Rolling bearing fault diagnosis using time–frequency image-based deep learning networks

Rodrigo C. Fardin[1], Marcus L. Neves Filho[1], Thomas W. Rauber[1], Berilhes B. Garcia[1], Flávio M. Varejão[1]

[1] *Federal University of Espírito Santo (UFES), Vitória, ES, 29075-910, Brazil*
rodrigo.correa.fardin@gmail.com
marcus.l.neves@edu.ufes.br
thomas.rauber@ufes.br
berilhes.garcia@ufes.br
flavio.varejao@ufes.br

ABSTRACT

Deep learning approaches have shown strong potential for vibration-based fault diagnosis; however, their evaluation is often affected by similarity bias caused by inappropriate data splitting strategies. In such cases, highly similar samples may be simultaneously present in training and testing sets, leading to overly optimistic performance estimates and limited generalization assessment. This paper investigates the use of time–frequency signal representations and transfer learning under a fault diagnosis framework that avoids similarity bias conditions. Raw vibration signals are transformed into two-dimensional representations using the Short-Time Fourier Transform (spectrogram) and the Continuous Wavelet Transform (scalogram), which are used as inputs to ImageNet-pretrained convolutional neural networks. To reduce the impact of similarity bias, a controlled data splitting strategy is adopted to prevent highly similar samples from being shared between training and testing subsets. The framework is evaluated using four publicly available benchmark datasets widely adopted in intelligent fault diagnosis research, enabling consistent comparison with existing studies. Different fine-tuning strategies and network architectures are considered to analyze the influence of signal representation and model adaptation under similarity bias.

1. INTRODUCTION

The increasing use of sensors in industrial systems has expanded the availability of vibration data and stimulated the development of automated fault diagnosis methods (C. Wang et al., 2024). In this context, signal processing and deep learning techniques have enabled the transformation of one-dimensional vibration signals into two-dimensional time–frequency representations, allowing convolutional neural networks to capture discriminative frequency patterns that are difficult to identify in the time domain (Ahmed & Nandi, 2022; Gültekin et al., 2022). However, practical applications remain challenging due to limited dataset sizes, controlled acquisition conditions, and the non-stationary nature of vibration signals, which restrict model generalization. Transfer learning has emerged as an effective strategy to address these limitations by leveraging knowledge from large-scale pretrained models (H. Chen et al., 2024).

In addition to representation and model design, experimental methodology plays a crucial role in ensuring reliable evaluation. Conventional random data splits may introduce similarity bias, leading to overly optimistic results when similar samples appear in both training and testing sets (dos Santos Varejão et al., 2025). To address this issue, this study adopts a similarity-aware data partitioning strategy that groups samples according to acquisition conditions, promoting more realistic performance evaluation and improving the assessment of model generalization.

This work investigates the effectiveness of time–frequency image representations combined with transfer learning for vibration signal-based fault classification under similarity-aware evaluation settings. The study compares partial and full fine-tuning strategies using ImageNet-pretrained convolutional neural networks and evaluates multiple representations and architectures, highlighting their impact on robustness and generalization (Wu et al., 2023; Arun Balaji et al., 2025; Deng et al., 2009).

The previous study of (dos Santos Varejão et al., 2025), which established the similarity-aware evaluation protocol for vibration-based fault diagnosis, showed that conventional random splits lead to overly optimistic results, and provided a benchmark of baselines operating on raw time-domain signals: statistical features with shallow classifiers (Nearest

Rodrigo Fardin et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.
https://doi.org/10.36001/IJPHM.2026.v17i2.5068

Neighbor and Random Forest) and a one-dimensional convolutional network (XResNet18-1D). Building on that protocol, the present work adds the following contributions. First, it evaluates two-dimensional time–frequency representations (STFT and CWT) combined with an ImageNet-pretrained EfficientNetV2-S, under partial and full fine-tuning, on the same datasets, folds, and rounds as the benchmark, which allows paired statistical comparisons with its baselines. Second, it isolates the contribution of transfer learning by comparing it with training from scratch, and it examines the choice of architecture against a lightweight alternative (MobileNetV3). Third, it reports the computational cost of each strategy (parameters, training time, inference latency, and GPU memory), which is relevant for PHM deployment. Unlike studies that combine STFT/CWT images with pretrained CNNs but are typically evaluated with random data splits, where similarity bias may inflate performance, all results reported here are obtained without similarity bias. To the best of our knowledge, this is the first work to experimentally compare transfer learning-based classification techniques applied to 2D representations of vibration signals for fault diagnosis under a similarity-bias-free protocol.

The remainder of this article is organized as follows. Section 2 reviews the related work on vibration-based fault diagnosis, with particular emphasis on time–frequency representations and deep learning-based classification methods. Section 3 presents the conceptual background, covering the fundamental principles of time–frequency signal analysis, image-based representations, and transfer learning strategies. Section 4 introduces the proposed time–frequency representation and transfer learning-based framework for intelligent fault diagnosis, detailing the overall architecture, network backbones, and fine-tuning strategies, and describes the experimental methodology, including dataset descriptions, the unbiased data partitioning strategy, the training protocol, and the evaluation metrics. Section 5 reports and discusses the experimental results, including the computational cost analysis, the statistical comparisons, and the limitations of the study. Finally, Section 6 concludes the paper and outlines potential directions for future research.

2. RELATED WORK

Recent studies in vibration-based fault diagnosis increasingly transform one-dimensional vibration signals into two-dimensional time–frequency representations, enabling the application of image-based deep learning models (Ahmed & Nandi, 2022; C. Wang et al., 2024; Ren & Guo, 2024). In this context, convolutional neural networks (CNNs) have demonstrated strong performance by learning discriminative spatial patterns from spectrogram-like representations (Gültekin et al., 2022; Y. Zhang et al., 2020; Wu et al., 2023; Łuczak, 2024).

A variety of signal-to-image transformation techniques have been explored in the literature. (C. Wang et al., 2024) provide a comprehensive review of these approaches, highlighting methods such as Short-Time Fourier Transform (STFT), Continuous Wavelet Transform (CWT), and Markov Transition Field (MTF), combined with deep learning architectures. Their analysis indicates that CNN-based models consistently achieve high diagnostic performance across multiple datasets. Complementing this perspective, (Yang et al., 2015) propose a parameterized time–frequency analysis method tailored for non-stationary vibration signals, in which transform parameters—such as those controlling the time–frequency resolution and the estimation of instantaneous frequency and amplitude— are adaptively estimated based on the instantaneous characteristics of the signal. The method includes parameter estimation and component separation steps to better represent mono- and multi-component signals. Experimental results on rotor systems demonstrate improved time–frequency concentration and more accurate extraction of fault-related features.

Building on these representations, several studies focus on designing effective CNN-based architectures for fault classification. (Xu et al., 2019) transform vibration signals into time–frequency images and employ an optimized VGG16-based model, achieving accuracies above 99.9% on the Case Western Reserve University (CWRU) (Case Western Reserve University, 2014) bearing dataset, using randomly split samples from the same experimental acquisitions. Similarly, (Wu et al., 2023) utilize CWT-based images combined with squeeze-and-excitation residual networks, reporting an accuracy of 96.42% on the same dataset under comparable sampling conditions. (Nie et al., 2025) further enhance CNN performance by incorporating image enhancement and attention mechanisms in the Image Enhancement and Improved Convolutional Neural Network (IEICNN) framework, reaching an average accuracy of 99.14% with a 30%–70% train–test split.

Other works explore multi-sensor data and feature fusion strategies to improve robustness. Gültekin et al. (Gültekin et al., 2022) propose a multi-sensor framework in which STFT-based representations are processed by multiple ResNet-18 networks, followed by feature-level fusion, achieving improved diagnostic performance on the Paderborn University (PU) (Lessmeier et al., 2016; Konstruktions- und Antriebstechnik (KAt), Universität Paderborn, 2026) with a 60%–20%–20% data split. In a related direction, (Fan et al., 2025) combine the Local Maximum Synchrosqueezing Wavelet Transform with a multiscale CNN and channel attention mechanisms, reporting accuracies above 98% on CWRU, PU, and Huazhong University of Science and Technology (HUST) (Thuan & Hong, 2023) datasets.

In addition to conventional transformation pipelines, alternative signal encoding strategies have also been investigated.

(Łuczak, 2024), for instance, convert Inertial Measurement Unit (IMU) vibration signals directly into grayscale and RGB images, generating 7,680 samples and achieving accuracies between 99.88% and 99.97% using an 80%–20% train–test split.

Despite the high diagnostic accuracies reported across these studies, a common limitation lies in the evaluation methodology. Most works adopt random or stratified data splitting strategies in which samples from the same experimental acquisitions or operating conditions may appear in both training and testing sets. This practice can introduce similarity bias, allowing models to learn experiment-specific patterns rather than truly generalizable fault characteristics. Consequently, the reported results may be overly optimistic and may not accurately reflect model robustness when applied to unseen operating conditions or independent datasets.

3. CONCEPTUAL BACKGROUND

This section reviews important concepts relevant to this work. It first summarizes the similarity bias problem, then presents the two most commonly used methods for two-dimensional representation of vibration data in the time–frequency domain, and finally briefly describes the transfer learning approach.

3.1. The Similarity Bias Problem

In machine learning, bias refers to systematic distortions that compromise reliable model evaluation and generalization. Several forms of bias have been identified, including historical, representational, and measurement biases (Mehrabi, Morstatter, Saxena, Lerman, & Galstyan, 2021). In vibration-based intelligent fault diagnosis, a particularly critical form is *similarity bias*, first systematically investigated in experimental studies focusing on vibration datasets (dos Santos Varejão et al., 2025; Venturim & de Assis Boldt, 2021; Rauber et al., 2021). This type of bias emerges when samples are highly correlated due to originating from the same acquisition process or similar operating conditions.

Similarity bias occurs when strongly related signal segments are simultaneously included in both training and testing sets, leading models to learn acquisition-specific patterns rather than true fault characteristics. As a consequence, classification results may appear overly optimistic and fail to reflect real-world generalization capability (Venturim & de Assis Boldt, 2021). From a probabilistic perspective, a dataset is considered free from similarity bias when training and testing samples are conditionally independent given the data–generating parameters; however, this assumption is often violated in practice due to segmentation methods and limited variability in operating conditions (dos Santos Varejão et al., 2025).

Experimental studies using widely adopted benchmark datasets, such as CWRU dataset, have demonstrated the severity of similarity bias in conventional evaluation protocols. Random data splits frequently yield near-perfect classification accuracy, while more rigorous splitting strategies based on acquisition conditions, loads, or fault severities lead to noticeable performance degradation, highlighting the impact of biased evaluation procedures (Rauber et al., 2021; dos Santos Varejão et al., 2025). These findings reinforce the importance of adopting more reliable validation strategies to ensure meaningful performance assessment.

Similarity bias should be distinguished from related problems studied in out-of-distribution (OOD) generalization and in domain adaptation and generalization. OOD generalization requires an explicit difference between the training and test distributions (Liu et al., 2021). This may coexist with similarity bias, since a shift in the distribution does not prevent samples from the same acquisition from being shared between partitions. Domain generalization requires that the test data come from domains different from those used in training (J. Wang et al., 2022; Gulrajani & Lopez-Paz, 2020), and it has been applied to bearing fault diagnosis across operating conditions and across machines (Jia, Li, Wang, Sun, & Deng, 2023). The notion of domain is open: it most often refers to operating conditions, but it may also refer to individual equipment or types of equipment. Regardless of how the domain is defined, using distinct domains for training and testing eliminates similarity bias. Domain adaptation, in turn, assumes access to (typically unlabeled) data from the target domain during training (Ben-David et al., 2010; Jia et al., 2023); since the data used for adaptation and the data used for testing come from the same acquisition, similarity between them is usually present. The protocol adopted in this work shares with domain generalization the principle that examples from the same acquisition condition are never split between training and test. Its purpose, however, is different: it provides a leakage-free evaluation of supervised classifiers, rather than learning representations that are invariant across domains.

Motivated by these observations, the present work adopts an unbiased data splitting strategy in which all samples originating from the same acquisition condition are grouped and assigned exclusively to either the training or testing set, following established recommendations in the literature (dos Santos Varejão et al., 2025). This acquisition-aware approach enables the evaluation protocol to focus on true model generalization rather than exploiting hidden correlations, improving results interpretability and promoting fair and reproducible assessments of vibration-based fault diagnosis systems.

3.2. Time–Frequency Domain Feature Representation

Time–frequency analysis maps one-dimensional non-stationary signals into a two-dimensional space that jointly

captures temporal evolution and spectral content, revealing dynamics that neither the time nor the frequency domain describes adequately on its own (Q. Zhang & Deng, 2023; X. Zhang et al., 2021). Representative techniques include the STFT (Q. Zhang & Deng, 2023), the CWT (Y. Wang et al., 2010), and the S-transform (Shafi et al., 2009). This study focuses on the STFT and the CWT.

3.2.1. Short-Time Fourier Transform (STFT)

The STFT applies a sliding window to the signal and computes a Fourier transform of each segment, yielding a localized spectrum that shows how spectral components evolve over time (Y. Zhang et al., 2020; Jang et al., 2024; Shafi et al., 2009):

$$\mathrm{STFT}(t,f) = \int_{-\infty}^{\infty} x(\tau)\, w(\tau - t)\, e^{-j2\pi f \tau}\, d\tau, \qquad (1)$$

where $x(\tau)$ is the signal, $w(\cdot)$ is the window function, t is the time shift, and f is the frequency. The window length controls the resolution trade-off (shorter windows favor temporal resolution, longer ones favor frequency resolution), and the hop size defines the temporal granularity (Y. Zhang et al., 2020; Łuczak, 2024). In this study, each one-second segment is first standardized (zero mean and unit variance), and the STFT is computed with a Hann window (the default window of `scipy.signal.stft`) of 512 samples, an overlap of 384 samples (75%, i.e., a hop size of 128 samples) and a 512-point FFT, which yields 257 frequency bins. The magnitude spectrogram is log-compressed ($\log(1 + |\cdot|)$), clipped between its 1st and 99th percentiles and min–max normalized to $[0, 1]$. The same parameters were used for all datasets; only the sampling rate changes. The resulting spectrograms are used as input images for the CNN models (Figure 1).

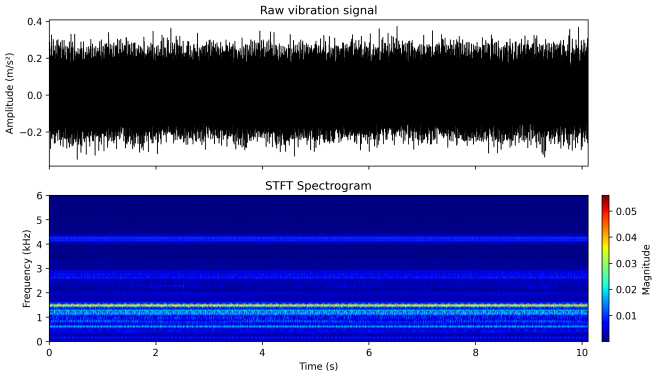


Figure 1. Example of time–frequency transformation using STFT. The figure illustrates the conversion of a raw vibration signal sample from the CWRU dataset into its corresponding time–frequency representation.

3.2.2. Continuous Wavelet Transform (CWT)

Unlike the STFT, which relies on a fixed window, the CWT provides a multi-resolution representation by scaling and translating a mother wavelet, which favors the detection of localized and transient components (Sanz et al., 2007; Y. Wang et al., 2010; Łuczak, 2024; T. Wang et al., 2021):

$$\mathrm{CWT}(a,b) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} x(t)\, \psi^*\left(\frac{t-b}{a}\right) dt, \qquad (2)$$

where ψ is the mother wavelet, ψ^* its complex conjugate, a the scale (related to frequency), and b the translation (temporal localization). In this study, the mother wavelet is the real Morlet wavelet (`morl` in PyWavelets, central frequency $f_c \approx 0.8125$). Each one-second segment is standardized (zero mean and unit variance) and analyzed with 256 scales logarithmically spaced between $a_{\min} = 1$ and $a_{\max} = 256$, i.e., $a_k = 256^{k/255}$, $k = 0, \ldots, 255$. The pseudo-frequency associated with each scale is $f = f_c f_s / a$, so the scale range corresponds to approximately 38–9750 Hz for CWRU-12kHz, 152–39000 Hz for CWRU-48kHz, 155–39673 Hz for MFPT and 203–52000 Hz for PU. The scalogram is the log-compressed magnitude of the coefficients ($\log(1 + |\cdot|)$), clipped between its 1st and 99th percentiles and min–max normalized to $[0, 1]$. The resulting scalograms are also used as CNN inputs (Figure 2).

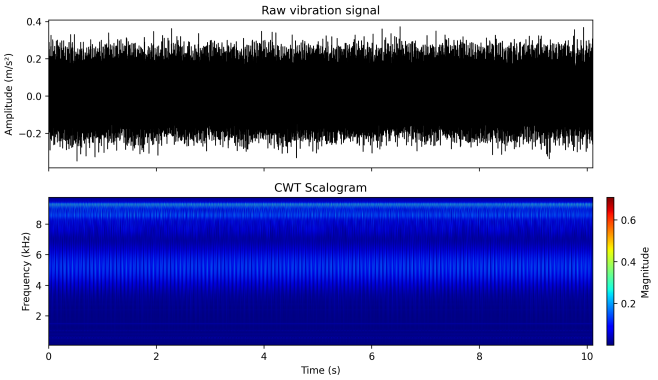


Figure 2. Example of time–frequency transformation using CWT. The figure illustrates the conversion of a raw vibration signal sample from the CWRU dataset into its corresponding time–frequency representation.

3.3. Transfer Learning

Transfer learning addresses the limited dataset sizes and high training costs of deep learning-based fault diagnosis (Z. Chen et al., 2020). Instead of training CNNs from scratch, it adapts knowledge learned on large-scale datasets such as ImageNet (Deng et al., 2009) to smaller task-specific ones. Generic visual features (e.g., edges and textures) can be transferred to vibration signals converted into time–frequency images (Wu et al., 2023; Arun Balaji et al., 2025). Fine-tuning updates part or all of the pre-trained layers: in partial fine-tuning, the

lower layers are frozen to preserve generic features while the higher layers are retrained; in full fine-tuning, all layers are retrained. Partial fine-tuning balances efficiency and performance, whereas full fine-tuning allows deeper adaptation at a higher computational cost and risk of overfitting (Z. Chen et al., 2020). Both strategies are evaluated in this work, together with training from scratch as a reference, to analyze robustness, generalization, and sensitivity to similarity bias under similarity-aware data splitting.

4. A TIME–FREQUENCY REPRESENTATION AND TRANSFER LEARNING BASED MODEL FOR INTELLIGENT FAULT DIAGNOSIS

This study investigates fault diagnosis methods that integrate two-dimensional time–frequency signal representations and deep neural networks with transfer learning.

Figure 3 illustrates an overview of the proposed approach. Initially, raw vibration signals acquired from rotating machinery are segmented into fixed-length windows. Each signal segment is then transformed into a two-dimensional time–frequency representation using either the STFT or the CWT. These representations enable the extraction of joint temporal and spectral characteristics that are strongly associated with different mechanical fault conditions.

Deep neural networks and convolutional neural networks are employed for fault classification. Distinct time–frequency representations are generated and used independently to train separate diagnostic models, allowing each network to specialize in learning discriminative features from a specific signal-to-image transformation. Pre-trained architectures initialized with weights learned from the ImageNet dataset are adopted as feature extractors, leveraging knowledge acquired from large-scale natural image classification. Transfer learning is applied by adapting the network parameters to the vibration-based time–frequency images, while fine-tuning is performed on the target datasets to enhance task-specific discrimination. This strategy enables efficient learning, accelerates convergence, and improves robustness, particularly when the amount of labeled training data is limited.

Finally, the fine-tuned models are employed to predict the condition of equipment in operation, using vibration signals that are acquired and subsequently transformed into the time –frequency representation required by the model.

4.1. Deep Neural Networks and Convolutional Neural Networks

Deep Neural Networks (DNNs) learn hierarchical representations through multiple nonlinear transformations, and Convolutional Neural Networks (CNNs) are particularly effective for structured inputs such as images and time–frequency representations (Ren & Guo, 2024; Arun Balaji et al., 2025). A CNN combines convolutional layers for feature extraction, nonlinear activations (e.g., ReLU), pooling layers for spatial downsampling, and fully connected layers for classification (Wu et al., 2023; L. Wang & Zhang, 2023); local receptive fields and parameter sharing allow it to learn spatial patterns with manageable complexity. CNNs have been widely applied in computer vision, anomaly detection, and vibration-based fault diagnosis (Wu et al., 2023), and here they classify faults from vibration signals converted into time–frequency images.

EfficientNetV2–S (Tan & Le, 2021) is adopted as the backbone network due to its favorable balance between accuracy and computational efficiency, which is relevant in PHM applications, where models are often deployed on resource-constrained hardware. It consists of a convolutional stem followed by Fused-MBConv and MBConv blocks (Table 3). To assess whether a lighter model is sufficient, MobileNetV3-Large (Howard et al., 2019) is used as an alternative backbone. It is built on inverted residual (MBConv) blocks with depthwise convolutions, squeeze-and-excitation modules in part of the blocks, and the h-swish activation, and its architecture was obtained through a combination of hardware-aware neural architecture search and the NetAdapt algorithm, aiming at resource-constrained devices. Its layer-wise architecture is given in Table 2, and Table 1 compares both backbones: MobileNetV3-Large has about 4.8 times fewer parameters and about 12 times fewer multiply-accumulate operations than EfficientNetV2–S. Both are adapted to the task by replacing the final classification layer with a task-specific classifier and are evaluated under the configurations described in Section 4.2.

Table 1. Backbones compared in this study. Parameters and GMACs were computed for the CWT representation (256×256 input) and the four-class PU dataset. The MobileNetV3-Large classifier includes an additional hidden layer with 1280 units.

Property	EfficientNetV2–S	MobileNetV3-Large
Main blocks	Fused-MBConv, MBConv	MBConv (inverted residual)
Activation	SiLU	ReLU, h-swish
Total parameters	20.18 M	4.21 M
GMACs per sample	3.787	0.304
Feature dimension	1280	960

4.2. Transfer Learning and Fine-Tuning

Transfer learning is applied by initializing the backbones with weights pre-trained on ImageNet (Deng et al., 2009). The convolutional backbone acts as a feature extractor, and the original classification head is replaced by a task-specific output layer whose dimension equals the number of fault classes of the target dataset. For EfficientNetV2–S, the backbone produces a 1280-dimensional feature vector after global average pooling. Two fine-tuning strategies

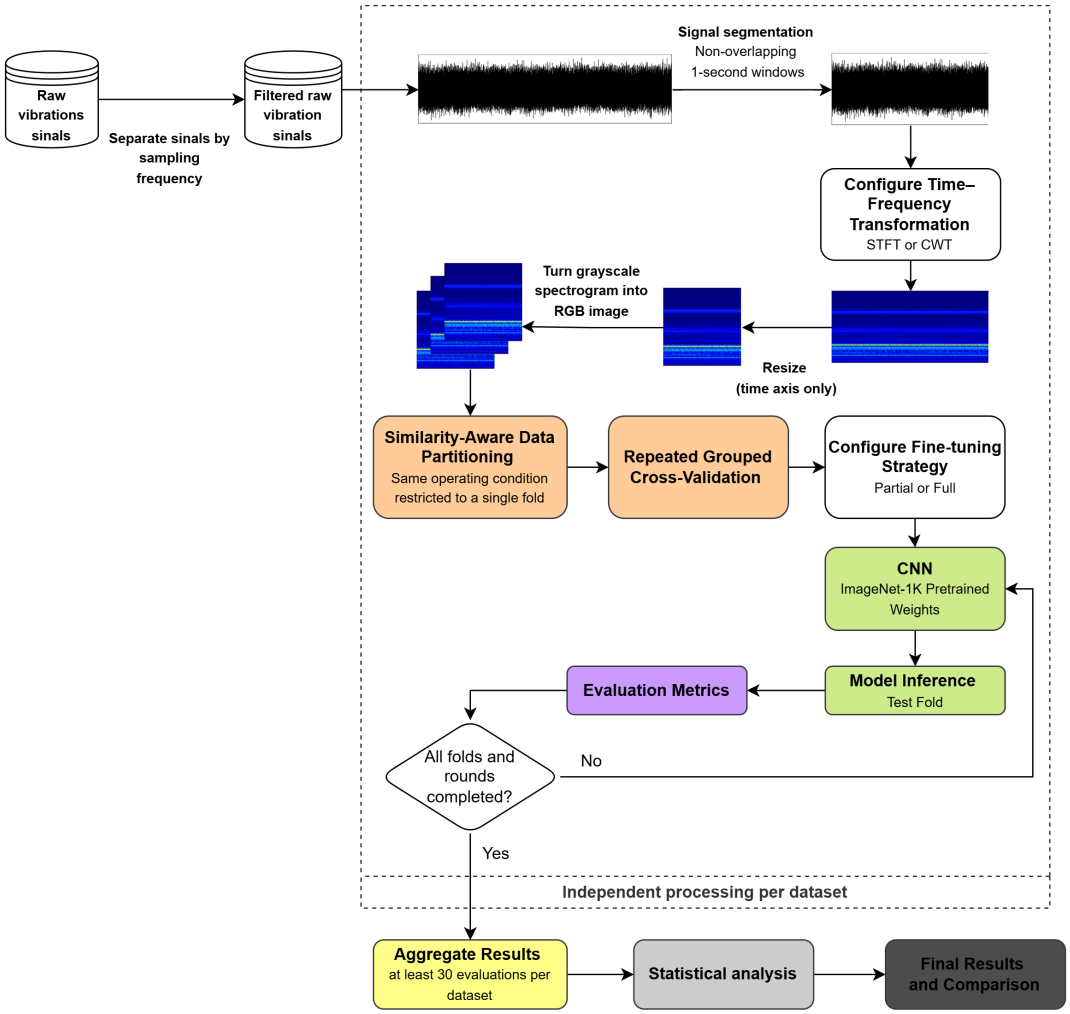


Figure 3. Overview of the proposed fault diagnosis framework. Raw vibration signals are segmented and transformed into time –frequency representations; only the time axis is resized, to a common length of 256 frames, while the frequency (STFT) or scale (CWT) axis is kept at its native resolution, yielding 257×256 images for STFT and 256×256 for CWT (see Section 4.3 for details). These representations are then used as input to pre-trained deep neural networks. The models are fine-tuned using transfer learning under different configurations (partial and full fine-tuning) for fault classification.

are considered for both backbones (EfficientNetV2–S and MobileNetV3-Large):

- *Partial fine-tuning:* the lower blocks of the backbone remain frozen, preserving the low-level visual features learned from ImageNet (Deng et al., 2009), while the last n blocks of the feature extractor and the classification layer are updated, with n tuned between one and three (Section 4.5; selected values in Appendix A). The last block of the feature extractor is the 1×1 convolutional head; for EfficientNetV2–S, $n = 2$ therefore updates Stage 6 and the head, and $n = 3$ also Stage 5 (Table 3), whereas for MobileNetV3-Large n counts the individual inverted residual blocks preceding the head (Table 2). This adapts high-level representations to the fault classification task while reducing the risk of overfitting with limited training data.
- *Full fine-tuning:* all parameters are initialized with the ImageNet weights and updated during training, allowing full adaptation to the target domain at the cost of higher computational complexity and a higher risk of overfitting.

In addition, to isolate the contribution of transfer learning, EfficientNetV2–S is also trained from scratch, with randomly initialized weights and no ImageNet pre-training, with all layers trainable and under the same training protocol as full fine-tuning, but with its own hyperparameter search space (Section 4.5).

The experimental evaluation compares these configurations to assess the trade-off between representation adaptability

Table 2. Layer-wise architecture of the MobileNetV3-Large network for a 256×256 input (CWT); for STFT, the input is 257×256. Bneck: inverted residual block; SE: squeeze-and-excitation; RE: ReLU; HS: h-swish.

Stage	Operator	Kernel	Channels	Output Size
Input	–	–	3	256×256
Stem	Conv2D + BN + HS	3×3	16	128×128
1	Bneck $\times 1$ (RE)	3×3	16	128×128
2	Bneck $\times 2$ (RE)	3×3	24	64×64
3	Bneck $\times 3$ (SE, RE)	5×5	40	32×32
4	Bneck $\times 4$ (HS)	3×3	80	16×16
5	Bneck $\times 2$ (SE, HS)	3×3	112	16×16
6	Bneck $\times 3$ (SE, HS)	5×5	160	8×8
Head	Conv2D + BN + HS + GAP	1×1	960	1×1
Output	FC (1280) + HS + FC	–	N_{classes}	–

Table 3. Layer-wise architecture of the EfficientNetV2–S network. The listed dimensions correspond to the 256×256 input case (CWT); for STFT, the input is 257×256.

Stage	Operator	Kernel	Channels	Output Size
Input	–	–	3	256×256
Stem	Conv2D	3×3	24	128×128
1	Fused-MBConv	3×3	24	128×128
2	Fused-MBConv	3×3	48	64×64
3	Fused-MBConv	3×3	64	32×32
4	MBConv	3×3	128	16×16
5	MBConv	3×3	160	16×16
6	MBConv	3×3	256	8×8
Head	Conv2D + BN + SiLU + GAP	1×1	1280	1×1
Output	Fully Connected	–	N_{classes}	–

and training stability, and compares the two backbones in terms of performance and computational cost, in vibration-based fault diagnosis.

This work adopts the unbiased, similarity-aware resampling strategy proposed in (dos Santos Varejão et al., 2025) for evaluating vibration-based fault diagnosis. The remaining aspects of the experimental methodology described in this section – the time–frequency representations, network architectures, training protocol, and computational cost analysis – are introduced in the present study, as detailed in Section 1. Only four datasets that allow experiments without similarity bias are considered: CWRU12, CWRU48, MFPT, and PU.

The following subsections describe the main aspects of the experimental methodology: Datasets (Section 4.3), Resampling Strategy (Section 4.4), Training Protocol and Hyperparameter Optimization (Section 4.5), and Evaluation Metrics and Hypothesis Test (Section 4.6).

4.3. Datasets

This study employs multiple publicly available datasets widely used in machine learning research for intelligent fault diagnosis using vibration signals. The results obtained are compared with those reported in (dos Santos Varejão et al., 2025), enabling a consistent and fair evaluation of the proposed methodology. While previous studies rely primarily on raw time-domain signals, the present work extends this setting by incorporating time–frequency representations to investigate their impact on feature representation and classification performance.

Following the experimental protocol established in (dos Santos Varejão et al., 2025), vibration signals are segmented into fixed-length, non-overlapping windows, where each segment corresponds to one second of signal acquisition.

To maintain consistency in the input dimensionality and experimental conditions, datasets originally collected under multiple sampling rates are separated and treated as independent datasets, following the preprocessing strategy described in (dos Santos Varejão et al., 2025).

Each time-domain segment is subsequently converted into a time–frequency representation using either the STFT or the CWT, resulting in two-dimensional inputs (spectrograms or scalograms). The magnitude of each representation is log-compressed, clipped to the 1st–99th percentile range, and min–max scaled to $[0, 1]$. For the STFT, computed with a 512-sample window and 75% overlap, this yields 257 frequency bins, which are kept at their native resolution, while the time axis is resampled to 256 frames, producing 257×256 images. For the CWT, the scale axis is defined with 256 log-spaced scales, matching the target size directly, so only the time axis is resampled to 256 frames, producing 256×256 images. Only the time axis is resampled in either case: the number of frequency bins (STFT) or scales (CWT) is fixed by construction, independent of the sampling rate or signal length, and is therefore already identical across all four datasets. The time axis, in contrast, is the one that requires resampling, since the segments correspond to a fixed *duration* rather than a fixed *number of samples* (see above); as the datasets are acquired at different sampling rates (12, 48, and 64 kHz, and 48,828 Hz), each 1-second segment contains a different number of raw samples, and consequently a different native number of STFT/CWT time frames, which must be resampled to a common size to obtain fixed-size network inputs across datasets. This choice also avoids interpolating across the frequency (STFT) or scale (CWT) axis, which carries directly computed, physically meaningful spectral values whose interpolation could blur the narrow spectral peaks associated with bearing fault characteristic frequencies; the time axis, in contrast, tolerates resampling well due to the 75% overlap between adjacent STFT frames. The resulting images (257×256 for STFT, 256×256 for CWT) are replicated across three channels to be compatible with image-based models pre-trained on ImageNet (Deng et al., 2009). Both backbones used in this study end with an adaptive average pooling layer before the classification head and therefore accept these input shapes

without any architectural modification.

4.4. Resampling Strategy

The resampling strategy adopted in this study follows the unbiased evaluation protocol proposed in (dos Santos Varejão et al., 2025), ensuring a fair and consistent comparison with the reference benchmark. Under this protocol, data partitioning is defined individually for each dataset, taking into account its specific characteristics in order to reduce similarity bias introduced by segmenting signals into windows under the same operating condition. Samples originating from the same acquisition setup are grouped and assigned to a single fold, ensuring that no data from a given condition is shared between training and test sets. This design prevents the model from being evaluated on highly similar samples and reduces the risk of overly optimistic performance estimates (dos Santos Varejão et al., 2025).

Table 4 summarizes, for each dataset, the acquisition condition used to define the groups that form the folds. The CWRU recordings, acquired at sampling rates of 12 kHz and 48 kHz, are treated as two independent datasets (CWRU12 and CWRU48). In both, the condition is the motor load (0, 1, 2, and 3 HP), resulting in four folds. In PU, the signals are sampled at 64 kHz, and a condition is a combination of rotational speed, load torque, and radial force (four conditions), also resulting in four folds. In MFPT, only the recordings sampled at 48,828 Hz are used, since those at 97,656 Hz were all acquired under the same working condition, which makes an unbiased division impossible. Here, a condition is the applied load (0 to 300 lbs); as the two lowest loads (0 and 25 lbs) contain examples of a single class each, they are merged into one condition so that every fold contains examples of both classes, resulting in seven folds.

Table 4. Acquisition condition used to form the folds, number of folds, number of rounds, and total number of fold evaluations (folds $\times$ rounds) for each dataset.

Dataset	Fold condition	Folds	Rounds	Total
CWRU12	Motor load	4	8	32
CWRU48	Motor load	4	8	32
PU	Speed, torque, radial force	4	8	32
MFPT	Load (0 and 25 lbs merged)	7	5	35

Following the recommendations of (dos Santos Varejão et al., 2025), multiple rounds of cross-validation are employed to improve the robustness of the evaluation. The multiple rounds are performed to obtain more than 30 performance measurements, a condition met by all datasets (Table 4). This choice is motivated by statistical considerations, as sample sizes of 30 or more are commonly regarded as sufficient to approximate asymptotic normality, enabling more reliable estimation of performance variability and supporting statistically meaningful comparisons.

In each round, the grouping constraint is preserved, such that samples belonging to the same class and acquisition condition remain within a single fold. Different rounds are generated by permuting, for each class, the assignment of its acquisition conditions across the folds, producing distinct data splits. Each round differs in at least one operating condition associated with a given class, ensuring diversity in the evaluation while maintaining the unbiased setting (dos Santos Varejão et al., 2025). A detailed description of the partitioning algorithm is beyond the scope of this work and can be found in (dos Santos Varejão et al., 2025).

4.5. Training Protocol and Hyperparameter Optimization

All models were implemented in PyTorch and trained under the protocol summarized in Table 5, using the same folds and rounds described in the previous subsection. The hyperparameters that most affect optimization were tuned with Optuna (Akiba, Sano, Yanase, Ohta, & Koyama, 2019) using its default Tree-structured Parzen Estimator (TPE) sampler, over the search spaces listed in Table 6. A separate search was run for each combination of dataset, time-frequency representation, backbone, and training strategy. Each search comprised 20 trials of 10 epochs, executed on a single split (first fold of the first round), and minimized the validation loss computed on 20% of the training data of that fold, held out at random (this internal train/validation split is not grouped by acquisition condition, unlike the external test folds). The test data of the fold were not used in the search. The hyperparameters selected in each search were then used for all folds and rounds of the corresponding experiment, and are reported in Appendix A.

Partial and full fine-tuning shared the same search space. Since the backbone starts from ImageNet weights, separate learning rates were tuned for the new classification layer and for the backbone, allowing the pre-trained features to be updated more conservatively. For partial fine-tuning, the number of unfrozen blocks was also tuned, between one and three. For full fine-tuning and training from scratch, all layers are trainable, so this parameter does not apply.

Training from scratch used a different search space. Because all weights are randomly initialized, there are no pre-trained features to preserve, and a single learning rate was tuned for the whole network, over a higher range. Batch size 8 was excluded, since Batch Normalization statistics estimated from very small batches are unstable at the beginning of training from random weights. The optimizer was also included in the search, choosing between AdamW and SGD with Nesterov momentum.

For MFPT, batch size 4 was added to the search range of partial and full fine-tuning, since each training split contains only 29 samples after the validation hold-out, and larger batches

would leave very few parameter updates per epoch.

Table 5. Training protocol used in all experiments. Hyperparameters marked as tuned were selected with Optuna (Table 6), and their selected values are listed in Appendix A.

Item	Setting
Software	Python 3.13.15, PyTorch 2.11.0 (CUDA 12.8), Optuna 5.0.0
Hardware	NVIDIA Tesla T4 GPU (15 GB)
Initialization	ImageNet weights (partial and full fine-tuning) or random weights (from scratch)
Input size	257×256 (STFT) / 256×256 (CWT)
Optimizer	AdamW; for training from scratch, the search could also select SGD with Nesterov momentum
Initial learning rate	Tuned; separate rates for the classifier and the backbone (partial and full fine-tuning) or a single rate for the whole network (from scratch)
Weight decay	Tuned
Batch size	Tuned (Table 6); range limited by GPU memory (Section 5.1) and, for MFPT, extended to smaller batches due to the training-set size; selected independently per dataset, representation, backbone, and training strategy
Loss function	Cross-entropy with label smoothing (tuned)
LR scheduler	None (constant learning rate)
Epochs	30
Early stopping	None (all 30 epochs are run)
Model selection	Epoch with the lowest validation loss (20% of the training data held out); weights kept in memory
New classification layer	PyTorch default initialization (`nn.Linear`)
Seeds	Fixed per fold via `torch.manual_seed` and `np.random.seed`, using seed = 42 + fold index
Data augmentation	None (inputs are only normalized)

4.6. Evaluation Metrics and Hypothesis Test

As in the work of (dos Santos Varejão et al., 2025), the performance of the classification models is evaluated using the confusion matrix and two commonly adopted metrics: Balanced Accuracy ($BAcc$) and Macro F1-score ($F1_M$). Let c denote the number of classes in the classification problem. For each class j, the true positives, false positives, and false negatives are denoted by TP_j, FP_j, and FN_j, respectively, and Recall and Precision are defined as:

$$\text{Recall}_j = \frac{TP_j}{TP_j + FN_j}, \qquad \text{Precision}_j = \frac{TP_j}{TP_j + FP_j}. \tag{3}$$

The evaluation criteria adopted in this study are defined as follows:

1. **Balanced Accuracy** ($BAcc$)

$$BAcc = \frac{1}{c}\sum_{j=1}^{c} \text{Recall}_j \tag{4}$$

Balanced Accuracy is defined as the arithmetic mean of the per-class Recall, generalizing the binary definition – the average between sensitivity and specificity – to the multiclass setting, in which each class's recall plays the role that sensitivity and specificity play for the positive and negative classes, respectively, in the binary case. Like the Macro F1-score below, $BAcc$ gives equal importance to each class regardless of its number of samples, providing a more reliable evaluation in imbalanced datasets.

2. **Macro F1-score** ($F1_M$)

The F1-score for each class is computed as the harmonic mean of its Precision and Recall:

$$F1_j = \frac{2 \cdot \text{Precision}_j \cdot \text{Recall}_j}{\text{Precision}_j + \text{Recall}_j} \tag{5}$$

The Macro F1-score is then calculated as the arithmetic mean of the F1-scores across all classes:

$$F1_M = \frac{1}{c}\sum_{j=1}^{c} F1_j \tag{6}$$

The Macro F1-score gives equal importance to each class, regardless of its number of samples, and is particularly useful for evaluating classification performance in imbalanced datasets.

Table 6. Optuna search spaces. Each search used 20 trials, minimized the validation loss, and trained each trial for 10 epochs on a single split (first round, first fold). "–" indicates that the hyperparameter does not apply.

Hyperparameter	Partial / Full	From scratch	Scale
Classifier learning rate	$[10^{-5}, 10^{-2}]$	–	log
Backbone learning rate	$[10^{-6}, 10^{-3}]$	–	log
Learning rate (single)	–	$[10^{-4}, 3 \times 10^{-2}]$	log
Weight decay	$[10^{-6}, 10^{-2}]$	$[10^{-6}, 10^{-2}]$	log
Batch size[a]	$\{8, 16, 32, 64\}$	$\{16, 32, 64\}$	categorical
Label smoothing	$[0, 0.2]$	$[0, 0.2]$	linear
Unfrozen blocks[b]	$\{1, 2, 3\}$	–	integer
Optimizer	–	{AdamW, SGD}	categorical

[a]For MFPT, batch size 4 was also included for partial and full fine-tuning, i.e., $\{4, 8, 16, 32, 64\}$, due to the small training set. [b]Partial fine-tuning only.

These evaluation metrics offer a comprehensive assessment of the models' capabilities in distinguishing between the classes under investigation.

Statistical hypothesis testing was conducted following the same protocol adopted in (dos Santos Varejão et al., 2025). For pairwise comparisons among models evaluated under the same data splits, the Wilcoxon signed-rank test (Wilcoxon, 1992) was employed. This non-parametric approach enables the assessment of statistically significant differences between paired model performances without assuming normality of the score distributions, providing a robust and fair comparison framework across experimental conditions.

5. RESULTS

This section presents the experimental results obtained using the proposed approach, alongside comparisons with baseline methods from the literature. Different datasets and experimental settings were considered in the evaluation, following the unbiased data division protocol described earlier. Performance is analysed using standard metrics and statistical tests are applied to evaluate the significance of any observed differences.

5.1. Training Environment and Computational Cost

All experiments were run on Google Colab using an NVIDIA T4 GPU (15 GB VRAM), which offered a suitable balance between processing efficiency, cost, and training time. GPU memory limited the largest batch size explored during hyperparameter search to 64. Training times, inference latency, and peak GPU memory reported in Table 7 refer to this GPU.

Table 7 reports the computational cost of each backbone under partial, full, and (for EfficientNetV2–S) from-scratch training, measured on the CWT representation of the PU dataset, consistent with Table 1. Note that training time and peak GPU memory are reported jointly with batch size (see caption), rather than being purely a function of architecture or fine-tuning strategy.

5.2. Benchmark Comparison with Baseline Methods

Tables 8 and 9 summarize the quantitative comparison between the baseline classifiers and the proposed models. The comparison covers two backbones (EfficientNetV2–S and MobileNetV3), two time–frequency representations (STFT and CWT), and two fine-tuning strategies (partial and full). For EfficientNetV2–S, two additional configurations trained without ImageNet pre-trained weights are also reported, in order to isolate the contribution of transfer learning. The baseline methods correspond to the best obtained in (dos Santos Varejão et al., 2025), namely: Deep Neural Networks (DNN, XResNet18-1D) for CWRU-12kHz, Random Forest (RF) for CWRU-48kHz and PU, and Nearest Neighbor (1-NN) for MFPT.

Among these baselines, RF and 1-NN are classical shallow classifiers trained on hand-crafted statistical features extracted from the raw time-domain vibration signals (e.g., Mean, Standard Deviation, RMS value, Kurtosis, and energy-envelope-based features), whereas XResNet18-1D is a deep learning model applied directly to the raw time-domain signals. The RF and 1-NN baselines therefore represent the classical feature-based approach, and their comparison with the proposed 2D time-frequency representations isolates the gain provided by the latter. All baselines were evaluated under the same similarity-aware partition protocol, folds, and repetitions as the proposed models.

Performance is reported as the mean and standard deviation across all folds and rounds. Figure 4 illustrates the distribution of Macro F1-score and Balanced Accuracy across folds and rounds, providing a visual complement to the statistical summary.

For the CWRU-12kHz dataset, the best results were obtained by MobileNetV3 with STFT representations and full fine-tuning (0.9553 Macro F1, 0.9552 Balanced Accuracy), followed by the same backbone with partial fine-tuning (0.9521 for both metrics). All configurations exceeded the baseline (0.8841 Macro F1, 0.8872 Balanced Accuracy), and the STFT representation outperformed the CWT for both backbones. The dispersion across folds was also reduced with respect to the baseline, from 0.0639 to 0.0316 in Macro F1 for the best configuration.

For CWRU-48kHz, EfficientNetV2–S with STFT representations achieved the highest scores, with partial fine-tuning reaching 0.9947 Macro F1 and 0.9956 Balanced Accuracy, and full fine-tuning close behind (0.9910 and 0.9911). These values represent a substantial gain over the baseline (0.7764 Macro F1, 0.7846 Balanced Accuracy), accompanied by a marked reduction in variability (0.0169 vs. 0.0708 in Macro F1). As in CWRU-12kHz, the STFT representation outperformed the CWT for both backbones, and the gap between representations is wider in this dataset.

For the PU dataset, EfficientNetV2–S with STFT and full fine-tuning obtained the best results (0.7243 Macro F1, 0.7625 Balanced Accuracy), followed by MobileNetV3 with CWT and full fine-tuning (0.7166 and 0.7542). Both exceeded the baseline (0.6273 Macro F1, 0.6730 Balanced Accuracy). Unlike the CWRU datasets, however, the dispersion was not reduced: the standard deviation of the best configuration (0.2001 in Macro F1) is comparable to that of the baseline (0.1993), and several configurations present higher variability than the baseline. Absolute performance on this dataset remains considerably lower than on the CWRU datasets for all methods, including the baseline.

For the MFPT dataset, the baseline achieved the highest scores (0.9688 Macro F1, 0.9714 Balanced Accuracy), and no proposed configuration surpassed it. The closest configurations were EfficientNetV2–S with STFT and partial fine-tuning (0.9234 Macro F1) and with CWT and full fine-tuning (0.9381 Balanced Accuracy). In contrast with the other datasets, the proposed models also exhibited substantially higher variability than the baseline on MFPT (e.g., 0.1618 vs. 0.0949 in Macro F1 for the best configuration), and the dispersion is high across all configurations, reaching 0.3222 for MobileNetV3 with CWT and partial fine-tuning. This dataset is the smallest of the four and contains only two classes, which may explain both the strong performance of the simpler distance-based baseline and the instability of the deep models across folds.

Table 7. Computational cost per backbone and fine-tuning strategy (PU dataset, CWT representation, batch size 1 for inference latency). Batch size (training) was tuned independently for each configuration by Optuna and strongly affects peak GPU memory and training time. Values are mean ± std over folds and rounds.

Backbone	Strategy	Batch	Train. params	GMACs/sample	Epoch time (s)	Latency, bs=1 (ms)	Peak mem., train/infer (MB)
EfficientNetV2–S	Partial	16	14.90 M	3.787	52.27 ± 0.09	18.73 ± 0.96	1098 ± 245 / 592 ± 2
EfficientNetV2–S	Full	64	20.18 M	3.787	83.10 ± 0.08	18.55 ± 0.37	12721 ± 0 / 1130 ± 119
EfficientNetV2–S	Scratch	16	20.18 M	3.787	98.92 ± 1.87	18.83 ± 0.74	3491 ± 239 / 869 ± 168
MobileNetV3-Large	Partial	64	2.99 M	0.304	9.59 ± 0.03	8.05 ± 1.31	756 ± 78 / 759 ± 1
MobileNetV3-Large	Full	8	4.21 M	0.304	28.16 ± 0.16	6.67 ± 0.18	583 ± 120 / 180 ± 1

Contribution of transfer learning. The two configurations trained without ImageNet pre-trained weights (EfficientNetV2–S† in Tables 8 and 9) performed below their pre-trained counterparts in most cases. With STFT and full fine-tuning, the Macro F1-score decreases from 0.9321 to 0.9022 on CWRU-12kHz and from 0.9910 to 0.9799 on CWRU-48kHz, while on PU the two configurations are comparable (0.7243 with pre-training, 0.6854 without). The largest gap occurs with the CWT representation on MFPT, where the Macro F1-score drops from 0.9223 to 0.4618, and on CWRU-48kHz, where it drops from 0.9035 to 0.8559. These results indicate that the ImageNet weights contribute to the performance of the proposed approach, even though the inputs are time–frequency representations of vibration signals rather than natural images, and that this contribution is most pronounced on the smaller datasets. As reported in Section 5.3, this difference is statistically significant in seven of the eight corresponding pairwise comparisons.

Comparison between architectures. Neither backbone dominates across all datasets. MobileNetV3 obtained the best results on CWRU-12kHz (0.9553 vs. 0.9411 Macro F1 for the best EfficientNetV2–S configuration), whereas EfficientNetV2–S was superior on CWRU-48kHz (0.9947 vs. 0.9782), on PU (0.7243 vs. 0.7166), and on MFPT (0.9234 vs. 0.8891). The difference is most pronounced on MFPT, where MobileNetV3 with the CWT representation degrades markedly (0.6650 and 0.7717 Macro F1 for partial and full fine-tuning, respectively). Considering that MobileNetV3 requires approximately 4.8 times fewer parameters and about 12 times fewer multiply-accumulate operations (Table 1), and that its training time per epoch is substantially lower (Table 7), these results suggest that the lightweight architecture is a viable alternative when computational resources are constrained, at the cost of lower stability on the smallest dataset.

5.3. Statistical test results

The statistical hypothesis tests performed in this study adopt a 95% confidence level; p-values below 0.05 are highlighted in bold, indicating that the null hypothesis can be rejected. Table 10 reports the comparison of each proposed configuration against the baseline for both metrics, and the complete set of pairwise comparisons among all configurations is given in Table 12.

For the CWRU datasets, the statistical analysis confirms the pattern observed in the mean values. On CWRU-48kHz, all ten proposed configurations outperform the baseline with $p < 0.001$ for both metrics. On CWRU-12kHz, all configurations also present higher means, and the difference is statistically significant for nine of them; the only exception is EfficientNetV2–S trained from scratch with STFT and full fine-tuning ($p = 0.134$ for Macro F1 and $p = 0.144$ for Balanced Accuracy), for which the improvement over the baseline cannot be considered significant.

For the PU dataset, the results are mixed. Three configurations outperform the baseline significantly: EfficientNetV2 –S with STFT and full fine-tuning ($p = 0.004$ for Macro F1, $p < 0.001$ for Balanced Accuracy), EfficientNetV2–S with CWT and partial fine-tuning ($p = 0.024$ and $p = 0.009$), and MobileNetV3 with CWT and full fine-tuning ($p = 0.027$ and $p = 0.010$). For the remaining configurations, the observed differences with respect to the baseline are not statistically significant, even when the mean values are higher; this includes both configurations trained from scratch. One configuration, MobileNetV3 with STFT and partial fine-tuning, presents a lower mean than the baseline, but the difference is also not significant ($p = 0.465$ and $p = 0.599$).

For the MFPT dataset, all proposed configurations present lower mean values than the baseline, but the significance of this difference depends on the configuration. For the four EfficientNetV2–S configurations using ImageNet weights, the difference is not statistically significant (p between 0.109 and 0.279 for both metrics), which indicates that these models perform comparably to the baseline despite their lower means. In contrast, the difference is significant for three MobileNetV3 configurations – STFT with partial fine-tuning ($p = 0.029$ and $p = 0.038$), CWT with partial fine-tuning ($p < 0.001$) and CWT with full fine-tuning ($p = 0.002$) – and for both configurations trained from scratch ($p = 0.038$ for STFT and $p < 0.001$ for CWT). On this dataset, therefore, the simpler baseline is not outperformed by any proposed configuration, and it is significantly better than the lightweight backbone and than training without pre-trained weights.

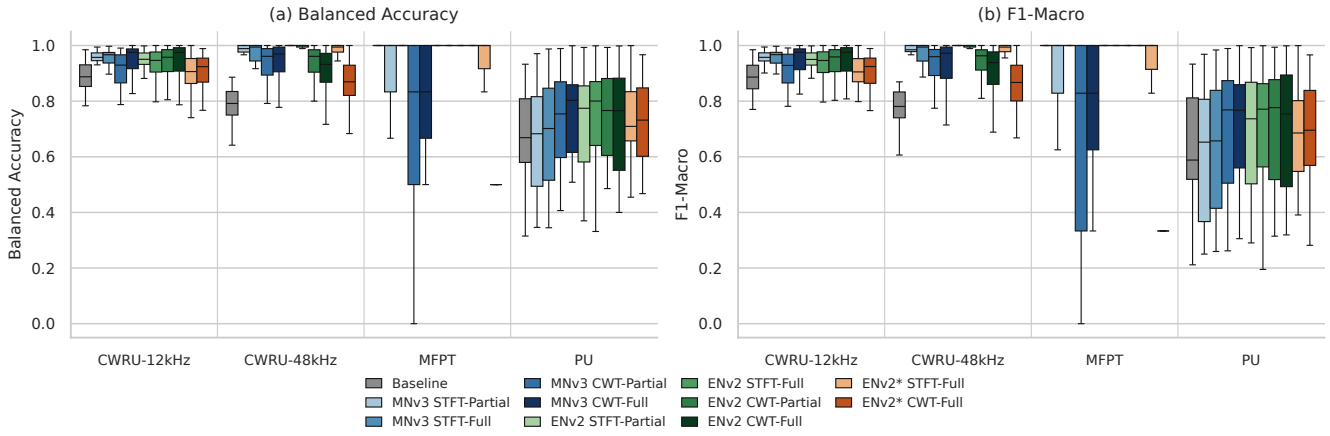


Figure 4. Boxplot comparison of Balanced Accuracy and F1-Macro scores across datasets and feature extraction strategies.

Table 8. Macro F1-score (mean ± std across folds and rounds) of the benchmark (dos Santos Varejão et al., 2025) and the proposed models. Baseline classifiers: DNN (CWRU-12kHz), RF (CWRU-48kHz and PU) and 1-NN (MFPT). †Trained without ImageNet pre-trained weights. Best results per dataset are in bold; second best are underlined.

Backbone	Configuration	CWRU-12kHz	CWRU-48kHz	MFPT	PU
Baseline (dos Santos Varejão et al., 2025)		0.8841 ± 0.0639	0.7764 ± 0.0708	**0.9688 ± 0.0949**	0.6273 ± 0.1993
MobileNetV3	STFT-Partial	<u>0.9521</u> ± 0.0292	0.9782 ± 0.0270	0.8623 ± 0.2421	0.6132 ± 0.2316
	STFT-Full	**0.9553 ± 0.0316**	0.9693 ± 0.0437	0.8891 ± 0.2402	0.6311 ± 0.2368
	CWT-Partial	0.9147 ± 0.0662	0.9315 ± 0.0740	0.6650 ± 0.3222	0.6882 ± 0.2239
	CWT-Full	0.9451 ± 0.0573	0.9133 ± 0.1156	0.7717 ± 0.2928	<u>0.7166</u> ± 0.1885
EfficientNetV2-S	STFT-Partial	0.9411 ± 0.0450	**0.9947 ± 0.0169**	<u>0.9234</u> ± 0.1618	0.6816 ± 0.2178
	STFT-Full	0.9321 ± 0.0611	<u>0.9910</u> ± 0.0197	0.9048 ± 0.2367	**0.7243 ± 0.2001**
	CWT-Partial	0.9355 ± 0.0637	0.9365 ± 0.0631	0.9024 ± 0.2253	0.7154 ± 0.2018
	CWT-Full	0.9401 ± 0.0667	0.9035 ± 0.0979	0.9223 ± 0.1972	0.6950 ± 0.2246
EfficientNetV2-S†	STFT-Full	0.9022 ± 0.0625	0.9799 ± 0.0297	0.8628 ± 0.2615	0.6854 ± 0.1797
	CWT-Full	0.9070 ± 0.0601	0.8559 ± 0.0932	0.4618 ± 0.2620	0.6984 ± 0.1861

Table 9. Balanced accuracy (mean ± std across folds and rounds) of the benchmark (dos Santos Varejão et al., 2025) and the proposed models. Baseline classifiers: DNN (CWRU-12kHz), RF (CWRU-48kHz and PU) and 1-NN (MFPT). †Trained without ImageNet pre-trained weights. Best results per dataset are in bold; second best are underlined.

Backbone	Configuration	CWRU-12kHz	CWRU-48kHz	MFPT	PU
Baseline (dos Santos Varejão et al., 2025)		0.8872 ± 0.0609	0.7846 ± 0.0673	**0.9714 ± 0.0856**	0.6730 ± 0.1574
MobileNetV3	STFT-Partial	<u>0.9521</u> ± 0.0293	0.9790 ± 0.0252	0.8857 ± 0.1929	0.6652 ± 0.1867
	STFT-Full	**0.9552 ± 0.0318**	0.9667 ± 0.0497	0.9143 ± 0.1824	0.6811 ± 0.1897
	CWT-Partial	0.9157 ± 0.0638	0.9342 ± 0.0710	0.7095 ± 0.2868	0.7280 ± 0.1693
	CWT-Full	0.9458 ± 0.0564	0.9195 ± 0.1046	0.8095 ± 0.2430	<u>0.7542</u> ± 0.1484
EfficientNetV2-S	STFT-Partial	0.9411 ± 0.0454	**0.9956 ± 0.0122**	0.9333 ± 0.1353	0.7252 ± 0.1747
	STFT-Full	0.9331 ± 0.0600	<u>0.9911</u> ± 0.0196	0.9286 ± 0.1775	**0.7625 ± 0.1576**
	CWT-Partial	0.9366 ± 0.0611	0.9344 ± 0.0669	0.9238 ± 0.1731	0.7510 ± 0.1592
	CWT-Full	0.9421 ± 0.0638	0.9072 ± 0.0889	<u>0.9381</u> ± 0.1517	0.7336 ± 0.1789
EfficientNetV2-S†	STFT-Full	0.9038 ± 0.0610	0.9795 ± 0.0303	0.8905 ± 0.2059	0.7289 ± 0.1483
	CWT-Full	0.9083 ± 0.0581	0.8637 ± 0.0857	0.5952 ± 0.1951	0.7318 ± 0.1543

Beyond the comparison with the baseline, Table 12 allows the effect of each design choice to be assessed separately. Regarding the contribution of transfer learning, comparing each EfficientNetV2–S configuration with its counterpart

trained from scratch under full fine-tuning yields statistically significant differences in seven of the eight comparisons, always in favour of the pre-trained model: CWRU-12kHz ($p < 0.001$ for both representations), CWRU-48kHz ($p = 0.014$ for STFT and $p < 0.001$ for CWT in Macro F1), PU ($p = 0.018$ and $p = 0.043$), and MFPT with the CWT representation ($p < 0.001$). The only comparison in which the difference is not significant is MFPT with the STFT representation ($p = 0.410$). This provides statistical support for the contribution of the ImageNet weights, which was apparent in the mean values reported in Tables 8 and 9.

Regarding the time–frequency representation, comparing STFT and CWT within the same backbone and fine-tuning strategy reveals a dataset-dependent pattern rather than a uniform advantage of one representation. On CWRU-48kHz, STFT is significantly better than CWT in all four comparisons ($p \leq 0.003$ for both metrics). On PU, the opposite holds for three of the four comparisons, with CWT significantly better for both MobileNetV3 configurations ($p < 0.001$) and for EfficientNetV2–S with partial fine-tuning ($p = 0.011$ for Macro F1 and $p = 0.021$ for Balanced Accuracy); for EfficientNetV2–S with full fine-tuning the difference favours STFT but is significant only for Balanced Accuracy ($p = 0.027$, against $p = 0.065$ for Macro F1). On CWRU-12kHz and MFPT, only the comparison between MobileNetV3 partial configurations is significant ($p < 0.001$ and $p = 0.011$, respectively), with the remaining comparisons showing no significant difference. The choice of representation is therefore relevant, but which representation is preferable depends on the dataset.

Regarding the fine-tuning strategy, full and partial fine-tuning are statistically indistinguishable in most comparisons on CWRU-12kHz, CWRU-48kHz, and MFPT, with only isolated exceptions (MobileNetV3 with CWT on CWRU-12kHz, $p < 0.001$, favouring full fine-tuning; and EfficientNetV2–S with CWT on CWRU-48kHz, $p = 0.006$, favouring partial fine-tuning). On PU, in contrast, full fine-tuning is significantly better than partial fine-tuning in three of the four comparisons ($p < 0.001$ for MobileNetV3 with STFT, $p = 0.025$ for MobileNetV3 with CWT, and $p < 0.001$ for EfficientNetV2–S with STFT, in Macro F1). This effect is therefore restricted to PU and should not be generalized to the other datasets.

Overall, the Wilcoxon signed-rank test results indicate that the proposed methods provide statistically significant improvements over the baseline on both CWRU datasets, and on PU for three specific configurations. On MFPT, no configuration improves upon the baseline, and the EfficientNetV2–S configurations with transfer learning are the only ones whose performance is statistically indistinguishable from it. Among the design choices evaluated, the use of ImageNet pre-trained weights is the one that produces the most consistent effect across datasets, whereas the effects of the time–frequency representation and of the fine-tuning strategy are dataset-dependent.

5.4. Limitations of the Study

Some limitations of this study should be acknowledged, both in the evaluation protocol and in the scope of the experiments.

It should be noted that the observations produced by repeated cross-validation are not statistically independent. Because each fold uses the remaining folds as training data, any two folds within a round share a large portion of their training sets, and the same holds across rounds, since the rounds are permutations of the same acquisition conditions rather than independent resamplings of new data. This overlap violates the independence assumption underlying most hypothesis tests and is known to underestimate the variance of the performance estimates, which in turn inflates the type I error rate (Dietterich, 1998; Nadeau & Bengio, 2003).

Two aspects of the adopted design mitigate, but do not eliminate, this concern. First, the comparison is strictly paired: for every pair of methods, each observation corresponds to the same dataset, the same round, and the same fold, so that the two methods are always compared on identical training and test partitions, and the test is applied to the vector of per-fold, per-round differences. Second, the Wilcoxon signed-rank test is non-parametric and does not assume normality of these differences, which makes it less sensitive to the shape of their distribution than a paired t-test. The dependence between observations, however, remains, and the reported p-values should therefore be interpreted as indicative rather than exact.

The second limitation concerns the internal train/validation split. While the external folds strictly respect the similarity-aware grouping, the 20% of the training data held out for model selection and hyperparameter search is drawn at random within each fold, and is therefore not grouped by acquisition condition. This does not compromise the test partitions, which remain free from similarity bias, but it means that the model selection criterion itself is measured under conditions that are more favourable than those of the final evaluation.

The third concerns the scope of the comparison. The hyperparameter search was performed on a single fold of a single round for each configuration, for computational reasons, and the selected values were then reused across all folds and rounds. Similarly, only two backbones and two time–frequency representations were evaluated, and the conclusions about which representation or fine-tuning strategy is preferable were shown to be dataset-dependent, so they should not be extrapolated beyond the settings considered here.

Table 10. P-values of the Wilcoxon signed-rank test for paired samples (across folds and rounds) comparing each proposed model with the benchmark (dos Santos Varejão et al., 2025), for Macro F1-score (F1) and balanced accuracy (BA). ↑/↓: mean of the proposed model higher/lower than the baseline. Statistically significant results ($p < 0.05$) are in bold. †Trained without ImageNet pre-trained weights. The complete pairwise comparisons are given in Table 12.

		CWRU-12kHz		CWRU-48kHz		MFPT		PU	
Backbone	Configuration	F1	BA	F1	BA	F1	BA	F1	BA
MobileNetV3	STFT-Partial	↑<**0.001**	↑<**0.001**	↑<**0.001**	↑<**0.001**	↓**0.029**	↓**0.038**	↓0.465	↓0.599
	STFT-Full	↑<**0.001**	↑<**0.001**	↑<**0.001**	↑<**0.001**	↓0.087	↓0.087	↑0.789	↑0.890
	CWT-Partial	↑**0.002**	↑**0.003**	↑<**0.001**	↑<**0.001**	↓<**0.001**	↓<**0.001**	↑0.115	↑0.068
	CWT-Full	↑<**0.001**	↑<**0.001**	↑<**0.001**	↑<**0.001**	↓**0.002**	↓**0.002**	↑**0.027**	↑**0.010**
EfficientNetV2-S	STFT-Partial	↑<**0.001**	↑<**0.001**	↑<**0.001**	↑<**0.001**	↓0.200	↓0.200	↑0.190	↑0.091
	STFT-Full	↑<**0.001**	↑<**0.001**	↑<**0.001**	↑<**0.001**	↓0.131	↓0.131	↑**0.004**	↑<**0.001**
	CWT-Partial	↑<**0.001**	↑<**0.001**	↑<**0.001**	↑<**0.001**	↓0.109	↓0.109	↑**0.024**	↑**0.009**
	CWT-Full	↑<**0.001**	↑<**0.001**	↑<**0.001**	↑<**0.001**	↓0.279	↓0.279	↑0.149	↑0.071
EfficientNetV2-S†	STFT-Full	↑0.134	↑0.144	↑<**0.001**	↑<**0.001**	↓**0.038**	↓**0.038**	↑0.224	↑0.068
	CWT-Full	↑**0.028**	↑**0.048**	↑<**0.001**	↑<**0.001**	↓<**0.001**	↓<**0.001**	↑0.091	↑0.065

Finally, all evaluations are performed within each dataset. No cross-dataset validation was carried out, which would characterize a domain generalization setting rather than the leakage-free evaluation of supervised classifiers addressed in this work. As discussed in Section 3.1, such a comparison is a natural extension of this study and is left as future work.

6. CONCLUSIONS AND FUTURE WORK

The experimental results show that time–frequency representations combined with pre-trained convolutional networks outperform the benchmark baseline (dos Santos Varejão et al., 2025) on the CWRU-12kHz and CWRU-48kHz datasets. On CWRU-48kHz, all ten evaluated configurations achieved statistically significant improvements in both Macro F1-score and Balanced Accuracy; on CWRU-12kHz, nine of the ten did, the only exception being EfficientNetV2–S trained from scratch with STFT and full fine-tuning ($p = 0.134$ for Macro F1 and $p = 0.144$ for Balanced Accuracy). On CWRU-12kHz, the best result was obtained by MobileNetV3 with STFT and full fine-tuning (0.9553 Macro F1, 0.9552 Balanced Accuracy, against 0.8841 and 0.8872 for the baseline), and on CWRU-48kHz by EfficientNetV2–S with STFT and partial fine-tuning (0.9947 and 0.9956, against 0.7764 and 0.7846). On both datasets, the improvement was accompanied by a reduction in the dispersion across folds and rounds, indicating more consistent behaviour under the similarity-aware evaluation protocol.

On the PU dataset, the improvement over the baseline was more limited: the best configuration, EfficientNetV2–S with STFT and full fine-tuning, reached 0.7243 Macro F1 and 0.7625 Balanced Accuracy against 0.6273 and 0.6730 for the baseline, but only three of the ten evaluated configurations differed significantly from it. Unlike the CWRU datasets, the dispersion was not reduced on PU, remaining comparable to that of the baseline. Absolute performance on this dataset is considerably lower than on CWRU for all methods, including the baseline, which suggests that the difficulty lies in the dataset itself rather than in the choice of representation or architecture.

A different behaviour was observed on the MFPT dataset, where the baseline achieved the highest mean scores (0.9688 Macro F1 and 0.9714 Balanced Accuracy) and no proposed configuration surpassed it. It is worth noting, however, that for the four EfficientNetV2–S configurations using ImageNet weights the difference with respect to the baseline is not statistically significant (p between 0.109 and 0.279), so these models should be regarded as performing comparably to the baseline rather than worse than it. The proposed models did, on the other hand, exhibit substantially higher variability across folds on this dataset than on the others. MFPT is the smallest of the four datasets and contains only two classes, which favours the simpler distance-based baseline – in this case a Nearest Neighbor (1-NN) classifier – and may also explain the instability of the deep models, whose performance depends more strongly on which acquisition conditions fall in the training partition of each fold.

Among the design choices evaluated, the use of ImageNet pre-trained weights produced the most consistent effect across datasets: comparing each EfficientNetV2–S configuration with its counterpart trained from scratch yielded statistically significant differences in seven of the eight comparisons, always favouring the pre-trained model. This indicates that features learned from natural images do transfer usefully to time–frequency representations of vibration signals, even under an evaluation protocol free from similarity bias. In contrast, the effects of the time –frequency representation and of the fine-tuning strategy proved to be dataset-dependent: STFT was significantly

better than CWT on CWRU-48kHz, while the opposite held for most configurations on PU, and full fine-tuning was significantly better than partial fine-tuning only on PU. These results suggest that conclusions about representation and fine-tuning strategy reported in the literature, often obtained under random data splits, should not be assumed to generalize across datasets.

The comparison between backbones also did not favour a single architecture. MobileNetV3 obtained the best results on CWRU-12kHz, while EfficientNetV2–S was superior on the remaining three datasets, with the largest gap on MFPT. Considering that MobileNetV3 requires approximately 4.8 times fewer parameters and about 12 times fewer multiply-accumulate operations, and that its training time per epoch is substantially lower, the lightweight architecture represents a viable alternative for deployment scenarios with constrained computational resources, at the cost of lower stability on the smallest dataset. This trade-off between diagnostic performance and computational cost is particularly relevant for PHM applications, where models are often deployed on edge devices with limited memory and inference-time budgets.

As future work, a comparative investigation between time –frequency and purely frequency-domain representations is envisaged. In particular, spectral images derived exclusively from frequency-domain transforms, such as the Fourier Transform, will be explored and contrasted with the STFT-based representations adopted in this study. This comparison aims to evaluate trade-offs between diagnostic accuracy, computational complexity, and processing time, especially in resource-constrained industrial environments.

Another promising research direction involves the fusion of multiple signal representations into unified multi-channel inputs (Arias-Vergara et al., 2021; da Silva, Rodrigues, Varejão, Ribeiro, & Oliveira-Santos, 2025). Different transforms, such as STFT, wavelet-based scalograms, and pure frequency spectra, may be mapped into separate RGB channels, allowing convolutional neural networks to jointly exploit complementary temporal and spectral features. Such representation-level fusion has the potential to enhance feature richness, robustness, and discriminative capability.

Another direction is to compare the transfer learning strategies investigated here with methods specifically designed for domain generalization in bearing fault diagnosis (e.g., (Jia et al., 2023)), in which training and test data are drawn from distinct domains (e.g., different operating conditions or equipment). Since the use of distinct domains eliminates similarity bias, such a comparison can be conducted under the same unbiased evaluation protocol adopted in this study.

Finally, future investigations will also focus on advanced data augmentation strategies to mitigate the limitations imposed by scarce labeled datasets. Time-domain perturbations, frequency masking, signal scaling, noise injection, and image-based augmentations applied to time–frequency representations may increase data diversity and improve generalization. These strategies are expected to enhance robustness and scalability, contributing to more reliable intelligent fault diagnosis systems for real-world industrial applications.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the financial support provided by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Brazil, and the Fundação de Amparo à Pesquisa e Inovação do Espírito Santo (FAPES), under Grant TO 1138/2025 (Project No. 2025-Z9VRS).

DECLARATION OF GENERATIVE AI AND AI-ASSISTED TECHNOLOGIES IN THE WRITING PROCESS

During the preparation of this manuscript, the authors used ChatGPT (OpenAI, 2023) and Claude (Anthropic, 2026) to assist with language refinement and readability improvements. The generated outputs were critically reviewed, revised, and validated by the authors. The authors assume full responsibility for the content and conclusions presented in this publication.

DECLARATION OF COMPETING INTEREST

The authors have no competing interests to declare that are relevant to the content of this article.

A. SELECTED HYPERPARAMETERS

Table 11. Hyperparameters selected by Optuna for each dataset, backbone, time-frequency representation, and training strategy (Table 6). MNv3: MobileNetV3; ENv2: EfficientNetV2-S. $\mathrm{LR_{cls}}$ and $\mathrm{LR_{bb}}$: learning rates of the classification layer and of the backbone (partial and full fine-tuning); for training from scratch, a single learning rate is used for the whole network. WD: weight decay; BS: batch size; LS: label smoothing; n: number of unfrozen blocks (partial fine-tuning only); Train.: percentage of trainable parameters. "–": not applicable.

Backbone	Repr.	Strategy	Optimizer	$\mathrm{LR_{cls}}$	$\mathrm{LR_{bb}}$	WD	BS	LS	n	Train. (%)
CWRU-12kHz										
MNv3	STFT	Partial	AdamW	1.21×10^{-4}	6.22×10^{-5}	1.24×10^{-6}	16	3.79×10^{-3}	2	52.0
MNv3	STFT	Full	AdamW	1.96×10^{-3}	1.90×10^{-5}	9.54×10^{-5}	16	2.92×10^{-3}	–	100
MNv3	CWT	Partial	AdamW	4.65×10^{-3}	2.45×10^{-4}	3.23×10^{-3}	8	2.49×10^{-4}	3	71.0
MNv3	CWT	Full	AdamW	4.84×10^{-4}	3.13×10^{-4}	2.88×10^{-3}	16	4.04×10^{-3}	–	100
ENv2	STFT	Partial	AdamW	4.11×10^{-4}	2.64×10^{-5}	1.09×10^{-5}	16	2.45×10^{-2}	3	91.0
ENv2	STFT	Full	AdamW	3.19×10^{-5}	1.85×10^{-4}	4.48×10^{-4}	8	6.41×10^{-4}	–	100
ENv2	CWT	Partial	AdamW	1.05×10^{-3}	4.80×10^{-4}	1.44×10^{-5}	32	5.31×10^{-3}	3	91.0
ENv2	CWT	Full	AdamW	2.71×10^{-4}	8.07×10^{-5}	2.82×10^{-5}	16	1.97×10^{-3}	–	100
ENv2	STFT	Scratch	SGD	1.83×10^{-2}		4.03×10^{-4}	16	5.15×10^{-3}	–	100
ENv2	CWT	Scratch	AdamW	3.23×10^{-4}		1.76×10^{-3}	32	3.54×10^{-3}	–	100
CWRU-48kHz										
MNv3	STFT	Partial	AdamW	2.36×10^{-3}	9.08×10^{-6}	2.74×10^{-3}	8	8.50×10^{-3}	2	52.0
MNv3	STFT	Full	AdamW	7.44×10^{-4}	5.22×10^{-4}	6.08×10^{-5}	16	1.70×10^{-2}	–	100
MNv3	CWT	Partial	AdamW	6.90×10^{-4}	2.16×10^{-5}	1.14×10^{-5}	8	6.81×10^{-2}	1	33.0
MNv3	CWT	Full	AdamW	1.96×10^{-5}	7.25×10^{-4}	1.41×10^{-4}	8	1.37×10^{-2}	–	100
ENv2	STFT	Partial	AdamW	2.81×10^{-3}	3.18×10^{-4}	1.16×10^{-3}	8	1.74×10^{-3}	3	91.0
ENv2	STFT	Full	AdamW	7.61×10^{-4}	1.57×10^{-4}	1.11×10^{-3}	64	1.40×10^{-3}	–	100
ENv2	CWT	Partial	AdamW	7.49×10^{-5}	9.09×10^{-5}	5.42×10^{-3}	64	1.36×10^{-2}	2	73.8
ENv2	CWT	Full	AdamW	6.87×10^{-3}	4.25×10^{-5}	6.10×10^{-6}	16	2.19×10^{-3}	–	100
ENv2	STFT	Scratch	AdamW	1.45×10^{-4}		1.65×10^{-3}	16	8.25×10^{-3}	–	100
ENv2	CWT	Scratch	SGD	1.21×10^{-2}		2.49×10^{-5}	16	9.19×10^{-3}	–	100
MFPT										
MNv3	STFT	Partial	AdamW	6.62×10^{-3}	2.08×10^{-5}	5.44×10^{-4}	8	5.73×10^{-2}	2	52.0
MNv3	STFT	Full	AdamW	5.04×10^{-3}	7.08×10^{-4}	1.87×10^{-4}	8	1.37×10^{-1}	–	100
MNv3	CWT	Partial	AdamW	8.63×10^{-3}	9.88×10^{-6}	3.00×10^{-4}	8	6.47×10^{-2}	3	70.9
MNv3	CWT	Full	AdamW	6.62×10^{-3}	2.20×10^{-5}	1.29×10^{-6}	8	6.22×10^{-2}	–	100
ENv2	STFT	Partial	AdamW	7.66×10^{-4}	3.55×10^{-5}	2.77×10^{-5}	4	8.34×10^{-4}	3	91.0
ENv2	STFT	Full	AdamW	1.68×10^{-3}	1.75×10^{-5}	1.19×10^{-5}	4	4.62×10^{-3}	–	100
ENv2	CWT	Partial	AdamW	8.32×10^{-4}	2.19×10^{-5}	2.94×10^{-5}	4	9.33×10^{-3}	3	91.0
ENv2	CWT	Full	AdamW	1.95×10^{-4}	1.63×10^{-4}	1.00×10^{-3}	4	4.49×10^{-4}	–	100
ENv2	STFT	Scratch	AdamW	2.02×10^{-2}		7.13×10^{-6}	16	7.93×10^{-2}	–	100
ENv2	CWT	Scratch	AdamW	1.61×10^{-2}		2.54×10^{-4}	64	1.54×10^{-2}	–	100
PU										
MNv3	STFT	Partial	AdamW	1.91×10^{-4}	2.16×10^{-6}	2.28×10^{-6}	16	5.65×10^{-3}	3	71.0
MNv3	STFT	Full	AdamW	3.80×10^{-3}	1.88×10^{-5}	1.04×10^{-3}	32	3.65×10^{-4}	–	100
MNv3	CWT	Partial	AdamW	2.58×10^{-5}	6.11×10^{-4}	2.93×10^{-4}	64	1.36×10^{-2}	3	71.0
MNv3	CWT	Full	AdamW	2.62×10^{-3}	8.72×10^{-4}	3.60×10^{-4}	8	1.51×10^{-3}	–	100
ENv2	STFT	Partial	AdamW	6.09×10^{-4}	9.75×10^{-4}	4.44×10^{-5}	32	2.22×10^{-3}	2	73.8
ENv2	STFT	Full	AdamW	2.38×10^{-5}	6.38×10^{-5}	2.43×10^{-4}	32	8.86×10^{-5}	–	100
ENv2	CWT	Partial	AdamW	6.15×10^{-5}	2.46×10^{-4}	5.47×10^{-5}	16	6.78×10^{-4}	2	73.8
ENv2	CWT	Full	AdamW	4.10×10^{-3}	1.90×10^{-4}	2.48×10^{-3}	64	2.48×10^{-3}	–	100
ENv2	STFT	Scratch	AdamW	7.51×10^{-4}		7.84×10^{-6}	64	1.50×10^{-4}	–	100
ENv2	CWT	Scratch	AdamW	1.74×10^{-2}		6.09×10^{-5}	16	1.38×10^{-3}	–	100

B. PAIRWISE STATISTICAL COMPARISONS

Table 12. Complete pairwise p-values of the Wilcoxon signed-rank test for paired samples (across folds and rounds). The upper triangular part reports results based on the Macro F1-score and the lower triangular part on balanced accuracy. Statistically significant results ($p < 0.05$) are in bold. BL: baseline (dos Santos Varejão et al., 2025); M: MobileNetV3; E: EfficientNetV2-S; E†: EfficientNetV2-S trained from scratch (without ImageNet weights); S/C: STFT/CWT; P/F: partial/full fine-tuning. "–": not applicable.

	BL	M-SP	M-SF	M-CP	M-CF	E-SP	E-SF	E-CP	E-CF	E†-SF	E†-CF
CWRU-12kHz											
BL	–	**<0.001**	**<0.001**	**0.002**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	0.134	**0.028**
M-SP	**<0.001**	–	0.062	**<0.001**	0.775	**0.010**	0.059	0.068	0.524	**<0.001**	**<0.001**
M-SF	**<0.001**	0.067	–	**<0.001**	0.400	**<0.001**	**0.016**	**0.011**	0.379	**<0.001**	**<0.001**
M-CP	**0.003**	**<0.001**	**<0.001**	–	**<0.001**	**<0.001**	**0.007**	**<0.001**	**<0.001**	0.184	**0.043**
M-CF	**<0.001**	0.733	0.421	**<0.001**	–	0.217	**0.048**	**0.016**	0.088	**<0.001**	**<0.001**
E-SP	**<0.001**	**0.010**	**<0.001**	**<0.001**	0.166	–	0.548	0.477	0.710	**<0.001**	**<0.001**
E-SF	**<0.001**	0.071	**0.016**	**0.008**	**0.046**	0.640	–	0.784	0.203	**<0.001**	**<0.001**
E-CP	**<0.001**	0.057	**0.011**	**<0.001**	**0.019**	0.550	0.813	–	0.142	**<0.001**	**<0.001**
E-CF	**<0.001**	0.601	0.443	**<0.001**	0.117	0.544	0.183	0.104	–	**<0.001**	**<0.001**
E†-SF	0.144	**<0.001**	**<0.001**	0.166	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	–	0.747
E†-CF	**0.048**	**<0.001**	**<0.001**	**0.034**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	0.830	–
CWRU-48kHz											
BL	–	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**
M-SP	**<0.001**	–	0.626	**<0.001**	**0.013**	**<0.001**	**0.012**	**0.001**	**<0.001**	0.183	**<0.001**
M-SF	**<0.001**	0.339	–	**<0.001**	**0.002**	**0.003**	0.054	**<0.001**	**<0.001**	0.550	**<0.001**
M-CP	**<0.001**	**0.004**	**<0.001**	–	0.544	**<0.001**	**<0.001**	0.789	**0.009**	**0.001**	**<0.001**
M-CF	**<0.001**	**0.013**	**0.003**	0.311	–	**<0.001**	**<0.001**	0.374	**0.044**	**0.009**	**0.003**
E-SP	**<0.001**	**<0.001**	**0.002**	**<0.001**	**<0.001**	–	0.425	**<0.001**	**<0.001**	**0.002**	**<0.001**
E-SF	**<0.001**	**0.011**	**0.047**	**<0.001**	**<0.001**	0.296	–	**<0.001**	**<0.001**	**0.014**	**<0.001**
E-CP	**<0.001**	**0.002**	**<0.001**	0.873	0.471	**<0.001**	**<0.001**	–	**0.006**	**0.003**	**<0.001**
E-CF	**<0.001**	**<0.001**	**<0.001**	**0.006**	**0.025**	**<0.001**	**<0.001**	**0.010**	–	**<0.001**	**0.007**
E†-SF	**<0.001**	0.146	0.434	**0.001**	**0.008**	**0.001**	**0.017**	**0.002**	**<0.001**	–	**<0.001**
E†-CF	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**0.003**	**<0.001**	**<0.001**	**<0.001**	**0.007**	**<0.001**	–
MFPT											
BL	–	**0.029**	0.087	**<0.001**	**0.002**	0.200	0.131	0.109	0.279	**0.038**	**<0.001**
M-SP	**0.038**	–	0.504	**0.011**	0.097	0.256	0.286	0.345	0.289	0.958	**<0.001**
M-SF	0.087	0.294	–	**0.006**	0.075	0.285	0.755	0.834	0.389	0.608	**<0.001**
M-CP	**<0.001**	**0.005**	**0.002**	–	0.123	**<0.001**	**<0.001**	**0.001**	**0.003**	**0.004**	**0.006**
M-CF	**0.002**	0.083	0.081	0.064	–	**0.007**	**0.005**	**0.011**	**0.022**	0.128	**<0.001**
E-SP	0.200	0.130	0.811	**<0.001**	**0.014**	–	0.391	0.368	0.892	0.256	**<0.001**
E-SF	0.131	0.220	0.662	**<0.001**	**0.002**	0.493	–	0.655	0.730	0.410	**<0.001**
E-CP	0.109	0.261	0.754	**<0.001**	**0.006**	0.857	0.655	–	0.777	0.520	**<0.001**
E-CF	0.279	0.178	0.490	**<0.001**	**0.024**	0.684	1.000	1.000	–	0.251	**<0.001**
E†-SF	**0.038**	0.916	0.564	**<0.001**	0.068	0.297	0.410	0.547	0.226	–	**<0.001**
E†-CF	**<0.001**	**<0.001**	**<0.001**	0.103	**0.003**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	–
PU											
BL	–	0.465	0.789	0.115	**0.027**	0.190	**0.004**	**0.024**	0.149	0.224	0.091
M-SP	0.599	–	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**0.002**	**<0.001**
M-SF	0.890	**0.001**	–	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**<0.001**	**0.022**	**0.002**
M-CP	0.068	**<0.001**	**<0.001**	–	**0.025**	0.665	**0.033**	**0.036**	0.638	0.905	0.905
M-CF	**0.010**	**<0.001**	**<0.001**	**0.006**	–	0.071	0.477	0.761	0.350	0.111	0.295
E-SP	0.091	**<0.001**	**<0.001**	0.875	**0.036**	–	**<0.001**	**0.011**	0.231	0.919	0.993
E-SF	**<0.001**	**<0.001**	**<0.001**	**0.005**	0.322	**<0.001**	–	0.612	0.065	**0.018**	**0.043**
E-CP	**0.009**	**<0.001**	**<0.001**	**0.023**	0.638	**0.021**	0.331	–	0.139	0.197	0.262
E-CF	0.071	**<0.001**	**<0.001**	0.548	0.231	0.313	**0.027**	0.068	–	0.818	0.665
E†-SF	0.068	**<0.001**	**0.009**	0.934	0.106	0.934	**0.033**	0.287	0.963	–	0.246
E†-CF	0.065	**<0.001**	**<0.001**	0.934	0.178	0.993	**0.018**	0.270	0.665	0.761	–

REFERENCES

Ahmed, H. O. A., & Nandi, A. K. (2022). Vibration image representations for fault diagnosis of rotating machines: A review. *Machines*, *10*(12), 1113. doi: 10.3390/machines10121113

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2623–2631). New York, NY, USA: Association for Computing Machinery. doi: 10.1145/3292500.3330701

Anthropic. (2026). *Claude Sonnet 4.6*. `https://www.anthropic.com/news/claude-sonnet-4-6`. (Large language model developed by Anthropic. Accessed: 29 September 2026)

Arias-Vergara, T., Klumpp, P., Vasquez-Correa, J. C., Nöth, E., Orozco-Arroyave, J. R., & Schuster, M. (2021). Multi-channel spectrograms for speech processing applications using deep learning methods. *Pattern Analysis and Applications*, *24*(2), 423–431. doi: 10.1007/s10044-020-00921-5

Arun Balaji, M., Naveen Venkatesh, S., Sugumaran, V., & Ramachandran, K. (2025). A comparative study of image representation for roller bearing fault diagnosis using pretrained networks. *Journal of Engineering*, *2025*(1), 4707723. doi: 10.1155/je/4707723

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. W. (2010). A theory of learning from different domains. *Machine Learning*, *79*(1), 151–175. doi: 10.1007/s10994-009-5152-4

Case Western Reserve University. (2014). *Bearing Data Center*. `http://csegroups.case.edu/bearingdatacenter`. (Accessed: 17 December 2025)

Chen, H., Luo, H., Huang, B., Jiang, B., & Kaynak, O. (2024). Transfer learning-motivated intelligent fault diagnosis designs: A survey, insights, and perspectives. *IEEE Transactions on Neural Networks and Learning Systems*, *35*(3), 2969–2983. doi: 10.1109/TNNLS.2023.3290974

Chen, Z., Cen, J., & Xiong, J. (2020). Rolling bearing fault diagnosis using time-frequency analysis and deep transfer convolutional neural network. *IEEE Access*, *8*, 150248–150261. doi: 10.1109/ACCESS.2020.3016888

da Silva, L. H. P., Rodrigues, A., Varejão, F., Ribeiro, M. P., & Oliveira-Santos, T. (2025). Electrical submersible pump fault diagnosis based on 2D transformation of vibration signals and transfer learning of image classification networks. *Neural Computing and Applications*, *37*(14), 8509–8525. doi: 10.1007/s00521-025-10998-w

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248–255). doi: 10.1109/CVPR.2009.5206848

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, *10*(7), 1895–1923. doi: 10.1162/089976698300017197

dos Santos Varejão, I. M., de Oliveira Costa, L. G., da Silva, L. H. P., Rodrigues, A., Ribeiro, M. P., Varejão, F. M., & Oliveira-Santos, T. (2025). The similarity bias problem: What it is and how it impacts vibration based intelligent fault diagnosis. *Mechanical Systems and Signal Processing*, *235*, 112822. doi: 10.1016/j.ymssp.2025.112822

Fan, Y., Fu, Z., Li, H., & Yang, Y. (2025). Motor bearing fault diagnosis based on LMSWT with improved multiscale convolutional neural network. *IEEE Transactions on Instrumentation and Measurement*. doi: 10.1109/TIM.2025.3565707

Gulrajani, I., & Lopez-Paz, D. (2020). In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*. doi: 10.48550/arXiv.2007.01434

Gültekin, Ö., Çinar, E., Özkan, K., & Yazıcı, A. (2022). A novel deep learning approach for intelligent fault diagnosis applications based on time-frequency images. *Neural Computing and Applications*, *34*(6), 4803–4812. doi: 10.1007/s00521-021-06668-2

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., … Adam, H. (2019). Searching for MobileNetV3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 1314–1324). doi: 10.1109/ICCV.2019.00140

Jang, J.-g., Lee, S.-s., Hwang, S.-y., & Lee, J.-c. (2024). Noise reduction in CWRU data using DAE and classification with ViT. *Applied Sciences*, *14*(24), 11771. doi: 10.3390/app142411771

Jia, S., Li, Y., Wang, X., Sun, D., & Deng, Z. (2023). Deep causal factorization network: A novel domain generalization method for cross-machine bearing fault diagnosis. *Mechanical Systems and Signal Processing*, *192*, 110228. doi: 10.1016/j.ymssp.2023.110228

Konstruktions- und Antriebstechnik (KAt), Universität Paderborn. (2026). *Bearing DataCenter*. `https://mb.uni-paderborn.de/kat/forschung/bearing-datacenter`. (Accessed: 23 May 2026)

Lessmeier, C., Kimotho, J. K., Zimmer, D., & Sextro, W. (2016). Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification. *PHM Society European Conference*, *3*(1). doi: 10.36001/phme.2016.v3i1.1577

Liu, J., Shen, Z., He, Y., Zhang, X., Xu, R., Yu, H., & Cui, P. (2021). Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*. doi: 10.48550/arXiv.2108.13624

Łuczak, D. (2024). Machine fault diagnosis through vibration analysis: Time series conversion to grayscale and RGB images for recognition via convolutional neural networks. *Energies*, *17*(9), 1998. doi: 10.3390/en17091998

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys*, *54*(6), 1–35. doi: 10.1145/3457607

Nadeau, C., & Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, *52*(3), 239–281. doi: 10.1023/A:1024068626366

Nie, G., Zhang, Z., Jiao, Z., Li, Y., Shao, M., & Dai, X. (2025). A novel intelligent bearing fault diagnosis method based on image enhancement and improved convolutional neural network. *Measurement*, *242*, 116148. doi: 10.1016/j.measurement.2024.116148

OpenAI. (2023). *ChatGPT*. `https://chat.openai.com/`. (Accessed: 23 September 2026)

Rauber, T. W., da Silva Loca, A. L., de Assis Boldt, F., Rodrigues, A. L., & Varejão, F. M. (2021). An experimental methodology to evaluate machine learning methods for fault diagnosis based on vibration signals. *Expert Systems with Applications*, *167*, 114022. doi: 10.1016/j.eswa.2020.114022

Ren, Z., & Guo, J. (2024). On fault diagnosis using image-based deep learning networks based on vibration signals. *Multimedia Tools and Applications*, *83*(15), 44555–44580. doi: 10.1007/s11042-023-17384-5

Sanz, J., Perera, R., & Huerta, C. (2007). Fault diagnosis of rotating machinery based on auto-associative neural networks and wavelet transforms. *Journal of Sound and Vibration*, *302*(4-5), 981–999. doi: 10.1016/j.jsv.2007.01.006

Shafi, I., Ahmad, J., Shah, S. I., & Kashif, F. M. (2009). Techniques to obtain good resolution and concentrated time-frequency distributions: a review. *EURASIP Journal on Advances in Signal Processing*, *2009*(1), 673539. doi: 10.1155/2009/673539

Tan, M., & Le, Q. (2021). EfficientNetV2: Smaller models and faster training. In *International Conference on Machine Learning* (pp. 10096–10106).

Thuan, N. D., & Hong, H. S. (2023). HUST bearing: a practical dataset for ball bearing fault diagnosis. *BMC Research Notes*, *16*(1), 138. doi: 10.1186/s13104-023-06400-4

Venturim, L. A. S., & de Assis Boldt, F. (2021). A proposal to mitigate similarity bias for the Paderborn bearing data set. In *IECON 2021 – 47th Annual Conference of the IEEE Industrial Electronics Society* (pp. 1–6). doi: 10.1109/IECON48115.2021.9589342

Wang, C., Sun, Y., & Wang, X. (2024). Image deep learning in fault diagnosis of mechanical equipment. *Journal of Intelligent Manufacturing*, *35*(6), 2475–2515. doi: 10.1007/s10845-023-02176-3

Wang, J., Lan, C., Liu, C., Ouyang, Y., Qin, T., Lu, W., ... Yu, P. S. (2022). Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, *35*(8), 8052–8072. doi: 10.1109/TKDE.2022.3178128

Wang, L., & Zhang, X. (2023). Anomaly detection for automated vehicles integrating continuous wavelet transform and convolutional neural network. *Applied Sciences*, *13*(9), 5525. doi: 10.3390/app13095525

Wang, T., Lu, C., Sun, Y., Yang, M., Liu, C., & Ou, C. (2021). Automatic ECG classification using continuous wavelet transform and convolutional neural network. *Entropy*, *23*(1), 119. doi: 10.3390/e23010119

Wang, Y., He, Z., & Zi, Y. (2010). Enhancement of signal denoising and multiple fault signatures detecting in rotating machinery using dual-tree complex wavelet transform. *Mechanical Systems and Signal Processing*, *24*(1), 119–137. doi: 10.1016/j.ymssp.2009.06.015

Wilcoxon, F. (1992). Individual comparisons by ranking methods. In *Breakthroughs in statistics: Methodology and distribution* (pp. 196–202). Springer.

Wu, G., Ji, X., Yang, G., Jia, Y., & Cao, C. (2023). Signal-to-image: Rolling bearing fault diagnosis using ResNet family deep-learning models. *Processes*, *11*(5), 1527. doi: 10.3390/pr11051527

Xu, M., Qiu, J., Zhu, B., & Chen, Z. (2019). Time–frequency map-based abnormal signal detection. *IEEE Access*, *7*, 172350–172361. doi: 10.1109/ACCESS.2019.2956264

Yang, Y., Dong, X., Peng, Z., Zhang, W., & Meng, G. (2015). Vibration signal analysis using parameterized time–frequency method for features extraction of varying-speed rotary machinery. *Journal of Sound and Vibration*, *335*, 350–366. doi: 10.1016/j.jsv.2014.09.025

Zhang, Q., & Deng, L. (2023). An intelligent fault diagnosis method of rolling bearings based on short-time Fourier transform and convolutional neural network. *Journal of Failure Analysis and Prevention*, *23*(2), 795–811. doi: 10.1007/s11668-023-01616-9

Zhang, X., Liu, S., Li, L., Lei, J., & Chang, G. (2021). Multiscale holospectrum convolutional neural network-based fault diagnosis of rolling bearings with variable operating conditions. *Measurement Science and Technology*, *32*(10), 105027. doi: 10.1088/1361-6501/ac05f8

Zhang, Y., Xing, K., Bai, R., Sun, D., & Meng, Z. (2020). An enhanced convolutional neural network for bearing fault diagnosis based on time–frequency image. *Measurement*, *157*, 107667. doi: 10.1016/j.measurement

.2020.107667