mathematics

MDPI

Article

Robust-MBDL: A Robust Multi-Branch Deep-Learning-Based Model for Remaining Useful Life Prediction of Rotating Machines

Khoa Tran [1], Hai-Canh Vu [2,3,*], Lam Pham [4], Nassim Boudaoud [5] and Ho-Si-Hung Nguyen [6]

[1] AIWARE Limited Company, Da Nang City 550000, Vietnam; khoa.tran@aiware.website
[2] Laboratory for Applied and Industrial Mathematics, Institute for Computational Science and Artificial Intelligence, Van Lang University, Ho Chi Minh City 70000, Vietnam
[3] Faculty of Mechanical-Electrical and Computer Engineering, School of Technology, Van Lang University, Ho Chi Minh City 70000, Vietnam
[4] AIT Austrian Institute of Technology GmbH, 1020 Vienna, Austria; lam.pham@ait.ac.at
[5] Roberval Laboratory, Department of Mechanical Engineering, University of Technology of Compiègne, 60200 Compiègne, France; nassim.boudaoud@utc.fr
[6] Faculty of Electrical Engineering, University of Science and Technology—The University of Danang, Da Nang City 550000, Vietnam; nhshung@dut.udn.vn
* Correspondence: canh.vuhai@vlu.edu.vn

Abstract: Predictive maintenance (PdM) is one of the most powerful maintenance techniques based on the estimation of the remaining useful life (RUL) of machines. Accurately estimating the RUL is crucial to ensure the effectiveness of PdM. However, current methods have limitations in fully exploring condition monitoring data, particularly vibration signals, for RUL estimation. To address these challenges, this research presents a novel Robust Multi-Branch Deep Learning (Robust-MBDL) model. Robust-MBDL stands out by leveraging diverse data sources, including raw vibration signals, time–frequency representations, and multiple feature domains. To achieve this, it adopts a specialized three-branch architecture inspired by efficient network designs. The model seamlessly integrates information from these branches using an advanced attention-based Bi-LSTM network. Furthermore, recognizing the importance of data quality, Robust-MBDL incorporates an unsupervised LSTM-Autoencoder for noise reduction in raw vibration data. This comprehensive approach not only overcomes the limitations of existing methods but also leads to superior performance. Experimental evaluations on benchmark datasets such as XJTU-SY and PRONOSTIA showcase Robust-MBDL's efficacy, particularly in rotating machine health prognostics. These results underscore its potential for real-world applications, heralding a new era in predictive maintenance practices.

Keywords: remaining useful life; industrial prognostics; rotating machines; deep learning; multimodal neural network

MSC: 68T20

Citation: Tran, K.; Vu, H.-C.; Pham, L.; Boudaoud, N.; Nguyen, H.-S.-H. Robust-MBDL: A Robust Multi-Branch Deep-Learning-Based Model for Remaining Useful Life Prediction of Rotating Machines. *Mathematics* **2024**, *12*, 1569. https://doi.org/10.3390/math12101569

Academic Editor: Faheim Sufi

Received: 19 March 2024
Revised: 3 May 2024
Accepted: 16 May 2024
Published: 17 May 2024

1. Introduction

Accurately estimating the remaining useful life (RUL) plays a pivotal role in predictive maintenance for rotating machines. The prediction of RUL has garnered significant attention from both academic researchers and industry professionals. This is because accurately predicting RUL can significantly enhance the effectiveness of predictive maintenance, leading to increased machine reliability and reduced incidences of failures and associated repair costs.

Existing RUL prediction models generally fall within two primary categories: the model-based [1,2] and the data-driven approaches [3,4]. The model-based approach relies on a certain level of physical knowledge about machine degradation to predict RUL, such as

employing theories of the Paris law for bearing defect growth [5] and reliability laws [6–8]. However, integrating such physical knowledge into models can be challenging, especially concerning complex machinery where such insights might not always be readily available.

The advent of Industrial Internet of Things (IIoT) technologies has facilitated the accumulation of extensive data (evidenced by benchmark datasets for RUL detection, e.g., [9,10]). This influx of data has significantly bolstered the application of the data-driven approach for RUL detection. Unlike model-based methods, the data-driven approach primarily relies on collected data, enabling its application to complex machines/systems without a prerequisite for extensive physical knowledge.

Machine Learning (ML) is a popular data-driven approach that has been extensively used in predicting the RUL of rotating machines. Several studies, including [11–14], have employed well-known ML models such as Linear Regression (LR), Random Forest (RF), and Support Vector Machines (SVM) to forecast RUL. However, these methods have some significant drawbacks, such as suboptimal performance due to inflexible mathematical formulas and time-consuming computations for big input data. Therefore, there has been a significant shift towards Deep Learning (DL) [15,16] in preference to traditional ML techniques.

In the literature, several DL models have been proposed to estimate the RUL of rotating machines. These models often consist of simple neural network architectures, including a series of several Long Short-Term Memory (LSTM) or convolutional layers [3,17,18]. The convolutional layers are directed towards the identification of spatial dependencies within time series data, while the LSTM layers excel at identifying and capturing long-term correlations. These models provide better results compared to ML models; however, their performance is still limited due to their simple architecture, which does not allow for the deep extraction of different vibration features. The aim of this work is to develop a deep learning model with an advanced architecture that can efficiently extract vibration features and provide a more accurate estimation of the remaining useful life (RUL) of rotating machinery.

2. Related Work and Contributions

Recent advancements in deep learning have shown great potential in accurately predicting remaining useful life (RUL). One effective technique is to combine multiple deep learning networks, such as the convolutional neural network (CNN), LSTM, Autoencoder, and Transformer, to leverage their individual strengths. Hinchi and Tkiouat [19] proposed a deep learning framework based on convolutional and LSTM units. The model extracts local features directly from sensor data using a convolutional layer, captures the degradation process using an LSTM layer, and finally estimates the RUL using the LSTM outputs. The performance of the model was evaluated on the PRONOSTIA dataset. Zhu et al. [20] used the wavelet transform to obtain a time–frequency representation (2D feature) instead of using raw vibration data. They proposed a Multiscale CNN model that simultaneously maintains global and local information, in contrast to a traditional CNN. The model showed superior performance compared to the traditional CNN when tested on the PRONOSTIA dataset. Mo et al. [21] proposed a combination of the Transformer encoder and gated convolutional unit (GCU). While the Transformer encoder captures short-term and long-term dependencies in time series, the GCU facilitates the consideration of local contexts at each time step. The experimental study confirmed that the proposed model outperforms the CNN, LSTM, and Auto-encoder models. In [18], the raw data are first denoised using a Bi-LSTM autoencoder before being passed into the Transformer encoder for RUL prediction. The denoising step's effectiveness was confirmed through various experiments. Wei et al. [22] presented a new approach called the self-adaptive graph convolutional network (SAGCN) that incorporates a self-attention mechanism to capture the correlation of features at different time points without using recurrent characteristics. The proposed method was evaluated on the XJTU-SY dataset, and the results demonstrated its superiority over existing data-driven methods, such as graph CNN, CNN-LSTM, CNN, and Generative

Adversarial Network (GAN). Zheng et al. [23] utilized an autoencoder to extract crucial features and minimize noise. They estimated the RUL using a deep reinforcement learning model based on the Twin Delayed Deep Deterministic Policy Gradient algorithm (TD3). The researchers proved that their model outperformed the LSTM and CNN models on the XJTU-SY dataset. Xu et al. [24] and Al-Dulaimi et al. [25] introduced effective models for the RUL estimation. These models include two parallel branches: one based on LSTM and the other based on CNN. The outputs of these two branches are then combined by a fully connected (FC) multilayer. Using parallel architecture, features are independently extracted by two branches to maximize CNN and LSTM advantages. Experiment studies have shown that these models outperform other existing models, such as deep CNN, deep LSTM, and the multiobjective deep belief network, on both the PRONOSTIA and XJTU datasets. Huang et al. [26] highlighted the significant role of operational data in predicting RUL and proposed an architecture consisting of two parallel Bi-LSTM networks to extract features from both raw and operational data. In 2021, Huang et al. [27] proposed a novel parallel architecture consisting of a deep convolutional neural network (CNN) and a multilayer perceptron (MLP). This architecture aims to extract informative representations simultaneously from both raw data and 2D features. The model's performance on the XJTU-SY and PRONOSTIA datasets surpasses that of MLP, Bi-LSTM, and Multiscale-CNN. Recently, Cheng et al. [28] developed a novel parallel model with two branches: Bi-LSTM and Bi-directional Gated Recurrent Unit (Bi-GRU) to extract different 1D features (frequency domain and time domain) and 2D features (time–frequency domain). The model outperforms the Bi-LSTM, Bi-GRU, RNN, stacked denoising auto-encoder (SDAE), extreme learning machine (ELM), and MLP on the XJTU-SY dataset.

Table 1 presents a summary of related work. Despite the numerous advantages, the existing DL models have the following drawbacks:

- Most works only exploit a limited set of features or data, particularly raw data. This can result in a loss of information and reduced performance of the models.
- The existing models typically have a maximum of two branches. These branches consist of several layers of either CNN or LSTM. They lack the depth required to efficiently extract complex vibration data.
- The linear fully connected layer (FC) used to fuse the different data branches is not flexible or efficient enough for RUL prediction.
- The performance of current multi-branch models is adversely affected by noise and anomaly data, which makes them less robust. Noise filtering is usually necessary to ensure the models' robustness [29].

Table 1. Summary of the related work.

Papers	Input Data	Branch	Branch Architecture	Fusion Number	Noise Filtering
[19]	Raw data	1	CNN + LSTM	No	No
[20]	2D feature	1	Multiscale CNN	No	No
[21]	Raw data	1	Transformer + GCU	No	No
[18]	Raw data	1	Transformer encoder	No	Yes
[22]	Raw data	1	Graph CNN + Self-attention	No	No
[23]	Raw data	1	Reinforcement learning	No	Yes
[24]	Raw data	2	LSTM + CNN	FC	No
[25]	Raw data	2	LSTM + CNN	FC	No
[26]	Raw data + Operational data	2	Bi-LSTM + Bi-LSTM	FC	No
[27]	Raw data + 2D feature	2	DCNN + MLP	FC	No
[28]	1D feature + 2D feature	2	Bi-LSTM + Bi-GRU	FC	No

To address the above limitations, this study proposes a Robust Multi-Branch Deep Learning (Robust-MBDL) model. This model comprises three branches specifically designed to predict the RUL based on vibration data from rotating machinery. The Robust-MBDL model has the following advantages:

1. Maximum Information exploitation: Multiple types of input data are considered, including raw vibration signals, 11 time-domain features, three frequency-domain features (1D data), and time–frequency representation (TFR) features generated by Wavelet transformation (2D data). The use of raw vibration data, along with their features, and operational condition information, improves the learning capacity of our model while preserving important information.
2. Specialized Architecture consisting of three DL branches: Efficiently extracting various types of features requires different network architectures. This paper introduces the Robust-MBDL model, employing an advanced architecture consisting of three distinct branches: a 1D data branch, a 2D data branch, and a raw data branch. These branch architectures are largely adapted from the lightweight ResNet-34 architecture [30] and the convolutional building block (CBB) [31]. They use skip connections to facilitate learning, enabling the creation of complex models with many blocks, and improving the ability to learn from complex vibration data.
3. Branches' fusion via attention-based Bi-LSTM (AB-LSTM): By leveraging the outputs of three data branches, the AB-LSTM fusion network focuses on significant features and considers past and future information to provide an accurate prediction of RUL.
4. Noise Reduction using LSTM-Autoencoder: An unsupervised noise filter was developed based on the LSTM-Autoencoder architecture to reduce noise, remove abnormal data from raw vibration signals, and thus enhance the model's robustness [32,33].

The rest of this paper is organized as follows: Section 3 represents the high-level architecture of our proposed Robust-MBDL model. We then comprehensively present all the main components of our proposed model in Sections 4–7. Sections 8 and 9 show our experimental setting and results. Finally, some conclusions drawn from this work are presented in Section 10.

3. The High-Level Architecture of the Robust-MBDL Model

The architecture of our proposed Robust-MBDL consists of four primary components, as shown in Figure 1.

- Noise filtering using LSTM-Autoencoder;
- Feature extraction;
- Health Indicators (HI) construction;
- Multi-branch deep learning (MBDL) network.

The process starts with data denoising and abnormal data clearing through an LSTM-Autoencoder-based filter. The denoised data are then used to extract the different features and also to construct the HI. Given the denoised data, 14 distinct 1D features (e.g., root mean square and variance) and a 2D feature are extracted (i.e., the 2D feature is the spectrogram obtained via the wavelet transform). The MBDL network is composed of three separate branches that extract information from denoised data, 1D features, and 2D features. Two blocks, AB-LSTM and GAP, follow each branch to proficiently handle the OC identification and RUL prediction simultaneously.

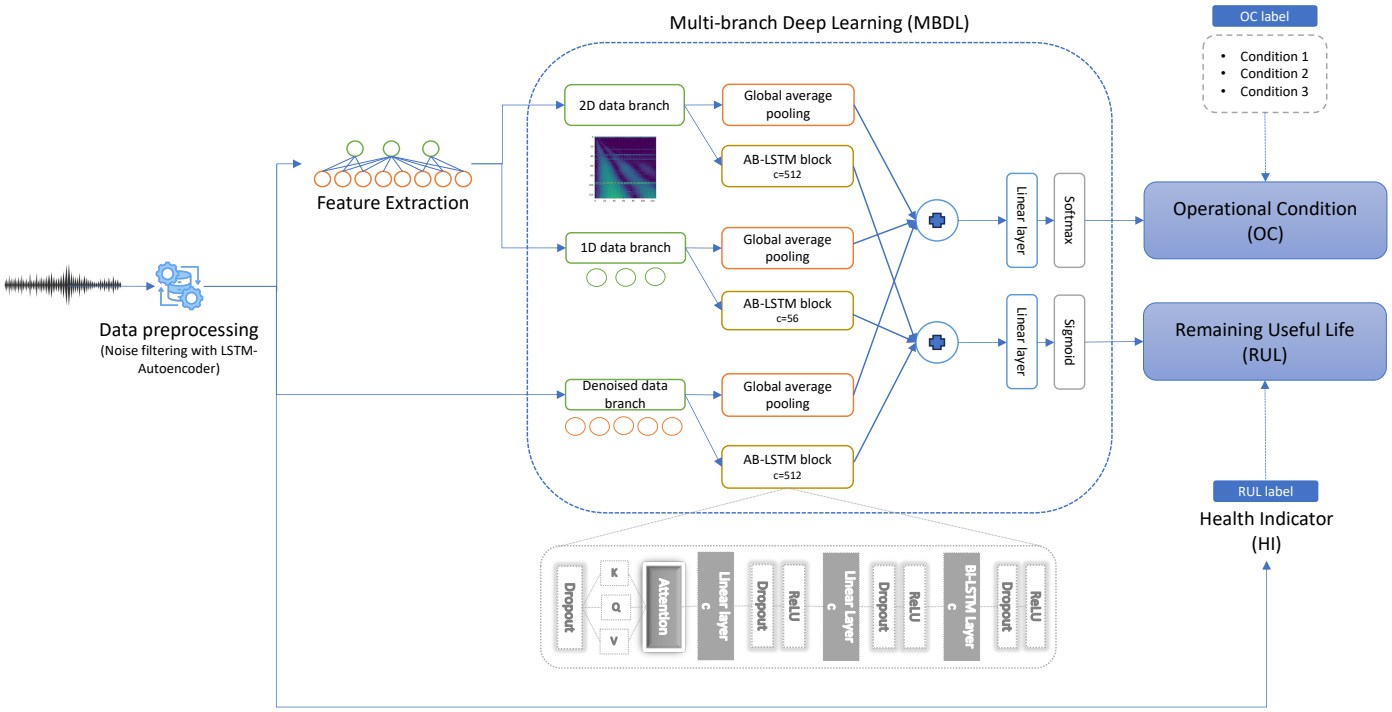

Figure 1. The high-level architecture of our proposed Robust-MBDL model.

4. Noise Filtering Using LSTM-Autoencoder

LSTM, a specialized form of RNN, effectively handles short- and long-term dependencies in time series predictions by maintaining memory across numerous time steps. Unlike traditional RNN, LSTM circumvents the vanishing gradient problem during training [34]. It employs input, forget, and output gates to manage information flow, enabling the retention of pertinent data and discarding unnecessary information. These mechanisms significantly enhance the accuracy of time series predictions. The core of an LSTM cell involves several gates regulating information flow: the input gate controls what enters the cell, the forget gate manages what is removed from memory, and the cell state is updated by balancing incoming and outgoing information, influencing the output and hidden state. Based on these reasons, LSTM is applied in the proposed LSTM-Autoencoder model.

An autoencoder is an artificial neural network widely used for learning the hidden patterns of unlabeled data. An autoencoder contains two parts: an encoder and a decoder. The encoder maps the input data to hidden patterns and the decoder tries to reconstruct the output from the hidden patterns. The autoencoder is trained to minimize the difference between the input and the reconstructed output. The autoencoder has been successfully applied to different problems such as dimension reduction, anomaly detection, noise reduction, etc. Notably, both the encoder and decoder in an autoencoder are designed to adapt the data types for better learning [35]. In our paper, the proposed autoencoder is used to reduce the noise in vibration data. To this end, the encoder and decoder are composed of LSTM layers recently mentioned to explore the short- and long-term dependencies of the vibration data. The detailed structure of our LSTM-Autoencoder is presented in Figure 2.

For more detail, the architecture contains two LSTM layers with 64 and 512 cells. To enhance model robustness, ReLU activation and dropout layers are added after each LSTM layer, inspired by Kunang et al. [36]. Moreover, a repeat vector layer is employed to duplicate the previous vector. Finally, a time-distributed layer is applied to each temporal slice of the input data. During the training process, the following mean squared error (MSE) is minimized [37].

$$L_{Autoencoder} = \sum_{t=1}^{T} [f_{Autoencoder}(x(t)) - x(t)]^2, \quad (1)$$

where T represents the total number of segments within the training data. $f_{\text{Autoencoder}}(x(t))$ denotes the LSTM-Autoencoder output derived from the input $x(t)$ at time t.

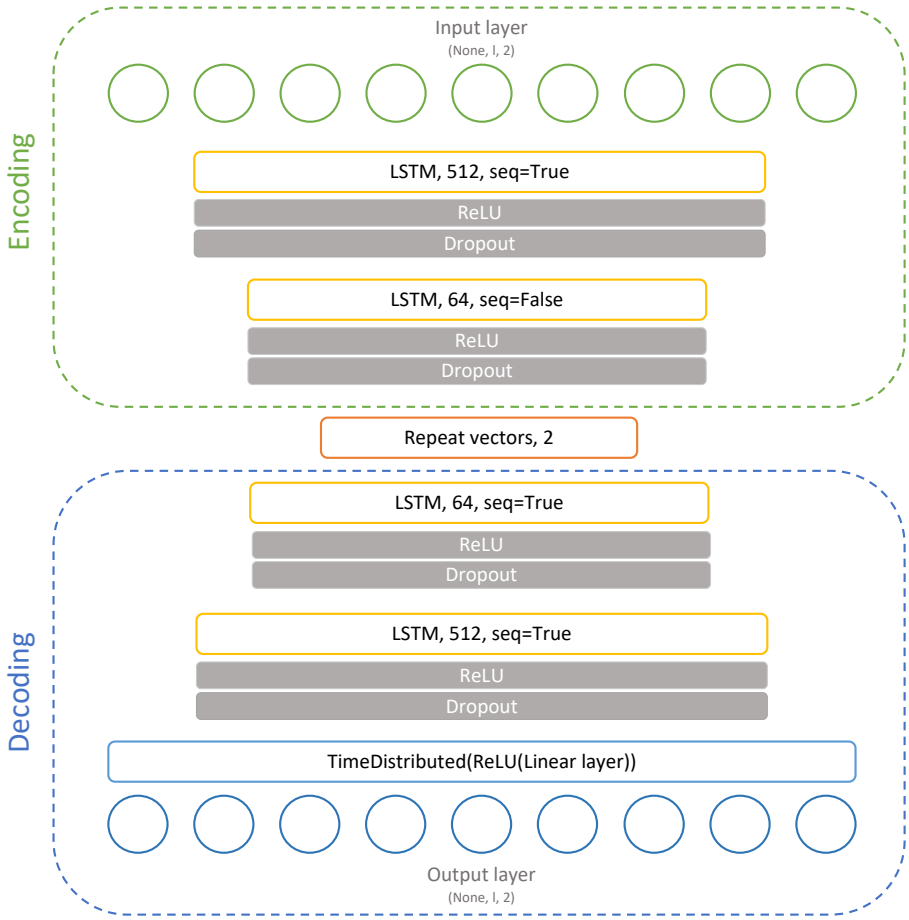

Figure 2. The architecture of LSTM-Autoencoder.

The optimization process involves minimizing $L_{\text{Autoencoder}}$ via Adam Optimization [38]. This proposed LSTM-Autoencoder is crucial for denoising vibration signals, strengthening the overall Robust-MBDL model towards higher resilience.

5. Health Indicator (HI) Construction

The purpose of this step is to determine the remaining useful life (RUL) at every time step. We employ two popular methods for this purpose: HI construction based on the first prediction time (HI-FPT), which is inspired by the work of Huang et al. (2021) [27], and HI construction based on Principal Component Analysis (PCA) using the Euclidean distance metric (HI-PCA), as explained in detail in Xu et al. (2022) [24].

5.1. HI-FPT

Most industrial equipment, including rotating machines, tend to degrade only after some time of operation. Trying to predict their remaining useful life (RUL) before any signs of degradation is unreliable and unnecessary. Hence, it is crucial to detect the initial degradation time, also known as the "First Prediction Time" (FPT) point. This time is significant because it marks the point at which the RUL prediction becomes reasonable. In this paper, the 3σ method, which has been recognized as a simple but efficient method to detect the FPT point according to the literature [39,40], is applied. This method comprises two phases, which are explained below:

- Learning phase: We first select the data in the period in which degradation does not exist, denoted $(1, T_0)$. The mean μ and the standard deviation σ are calculated from the selected data as follows:

$$\mu = \frac{1}{T_0}\sum_{i=1}^{T_0} x_i \text{ and } \sigma = \sqrt{\frac{1}{T_0}\sum_{i=1}^{T_0}(x_i - \mu)^2} \quad (2)$$

  where x_i represents the i^{th} data point.
- Detecting phase: If there exist two consecutive data points that are out of the normal interval $[\mu - 3\sigma, \mu + 3\sigma]$, the second point is considered as the FPT point. The condition of two consecutive points is used to reduce the likelihood of making a wrong decision due to noise.

The RUL is a function that increases linearly over time. Its maximal value is equal to 1 at the FPT point and decreases to 0 at the failure time, denoted by t_N. The value of RUL at an instant $t \in [FPT, t_N]$ is calculated as follows:

$$RUL_t = \frac{t_N - t}{t_N - FPT}. \quad (3)$$

5.2. HI-PCA

According to the HI-PCA method, the RUL values are determined based on the covariance matrix V calculated by PCA [24]. This matrix displays the shared features between time series data and its neighboring points, which accurately reflect the surrounding points' degradation trend. The calculation of the RUL value at t^{th} time involves determining the average Euclidean distance from that point in V to its sequential neighboring points.

$$RUL_t = \frac{1}{2}\left(\sqrt{\sum_{j=1}^{k}(V_j - V_{(t+1)_j})^2} + \sqrt{\sum_{j=1}^{k}(V_j - V_{(t-1)_j})^2}\right) \quad (4)$$

where k represents the k^{th} principal component.

6. Feature Extraction

In this paper, we incorporate a comprehensive approach to feature extraction by considering three essential categories: time domain, frequency domain, and time–frequency domain features. Time domain features provide insights into the overall behavior of vibration signals, capturing characteristics such as amplitude, mean, variance, and statistical measures directly in the time dimension. These features enable the detection of changes in signal morphology and amplitude, which are crucial for identifying early signs of machinery degradation. On the other hand, frequency domain features offer valuable information about the frequency content of the signals, aiding in the identification of fault-related frequencies and patterns. Additionally, time–frequency domain features, which combine both time and frequency information, provide a comprehensive understanding of signal dynamics over time. By considering features from all three domains, our approach enhances the signal-to-noise ratio and underscores relevant patterns in vibration data, facilitating effective RUL prediction.

6.1. Time-Domain Features

Eleven popular time-domain features, including root mean square, variance, kurtosis, etc., are used and reported in Table 2. These features have proved useful in detecting machinery faults. They are simple and can be quickly calculated. However, it is difficult to detect the change in frequencies based on these features.

Table 2. Time-domain features.

No.	Formula	Features
1	$RMS = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}$	Root Mean Square
2	$Var = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$	Variance
3	$PvT = \max(\lvert x_i \rvert)$	Peak value
4	$cf = \frac{PvT}{RMS}$	Crest factor
5	$Kur = \sum_{i=1}^{n} \frac{(x_i - \bar{x})}{n \cdot \mathrm{var}^2}$	Kurtosis
6	$Clf = \frac{PvT}{\frac{1}{n}\sum_{i=1}^{n}\lvert x_i \rvert}$	Clearance factor
7	$SF = \frac{RMS}{\frac{1}{n}\sum_{i=1}^{n}\lvert x_i \rvert}$	Shape factor
8	$LI = \sum_{i=0}^{n} \lvert x_{i+1} - x_i \rvert$	Line integral
9	$PP = \max(x_i) - \min(x_i)$	Peak to peak value
10	$Sk = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^3}{(\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2})^3}$	Skewness
11	$IF = \frac{PvT}{\frac{1}{n}\sum_{i=1}^{n}\lvert x_i \rvert}$	Impulse factor

6.2. Frequency-Domain Features

In reality, many types of bearing defects, such as outer race, inner race, or ball defects, can be efficiently detected in the frequency domain with the Fast Fourier Transform (FFT) [41]. We first used the FFT to convert the original signals to frequency-domain data.

$$X_k = \sum_{j=0}^{n-1} x_j \cdot e^{-i2\pi kj/n} \tag{5}$$

where x_j and X_k are the raw and frequency data, respectively.

The FFT transformation results are used to compute three frequency-domain features: FFT peak-to-peak values, energy, and power spectral density. These features are listed in Table 3. The features are a useful tool for stationary periodic signals but less effective for non-stationary signals that arise from time-dependent events, such as motor startup or changes in operating conditions.

Table 3. Frequency-domain features.

No.	Formula	Features
1	$r_k = \sum_{i=-\infty}^{\infty} x(t)e^{-iwt}$ $PvF = \max(r_k)$	Peak-to-peak value of FFT
2	$En = \sum_{k=1}^{N} r_k$	Energy of FFT
3	$PSD = \sum_{k=-\infty}^{\infty} r_k e^{-iwk}$	Power spectral density of FFT

6.3. Time–Frequency Domain Features

To capture the changes in frequencies over time due to the dynamic operation of rotating machines, the time–frequency features are extracted by using the Wavelet Continuous

Transform (CWT) [42]. The CWT uses a series of wavelets (small waves). The wavelet transform of a continuous signal $x(t)$ is defined as

$$CWT(a,b) = \frac{1}{\sqrt{c_\psi |a|}} \int_{-\infty}^{\infty} x(t)\psi\left(\frac{t-b}{a}\right)dt \quad (6)$$

where a in $\mathbb{R}$ and b in $\mathbb{R}^+$ are the location parameter and the scaling (dilation) parameter of the wavelet, respectively. $\psi(t)$ is the mother wavelet function, which is defined according to the signal inputs. In the paper, the Morlet wavelet [43] was chosen. This mother wavelet is similar to human perception (both hearing and vision). The formula for the Morlet wavelet is as follows:

$$\psi(t) = e^{-\frac{\beta t^2}{2}} e^{j\omega_0 t} \quad (7)$$

where $\beta = \omega_0^2$ and $c_\psi = \sqrt{\pi/\beta}$.

It is important to mention that while feature extraction can help in predicting the RUL by highlighting key patterns in the data, it can also result in the loss or distortion of information. Therefore, in addition to the 1D and 2D features, we also incorporate denoised data as the third input for our DL model.

7. Multi-Branch Deep Learning Network

Each type of feature recently mentioned has its own characteristics and, thus, requires a specific learning mechanism. Therefore, the proposed MBDL model comprises three individual learning branches that are designed to be compatible with each type of feature.

7.1. 1D Data Branch

This section is specifically tailored to explore the 1D data. To address this, we empirically developed a CNN-based architecture, illustrated in Figure 3.

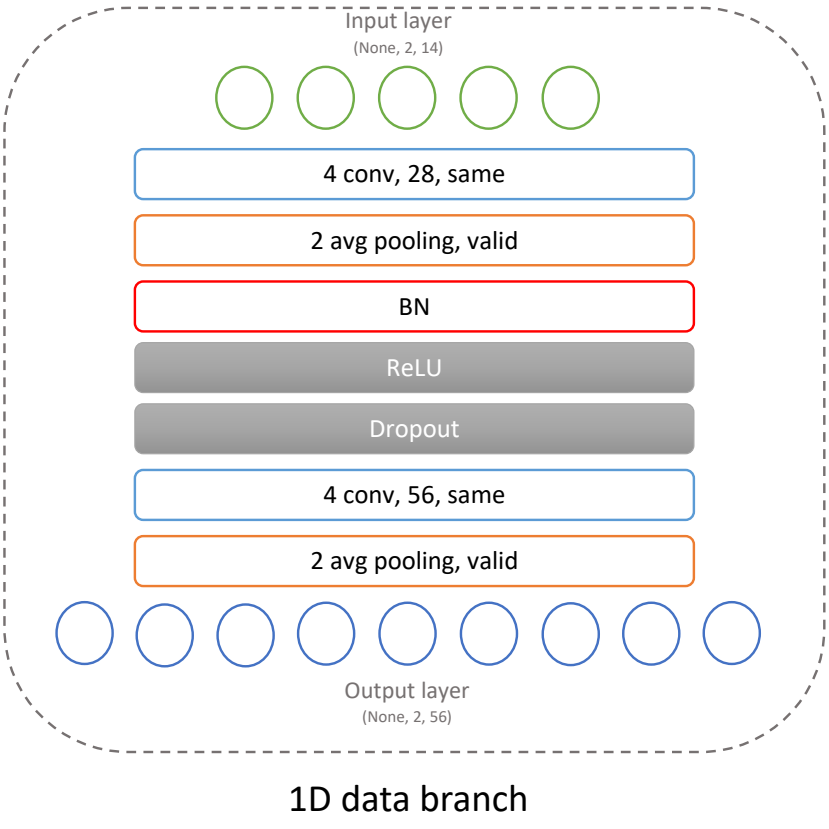

Figure 3. The architecture of the 1D data branch.

The main elements of this branch consist of convolutional layers, pooling layers, batch normalization (BN), dropout layers, and the Rectified Linear Unit (ReLU) activation function layers. The convolutional layers perform operations that involve the dot product or element-wise product between an input region, defined by a sliding window, and a trainable kernel to extract pertinent information from the input data. This process generates a feature map that encapsulates essential features from the entire input dataset. The ReLU activation function, represented as $ReLU(x) = \max(0, x)$, introduces non-linear characteristics into the network. Moreover, a batch normalization block is incorporated to optimize the training process by reducing internal covariance shift and normalizing the inputs between batches [44]. The pooling layers are integrated to decrease the dimensionality of the feature map by reducing redundant information. Similar to the convolutional layers, a sliding window traverses the feature map, and the average value (AVG pooling) within this window is computed. This reduction in dimensionality aims to retain essential information while improving computational efficiency.

It is important to note that the output dimension is larger than the input dimension. The purpose of this extension is to provide a more detailed and comprehensive depiction of the input information. By expanding the available space, the model becomes capable of capturing more intricate and meaningful relationships between the features, which ultimately improves its ability to learn from the data.

7.2. 2D Data Branch

This branch, as shown in Figure 4, is designed to process the 2D feature (time–frequency domain features) obtained from the wavelet transform. The underlying structure of this branch relies on ResNet-34 [30]. The ResNet-34 is a lightweight yet effective deep learning architecture with 34 layers that utilizes residual blocks. It integrates shortcuts and skip connections, facilitating the training of remarkably deep networks and mitigating the complexities associated with identifying intricate features within data. In addition, recognizing the limitations of traditional residual blocks in handling complex vibration data with sudden changes, we propose replacing them with the convolutional building block (CBB), proposed by Shaofeng Cai et al. in 2019 [31]. For more details, our 2D feature branch consists of four groups of CBBs. Each group contains three, three, five, and two CBBs, respectively. Finally, in each CBB, we employ batch normalization (BN), ReLU activation, and a dropout layer with a dropout rate of 0.2.

7.3. Denoised Data Branch

The purpose of this branch is to analyze the vibration data that are directly obtained from the denoising LSTM-Autoencoder. The direct explosion of the denoised vibration data is important since the information may be lost or deformed during the extraction of 1D and 2D data. The architecture of this branch (see Figure 5) was designed as an extension of the 2D feature branch, specifically tailored to better explore the vibration features. In particular, this branch consists of the same number of CBBs as that of our 2D feature branch; however, 1D convolutional layers were used instead of 2D convolutional layers. In addition, an average pooling layer with a window size of 4 was added after each CBB to capture all relevant features by considering their relationship, while the overall shape is smaller.

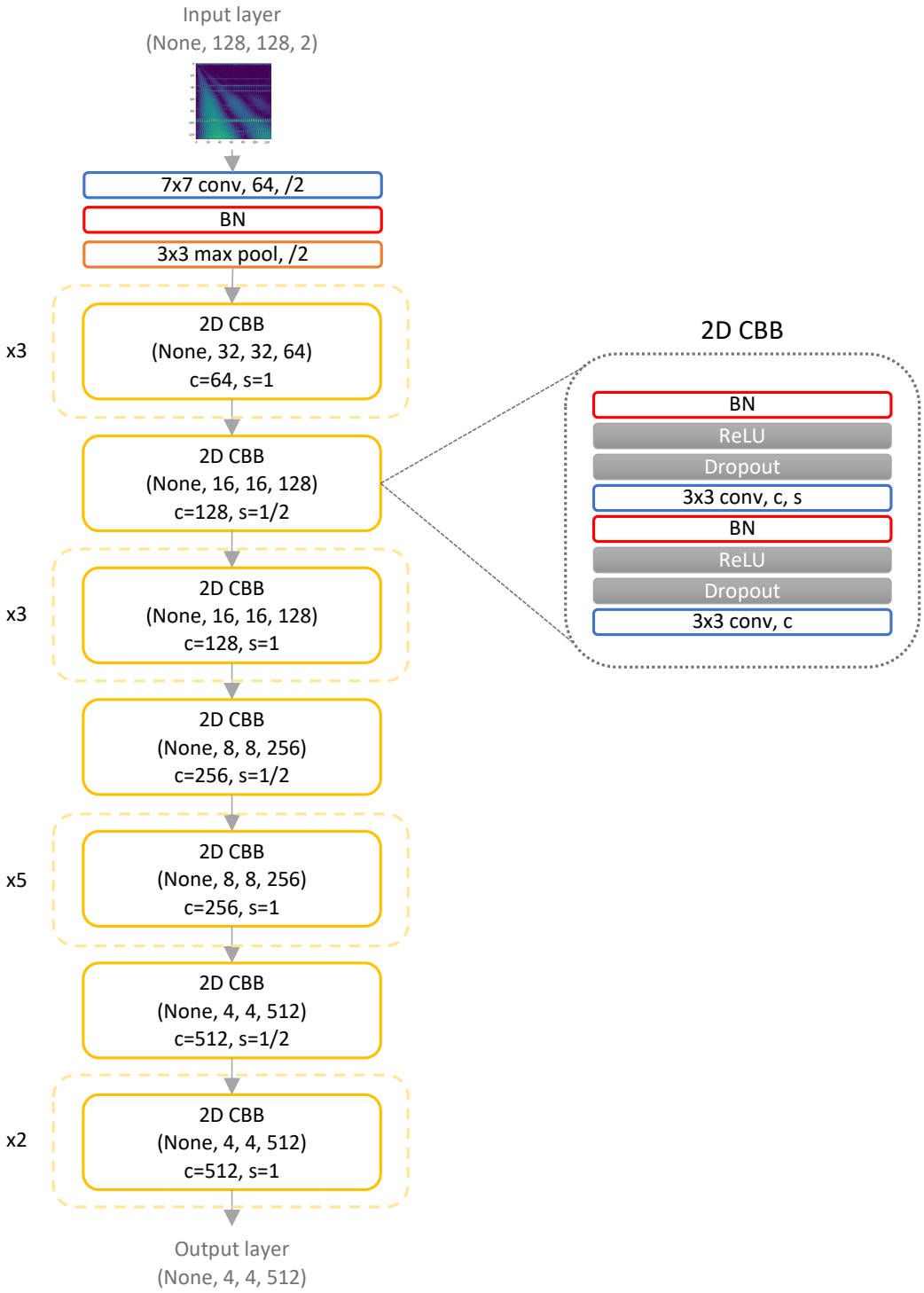

Figure 4. The architecture of the 2D data branch.

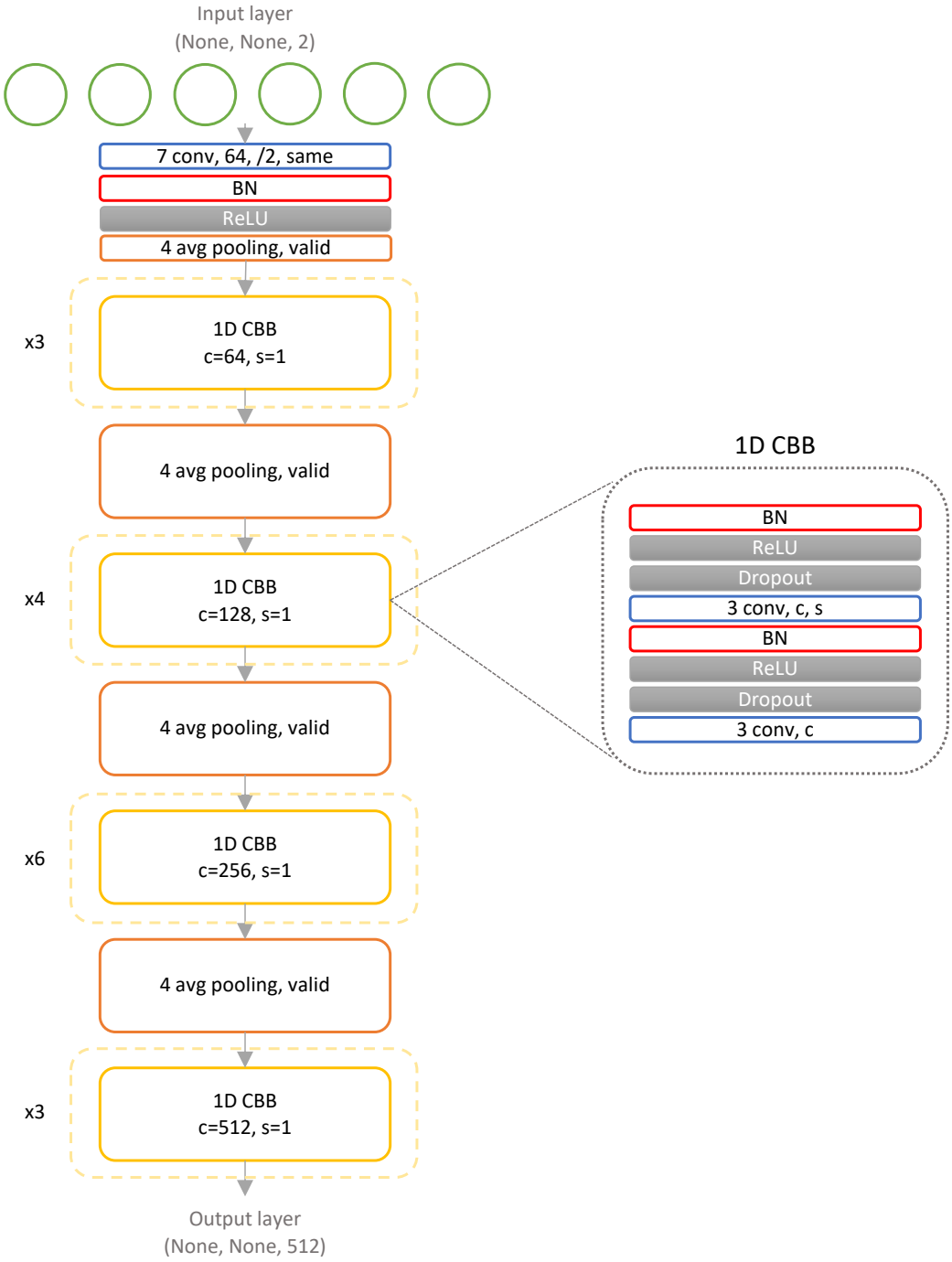

Figure 5. The architecture of the denoised data branch.

7.4. AB-LSTM and GAP

The AB-LSTM blocks are designed based on the Bi-LSTM architecture to optimize the RUL prediction task. The Bi-LSTM integrates both forward and backward hidden layers. This design allows the model to assimilate information from both past and future sequences, proving superior in tasks like RUL prediction compared to traditional LSTM networks [45]. Furthermore, self-attention mechanisms are also used to assist the Bi-LSTM in identifying significant input segments, leading to quicker convergence and improving the model performance [46]. For more details, Vaswani et al. [47] describe attention mechanisms as "mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query

with the corresponding key". Let Q, K, and V denote the query, key, and value vectors, respectively. The attention mechanism is described mathematically as follows:

$$Attention(Q, K, V) = Softmax[\frac{QK^{\top}}{\sqrt{d_k}}]V \tag{8}$$

and each head

$$H_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{9}$$

where $W_i^Q, W_i^K \in \mathbb{R}^{d_h \times d_v}$, and $W_i^V \in \mathbb{R}^{d_h \times d_v}$ are weight matrices, and d_v, d_k denote the projection subspaces' hidden dimensions. $\frac{1}{\sqrt{d_k}}$ is the scale factor that helps dot-product attention be faster when using a feed-forward network. Each H_i is concatenated into a matrix $W^O \in \mathbb{R}^{hd_v \times d_h}$ that integrates with projections to compile the data gathered from various positions on particular subspaces.

$$Attention(Q, K, V) = Concat(H_1, ..., H_h)W^O \tag{10}$$

In this paper, the number of heads (parallel attention layers) was fixed at $h = 16$ according to our tests. Hence, $\frac{d_v}{h} = \frac{d_k}{h} = 32$. The overall computing cost is comparable to that of single-head attention with full dimensionality because of the lower dimension of each head. The three AB-LSTM blocks' outputs are concatenated and passed to a linear layer with a Sigmoid activation function, ensuring a final output range of (0,1) [48].

The GAP layers are designed to automatically identify the machine's OC. The GAP layers are designed to automatically identify the machine's operating characteristics. The idea behind using GAP is to calculate the average of each feature map and feed it into a softmax layer, rather than using a fully connected layer. Compared to a fully connected layer, GAP is more suited to convolutional structures as it enforces correspondences between feature maps and categories and is more tolerant of spatial translations of the input. Additionally, there are no parameters to optimize [49]. Finally, three GAP layers' outputs are concatenated and fed to a linear layer with softmax activation to compute OC probabilities.

8. Experimental Settings

8.1. Datasets

In this paper, our proposed model was evaluated using the two benchmark datasets: XJTU-SY [9] and PRONOSTIA [10].

The XJTU-SY dataset was created by the Institute of Design Science and Basic Component at Xi'an Jiaotong University. It consists of 15 trials under three different operational conditions, referred to as from *Bearing*1_1 to *Bearing*3_5 in Table 4. The vibration data were collected from two PCB 352C33 accelerometers, each of which was installed at a 90° angle, with one on the horizontal axis and the other on the vertical axis. Each data segment contains 32,768 data points and was collected in one minute.

The PRONOSTIA dataset was published by the FEMTO-ST Institute in France and used in the 2012 IEEE Prognostic Challenge [10]. It consists of 17 accelerated run-to-failures on a small-bearing test rig, referred to as from *Bearing*1_1 to *Bearing*3_3 (Table 5). The bearing was operated under three operating conditions with different levels of rotation speed and load. The vibration signals include vertical and horizontal data, which were gathered by two miniature accelerometers positioned at 90°. Each data segment contains 2560 data points and was collected in 0.1 s.

Table 4. The XJTU-SY bearing dataset [9].

OC	Bearing Dataset	Bearing Lifetime (t_N)	Estimated FPT	Real FPT
Condition 1 (2100 rpm; 12,000 N)	*Bearing*1_1	2 h 3 m	1 h 16 m	-
	*Bearing*1_2	2 h 3 m	44 m	-
	*Bearing*1_3	2 h 38 m	1 h	-
	*Bearing*1_4	2 h 38 m	1 h 20 m	-
	*Bearing*1_5	52 m	39 m	-
Condition 2 (2250 rpm; 11,000 N)	*Bearing*2_1	8 h 11 m	7 h 35 m	-
	*Bearing*2_2	2 h 41 m	48 m	-
	*Bearing*2_3	8 h 53 m	5 h 27 m	-
	*Bearing*2_4	42 m	32 m	-
	*Bearing*2_5	5 h 39 m	2 h 21 m	-
Condition 3 (2400 rpm; 10,000 N)	*Bearing*3_1	42 h 18 min	39 h 4 min	-
	*Bearing*3_2	41 h 36 m	20 h 30 m	-
	*Bearing*3_3	6 h 11 m	5 h 40 m	-
	*Bearing*3_4	25 h 15 m	23 h 38 m	-
	*Bearing*3_5	1 h 54 m	9 m	-

Table 5. The PRONOSTIA bearing dataset [10].

OC	Bearing Dataset	Bearing Lifetime (t_N)	Estimated FPT	Real FPT
Condition 1 (1800 rpm; 4000 N)	*Bearing*1_1	28,030 s	5000 s	-
	*Bearing*1_2	8710 s	660 s	-
	*Bearing*1_3	18,020 s	5740 s	5730 s
	*Bearing*1_4	11,390 s	340 s	339 s
	*Bearing*1_5	23,020 s	1600 s	1610 s
	*Bearing*1_6	23,020 s	1460 s	1460 s
	*Bearing*1_7	15,020 s	7560 s	7570 s
Condition 2 (1650 rpm; 4200 N)	*Bearing*2_1	9110 s	320 s	-
	*Bearing*2_2	7970 s	2490 s	-
	*Bearing*2_3	12,020 s	7530 s	7530 s
	*Bearing*2_4	6120 s	1380 s	1390 s
	*Bearing*2_5	20,020 s	3100 s	3090 s
	*Bearing*2_6	5720 s	1280 s	1290 s
	*Bearing*2_7	1720 s	580 s	580 s
Condition 3 (1500 rpm; 5000 N)	*Bearing*3_1	5150 s	670 s	-
	*Bearing*3_2	16,370 s	1330 s	-
	*Bearing*3_3	3520 s	800 s	820 s

Tables 4 and 5 show detailed information on the two datasets. h, m, and s denote hours, minutes, and seconds, respectively. The tables report the estimated and real FPT. The estimated FPT is calculated using the FPT detection method in Section 5.1, and the real FPT is taken from the dataset if available.

8.2. Data Splitting

Almost all the state-of-the-art systems proposed for RUL detection on the XJTU-SY and PRONOSSTIA datasets used the data splitting methods from [27] and [24], respectively. We obey the data-splitting methods from these papers to compare our experimental results to

state-of-the-art systems. In particular, two splitting methods are proposed and referred to as the operating-condition-dependent rule (OC-dependent rule) and the operating-condition-independent rule (OC-independent rule).

- OC-independent method: This data-splitting method does not consider the operating condition of bearings [27]. Each bearing is selected as the test data, while all other bearings are utilized as training and validation data regardless of their operating conditions.
- OC-dependent method: The data-splitting method takes into account the bearing's operating condition [24]. Within each OC, two bearings are assigned to be the training and validation data, while the remaining bearings are reserved for model testing.

8.3. Validation Methods

To evaluate the performance of our model in RUL forecasting, we calculate the root mean square error (RMSE) and the mean absolute error (MAE) using the following equations:

$$RMSE = \sqrt{\sum_{t=FPT}^{t_N} \frac{(RUL_t - \widehat{RUL}_t)^2}{t_N - FPT}} \tag{11}$$

$$MAE = \sum_{t=FPT}^{t_N} \frac{|RUL_t - \widehat{RUL}_t|}{t_N - FPT} \tag{12}$$

The accuracy of the model in OC identification is determined by the accuracy score (Acc).

$$\text{Acc} = \frac{M}{P} \times 100 \tag{13}$$

where M denotes the number of well-classified segments among P classified segments.

8.4. Loss Functions

We used the mean squared logarithmic error (MSLE) [50] to calculate the difference between the real RUL (RUL_t) and the RUL estimated by our Robust-MBDL model ($\widehat{RUL}_t$) during both the training and testing phases:

$$L_{RUL} = \sum_{t=FPT}^{t_N} \frac{[log(RUL_t + 1) - log(\widehat{RUL}_t + 1)]^2}{t_N - FPT} \tag{14}$$

It is noted that in the above equation, the RUL values are increased by 1 to prevent taking the logarithm of zero when the RUL equals 0.

For the OC classification task, we employed categorical cross-entropy loss [51], a widely used loss function for multiclass classification problems [52]. Let m denote the total number of possible operational conditions: $OC = (c_1, c_2, ..., c_m)$ represents the real operational condition; $\widehat{OC} = (\hat{c}_1, \hat{c}_2, ..., \hat{c}_m)$ represents the operational condition classified by our model. The cross-entropy loss can be calculated as

$$L_{OC} = -\sum_{i=1}^{m} c_i \cdot \log(\hat{c}_i) \tag{15}$$

Our model simultaneously addresses RUL prediction and OC classification. The two above loss functions are then combined to form the following global loss function:

$$L = \lambda L_{OC} + (1 - \lambda) L_{RUL} \tag{16}$$

where λ is a real number that ranges between 0 and 1. By adjusting the value of λ, two things can be achieved: *(i)* offset any imbalances between the two loss functions in the

global one; *(ii)* give varying degrees of importance to each task depending on the particular study case. In our paper, we determined through experimentation that λ is best set to 0.6.

8.5. Deep Neural Network Implementation

In this study, we implemented all proposed deep neural networks using the Tensorflow framework and utilized the Root Mean Squared Propagation (RMSProp) method for model optimization [53]. We conducted all experiments on an Nvidia A100 GPU.

Table 6 details the specific settings applied during the training processes for both the denoised LSTM-Autoencoder and the MBDL parts. Moreover, it is crucial to optimize the number of attention heads as it greatly impacts the model's performance [54]. Table 7 shows results for different numbers of heads tested. Sixteen attention heads were selected to enhance RUL predictions by allowing the model to focus on critical input aspects. It should be noted that during the training process, we applied the K-fold cross-validation method for time series with K = 3 [55] to ensure the model's robustness.

Table 6. Parameters of training process.

Model	Optimizer	Learning Rate	Batch Size	Epochs
MBDL	RMSProp	1×10^{-4}	16	1000
LSTM-Autoencoder	RMSProp	1×10^{-4}	16	300

Table 7. Model's performance with respect to the different head sizes.

Number of Heads	OC Acc	MAE	RMSE
32	20.9446	0.2104	0.2653
24	27.6873	0.2319	0.286
16	37.8936	0.206	0.2566
8	30.4622	0.2203	0.2857

9. Experimental Results and Discussions

We evaluated the performance of our proposed Robust-MBDL model for various scenarios, including RUL prediction and OC identification, using the PRONOSTIA and XJTU-SY datasets, with both OC-dependent and OC-independent rules, with and without the denoised LSTM-Autoencoder. The model's performance was also compared to various state-of-the-art ones, including BLSTM [26], MLP and DCNN–MLP [27], SACGNet [24], and MSCNN [20]. The obtained results are reported in Tables 8–11.

Table 8. Results of the performance analysis for the XJTU-SY dataset with OC-independent rule.

Test Bearing	MLP [27]		BLSTM [26]		MSCNN [20]		DCNN–MLP [27]		MBDL			Robust-MBDL		
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	Acc (%)	RMS	MAE	Acc (%)
*Bearing*1_1	0.274	0.240	0.228	0.191	0.242	0.213	0.206	0.176	0.0944	0.0745	**100.0**	**0.0922**	**0.0739**	**100.0**
*Bearing*1_2	0.313	0.270	0.305	0.231	0.262	0.229	0.240	0.207	0.0453	0.037	**100.0**	**0.033**	**0.021**	**100.0**
*Bearing*1_3	0.261	0.221	0.130	0.106	0.184	0.155	0.178	0.151	**0.0552**	**0.049**	100.0	0.057	0.52	**100.0**
*Bearing*1_5	0.318	0.265	0.362	0.314	0.215	0.181	0.184	0.155	0.0592	0.0531	**100.0**	**0.0491**	**0.0376**	**100.0**
*Bearing*2_1	0.203	0.172	0.152	0.129	0.148	0.126	0.117	0.099	**0.0867**	**0.0806**	**100.0**	0.0877	0.0803	**100.0**
*Bearing*2_2	0.266	0.214	0.134	0.094	0.232	0.194	0.122	0.102	0.0555	0.0453	**100.0**	**0.0365**	**0.0321**	**100.0**
*Bearing*2_3	0.230	0.204	0.216	0.170	0.199	0.164	0.158	0.126	0.0588	0.0525	**100.0**	**0.0576**	**0.0512**	**100.0**
*Bearing*2_4	0.251	0.213	0.311	0.267	0.231	0.195	0.177	0.141	**0.0771**	0.0657	**100.0**	0.0775	0.0639	**100.0**
*Bearing*2_5	0.234	0.202	0.308	0.278	0.108	0.090	0.0918	0.075	0.0596	0.0505	**100.0**	**0.0429**	**0.0398**	**100.0**
*Bearing*3_1	0.305	0.262	0.351	0.297	0.247	0.214	0.244	0.204	0.0575	0.0489	**100.0**	**0.0509**	**0.0418**	**100.0**
*Bearing*3_3	0.318	0.276	0.188	0.162	0.191	0.156	0.158	0.129	0.0575	0.0459	**100.0**	**0.0365**	**0.0214**	**100.0**
*Bearing*3_4	0.252	0.220	0.175	0.135	0.165	0.139	0.132	0.107	0.0837	0.0709	**100.0**	**0.0792**	**0.0708**	**100.0**
*Bearing*3_5	0.376	0.310	0.305	0.251	0.267	0.225	0.266	0.219	0.0733	0.0598	**100.0**	**0.0685**	**0.0517**	**100.0**
Average performance	0.282	0.233	0.232	0.188	0.204	0.170	0.170	0.137	0.0673	0.0561	**100.0**	**0.0564**	**0.0471**	**100.0**

Table 9. Results of the performance analysis for the PRONOSTIA dataset with OC-independent rule.

Test Bearing	MLP [27]		BLSTM [26]		MSCNN [20]		DCNN–MLP [27]		MBDL		Robust-MBDL	
	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE
*Bearing*1_1	0.332	0.277	0.268	0.245	0.152	0.122	0.194	0.161	0.158	0.121	**0.0864**	**0.0699**
*Bearing*1_2	0.256	0.213	0.281	0.242	0.484	0.386	0.254	0.219	0.167	0.146	**0.0964**	**0.0854**
*Bearing*1_3	0.235	0.186	0.331	0.270	0.251	0.208	0.199	0.164	**0.135**	0.112	0.1467	**0.0691**
*Bearing*1_4	0.515	0.439	0.513	0.443	0.397	0.329	0.132	0.107	0.101	0.081	**0.1038**	**0.0768**
*Bearing*1_5	0.107	0.320	0.208	0.174	0.326	0.276	0.187	0.158	0.165	0.136	**0.1027**	**0.0779**
*Bearing*1_6	0.480	0.480	0.329	0.278	0.340	0.273	0.328	0.270	0.088	0.071	**0.0746**	**0.0593**
*Bearing*1_7	0.170	0.153	0.165	0.141	0.357	0.299	0.205	0.172	**0.088**	**0.071**	0.0997	0.0822
Average performance	0.272	0.272	0.267	0.229	0.296	0.251	0.231	0.194	0.145	0.120	**0.0927**	**0.0768**

The results presented in Tables 8 and 9 demonstrate the superior performance of our proposed Robust-MBDL model under the OC-independent rule for data splitting. Whether the denoised LSTM-Autoencoder is applied or not, it outperforms the state-of-the-art models for RUL prediction in terms of RMSE and MAE scores across all bearing types. Figure 6 shows an example of the RUL prediction for *Bearing*1_3 and *Bearing*1_4. We consistently observe minimal disparity between actual and predicted RUL, providing strong evidence of our approach's reliability and effectiveness.

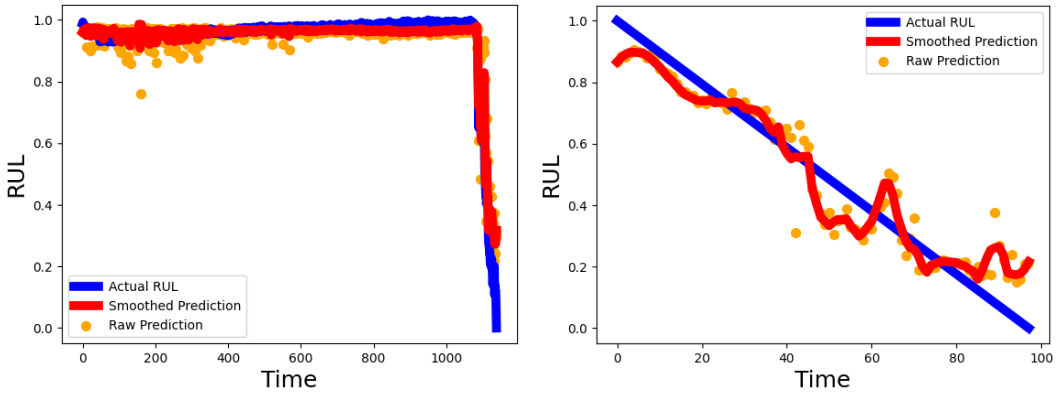

Figure 6. Illustration of the RUL prediction by the Robust-MBDL model. (**a**) *Bearing*1_4, PRONOSTIA dataset; (**b**) *Bearing*1_3, XJTU-SY dataset.

Regarding the OC identification task, the network shows exceptional performance, achieving 100% accuracy for all bearing types. The OC-independent data splitting approach allows for a substantial volume of training data relative to testing data—testing on a signal bearing while using the others for training. This methodology makes the OC classification task relatively straightforward. In contrast, using the OC-dependent method, when the amount of training data is less than the testing data, accuracy is likely to decrease substantially.

It is important to highlight that by training with two tasks (RUL prediction and OC classification) simultaneously, the proposed models are able to learn the complex relationships between the operating conditions of the bearings and their degradation patterns, leading to the high performance of these models. Finally, utilizing the denoised LSTM-Autoencoder, the Robust-MBDL shows outstanding performance in most bearings, proving the efficacy and necessity of the data denoising.

Tables 10 and 11 show the performance analysis of our model using the OC-dependent splitting rule. It is worth noting that only SACGNet was considered for the analysis because the other models did not utilize the OC-dependent rule. Our proposed models showed significant superiority over the SACGNet model for all bearings of the XJTU-SY dataset. In the PRONOSTIA dataset, our models performed notably better than SACGNet in almost all bearings, except for *Bearing*1_5 and *Bearing*1_7 in terms of RMSE. Our proposed model demonstrated competitive performance compared to the SACGNet model regarding MAE scores in the PRONOSTIA dataset. It is worth noting that the OC classification of *Bearing*1_4 in Table 10 was relatively poor. The poor performance of this bearing can be attributed to its unique features, which significantly differ from other bearings operating under the same conditions. This observation has been reported in related works [27]. Finally, the obtained results again underscore the significant improvements in RMSE and MAE scores across almost all bearing types when the denoised LSTM-Autoencoder is used.

Table 10. Results of the performance analysis for the XJTU-SY dataset with the OC-dependent rule.

Type	SACGNet [24]		MBDL			Robust-MBDL		
	RMSE	MAE	RMSE	MAE	Acc	RMSE	MAE	Acc
*Bearing*1_3	0.147	0.117	**0.126**	0.076	**100.0**	0.139	**0.072**	**100.0**
*Bearing*1_4	0.166	0.088	**0.08**	0.043	**4.91**	0.087	**0.035**	0.0
*Bearing*1_5	0.360	0.206	0.199	0.093	98.07	**0.177**	**0.091**	**100.0**
*Bearing*2_3	0.320	0.307	**0.133**	**0.087**	85.17	0.218	0.164	**85.74**
*Bearing*2_4	0.511	0.428	**0.105**	**0.056**	88.09	0.223	0.103	**90.47**
*Bearing*2_5	0.341	0.249	**0.189**	**0.123**	66.07	0.201	0.169	**77.87**
*Bearing*3_3	0.369	0.256	**0.035**	**0.018**	**99.73**	0.177	0.054	97.8437
*Bearing*3_4	0.193	0.069	**0.038**	**0.021**	29.17	0.159	0.129	**87.78**
*Bearing*3_5	0.500	0.447	**0.263**	**0.231**	**96.49**	0.312	0.24	**96.49**

Table 11. Results of the performance analysis for the PRONOSTIA dataset with the OC-dependent rule.

Type	SACGNet [24]		MBDL			Robust-MBDL		
	RMSE	MAE	RMSE	MAE	Acc	RMSE	MAE	Acc
*Bearing*1_3	0.101	0.041	0.0624	**0.0241**	99.3341	**0.0594**	0.0281	**99.4451**
*Bearing*1_4	0.230	0.157	0.045	0.0222	**99.4732**	**0.0394**	**0.0213**	97.2783
*Bearing*1_5	**0.197**	**0.077**	0.2407	0.1953	**99.3918**	0.2259	0.1777	99.2615
*Bearing*1_6	0.205	0.079	0.1376	0.0879	99.5656	**0.1304**	**0.079**	**99.305**
*Bearing*1_7	**0.108**	**0.022**	0.224	0.1854	**100.0**	0.2038	0.1635	99.8668
*Bearing*2_3	0.131	0.033	0.1306	0.1012	**98.3361**	**0.1288**	**0.0993**	98.9185
*Bearing*2_4	0.204	**0.081**	**0.1579**	0.1295	96.732	0.1669	0.1374	**97.7124**
*Bearing*2_5	0.202	**0.071**	**0.1319**	0.116	88.2617	0.1523	0.1311	**94.955**
*Bearing*2_6	0.205	**0.083**	**0.2167**	0.1566	**100.0**	0.2275	0.1739	**100.0**
*Bearing*2_7	0.397	0.220	0.1398	0.1113	**100.0**	**0.1391**	**0.1082**	**100.0**
*Bearing*3_3	0.280	0.161	**0.2142**	**0.1097**	**100.0**	0.2163	0.1125	93.75

Table 12 shows the comparative results of our proposed methods and other state-of-the-art methods, as well as traditional ML methods. To ensure a fair comparison, we conducted our experiments to produce the scores in this table based on the configuration described in Zheng et al. (2024) [23]. The validation metric employed in this experiment is the RMSE. The experiments utilized condition 2 of the XJTU-SY dataset. According to the study [23], we adopted a data splitting method analogous to our OC-independent method, where four bearings were selected for the training set, and the remaining one served as the test set for each iteration of training. The results demonstrate that our Robust-MBDL with the denoising strategy (the proposed LSTM-Autoencoder) outperforms models like LSTM, CNN, and GCN-SA by achieving lower RMSE scores in several cases, such as Bearing2-1 and Bearing2-4. It also presents the lowest average RMSE score among all evaluated state-of-the-art methods at 0.15386. This indicates a significant advancement in prediction accuracy and showcases the effectiveness of the denoising component in enhancing RUL prediction performance. Conversely, while the Robust-MBDL without the denoising method yielded improvements over several older models, it did not surpass the denoising version, underscoring the value added by this step in our methodology.

Table 12. Performance comparison of different models with OC-independent rule on XJTU-SY dataset.

Model	*Bearing*2_1	*Bearing*2_2	*Bearing*2_3	*Bearing*2_4	*Bearing*2_5	Average
LSTM [23]	0.364	0.351	0.334	0.386	0.355	0.358
CNN [23]	0.253	0.336	0.258	0.259	0.121	0.245
GCN-SA [22]	0.190	0.184	0.217	0.333	0.166	0.218
MSCNN [20]	0.148	0.232	0.199	0.231	0.108	0.184
GCU-Transformer [21]	0.356	0.141	0.197	0.161	0.149	0.201
DMWBT [56]	0.195	**0.120**	0.176	0.101	0.233	0.165
DRL [23]	0.184	0.212	**0.127**	0.092	**0.095**	0.142
Ridge Regression [57]	0.2647	0.5161	0.6669	0.328	0.3422	0.4238
Random Forest Regression [58]	0.4458	0.3287	0.3576	0.2889	0.2658	0.3374
XGBoost Regression [59]	0.3464	0.2819	0.3101	0.3226	0.2878	0.30976
MBDL	0.1217	0.1383	0.2258	0.2221	0.1135	0.164
Robust-MBDL	**0.1069**	0.1262	0.1771	**0.0862**	0.2729	**0.15386**

Figure 7 depicts the adaptation of the proposed Robust-MBDL model for SHapley Additive exPlanations (SHAP) [60] validation to understand the impact of input features on the prediction output. In this adaptation, the model undergoes slight modifications to include linear layers with one-dimensional output spaces and Sigmoid activation functions, positioned after the AB-LSTM blocks. These additional layers generate outputs that represent the impact of input features (2D data, 1D data, and denoised data) on the model output. Subsequently, these representation points are utilized in the SHAP validation process. Notably, the final linear layer of the revised Robust-MBDL model, equipped with one-dimensional output spaces and Sigmoid activation functions, serves as the prediction model in this SHAP mechanism.

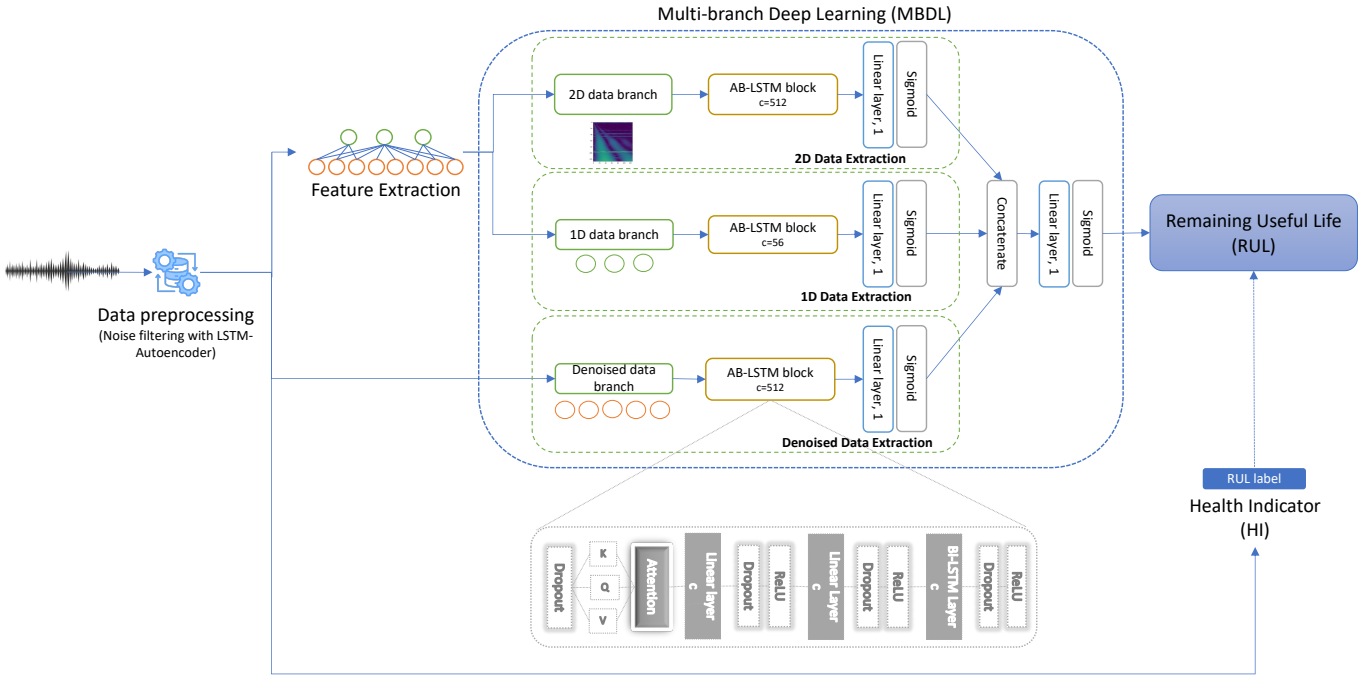

Figure 7. The high-level architecture of our proposed Robust-MBDL model used for SHAP validation.

In Figure 8, the SHAP summary plots for the XJTU bearings under condition 2 trained under the OC-independent rule demonstrate how feature values influence the predictive model. For 1D data, higher values generally lead to a negative impact on the model's output, while lower values have a positive impact. In the case of 2D data, high values slightly detract from the model's predictions, whereas low values greatly enhance them.

Denoised data appear to have a neutral effect regardless of the feature value. These patterns suggest that while 1D and 2D data features significantly drive the model's predictions, denoised data do not alter the outcome, highlighting the importance of feature selection in model performance.

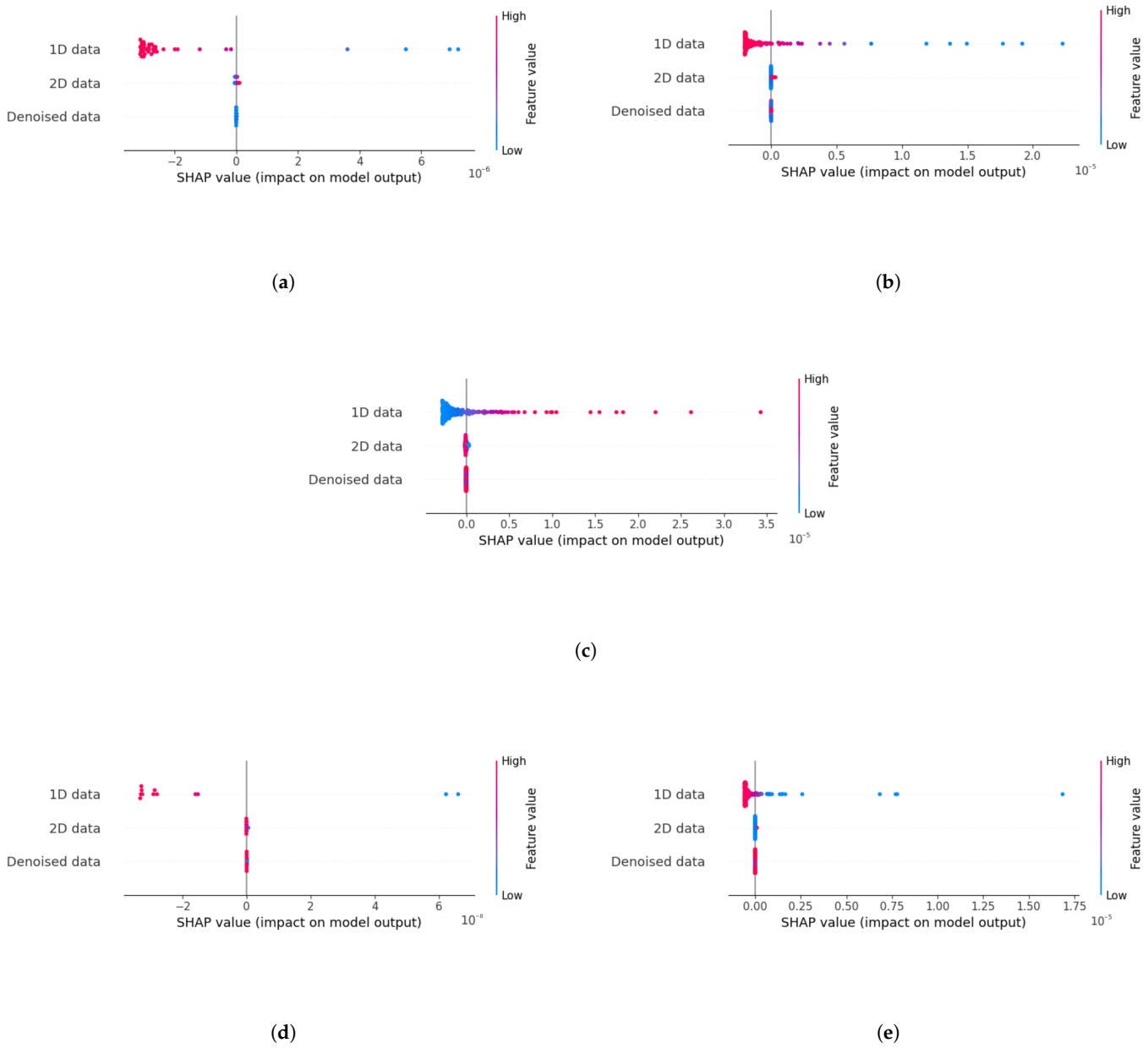

Figure 8. SHAP plots for five test cases. (**a**) *Bearing*2_1; (**b**) *Bearing*2_2; (**c**) *Bearing*2_3; (**d**) *Bearing*2_4; (**e**) *Bearing*2_5.

The figure in reference to Figure 9 illustrates the effect of noise on the predictions made by our Robust-MBDL model with denoising. The testing environment setting aligns with that of Table 12. In this instance, *Bearing*2_2 serves as the testing data, while all other bearings in condition 2 are used for training. Gaussian random noise, as described by Peebles (2001) [61], is artificially applied to simulate various noise levels encountered in real-life manufacturing scenarios. Five different standard deviation values are utilized: 0.01, 0.2, 0.3, and 0.5. The graph reveals that while our model's predictive accuracy is impacted

by higher levels of noise, overall performance remains satisfactory. Notably, even with a high noise level of 0.5 standard deviation, the model achieves a good RMSE score of 0.22.

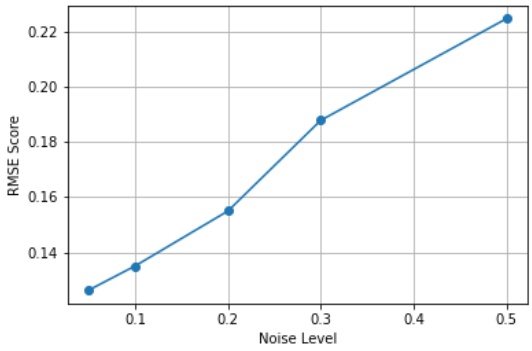

Figure 9. Effect of noise on the prediction of our Robust-MBDL.

10. Conclusions

This paper presented the robust MDL model for the prediction of remaining useful life (RUL) and the classification of the Operating Conditions (OC) of rotating machines. The model comprises several key components: a denoising LSTM-autoencoder responsible for data denoising, three parallel branches (1D data branch, 2D data branch based on Resnet-34 architecture, and a denoised data branch) for feature extraction, AB-LSTM blocks for RUL prediction, and GAP blocks for OC classification. This parallel architecture empowers the proposed model to capture intricate relationships between bearing operating conditions and degradation patterns, resulting in superior performance in both RUL prediction and OC classification tasks. Furthermore, in addition to the raw data, a comprehensive set of features, including 11 time-domain, 3 frequency-domain, and 2D time–frequency domain features, is computed and utilized as rich input for our model. To assess the model's performance, we compared it to state-of-the-art models on both the PRONOSTIA and XJTU-SY datasets. The obtained results indicate that our model outperforms others on both datasets. It serves as a reliable diagnostic tool, helping to identify, monitor, and prevent mechanical failures. In addition, the paper's broader concept proposes a model that can leverage different data types, making it a suitable tool for any industrial fault detection and prediction applications. In our upcoming endeavors, we intend to evaluate the effectiveness and resilience of our models in practical scenarios. We also plan to enhance the models by incorporating transfer learning and data augmentation techniques. Additionally, we aim to explore more computationally efficient architectures like MobileNets or EfficientNets for feature extraction.

Author Contributions: Methodology, K.T. and H.-C.V.; Validation, H.-C.V., L.P. and N.B.; Data curation, K.T.; Writing—original draft, K.T.; Writing—review & editing, H.-C.V.; Supervision, H.-C.V., L.P., N.B. and H.-S.-H.N.; Project administration, H.-S.-H.N.; Funding acquisition, H.-S.-H.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by the Vietnamese Ministry of Education and Training under project number: B2024.DNA.18.

Data Availability Statement: No new data were created or analyzed in this study.

Conflicts of Interest: Khoa Tran was employed by the company AIWARE Limited Company. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest. The authors declare that this study received funding from the Vietnamese Ministry of Education and Training under project number. The funder was involved in the study by providing necessary funding for group meetings and a portion of the computational resources.

References

1. Lv, Y.; Zheng, P.; Yuan, J.; Cao, X. A Predictive Maintenance Strategy for Multi-Component Systems Based on Components' Remaining Useful Life Prediction. *Mathematics* **2023**, *11*, 3884. [CrossRef]
2. Louhichi, R.; Sallak, M.; Pelletan, J. A Study of the Impact of Predictive Maintenance Parameters on the Improvment of System Monitoring. *Mathematics* **2022**, *10*, 2153. [CrossRef]
3. Lyu, Y.; Zhang, Q.; Wen, Z.; Chen, A. Remaining useful life prediction based on multi-representation domain adaptation. *Mathematics* **2022**, *10*, 4647. [CrossRef]
4. Deng, F.; Bi, Y.; Liu, Y.; Yang, S. Deep-learning-based remaining useful life prediction based on a multi-scale dilated convolution network. *Mathematics* **2021**, *9*, 3035. [CrossRef]
5. Li, Y.; Kurfess, T.; Liang, S. Stochastic prognostics for rolling element bearings. *Mech. Syst. Signal Process.* **2000**, *14*, 747–762. [CrossRef]
6. Zhang, J.X.; Du, D.B.; Si, X.S.; Liu, Y.; Hu, C.H. Prognostics based on stochastic degradation process: The last exit time perspective. *IEEE Trans. Reliab.* **2021**, *70*, 1158–1176. [CrossRef]
7. Lorton, A.; Fouladirad, M.; Grall, A. Computation of remaining useful life on a physic-based model and impact of a prognosis on the maintenance process. *Proc. Inst. Mech. Eng. Part O J. Risk Reliab.* **2013**, *227*, 434–449. [CrossRef]
8. Xi, X.; Zhou, D. Prognostics of fractional degradation processes with state-dependent delay. *Proc. Inst. Mech. Eng. Part O J. Risk Reliab.* **2022**, *236*, 114–124. [CrossRef]
9. Wang, B.; Lei, Y.; Li, N.; Li, N. A hybrid prognostics approach for estimating remaining useful life of rolling element bearings. *IEEE Trans. Reliab.* **2018**, *69*, 401–412. [CrossRef]
10. Nectoux, P.; Gouriveau, R.; Medjaher, K.; Ramasso, E.; Chebel-Morello, B.; Zerhouni, N.; Varnier, C. PRONOSTIA: An experimental platform for bearings accelerated degradation tests. In Proceedings of the IEEE International Conference on Prognostics and Health Management, Denver, CO, USA, 18–21 June 2012; pp. 1–8.
11. Xu, Z.; Guo, Y.; Saleh, J.H. VisPro: A prognostic SqueezeNet and non-stationary Gaussian process approach for remaining useful life prediction with uncertainty quantification. *Neural Comput. Appl.* **2022**, *34*, 14683–14698. [CrossRef]
12. Villalón-Falcón, A.; Prieto-Moreno, A.; Quiñones-Grueiro, M.; Llanes-Santiago, O. Computational adaptive multivariable degradation model for improving the remaining useful life prediction in industrial systems. *Comput. Appl. Math.* **2022**, *41*, 48. [CrossRef]
13. Chen, W.; Liu, C.; Chen, Q.; Wu, P. Multi-scale memory-enhanced method for predicting the remaining useful life of aircraft engines. *Neural Comput. Appl.* **2023**, *35*, 2225–2241. [CrossRef]
14. Mohril, R.S.; Solanki, B.S.; Kulkarni, M.S.; Lad, B.K. XGBoost based residual life prediction in the presence of human error in maintenance. *Neural Comput. Appl.* **2023**, *35*, 3025–3039. [CrossRef]
15. Ai, S.; Song, J.; Cai, G. Sequence-to-sequence remaining useful life prediction of the highly maneuverable unmanned aerial vehicle: A multilevel fusion transformer network solution. *Mathematics* **2022**, *10*, 1733. [CrossRef]
16. Chen, W.; Chen, W.; Liu, H.; Wang, Y.; Bi, C.; Gu, Y. A RUL prediction method of small sample equipment based on DCNN-BiLSTM and domain adaptation. *Mathematics* **2022**, *10*, 1022. [CrossRef]
17. Wang, X.; Huang, T.; Zhu, K.; Zhao, X. LSTM-based broad learning system for remaining useful life prediction. *Mathematics* **2022**, *10*, 2066. [CrossRef]
18. Fan, Z.; Li, W.; Chang, K.C. A Bidirectional Long Short-Term Memory Autoencoder Transformer for Remaining Useful Life Estimation. *Mathematics* **2023**, *11*, 4972. [CrossRef]
19. Hinchi, A.Z.; Tkiouat, M. Rolling element bearing remaining useful life estimation based on a convolutional long-short-term memory network. *Procedia Comput. Sci.* **2018**, *127*, 123–132. [CrossRef]
20. Zhu, J.; Chen, N.; Peng, W. Estimation of bearing remaining useful life based on multiscale convolutional neural network. *IEEE Trans. Ind. Electron.* **2018**, *66*, 3208–3216. [CrossRef]
21. Mo, Y.; Wu, Q.; Li, X.; Huang, B. Remaining useful life estimation via transformer encoder enhanced by a gated convolutional unit. *J. Intell. Manuf.* **2021**, *32*, 1997–2006. [CrossRef]
22. Wei, Y.; Wu, D.; Terpenny, J. Bearing remaining useful life prediction using self-adaptive graph convolutional networks with self-attention mechanism. *Mech. Syst. Signal Process.* **2023**, *188*, 110010. [CrossRef]
23. Zheng, G.; Li, Y.; Zhou, Z.; Yan, R. A Remaining Useful Life Prediction Method of Rolling Bearings Based on Deep Reinforcement Learning. *IEEE Internet Things J.* **2024**. [CrossRef]
24. Xu, J.; Duan, S.; Chen, W.; Wang, D.; Fan, Y. SACGNet: A Remaining Useful Life Prediction of Bearing with Self-Attention Augmented Convolution GRU Network. *Lubricants* **2022**, *10*, 21. [CrossRef]
25. Al-Dulaimi, A.; Zabihi, S.; Asif, A.; Mohammadi, A. Hybrid deep neural network model for remaining useful life estimation. In Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 3872–3876.
26. Huang, C.G.; Huang, H.Z.; Li, Y.F. A bidirectional LSTM prognostics method under multiple operational conditions. *IEEE Trans. Ind. Electron.* **2019**, *66*, 8792–8802. [CrossRef]
27. Huang, C.G.; Huang, H.Z.; Li, Y.F.; Peng, W. A novel deep convolutional neural network-bootstrap integrated method for RUL prediction of rolling bearing. *J. Manuf. Syst.* **2021**, *61*, 757–772. [CrossRef]

28. Cheng, Y.; Hu, K.; Wu, J.; Zhu, H.; Lee, C.K. A deep learning-based two-stage prognostic approach for remaining useful life of rolling bearing. *Appl. Intell.* **2022**, *52*, 5880–5895. [CrossRef]
29. An, Q.; Tao, Z.; Xu, X.; El Mansori, M.; Chen, M. A data-driven model for milling tool remaining useful life prediction with convolutional and stacked LSTM network. *Measurement* **2020**, *154*, 107461. [CrossRef]
30. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
31. Cai, S.; Shu, Y.; Chen, G.; Ooi, B.C.; Wang, W.; Zhang, M. Effective and efficient dropout for deep convolutional neural networks. *arXiv* **2019**, arXiv:1904.03392.
32. Zhang, J.; Yin, P. Multivariate time series missing data imputation using recurrent denoising autoencoder. In Proceedings of the 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), San Diego, CA, USA, 18–21 November 2019; pp. 760–764.
33. Essien, A.; Giannetti, C. A deep learning framework for univariate time series prediction using convolutional LSTM stacked autoencoders. In Proceedings of the 2019 IEEE International Symposium on INnovations in Intelligent SysTems and Applications (INISTA), Sofia, Bulgaria, 3–5 July 2019; pp. 1–6.
34. Sugiyama, M.; Sawai, H.; Waibel, A.H. Review of tdnn (time delay neural network) architectures for speech recognition. In Proceedings of the 1991 IEEE International Symposium on Circuits and Systems (ISCAS), Singapore, 11–14 June 1991; pp. 582–585.
35. Vincent, P.; Larochelle, H.; Lajoie, I.; Bengio, Y.; Manzagol, P.A.; Bottou, L. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.* **2010**, *11*, 3371–3408.
36. Kunang, Y.N.; Nurmaini, S.; Stiawan, D.; Zarkasi, A. Automatic features extraction using autoencoder in intrusion detection system. In Proceedings of the 2018 International Conference on Electrical Engineering and Computer Science (ICECOS), Pangkal, Indonesia, 2–4 October 2018; pp. 219–224.
37. Song, X.; Yang, F.; Wang, D.; Tsui, K.L. Combined CNN-LSTM network for state-of-charge estimation of lithium-ion batteries. *IEEE Access* **2019**, *7*, 88894–88902. [CrossRef]
38. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
39. Lei, Y.; Li, N.; Guo, L.; Li, N.; Yan, T.; Lin, J. Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mech. Syst. Signal Process.* **2018**, *104*, 799–834. [CrossRef]
40. Li, N.; Lei, Y.; Lin, J.; Ding, S.X. An improved exponential model for predicting remaining useful life of rolling element bearings. *IEEE Trans. Ind. Electron.* **2015**, *62*, 7762–7773. [CrossRef]
41. Nussbaumer, H.J.; Nussbaumer, H.J. *The Fast Fourier Transform*; Springer: Berlin/Heidelberg, Germany, 1982.
42. Yoo, Y.; Baek, J.G. A novel image feature for the remaining useful lifetime prediction of bearings based on continuous wavelet transform and convolutional neural network. *Appl. Sci.* **2018**, *8*, 1102. [CrossRef]
43. Lin, J.; Qu, L. Feature extraction based on Morlet wavelet and its application for mechanical fault diagnosis. *J. Sound Vib.* **2000**, *234*, 135–148. [CrossRef]
44. Kiranyaz, S.; Avci, O.; Abdeljaber, O.; Ince, T.; Gabbouj, M.; Inman, D.J. 1D convolutional neural networks and applications: A survey. *Mech. Syst. Signal Process.* **2021**, *151*, 107398. [CrossRef]
45. Jin, R.; Chen, Z.; Wu, K.; Wu, M.; Li, X.; Yan, R. Bi-LSTM-based two-stream network for machine remaining useful life prediction. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 3511110. [CrossRef]
46. Zhou, H.; Yang, G.; Wang, B.; Li, X.; Wang, R.; Huang, X.; Wu, H.; Wang, X.V. An attention-based deep learning approach for inertial motion recognition and estimation in human-robot collaboration. *J. Manuf. Syst.* **2023**, *67*, 97–110. [CrossRef]
47. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
48. Rasamoelina, A.D.; Adjailia, F.; Sinčák, P. A review of activation function for artificial neural network. In Proceedings of the 2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI), Herlany, Slovakia, 23–25 January 2020; pp. 281–286.
49. Lin, M.; Chen, Q.; Yan, S. Network in network. *arXiv* **2013**, arXiv:1312.4400.
50. Rengasamy, D.; Rothwell, B.; Figueredo, G.P. Asymmetric loss functions for deep learning early predictions of remaining useful life in aerospace gas turbine engines. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–7.
51. Zhang, Z.; Sabuncu, M. Generalized cross entropy loss for training deep neural networks with noisy labels. In Proceedings of the Advances in Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; Volume 31.
52. Zhang, Y.; Zhao, Y.F. Hybrid sparse convolutional neural networks for predicting manufacturability of visual defects of laser powder bed fusion processes. *J. Manuf. Syst.* **2022**, *62*, 835–845. [CrossRef]
53. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
54. Povey, D.; Hadian, H.; Ghahremani, P.; Li, K.; Khudanpur, S. A time-restricted self-attention layer for ASR. In Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, 15–20 April 2018; pp. 5874–5878.
55. Anguita, D.; Ghelardoni, L.; Ghio, A.; Oneto, L.; Ridella, S. The'K'in K-fold Cross Validation. In Proceedings of the ESANN, Bruges, Belgium, 25–27 April 2012; Volume 102, pp. 441–446.

56. Hu, G.; Xu, L.; Gao, B.; Chang, L.; Zhong, Y. Robust unscented Kalman filter based decentralized multi-sensor information fusion for INS/GNSS/CNS integration in hypersonic vehicle navigation. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 8504011. [CrossRef]
57. McDonald, G.C. Ridge regression. *Wiley Interdiscip. Rev. Comput. Stat.* **2009**, *1*, 93–100. [CrossRef]
58. Segal, M.R. Machine Learning Benchmarks and Random Forest Regression. UCSF: Center for Bioinformatics and Molecular Biostatistics. 2004. Available online: https://escholarship.org/uc/item/35x3v9t4 (accessed on 1 March 2024).
59. Zhang, X.; Yan, C.; Gao, C.; Malin, B.A.; Chen, Y. Predicting missing values in medical data via XGBoost regression. *J. Healthc. Informatics Res.* **2020**, *4*, 383–394. [CrossRef]
60. Lipovetsky, S.; Conklin, M. Analysis of regression in game theory approach. *Appl. Stoch. Model. Bus. Ind.* **2001**, *17*, 319–330. [CrossRef]
61. Peebles Jr, P.Z. *Probability, Random Variables, and Random Signal Principles*; McGraw-Hill: New York, NY, USA, 2001.