

tasks involving sequences, time series, or sequential data, as it can identify complicated temporal dependencies in the data.

### 3.2.2. Fine-Tuning Network

Once the model has been trained on the available labeled source data, its performance is assessed to determine how well it retained the information. However, the main goal is to effectively adapt this pre-trained model to the target domain, which may have distinct features or a different distribution despite sharing the same label space. Consequently, adjustments are made to the pre-trained model to align it with the target task. Typically, this involves removing the existing output layer(s) and integrating new layers tailored to the target task. Unlike traditional fine-tuning approaches that only adjust the last layers, this model fine-tunes all layers to enhance accuracy [63]. Subsequently, the model parameters undergo fine-tuning using limited, labeled target data samples. Training persists until the model reaches convergence or achieves satisfactory performance. As a result, this approach allows the model to adapt to new data while retaining the knowledge gained from the initial training on the original data. Figure 3 details the proposed Triplex Transfer LSTM (TTLSTM), a fine-tuning strategy-based transfer learning model.

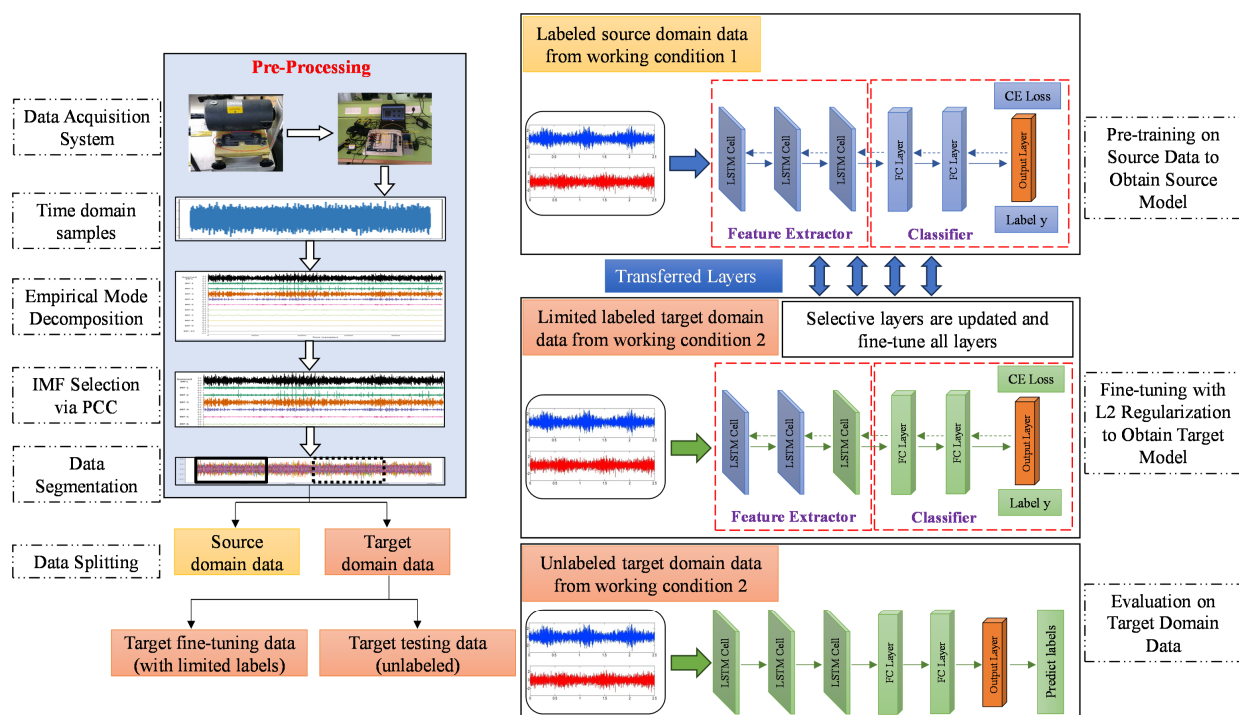


**Figure 3.** The architecture of the proposed TTLSTM model.

### 3.2.3. Optimizer and Activation Function

In this study, the parameters of the TTLSTM model's layers are optimized using the Adam optimizer. The following equations govern the optimization process:

Exponential moving average of gradients ( $m_t$ ):

$$m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * g_t \quad (13)$$

such that  $g_t = \frac{\partial f(x, w)}{\partial w}$ . Moreover, exponential moving average of squared gradients ( $v_t$ ):

$$v_t = \beta_2 * v_{t-1} + (1 - \beta_2) * (g_t)^2 \quad (14)$$

Bias-corrected moving averages:

$$\hat{m}_t = m_t / (1 - \beta_1^t) \quad (15)$$

$$\hat{v}_t = v_t / (1 - \beta_2^t) \quad (16)$$

Parameter update:

$$\theta_t + 1 = \theta_t - \frac{\alpha * \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \quad (17)$$

Here,  $m_t$  represents the moving average of past gradients  $g_t$ ,  $v_t$  is the moving average of past squared gradients, “ $\alpha$ ” is the learning rate, and “ $\epsilon$ ” is a small constant used to prevent division by zero. These equations collectively govern the optimization process, ensuring the model’s parameters are updated efficiently and adaptively during training. Moreover, this model uses the activation function ReLU because it can address vanishing gradient issues, introduce non-linearity, and improve training efficiency. ReLU can be expressed as

$$\text{ReLU}(x) = \max(0, x) \quad (18)$$

The ReLU function returns  $x$  when it is positive and gives 0 when it is negative. In short, this activation function is applied to each element (neuron) in a feature map independently, and it returns the transformed output of that element.

### 3.2.4. Classification Loss and Classifier

A classification problem is to classify different types of faults, and a categorical cross-entropy loss function is used, making it suitable for multi-class classification problems. For this, actual labels are converted into one-hot encoded labels, making them directly appropriate to use with the categorical cross-entropy loss function. The cross-entropy represents the Kullback–Leibler (KL) divergence between the model’s predicted and the true labels. The output layer is configured with four nodes (one for each class). Cross-entropy (CE) loss can be defined as

$$\text{Loss}_{CE} = - \sum_{i=1}^N y_i \cdot \log \hat{y}_i \quad (19)$$

The SoftMax classifier is a critical component for multi-class classification tasks, enabling models to produce meaningful class probabilities and make predictions in scenarios where instances can belong to one of several possible classes. The SoftMax classifier uses the following formula:

$$\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_{j=1}^N e^{z_j}}, \quad (20)$$

where  $z$  is the output from the fully connected layer, and  $N$  is the number of classes to be predicted. Both the pre-training and the fine-tuning stages utilize the specific loss function for the classification outputs. This approach adjusts the model’s learned features to better align with the characteristics of the target domain, enhancing its performance on the target task under varying working conditions.

### 3.2.5. L2 Regularization Transfer Learning

“Regularization for simplicity” is another term for this technique. Finding the model’s complexity as a function of weights reveals that a feature’s complexity is proportional to its absolute value. Although L2 regularization does not reach absolute zero, it does influence weights that get closer to zero. L2 regularization helps to eliminate a nominal proportion

of weights. Hence, weights can never be zero. Mathematically, L2 regularization for  $N$  number of weights can be expressed as

$$L_{reg}term = w_1^2 + w_2^2 + \dots + w_N^2 = \sum_{i=1}^N w_i^2 \quad (21)$$

The regularization rate (lambda  $\lambda$ ) is an extra parameter that can be used to adjust the L2 regularization term. The L2 regularization term is multiplied by the regularization rate, which is a scalar. It is imperative to choose an ideal value for lambda. The model tends to underfit if lambda is set too large because it becomes too simplistic. However, the model is prone to overfitting, and the regularization effect becomes insignificant when lambda is too low. Setting lambda to zero will disable regularization entirely, which increases the likelihood of overfitting.

The goal is to reduce the total loss as much as possible during the fine-tuning step, which is controlled by Equation (21), while also making it easier to generalize to the target domain data. Therefore, we can use the hybrid loss function to effectively balance different learning goals across domains and achieve effective transfer learning.

$$Objective = -\sum_{i=1}^N y_i \cdot \log \hat{y}_i + \lambda \sum_{i=1}^N w_i^2 \quad (22)$$

In this formulation, the L2 regularization term penalizes large weights in the network, helping to prevent overfitting to the target domain data during fine-tuning. It also generates stable solutions and manages features with high correlations. By minimizing this combined objective function, the model aims to achieve excellent performance on the target task while also generalizing well to unseen data. Hence, the hybrid loss function illustrates an approach to efficient transfer learning by balancing various learning goals across domains.

### 3.3. Overview of Proposed Methodology

Figure 4 illustrates the general process of the proposed methodology and summarizes it as follows:

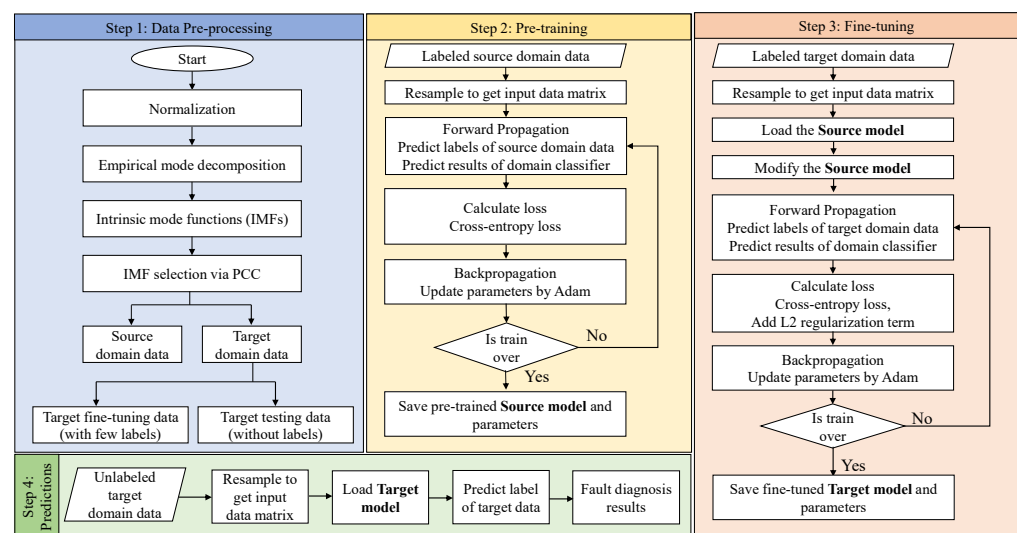


**Figure 4.** General procedure of proposed EMD-TTLSTM.

**Step 1:** The COMFAULDA dataset containing vibration data from both domains is utilized. Data pre-processing begins with normalization, followed by extracting valuable information from data in both domains using EMD. The selection of IMFs with more information about faults is carried out via PCC. The source domain data are divided into three subsets: 70% for training, 15% for testing, and 15% for validation. Additionally, the

data are resampled to make it match the TTLSTM model's input requirements using a sequential data pre-processing method, employing a sliding window approach, as explained in Section 2.3.

**Step 2:** In the pre-training phase, the model is trained on data from the source domain to identify significant features and patterns, resulting in the creation of the **source model**. This is done by optimizing the model's parameters to minimize an objective function, as described in Equation (20). The deep features acquired are then mapped to outputs using the SoftMax classifier, thereby allowing the model to effectively capture the inherent knowledge present in the source domain dataset, as discussed in Section 3.2.

**Step 3:** After pre-training, the next stage involves constructing the **target model**, which adapts the pre-trained model to the target task through fine-tuning with limited labeled target domain data. This process entails modifying the pre-trained model's parameters to match the characteristics of the target data. By minimizing the combined objective function (Equation (22)), the fine-tuning aims to enhance the model's efficiency on the target task while preventing overfitting. By varying the percentages of labeled target training samples, the fine-tuning of the **target model** is systematically adjusted to align well with the features of the remaining target testing samples.

Collectively, this fine-tuning strategy-based transfer learning method for fault diagnosis is called **Triplex Transfer LSTM (TTLSTM)**, and it includes both the **source model** and **target model**.

**Step 4:** After fine-tuning, the model is deployed to predict unknown faults from an unlabeled target domain and evaluate its accuracy and generalizability to unseen data under varying working conditions. To mitigate overfitting, the L2 regularization term is employed to validate the model's performance. Additionally, early stopping criteria are applied to prevent unnecessary training epochs and ensure optimal convergence.

## 4. Experimental Study

### 4.1. Dataset Description

To diagnose faults in rotating machine systems, the proposed TTLSTM approach considers the influence of varying working conditions. The approach is thoroughly described, incorporating extensive simulations and experimental investigations. The results obtained from these experiments demonstrate the efficiency, effectiveness, and generalizability of the method. Specifically, experiments are conducted on the benchmark COMFAULDA dataset to identify faults in rotating machines [64], which consists of vibration signals. Spectra Quest's Alignment Balance Vibration Trainer (ABVT) records vibration signals at an experimental bench for this dataset. These data are collected at the Federal University of Rio de Janeiro's Dynamic Testing and Vibration Analysis Laboratory (LEDAV/COPPE/UFRJ). The essential features of the bench are summarized in Table 1.

**Table 1.** Detail description of training parameters in different experiments.

| ABVT System               | Features            |
|---------------------------|---------------------|
| Motor type                | Direct current (DC) |
| Motor power               | 0.25 HP             |
| Speed range               | [12, 60] Hz         |
| System mass               | 22 kg               |
| Shaft length              | 520 mm              |
| Shaft diameter            | 16 mm               |
| Rotor diameter            | 152.4 mm            |
| Distance between bearings | 390 mm              |

Figure 5 displays the experimental benchmark of COMFAULDA. In this study, four health conditions are considered, i.e., normal (N), unbalance (UN), horizontal misalignment (HM), and vertical misalignment (VM) for different sets of RPMs (revolutions per minute-unit of speed). Each fault type (health condition) contains piezoelectric sensor signal data

collected in the axial (X), vertical (Y), and horizontal (Z) directions. Module sensors' main features are signal conditioning for piezoelectric sensors, an operating temperature range of [12, 60] °C, an anti-aliasing filter, and 24-bit resolution at a maximum sampling frequency of 51.2 kHz. The following steps should be taken:

- To induce the unbalance fault into the ABVT, 20 g screws are inserted in a center-hug configuration on the inertia disc.
- To create the horizontal misalignment fault, move the DC motor's base in the horizontal plane and measure with a digital caliper. The horizontal misalignments are limited to no more than 0.5 mm.
- Placing shims of defined thickness on the electric motor's base causes a vertical misalignment. A vertical deviation of 0.51 mm is considered.

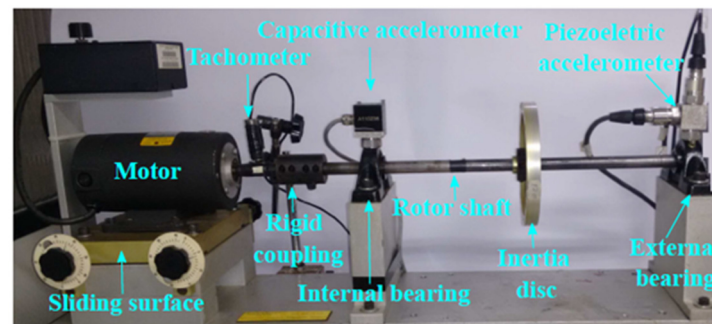


**Figure 5.** Experimental benchmark of COMFAULDA.

The experimental setup observes motor operation across distinct frequency ranges, specifically within the [13, 16] Hz range, classified as “low WC”, and the [47, 50] Hz range, classified as “high WC”. Different sets of low- and high-frequency motor speeds are considered, including various frequency ranges within the designated categories. Table 2 outlines the types of faults under varying working conditions. This approach enables comprehensive analysis across a spectrum of motor operation frequencies to validate the proposed method effectively. Furthermore, to assess the generalizability of the proposed method, experiments are conducted spanning from low (L1, L2, or L3) to high (H1, H2, or H3) speeds and vice versa. In the low-to-high motor speed working condition, the source domain was data from low-speed conditions, and the target domain was data from high-speed conditions. In contrast, the source domain in the high-to-low motor speed working state was high-speed data, whereas the target domain was low-speed data. Each sensor records 150,000 data points for every fault scenario, corresponding to a duration of 3 seconds for each motor speed, with a sampling rate of 50 kHz.

**Table 2.** Detail description of training parameters in different experiments.

| Serial No. | Working Conditions | Motor Speed (Hz) | Health Type   | Working Conditions | Motor Speed (Hz) |
|------------|--------------------|------------------|---------------|--------------------|------------------|
| 1          | L1                 | [13, 14]         | N, UN, HM, VM | H1                 | [47, 48]         |
| 2          | L2                 | [15, 16]         |               | H2                 | [49, 50]         |
| 3          | L3                 | [13, 16]         |               | H3                 | [47, 50]         |

#### 4.2. Detailed Pre-Processing Step

The raw signals from both domains are normalized using Equation (7) for consistent scale and distribution. After that, the EMD technique is applied to all normalized vibrational signals to extract multiple IMFs. As can be noticed in Figure 6, for instance, the unbalance fault signal is decomposed into 12 IMFs using Equations (8)–(11). However, not all IMFs possess useful information, making IMF selection a crucial task. Therefore, a linear relationship between the original signal and each individual IMF via PCC is calculated

using Equation (12). The stronger the link between the two variables is, the closer they are to the extremes. In this case, we use a threshold of 0.01 and select those IMFs whose PCC is more than this threshold. Therefore, the initial six IMFs are chosen, and the rest of the IMFs are considered residual signals. Overall, eighteen (18) IMFs are selected from all sensors for each fault type.

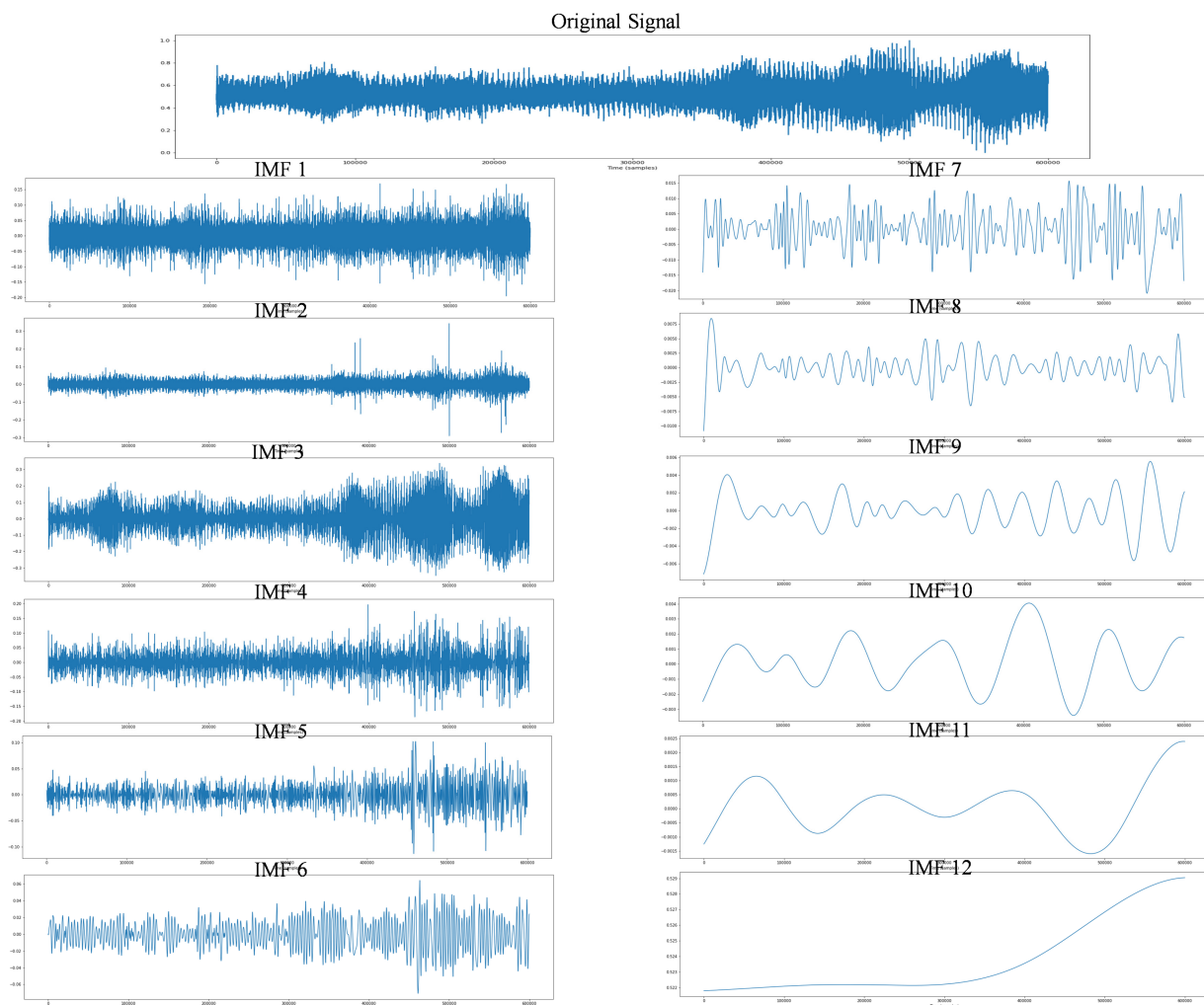


**Figure 6.** Intrinsic mode functions (IMFs) extracted from the original signal.

### 4.3. Diagnostic Performance of the Proposed Method

#### 4.3.1. Experimental Design

The TTLSTM model generates **source** and **target** models through a series of pre-training and fine-tuning steps. To demonstrate the effectiveness of TTLSTM under varying WCs, the first step, i.e., the pre-training process, uses labeled source domain data  $\mathcal{D}_S = \{\mathbb{X}_S, \mathcal{Y}_S\}$ . The labeled source domain data provides batches of data with consistent lengths for each training step. For fine-tuning, target data are divided into labeled  $\mathcal{D}_T = \{\mathbb{X}_T, \mathcal{Y}_T\}$  and unlabeled  $\mathcal{D}_T = \{\mathbb{X}_T\}$  domain datasets. Table 3 outlines a series of experiments conducted for fine-tuning batches in the target domain and pre-training batches in the source domain. Each experiment involves transferring knowledge from low- to high-speed WCs. For each experiment, different percentages of labeled training samples in the target domain are considered for fine-tuning, ranging from 10% to 50% of total target data. Corresponding testing sets are also provided for evaluation and validation. Additionally, the pre-training batches in the source domain consist of specific numbers of training, testing, and validation samples.

**Table 3.** Detail description of training parameters in different experiments of (L → H) WCs.

| Experiments | Transfer Tasks | Fine-Tuning Batches for Different % of Training Samples in the Target Domain                                                                                                                                         | Pre-Training Batches in the Source Domain           |
|-------------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------|
| 1           | L1 → H1        | 10%: 1875 (training), 16,875 (testing)                                                                                                                                                                               | Train: 26,250<br>Test: 5625<br>Validation: 5625     |
| 2           | L2 → H2        | 20%: 3750 (training), 15,000 (testing)                                                                                                                                                                               |                                                     |
| 3           | L1 → H2        | 30%: 5625 (training), 13,125 (testing)                                                                                                                                                                               |                                                     |
| 4           | L2 → H1        | 40%: 7500 (training), 11,250 (testing)<br>50%: 9375 (training), 9375 (testing)                                                                                                                                       |                                                     |
| 5           | L3 → H3        | 10%: 3750 (training), 67,500 (testing)<br>20%: 7500 (training), 60,000 (testing)<br>30%: 11,250 (training), 52,500 (testing)<br>40%: 15,000 (training), 45,000 (testing)<br>50%: 18,750 (training), 37,500 (testing) | Train: 52,500<br>Test: 11,250<br>Validation: 11,250 |

#### 4.3.2. Source Domain-Based Effectiveness Evaluation (Pre-Training)

This section involves the training process and deep feature extraction via the proposed method. The proposed pre-training source model comprises an input sequence layer followed by three LSTM layers, dense layers, activation layers, and a classification layer. Hyperparameters are determined using a random search method [65], selecting 142 hidden units for each LSTM layer and a classification layer length of 4. With an initial learning rate of 0.001 and a batch size of 32, training proceeds for 300 epochs. The network parameters are optimized using the Adam optimizer. Randomization of training samples is implemented to mitigate network overfitting. Choosing appropriate hyperparameters during the construction and training of the source model is crucial for achieving high diagnostic accuracy, faster convergence, and overall robustness. Table 4 describes the detailed list of parameters used during training (pre-training and fine-tuning).

**Table 4.** List of parameters used during pre-training and fine-tuning.

| List of Parameters                                                  |                              |
|---------------------------------------------------------------------|------------------------------|
| Number of hidden layers during pre-training and fine-tuning process | 3                            |
| Number of memory units/nodes in pre-training process                | 142, 142, 142                |
| Number of memory units/nodes in fine-tuning process                 | 142, 142, 64                 |
| Output units/nodes                                                  | 4                            |
| Activation function of hidden units                                 | ReLU                         |
| Classification activation function                                  | softmax                      |
| Optimizer                                                           | Adam (learning rate = 0.001) |
| L2 regularization during fine-tuning process                        | $\lambda = 0.001$            |
| Batch size in pre-training and fine-tuning process                  | 32 and 64                    |
| Epochs for pre-training process and fine-tuning process             | 300 and 50                   |
| Number of features after PCC feature selection method               | 18                           |

The first experiment trains the source data at a frequency of [13, 14] Hz in a supervised manner, validates it, and tests it on the same domain data. In this scenario, we got accuracy: 99.42% for training, 99.46% for validation, and 99.95% for testing. Experiment 3 also uses the same source domain model for fine-tuning. Similarly, experiments 2 and 4 yield high accuracy of 99.54% for training and 99.88% for testing, utilizing identical source domain speed ranges of [15, 16] Hz. In Experiment 5, the model is trained using data that possess the entire frequency spectrum of the source domain ([13, 16] Hz). This comprehensive approach achieves impressive accuracies of 99.36% for training and 99.99% for testing, maintaining consistency with the number of training parameters used across experiments.

The presented Table 5 summarizes the performance metrics of the pre-training model across various motor speed ranges.

**Table 5.** Pre-training source model performance: classification loss and accuracy metrics for different motor speed ranges.

| Working Conditions | Motor Speed | Training CE Loss | Training Acc | Val Loss | Val Acc | Test Acc |
|--------------------|-------------|------------------|--------------|----------|---------|----------|
| L1                 | [13, 14] Hz | 0.01598          | 99.42        | 0.01553  | 99.46   | 99.95    |
| L2                 | [15, 16] Hz | 0.01394          | 99.54        | 0.01423  | 99.57   | 99.88    |
| L3                 | [13, 16] Hz | 0.01811          | 99.36        | 0.01898  | 99.36   | 99.99    |

Val = validation, Acc = accuracy.

Notably, the model demonstrates high class accuracy and low classification loss values across all speed ranges, indicating its effectiveness in capturing relevant features and patterns in the data. These results highlight the efficiency of the pre-training process in capturing relevant features, thereby facilitating accurate classification across diverse working conditions during the fine-tuning phase.

#### 4.3.3. Evaluating Transferability Under Varying WCs of (L → H)

To ensure the adaptability and domain generalization of the diagnostic model, a fine-tuning strategy emerges as a promising solution. This strategy adjusts the model's parameters based on a limited number of labeled target samples, enabling effective adaptation to new WCs. To achieve this, we modify the pre-trained model by taking the initial two LSTM layers from the source domain and then adding a third LSTM layer with 64 hidden units. Fine-tuning lasts over 50 epochs, incorporating the L2 regularization technique ( $\lambda = 0.001$ ) tuned by the random search method. Similar to the pre-training phase, optimization of network parameters employs the Adam optimizer. The pre-trained source model is fine-tuned on varying percentages of available labeled target domain data and then evaluated on the remaining unlabeled target domain data. Most papers [66,67] assume that 50% of labeled data are available in target domain data. To validate our proposed method's robustness and effectiveness, this process involves reducing the percentage of labeled data from 50% to 10%.

Four evaluation metrics are calculated to evaluate the proposed method's domain adaptation capacity, including accuracy, precision, recall, and F1-score. Table 6 presents the evaluation metrics for various experiments conducted with different percentages of training samples in the target domain data. In the first experiment, the model is initially trained using source data from L1 ([13, 14] Hz) to acquire knowledge or features. Following that, target data from H1 ([47, 48] Hz) is used to fine-tune the model and see how well it can generalize and use the new information. Fine-tuning is performed using varying percentages of the target domain data, ranging from 50% to 10%, resulting in accuracy ranging from 99.91% to 99.004%. In experiment 3, the same source model from L1 is employed, but this time, the target domain data are from H2 ([49, 50] Hz). After fine-tuning the source model on these data, achieved accuracy ranges from 99.96% to 98.7%.

Experiment 2 involves feature extraction from the source domain L2 ([15, 16] Hz), followed by fine-tuning on target domain data H2 ([49, 50] Hz). Here, accuracy falls in the range of 99.94% to 98.56%. Similarly, in experiment 4, the model achieves accuracy ranging from 99.97% to 98.62% after fine-tuning on H1 ([47, 48] Hz) using source domain data L2. In the final experiment, considering the wide speed range of the motor as a working condition, source domain data L3 ([13, 16] Hz) and target domain data H3 ([47, 50] Hz) are utilized. When the labeled samples are reduced from 50% to 10% during fine-tuning, accuracy varies between 99.82% and 97.66%. These results indicate that the model's performance is robust and resilient to changes in the amount of labeled data, showcasing its ability to maintain accuracy within a narrow range of variation, typically within approximately  $\pm 1\%$  range.

**Table 6.** Performance analysis for transfer tasks with varying target training samples from domain (L → H).

| Transfer Tasks | Accuracy (%) |       |       |       |       | Precision (%) |       |       |       |       |
|----------------|--------------|-------|-------|-------|-------|---------------|-------|-------|-------|-------|
|                | 50%          | 40%   | 30%   | 20%   | 10%   | 50%           | 40%   | 30%   | 20%   | 10%   |
| L1 → H1        | 99.92        | 99.95 | 99.90 | 99.65 | 99.00 | 99.92         | 99.95 | 99.90 | 99.65 | 99.01 |
| L2 → H2        | 99.94        | 99.93 | 99.86 | 99.65 | 98.56 | 99.94         | 99.93 | 99.86 | 99.65 | 98.57 |
| L1 → H2        | 99.96        | 99.98 | 99.81 | 99.70 | 98.71 | 99.96         | 99.98 | 99.81 | 99.70 | 98.71 |
| L2 → H1        | 99.97        | 99.94 | 99.78 | 99.64 | 98.62 | 99.97         | 99.94 | 99.78 | 99.64 | 98.63 |
| L3 → H3        | 99.82        | 99.68 | 99.46 | 98.88 | 97.66 | 99.82         | 99.68 | 99.46 | 98.88 | 97.67 |

| Transfer Tasks | Recall (%) |       |       |       |       | F1-Score (%) |       |       |       |       |
|----------------|------------|-------|-------|-------|-------|--------------|-------|-------|-------|-------|
|                | 50%        | 40%   | 30%   | 20%   | 10%   | 50%          | 40%   | 30%   | 20%   | 10%   |
| L1 → H1        | 99.92      | 99.95 | 99.90 | 99.65 | 99.00 | 99.92        | 99.95 | 99.90 | 99.65 | 99.01 |
| L2 → H2        | 99.94      | 99.93 | 99.86 | 99.65 | 98.57 | 99.94        | 99.93 | 99.86 | 99.65 | 98.57 |
| L1 → H2        | 99.96      | 99.77 | 99.81 | 99.70 | 98.71 | 99.96        | 99.98 | 99.81 | 99.70 | 98.71 |
| L2 → H1        | 99.97      | 99.94 | 99.78 | 99.64 | 98.62 | 99.97        | 99.94 | 99.78 | 99.64 | 98.62 |
| L3 → H3        | 99.82      | 99.68 | 99.46 | 98.88 | 97.66 | 99.82        | 99.68 | 99.46 | 98.88 | 97.66 |

For a comprehensive assessment of the diagnostic performance of individual fault categories across different WCs and training sample sizes, we calculated the percentage of correctly classified instances for each fault category as depicted in Table 7. The experiments demonstrate that reducing the percentage of labeled data has a minimal impact on accuracy, with the model maintaining its performance within a narrow margin of variation. The Average Classification Accuracy (ACA) is also calculated, indicating the average accuracy achieved for each fault category under the specified conditions. ACA is calculated by using Equation (23). Furthermore, Overall Classification Accuracy (OCA) is also calculated to get the insights of the final experimental analysis. OCA is measured by utilizing Equation (24). Overall, this table offers valuable insight into the model's capability to accurately classify different faults while adapting to varying operational scenarios and data availability.

$$ACA = \frac{\sum(\text{class-wise accuracy})}{\sum(\text{total number of classes})} * 100 \quad (23)$$

$$OCA = \frac{\sum(ACA)}{\sum \text{total number of scenarios of training samples in particular experiment}} * 100 \quad (24)$$

For instance, in Experiment 5 (L3 → H3), the percentage of correctly classified samples for each fault category is as follows: using 50% of the target domain's training samples for fine-tuning yields N = 99.81%, UN = 99.98%, HM = 99.66%, and VM = 99.82%. However, under the same conditions, using only 10% of the training samples for fine-tuning results in a slight decrease in the accuracy rates for fault diagnosis. For instance, the accuracy rates for N, UN, HM, and VM are 97.27%, 99.7%, 97.54%, and 96.12%, respectively. With the training samples decreasing from 50% to 10%, we achieved ACAs of 99.82%, 99.68%, 99.46%, 98.88%, and 97.66%, respectively. Consequently, the OCA of this experiment is calculated as 99.10%. Similarly, for the other experiments conducted, the impact of varying the percentage of training samples on fault diagnosis accuracy was analyzed. It is noteworthy that in each experiment, reducing the training dataset from 50% to 10% results in a decrease in accuracy. However, this observation demonstrates the proposed model's robustness in effectively adapting to domain variations, even when faced with significant motor speed variations, as seen particularly in Experiment 5.

All experiments are conducted on a personal laptop with the i7-1165G7 processor and 16 GB of RAM. Python software of the 3.9.12 version and a scikit-learn library are used to calculate all the evaluation matrices.

**Table 7.** Diagnostic performance of individual faults in varying WCs with different % of training samples in target domain data for case (L  $\rightarrow$  H).

| Transfer Tasks      | % of Training Samples in Target Domain Data | Individual Faults |       |       |       | ACA (%) | OCA (%) |
|---------------------|---------------------------------------------|-------------------|-------|-------|-------|---------|---------|
|                     |                                             | N                 | UN    | HM    | VM    |         |         |
| L1 $\rightarrow$ H1 | 10%                                         | 98.78             | 99.82 | 99.32 | 98.09 | 99.00   | 99.68   |
|                     | 20%                                         | 99.61             | 99.99 | 99.78 | 99.2  | 99.65   |         |
|                     | 30%                                         | 99.93             | 99.99 | 99.82 | 99.85 | 99.90   |         |
|                     | 40%                                         | 99.96             | 99.98 | 99.91 | 99.94 | 99.95   |         |
|                     | 50%                                         | 99.93             | 99.94 | 99.91 | 99.9  | 99.92   |         |
| L2 $\rightarrow$ H2 | 10%                                         | 98.7              | 99.99 | 97.11 | 98.46 | 98.57   | 99.59   |
|                     | 20%                                         | 99.49             | 99.97 | 99.51 | 99.6  | 99.64   |         |
|                     | 30%                                         | 99.91             | 99.99 | 99.81 | 99.88 | 99.90   |         |
|                     | 40%                                         | 99.99             | 100   | 99.74 | 99.97 | 99.93   |         |
|                     | 50%                                         | 99.98             | 100   | 99.87 | 99.92 | 99.94   |         |
| L1 $\rightarrow$ H2 | 10%                                         | 98.95             | 99.98 | 97.61 | 98.28 | 98.71   | 99.63   |
|                     | 20%                                         | 99.69             | 100   | 99.46 | 99.66 | 99.70   |         |
|                     | 30%                                         | 99.67             | 99.99 | 99.93 | 99.66 | 99.81   |         |
|                     | 40%                                         | 99.97             | 100   | 99.99 | 99.95 | 99.98   |         |
|                     | 50%                                         | 99.98             | 100   | 99.89 | 99.97 | 99.96   |         |
| L2 $\rightarrow$ H1 | 10%                                         | 98.68             | 99.78 | 99.16 | 96.86 | 98.62   | 99.59   |
|                     | 20%                                         | 99.4              | 99.98 | 99.8  | 99.37 | 99.64   |         |
|                     | 30%                                         | 99.94             | 99.46 | 99.95 | 99.79 | 99.79   |         |
|                     | 40%                                         | 99.93             | 99.99 | 99.96 | 99.87 | 99.94   |         |
|                     | 50%                                         | 99.98             | 100   | 99.95 | 99.94 | 99.97   |         |
| L3 $\rightarrow$ H3 | 10%                                         | 97.27             | 99.7  | 97.54 | 96.12 | 97.66   | 99.10   |
|                     | 20%                                         | 99.03             | 99.92 | 98.57 | 97.99 | 98.88   |         |
|                     | 30%                                         | 99.57             | 99.97 | 99.27 | 99.02 | 99.46   |         |
|                     | 40%                                         | 99.78             | 99.99 | 99.63 | 99.31 | 99.68   |         |
|                     | 50%                                         | 99.81             | 99.98 | 99.66 | 99.82 | 99.82   |         |

#### 4.3.4. Special Case: Evaluating Transferability Under Varying WCs of (H $\rightarrow$ L)

In addition to assessing the transferability of the diagnostic model from low to high-speed WCs, it is equally crucial to evaluate its performance under the reverse scenario, i.e., from high to low speed. This special case analysis aims to elucidate the generalizability and adaptability of the model across a wide spectrum of motor operating speeds. By examining how effectively the model applies its learned features and knowledge to new conditions, we gain insight into its generalization capabilities and potential real-world applicability. In this section, we delve into the experimental setup and results of this unique evaluation.

In this scenario, the high-speed WCs (H1, H2, or H3) serve as the source domain data, while the low-speed WCs (L1, L2, or L3) constitute the target domain data. The hyperparameters utilized both the pre-training and fine-tuning processes to construct the source and target models, which remain consistent with those employed in the previous case. Here, we conduct five experiments, where we transfer knowledge from the high-speed to the low-speed domain (denoted as H  $\rightarrow$  L). For each experiment, varying percentages of labeled training samples in the target domain are considered for fine-tuning, ranging from 50% to 10% of the total target data. Corresponding testing sets are allocated for evaluation and validation purposes. Experiments 1 and 4 focus on training, validating, and testing the model using data from the [47, 48] Hz frequency range, resulting in accuracy of 99.69% for training and 99.89% for testing. Similarly, experiments 2 and 3 achieve impressive accuracies of 99.69% for training and 99.93% for testing, utilizing the [49, 50] Hz frequency range from the source domain. In Experiment 5, the model is trained on the whole frequency range of the source domain [47, 50] Hz, which leads to impressive accuracy rates of 99.62% for training and 99.904% for testing. Table 8 summarizes the results, showcasing the source model's performance across various motor speed ranges.

**Table 8.** Performance metrics of pre-training source model across different motor speed ranges for case (H  $\rightarrow$  L).

| Working Conditions | Motor Speed | Training CE Loss | Training Acc | Val Loss | Val Acc | Test Acc |
|--------------------|-------------|------------------|--------------|----------|---------|----------|
| H1                 | [47–48] Hz  | 0.0088           | 99.69        | 0.0076   | 99.74   | 99.89    |
| H2                 | [49–50] Hz  | 0.0088           | 99.689       | 0.0072   | 99.76   | 99.93    |
| H3                 | [47–50] Hz  | 0.011            | 99.628       | 0.0104   | 99.67   | 99.904   |

Val = validation, Acc = accuracy.

Following pre-training, the model undergoes fine-tuning on different percentages of labeled target domain data, followed by evaluation on unlabeled target domain data to assess the method's robustness and generalization. Evaluation metrics, including accuracy, precision, recall, and F1-score, are computed. Table 9 illustrates the performance metrics across various experiments with varying percentages of training samples in the target domain. In the first experiment (H1  $\rightarrow$  L1), when the training samples are reduced from 50% to 10%, the accuracy falls between 99.19% and 94.14%. Experiment 4 (H1  $\rightarrow$  L2) has an identical source model, but the target domain is L2 ([15, 16] Hz), yielding accuracy between 99.42% and 95.46%. The accuracy achieved for experiment 2 (H2  $\rightarrow$  L2) ranged from 99.46% to 95.88%. Furthermore, experiment 3 (H2  $\rightarrow$  L1) attains 99.25% to 94.73% accuracy. The accuracies in the final experiment, achieved by using H3 ([47, 50] Hz) and L3 ([13, 16] Hz), varied between 96.67% and 90.28% when the number of labeled samples was reduced from 50% to 10%.

**Table 9.** Performance analysis for transfer tasks with varying target training samples from domain (H  $\rightarrow$  L).

| Transfer Tasks      | Accuracy (%) |       |       |       |       | Precision (%) |       |       |       |       |
|---------------------|--------------|-------|-------|-------|-------|---------------|-------|-------|-------|-------|
|                     | 50%          | 40%   | 30%   | 20%   | 10%   | 50%           | 40%   | 30%   | 20%   | 10%   |
| H1 $\rightarrow$ L1 | 99.19        | 98.33 | 98.27 | 97.39 | 94.14 | 99.19         | 98.33 | 98.28 | 97.39 | 94.16 |
| H2 $\rightarrow$ L2 | 99.46        | 98.82 | 98.83 | 98.26 | 95.88 | 99.46         | 98.83 | 98.83 | 98.26 | 95.93 |
| H2 $\rightarrow$ L1 | 99.25        | 98.74 | 98.24 | 97.37 | 94.73 | 99.25         | 98.74 | 98.24 | 97.39 | 94.73 |
| H1 $\rightarrow$ L2 | 99.42        | 98.99 | 98.69 | 97.84 | 95.46 | 99.42         | 98.99 | 98.69 | 97.84 | 95.51 |
| H3 $\rightarrow$ L3 | 96.67        | 96.16 | 94.91 | 93.64 | 90.28 | 96.68         | 96.16 | 94.92 | 93.64 | 90.31 |
| Transfer Tasks      | Recall (%)   |       |       |       |       | F1-Score (%)  |       |       |       |       |
|                     | 50%          | 40%   | 30%   | 20%   | 10%   | 50%           | 40%   | 30%   | 20%   | 10%   |
| H1 $\rightarrow$ L1 | 99.19        | 98.33 | 98.27 | 97.39 | 94.14 | 99.19         | 98.33 | 98.27 | 97.39 | 94.14 |
| H2 $\rightarrow$ L2 | 99.46        | 98.82 | 98.83 | 98.26 | 95.88 | 99.46         | 98.82 | 98.83 | 98.26 | 95.88 |
| H2 $\rightarrow$ L1 | 99.25        | 98.74 | 98.24 | 97.37 | 94.73 | 99.25         | 98.74 | 98.24 | 97.37 | 94.73 |
| H1 $\rightarrow$ L2 | 99.42        | 98.99 | 98.69 | 97.84 | 95.46 | 99.42         | 98.99 | 98.69 | 97.84 | 95.46 |
| H3 $\rightarrow$ L3 | 96.67        | 96.16 | 94.91 | 93.64 | 90.28 | 96.68         | 96.15 | 94.90 | 93.63 | 90.26 |

These results demonstrate the model's resilience to change in labeled data, maintaining accuracy within approximately  $\pm 3\%$  to  $\pm 5\%$  range. For a thorough evaluation of the diagnostic accuracy across various fault categories under diverse WCs (H  $\rightarrow$  L) and training sample sizes, we conducted a meticulous analysis, focusing on the percentage of correctly classified instances for each specific fault category. The Average Classification Accuracy, and Overall Classification Accuracy are also computed to provide a holistic assessment of the model's diagnostic capabilities. Despite the reduction in training samples from 50% to 10%, ACAs and OCAs remained relatively high. Table 10 provides further details on the accuracy rates for each fault category under varying experimental conditions.

**Table 10.** Diagnostic performance of individual faults in varying working conditions with different % of training samples in target domain data for (H  $\rightarrow$  L).

| Transfer Tasks      | % of Training Samples in Target Domain Data | Individual Faults |       |       |       | ACA (%) | OCA (%) |
|---------------------|---------------------------------------------|-------------------|-------|-------|-------|---------|---------|
|                     |                                             | N                 | UN    | HM    | VM    |         |         |
| H1 $\rightarrow$ L1 | 10%                                         | 94.71             | 91.11 | 95.16 | 95.59 | 94.14   | 97.47   |
|                     | 20%                                         | 98.11             | 96.33 | 97.92 | 97.20 | 97.39   |         |
|                     | 30%                                         | 98.65             | 98.52 | 97.11 | 98.80 | 98.27   |         |
|                     | 40%                                         | 99.39             | 98.06 | 97.80 | 98.07 | 98.33   |         |
|                     | 50%                                         | 99.32             | 98.75 | 99.17 | 99.53 | 99.19   |         |
| H2 $\rightarrow$ L2 | 10%                                         | 97.47             | 95.34 | 97.44 | 93.28 | 95.88   | 98.25   |
|                     | 20%                                         | 98.68             | 97.90 | 98.81 | 97.63 | 98.26   |         |
|                     | 30%                                         | 99.34             | 98.51 | 98.82 | 98.64 | 98.83   |         |
|                     | 40%                                         | 99.33             | 98.85 | 99.55 | 97.56 | 98.82   |         |
|                     | 50%                                         | 99.59             | 99.16 | 99.53 | 99.56 | 99.46   |         |
| H2 $\rightarrow$ L1 | 10%                                         | 96.49             | 92.94 | 95.92 | 93.58 | 94.73   | 97.67   |
|                     | 20%                                         | 97.24             | 97.97 | 96.72 | 97.56 | 97.37   |         |
|                     | 30%                                         | 98.43             | 98.08 | 98.28 | 98.16 | 98.24   |         |
|                     | 40%                                         | 98.98             | 98.25 | 99.02 | 98.69 | 98.74   |         |
|                     | 50%                                         | 99.31             | 98.80 | 99.40 | 99.48 | 99.25   |         |
| H1 $\rightarrow$ L2 | 10%                                         | 94.76             | 97.50 | 95.89 | 93.71 | 95.47   | 98.08   |
|                     | 20%                                         | 98.35             | 97.94 | 97.46 | 97.60 | 97.84   |         |
|                     | 30%                                         | 99.03             | 98.93 | 98.47 | 98.30 | 98.68   |         |
|                     | 40%                                         | 99.15             | 98.27 | 99.30 | 99.24 | 98.99   |         |
|                     | 50%                                         | 99.71             | 99.30 | 99.47 | 99.20 | 99.42   |         |
| H3 $\rightarrow$ L3 | 10%                                         | 92.14             | 87.38 | 93.75 | 87.84 | 90.28   | 94.33   |
|                     | 20%                                         | 94.03             | 91.63 | 95.24 | 93.64 | 93.64   |         |
|                     | 30%                                         | 96.52             | 91.90 | 95.70 | 95.52 | 94.91   |         |
|                     | 40%                                         | 97.27             | 94.78 | 95.97 | 96.61 | 96.16   |         |
|                     | 50%                                         | 96.68             | 96.79 | 96.72 | 96.51 | 96.68   |         |

#### 4.3.5. Analysis

In transitioning from low to high (L  $\rightarrow$  H) working conditions, the model consistently demonstrated accuracy within a narrow range of about  $\pm 1\%$  across varying percentages of labeled target domain data that were used for fine-tuning. However, during the reverse transition from high to low (H  $\rightarrow$  L) speed, the variation in accuracy widened, ranging from approximately  $\pm 3\%$  to  $\pm 5\%$ . Specifically, while experiments under low to high conditions consistently yielded high OCAs, averaging approximately 99.52%, those under high to low conditions exhibited less consistent performance, with an average OCA of approximately 97.16%. Notably, while experiments 2 and 4 achieved high OCAs of 98.25% and 98.08%, respectively, others, particularly experiment 5, experienced a slightly decrease in accuracy with an OCA of 94.33%. The disparity in accuracy between transitions from low to high speed compared with those from high to low speed highlights an intriguing phenomenon warranting further investigation. Several contributing factors likely observed this discrepancy.

Firstly, the distribution of data points and their intrinsic characteristics may differ significantly between the low and high-speed regimes compared with the transition from high to low speed. Furthermore, the relevance and transferability of extracted features during pre-training could vary depending on the direction of speed transition. Features may be inherently more applicable to high-speed domains, facilitating smoother adaptation during fine-tuning. Conversely, discerning relevant patterns and features in low-speed conditions may pose greater challenges, leading to more pronounced fluctuations in accuracy. Lastly, the size of the training sample used for fine-tuning is a critical factor. Smaller training sample sizes, particularly during transitions from high to low speed, may

introduce more variability in the model's learned representations, thereby influencing accuracy fluctuations.

#### 4.3.6. Ablation Study: Effect of L2 Regularization on the TTLSTM Model

Our suggested Triplex Transfer LSTM (TTLSTM) is a fine-tuning strategy-based transfer learning model that employs L2 regularization to prevent overfitting. This improves the model generalizability, especially in high-dimensional feature spaces, by adding a penalty to the loss function. To determine its impact, a series of experiments are conducted with and without the use of L2 regularization. We evaluate the performance with and without L2 regularization while utilizing 10% labeled target domain data for fine-tuning, particularly the (L  $\rightarrow$  H) scenario, employing a **source model** and a **target model**. All other hyperparameters and training conditions were kept the same to ensure the effect of L2 regularization.

The t-SNE method, which reduces dimensionality by mapping high-dimensional data onto a two-dimensional plane, illustrates the impact of L2 regularization on the feature space. This visualization provides a better understanding of the model's capacity to differentiate between different fault types (classes). On the other hand, applying L2 regularization makes the classes appear more clearly differentiated and have smooth, compact shapes; however, not applying L2 regularization results in the classes appearing as tangled and overlapped clusters. Different clustering patterns demonstrate how L2 regularization reduces overfitting and improves model generalizability. Figure 7 shows how L2 regularization affects the categorization output.

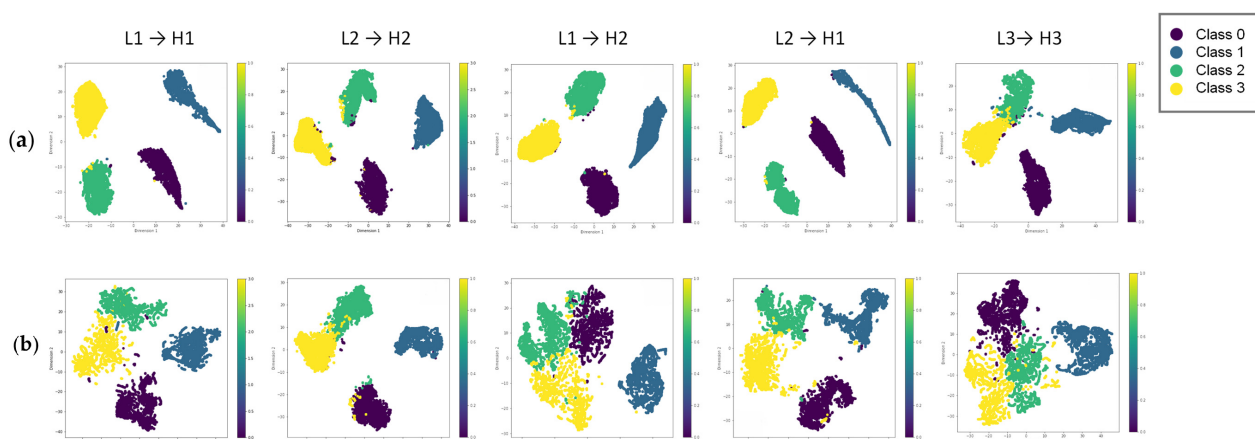


**Figure 7.** An ablation study: t-SNE feature visualization (a) with L2 regularization (b) without L2 regularization.

#### 4.4. Comparative Analysis

##### 4.4.1. Transfer Learning vs. Without Transfer Learning Across Varying Working Conditions

The objective of examination without transfer learning is to evaluate the feasibility of using the trained models directly across various working conditions. However, the results highlight a significant inadequacy in this approach. It becomes evident that without transfer learning, the models' efficiency significantly suffers. This limitation arises from the challenge of transferring knowledge from the source domain to the target domain, especially when significant differences exist between the two domains.

To validate the efficacy of the TTLSTM methodology, several investigations were conducted. We compared the performance of LSTM with transfer learning (utilizing the proposed method) and LSTM without transfer learning (LSTM without the fine-tuning step). The absence of transfer learning implies that the model exclusively trains on data from the source domain and evaluates it using unlabeled data from the target domain. In this case, the training-to-testing ratio is kept consistent with transfer learning-based

scenarios. The comprehensive analysis provided in Table 11 contrasts the accuracy achieved with and without transfer learning across different transfer tasks and varying percentages of training samples. The results clearly demonstrate a notable disparity, confirming the challenge of utilizing models trained from the source domain directly in another domain.

**Table 11.** Performance comparison: transfer learning vs. without transfer learning with varying training samples.

| Transfer Tasks | With Proposed Transfer Learning |       |       |       |       | Without Transfer Learning |       |       |       |       |
|----------------|---------------------------------|-------|-------|-------|-------|---------------------------|-------|-------|-------|-------|
|                | 50%                             | 40%   | 30%   | 20%   | 10%   | 50%                       | 40%   | 30%   | 20%   | 10%   |
| L1 → H1        | 99.92                           | 99.95 | 99.90 | 99.65 | 99.00 | 24.06                     | 24.08 | 24.00 | 24.03 | 24.08 |
| L2 → H2        | 99.94                           | 99.93 | 99.86 | 99.65 | 98.56 | 27.24                     | 27.21 | 27.23 | 27.20 | 27.20 |
| L1 → H2        | 99.96                           | 99.98 | 99.81 | 99.70 | 98.71 | 20.58                     | 20.52 | 20.64 | 20.61 | 20.64 |
| L2 → H1        | 99.97                           | 99.94 | 99.78 | 99.64 | 98.62 | 28.60                     | 28.60 | 28.55 | 28.59 | 28.61 |
| L3 → H3        | 99.82                           | 99.68 | 99.46 | 98.88 | 97.66 | 25.84                     | 25.87 | 25.94 | 25.94 | 25.92 |
| H1 → L1        | 99.19                           | 98.33 | 98.27 | 97.39 | 94.14 | 31.06                     | 31.15 | 31.19 | 31.13 | 31.11 |
| H2 → L2        | 99.46                           | 98.82 | 98.83 | 98.26 | 95.88 | 29.50                     | 29.61 | 29.54 | 29.57 | 29.51 |
| H2 → L1        | 99.25                           | 98.74 | 98.24 | 97.37 | 94.73 | 28.63                     | 28.67 | 28.71 | 28.68 | 28.71 |
| H1 → L2        | 99.42                           | 98.99 | 98.69 | 97.84 | 95.51 | 33.47                     | 33.49 | 33.52 | 33.49 | 33.44 |
| H3 → L3        | 96.67                           | 96.16 | 94.91 | 93.64 | 90.28 | 31.17                     | 31.18 | 31.20 | 31.19 | 31.15 |

#### 4.4.2. Comparison with Other Methods

To demonstrate the effectiveness of our proposed TTLSTM model, this study conducts an experimental comparison among several state-of-the-art and classical models on the dataset mentioned above. Support Vector Machine (SVM), Random Forest (RF), and Extreme Gradient Boosting (XGBoost) are some of the classical models that were considered. From the raw data of each sensor, 28 statistical features were extracted (total 84 features) and fed into the classifier. To ensure a fair comparison with the proposed methodology, principal component analysis (PCA) is applied to extract 18 valuable features from these 84 features, then input them into the classifier. Moreover, three experiments are conducted using the extracted 18 features from EMD on raw data, following the proposed method, and subsequently applying classical models.

The comparison also includes various traditional transfer learning algorithms, such as Transfer Component Analysis (TCA) [68], Balanced Distribution Adaptation (BDA) [69], and Joint Distribution Adaptation (JDA) [70]. These established transfer learning methods consider source and target domain data together, aiming to align their features and uniformly distribute both marginal and conditional distributions between the domains. Table 12 provides details about each model, including its inputs and model descriptions. Classical models such as SVM, RF, and XGBoost are configured with specific parameters, while traditional transfer learning approaches, such as TCA, BDA, and JDA, employ Random Forest Classifier with kernel “RBF” and specific values for dim, lamb, mu, and gamma. The proposed TTLSTM model (mainly for the task (L3 → H3)) is compared with these models to evaluate its performance against state-of-the-art and classical methodologies.

The classical models exhibit a slight improvement in accuracy across different classifiers. This phenomenon stems from the inherent disparity between source and target signal data, attributed to variations in working conditions. Consequently, achieving competitive results by directly applying source-trained models to the target domain remains challenging, even with well-optimized parameters. Traditional TL methods demonstrate accuracy values ranging from 46% to 47.1%, indicating moderate performance. Their performance typically falls between 40–60% [58], as these methods leverage shallow models that utilize a matrix for mapping data into a latent space, which may be less expressive than deep models. Furthermore, TCA, JDA, and BDA suffer significant information loss because they do not use the labels of target domain data. In contrast, the proposed method achieves 99.09% accuracy. The LSTM architecture in TTLSTM is proficient in learning temporal dependencies within vibrational time series data, allowing the model to capture dynamic

fault characteristics with improved efficiency. These results highlight the effectiveness of the proposed method in improving accuracy and classification performance, emphasizing its potential as a highly effective strategy for domain adaptation.

**Table 12.** Model specifications, descriptions for experimental comparison, and their diagnostic accuracy.

| Ref                | Model         | Input                                                                   | Model Description                                                                                                                                                                                                                                                      | Acc (%)                                |
|--------------------|---------------|-------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------|
| Proposed (average) | TTLSTM        | 18 features extracted from EMD                                          | Extraction of features using EMD is followed by feature selection using PCC. Pre-training on the source model followed by fine-tuning using a small amount of labeled data using L2 regularization transfer learning strategy                                          | 99.09                                  |
| [71]               | CNN           | Raw signal combines with data from virtual sensors input to a scalogram | Used pre-trained models of ResNet18 and customized CNN model.<br><b>Limitation:</b> focused primarily on training and testing split, neglecting diverse working conditions and domain shift problems.                                                                  | ResNet18 (98.2) customized CNN (97.22) |
| [72]               | CNN           | Raw signal                                                              | Continuous wavelets transform to convert raw signal to scalogram and CNN model.<br><b>Limitation:</b> failed to address the complexities introduced by variations in working conditions and domain shifts.                                                             | 97.14                                  |
| [73]               | 1D-CNN and AE | Raw, noisy data                                                         | AE, 1D-CNN, MMD, categorical cross-entropy loss, DNN<br><b>Limitation:</b> did not fully consider the crucial factors of diverse working conditions and domain shift problems, potentially limiting the model's effectiveness in real-world fault diagnosis scenarios. | 41.63–76.8                             |
| [52]               | LSTM and RNN  | Raw, noisy data                                                         | LSTM, RNN, categorical cross-entropy loss, domain loss.                                                                                                                                                                                                                | 93.20                                  |
| [51]               | LSTM          | Statistical features of raw signal                                      | Stacked LSTM, categorical cross-entropy loss.                                                                                                                                                                                                                          | 96.9                                   |
| Classical          | SVM           | 84 statistical features of raw signal                                   | SVM with RBF kernel and C = 10.                                                                                                                                                                                                                                        | 34.68                                  |
|                    | SVM + PCA     | 18 statistical features of raw signal                                   | SVM with PCA for dimensionality reduction.                                                                                                                                                                                                                             | 41.67                                  |
|                    | RF            | 84 statistical features of raw signal                                   | Random Forest with n_estimators = 100.                                                                                                                                                                                                                                 | 32.73                                  |
|                    | RF + PCA      | 18 statistical features of raw signal                                   | Random Forest with PCA for dimensionality reduction.                                                                                                                                                                                                                   | 35.55                                  |
|                    | XGBoost       | 84 statistical features of raw signal                                   | XGBoost with multi-SoftMax objective.                                                                                                                                                                                                                                  | 34.75                                  |
|                    | XGBoost + PCA | 18 statistical features of raw signal                                   | XGBoost with PCA for dimensionality reduction                                                                                                                                                                                                                          | 31.8                                   |
|                    | EMD + SVM     | 18 features extracted from EMD                                          | SVM trained on features extracted from EMD of raw signal.                                                                                                                                                                                                              | 26.26                                  |
|                    | EMD + RF      | 18 features extracted from EMD                                          | Random forest trained on features extracted from EMD of the raw signal.                                                                                                                                                                                                | 26.1                                   |
|                    | EMD + XGBoost | 18 features extracted from EMD                                          | XGBoost trained on features extracted from EMD of the raw signal.                                                                                                                                                                                                      | 24.71                                  |

Table 12. Cont.

| Ref                                      | Model | Input                          | Model Description                                                                                                                                                                          | Acc (%) |
|------------------------------------------|-------|--------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| Traditional transfer learning approaches | TCA   | 18 features extracted from EMD | Transfer Component Analysis, based on data distribution adaptation, applied to features from both domains with Random Forest Classifier using kernel = RBF, dim = 30, lamb = 1, gamma = 1. | 46      |
|                                          | BDA   | 18 features extracted from EMD | Balanced Distribution Adaptation applied to features from both domains with Random Forest Classifier using kernel= RBF, dim = 30, mu = 0.5.                                                | 47.1    |
|                                          | JDA   | 18 features extracted from EMD | Joint Distribution Adaptation applied to features from both domains with Random Forest Classifier using kernel = RBF, dim = 30, gamma = 1.                                                 | 46.2    |

Ref = references, Acc = accuracy.

#### 4.5. Practical Implications in Industrial Operations

The proposed method aims to improve fault diagnosis through transfer learning, addressing significant issues in industries where labeled fault data are often scarce, particularly for rare faults like unbalance and misalignment. Collecting sufficient labeled fault data is time-consuming and expensive. To eliminate the need for large, labeled fault data in all possible working environments across industries, we propose a Triplex Transfer Long Short-Term Memory (TTLSTM) model, which is based on a pre-training and fine-tuning strategy. The robustness and generalization of the TTLSTM model to domain variations make it well-suited for real-world applications where operating conditions may vary over time. Industrial practitioners can leverage the model to improve maintenance strategies, optimize resource allocation, and enhance operational efficiency. This move can result in reduced downtime, improved productivity, and cost savings. The TTLSTM model provides practical benefits by achieving high accuracy with limited labeled training data, which is a significant advantage in scenarios where acquiring large datasets is difficult. Leveraging transfer learning and fine-tuning techniques enables the model to adapt effectively to new domains, enhancing its practical utility in real-world applications.

Implementing the proposed paradigm in practical scenarios poses significant challenges. There is a lot of variation in the real world. Changes in operational conditions, sensor noise, and machine wear and tear often cause domain shifts that are different from what was seen during model training, which could affect how well the model works. Model interpretability is another crucial element; engineers and operators frequently require insight into the causes of detected faults; however, deep learning models are generally seen as “black boxes”, making interpretability essential for developing confidence and facilitating decision-making. Ultimately, fault characteristics may change as machines undergo repairs, requiring ongoing model adaptation to maintain accuracy. This necessitates precise model updates and maintenance to prevent operational downtime.

The proposed model shows potential for enhancing fault diagnosis in complex industrial situations. However, it is crucial to deal with challenges such as adapting to real-world variability, ensuring interpretability, and facilitating continuous model updates for effective and sustainable implementation. Bridging the gap between experimental performance and real-world applicability, these considerations are essential, which will ultimately result in an increase in the practicality of the model in operational scenarios. The comparison of the TTLSTM model with state-of-the-art and classical models provides practical guidance for fault diagnosis practitioners. Understanding the relative performance of different models facilitates informed decision-making when selecting appropriate approaches for fault diagnosis tasks. This knowledge improves the efficiency and effectiveness of fault diagnosis systems in practice. These implications contribute to advancing fault diagnosis techniques,

improving operational efficiency in diverse industries, and encouraging further research and development in this field.

## 5. Conclusions

This paper presents an intelligent, data-driven approach based on the Triplex Transfer LSTM (TTLSTM) network for fault diagnosis in rotating machinery with limited labeling. During the pre-processing stage of the data, features are extracted from the raw vibrational signals using the EMD approach, and pertinent IMFs are chosen using PCC. The proposed method overcomes the limitations of insufficient labeled data and domain shift problems by leveraging transfer learning and fine-tuning strategies to improve the model's adaptability across diverse working conditions (from low motor speed to high motor speed and vice versa). L2 regularization TL is utilized to mitigate the overfitting problem, especially in scenarios with limited labeled target data. Comprehensive experiments are conducted, considering a range of scenarios with varying percentages of labeled data in the target domain, from 50% to 10%. The results demonstrate the robustness and generalization ability of the proposed approach in varying working conditions ( $L \rightarrow H$  and  $H \rightarrow L$ ). Our model gives satisfying results in recognizing machinery faults even with minimal labeled data compared with classical and conventional transfer learning methods.

In future research, it is essential to explore methods that minimize the reliance on labeled data in the target domain for fine-tuning, aiming for more efficient and scalable transfer learning approaches. In cases where the target domain data does not have labels, there are two main ways to improve cross-domain generalization: either using distance-based methods like Maximum Mean Discrepancy (MMD), Correlation Alignment (CORAL), or Marginalized Kernel MMD (MK-MMD) to minimize discrepancies or employing the mutual information (MI) method to maximize similarities. Additionally, investigating more competitive deep architectures tailored for transfer learning tasks presents both challenges and promising avenues for advancement. Moreover, the proposed methods should be rigorously tested on diverse datasets to validate the generalization capabilities of the model across different domains and application scenarios.

**Author Contributions:** M.I.: Writing—Original Draft, Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Data Curation, Visualization. C.K.M.L.: Writing—Review and Editing, Supervision, Funding Acquisition, Conceptualization, Methodology, Visualization. K.L.K.: Writing—Review and Editing, Validation, Visualization. Z.Z.: Review and Editing. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Department of Industrial and Systems Engineering (RHW0), the Hong Kong Polytechnic University, Hong Kong, and the Centre for Advances in Reliability and Safety Limited (CAiRS).

**Data Availability Statement:** The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding author.

**Acknowledgments:** Our gratitude is also extended to the Research Committee and the Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University (RHW0), Hong Kong, and the Centre for Advances in Reliability and Safety Limited (CAiRS) and Innovation and Technology Commission, The Government of the Hong Kong SAR, under an AIR@InnoHK Project by Innovation and Technology Commission (ITC) Project (Project no. P3.2).

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

## Nomenclature

|         |                                  |
|---------|----------------------------------|
| ACA     | Average Classification Accuracy  |
| AE      | Auto Encoder                     |
| ANN     | Artificial Neural Network        |
| BDA     | Balanced Distribution Adaptation |
| CNN     | Convolutional Neural Networks    |
| DBN     | Deep Belief Network              |
| DL      | Deep Learning                    |
| EMD     | Empirical Mode Decomposition     |
| FFT     | Fast Fourier Transform           |
| GNN     | Graph Neural Networks            |
| IMFs    | Intrinsic Mode Decompositions    |
| JDA     | Joint Distribution Adaptation    |
| KNN     | K-nearest Neighbor               |
| LSTM    | Long Short-Term Memory           |
| OCA     | Overall Classification Accuracy  |
| PCA     | Principal Component Analysis     |
| PCC     | Pearson Correlation Coefficients |
| RF      | Random Forest                    |
| RNN     | Recurrent Neural Networks        |
| SVM     | Support Vector Machine           |
| TCA     | Transfer Component Analysis      |
| TL      | Transfer Learning                |
| WCs     | Working Conditions               |
| WT      | Wavelet Transform                |
| XGBoost | Extreme Gradient Boosting        |

## References

1. Sun, C.; Ma, M.; Zhao, Z.; Chen, X. Sparse deep stacking network for fault diagnosis of motor. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3261–3270. [\[CrossRef\]](#)
2. Shao, H.; Jiang, H.; Zhang, H.; Liang, T. Electric locomotive bearing fault diagnosis using a novel convolutional deep belief network. *IEEE Trans. Ind. Electron.* **2017**, *65*, 2727–2736. [\[CrossRef\]](#)
3. Chen, J.; Pan, J.; Li, Z.; Zi, Y.; Chen, X. Generator bearing fault diagnosis for wind turbine via empirical wavelet transform using measured vibration signals. *Renew. Energy* **2016**, *89*, 80–92. [\[CrossRef\]](#)
4. Xi, W.; Li, Z.; Tian, Z.; Duan, Z. A feature extraction and visualization method for fault detection of marine diesel engines. *Measurement* **2018**, *116*, 429–437. [\[CrossRef\]](#)
5. Sharma, V.; Parey, A. A review of gear fault diagnosis using various condition indicators. *Procedia Eng.* **2016**, *144*, 253–263. [\[CrossRef\]](#)
6. Li, W.; Huang, R.; Li, J.; Liao, Y.; Chen, Z.; He, G.; Yan, R.; 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. [\[CrossRef\]](#)
7. Li, C.; Zhang, S.; Qin, Y.; Estupinan, E. A systematic review of deep transfer learning for machinery fault diagnosis. *Neurocomputing* **2020**, *407*, 121–135. [\[CrossRef\]](#)
8. Shao, H.; Jiang, H.; Wang, F.; Wang, Y. Rolling bearing fault diagnosis using adaptive deep belief network with dual-tree complex wavelet packet. *ISA Trans.* **2017**, *69*, 187–201. [\[CrossRef\]](#)
9. Yuan, H.; Wang, X.; Sun, X.; Ju, Z. Compressive sensing-based feature extraction for bearing fault diagnosis using a heuristic neural network. *Meas. Sci. Technol.* **2017**, *28*, 065018. [\[CrossRef\]](#)
10. Neupane, D.; Seok, J. Bearing fault detection and diagnosis using case western reserve university dataset with deep learning approaches: A review. *IEEE Access* **2020**, *8*, 93155–93178. [\[CrossRef\]](#)
11. Boudiaf, A.; Moussaoui, A.; Dahane, A.; Atoui, I. A comparative study of various methods of bearing faults diagnosis using the case Western Reserve University data. *J. Fail. Anal. Prev.* **2016**, *16*, 271–284. [\[CrossRef\]](#)
12. Bangalore, P.; Tjernberg, L.B. An artificial neural network approach for early fault detection of gearbox bearings. *IEEE Trans. Smart Grid* **2015**, *6*, 980–987. [\[CrossRef\]](#)
13. Abid, F.B.; Zgarni, S.; Braham, A. Distinct bearing faults detection in induction motor by a hybrid optimized SWPT and aiNet-DAG SVM. *IEEE Trans. Energy Convers.* **2018**, *33*, 1692–1699. [\[CrossRef\]](#)
14. Wang, D. K-nearest neighbors based methods for identification of different gear crack levels under different motor speeds and loads: Revisited. *Mech. Syst. Signal Process.* **2016**, *70*, 201–208. [\[CrossRef\]](#)
15. Roy, S.S.; Dey, S.; Chatterjee, S. Autocorrelation aided random forest classifier-based bearing fault detection framework. *IEEE Sens. J.* **2020**, *20*, 10792–10800. [\[CrossRef\]](#)

16. Behseresht, S.; Love, A.; Valdez Pastrana, O.A.; Park, Y.H. Enhancing Fused Deposition Modeling Precision with Serial Communication-Driven Closed-Loop Control and Image Analysis for Fault Diagnosis-Correction. *Materials* **2024**, *17*, 1459. [\[CrossRef\]](#)
17. Sulaiman, M.; Khan, N.A.; Alshammari, F.S.; Laouini, G. Performance of heat transfer in micropolar fluid with isothermal and isoflux boundary conditions using supervised neural networks. *Mathematics* **2023**, *11*, 1173. [\[CrossRef\]](#)
18. Khan, N.A.; Hussain, S.; Spratford, W.; Goecke, R.; Kotecha, K.; Jamwal, P. Deep Learning-Driven Analysis of a Six-Bar Mechanism for Personalized Gait Rehabilitation. *J. Comput. Inf. Sci. Eng.* **2024**, *15*, 011001. [\[CrossRef\]](#)
19. Banitaba, F.S.; Aygun, S.; Najafi, M.H. Late Breaking Results: Fortifying Neural Networks: Safeguarding Against Adversarial Attacks with Stochastic Computing. *arXiv* **2024**, arXiv:2407.04861.
20. Shao, H.; Jiang, H.; Li, X.; Wu, S. Intelligent fault diagnosis of rolling bearing using deep wavelet auto-encoder with extreme learning machine. *Knowl. -Based Syst.* **2018**, *140*, 1–14.
21. Xie, J.; Du, G.; Shen, C.; Chen, N.; Chen, L.; Zhu, Z. An end-to-end model based on improved adaptive deep belief network and its application to bearing fault diagnosis. *IEEE Access* **2018**, *6*, 63584–63596. [\[CrossRef\]](#)
22. Jiang, H.; Li, X.; Shao, H.; Zhao, K. Intelligent fault diagnosis of rolling bearings using an improved deep recurrent neural network. *Meas. Sci. Technol.* **2018**, *29*, 065107. [\[CrossRef\]](#)
23. Pan, J.; Zi, Y.; Chen, J.; Zhou, Z.; Wang, B. LiftingNet: A novel deep learning network with layerwise feature learning from noisy mechanical data for fault classification. *IEEE Trans. Ind. Electron.* **2017**, *65*, 4973–4982. [\[CrossRef\]](#)
24. Wu, J. *Introduction to Convolutional Neural Networks*; National Key Lab for Novel Software Technology, Nanjing University: Nanjing, China, 2017; Volume 5, pp. 1–31.
25. Liu, H.; Zhou, J.; Zheng, Y.; Jiang, W.; Zhang, Y. Fault diagnosis of rolling bearings with recurrent neural network-based autoencoders. *ISA Trans.* **2018**, *77*, 167–178. [\[CrossRef\]](#) [\[PubMed\]](#)
26. Qiao, M.; Yan, S.; Tang, X.; Xu, C. Deep convolutional and LSTM recurrent neural networks for rolling bearing fault diagnosis under strong noises and variable loads. *IEEE Access* **2020**, *8*, 66257–66269. [\[CrossRef\]](#)
27. Zhang, Z.; Wu, L. Graph neural network-based bearing fault diagnosis using Granger causality test. *Expert Syst. Appl.* **2024**, *242*, 122827. [\[CrossRef\]](#)
28. Cui, M.; Wang, Y.; Lin, X.; Zhong, M. Fault diagnosis of rolling bearings based on an improved stack autoencoder and support vector machine. *IEEE Sens. J.* **2020**, *21*, 4927–4937. [\[CrossRef\]](#)
29. Niu, G.; Wang, X.; Golda, M.; Mastro, S.; Zhang, B. An optimized adaptive PReLU-DBN for rolling element bearing fault diagnosis. *Neurocomputing* **2021**, *445*, 26–34. [\[CrossRef\]](#)
30. Chia, Z.C.; Lim, K.H.; Tan, T.P.L. Two-phase Switching Optimization Strategy in LSTM Model for Predictive Maintenance. In Proceedings of the 2021 International Conference on Green Energy, Computing and Sustainable Technology (GECOST), Miri, Malaysia, 7–9 July 2021; pp. 1–6.
31. Ma, M.; Mao, Z. Deep-convolution-based LSTM network for remaining useful life prediction. *IEEE Trans. Ind. Inform.* **2020**, *17*, 1658–1667. [\[CrossRef\]](#)
32. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaria, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 53. [\[CrossRef\]](#)
33. Farahani, A.; Voghoei, S.; Rasheed, K.; Arabnia, H.R. A brief review of domain adaptation. In *Advances in Data Science and Information Engineering, Proceedings of the ICDATA 2020, Las Vegas, NV, USA, 27–30 July 2020 and IKE 2020, Las Vegas, NV, USA, 27–30 July 2020*; Springer: Cham, Switzerland, 2021; pp. 877–894.
34. Misbah, I.; Lee, C.; Keung, K. Fault diagnosis in rotating machines based on transfer learning: Literature review. *Knowl. -Based Syst.* **2023**, *283*, 111158. [\[CrossRef\]](#)
35. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [\[CrossRef\]](#)
36. Han, T.; Liu, C.; Wu, R.; Jiang, D. Deep transfer learning with limited data for machinery fault diagnosis. *Appl. Soft Comput.* **2021**, *103*, 107150. [\[CrossRef\]](#)
37. Shao, S.; McAleer, S.; Yan, R.; Baldi, P. Highly accurate machine fault diagnosis using deep transfer learning. *IEEE Trans. Ind. Inform.* **2018**, *15*, 2446–2455. [\[CrossRef\]](#)
38. Su, J.; Wang, H. Fine-tuning and efficient VGG16 transfer learning fault diagnosis method for rolling bearing. In Proceedings of the InCoME-VI and TEPEN 2021: Performance Engineering and Maintenance Engineering, Tianjin, China, 20–23 October 2021; Springer: Cham, Switzerland, 2022; pp. 453–461.
39. Sun, Q.; Zhang, Y.; Chu, L.; Tang, Y.; Xu, L.; Li, Q. Fault Diagnosis of Gearbox Based on Cross-domain Transfer Learning with Fine-tuning Mechanism Using Unbalanced Samples. *IEEE Trans. Instrum. Meas.* **2024**, *73*, 1–10. [\[CrossRef\]](#)
40. Zhong, H.; Yu, S.; Trinh, H.; Lv, Y.; Yuan, R.; Wang, Y. Fine-tuning transfer learning based on DCGAN integrated with self-attention and spectral normalization for bearing fault diagnosis. *Measurement* **2023**, *210*, 112421. [\[CrossRef\]](#)
41. Liu, Q.; Huang, C. A fault diagnosis method based on transfer convolutional neural networks. *IEEE Access* **2019**, *7*, 171423–171430. [\[CrossRef\]](#)
42. Tang, T.; Wu, J.; Chen, M. Lightweight model-based two-step fine-tuning for fault diagnosis with limited data. *Meas. Sci. Technol.* **2022**, *33*, 125112. [\[CrossRef\]](#)

43. Shao, H.; Li, W.; Xia, M.; Zhang, Y.; Shen, C.; Williams, D.; Kennedy, A.; de Silva, C.W. Fault diagnosis of a rotor-bearing system under variable rotating speeds using two-stage parameter transfer and infrared thermal images. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–11. [\[CrossRef\]](#)
44. Han, T.; Zhou, T.; Xiang, Y.; Jiang, D. Cross-machine intelligent fault diagnosis of gearbox based on deep learning and parameter transfer. *Struct. Control Health Monit.* **2022**, *29*, e2898. [\[CrossRef\]](#)
45. Kim, H.E.; Cosa-Linan, A.; Santhanam, N.; Jannesari, M.; Maros, M.E.; Ganslandt, T. Transfer learning for medical image classification: A literature review. *BMC Med. Imaging* **2022**, *22*, 69. [\[CrossRef\]](#) [\[PubMed\]](#)
46. Sunyoto, A.; Pristyanto, Y.; Setyanto, A.; Alarfaj, F.; Almusallam, N.; Alreshoodi, M. The Performance Evaluation of Transfer Learning VGG16 Algorithm on Various Chest X-ray Imaging Datasets for COVID-19 Classification. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 196–203. [\[CrossRef\]](#)
47. Zheng, W.; Xue, F.; Chen, Z.; Chen, D.; Guo, B.; Shen, C.; Ai, X.; Wang, N.; Zhang, M.; Ding, Y. Disruption prediction for future tokamaks using parameter-based transfer learning. *Commun. Phys.* **2023**, *6*, 181. [\[CrossRef\]](#)
48. Dibaj, A.; Ettetfagh, M.M.; Hassannejad, R.; Ehghaghi, M.B. A hybrid fine-tuned VMD and CNN scheme for untrained compound fault diagnosis of rotating machinery with unequal-severity faults. *Expert Syst. Appl.* **2021**, *167*, 114094. [\[CrossRef\]](#)
49. Wang, R.; Huang, W.; Lu, Y.; Wang, J.; Ding, C.; Liao, Y.; Shi, J. Cloud-edge collaborative transfer fault diagnosis of rotating machinery via federated fine-tuning and target self-adaptation. *Expert Syst. Appl.* **2024**, *250*, 123859. [\[CrossRef\]](#)
50. Xia, M.; Li, T.; Liu, L.; Xu, L.; de Silva, C.W. Intelligent fault diagnosis approach with unsupervised feature learning by stacked denoising autoencoder. *IET Sci. Meas. Technol.* **2017**, *11*, 687–695. [\[CrossRef\]](#)
51. Tang, Z.; Bo, L.; Liu, X.; Wei, D. A semi-supervised transferable LSTM with feature evaluation for fault diagnosis of rotating machinery. *Appl. Intell.* **2022**, *52*, 1703–1717. [\[CrossRef\]](#)
52. An, Z.; Li, S.; Xin, Y.; Xu, K.; Ma, H. An intelligent fault diagnosis framework dealing with arbitrary length inputs under different working conditions. *Meas. Sci. Technol.* **2019**, *30*, 125107. [\[CrossRef\]](#)
53. Zhang, G.; Li, Y.; Jiang, W.; Shu, L. A fault diagnosis method for wind turbines with limited labeled data based on balanced joint adaptive network. *Neurocomputing* **2022**, *481*, 133–153. [\[CrossRef\]](#)
54. Zhu, D.; Song, X.; Yang, J.; Cong, Y.; Wang, L. A bearing fault diagnosis method based on L1 regularization transfer learning and LSTM deep learning. In Proceedings of the 2021 IEEE International Conference on Information Communication and Software Engineering (ICICSE), Chengdu, China, 19–21 March 2021; pp. 308–312.
55. Weiss, K.; Khoshgoftaar, T.M.; Wang, D. A survey of transfer learning. *J. Big Data* **2016**, *3*, 9. [\[CrossRef\]](#)
56. Manaswi, N.K.; Manaswi, N.K. *Rnn and Lstm. Deep Learning with Applications Using Python: Chatbots and Face, Object, and Speech Recognition with TensorFlow and Keras*; Springer Nature: London, UK, 2018; pp. 115–126.
57. Wang, J.; Jiang, W.; Li, Z.; Lu, Y. A new multi-scale sliding window LSTM framework (MSSW-LSTM): A case study for GNSS time-series prediction. *Remote Sens.* **2021**, *13*, 3328. [\[CrossRef\]](#)
58. Vrba, J.; Cejnek, M.; Steinbach, J.; Krbcova, Z. A machine learning approach for gearbox system fault diagnosis. *Entropy* **2021**, *23*, 1130. [\[CrossRef\]](#) [\[PubMed\]](#)
59. Samanta, B. Gear fault detection using artificial neural networks and support vector machines with genetic algorithms. *Mech. Syst. Signal Process.* **2004**, *18*, 625–644. [\[CrossRef\]](#)
60. Murari, A.; Lungaroni, M.; Peluso, E.; Gaudio, P.; Lerche, E.; Garzotti, L.; Gelfusa, M.; Contributors, J. On the use of transfer entropy to investigate the time horizon of causal influences between signals. *Entropy* **2018**, *20*, 627. [\[CrossRef\]](#)
61. Wang, X.; Shen, C.; Xia, M.; Wang, D.; Zhu, J.; Zhu, Z. Multi-scale deep intra-class transfer learning for bearing fault diagnosis. *Reliab. Eng. Syst. Saf.* **2020**, *202*, 107050. [\[CrossRef\]](#)
62. Zhou, J.; Yang, X.; Zhang, L.; Shao, S.; Bian, G. Multisignal VGG19 network with transposed convolution for rotating machinery fault diagnosis based on deep transfer learning. *Shock Vib.* **2020**, *2020*, 863388. [\[CrossRef\]](#)
63. Li, X.; Hu, Y.; Li, M.; Zheng, J. Fault diagnostics between different type of components: A transfer learning approach. *Appl. Soft Comput.* **2020**, *86*, 105950. [\[CrossRef\]](#)
64. COMPOSED FAULT DATASET (COMFAULDA). Available online: <https://ieee-dataport.org/documents/composed-fault-dataset-comfaulda#files> (accessed on 21 November 2024).
65. Mantovani, R.G.; Rossi, A.L.; Vanschoren, J.; Bischl, B.; De Carvalho, A.C. Effectiveness of random search in SVM hyper-parameter tuning. In Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–17 July 2017; pp. 1–8.
66. Cao, X.; Chen, B.; Zeng, N. A deep domain adaption model with multi-task networks for planetary gearbox fault diagnosis. *Neurocomputing* **2020**, *409*, 173–190. [\[CrossRef\]](#)
67. Wang, Z.; Liu, Q.; Chen, H.; Chu, X. A deformable CNN-DLSTM based transfer learning method for fault diagnosis of rolling bearing under multiple working conditions. *Int. J. Prod. Res.* **2021**, *59*, 4811–4825. [\[CrossRef\]](#)
68. Pan, S.J.; Tsang, I.W.; Kwok, J.T.; Yang, Q. Domain adaptation via transfer component analysis. *IEEE Trans. Neural Netw.* **2010**, *22*, 199–210. [\[CrossRef\]](#)
69. Wang, J.; Chen, Y.; Hao, S.; Feng, W.; Shen, Z. Balanced distribution adaptation for transfer learning. In Proceedings of the 2017 IEEE International Conference on Data Mining (ICDM), New Orleans, LA, USA, 18–21 November 2017; pp. 1129–1134.
70. Long, M.; Wang, J.; Ding, G.; Sun, J.; Yu, P.S. Transfer feature learning with joint distribution adaptation. In Proceedings of the 2013 IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 2200–2207.

71. Khan, A.; Hwang, H.; Kim, H.S. Synthetic data augmentation and deep learning for the fault diagnosis of rotating machines. *Mathematics* **2021**, *9*, 2336. [[CrossRef](#)]
72. Khan, A.; Kim, J.-S.; Kim, H.S. Damage detection and isolation from limited experimental data using simple simulations and knowledge transfer. *Mathematics* **2021**, *10*, 80. [[CrossRef](#)]
73. Arora, J.K.; Rajagopalan, S.; Singh, J.; Purohit, A. Low-Frequency Adaptation-Deep Neural Network-Based Domain Adaptation Approach for Shaft Imbalance Fault Diagnosis. *J. Vib. Eng. Technol.* **2023**, *12*, 375–394. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.