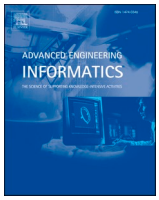
Advanced Engineering Informatics 62 (2024) 102806

Contents lists available at ScienceDirect

Advanced Engineering Informatics

journal homepage: www.elsevier.com/locate/aei



Full length article

Physics-informed probabilistic deep network with interpretable mechanism for trustworthy mechanical fault diagnosis

Zifei Xu [a], Kaicheng Zhao [b], Jin Wang [c], Musa Bashir [a,*]

[a] Department of Civil and Environmental Engineering, University of Liverpool, The Quadrangle, Brownlow Hill, L69 3GH Liverpool, UK
[b] School of Energy and Power Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China
[c] Department of Marine and Mechanical Engineering, Liverpool John Moores University, James Parsons Building, Byrom Street, Liverpool L3 3AF, UK

ARTICLE INFO

Keywords:
Physics-informed neural network
Smart data
Trustworthy and interpretable Fault diagnostics
Deep probabilistic neural network

ABSTRACT

The application of data-driven models based on the neural network is pivotal to developing an intelligent fault diagnostic flowchart. However, the reliability and interpretability of these models for fault prediction have presented challenges in neural network-based diagnostic approach. Another challenge is data availability, which has also been a limiting factor in using artificial Intelligence for condition monitoring and fault assessment in industrial mechanical systems. Consequently, this study proposes a Physics Informed Probabilistic Deep Network (PIPDN) framework to overcome these challenges. The PIPDN comprises two main components: the physical labelling module, designed to enhance physical labels with mechanical failure information, and the main body of PIPDN, responsible for learning fault representative features and generating smart data guided by conditional and physical labels. Furthermore, a multi-scale PIPDN model is developed to integrate the proposed uncertainty quantification (UQ) with decision-fusion module for accurate interpretation and enhanced fault diagnosis. The applicability, effectiveness, and superiority of the proposed framework and approach are validated using an experimental bearing dataset. The results indicate that integrating physical labels significantly assists the PIPDN model in capturing more accurate fault characteristics. This increases the importance of latent space features for subsequent fault diagnosis and also enhances the diagnostic interpretability. Furthermore, the addition of UQ-based decision-making module improves the reliability of the MS-PIPDN model by reducing epistemic uncertainty in the predictions.

1. Introduction

1.1. Background

The modernization of industries has led to an increased demand for rotating machinery, including engines, turbines, and centrifuges [25]. The high-intensity and continuous nature of these machines' operations can result in fatigue, structural failures and safety concerns, leading to expensive maintenance. Developing accurate and reliable monitoring and assessment framework to forecast and manage the health conditions of critical mechanical components underscores the need to improve machinery reliability, operational safety and reduce costs [34].

The application of artificial intelligence (AI) in condition monitoring of industrial machinery, with the primary goal of achieving accurate and reliable fault diagnosis for optimal maintenance, has been receiving significant attention. The focus on AI application in fault diagnosis approaches is the development of a Prognostics and Health Management (PHM) flowchart, which is a crucial foundation in modern machine maintenance. This is because the effectiveness and reliability of any fault diagnosis method employed in machinery maintenance have direct consequences on the stability and safe operations of the PHM system [33]. In addition, the robustness of fault diagnostic models is essential in ensuring the reliability of the PHM system during prolonged operation [19]. Therefore, ensuring both the practicality and robustness of fault diagnosis methods is a paramount requirement for the effective functioning of PHM, which potentially ensures operator safety while potentially reducing Operation and Maintenance (O&M) costs [18].

The application of Deep Learning (DL), an innovative subset of AI, has been receiving attention because of its capability to enhance the fault diagnosis methodology of mechanical systems. DL can significantly reduce the hazards and risks associated with reliance on manual decision-making in machine maintenance [2]. In addition, other AI components, such as Convolutional neural networks (CNNs), Recurrent

* Corresponding author.
E-mail address: m.b.bashir@liverpool.ac.uk (M. Bashir).

https://doi.org/10.1016/j.aei.2024.102806
Received 10 December 2023; Received in revised form 24 August 2024; Accepted 1 September 2024
Available online 12 September 2024

Nomenclature

Acronym	Full Name
PIPDN	Physics-Informed Probabilistic Deep Network
DCNN	Deep Convolutional Neural Network
MS-PIPDN	Multi-Scale Physics-Informed Probabilistic Deep Network
MSCNN	Multiscale Convolutional Neural Network
UQ	Uncertainty Quantification
AUQ	Aleatoric Uncertainty Quantification
EUQ	Epistemic Uncertainty Quantification
PINN	Physics-Informed Neural Network
PINet	Physics-Informed Residual Network
FFT	Fast Fourier Transform
BPFI	Ball pass frequency, inner race
BPFO	Ball pass frequency, outer race
BSF	Ball spin frequency

neural networks (RNNs) and Self Attention network (Transformer) have also been successfully applied to model and diagnose faults [15,10] (Y. [5]. These tools benefit from AI's adaptable applications in transfer learning and generative learning. Their effectiveness in addressing various challenges, including adapting to evolving operational conditions in fault diagnosis, handling small-sample fault diagnosis, and resolving issues arising from imbalanced diagnostic datasets have been established [32].

However, data availability, as well as the reliability and complexity of AI models, are constraints that limit the development and application of large-scale AI models in the field of fault diagnosis [9]. Conversely, with the growing maturity and application of Internet of Things (IoT) technology in industrial applications (D. [23,16], the issue of data scarcity for DL model learning has been addressed in other sectors. For example, data availability in other fields has contributed to a shift towards establishing large-scale AI models, as evidenced by the remarkable success in natural language processing. Nevertheless, such transformations cannot be directly transferred and applied to mechanical fault diagnostics due to fundamental differences between mechanical dynamics and natural language data.

Consequently, this study is motivated by the need to advance the development of large-scale AI models for diagnosing mechanical system faults. The study aims to address the challenges of limited and reliable data, model trustworthiness, and interpretability to achieve accurate diagnoses that support intelligent machinery maintenance processes. To achieve this, the study's objectives include the development of an AI model based on a Probabilistic Neural Network and a Physics-Informed Neural Network, integrated with an interpretable mechanism that incorporates uncertainty quantification and a decision-fusion module. The findings from this study contribute to the advancement of a novel paradigm in intelligent fault data modelling and interpretation, known as "Smart Data," which represents fundamental components in fault diagnostics for enhancing both the predictive reliability and clarity of the models.

1.2. Related works

Deep learning applications in mechanical system diagnostics have been receiving attention due to their role in advancing intelligent fault diagnostics. Examining previous research that employs conventional deep neural networks, probabilistic neural networks, and physics-informed neural networks for fault diagnostics is necessary to gain insights into their applications [14]. Conventional deep neural network models, necessary to gain insights into their applications [14]. Conventional deep neural network models, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), and self-attention networks, among others, are currently used for various aspects of fault diagnosis.

CNN-based models, for instance, have been employed to develop intelligent models for bearing diagnosis [35]. The results suggest that 1-D CNNs can directly extract advanced features from vibration signals for condition classification, thereby enhancing diagnostic robustness compared to 2-D CNNs. However, employing 1-D CNNs directly on vibration signals may struggle to capture fault details when the bearing's working conditions are changing. Therefore, integrating vibration analysis as an additional module with CNNs motivates the development of a fault diagnosis model. The results indicate that using variational mode decomposition (VMD) as a filter to preprocess raw signals can enhance diagnostic performance under complex working conditions [39]. Combining methods like VMD or empirical mode decomposition (EMD) as preprocessing steps with deep neural networks has proven to enhance fault diagnosis performance (H. [24]. However, a challenge arises because these external algorithms require pre-setting parameters, making model maintenance particularly challenging. To ensure the model captures sufficient fault features and to alleviate maintenance difficulties, multi-scale models have been developed for mechanical system fault diagnosis [3]. [12]. However, using convolutional kernels to construct a multi-scale feature extraction module increases the model's computational complexity and lacks adaptability. Thus, a multi-scale coarse-grained approach is employed for multi-scale fault diagnosis modelling. The examination of the model demonstrates that the multi-scale CNN model exhibits greater robustness compared to the CNN-based model [17].

Taking into account the differences in the contribution of scale features to fault diagnosis decisions, the attention mechanism is employed to optimize the significance of scale features in fault recognition, adopting a feature fusion perspective [40]. Decision fusion may enhance the robustness of fault diagnosis models more than feature fusion. Therefore, a CNN model was developed based on weighted soft voting and a multi-scale module for diagnosing wind turbine gearboxes. The results indicate that decision fusion significantly reduces the false alarm rate [38]. To improve the capability of data-driven approaches in bearing health management, a gated recurrent unit and multi-scale information fusion were proposed to extract degradation information from vibration signals [26]. Whether by combining vibration signal processing modules with CNNs or developing end-to-end models with multi-scale feature extraction modules, the underlying idea is to increase the richness of fault information. The CNN-based model excels in local feature extraction. To enhance global feature extraction capability, a self-attention network was introduced in bearing fault diagnosis. The results indicate that the application of Transformers in mechanical fault diagnosis performs as effectively as CNNs [7].

However, fault diagnosis models based on conventional deep neural networks lack the capability to provide confidence assessments for their predictions. Overconfident predictions can significantly impact the reliability and effectiveness of an intelligent diagnosis model. Therefore, it is essential to equip the model with interval predictions for estimating fault modes in the form of ranges, rather than point estimates. **Probabilistic neural networks** are used as the basis for modelling, allowing the model to accurately reflect its level of confidence in predictions. Bayesian neural networks, a type of probabilistic neural network framework, are commonly used to develop fault prognostic models. The results indicate that uncertain predictions can deliver more fault information than traditional point-estimate models [28,37]. Prediction uncertainty comprises aleatoric and epistemic uncertainty (G. [20] Quantifying and assessing prediction uncertainty can enhance the interpretability and reliability of fault diagnosis models [1]. By decomposing and analysing uncertainty quantification (UQ), a Bayesian neural network serves as the foundation for developing a fault diagnostic model. This approach enables the recognition of faults beyond the model's inherent cognitive capabilities, thereby reducing the false

positive rate [11,42].

Motivated by the need to improve diagnostic performance, Transformers are used as the foundation for developing the intelligent model. The findings suggest that by leveraging self-attention's global feature extraction capability, the Bayesian Transformer-based fault diagnosis model outperforms the 1-D CNN Bayesian model when dealing with out-of-distribution data [36]. To achieve cross-device fault prognosis, Bayesian neural networks are also used for transfer learning. The results demonstrate that the Bayesian neural network-based model can accurately predict faults even with limited data [43]. Developing a probabilistic fault diagnosis model based on Bayesian neural networks involves higher computational complexities compared to Monte Carlo (MC) dropout-based probabilistic neural network modelling. A probabilistic model for fatigue life prediction is developed using a self-attention network with MC dropout (L. [4]. The examination reveals that MC dropout can address uncertain predictions as effectively as Bayesian neural networks, with the added advantage of greater modelling flexibility.

Although the aforementioned studies have successfully applied probabilistic neural networks to mechanical system fault diagnosis and provided interpretability to deep learning fault diagnosis through uncertainty quantification and assessment, the underlying mechanisms of fault diagnosis remain inherently challenging to visualize through model interpretation. Furthermore, purely data-driven neural networks, which lack crucial physical information related to mechanical system failures, compromise the model's expertise in fault diagnostics. To address these challenges, **physics-informed neural networks** have been proposed as a foundation for fault diagnostic modelling and structural health monitoring [41]. There are various approaches to embedding physics into neural networks. For example, fault characteristic information and their corresponding harmonics can be incorporated as additional features for neural network learning. Studies have shown that these supplementary features enable the neural network to acquire a more comprehensive understanding of mechanical faults, enriching the model's knowledge of the physical system [31]. This physics-informed approach is similar to manually extracting physical characteristics for neural network learning, a process that can be challenging for model maintenance and updating. For instance, a fault characteristic extractor based on kurtosis spectrum has been developed for feature filtering. The results indicate that employing frequency domain analysis and judiciously incorporating frequency domain fault characteristics enhance the neural network's comprehension of mechanical faults, potentially providing interpretable insights [29].

Considering the time dependencies in vibration data, a gear failure vibration mechanism was used as the physical information to guide the selection of hyperparameters for a long short-term memory (LSTM) network, demonstrating the diversity of physics-informed neural network approaches [6]. To further facilitate the neural network's understanding of the physical aspects of mechanical faults, a physics-informed module was developed to extract vibration-dominant modes, showing that embedding physical information enhances the model's robustness in fault diagnostics under non-stationary conditions [27]. Additionally, the combination of wavelet transforms and CNNs has been used to visualize fault characteristics in the frequency domain, enabling a more interpretable diagnostic process. This research demonstrates that frequency analysis of model features can provide evidence of mechanical failures (T. [21]. Moreover, fault diagnostics based on digital twins also fall under the category of physics-informed data-driven approaches. The knowledge learned by neural network-based models from the Digi-Twins of a gearbox provides accurate degradation information about gear wear progression [8]. A digital twin-assisted dual transfer method was developed to better migrate physical information from the digital twin model to match real sensor information for accurate fault diagnosis Z. [22].

However, the aforementioned research, which involves training a physics-informed neural network through a supervised learning approach and constraining fault categories with state labels, can lead the model to draw inferences that violate physical laws. Therefore, it is necessary to redefine the training approach for fault diagnosis to make the model more physically meaningful.

1.3. Summary and main contribution

Based on a state-of-the-art review of traditional neural networks, probabilistic neural networks, and physics-informed neural networks in mechanical system fault diagnosis, it can be summarized that both conventional and physics-informed neural networks fall under the category of point-estimate neural networks. Models from these networks are incapable of providing the necessary confidence in fault diagnosis assessments. The black-box nature of conventional neural network-based models makes them challenging to interpret and prone to over-confident fault predictions. Despite efforts to embed physical information, state-of-the-art physics-informed neural networks fundamentally rely on conditional labels for supervised learning. This reliance limits their direct physical application due to constraints between data and labels, rendering the models inexplicable and difficult to trust.

In contrast, probabilistic neural networks can achieve both fault pattern recognition and uncertainty quantification, offering a more trustworthy interpretation of fault diagnosis. However, both conventional and probabilistic models fall short in explaining the physical mechanisms behind fault diagnosis. Additionally, improving the reliability of these models should be rooted in their interpretability, requiring both aspects to be studied together to enhance trustworthy fault diagnosis.

To address the limitations and challenges associated with neural networks in fault diagnosis, this paper proposes a framework called Physics-Informed Probabilistic Deep Neural Networks (PIPDN) with an interpretable mechanism to develop a trustworthy fault diagnostic model. PIPDN possesses the capability for uncertainty predictions inherent in probabilistic neural networks. The PIPDN model exhibits high reliability and strong interpretability by accurately analysing predicted uncertainties and vibration characteristics. Additionally, PIPDN can generate Smart Data to provide precise fault information corresponding to mechanical dynamics, serving as a valuable foundation for establishing large-scale AI models for Prognostics and Health Management (PHM).

The main contributions of this paper are summarized as follows:

i. A new physical labelling module based on conditional labels to supervise the neural network in learning the vibration characteristics of mechanical failures for physics-informed neural network training.
ii. A Physics-Informed Probabilistic Neural Network (PIPDN) framework for developing a mechanical fault diagnostic model, enabling the intelligent model to possess high trustworthiness, strong interpretability, and the capability to generate Smart Data that elucidates fault information for the development of large-scale AI models.
iii. An Uncertainty Quantification (UQ)-based decision fusion module for multi-scale fault information fusion, aimed at controlling uncertainty prediction and reducing the false positive rate in fault diagnosis.
iv. An innovative integration of a physics-informed probabilistic neural network with the mechanical system fault knowledge domain to enhance the interpretability of the neural network's decision-making process. This integration provides a deeper understanding of how the model arrives at specific diagnostic outcomes.

The performance of the developed PIPDN framework, along with the Multi-Scale PIPDN diagnosis model, is examined using a rotating machinery experimental dataset. Overall, the proposed PIPDN framework

Input: $\mathcal{X}$: Set of input signals

$\mathcal{Y}$: Set of labels

fs: Sampling frequency

f_r: Rotational frequency

Z: Number of rolling elements

D: Pitch diameter of bearing

d: Diameter of rolling elements

θ: Contact angle of bearing

n: Number of harmonics to extract

Define physical Information:

For each x, y **in** $\mathcal{X}, \mathcal{Y}$ do:

> **Obtain frequency domain information** $X(f)$ **for** x(t)
>
> **Determine the fault type: based on label** y, **calculate fault characteristics** using physical parameters given in Equation (3) to (5)
>
> **Extract** n harmonics within the range of fault characteristics: $[\mathrm{FCF}_i - 5 \times \Delta f, \mathrm{FCF}_i + 5 \times \Delta f]$ from $X(f)$, setting the rest of the band to zero.
>
> **Obtain filtered features:** $X_{physics}(f)$

Output: Angle information of $X_{physics}(f)$: β through Eq. (6) and the result of the inverse Fourier transform of $x_{physics}(t)$

Output: Physical labels $\mathcal{X}_{physics}$ including $x_{physics}$ and $\beta_{physics}$

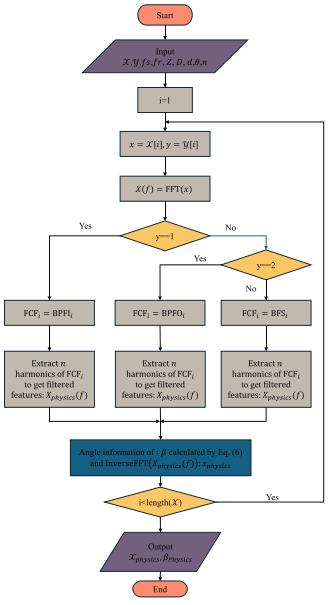


Figure 1 flowchart of Algorithm 1

contributes to improving prediction reliability and explaining the fault diagnosis mechanism of neural networks from physical perspectives. The PIPDN can generate Smart Data for developing large-scale AI models. The development of a multi-scale PIPDN fault diagnosis model serves as an illustration of the PIPDN framework and UQ-based decision-fusion module. The diagnostic mechanism of the multi-scale PIPDN is explained by considering the perspectives of uncertainty quantification and physical principles, interpreting the reasons for the reduction in false positive rates in fault diagnosis.

The structure of the rest of the paper is as follows: Section 2 introduces the proposed methodology, Section 3 describes the dataset and experimental setup, Section 4 presents case studies, and Section 5 offers conclusions.

2. Proposed methodology

2.1. Physical labelling module

In studies of supervised deep learning (DL)-based fault diagnosis modelling, conditional labels are often assigned to features in a mandatory manner. Using conditional labels to constrain the learned representational features in models is typically limited to classification tasks. This process frequently overlooks the fact that these representational features should have physical significance. To address the limitations of conditional labels, a novel physical labelling module is proposed in this section. These labels are designed to professionally and physically address the fault diagnostic task.

Generally, bearing fault diagnosis relies on vibration analysis to obtain reliable conclusions. Thus, in simple terms, the proposed physical labelling module incorporates the theoretical fault characteristic frequency of the bearing as a value label.

$$X(f) = \mathrm{FFT}(x(t)) \tag{1}$$

where $x(t)$ represents a vibration signal sample, and $\mathrm{FFT}(x(t))$ denotes the application of the Fast Fourier Transform (FFT) to $x(t)$, resulting in $X(f)$ containing amplitude and phase information at different frequency components f. The normalized frequency f_n can be calculated by Eq. (2).

$$f_n = f/f_r \tag{2}$$

where, f_r is the rotational frequency. In this study, we focus on three specific types of faults. For a given type of machinery, the characteristic frequency of the fault can be determined using Eqs. (3)–(5).

$$\mathrm{BPFI} = \frac{Z \bullet f_r}{2}\left(1 + \frac{d}{D}cos\theta\right) \tag{3}$$

$$\mathrm{BPFO} = \frac{Z \bullet f_r}{2}\left(1 - \frac{d}{D}\cos\theta\right) \tag{4}$$

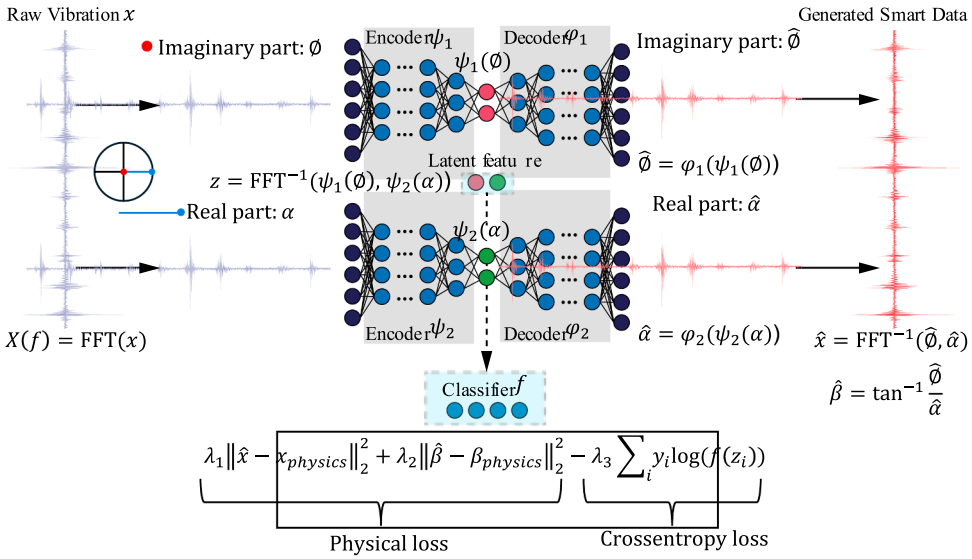


Fig. 2. Schematic of the PIPDN framework.

$$\mathrm{BSF} = \frac{f_r \bullet D}{d}\left(1 - \left(\frac{d}{D}\cos\theta\right)^2\right) \tag{5}$$

The fault characteristic frequencies (FCFs), such as BPFI, BPFO, and BSF, are typically considered the gold standard in vibration analysis for diagnosing bearing faults. These frequencies serve as the basis for our analysis. For each fault condition, we extract the fault characteristic frequencies and their harmonics from $X(f)$ as essential physical labels for subsequent analysis. For example, for a given characteristic frequency like BPFI, the frequency interval for the i^{th} harmonic is defined as $[\mathrm{BPFI}_i - 5 \times \Delta f,\ \mathrm{BPFI}_i + 5 \times \Delta f]$, where Δf is the frequency spacing, calculated as the sampling frequency divided by the length of the signal (i.e. the number of samples). The process for labeling physical information for each fault is detailed in Algorithm 1, which considers a total of n harmonics in feature extraction.

Algorithm 1 defines the physical labelling procedure used to transform and process the input data in the frequency domain, with the goal of obtaining filtered information from the input signal. A step-by-step explanation of Algorithm 1 *is as follows. (I) Perform Fourier transform on input signal domain signal x via Eq.* (1) *to obtain frequency signal* $X(f)$*. (II) Define bearing fault characteristics of each sample x in dataset* $\mathscr{X}$*, which is calculated by the corresponding theoretical fault characteristic frequency (FCF) provided label y. (III) Calculate the frequency spacing and normalize the fault characteristics. By iterating through each sample, generate masks for each sample. (IV) Apply these masks on original frequency signal* $X(f)$ *to obtain masked features* $X_{physics}(f)$*. (V) Calculate the angle information of* $\beta_{physics}$ *via Eq.* (6)*, where* $\varnothing = X_{physics}.imag$ *and* $\alpha = X_{physics}.real$ *are imaginary part the real parts of* $X_{physics}(f)$*, respectively. Perform the inverse FFT on* $X_{physics}(f)$ *to obtain* $x_{physics}(t)$ *for each sample. The angle information* $\beta_{physics}$ *and the* $x_{physics}(t)$ *are the physical labels* $\mathscr{X}_{physics}$ *for the proposed PIPDN model training.*

$$\beta = \tan^{-1}\frac{\varnothing}{\alpha} \tag{6}$$

2.2. Physics-informed probabilistic deep network (PIPDN) main body

Fig. 2 illustrates the framework of the PIPDN network, which comprises of two autoencoders and uses the proposed Algorithm 1.

As shown in Fig. 2, the collected raw vibration signal $x(t) \in \Re^n$ serves the input data. Real part α and imaginary part ϕ are obtained by applying FFT to the $x(t)$. Two encoders are then employed to extract advanced information from α and ϕ, resulting in $\psi(\alpha)$ and $\psi(\phi)$, respectively. The encoder ψ is developed based on a stack of layers including convolutional layer $\mathrm{Conv1d}(\bullet)$, a dropout layer $\mathrm{Dropout}(\bullet)$ for regularization and uncertainty prediction based on Monte Carlo dropout, an activation layer $\mathrm{ReLU}(\bullet)$, and batch normalization layer $\mathrm{Batchnorm1d}(\bullet)$.

$$\mathrm{Conv1d}(x) = \mathbb{W} * x + \mathbb{b} \tag{7}$$

$$\mathrm{Dropout}(\mathrm{x}) = \begin{cases} 0 \\ \dfrac{\mathrm{Conv1d}(x)}{1 - \mathbb{p}} \end{cases} \tag{8}$$

$$\mathrm{Batchnorm1d}(\mathrm{x}) = \frac{\mathrm{Conv1d}(x) - \mathrm{E}(\mathrm{Conv1d}(x))}{\sqrt{\mathrm{Var}(\mathrm{Conv1d}(x)) + \varepsilon}} \tag{9}$$

$$\mathrm{ReLU}(\mathrm{x}) = \max(0, x) \tag{10}$$

The decoder φ is similarly constructed using a stack of layers, but with the convolutional layer replaced by transpose convolution layer $\mathrm{ConvTranspose1d}(\bullet)$. The operation of ConvTranspose1d is defined as:

$$\mathrm{ConvTranspose1d}(x) = \mathbb{W} \circledast x + \mathbb{b} \tag{11}$$

where * denotes convolutional operation, $\mathrm{E}(\bullet)$ represents the sample mean operation and $\mathrm{Var}(\bullet)$ is sample variance operation, and $\circledast$ is transpose convolutional operation.

The latent space feature $z(t) \in \Re^r$ is obtained by inverse FFT, following $z(t) = \mathrm{FFT}^{-1}(\psi_1(\alpha), \psi_2(\phi))$. This feature reflects the same fault representative information as the generated features in output space $\hat{x}(t) \in \Re^n$. Consequently, it is also used in the classifier $f(\bullet)$ to obtain diagnostic results $\hat{y}$. Unlike conventional neural network-based fault diagnosis applications, the features $z(t) \in \Re^r$ in the latent space are determined by conditional labels and physical labels $x_{physics}$ and $\beta_{physics}$.

To enhance the physical representation of the feature zzz in the latent space, the distance between the physical label $x_{physics}$ and generated smart data $\hat{x}$ is used as loss function for optimizing the PIPDN model parameters. Additionally, the similarity between physical label $\beta_{physics}$ and the generated angle information $\hat{\beta}$ is introduced as the penalty loss component in the overall loss, where $x_{physics}$ and $\beta_{physics}$ are the outputs from Algorithm 1, $\hat{\phi} = \varphi_1(\psi_1(\phi))$ and $\hat{x} = \mathrm{FFT}^{-1}(\hat{\phi}, \hat{\alpha})$. The parameters of the encoders ψ and decoders φ used to process amplitude α and phase ϕ are denoted by θ_1, θ_2 and μ_1, μ_2 respectively. The pa-

rameters of the classifier $f(\bullet)$ are ω. Thus, the output of model PIPDN($\bullet$) for input data x can be calculated by Eq. (11). Typically, the output $\hat{y}$ is the result of softmax normalization.

$$\hat{x}, \hat{y} = \mathrm{PIPDN}(x|\theta_1, \mu_1, \theta_2, \mu_2, \omega) \tag{12}$$

The training and testing processes of the PIPDN framework, including model optimization, uncertainty prediction, uncertainty quantification, are presented in Algorithm 2.

Algorithm 2: PIPDN model optimization and uncertainty quantification of its uncertainty prediction

Model Optimization
Input:
Training Data $\mathscr{D}_{train} = \{\mathscr{X}, \mathscr{Y}\}$ Models $\psi_i(\bullet)$, $\varphi_i(\bullet)$ and $f(\bullet)$ with initialized parameters θ_i, μ_i and ω; learning rate η
Output: $\psi_i(\bullet|\hat{\theta}_i)$, $\varphi_i(\bullet|\hat{\mu}_i)$ and $\hat{f}_j(\bullet|\hat{\omega})$
Training PIPDN branch network:
for *epoch* **in** maximum epochs:
Samples $\left[x, y, \left(x_{physics}, \beta_{physics}\right)\right]$ **in** $(\mathscr{X}, \mathscr{Y}, \mathscr{X}_{physics})$ with usage of **Algorithm 1**
Down sampling by encoders: $\psi_1(\alpha)$ and $\psi_2(\phi)$
Laten space features: $z = \mathrm{FFT}^{-1}[\psi_1(\alpha), \psi_2(\phi)]$
Up sampling by decoders: $\hat{\phi} = \varphi_2[\psi_2(\phi)]$ and $\hat{x} = \mathrm{FFT}^{-1}(\hat{\phi}, \hat{\alpha})$, where $\hat{\alpha} = \varphi_1[\psi_1(\alpha)]$
Fault prediction: $\hat{y} = f(z)$
loss calculation: by
$\lambda_1 \|\hat{x} - x_{physics}\|_2^2 + \lambda_2 \|\hat{\beta} - \beta_{physics}\|_2^2 - \lambda_3 \sum y \log(\hat{y})$
(By performing k iterations of Monte Carlo sampling, the loss and performance are averaged over the k iterations)
Parameters θ_i, μ_i and ω are optimized by optimizer with learning rate η
Saving the best performance models (By performing k iterations of Monte Carlo sampling, the loss and performance are averaged over the k iterations)
end
Output: $\psi_i(\bullet|\hat{\theta}_i)$, $\varphi_i(\bullet|\hat{\mu}_i)$ and $f(\bullet|\hat{\omega})$
end

After training the model, MC dropout is used to conduct uncertainty prediction and quantify the uncertainty of this dynamic prediction results. Assuming the test data is represented by $\mathbb{x}$, to sample the model parameters K times, for the k^{th} sampling, the prediction of the PIPDN model is.

$\hat{\mathbb{x}}^k, \hat{\mathbb{y}}^k = \mathrm{PIPDN}\left(\mathbb{x}|\theta_1^k, \theta_2^k, \mu_1^k, \mu_2^k, \omega^k\right)$ according to Eq. (12), where $\hat{\mathbb{y}}^k$ represents the probability distribution of the estimated status. $\hat{\mathbb{x}}^k$ is the smart data, which can be used for diagnostic explanation in this study. In the case where $\hat{\mathbb{y}}^k$ equals to k^{th} output of the PIPDN model, the average prediction after sampling K times is $\overline{\mathbb{y}}$. Based on the entropy, the model is able to quantify the prediction uncertainty [13]. The total uncertainty PU for each $\mathbb{x}$ can be quantified by Eq. (12).

$$\mathrm{PU} = \mathbb{H}(\overline{\mathbb{y}}|\mathbb{x}) = -\sum_{c=1}^{C} \mathbb{p}(\overline{\mathbb{y}} = c|\mathbb{x}) \times \log[\mathbb{p}(\overline{\mathbb{y}} = c|\mathbb{x})] \tag{13}$$

By considering total uncertainties, PU can be decomposed into aleatoric (AU) and epistemic uncertainties (EU) that respectively refer to the uncertainty inherent to the input samples and model parameters. The AU can be approximated by Eq. (14).

$$\mathrm{AU} \approx -\frac{1}{K}\sum_{k=1}^{K}\sum_{c=1}^{C} \mathbb{p}\left(\overline{\mathbb{y}}^k = c|\mathbb{x}\right) \times \log\left[\mathbb{p}\left(\overline{\mathbb{y}}^k = c|\mathbb{x}\right)\right] \tag{14}$$

According to the definition, epistemic uncertainty is the difference between total uncertainty and aleatoric uncertainty, which can be approximated as:

$$\mathrm{EU} \approx -\sum_{c=1}^{C} \mathbb{p}(\overline{\mathbb{y}} = c|\mathbb{x}) \times \log[\mathbb{p}(\overline{\mathbb{y}} = c|\mathbb{x})] + \frac{1}{K}\sum_{k=1}^{K}\sum_{c=1}^{C} \mathbb{p}\left(\overline{\mathbb{y}}^k = c|\mathbb{x}\right) \times \log\left[\mathbb{p}\left(\overline{\mathbb{y}}^k = c|\mathbb{x}\right)\right] \tag{15}$$

2.3. Multi-scale diagnostic framework with uncertainty quantification (UQ) based decision-making module

Motivated by the multi-scale (MS) module's ability to improve diagnostic performance, this section introduces a novel MS-PIPDN model, developed based on the PIPDN framework. The fusion of information at the decision-making level has the potential to enhance the model's robustness. Therefore, a UQ-based decision-fusion module is proposed as a component of the MS-PIPDN model. Fig. 3 presents the diagnostic flowchart using the MS-PIPDN model with the UQ-based decision-making module.

Fig. 3 illustrates how the MS-PIPDN framework establishes an explainable and reliable neural network for fault diagnosis. The MS-PIPDN model operates directly on the raw data collected by sensors from rotating machinery, using a *multi-scale extractor* to capture multi-scale features from the raw vibration signals. The multi-scale extractor employs average pooling, and for each data point, the multi-scale feature is calculated using Eq. (16).

$$\mathrm{output}[l'] = \frac{1}{\tau}\sum_{l=l'}^{l'+\tau-1} \mathrm{input}[l] \tag{16}$$

The input is a raw vibration signal $x(t)$, where l represents the position over the time series $x(t)$, and τ is the kernel size of the average pool, representing the multi-scale factor. A padding procedure is used to ensure that the length of the output sequence remains the same as the input sequence. In this study, the maximum scale factor is set to three, consistent with the multiscale diagnostic research by .

[17]. Consequently, there are three PIPDN models applied to each scale of the vibration features. The PIPDN models the multi-scale features. Specifically, at each scale, physical labels are obtained through the physical label module to train a PIPDN model in addition to traditional conditional labels. In this study, the hidden features extracted by a PIPDN model from the multi-scale features are defined as branch networks.

The training process for the MS-PIPDN model follows a similar approach to training the PIPDN model, with the distinction that the loss function comprises the sum of losses (Eq. (17) across all multi-scale branch networks, where n is the number of network branches.

$$\frac{1}{n}\sum_{j=1}^{j=n}\left[\lambda_1\|\hat{x}_j - x_{physics(j)}\|_2^2 + \lambda_2\|\hat{\beta}_j - \beta_{physics(j)}\|_2^2 - \lambda_3\sum y\log\left(\hat{y}_j\right)\right] \tag{17}$$

According to Eq. (14), by quantifying and decomposing the uncertainty in the PIPDN model's fault predictions, the epistemic uncertainty for each sample can be obtained. To make multi-scale feature fusion more interpretable and to improve diagnostic reliability, this study proposes a decision fusion module based on uncertainty quantification, referred to as the UQ-based decision fusion module.

Algorithm 3 implements the reconsideration of uncertain predictions by MS-PIPDN based on the epistemic uncertainty. Where j is the index of the scale-factor, $j = 1$ means the branch network 1 in the MS-PIPDN model. Traditional multi-scale neural network models [40] use attention mechanisms to fuse each neural network branch with weighted coefficients. However, since the weight computation is still based on neural network outputs, the interpretability remains weak. Therefore, the proposed UQ-based decision module reconsiders the diagnostic results of each branch network based on

the cognitive uncertainty of the test samples. Lower epistemic uncertainty indicates that the branch is more confident and reliable in its diagnosis.

Algorithm 3: *UQ-based decision-fusion module*

Uncertainty Quantification and UQ-based decision fusion module:

Input: *Testing Data* $\mathscr{X}_{test} = \mathscr{X}_{test}$, *MS-PIPDN model:* $\mathrm{PIPDN}_j(\bullet|\widehat{\theta}_1, \widehat{\mu}_1, \widehat{\theta}_2, \widehat{\mu}_2, \widehat{\omega})$, *sampling time* K

Output: *Diagnostic result* $\widehat{\mathscr{Y}}$

for $\mathbb{x}$ *in* $\mathscr{X}_{test}$

Sampling *K times from the branch network of MS-PIPDN model:*$\mathrm{PIPDN}_j(\mathbb{x}|\widehat{\theta}_1, \widehat{\mu}_1, \widehat{\theta}_2, \widehat{\mu}_2, \widehat{\omega})$

where k^{th} *prediction is* $\widehat{\mathbb{y}}_j^k = \mathrm{PIPDN}_j\left(\mathbb{x}|\widehat{\theta}^k{}_1, \widehat{\mu}^k{}_1, \widehat{\theta}^k{}_2, \widehat{\mu}^k{}_2, \widehat{\omega}^k\right)$

Record $\widehat{\mathbb{y}}_j = \left\{\widehat{\mathbb{y}}_j^k \middle| k \in N, 1 \le k \le \mathrm{K}\right\}$

Obtain *epistemic uncertainty* $\mathbb{EU} = \left\{\mathrm{EU}_j \middle| j \in N, 1 \le k \le \mathrm{n}\right\}$ *by* ***Eq.*** **(13)** *to* ***Eq.*** **(15)**

Select trustworthy prediction:

Trustworthy index: $idx = \mathrm{argmin}(\mathbb{EU})$

Trustworthy diagnosis *for* $\mathbb{x}$**:** $\widehat{\mathbb{y}} = \left\{\widehat{\mathbb{y}}_j \middle| j \in N, 1 \le k \le \mathrm{n}\right\}[idx]$

end

Diagnosis result *for* $\mathscr{X}_{test}$**:** $\widehat{\mathscr{Y}} = \{\widehat{\mathbb{y}}\}$

end

Output: Fault diagnostic: $\widehat{\mathscr{Y}}$***; Smart Data:*** $\widehat{\mathscr{X}}$ *calculated by* ***Eq.*** **(12)**

end

3. Experimental setup and investigation

3.1. Bearing dataset

In this study, two bearing fault datasets are used to validate the effectiveness, reliability, and superiority of the proposed PIPDN framework and MS-PIPDN model in fault diagnosis. Dataset I [30], encompasses a more comprehensive array of fault types, providing a richer and more diverse set of scenarios for analysis. This diversity is crucial for developing robust fault detection methods that can accurately identify various types of bearing malfunctions. Dataset I also includes early-stage fault instances, enabling the development and validation of models capable of detecting faults at their nascent stages. This allows for the implementation of generalization tests that track the evolution of faults, ensuring that the developed models not only perform well on specific fault patterns but also exhibit robustness and adaptability as faults progress over time.

As illustrated in Fig. 4, the experimental rig for Dataset I comprises six components, each serving a specific function: (1) an accelerometer, (2) a microphone, (3) a load cell, (4) a Hall effect sensor, (5) a motor temperature thermocouple, and (6) a room temperature thermocouple. All data were recorded with the accelerometer positioned inside the motor casing, close to the drive-end bearing, to facilitate accurate data capture. Each data sample is 10 s in length and is collected at a sampling rate of 42,000 Hz.

All fault characteristics in the experiment are determined through spectrum analysis. Each bearing corresponds to a unique characteristic frequency. The physical parameters and characteristic frequencies of the bearings are outlined in Table 1. In this dataset, each raw sample is labelled in the format {Letter}-{Number}-{Number}. The letters in the dataset labelling signify the condition of the tested bearing. Specifically, "H" denotes a healthy bearing, "I" indicates an inner race fault, "O" represents an outer race fault, and "B" signifies a ball fault. For example, the data sample labelled "I-2-1" corresponds to an inner race fault in bearing 2 with a developing fault, whereas "I-2-2" represents an inner race fault in bearing 2 that is fully developed.

In addition to Dataset I, Bearing Dataset II is used to further validate the performance of the PIPDN and MS-PIPDN models under time-varying speed conditions. In this experiment (Fig. 5), we incorporate

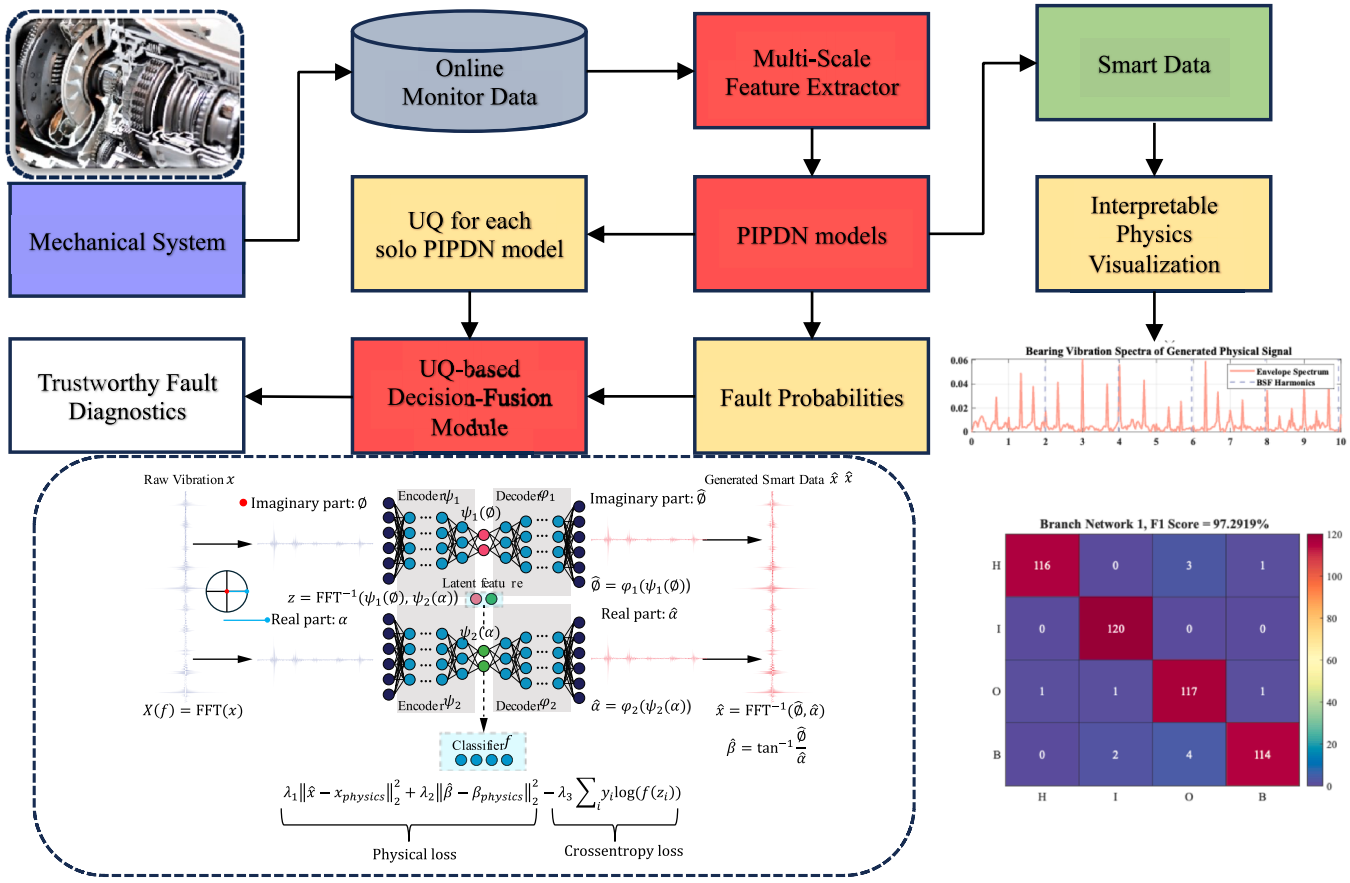


Fig. 3. MS-PIPDN diagnostic framework with UQ-based Decision-Fusion module.

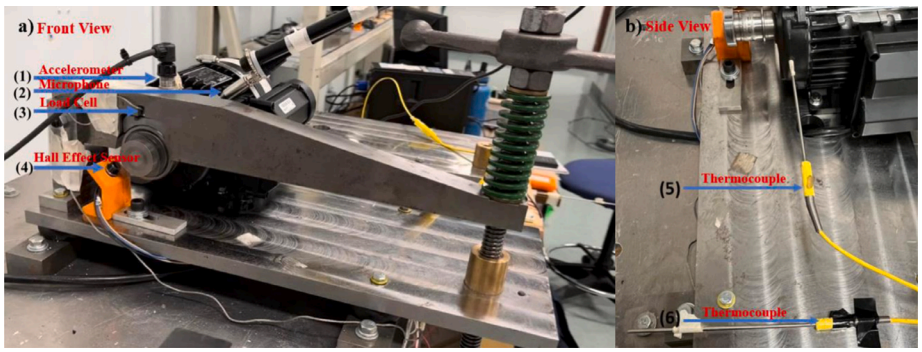


Fig. 4. Experimental rig setup for dataset-I operation.

Table 1
Parameters of bearing dataset-II.

Bearing type	Pitch diameter	Ball diameter	Number of balls	BPFI	BPFO	BSF
NSK 6203ZZ	28.50	6.77	8	4.95 f_r	3.05 f_r	3.98 f_r

three distinct bearing health states: normal, inner race wear, and outer race wear. The setup for the working conditions is innovative, with the rotational speed varying over time. This study considers three working states: increasing rotational speed, decreasing rotational speed, and rotational speed initially increasing and then decreasing. These states are used to validate the reliability of the diagnostic models.

For example, in test H-A-1, "H" represents the healthy state of the bearing, "A" indicates the change in rotational speed, and "1" signifies the first test. For each fault and each speed condition, a total of three tests were performed. As another example, in test O-3, "O" stands for outer race damage, and "3" represents the third test. The experiments for Dataset II were conducted on a SpectraQuest Machinery Fault Simulator (MFS-PK5M). The shaft is driven by a motor, with its rotational speed regulated by an AC drive. The shaft is supported by two ER16K ball bearings: one on the left, which is in optimal condition, and the other on the right, which serves as the experimental bearing. This experimental bearing is replaced with bearings exhibiting various health conditions. An ICP accelerometer (Model 623C01) is mounted on the housing of the experimental bearing to collect vibration data. Additionally, an incremental encoder (EPC Model 775) is used to monitor the rotational speed of the shaft.

All fault characteristics in the experiment are determined through spectrum analysis, with each bearing corresponding to a unique characteristic frequency. The physical parameters and characteristic frequencies of the bearings are outlined in Table 2. In this dataset, the vibration is sampled at 200,000 Hz for 10 s per sample.

Table 2
Parameters of bearing dataset-II.

Bearing type	Pitch diameter	Ball diameter	Number of balls	BPFI	BPFO
ER16K	38.52	7.97	9	5.43 f_r	3.57 f_r

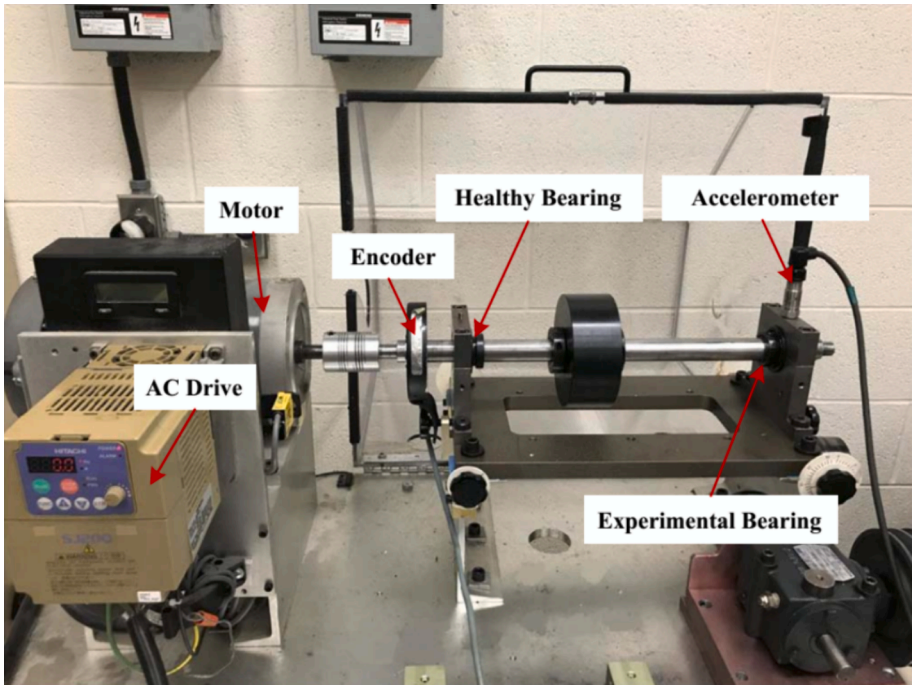


Fig. 5. Experimental rig setup for dataset-II.

Table 3
Overview of Diagnostic Examination Scenarios.

Scenario	Training data	Testing dataset
I-A	70 % × H, I-1, O-6, B-11	30 % × H, I-1, O-6, B-11 with SNR=-9dB~9dB
I-B	70 % × H, I-2, O-7, B-12	30 % × H, I-2, O-7, B-12with SNR=-9dB~9dB
I-C	70 % × H, I-3, O-8, B-13	30 % × H, I-3, O-8, B-13with SNR=-9dB~9dB
I-D	70 % × H, I-5, O-10, B-15	30 % × H, I-5, O-10, B-15with SNR=-9dB~9dB
II-A	H O-6,O-7,O-8 I-1,I-2,I-3 B-11,B-12, B-13	H O-10 I-5 B-15
II-B	H O-6,O-7,O-10 I-1,I-2,I-5 B-11,B-12, B-15	H O-8 I-3 B-13
II-C	H O-6,O-8,O-10 I-1,I-3,I-5 B-11,B-13, B-15	H O-7 I-2 B-12
II-D	H O-7,O-8,O-10 I-2,I-3,I-5 B-12,B-13, B-15	H O-6 I-1 B-11
III-A	H, I-1-1, O-6-1, B-11-1	H, I-1-2, O-6-2, B-11-2 with SNR=-9dB
III-B	H, I-2-1, O-7-1, B-12-1	H, I-1-2, O-7-2, B-12-2 with SNR=-9dB
III-C	H, I-3-1, O-8-1, B-13-1	H, I-1-2, O-8-2, B-13-2 with SNR=-9dB
III-D	H, I-5-1, O-10-1, B-15-1	H, I-1-2, O-10-2, B-15-2 with SNR=-9dB
IV-A	H-1, H-2, I-1, I-2, O-1, O-2	H-3, I-3, O-3
IV-B	H-1, H-2, I-1, I-2, O-1, O-2	H-3, I-3, O-3
IV-C	H-1, H-2, I-1, I-2, O-1, O-2	H-3, I-3, O-3

3.2. Diagnostic experimental setup

In this study, all the diagnostic models are implemented using Pytorch library. Five state-of-the-art methods, including DCNN [42], PINN [29], PINet [27], MSCNN [40] and Transformer [42] are used for comparison to demonstrate the superiority of the proposed diagnostic framework. The DCNN serves as the baseline architecture for feature extraction and classification in the comparison models.

In Table 3, four main scenarios are designed to examine the diagnostic performance of the models. White Gaussian noise is added to the raw signals to simulate environmental pollution. This process is defined by $x_{noisy} = \text{awgn}(x, \text{snr})$ based on Eq. (18), where x is the raw signal, x_{noisy} the signal with added white Gaussian noise, and *snr*, represents the signal–noise-ratio between x_{noisy} and x, where $\text{snr} = 10\log_{10}\left(\frac{P_x}{P_s}\right)$, $P_x = \frac{1}{N}\sum_{n=0}^{N-1}|x[n]|^2$, $P_{noisy} = \frac{1}{N}\sum_{n=0}^{N-1}|x_{noisy}[n]|^2$.

$$x_{noisy} = x + s \tag{18}$$

Under each main scenario, there are four sub-scenarios that represent different aspects of how the main scenarios are composed. Scenarios I to III are constructed using dataset-I, while Scenario IV is constructed with dataset-II. **Scenario-I** aims to evaluate the model's resilience to noise interference. Both the training and test sets are carefully curated to ensure there is no data overlap. To simulate a noisy test environment, noise is intentionally introduced to the test samples, allowing us to assess the model's performance amidst environmental disturbances. **Scenario-II** is designed to showcase the effectiveness of part inspection. In this scenario, the training and test datasets are sourced from entirely separate bearings. This deliberate separation ensures that the model's ability to inspect parts is thoroughly tested across diverse datasets, reflecting real-world scenarios where parts may vary significantly. **Scenario-III** assesses the model's robustness against data drift due to equipment aging. In this scenario, the training data comes from bearings

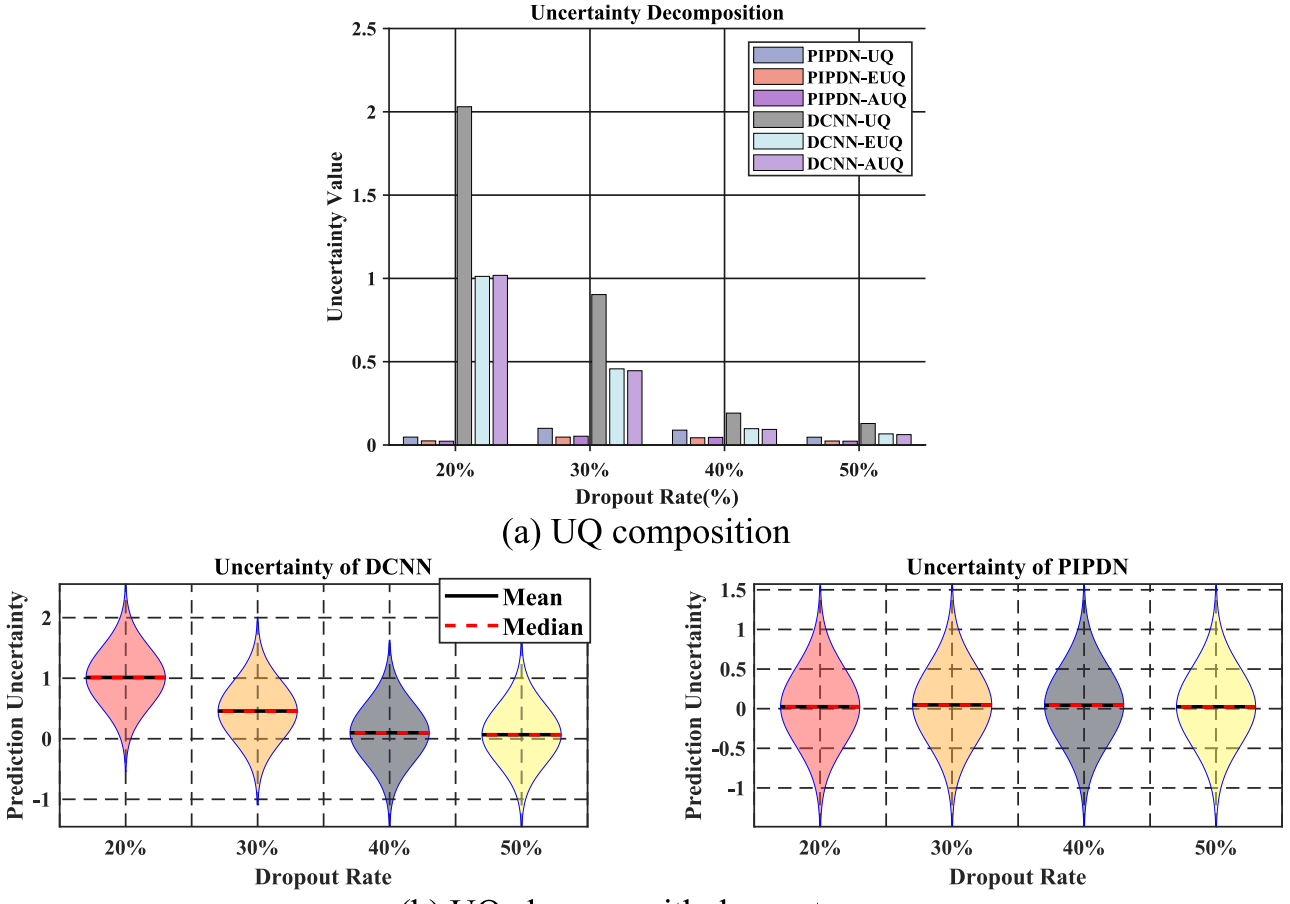


Fig. 6. Prediction uncertainties of the DCNN in the PIPDN model.

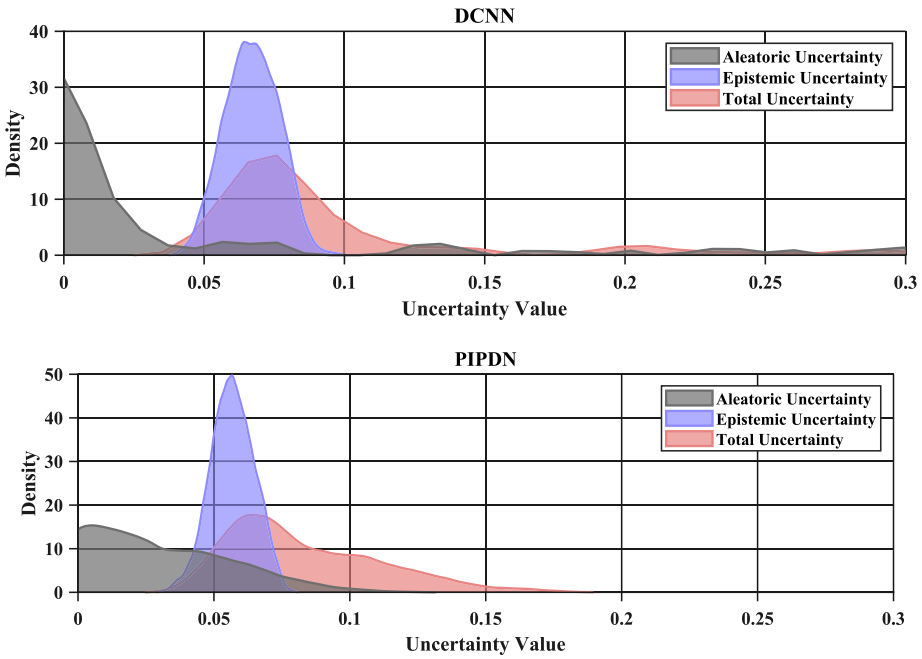


Fig. 7. Uncertainty decomposition and analysis for the baseline and proposed models.

with early-stage structural damage, while the test data is from bearings with advanced deterioration. By exposing the model to different stages of equipment degradation, we can evaluate its ability to maintain performance despite changes in the data distribution over time. **Scenario-IV** focuses on fault diagnosis under dynamic rotational speed conditions. Both the training and test sets are sourced from distinct trials, ensuring no overlap between the datasets.

3.3. Hyper parameters setup

The hyperparameter setup for the PIPDN model is detailed in Table 3. The hyperparameters for the PIPDN model are set to be consistent with those of the baseline model and other state-of-the-art models. The dropout rate, which controls prediction uncertainty, is examined in the range from 0.1 to 0.5, with 0.5 being selected for analysis and comparison. The fully connected layer is configured with dimensions of 8196 (input), 420 (hidden), and the number of output units corresponding to the bearing working conditions. The mini-batch size during training is set to 64. The learning rate is initialized at 0.001 to control the step size during optimization, decreasing by 2 % after every 50 steps. The model undergoes a maximum of 200 training epochs, with early stopping enabled to enhance training efficiency. The convolutional kernel sizes are specified as [5, 4, 3, 2], and the stride values are set at [5, 4, 3, 2]. Transpose convolutional operations use kernels of sizes [5, 4, 3, 2]. Additionally, minor vibration signals and normalized rotating speeds are processed by the physical labelling module. The number of selected harmonics is set to 10, as considering excessively high harmonic orders would increase computational complexity. Consequently, interpretable fault information is typically observed in the low-frequency range. These hyperparameter configurations collectively form the foundation of the PIPDN model's architecture and contribute to its effective performance in fault diagnosis tasks.

4. Discussion and analysis

4.1. Uncertainty quantification and analysis

In this subsection, the prediction uncertainty of the proposed PIPDN model is compared with that of the baseline model, DCNN, in the context of fault diagnosis. Information entropy is employed to quantify prediction uncertainties. Since the dropout rate plays a significant role in addressing uncertainty quantification (UQ) through the MC dropout methodology, the examination of UQ is conducted with four different dropout rates under Scenario I. Fig. 6 illustrates the predictive uncertainty for both the baseline DCNN and the proposed PIPDN models.

As depicted in Fig. 6, UQ represents the total average predictive uncertainty, while AU and EU denote the average aleatoric and epistemic uncertainties, respectively. The total uncertainties for both the baseline and the proposed models decrease as the dropout rates increase. This occurs because higher dropout rates simplify the models and reduce their complexity, leading to a decrease in predictive uncertainties. Comparing the baseline DCNN model to the PIPDN model, the predictive uncertainty of the PIPDN model is consistently lower than that of the baseline DCNN model, regardless of whether low or high dropout rates are used. Epistemic uncertainty helps determine whether the models have acquired sufficient knowledge to confidently handle a specific task. Consequently, the epistemic uncertainty (EUQ) of the PIPDN model's predictions is significantly lower than that of the DCNN model, indicating that the PIPDN model has greater confidence in diagnosing bearing faults.

Examining Fig. 6b reveals that the uncertainty of the PIPDN model remains consistently stable as the dropout rate varies. This stability is a notable advantage of the PIPDN model over the DCNN. Additionally, the independence of the PIPDN model from dropout rate variations has been effectively validated, demonstrating the positive impact of incorporating physical labels to supervise model training. To ensure that each model can effectively identify healthy bearing conditions, a dropout rate of 0.5 will be used for all compared methods in the subsequent analysis and discussion.

Analysing aleatoric and epistemic uncertainties helps assess the stability and reliability of diagnostic models. Therefore, Fig. 7 illustrates the distribution of aleatoric and epistemic uncertainties in fault prediction, indicating whether the model can make stable predictions.

Fig. 7 illustrates the distributions of predictive uncertainty along with its decomposition. From the perspective of total uncertainty, the PIPDN model's average estimation is closer to zero, which is significantly lower than that of the DCNN model. Although the uncertainty

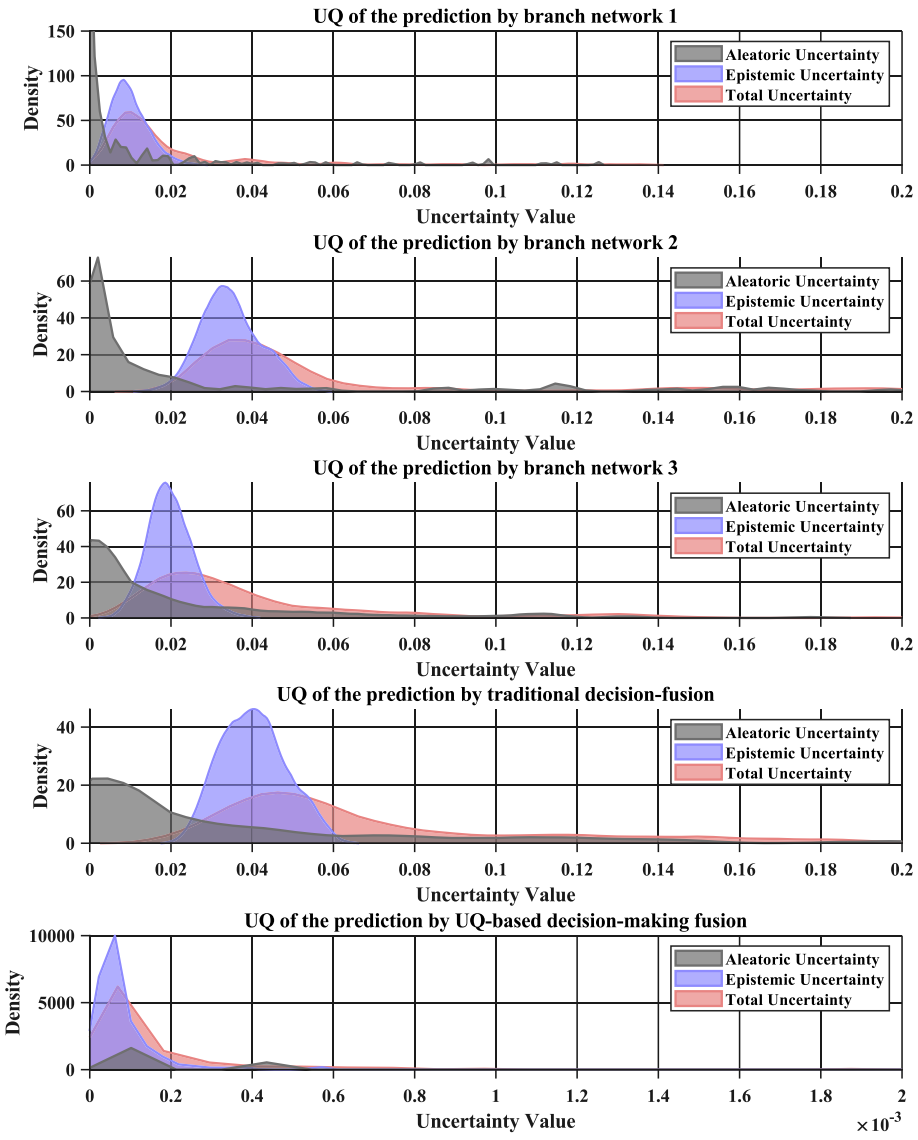


Fig. 8. The mechanism of the proposed UQ-based decision-making module for MS-PIPDN diagnostics.

distribution of PIPDN exhibits a longer tail, the predictive confidence of these extreme values also intersects with the lower bounds of DCNN's primary predictive confidence. It should be noted that epistemic uncertainties reflect the model's understanding of the task data. Observations of the epistemic uncertainties of the two models indicate that the upper bounds of epistemic uncertainties in PIPDN's distribution remain lower than the lower bounds of DCNN's epistemic uncertainties distribution. Consequently, PIPDN exhibits a better cognitive capacity for bearing fault diagnosis compared to DCNN. These results provide a second motivation for developing the UQ-based decision-making module.

Moreover, since the MS-PIPDN framework is developed based on the proposed PIPDN model, where UQ is used as the basis for the proposed UQ-based decision-making fusion module, the mechanism of this module is investigated to demonstrate its reliability and effectiveness for more trustworthy fault prediction.

Fig. 8 illustrates the uncertainty quantification (UQ) distributions of various branch networks and their fusion within the MS-PIPDN model, highlighting the importance of the proposed UQ-based decision-making module. It is evident that traditional decision fusion methods tend to increase the expected uncertainty quantification (EUQ) of predictions. EUQ is a critical indicator of a model's understanding of fault knowledge. Therefore, traditional fusion methods, such as averaging predictions from different branches in a multi-scale deep learning-based fault diagnosis model, may enhance prediction accuracy but also reduce confidence. In contrast, a UQ-based decision fusion approach not only improves prediction accuracy but also enhances confidence in the predictions.

4.2. *Reliability of the proposed trustworthy and interpretable diagnostic framework*

It is essential to ensure the interpretability of neural network-based fault diagnostic models. One of the innovations of this study is the introduction of a physical labelling module. These physical labels serve as guidance for the neural network to better incorporate the underlying physics associated with conditional labels. In the analysis section, the PIPDN model learns features generated from physical signals, and the latent features are used for fault classification. Here, the order is given by Hz/f_r defined as Eq. (2) for following analysis.

The right columns in Figs. 9–11 compare the raw features with those learned by the PIPDN model in both the time and frequency domains. In the time domain, the features learned by PIPDN exhibit clearer impact characteristics than the raw signals. In the frequency domain, the PIPDN

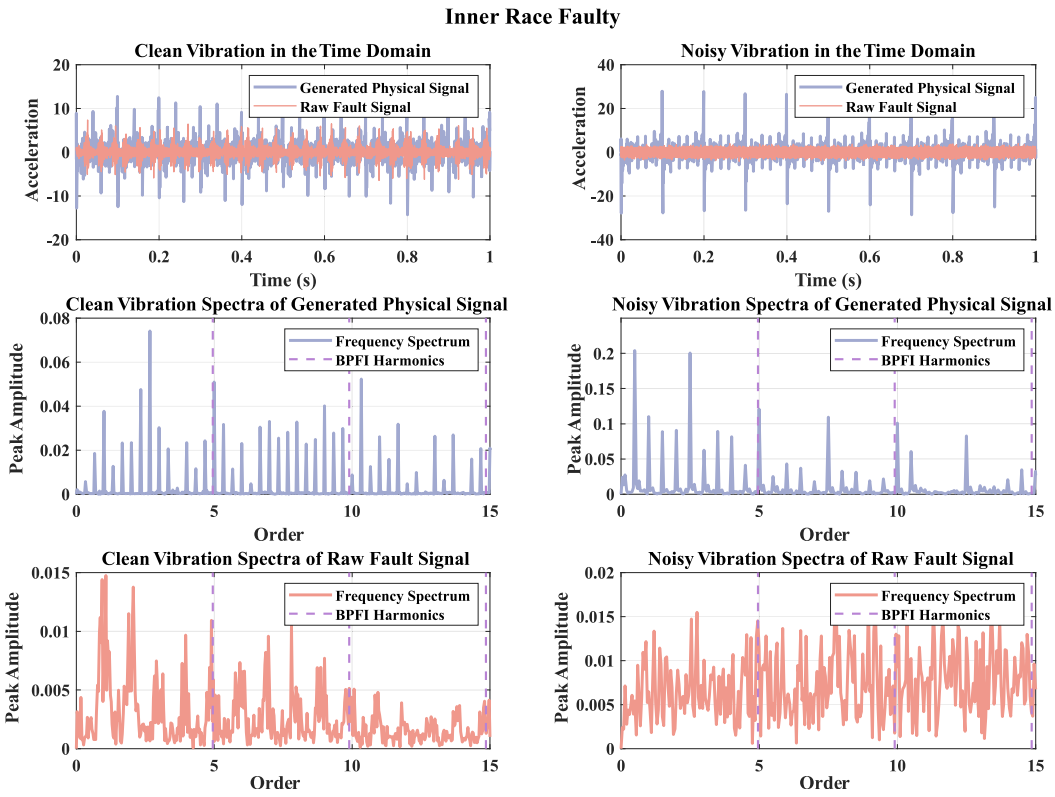


Fig. 9. Inner race fault: clean and noisy vibration.

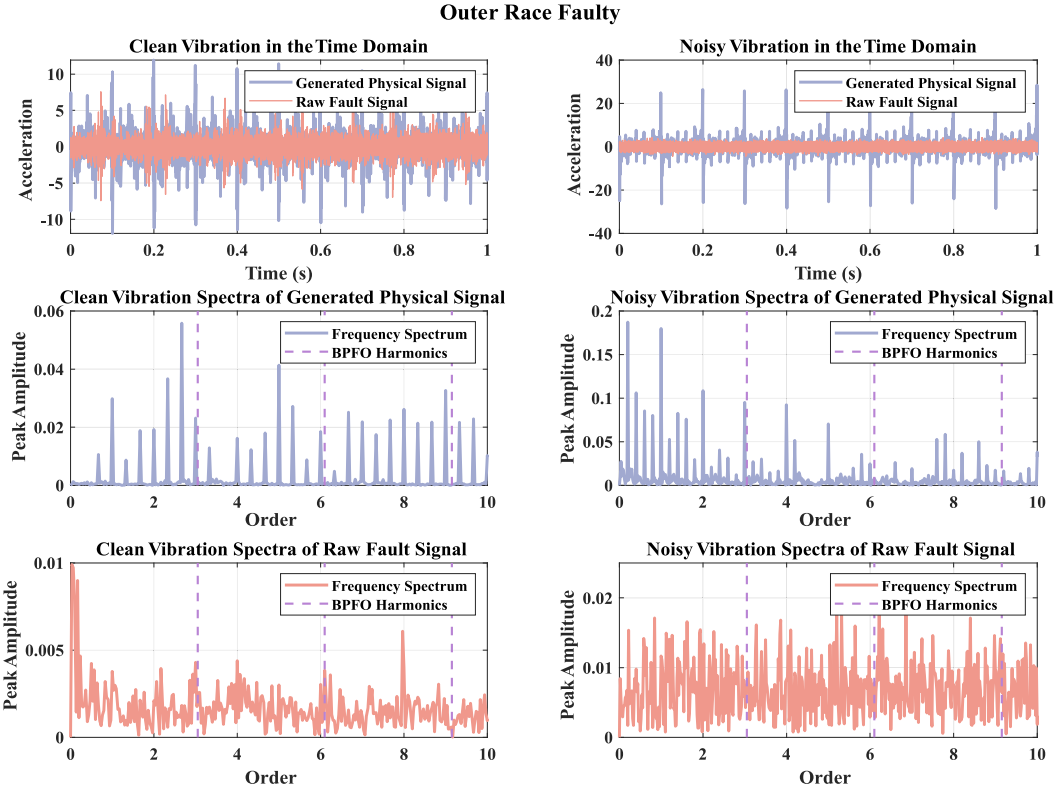


Fig. 10. Clean Vibration Signal of an Outer Race Fault.

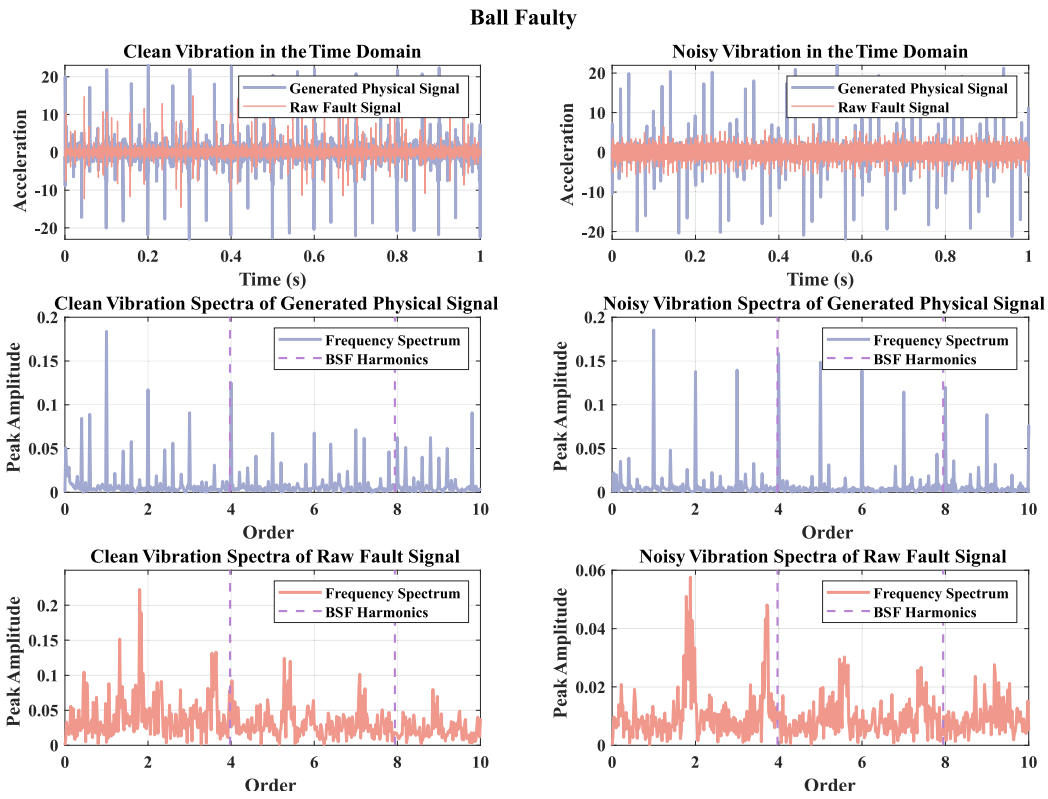


Fig. 11. Ball fault clean vibraion.

features clearly highlight fundamental frequencies and faulty harmonics, including the inner fault characteristic frequency (close to 5X), outer-race fault characteristic frequency (close to 3X), and ball fault characteristic frequency (close to 4X). The frequency information conveyed by these features aligns perfectly with the BPFI, BPFO, and BSF harmonics.

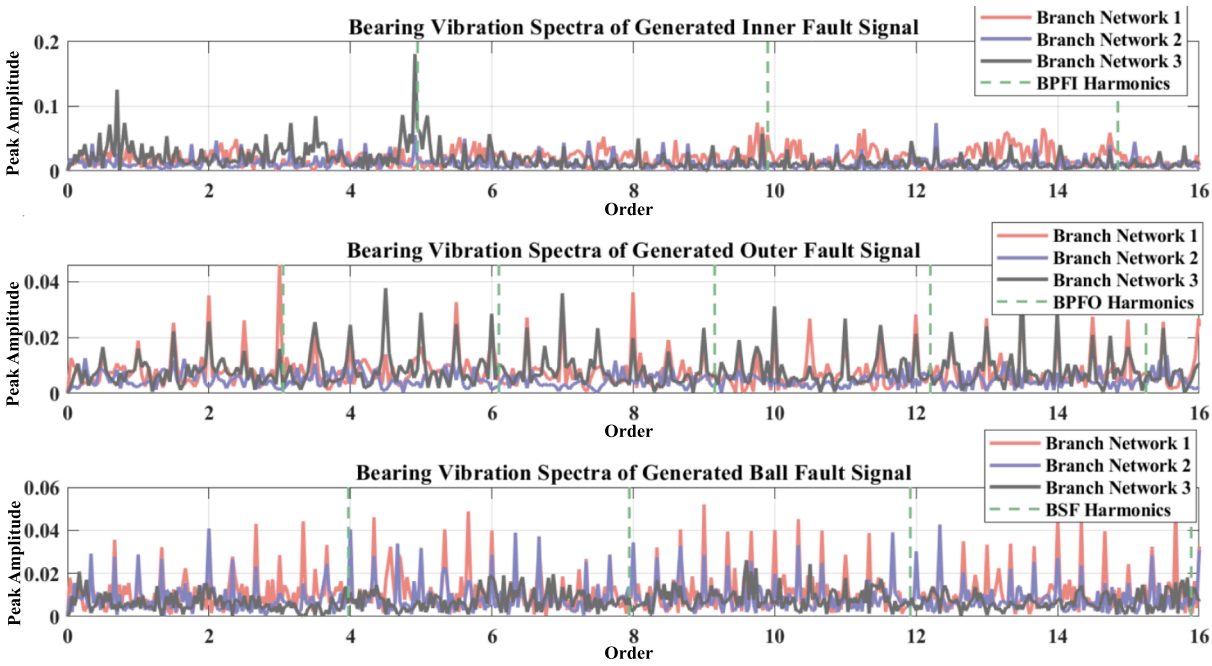
However, in comparison, the frequency domain characteristics of the raw vibration signal are strongly affected by harmonics due to sub-harmonics generated by nonlinear structural vibrations and noise interference, which mask the frequencies and make isolating fault characteristics challenging. This is why using raw vibration signals directly as input features for neural network modelling is prone to the influence of unrelated noise characteristics, resulting in inaccurate representations even with advanced fault diagnostic features. The results demonstrate that the proposed PIPDN model improves the accuracy of understanding the physical characteristics associated with bearing failure modes.

To further examine the capabilities of the physical labelling module, the right columns in Figs. 9–11 also assess the physical characteristic extraction ability of PIPDN for noisy signals. The results show that the PIPDN model effectively enhances physical understanding, even under additional noise interference. When a high level of noise is added, it becomes nearly impossible to discern any fault vibrations in the time or frequency domains, as they are largely obscured. However, the smart data generated by the PIPDN model clearly demonstrates the impact characteristics in both domains. It is easy to identify the actual fault characteristic frequencies corresponding to the theoretical positions of BPFI, BPFO, and BSF harmonics. The clarity of these fault harmonics explains why PIPDN can still provide effective fault diagnostic signals despite noise interference. This outcome illustrates the interpretability of the PIPDN model in deep learning-based fault diagnosis.

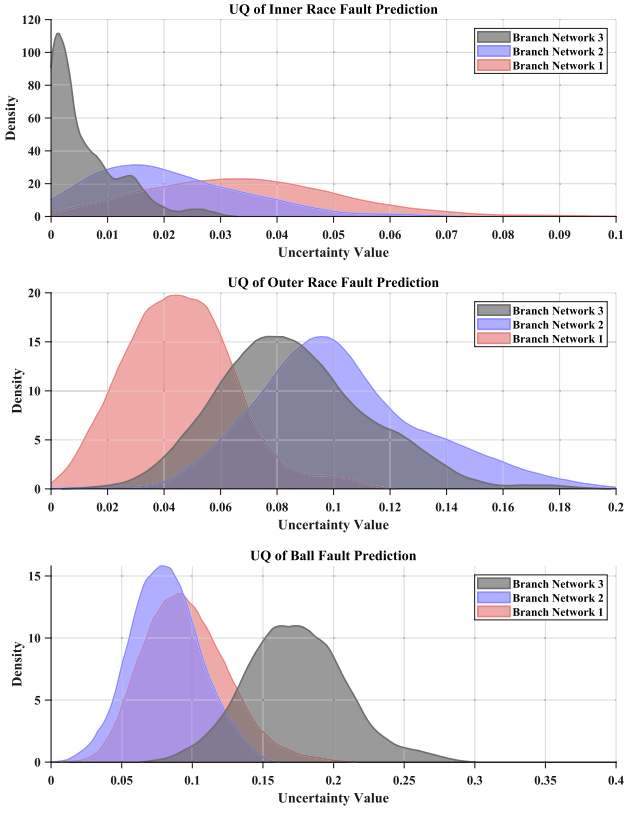
To elucidate the physical mechanism of the UQ-based decision-making, the module is used to assist the MS-PIPDN model in diagnosing faults more effectively and accurately. Fig. 12 illustrates the prediction uncertainty and the associated physical information for the MS-PIPDN in identifying each operational condition, combined with confusion matrix analysis to explain the diagnostic mechanism.

Fig. 12 illustrates the physics of the MS-PIPDN model and its associated uncertainty prediction capability, providing an explanation of the diagnostic mechanisms in terms of both uncertainty quantification (UQ) and physics. For diagnosing inner race faults in bearings, it is observed that branch network 3 in the MS-PIPDN model exhibits a higher level of confidence in predictions compared to other branch networks. This is attributed to the fact that this specific branch network more accurately captures the characteristics of inner race faults, particularly at the harmonics (5X) associated with inner race faults, compared to other branch networks. For diagnosing outer race faults, it is observed that branch network 1 in the MS-PIPDN model exhibits a higher level of confidence in predictions compared to other branch networks. This is because this specific branch network more accurately captures the characteristics of outer race faults, particularly at the harmonics (3X) associated with outer race faults, compared to other branch networks. However, a notable distinction arises in the analysis of outer race faults. For outer race diagnostics, the UQ distributions of the three branch networks are more similar. This phenomenon is also evident in the physics, as the outer race fault harmonics learned by each branch network closely align with the characteristic frequency of outer race faults. For ball creak faults, based on the analysis of UQ, branch network 2 plays a leading role in diagnostics because network 2 can more clearly extract ball creak harmonics (3.98X) from the vibration analysis compared to the others.

Combining the analyses from Sections 4.1 and 4.2, it is evident that while conventional decision-making fusion has the potential to enhance diagnostic accuracy, it also amplifies the uncertainty associated with its predictions. Visualizing the vibration in the frequency domain reveals the reliability of failure predictions. In the ensuing analysis, using a confusion matrix to explain the mechanism of the UQ-based decision fusion module demonstrates how it enhances diagnostic performance while maintaining reliability.



(a) Physical information of each branch networks



(b) Epistemic uncertainty of each branch network prediction

Fig. 12. Fault diagnosis mechanism of the MS-PIPDN model.

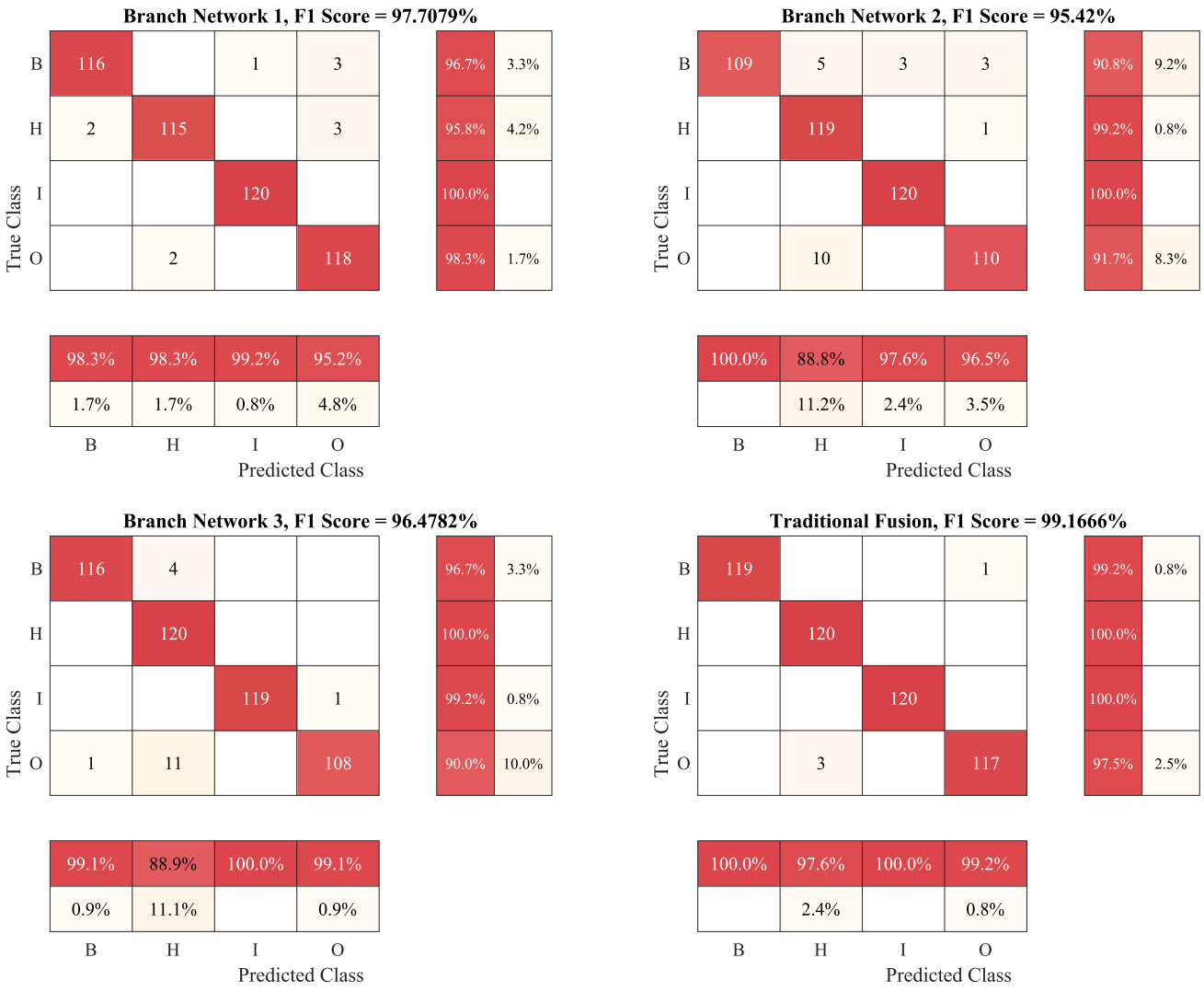


Fig. 13. Diagnostic mechanism analysis of MS-PIPDN.

The MS-PIPDN model is developed to address the high false alarm rate associated with diagnosis models based on single-branch networks. As illustrated in Fig. 13, Branch Network (BN) 2 and BN3 both exhibit false alarms when identifying outer race faults. However, the reasons for these errors differ between the two networks. For BN2, nearly half of the outer race faults are misclassified as ball faults, while the remaining faults are identified as healthy bearings. In contrast, more than half of the outer race faults are incorrectly classified as healthy bearings by BN3. At the same time, BN1 tends to mistakenly classify healthy bearings as outer race faults and ball faults as outer race faults. The difficulty in diagnosing outer race faults compared to other faults arises because their impact characteristics closely resemble those of healthy conditions. However, after applying the decision-level fusion with the proposed UQ-based decision-fusion module, the false alarms for outer race faults, ball faults, and healthy conditions are significantly reduced. By integrating UQ analysis, physical vibration analysis, and confusion matrix, the diagnostic mechanism is enhanced, improving the reliability of the neural network-based diagnostic model.

4.3. Diagnostic examination of the proposed framework

In real industrial engineering applications, the collected signals are often contaminated by various types of noise. A common practice is to manually add noise to the signals in order to assess the anti-noise capabilities of diagnostic models. Fig. 14 demonstrates the diagnostic performance of the models tested with noise levels ranging from −9 dB to 9 dB (Scenario I).

Fig. 14 illustrates the diagnostic performance of the proposed MS-PIPDN method under different noisy environments, compared to models from state-of-the-art research. For the test data at −9 dB, the performance of DCNN closely approximates that of PINN. As the signal-to-noise ratio increases, the diagnostic performance of PINN surpasses that of DCNN, indicating that embedding physical features provides limited assistance to fault diagnostic performance in high-noise environments. PINet constructs a physical feature extraction module and integrates it into the neural network model, effectively functioning as a filter that enhances fault information extraction. Consequently, there is a significant improvement in fault diagnosis performance compared to DCNN. Although the Transformer model does not explicitly embed a physical information extraction module, its unique self-attention mechanism allows it to focus on potential features within the signal itself, resulting in superior performance compared to PINet. The MSCNN model is more robust than PINet due to its multiple branch network structures combined with a coarse-grained feature extraction module that effectively filters high-frequency noise. This explains its superior performance compared to other fault diagnosis models based on single-branch neural network structures. However, despite these enhancements, all the aforementioned approaches improve the model's

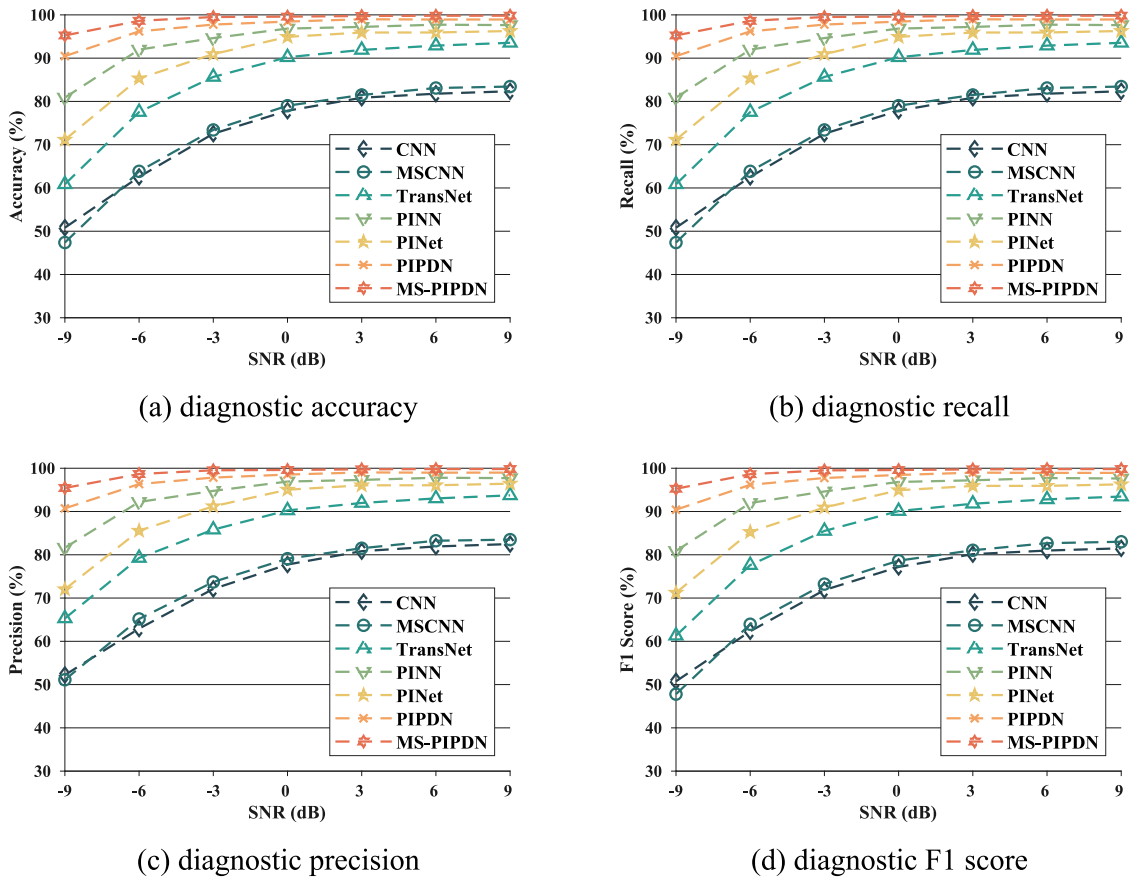


Fig. 14. Diagnostic performance examination in a noisy environment.

Table 4
Hyper parameters setup of PIPDN model.

Hyperparameter	value	Hyperparameter	value
Dropout Rate	0.5 (0.1 ~ 0.5)	Fully connected layer	8196/420/output
Mini Batchsize	64	Frequency	42000/200000
Learning Rate	0.001	Minor vibration signal	True
Max Epochs	200	Normalized rotating speed	True
Early stop	True	Number of selected harmonious	10
Conv Kernel	[5,4,3,2]	Optimizer	Adam
Stride	[5,4,3,2]	Scheduler	Step_size = 50/ gama = 0.98
TransposeConv Kernel	[5,4,3,2]	If Multi-Scale	Scale = (1, 3, 5)

knowledge about bearing failure physics and serve as filters before the learnable components. The primary learnable components of these models are still constrained by conditional labels, leading to a loss of physical meaning. This results in weaker robustness when dealing with vibration data contaminated by strong noise. The proposed PIPDN model fundamentally differs from conventional fault diagnosis models. The inclusion of the proposed physical labelling module makes it less susceptible to noise interference, thus exhibiting superior robustness compared to state-of-the-art models. Based on the PIPDN structure, the developed MS-PIPDN model captures a more comprehensive set of fault information and learns the physical characteristics. Moreover, with the inclusion of the proposed UQ-based decision-level fusion module, the MS-PIPDN model maintains diagnostic performance exceeding 95 % even in the presence of substantial noise in the collected data. The superiority of the proposed diagnostic framework is demonstrated through

Table 5
The average diagnostics performance across from Scenario II-A to I-D.

Model	Accuracy (%)	Recall (%)	Precision (%)	F1 Score (%)
DCNN	83.25 ± 1.77	83.25 ± 1.77	83.67 ± 2.03	82.40 ± 1.88
PINN	84.51 ± 2.15	84.51 ± 2.15	84.68 ± 2.31	84.07 ± 2.24
PINet	93.78 ± 1.51	93.78 ± 1.51	94.01 ± 1.52	93.70 ± 1.54
MSCNN	97.75 ± 0.94	97.75 ± 0.94	97.82 ± 0.91	97.75 ± 0.94
Transformer	96.41 ± 1.30	96.41 ± 1.30	96.55 ± 1.25	96.41 ± 1.30
PIPDN	98.97 ± 0.81	98.97 ± 0.81	99.04 ± 0.76	98.97 ± 0.80
MS-PIPDN	**99.88 ± 0.23**	**99.88 ± 0.23**	**99.88 ± 0.24**	**99.86 ± 0.22**

a comparison of its diagnostic performance with state-of-the-art fault diagnosis models. Table 4 presents the diagnostic results of the models examined in Scenario II, where the training and testing data are sourced from multiple working conditions, aligning with the assumptions made in most supervised learning-based fault diagnostic works.

In this examination, the diagnostic performance of the models is generally better than that observed in Scenario I due to the absence of environmental noise contaminating the collected data. The diagnostic performances of DCNN and PINN are relatively close, indicating that simply embedding fault characteristics as additional input features is not sufficient for a model to effectively learn mechanical failures (Table 5). PINet diagnoses bearing faults more effectively than PINN because it includes a physics-informed module, which acts as a filter to capture more representative fault features, thereby improving model learning. MS-CNN and Transformer outperform PINet by 4–5 %, due to MS-CNN's ability to acquire more comprehensive information for fault diagnosis and the self-attention mechanism in the Transformer, which enables it to capture detailed fault features.

As the foundational physics-informed neural network framework proposed in this study, the PIPDN model achieves performance

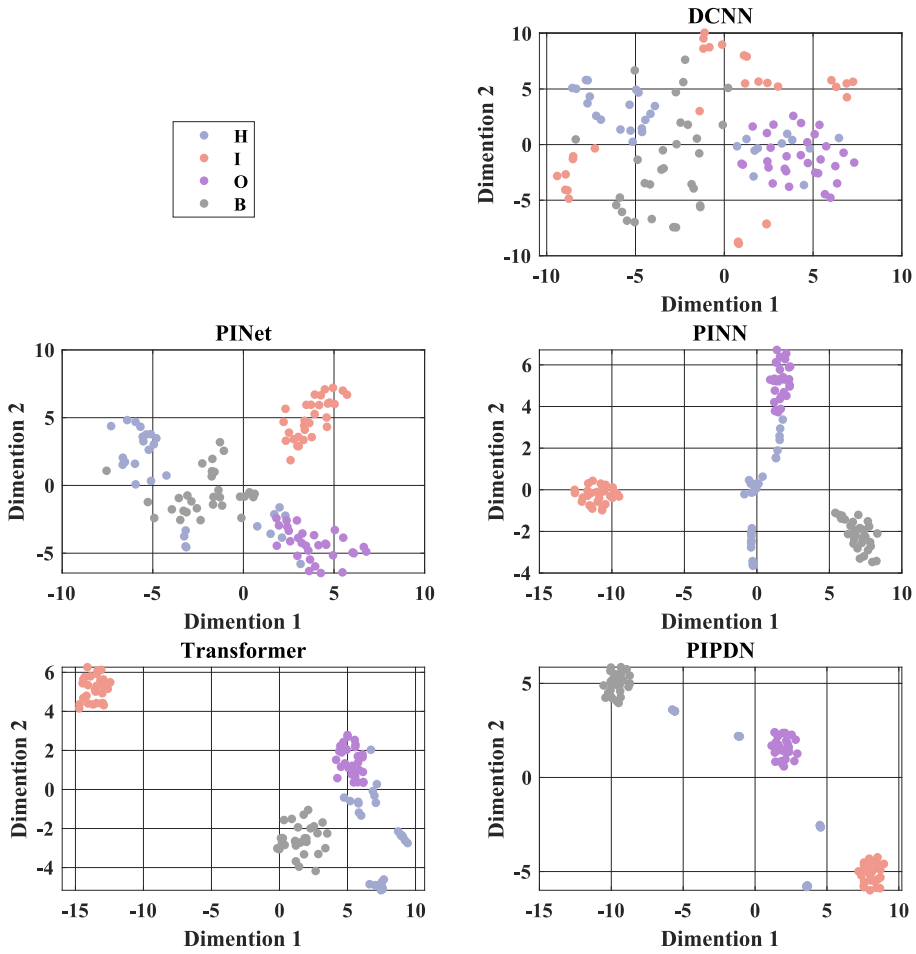


Fig. 15. Visualization of advanced features in the deeper layers of the models.

exceeding 98.95 % in Scenario II. This robust performance is attributed to its capability to learn accurate fault physics through the proposed physical labelling module. The features in PIPDN are traceable in the frequency domain and, as hidden features, provide clearer expressions for classification. MS-PIPDN achieves its highest diagnostic performance of 99.88 %. This model builds on the PIPDN framework, leveraging uncertainty prediction and the UQ-based decision-fusion module to reduce the false alarm rate. Fig. 15 visualizes the distance between advanced features of different conditional data in DCNN, PINN, PINet, Transformer, and the PIPDN model using T-SNE.

As shown in Fig. 15, the distance between each sample in the low-dimensional space reflects their true discrepancies in the original space. The distance scale in the low-dimensional space representing DCNN features is larger than that of the other models. This observation indicates that features in the corresponding original space may be over-diffused, suggesting that DCNN features may not accurately convey fault information. In contrast, the distance scale in the low-dimensional space representing advanced features in the Transformer, PINet, PINN, and PIPDN models decreases significantly. Given that the Transformer framework is one of the physics-informed methods, this suggests that embedding failure physics into a neural network can enhance its diagnostic capabilities, thereby reducing the uncertainty of the features in their original spaces. Comparing the cluster distributions of features in the Transformer, PINet, PINN, and PIPDN models, the PIPDN model demonstrates superior clustering performance, effectively grouping bearing faults based on their similarities. No misclassifications are observed because the PIPDN model can accurately learn from the representative fault features.

The superiority of the proposed diagnostic framework is demonstrated through a comparison of its diagnostic performance with state-of-the-art fault diagnosis models. Fig. 16 presents the diagnostic results of the models examined in Scenario III, where the training and testing data are sourced from different degradation stages to evaluate the models' extrapolation abilities.

In this examination, the diagnostic accuracy of each model showed a slight decline compared to the assessments in Scenarios I and II. This decline occurs because the distribution of test data changes with the degradation of bearing performance. The purpose of this test is to demonstrate the models' extrapolation capability. In industrial applications, bearing health deteriorates over time. Therefore, modelling degradation at a specific stage may significantly increase the model's false alarm rate when applied to bearings at more advanced stages of degradation. A model capable of diagnosing faults at various degradation stages can greatly reduce training costs, as it would not need to be retrained for each stage of the bearing's lifespan.

The poor performance of the models stems from their failure to incorporate physical information. Although multiscale features provide more information than those captured by DCNNs, additional information becomes less effective if the test data extends beyond the model's original knowledge due to the bearing's performance degradation. The Transformer model outperforms DCNNs due to its capability for global feature extraction. However, despite its ability to capture global features through self-attention mechanisms, the Transformer model exhibits poor extrapolation capability due to the lack of embedded physical information.

The models with embedded physical information can reduce false alarms by around 40 % for ball fault diagnosis compared to the DCNN. This improvement is attributed to PIPDN's unique approach to physical

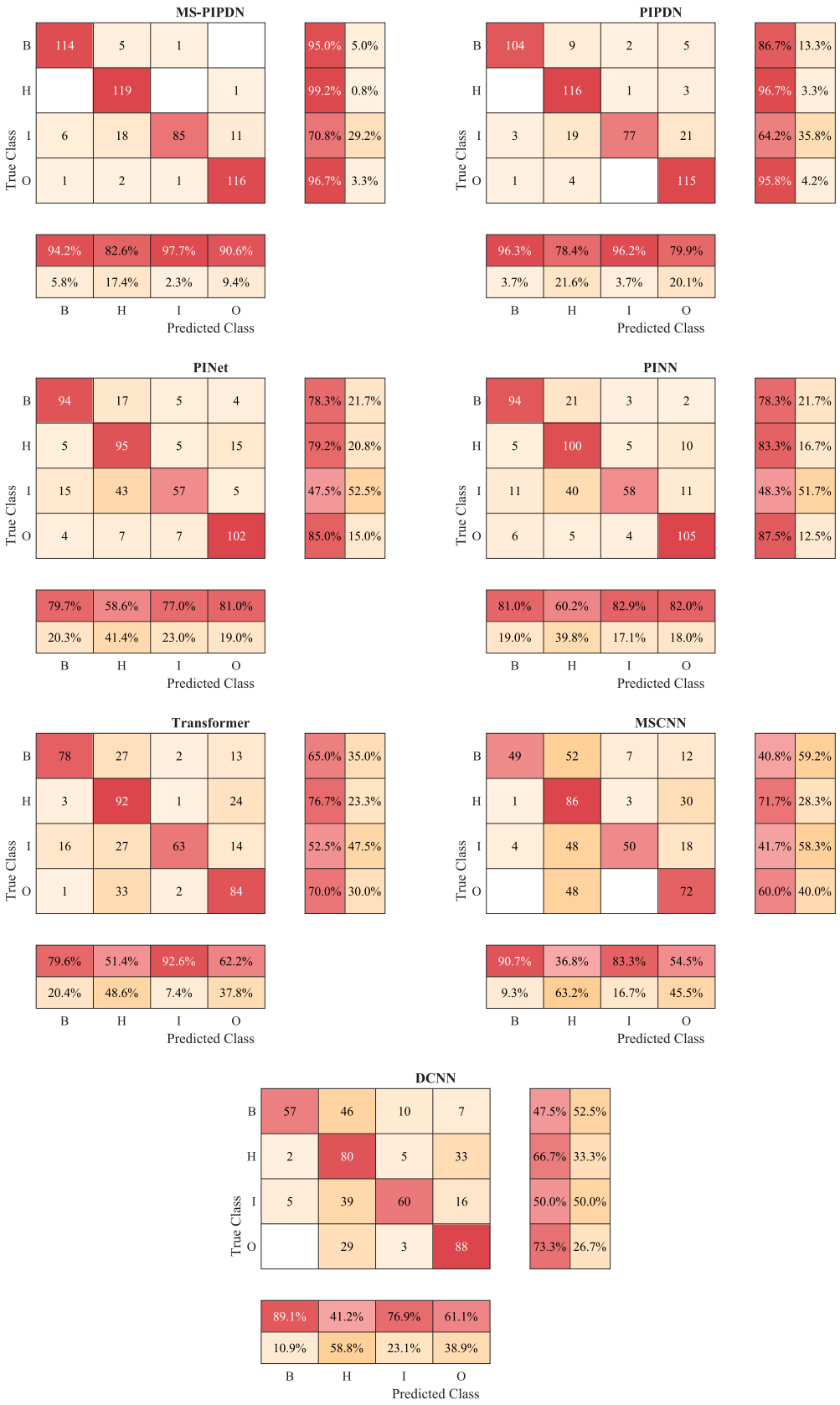
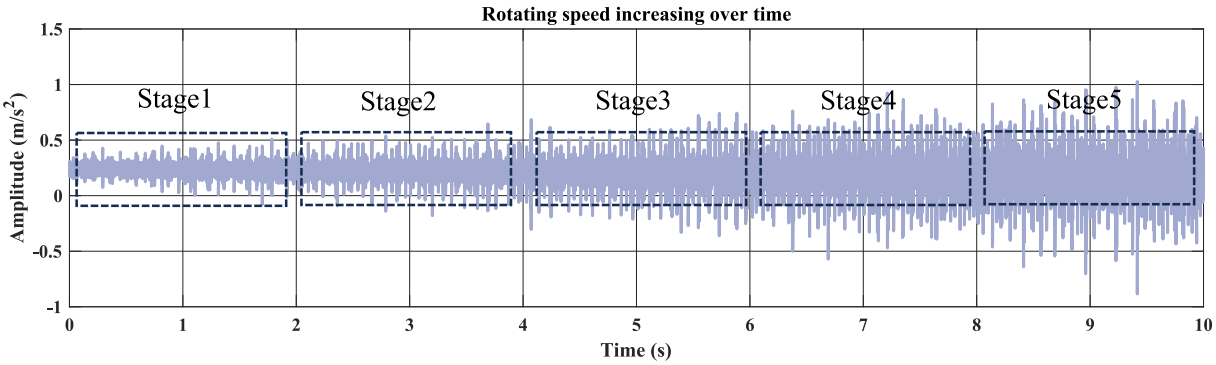
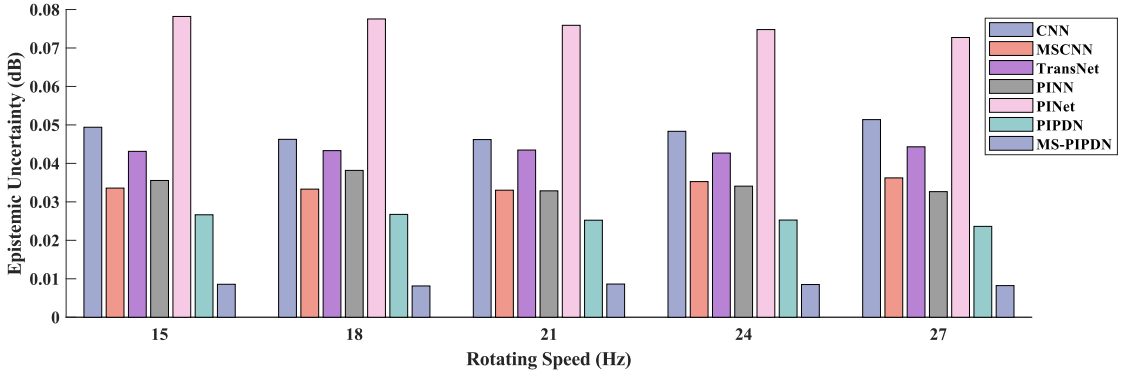


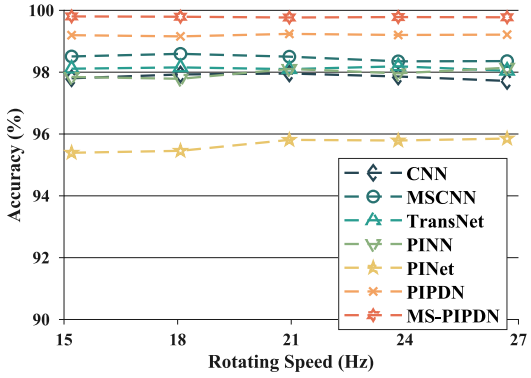
Fig. 16. Diagnostic performance examined across different degradation stages.



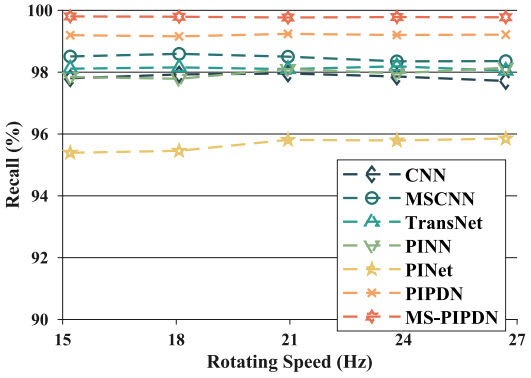
(a) raw vibration



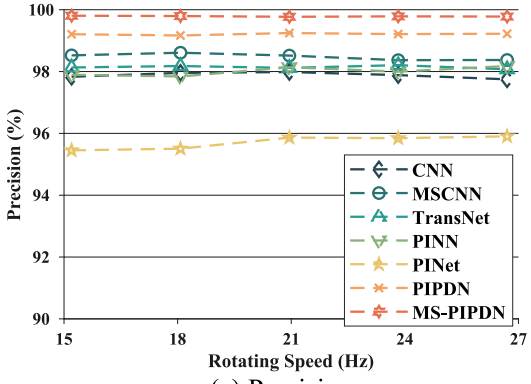
(b) epistemic uncertainty



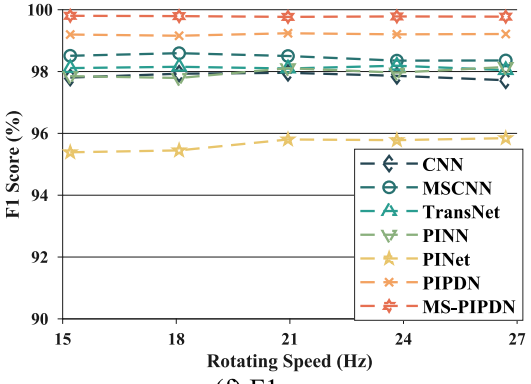
(c) Accuracy



(d) Recall

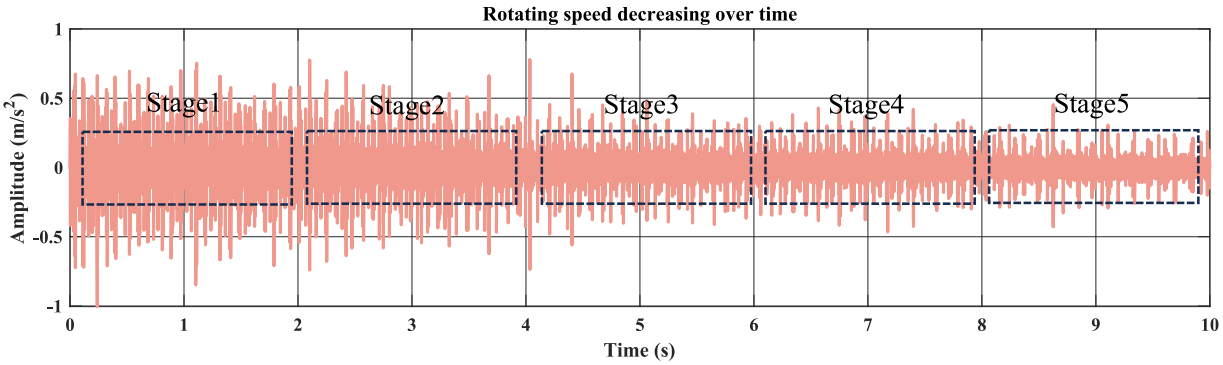


(e) Precision

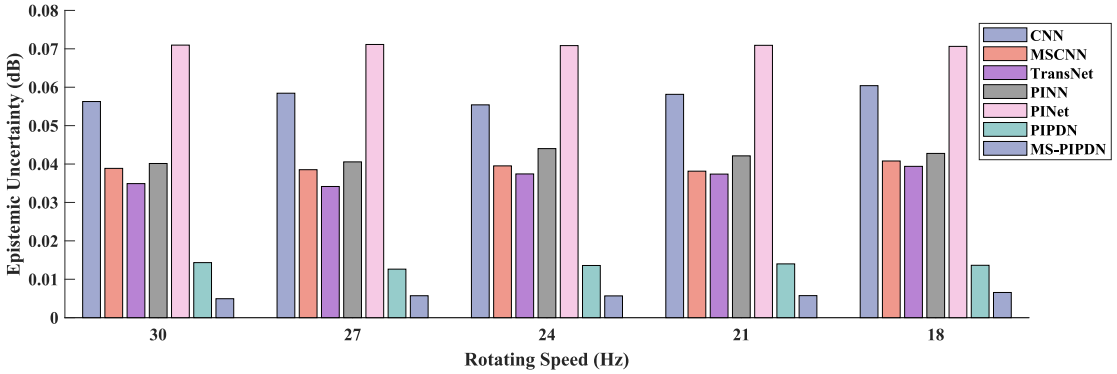


(f) F1 score

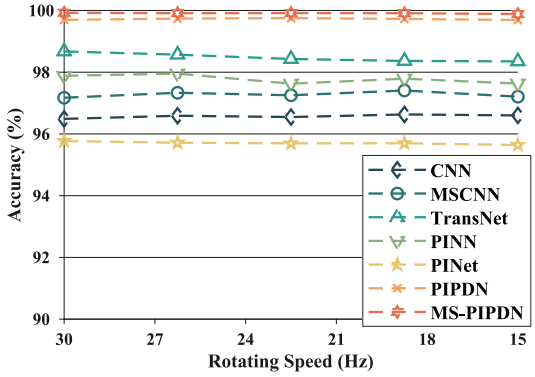
Fig. 17. Diagnostic performance under rotating speed increasing.



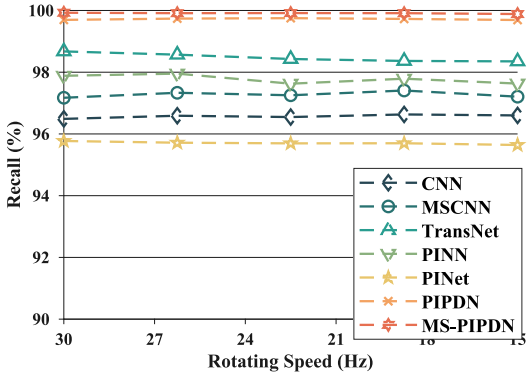
(a) raw vibration



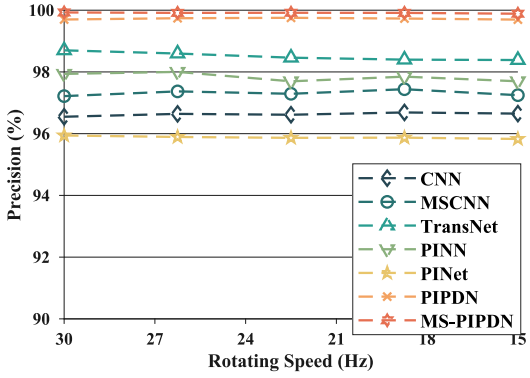
(b) epistemic uncertainty



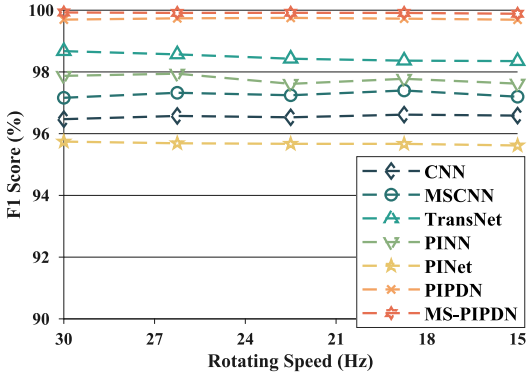
(c) Accuracy



(d) Recall

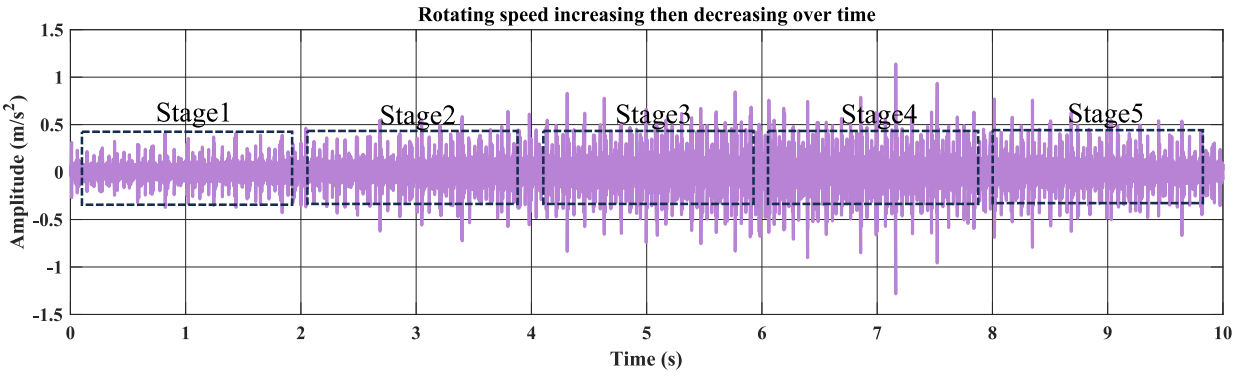


(e) Precision

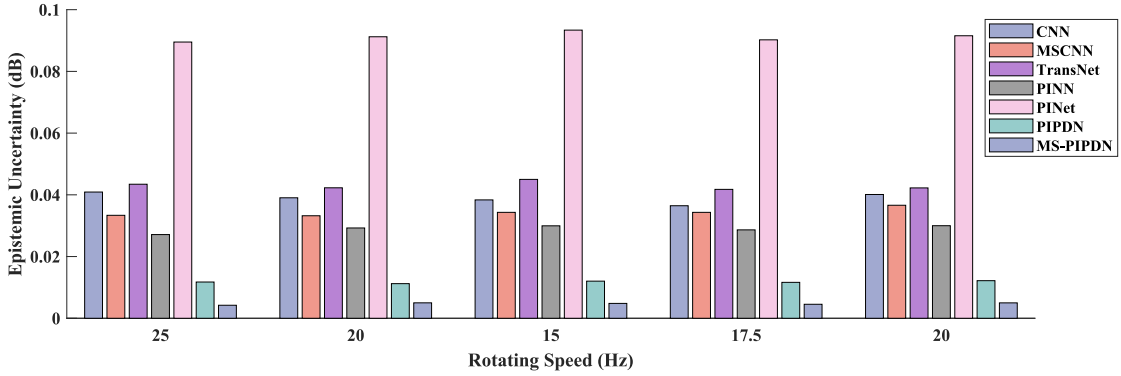


(f) F1 score

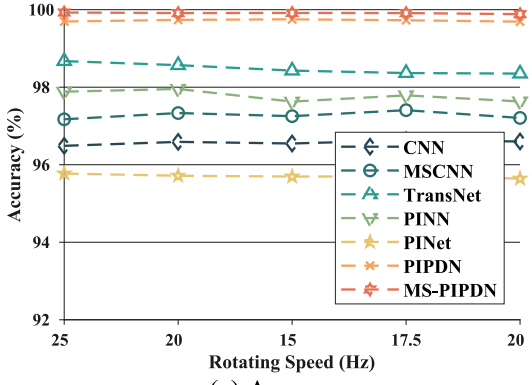
Fig. 18. Diagnostic performance under rotating speed increasing.



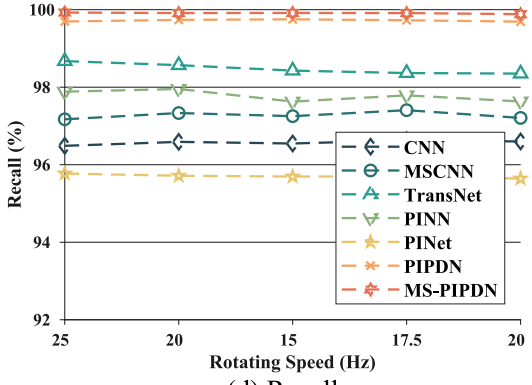
(a) raw vibration



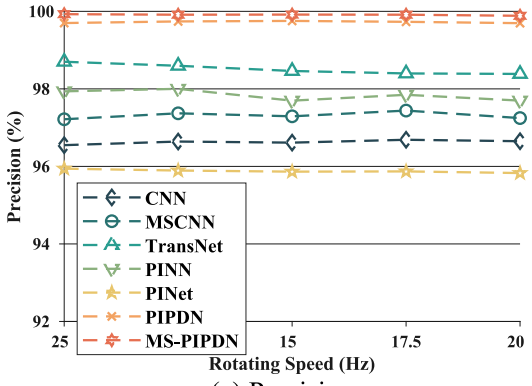
(b) epistemic uncertainty



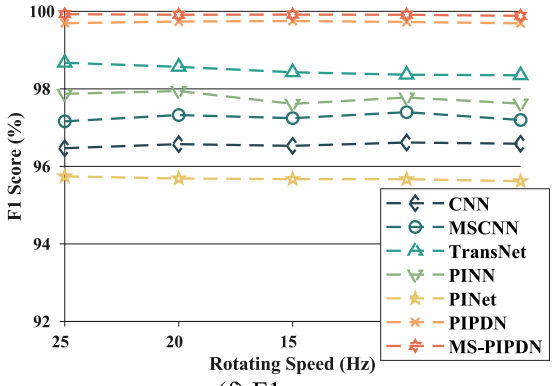
(c) Accuracy



(d) Recall



(e) Precision



(f) F1 score

Fig. 19. Diagnostic performance under rotating speed increasing.

enhancement, which guides the model to learn physical features through the proposed physical labelling module, rather than merely augmenting the features for the model to learn. Furthermore, MS-PIPDN builds upon PIPDN's performance, significantly improving diagnosis accuracy in inner race fault identification, increasing from 64 % to 71 %. This underscores the effectiveness of the UQ-based decision fusion module.

To validate the effectiveness and reliability of the proposed MS-PIPDN and PIPDN models in diagnosing faults in mechanical systems under varying rotating speed conditions, Figs. 17–19 evaluate their performance in Scenario IV.

Fig. 17(a) shows the vibration signals in scenario IV-A. The amplitude increases with the rotating speed. The diagnostic performance of the models is evaluated by dividing the 10-second raw signals into five stages. Fig. 17(b) depicts the epistemic uncertainty of various diagnostic models across different speed cases. It is evident that the uncertainty associated with PIPDN is consistently lower than that of any existing methods, highlighting the effectiveness of integrating a physical approach to enhance the model's understanding of fault diagnosis tasks. Furthermore, the epistemic uncertainty observed in MS-PIPDN is even lower than that in PIPDN. This improvement is facilitated by integrating a more comprehensive understanding of physical principles and implementing the proposed UQ-based decision-making module. This module prioritizes reasoning with significantly lower epistemic uncertainty within the MS-PIPDN model, resulting in improved final diagnostics. Additionally, the reliability of fault diagnosis for each model is validated through two additional scenarios. Fig. 18(a) and Fig. 19 illustrate the results of fault diagnosis in scenarios involving decreasing rotating speed and alternating between decreasing and then increasing rotating speed, respectively. Regardless of how the scenarios change, the proposed method (MS-PIPDN) consistently exhibits the lowest epistemic uncertainty and achieves the best fault diagnosis results across all four test metrics. This highlights the superiority and reliability of the proposed diagnostic approach.

5. Conclusion

In this study, a physics-informed probabilistic deep neural network (PIPDN) is proposed to develop an intelligent fault diagnosis model for mechanical systems. The PIPDN integrates a physical labelling module and an interpretable mechanism to enhance the model's trustworthiness and interpretability in diagnosing faults. This model generates smart data based on accurate fault information, which supports fault analysis and large-scale AI modelling in mechanical systems. Furthermore, a multi-scale physics-informed probabilistic deep neural network (MS-PIPDN) is developed by extending the PIPDN framework. This extension enhances the model's reliability in providing accurate fault diagnoses. To further improve diagnostic performance and reduce false positive alarms, uncertainty quantification (UQ) is incorporated into the MS-PIPDN model through a decision fusion module. The availability, reliability, and effectiveness of the proposed PIPDN and MS-PIPDN approaches are validated using a bearing experimental dataset. The diagnostic examination of these approaches indicates the following:

i. The proposed physical labelling module, combined with conditional labels, guides the PIPDN model to learn fault-representative features, making the model highly interpretable and trustworthy.
ii. The PIPDN model offers improved accuracy in diagnosing mechanical system (bearing) faults during the inference phase, generating smart data with more accurate fault characteristics compared to many state-of-the-art diagnostic models, which often provide such information at a later stage.
iii. The fault diagnosis mechanism of the PIPDN model is enhanced by incorporating UQ and smart data vibration analysis capabilities, revealing deeper insights into fault characteristics.
iv. The integration of the UQ-based decision fusion module significantly enhances the diagnostic performance and prediction reliability of the MS-PIPDN model compared to existing state-of-the-art models.

6. Future Work

In this study, the performance and reliability of neural networks in mechanical system fault diagnosis are improved by embedding physical information into the networks. This process involves using physical information to guide the neural network in learning more interpretable features. However, a limitation is that the computational cost of physical information labelling via encoder-decoder frameworks is relatively high, and subsequent efforts should focus on reducing this computational complexity. Future research will aim to further develop physics-informed neural network models for diagnostics by exploring various methods of embedding physical information to enhance the reliability and explainability of the models.

CRediT authorship contribution statement

Zifei Xu: Writing – original draft, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Kaicheng Zhao:** Visualization. **Jin Wang:** Supervision. **Musa Bashir:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

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

Data availability

No data was used for the research described in the article.

Acknowledgements

This research is part of the ULTIMATE project funded by the UKRI Marie Skłodowska-Curie Postdoctoral Fellowship [UKRI/EPSRC: EP/Y014235/1]. The authors would like to acknowledge the financial support from UKRI Innovate UK [grant number: TS/Y006364/1, TS/Y005236/1 and TS/X018407/1], and State Key Laboratory of Mechanical System and Vibration [China, No. MSV202411].

Appendix 1. PIPDN details based on Pytorch

PIPDN((ED): Physics_Informed_Encoder_Decoder((encoder4real): Sequential((0): Conv1d(1, 32, kernel_size=(5,), stride=(5,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): Conv1d(32, 64, kernel_size=(4,), stride=(4,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): Conv1d(64, 128, kernel_size=(3,), stride=(3,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(128, eps = 1e-

(continued on next page)

(*continued*)

05, momentum = 0.1, affine = True, track_running_stats = True) (12): Conv1d(128, 256, kernel_size=(2,), stride=(2,), bias = False) (13): Dropout(p = 0.5, inplace = False)) (encoder4imag): Sequential((0): Conv1d(1, 32, kernel_size=(5,), stride=(5,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): Conv1d(32, 64, kernel_size=(4,), stride=(4,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): Conv1d(64, 128, kernel_size=(3,), stride=(3,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): Conv1d(128, 256, kernel_size=(2,), stride=(2,), bias = False) (13): Dropout(p = 0.5, inplace = False)) (decoder4real): Sequential((0): ConvTranspose1d(256, 128, kernel_size=(2,), stride=(2,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): ConvTranspose1d(128, 64, kernel_size=(3,), stride=(3,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): ConvTranspose1d(64, 32, kernel_size=(4,), stride=(4,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): ConvTranspose1d(32, 1, kernel_size=(5,), stride=(5,), bias = False) (13): Dropout(p = 0.5, inplace = False) (14): ReLU() (15): Conv1d(1, 1, kernel_size=(1,), stride=(1,), bias = False)) (decoder4imag): Sequential((0): ConvTranspose1d(256, 128, kernel_size=(2,), stride=(2,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): ConvTranspose1d(128, 64, kernel_size=(3,), stride=(3,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): ConvTranspose1d(64, 32, kernel_size=(4,), stride=(4,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): ConvTranspose1d(32, 1, kernel_size=(5,), stride=(5,), bias = False) (13): Dropout(p = 0.5, inplace = False) (14): ReLU() (15): Conv1d(1, 1, kernel_size=(1,), stride=(1,), bias = False)) (classifier4u): Sequential((0): Linear(in_features = 8960, out_features = 420, bias = True) (1): ReLU() (2): Linear(in_features = 420, out_features = 4, bias = True))))

Appendix 2. MS-PIPDN details based on Pytorch

MS-PIPDN((MS_1): AvgPool1d(kernel_size=(1,), stride=(1,), padding=(0,))(MS_2): AvgPool1d(kernel_size=(3,), stride=(1,), padding=(1,))(MS_3): AvgPool1d(kernel_size=(5,), stride=(1,), padding=(2,))(ED_1): Physics_Informed_Encoder_Decoder((encoder4real): Sequential((0): Conv1d(1, 32, kernel_size=(5,), stride=(5,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): Conv1d(32, 64, kernel_size=(4,), stride=(4,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): Conv1d(64, 128, kernel_size=(3,), stride=(3,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): Conv1d(128, 256, kernel_size=(2,), stride=(2,), bias = False) (13): Dropout(p = 0.5, inplace = False)) (encoder4imag): Sequential((0): Conv1d(1, 32, kernel_size=(5,), stride=(5,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): Conv1d(32, 64, kernel_size=(4,), stride=(4,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): Conv1d(64, 128, kernel_size=(3,), stride=(3,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): Conv1d(128, 256, kernel_size=(2,), stride=(2,), bias = False) (13): Dropout(p = 0.5, inplace = False)) (decoder4real): Sequential((0): ConvTranspose1d(256, 128, kernel_size=(2,), stride=(2,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): ConvTranspose1d(128, 64, kernel_size=(3,), stride=(3,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): ConvTranspose1d(64, 32, kernel_size=(4,), stride=(4,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): ConvTranspose1d(32, 1, kernel_size=(5,), stride=(5,), bias = False) (13): Dropout(p = 0.5, inplace = False) (14): ReLU() (15): Conv1d(1, 1, kernel_size=(1,), stride=(1,), bias = False)) (decoder4imag): Sequential((0): ConvTranspose1d(256, 128, kernel_size=(2,), stride=(2,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): ConvTranspose1d(128, 64, kernel_size=(3,), stride=(3,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): ConvTranspose1d(64, 32, kernel_size=(4,), stride=(4,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): ConvTranspose1d(32, 1, kernel_size=(5,), stride=(5,), bias = False) (13): Dropout(p = 0.5, inplace = False) (14): ReLU() (15): Conv1d(1, 1, kernel_size=(1,), stride=(1,), bias = False)) (classifier4u): Sequential((0): Linear(in_features = 8960, out_features = 420, bias = True) (1): ReLU() (2): Linear(in_features = 420, out_features = 4, bias = True))) (ED_2): Physics_Informed_Encoder_Decoder((encoder4real): Sequential((0): Conv1d(1, 32, kernel_size=(5,), stride=(5,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): Conv1d(32, 64, kernel_size=(4,), stride=(4,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): Conv1d(64, 128, kernel_size=(3,), stride=(3,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): Conv1d(128, 256, kernel_size=(2,), stride=(2,), bias = False) (13): Dropout(p = 0.5, inplace = False)) (encoder4imag):

(*continued on next page*)

(*continued*)

Sequential((0): Conv1d(1, 32, kernel_size=(5,), stride=(5,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): Conv1d(32, 64, kernel_size=(4,), stride=(4,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): Conv1d(64, 128, kernel_size=(3,), stride=(3,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): Conv1d(128, 256, kernel_size=(2,), stride=(2,), bias = False) (13): Dropout(p = 0.5, inplace = False)) (decoder4real): Sequential((0): ConvTranspose1d(256, 128, kernel_size=(2,), stride=(2,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): ConvTranspose1d(128, 64, kernel_size=(3,), stride=(3,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): ConvTranspose1d(64, 32, kernel_size=(4,), stride=(4,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): ConvTranspose1d(32, 1, kernel_size=(5,), stride=(5,), bias = False) (13): Dropout(p = 0.5, inplace = False) (14): ReLU() (15): Conv1d(1, 1, kernel_size=(1,), stride=(1,), bias = False)) (decoder4imag): Sequential((0): ConvTranspose1d(256, 128, kernel_size=(2,), stride=(2,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): ConvTranspose1d(128, 64, kernel_size=(3,), stride=(3,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): ConvTranspose1d(64, 32, kernel_size=(4,), stride=(4,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): ConvTranspose1d(32, 1, kernel_size=(5,), stride=(5,), bias = False) (13): Dropout(p = 0.5, inplace = False) (14): ReLU() (15): Conv1d(1, 1, kernel_size=(1,), stride=(1,), bias = False)) (classifier4u): Sequential((0): Linear(in_features = 8960, out_features = 420, bias = True) (1): ReLU() (2): Linear (in_features = 420, out_features = 4, bias = True))) (ED_3): Physics_Informed_Encoder_Decoder((encoder4real): Sequential((0): Conv1d(1, 32, kernel_size=(5,), stride=(5,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): Conv1d(32, 64, kernel_size=(4,), stride=(4,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): Conv1d(64, 128, kernel_size=(3,), stride=(3,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): Conv1d(128, 256, kernel_size=(2,), stride=(2,), bias = False) (13): Dropout(p = 0.5, inplace = False)) (encoder4imag): Sequential((0): Conv1d(1, 32, kernel_size=(5,), stride=(5,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): Conv1d(32, 64, kernel_size=(4,), stride=(4,), bias = False) (5): Dropout (p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): Conv1d(64, 128, kernel_size=(3,), stride=(3,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): Conv1d(128, 256, kernel_size=(2,), stride=(2,), bias = False) (13): Dropout (p = 0.5, inplace = False)) (decoder4real): Sequential((0): ConvTranspose1d(256, 128, kernel_size=(2,), stride=(2,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): ConvTranspose1d(128, 64, kernel_size=(3,), stride=(3,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): ConvTranspose1d(64, 32, kernel_size= (4,), stride=(4,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): ConvTranspose1d(32, 1, kernel_size=(5,), stride=(5,), bias = False) (13): Dropout(p = 0.5, inplace = False) (14): ReLU() (15): Conv1d(1, 1, kernel_size=(1,), stride=(1,), bias = False)) (decoder4imag): Sequential((0): ConvTranspose1d (256, 128, kernel_size=(2,), stride=(2,), bias = False) (1): Dropout(p = 0.5, inplace = False) (2): ReLU() (3): BatchNorm1d(128, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (4): ConvTranspose1d(128, 64, kernel_size=(3,), stride=(3,), bias = False) (5): Dropout(p = 0.5, inplace = False) (6): ReLU() (7): BatchNorm1d(64, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (8): ConvTranspose1d(64, 32, kernel_size=(4,), stride=(4,), bias = False) (9): Dropout(p = 0.5, inplace = False) (10): ReLU() (11): BatchNorm1d(32, eps = 1e-05, momentum = 0.1, affine = True, track_running_stats = True) (12): ConvTranspose1d(32, 1, kernel_size=(5,), stride=(5,), bias = False) (13): Dropout(p = 0.5, inplace = False) (14): ReLU() (15): Conv1d(1, 1, kernel_size=(1,), stride=(1,), bias = False)) (classifier4u): Sequential((0): Linear(in_features = 8960, out_features = 420, bias = True) (1): ReLU() (2): Linear (in_features = 420, out_features = 4, bias = True)))))

References

[1] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U.R. Acharya, V. Makarenkov, S. Nahavandi, A review of uncertainty quantification in deep learning: Techniques, applications and challenges, Inform. Fusion 76 (2021) 243–297, https://doi.org/10.1016/j.inffus.2021.05.008.

[2] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, M. Gabbouj, D.J. Inman, A review of vibration-based damage detection in civil structures: From traditional methods to machine learning and deep learning applications, Mech. Syst. Sig. Process. 147 (2021) 107077.

[3] M. Bashir, Z. Xu, J. Wang, C. Guedes Soares, Data-driven damage quantification of floating offshore wind turbine platforms based on multi-scale encoder–decoder with self-attention mechanism, J. Marine Sci. Eng. 10 (12) (2022) 1830, https://doi.org/10.3390/jmse10121830.

[4] L. Cao, H. Zhang, Z. Meng, X. Wang, A parallel GRU with dual-stage attention mechanism model integrating uncertainty quantification for probabilistic RUL prediction of wind turbine bearings, Reliabil. Eng. Syst. Safety 235 (2023), https://doi.org/10.1016/j.ress.2023.109197.

[5] Y. Cao, Y. Ding, M. Jia, R. Tian, A novel temporal convolutional network with residual self-attention mechanism for remaining useful life prediction of rolling bearings, Reliab. Eng. Syst. Saf. 215 (2021) 107813, https://doi.org/10.1016/j.ress.2021.107813.

[6] Y. Chen, M. Rao, K. Feng, M.J. Zuo, Physics-Informed LSTM hyperparameters selection for gearbox fault detection, Mech. Syst. Signal Proc. 171 (2022), https://doi.org/10.1016/j.ymssp.2022.108907.

[7] Y. Ding, M. Jia, Q. Miao, Y. Cao, A novel time–frequency Transformer based on self–attention mechanism and its application in fault diagnosis of rolling bearings, *Mech. Syst. Signal Proc.* 168 (2022) 108616, https://doi.org/10.1016/j.ymssp.2021.108616.

[8] K. Feng, J.C. Ji, Y. Zhang, Q. Ni, Z. Liu, M. Beer, Digital twin-driven intelligent assessment of gear surface degradation, Mech. Syst. Sig. Process. 186 (2023), https://doi.org/10.1016/j.ymssp.2022.109896.
[9] D. García-Gil, J. Luengo, S. García, F. Herrera, Enabling smart data: noise filtering in big data classification, Inf. Sci. 479 (2019), https://doi.org/10.1016/j.ins.2018.12.002.
[10] L. Guo, N. Li, F. Jia, Y. Lei, J. Lin, A recurrent neural network based health indicator for remaining useful life prediction of bearings, Neurocomputing 240 (2017) 98–109.
[11] T. Han, Y.F. Li, Out-of-distribution detection-assisted trustworthy machinery fault diagnosis approach with uncertainty-aware deep ensembles, Reliab. Eng. Syst. Saf. 226 (March) (2022) 108648, https://doi.org/10.1016/j.ress.2022.108648.
[12] W. Huang, J. Cheng, Y. Yang, G. Guo, An improved deep convolutional neural network with multi-scale information for bearing fault diagnosis, Neurocomputing 359 (2019) 77–92, https://doi.org/10.1016/j.neucom.2019.05.052.
[13] E. Hüllermeier, W. Waegeman, Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods, Mach. Learn. Springer, US 110 (3) (2021), https://doi.org/10.1007/s10994-021-05946-3.
[14] N. Huu, N. Hoang, K. Ngan, N. Viet, Computers in industry trans-lighter : a light-weight federated learning-based architecture for remaining useful lifetime prediction, Comput. Indust. 148 (2023), https://doi.org/10.1016/j.compind.2023.103888.
[15] W.N. Ismail, M.M. Hassan, H.A. Alsalamah, G. Fortino, CNN-based health model for regular health factors analysis in internet-of-medical things environment, IEEE Access 8 (2020) 52541–52549, https://doi.org/10.1109/ACCESS.2020.2980938.
[16] M.A. Jan, W. Zhang, F. Khan, S. Abbas, R. Khan, Lightweight and smart data fusion approaches for wearable devices of the Internet of Medical Things, Inform. Fusion 103 (2024), https://doi.org/10.1016/j.inffus.2023.102076.
[17] G. Jiang, H. He, J. Yan, P. Xie, Multiscale convolutional neural networks for fault diagnosis of wind turbine gearbox, IEEE Trans. Ind. Electron. 66 (4) (2019) 3196–3207, https://doi.org/10.1109/TIE.2018.2844805.
[18] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, J. Lin, Machinery health prognostics: A systematic review from data acquisition to RUL prediction, Mech. Syst. Sig. Process. 104 (2018) 799–834, https://doi.org/10.1016/j.ymssp.2017.11.016.
[19] Y. Lei, B. Yang, X. Jiang, F. Jia, N. Li, A.K. Nandi, Applications of machine learning to machine fault diagnosis: A review and roadmap, Mech. Syst. Sig. Process. 138 (2020) 106587, https://doi.org/10.1016/j.ymssp.2019.106587.
[20] G. Li, L. Yang, C.G. Lee, X. Wang, M. Rong, A bayesian deep learning RUL framework integrating epistemic and aleatoric uncertainties, IEEE Trans. Ind. Electron. 68 (9) (2021) 8829–8841, https://doi.org/10.1109/TIE.2020.3009593.
[21] T. Li, Z. Zhao, C. Sun, L. Cheng, X. Chen, R. Yan, R.X. Gao, WaveletKernelNet: an interpretable deep neural network for industrial intelligent diagnosis, IEEE Trans. Syst., Man, and Cybernetics: Systems 52 (4) (2022) 2302–2312, https://doi.org/10.1109/TSMC.2020.3048950.
[22] Z. Li, X. Ding, Z. Song, L. Wang, B. Qin, W. Huang, Digital twin-assisted dual transfer: A novel information-model adaptation method for rolling bearing fault diagnosis, Inform. Fusion 106 (2024), https://doi.org/10.1016/j.inffus.2024.102271.
[23] D. Liu, B. Chen, J. An, C. Li, G. Liu, J. Shao, W. Tang, C. Zhang, Z.L. Wang, Wind-driven self-powered wireless environmental sensors for Internet of Things at long distance, Nano Energy 73 (2020), https://doi.org/10.1016/j.nanoen.2020.104819.
[24] H. Liu, L. Li, J. Ma, Rolling bearing fault diagnosis based on STFT-deep learning and sound signals, Shock Vib. 2016 (2016), https://doi.org/10.1155/2016/6127479.
[25] R. Malekimoghadam, S. Krause, S. Czichon, in: M. Abdel Wahab (Ed.), A Critical Review on the Structural Health Monitoring Methods of the Composite Wind Turbine Blades, Proceedings of 1st International Conference on Structural Damage Modelling and Assessment, Springer Singapore, 2021, pp. 409–438.
[26] Q. Ni, J.C. Ji, K. Feng, Y. Zhang, D. Lin, J. Zheng, Data-driven bearing health management using a novel multi-scale fused feature and gated recurrent unit, Reliab. Eng. Syst. Saf. 242 (2024), https://doi.org/10.1016/j.ress.2023.109753.
[27] Q. Ni, J.C. Ji, B. Halkon, K. Feng, A.K. Nandi, Physics-Informed Residual Network (PIResNet) for rolling element bearing fault diagnostics, Mech. Syst. Sig. Process. 200 (2023), https://doi.org/10.1016/j.ymssp.2023.110544.
[28] W. Peng, Z.S. Ye, N. Chen, Bayesian deep-learning-based health prognostics toward prognostics uncertainty, IEEE Trans. Ind. Electron. 67 (3) (2020) 2283–2293, https://doi.org/10.1109/TIE.2019.2907440.
[29] M. Sadoughi, C. Hu, Physics-based convolutional neural network for fault diagnosis of rolling element bearings, IEEE Sens. J. 19 (11) (2019) 4181–4192, https://doi.org/10.1109/JSEN.2019.2898634.
[30] M. Sehri, P. Dumond, M. Bouchard, University of Ottawa constant load and speed rolling-element bearing vibration and acoustic fault signature datasets, Data Brief 49 (2023), https://doi.org/10.1016/j.dib.2023.109327.
[31] S. Shen, H. Lu, M. Sadoughi, C. Hu, V. Nemani, A. Thelen, K. Webster, M. Darr, J. Sidon, S. Kenny, A physics-informed deep learning approach for bearing fault detection, Eng. Appl. Artif. Intel. 103 (2021), https://doi.org/10.1016/j.engappai.2021.104295.
[32] S. Tang, H. Tang, M. Chen, Transfer-learning based gas path analysis method for gas turbines, Appl. Therm. Eng. 155 (2019) 1–13.
[33] P. Vrignat, F. Kratz, M. Avila, Sustainable manufacturing, maintenance policies, prognostics and health management: A literature review, Reliabil. Eng. Syst. Safety 218 (2022), https://doi.org/10.1016/j.ress.2021.108140.
[34] B. Wang, Y. Lei, N. Li, N. Li, A hybrid prognostics approach for estimating remaining useful life of rolling element bearings, IEEE Trans. Reliab. 69 (1) (2018) 401–412.
[35] C. Wu, P. Jiang, C. Ding, F. Feng, T. Chen, Intelligent fault diagnosis of rotating machinery based on one-dimensional convolutional neural network, Comput. Ind. 108 (2019), https://doi.org/10.1016/j.compind.2018.12.001.
[36] Y. Xiao, H. Shao, M. Feng, T. Han, J. Wan, B. Liu, Towards trustworthy rotating machinery fault diagnosis via attention uncertainty in transformer, J. Manuf. Syst. 70 (2023) 186–201, https://doi.org/10.1016/j.jmsy.2023.07.012.
[37] Z. Xu, M. Bashir, Q. Liu, Z. Miao, X. Wang, J. Wang, N. Ekere, A novel health indicator for intelligent prediction of rolling bearing remaining useful life based on unsupervised learning model, Comput. Ind. Eng. 176 (2023), https://doi.org/10.1016/j.cie.2023.108999.
[38] Z. Xu, M. Bashir, W. Zhang, Y. Yang, X. Wang, C. Li, An intelligent fault diagnosis for machine maintenance using weighted soft-voting rule based multi-attention module with multi-scale information fusion, Inform. Fusion 86–87 (2022), https://doi.org/10.1016/j.inffus.2022.06.005.
[39] Z. Xu, C. Li, Y. Yang, Fault diagnosis of rolling bearing of wind turbines based on the Variational Mode Decomposition and Deep Convolutional Neural Networks, Appl. Soft Comput. J. 95 (2020), https://doi.org/10.1016/j.asoc.2020.106515.
[40] Z. Xu, C. Li, Y. Yang, Fault diagnosis of rolling bearings using an Improved multi-scale convolutional neural network with feature attention mechanism, ISA Trans. 110 (2021), https://doi.org/10.1016/j.isatra.2020.10.054.
[41] S.A. Zargar, F.-G. Yuan, Physics-informed deep learning for scattered full wavefield reconstruction from a sparse set of sensor data for impact diagnosis in structural health monitoring, Struct. Health Monit. (2024) 14759217231202548, https://doi.org/10.1177/14759217231202547.
[42] T. Zhou, T. Han, E.L. Droguett, Towards trustworthy machine fault diagnosis: A probabilistic Bayesian deep learning framework, Reliab. Eng. Syst. Saf. 224 (2022) 108525, https://doi.org/10.1016/j.ress.2022.108525.
[43] R. Zhu, W. Peng, D. Wang, C.G. Huang, Bayesian transfer learning with active querying for intelligent cross-machine fault prognosis under limited data, Mech. Syst. Signal Proc. 183 (2022), https://doi.org/10.1016/j.ymssp.2022.109628.