공학박사학위논문

회전기계 내 저해상도 및 고해상도 신호를 활용한 딥러닝 기반 거시적 및 미시적 고장 진단 방법론

Deep-learning-based Methodology for Macro- and Micro-level Fault Diagnosis of Rotating Machinery Using Low- and High-resolution Signals

2023 년 2 월

서울대학교 대학원

기계항공공학부

고 진 욱

회전기계 내 저해상도 및 고해상도 신호를 활용한 딥러닝 기반 거시적 및 미시적 고장 진단 방법론

Deep-learning-based Methodology for Macro- and Micro-level Fault Diagnosis of Rotating Machinery Using Low- and High-resolution Signals

지도교수 윤 병 동

이 논문을 공학박사 학위논문으로 제출함

2022 년 10 월

서울대학교 대학원

기계항공공학부

고 진 욱

고진욱의 공학박사 학위논문을 인준함

2022 년 12 월

위 원 장 : 김 윤 영 **(인)**

부위원장 : 윤 병 동 **(인)**

위 원 : 김 도 년 **(인)**

위 원 : 양 진 규 **(인)**

위 원 : 정 준 하 **(인)**

Abstract

Deep-learning-based Methodology for Macro- and Micro-level Fault Diagnosis of Rotating Machinery Using Low- and High-resolution Signals

Jin Uk Ko
Department of Mechanical and Aerospace Engineering
The Graduate School
Seoul National University

Rotating machinery is widely used in many industrial sites, including manufacturing and power generation. Unpredicted failures in these systems can result in huge economic and human losses. To prevent this situation, fault diagnosis studies have gathered much attention, with the goal of operating rotating machines without the occurrence of any unpredicted problems. Fault diagnosis methods aim to accurately detect any abnormality prior to failure and classify the health conditions of the target system. Recently, fault diagnosis studies using deep learning have achieved excellent performance thanks to the ability of new methods to autonomously extract meaningful features.

For this purpose, two types of signals of different resolutions are measured from rotating machinery, specifically: operation signals and vibration signals. Operation signals, which are measured with a low sampling rate, are obtained in real-time and contain various types of condition parameters that enable global monitoring of the system. Vibration signals with a high sampling rate are obtained when an event occurs, not in real-time. Using these signals of different resolutions, two sub-tasks of fault diagnosis – anomaly detection and fault identification – are performed. Anomaly detection, which is conducted with operation signals, is a task to detect abnormalities in a system before those abnormalities develop into a hard failure. This is considered macro-level fault diagnosis. When performing anomaly detection, the normal data is modeled by unsupervised learning, a residual is calculated, and a threshold is determined. If the residual becomes larger than the threshold, the system is regarded as an anomaly condition. Fault identification is performed to classify the health conditions of the system using vibration signals; this is viewed as micro-level fault diagnosis. For fault identification, supervised learning is used to train a deep-learning-based classifier; thus, a large amount of labeled data is required for the training. Since fault data is insufficient in real industrial fields, data augmentation is necessary to augment the fault data. Currently, a variational auto-encoder or a generative adversarial network are the approaches most widely used for data augmentation.

Anomaly detection and fault identification have been studied separately. If both tasks are integrated, macro- and micro-level fault diagnosis can be implemented. However, there are three issues that must be handled to develop a deep-learning-based methodology for macro- and micro-level fault diagnosis. First, conventional anomaly detection methods produce frequent false alarms; in other words, they may indicate a

problem even if there is no anomaly in the system. This problem occurs because conventional approaches may model the normal data inadequately or set a wrong threshold; for example, one that does not consider the fluctuations in the normal data. Second, the prior generative-network-based augmentation approach has inborn limitations due to its structural properties. With this method, signals of various lengths cannot be generated because the architecture is fixed. Also, incorrect samples can be generated if the latent vectors are sampled wrongly. The final issue with health classification is that the performance of a classifier can be affected by noise in the input data. Since noise can distort the data distribution, it is difficult for a classifier to correctly classify the noisy data.

Based on the current state of the field, this doctoral dissertation proposes a deep-learning-based methodology for macro- and micro-level fault diagnosis using operation and vibration signals from rotating machinery. The first research thrust proposes new methods for modeling and threshold setting to reduce false alarms related to anomaly detection. The proposed modeling method is developed by applying ensemble and denoising techniques to auto-encoders. Further, a threshold is newly proposed using the joint distribution of the output and the residual. Consequently, the proposed method considers the fluctuations in the normal data, which can significantly reduce false alarms. The second research thrust proposes a new generative network to generate signals of variable lengths. The proposed network, whose input and output are the time and amplitude, respectively, is designed to learn the frequency information of the training data. The proposed method is implemented to reflect the signal processing knowledge, including the use of the Nyquist theorem. After the training is finished, the proposed model can produce signals of various lengths in the desired time range. The

proposed approach can also focus on the characteristic frequency components, thanks to attention blocks. The third research thrust proposes a novel training method that simultaneously learns the classification and denoising tasks. In the proposed scheme, multi-task learning is used to allow a classifier to solve the classification and denoising tasks concurrently. The proposed method can be applied to any deep-learning algorithm, regardless of the network type. The classifier that is trained by the proposed method can classify the health conditions, as well as remove noise in the input signals.

Keywords: Macro- and micro-level fault diagnosis
Rotating machinery
Low-resolution operation signals
High-resolution vibration signals
Deep learning
Prognostics and health management (PHM)

Student Number: 2017-20541

Table of Contents

List of Tables

List of Figures

Nomenclatures

PHM	prognostics and health management
DL	deep learning
DNN	deep neural network
CNN	convolutional neural network
1D CNN	one-dimensional convolutional neural network
2D CNN	two-dimensional convolutional neural network
LSTM	long short-term memory
VAE	variational auto-encoder
GAN	generative adversarial network
t-SNE	t-distributed stochastic neighborhood estimation
P	probabilistic modeling
x	input of a neural network
y	output of a neural network
$F(\cdot)$	modeling function of a neural network
W	parameters of a model
B	batch size
η	learning rate
Adam	adaptive momentum estimation
M	the number of samples
t	threshold value
r	residual value
$\bar{r}$	mean of residuals
$\mathbf{S}$	covariance matrix of the residual
p	confidence level

AE	auto-encoder
DAE	denoising auto-encoder
EDAE	ensemble denoising auto-encoder
DT	dynamic threshold
L	loss function of a neural network
$f(\cdot)$	joint distribution of the output and the residual of the EDAE
$h(\cdot)$	marginal distribution of $f(\cdot)$
q_k	critical point of DT
$g(\cdot)$	critical function of DT
s	proposed sensitivity of DT
MSE	mean squared error
MAE	mean absolute error
RMSE	root mean squared error
α	the number of false alarms per hour
β	the number of valid alarms per hour
T_e	detection time of experts
δ	difference between the T_e and first valid alarm
EI	expected improvement
FLGN	frequency-learning generative network
FE	frequency extractor
PE	phase extractor
ME	magnitude extractor
MAFAULDA	machinery fault database
T	period of a periodic function
ω_n	n^{th} angular frequency of a function
c_n	n^{th} magnitude of a function

ϕ_n	n^{th} phase of a function
c_0	bias of a function
FC	fully connected layer
SA	sample-wise average
N_f	the number of frequency components in FLGN
p	lower bound of deterministic frequency initialization
q	upper bound of deterministic frequency initialization
ELU	exponential linear unit
$s(f)$	power spectrum of a function
f_i	deterministic frequency
Δf_i	stochastic frequency extracted by *FE*
f_s	sampling frequency
f_0	fundamental frequency
MTL	multi-task learning
MLCD	multi-task learning of classification and denoising
ODR	omnidirectional regeneration
ReLU	rectified linear unit
ε	Gaussian noise
SNR_{dB}	signal-to-noise ratio in decibel

Chapter 1

Introduction

1.1 Motivation

Rotating machinery is widely used in various industrial fields, including manufacturing and power generation. Steam turbines, motors, and wind turbines are examples of rotating machinery. Unpredicted failures in rotating machines can result in huge economic and human losses. To prevent this situation, fault diagnosis studies have gathered much recent attention, with the goal of operating rotating machines without the occurrence of any unpredicted problems [1].

There are two main types of fault diagnosis approaches: physics-based approaches and data-driven approaches. Physics-based methods diagnose a system using domain knowledge. Domain knowledge includes expertise in an industrial field and signal processing knowledge. Though this method has the advantage of an explainable rationale, the method requires significant domain knowledge and a long decision time. The other approach, the data-driven method, uses a deep-learning-based classifier that is trained using raw signals or with handcrafted features extracted using signal processing techniques or statistical analysis. Although the data-driven method can achieve better performance than the physics-based method,

it requires many labeled samples to train the classifier successfully [2]. Since approaches have recently emerged that enable big data to be measured from mechanical systems, deep-learning (DL) has been extensively researched in recent years with the goal of developing approaches for accurate fault diagnosis [3].

From the rotating machinery, two types of signals of different resolutions are measured. The first type of signals that are measured is referred to as operation signals. Operation signals contain various types of condition parameters that are measured to globally monitor the operation of a system. Temperature, pressure, and turbine speed are examples of operation signals in a steam turbine. Operation signals are measured with a low sampling rate – 1 [sample/min] or 1 [sample/sec] – because they are acquired in real-time. The second type of signals is called vibration signals. They are measured at a high sampling rate (over 5000 [Hz]) and are saved when a fault occurs, rather than being measured in real-time, which would result in a significant load on the storage device. Vibration signals can contain the dynamic characteristics of a rotating machine [4].

Deep-learning-based fault diagnosis can be subdivided into two tasks – anomaly detection and fault identification – based on the data employed. Using the operation signals, anomaly detection is conducted in real-time to detect whether there is any anomaly in the target system. Because the operation signals contain overall information and are measured in real-time, anomaly detection can be viewed as macro-level fault diagnosis. Meanwhile, fault identification is a task designed to classify the health conditions of a target system using vibration signals [5]. Fault identification is considered micro-level fault diagnosis, since the task is conducted whenever a fault occurs, and locally measured vibration signals are used. Though

both tasks can be integrated to achieve macro- and micro-level fault diagnosis, each task is typically studied separately.

In deep-learning-based anomaly detection, an unsupervised learning algorithm, like an auto-encoder (AE), is used to model the normal data [6]. The model extracts important features from the input data and reconstructs the input from the features. After the training is finished, if the abnormal data is entered into the model, the output of the model has a huge error, since the model cannot reconstruct the input data. A residual is computed as the difference between the input and the output, and a threshold is determined heuristically or statistically. If the residual surpasses the threshold, the system is considered to be in an abnormal state.

To identify fault conditions, a large amount of labeled data is necessary to train a deep-learning-based classifier. Since these networks have many trainable parameters, many labeled samples are required to optimize the parameters. However, in the data from real industrial sites, fault samples are usually so small in number as to be insufficient, as compared to many normal samples. Therefore, data augmentation is required to augment the fault samples so that the classifier is trained properly. Among many kinds of data augmentation methods, the variational auto-encoder (VAE) and generative adversarial network (GAN) approaches have emerged in popularity due to their superior generation performance. VAE, which consists of an encoder and a decoder, uses the variational inference to fit the distribution of the latent vectors as a simple distribution, like a Gaussian distribution [7]. A GAN consists of a generator and a discriminator, and both are trained adversarially [8]. The generator tries to produce a fake sample to deceive the discriminator, while the discriminator works to distinguish the fake sample. For both VAE and GAN, after

training is finished, a latent vector is sampled from a Gaussian or random uniform distribution and entered into the generator to generate new samples.

If the labeled samples are augmented enough, they are used to train a deep-learning classifier. In the classifier, a feature extractor autonomously extracts meaningful features from the input data; then, the features are used to predict the label of the input. Deep neural network (DNN), convolutional neural network (CNN) [9], and long short-term memory (LSTM) [10] approaches can be used for the extractor. When making the final output, fully connected layers are often employed. In this way, the classifier can diagnose the health states of the input data.

Though the previous fault diagnosis studies employing deep learning have shown excellent performance, three issues still exist that hinder the development of a methodology for macro- and micro-level fault diagnosis that is applicable in real industrial fields. First, the prior DL-based anomaly detection studies produce incorrect alarms, even if the system is normal; these alarms are called false alarms. Since the conventional approaches set a constant threshold that cannot consider the fluctuations in the normal data, many false alarms occur, even though there are no abnormalities in the target system. Second, VAE and GAN-based data augmentation methods have inborn limitations due to their structural properties. During the inference procedure, since the architecture is fixed, the generated signals have the same length. This means that a user cannot generate signals of various lengths at the desired time ranges. Furthermore, incorrect samples can be generated if the latent vectors are sampled wrongly. Because the physical understanding of the latent space has not yet been studied when the input samples are signals, setting a criterion for the sampling of latent vectors is difficult. The final issue is that the performance of

a classifier can be affected by the noise in the input data. Since noise can distort the data distribution, it is difficult for a classifier to correctly classify the noisy data.

To overcome the aforementioned issues, the research presented in this doctoral dissertation aims to establish a new deep-learning-based methodology for macro- and micro-level fault diagnosis of rotating machinery using operation and vibration signals. Deep-learning techniques, statistical analysis, and signal processing knowledge are integrated to develop a methodology for macro- and micro-level fault diagnosis.

1.2 Research Scope and Overview

The purpose of this doctoral dissertation is to establish a DL-based methodology for macro- and micro-level fault diagnosis of rotating machinery utilizing low- and high-resolution signals. Each research thrust is as follows: (1) Research Thrust 1 – An ensemble denoising auto-encoder-based dynamic threshold (EDAE-DT) for anomaly detection; (2) Research Thrust 2 – A frequency-learning generative network (FLGN) for data augmentation; (3) Research Thrust 3 – Multi-task learning of classification and denoising (MLCD) for health classification.

Research Thrust 1: An Ensemble Denoising Auto-encoder-based Dynamic Threshold (EDAE-DT) for Anomaly Detection

Research Thrust 1 proposes an ensemble denoising auto-encoder-based dynamic

threshold (EDAE-DT) to reduce false alarms in anomaly detection. EDAE is developed to better model the normal data, and DT is proposed to set a variable threshold that considers the fluctuation in the normal data. Combining denoising and ensemble techniques with AE, EDAE can model the normal data well. The critical hyper-parameters – the number of latent nodes and learning rate – are selected by Bayesian optimization [11] to achieve maximal performance. Five DAEs are trained using the optimized hyper-parameters, and the outputs of those DAEs are averaged to make a final output. The residual is calculated as the L1 norm of the output and the true data. When computing the DT, the joint distribution of the output of EDAE and the residual is found by kernel density estimation [12]. Then, the output values are discretized, and the marginal distributions with respect to each grid of the output are obtained. Next, the critical point for each marginal distribution is found, where the upper tail of the marginal distribution becomes the confidence level. Finally, a critical function is obtained by linearly interpolating the critical points and flattening the upper and lower tails, since the interpolation becomes incorrect in both regions. This critical function determines the threshold value according to the output value. The proposed approach is verified with two datasets from a domestic thermal power plant. The results indicate that EDAE models the normal data better than AE and DAE. Also, the proposed scheme can detect anomalies faster than the experts, while significantly reducing false alarms.

Research Thrust 2: A Frequency-learning Generative Network (FLGN) for Data Augmentation

Research Thrust 2 proposes a frequency-learning generative network (FLGN) to generate signals of variable lengths. FLGN is a new generative model, which is completely different from VAE and GAN. The input is the time vector, and the output is the amplitude vector at the corresponding time. FLGN consists of three extractors: a stochastic frequency extractor, a phase extractor, and a magnitude extractor. These extractors are composed of several fully connected blocks, a sample-wise average layer, and an attention layer. In addition to the extractors, deterministic frequencies are learned in the form of trainable parameters; they are fixed if the model is trained. The summation of the deterministic and the stochastic frequencies becomes the final frequency. A sine-basis is built based on the final frequency and the phase feature. Given the sine-basis as the input, the magnitude extractor outputs a magnitude feature corresponding to the basis. A bias is added to the inner product of the magnitude feature and the sine-basis, which becomes the final output of the FLGN. Using the deterministic frequencies and the three extractors, the proposed approach can learn the frequency components of the training data. The proposed model is validated with one simulated signal and two testbed signals. The validation results indicate that the proposed method not only produces the signals for the desired time range but also learns the frequency information well. Of particular note, it is also discovered that the proposed model can focus on the characteristic frequency components thanks to the attention blocks.

Research Thrust 3: Multi-task Learning of Classification and Denoising (MLCD) for Health Classification

Research Thrust 3 proposes a novel training method called multi-task learning of classification and denoising (MLCD) to improve the generalization performance against noisy data. The main idea of MLCD is multi-task learning (MTL), which enables a classifier that can solve the primary task and auxiliary tasks simultaneously. Solving the auxiliary tasks prevents the classifier from being biased toward the primary task, which can lead to improved performance of the primary task. In this work, classification is the primary task, and denoising is the auxiliary task. For the denoising task, the MLCD-applied classifier is trained to output the clean data, given noisy data. Another advantage of the proposed approach is that it can be applied to any classifier regardless of the network type. The proposed method is applied to one-dimensional CNN (1D CNN) and LSTM and validated with the RK4 testbed dataset. The validation results present that MLCD-applied models can improve the classification performance and reduce the uncertainty in the output, as compared to the 1D CNN and LSTM models, respectively. The t-SNE (t-distributed stochastic neighbor embedding) visualization results show that the features of MLCD-1D CNN and MLCD-LSTM are better clustered for the same class and distinguished for different classes. When visualizing the features at the intermediate layers, although most features extracted by 1D CNN and LSTM overlapped, the MLCD-applied models extract more various and meaningful features than those of the comparative models. Specifically, some features of the MLCD-applied models are similar to the waveform of the input samples. This indicates that the proposed method can make a classifier learn the characteristics of the signals, while removing noise from the signals; this is achieved by learning both the denoising task and classification task together.

1.3 Dissertation Layout

This doctoral dissertation is organized as follows. Chapter 2 offers the theoretical background required to understand each research thrust. Chapter 3 explains the ensemble denoising auto-encoder-based dynamic threshold (EDAE-DT) approach that is proposed to reduce false alarms in anomaly detection. Chapter 4 describes the frequency-learning generative network (FLGN) that is able to generate signals of variable lengths. Chapter 5 presents a new learning scheme called multi-task learning of classification and denoising (MLCD) to make a classifier robust against noisy data. Finally, Chapter 6 concludes this doctoral dissertation with a summary of the contributions and suggestions for future research.

Sections of this chapter have been published or submitted as the following journal articles:

1) **Jin Uk Ko**, Kyumin Na, Joon-Seok Oh, Jaedong Kim, and Byeng D, Youn, "A new auto-encoder-based dynamic threshold to reduce false alarm rate for anomaly detection of steam turbines," *Expert Systems with Applications*, Vol. 189, pp. 116094, 2022.
2) **Jin Uk Ko**, Jinwook Lee, Taehun Kim, Yong Chae Kim, and Byeng D. Youn, "Frequency-learning generative network (FLGN) to generate vibration signals of variable lengths," *Expert Systems with Applications*, 2022
3) **Jin Uk Ko**, Joon Ha Jung, Myungyon Kim, Hyeon Bae Kong, Jinwook Lee, and Byeng D, Youn, "Multi-task learning of classification and denoising (MLCD) for noise-robust rotor system diagnosis," *Computers in Industry*, Vol. 125, pp. 103385, 2021.

Chapter 2

Technical Background and Literature Review

This chapter offers theoretical background and a comprehensive study of macro- and micro-level fault diagnosis approaches of rotating machinery using signals of different resolutions. First, Section 2.1 explains the fault diagnosis scheme to monitor the health conditions of rotating machinery. Physics-based and data-driven approaches are explained in detail. Characteristics of low and high-resolution signals from a rotating machine are offered in Section 2.2. Section 2.3 reviews the deep-learning algorithms that are used in this work. Then, Section 2.4 provides the theoretical background for each thrust are described. In particular, anomaly detection based on deep learning is presented in Section 2.4.1. The concept of data augmentation is provided in Section 2.4.2. Next, Section 2.4.3 introduces the concept of health classification using deep learning. Lastly, a summary and discussion of this chapter are provided in Section 2.5.

2.1 Fault Diagnosis Methods of Rotating Machinery

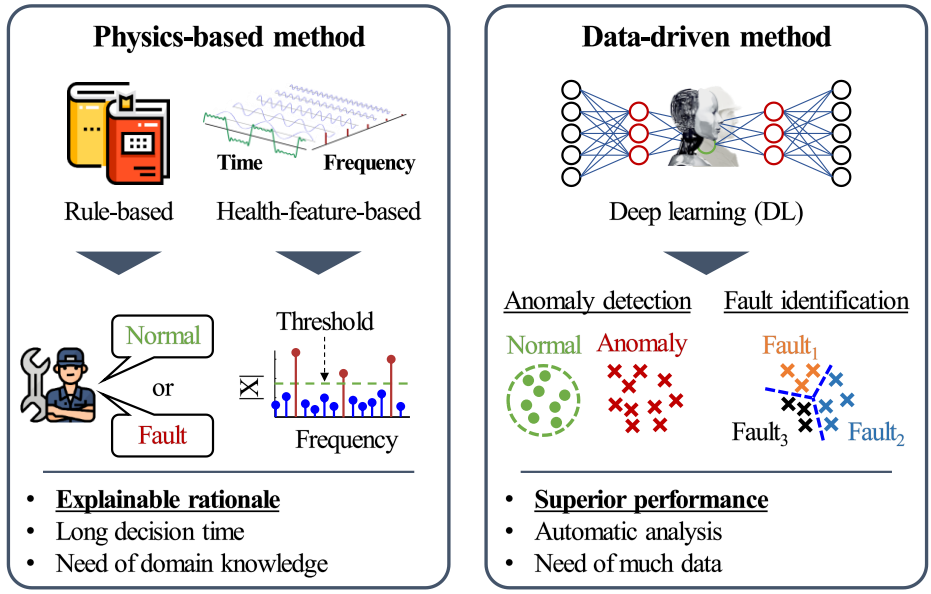
Rotating machinery, which is composed of a shaft and bearings that support the shaft,

transfers fluid or electrical energy into mechanical energy and vice versa [4]. Many rotating machines are used in various industrial sites; for example, steam turbines, gas turbines, and wind turbines are used in power generation. Since these mechanical systems operate under harsh conditions, many faults can occur in the systems. These faults can lead to severe failure of the systems, which can cause catastrophic disasters. To prevent unpredicted accidents, prognostics and health management (PHM) has been fervently studied to develop a comprehensive scheme to monitor the health conditions of the target system [1]. Specifically, PHM is a comprehensive technique that 1) recognizes whether the monitored condition parameters deviate from the normal state, 2) diagnoses the health conditions, and 3) predicts the remaining useful life or risk of failure [5]. The main effects that can be achieved by PHM are described in Figure 2-1. PHM can improve the quality of the product, availability, and productivity, ensure reliability and safety, and reduce operation and maintenance costs. Consequently, PHM techniques should be applied to rotating machinery to operate the systems safely and cost-effectively.

Figure 2-1 Purposes of prognostics and health management (PHM)

The main objective of this doctoral dissertation is to establish a methodology for macro- and micro-level fault diagnosis. Figure 2-2 illustrates physics-based and data-driven approaches of fault diagnosis. There are rule-based and health-feature-based approaches in the physics-based approach. The rule-based method uses domain knowledge, and experts determine whether or not the system is normal. In the health-feature-based approach, vibration signals are examined by using signal processing techniques and statistical analysis, and health features are extracted manually; finally, the health condition is predicted by analyzing the health features. For example, the condition of a steam turbine is diagnosed as a rubbing condition if the second sub-harmonic frequency component becomes greater. The physics-based approach is explainable, since it is based on physical or domain rationale. However, it usually takes lots of human resources and time for the final decision and heavily depends on the domain knowledge.

The data-driven method uses deep-learning algorithms for fault diagnosis. A classifier can autonomously extract features from the input data. A deep-learning algorithm is trained with training data to solve the assigned task; thus, lots of samples are necessary for the training. If there are not enough samples, the algorithm cannot be trained properly so that the diagnostic performance is decreased significantly. The data-driven approach can work automatically and shows superior performance across various engineered systems, but it requires much data for the training. Because of the advantages of deep-learning algorithms, fault diagnosis methods utilizing deep-learning techniques have been examined for diverse engineered systems [13].



(a) (b)

Figure 2-2 Types of fault diagnosis methods: (a) physics-based method and (b) data-driven method

2.2 Low- and High-resolution Signals from Rotating Machinery

Two signals of different resolutions are measured from a rotating machine: 1) operation signals and 2) vibration signals. Figure 2-3 describes those signals from a steam turbine. Operation signals are multi-variate signals that are relevant to the operation of a rotating machine [14]. Various condition parameters are measured to monitor the health states of the system; for example, turbine speed, steam temperature, steam pressure, and metal temperature are included. The sampling rate is very low – 1/60 [Hz] or 1 [Hz] – which means that operation signals have low resolution. They are measured in real-time to continuously monitor health conditions.

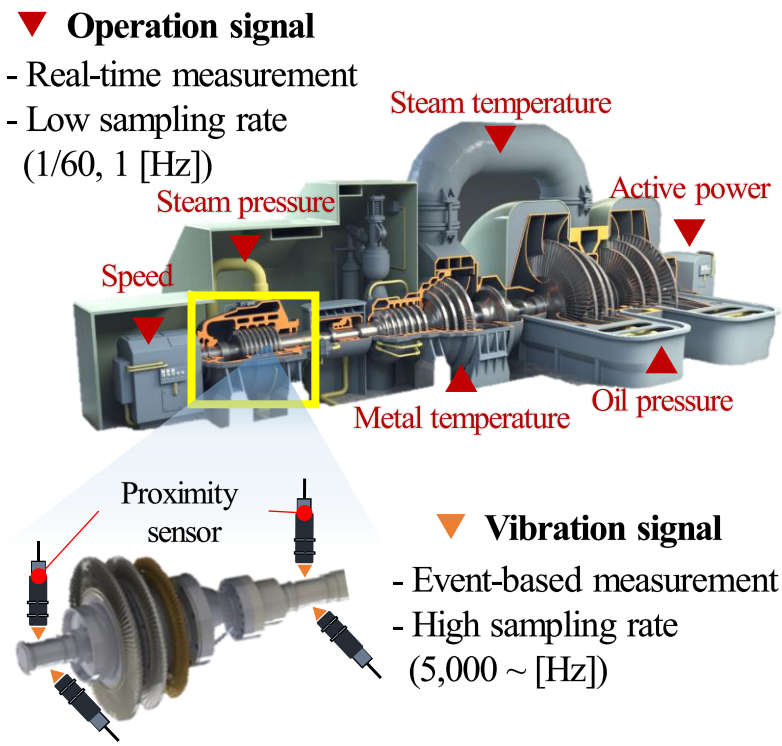


Figure 2-3 Low- and high-resolution signals from rotating machinery

Therefore, operation signals are macroscopic data that contain general information about the system.

The other signals that are obtained from rotating machinery are vibration signals. Vibration signals have been widely used to analyze the condition of a rotating machine since they represent the dynamic characteristics of the system [1]. Proximity sensors or accelerometers are utilized to measure the vibration signals. At each installation point, two sensors located at 90-degree intervals are used to measure two vibration signals that are orthogonal to each other [15]. This is to contain

information about asymmetric and anisotropic characteristics of the system. They are discrete signals, which are measured at a high sampling rate; 8500 [Hz] or 12800 [Hz]. According to the Nyquist theorem [16], discrete signals can be perfectly reconstructed into continuous signals if the sampling frequency is twice as large as the frequency bandwidth. Thus, if the sampling rate becomes greater, the frequency resolution becomes finer, which denotes that the vibration signals contain enough frequency information; however, more load is placed on data storage devices. The sensors are installed whenever a fault occurs, not in real-time, because of the heavy load on the storage devices. Also, they are installed locally, rather than globally, because those sensors require additional cost and space to install. Consequently, the vibration signals can be viewed as microscopic data.

To sum up, operation and vibration signals have three major differences. First, the range of sampling rate is different; operation signals are low resolution, and vibration signals are high resolution. Second, operation signals consist of various condition parameters, but vibration signals only have signals that are related to vibration. Finally, although operations signals are measured in real-time, vibration signals are obtained when an anomaly is detected, not in real-time.

2.3 Review of Deep-Learning Algorithms

As mentioned in Section Figure 2-2, deep learning has been extensively utilized for fault diagnosis of rotating machines. Among various deep-learning algorithms, one-dimensional convolutional neural network (1D CNN) and long short-term memory (LSTM) have achieved outstanding performance. Section 2.3.1 provides the concept

of 1D CNN, and Section 2.3.2 presents a detailed explanation of LSTM.

2.3.1 One-dimensional Convolutional Neural Network (1D CNN)

Convolutional neural network (CNN) is a popular deep-learning algorithm; it is primarily utilized for image recognition. Three characteristics of CNN – sparse connectivity, parameter sharing, and pooling – distinguish CNN from DNN [13]. The convolutional layers are locally connected rather than fully connected (sparse connectivity). For each filter, the weight is the same across all of the sparse connections (parameter sharing); this significantly reduces the number of trainable parameters. Pooling is a subsampling layer that provides a statistical summary of the input, remaining only the core information.

Thanks to these properties, CNN has been broadly used in fault diagnosis studies. When an image is input, the height of the filter for 2D CNN is smaller than the height of the input because pixels are usually correlated locally; the filters move in two-dimensional directions, as shown in Figure 2-4(a). Unlike the case of an

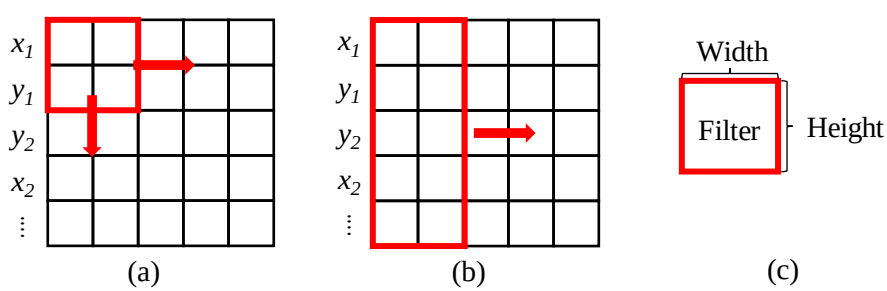


Figure 2-4 Moving of filters in CNN: (a) 2D CNN, (b) 1D CNN, and (c) shape of a filter

image, the multi-channel signals of a rotor system may be correlated with each other, rather than correlated locally. Therefore, it is more suitable to make the height of the filter the same as the input dimension and let the filter move in a one-dimensional direction, as described in Figure 2-4(b). This is a one-dimensional convolutional neural network (1D CNN). That is, 1D CNN has the properties of a CNN and can learn representation from the multi-channel signals that are correlated widely.

2.3.2 Long Short-term Memory (LSTM)

LSTM is a variant of recurrent neural networks; LSTM learns long-time-sequence patterns by preventing gradient exploding and vanishing problems [14]. LSTM consists of three gates and two states. Figure 2-5 describes the structure of an LSTM cell whose input and output at time step t are x_t and y_t, respectively. Eqs. (2.1), (2.2) and (2.3) express three sigmoid gates: a forget gate (f_t), an input gate (i_t), and an output gate (o_t). Eqs. (2.4), (2.5) and (2.6) define an output gate, a cell state (c_t) and

$$f_t = \sigma\left(W_{xf}^T x_t + W_{hf}^T h_{t-1} + b_f\right), W_{xf} \in \mathbb{R}^{n_{l-1} \times n_l}, W_{hf} \in \mathbb{R}^{n_l \times n_l}, b_f \in \mathbb{R}^{n_l} \quad (2.1)$$

$$i_t = \sigma\left(W_{xi}^T x_t + W_{hi}^T h_{t-1} + b_i\right), W_{xi} \in \mathbb{R}^{n_{l-1} \times n_l}, W_{hi} \in \mathbb{R}^{n_l \times n_l}, b_i \in \mathbb{R}^{n_l} \quad (2.2)$$

$$o_t = \sigma\left(W_{xo}^T x_t + W_{ho}^T h_{t-1} + b_o\right), W_{xo} \in \mathbb{R}^{n_{l-1} \times n_l}, W_{ho} \in \mathbb{R}^{n_l \times n_l}, b_o \in \mathbb{R}^{n_l} \quad (2.3)$$

$$g_t = \tanh\left(W_{xg}^T x_t + W_{hg}^T h_{t-1} + b_g\right), W_{xg} \in \mathbb{R}^{n_{l-1} \times n_l}, W_{hg} \in \mathbb{R}^{n_l \times n_l}, b_g \in \mathbb{R}^{n_l} \quad (2.4)$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes g_t \quad (2.5)$$

$$y_t = h_t = o_t \otimes \tanh\left(c_t\right) \quad (2.6)$$

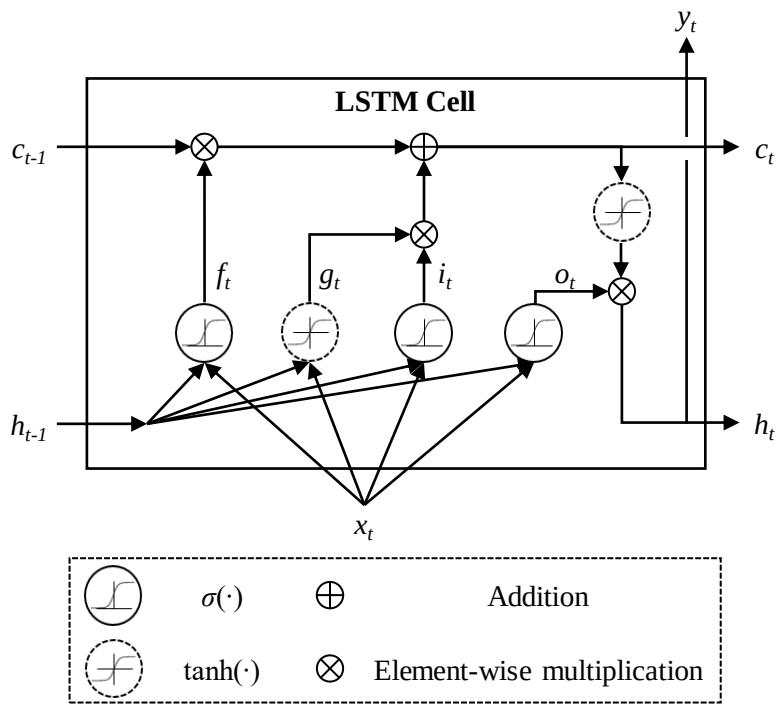


Figure 2-5 Structure of an LSTM cell

a hidden state (h_t), respectively. In the equations, $\sigma(\cdot)$ and $\tanh(\cdot)$ refer to the sigmoid and hyperbolic tangent function, respectively; n_l denotes the number of nodes at layer l. A detailed description of the gates and the cell states follows. The forget gate regulates the level of information from the previous cell state to remain. The input gate controls how much information from the input will be used; the output gate determines how much information from the previous cell state will be used for the next time step. Finally, as presented in Eq. (2.5), the current cell state is the summation of the previous cell state multiplied by the forget gate and current information from the input multiplied by the input gate. In this way, the cell state

conveys important information from the past and from the current input in each update.

2.4 Deep-learning-based Macro- and Micro-level Fault Diagnosis Methods

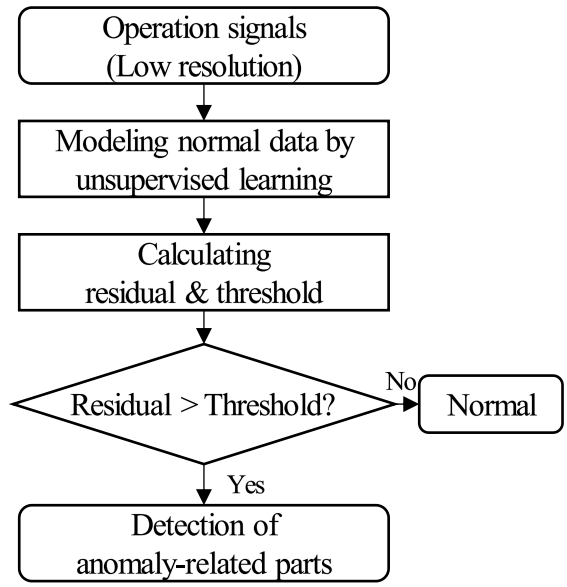
Deep-learning-based fault diagnosis is executed using low- and high-resolution signals. Fault diagnosis can be subdivided into two tasks: anomaly detection and fault identification. Anomaly and fault look similar, but they are distinctly different from each other. Anomaly means that a system deviates from its normal condition because of various reasons, including sensor error, environmental disturbance, and the occurrence of a fault. The fault is a condition that a mechanical defect occurs in the system; the change includes mechanical looseness, contact with other materials, and the occurrence of cracks. Due to the physical change, the damping coefficient or stiffness of the system may change, which leads to a change in the vibration signals. The general procedure of both tasks is described in Figure 2-6. Anomaly detection is a task that detects an abnormal change in the target system. Operation signals are used for anomaly detection to observe any unusual change in the entire system. The flowchart of deep-learning-based anomaly detection is shown in Figure 2-6(a). Using the operation signals, an unsupervised learning algorithm like an auto-encoder is used to learn the characteristics of normal data; only normal data is required to train the modeling algorithm. Then, the residual is calculated, and the threshold is determined heuristically or statistically. If the residual exceeds the threshold, it is considered that there is an anomaly in the system. Finally, among the various operation signals, the signals that are related to the anomaly are identified. A detailed

explanation of anomaly detection using deep learning is described in Section 2.4.1. Anomaly detection is difficult to be performed with vibration signals of high-resolution since 1) vibration signals are measured locally and 2) they are not measured in real-time. Anomaly detection can be considered as macro-level fault diagnosis because it is performed with macroscopic operation signals.

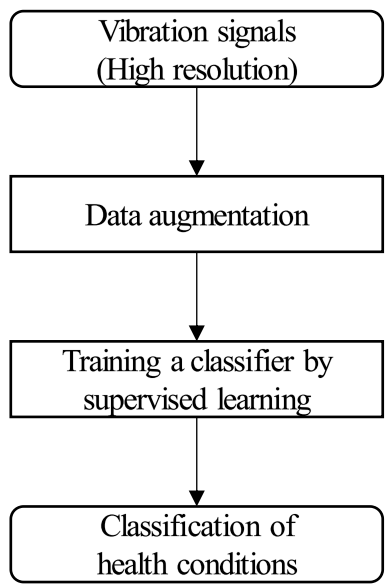
Fault identification is performed to classify the health conditions of the system. The common procedure of fault identification is presented in Figure 2-6(b). First, vibration signals are obtained with a high sampling frequency. Other than the normal data, signals of various fault conditions are required. Since fault samples are usually insufficient compared to normal samples in real industrial fields, data augmentation is necessary to augment the minor samples; the data augmentation is reviewed in Section 2.4.2. After data augmentation, a supervised learning algorithm like DNN or CNN is trained to diagnose the health conditions. The detailed contents of deep-learning-based fault identification are presented in Section 2.4.3. It is hard to conduct fault identification using operation signals of low resolution because operation signals cannot contain information about the change in the dynamic characteristics. Fault identification can be viewed as micro-level fault diagnosis since it is conducted by using vibration signals that are locally measured.

There have been few attempts to connect anomaly detection and fault identification; they are studied separately. The reason that both tasks are individually studied is that both tasks are conducted with different types of signals. If both tasks are integrated, the target system can be more thoroughly managed with a combination of macro- and micro-level fault diagnosis than with single-level diagnosis. Figure 2-7 describes the proposed methodology for macro- and micro-

level fault diagnosis in this doctoral dissertation. Through research thrust 1, an anomaly-related part is identified; this is macro-level fault diagnosis. Then, vibration signals near the abnormal part are measured. Next, if fault data is insufficient compared to the normal data, research thrust 2 augments the minor fault samples. Finally, with the augmented data, the health conditions are classified by research thrust 3; this is the micro-level diagnosis. In this way, the proposed methodology achieves macro- and micro-level fault diagnosis of a rotating machine.



(a)



(b)

Figure 2-6 Fault diagnosis schemes using low- and high-resolution signals: (a) anomaly detection and (b) fault identification

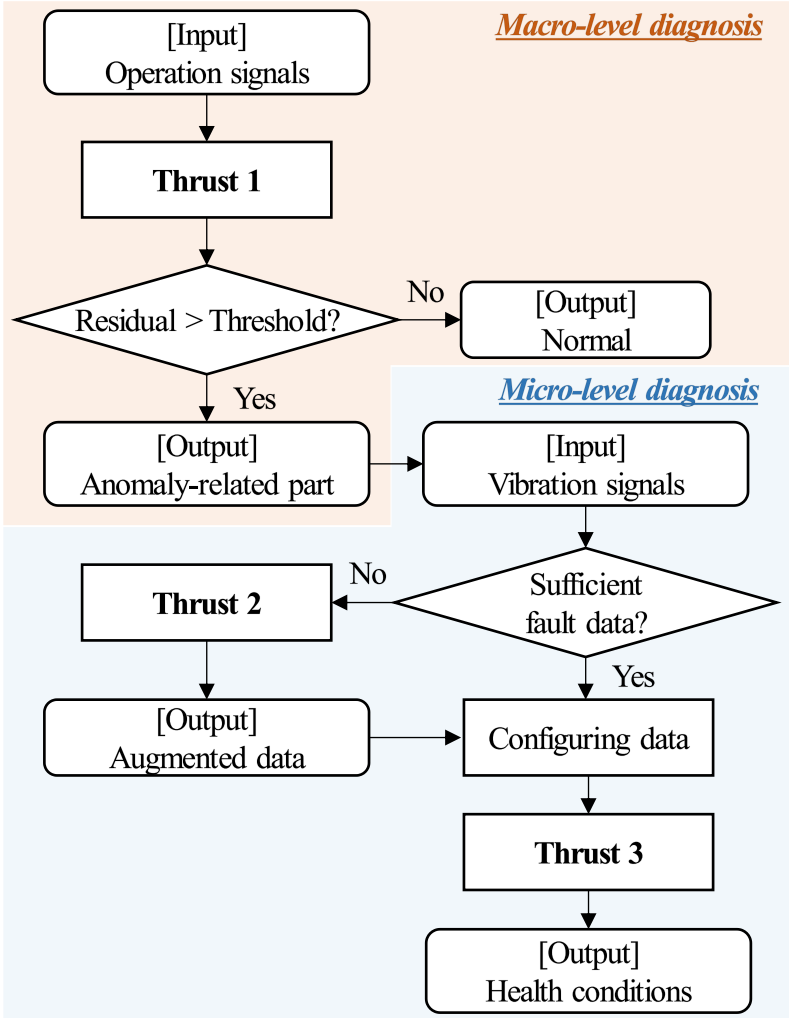


Figure 2-7 Proposed deep-learning-based methodology for macro- and micro-level fault diagnosis

2.4.1 Anomaly Detection

Deep-learning-based anomaly detection consists of two major steps: 1) modeling of normal data and 2) setting a threshold that becomes a criterion to judge whether or

not the target system is operating normally [17]. The first step is to learn the characteristics of the normal data using an unsupervised algorithm, like an auto-encoder. By training an auto-encoder with a bottleneck layer to reconstruct the input, the auto-encoder can learn the essential information of the input signals [17, 18]. If a model is well-trained with normal data, the output has little error with normal input data; however, there will be significant errors in the output if the input is abnormal data.

The general procedure of deep-learning-based anomaly detection is illustrated in Figure 2-8. It is composed of a training step and a testing step. In the training step, multi-variate operating signals of the normal condition are measured first. Preprocessing is needed because raw signals are improper to be directly used for a deep-learning algorithm. Preprocessing methods include filling in missing values, removing outliers, treating noise, etc. Next, hyper-parameters of an auto-encoder are selected, either heuristically or by grid search [19], random search [20], or Bayesian optimization [21]. Critical hyper-parameters, such as the learning rate, should be chosen carefully to ensure the maximal performance of a deep-learning algorithm. Then, by using the preprocessed normal data and the chosen hyper-parameters, an auto-encoder is trained by minimizing the objective function, such as mean squared error or mean absolute error. The trained auto-encoder can model the characteristics of a normal condition. Finally, a threshold is determined by using the residual, which is calculated as the L1 norm or L2 norm of the output and true data. For example, given a confidence level (p), a threshold can be set as the value where the cumulative distribution function of a residual becomes ($1 - p$). If the threshold is determined too sensitively, false alarms can frequently occur even though there is no abnormality in

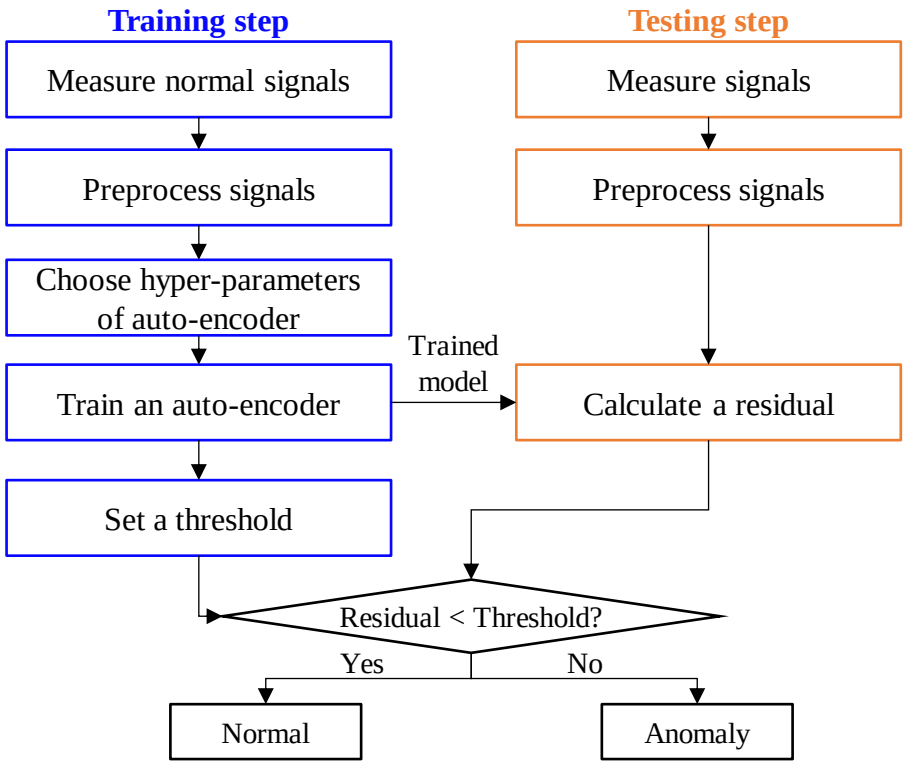


Figure 2-8 General procedure of deep-learning-based anomaly detection

the system. If it is set too conservatively, few alarms might be generated, even when an anomaly occurs. In the testing step, new signals are measured; they can be from either a normal or anomaly condition. Those signals are preprocessed using the same methods as in the training step. Then, a residual is calculated using the trained auto-encoder. Finally, the condition of the system is monitored by comparing the residual and the threshold.

Some prior studies related to deep-learning-based anomaly detection for turbines have focused on the modeling performance by developing a deep-learning

model. Arranz et al. [18] proposed a neural network of a single layer with a sigmoid function to characterize the normal data of a combined-cycle gas turbine plant. Several models have been trained to detect the condition. For example, Dhini et al. [22] developed a multilayer perceptron of a sigmoid function for anomaly detection of a steam turbine. The objective function was mean squared error (MSE), and the weights were trained by a back-propagation method. Liu et al. [23] developed a flowchart for wind turbine anomaly detection by using k-means clustering [24], t-distributed stochastic neighborhood estimation (t-SNE) [25], and a deep neural network. Specifically, k-means clustering and t-SNE were used to extract meaningful features from wind turbine data. Lu et al. [26] proposed a stacked denoising auto-encoder to consider the noise in the input signals. The auto-encoder was trained with a greedy approach, and sparsity was constrained to the hidden layers. A convolutional auto-encoder approach was also developed by Lee et al. [27] for anomaly detection of a gas turbine. When training the model, the computational cost was decreased using the sparse connectivity in the convolutional layer and through the reduction of a feature map through the use of a max-pooling layer.

A few prior studies have concentrated on developing an accurate threshold in the field of deep-learning-based anomaly detection. In early work, an intuitive threshold, called the N-sigma rule, was defined by using the mean and standard deviation of a health index [28, 29]. When the health index was assumed to follow a Gaussian distribution, it was considered to be normal when an index existed within three times the standard deviation from the mean. Chen et al. [17] proposed a stacked denoising auto-encoder to detect anomalies in a wind turbine. A health indicator was defined as a Mahalanobis distance (MD) of the residual, and the threshold was

determined as the point where the upper tail of the indicator's cumulative distribution function became the confidence level. Zeng et al. [30] proposed a new method that combined sparse Bayesian learning and hypothesis testing. Hypothesis testing was done to determine whether or not a sample falls into a confidence interval.

Even if the prior studies have shown good anomaly detection performance, they still make false alarms frequently. This is graphically illustrated in Figure 2-9. If a threshold is chosen properly, valid alarms are raised before the hard failure. However, when a rotating machine operates under the normal state, there can be fluctuation due to environmental disturbance, etc. In this situation, it is difficult for the auto-encoder to learn the normal data well. Furthermore, the constant threshold of the conventional approaches can be determined incorrectly. Frequent occurrence of false alarms causes unnecessary maintenance, which increases downtime. Therefore, false alarms should be reduced for accurate anomaly detection.

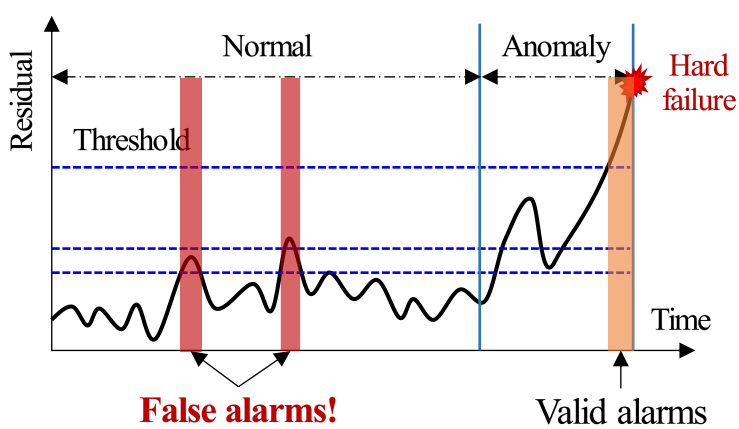


Figure 2-9 Limitation of the prior studies of anomaly detection

2.4.2 Data Augmentation

Deep-learning-based fault diagnosis studies have shown dramatic advances and promising applications in the health monitoring of various types of rotating machinery [31-37]. Unlike traditional diagnosis methods that need handcrafted features, a deep-learning classifier can autonomously learn meaningful features from input data to diagnose the health condition of the target machinery [31, 37, 38]. The deep-learning algorithms used in these methods need sufficient labeled samples to achieve high performance because they have a lot of trainable parameters. To optimize these parameters, a significant amount of data is needed in proportion to the number of parameters. However, in a real industrial facility, fault signals are scarce because engineered systems should generally operate under normal condition. If fault samples are insufficient compared to normal samples, the classifier will be biased to the majority normal condition, and minority fault conditions will not be classified well [39, 40]; this is called the class imbalance issue. There are mainly two methods to handle the issue: an algorithm-level approach and a data-level approach [39, 41]. The algorithm-level approach deals with the imbalance issue by modifying the loss function to impose more penalty on the misclassification [39, 41]. This method can be used as an end-to-end learning scheme, but it requires much knowledge about the classifier and the target data [41]. The data-level approach, rebalancing the data distribution by controlling the number of samples, can be divided into two schemes: undersampling and oversampling methods. An undersampling method rebalances the data distribution by reducing the majority class, which means that information loss is inevitable [39]. An oversampling method

is to balance the distribution by augmenting the minority class [39]. Among these methods, the oversampling method is the most versatile since little information is lost, and it does not depend on the classifier [39].

To augment the minority samples, researchers have fervently studied data augmentation approaches using generative networks; variational auto-encoder (VAE) and generative adversarial network (GAN) approaches are widely utilized. The general architecture of VAE and GAN is presented in Figure 2-10. VAE, which is composed of an encoder and a decoder, uses variational inference to fit the distribution of latent vectors as a simple distribution, like a Gaussian distribution [7]. After training the VAE, a latent vector is sampled from the distribution and entered into the decoder; then, the decoder produces a new sample corresponding to the latent vector [7]. A GAN is made up of a generator and a discriminator, which are

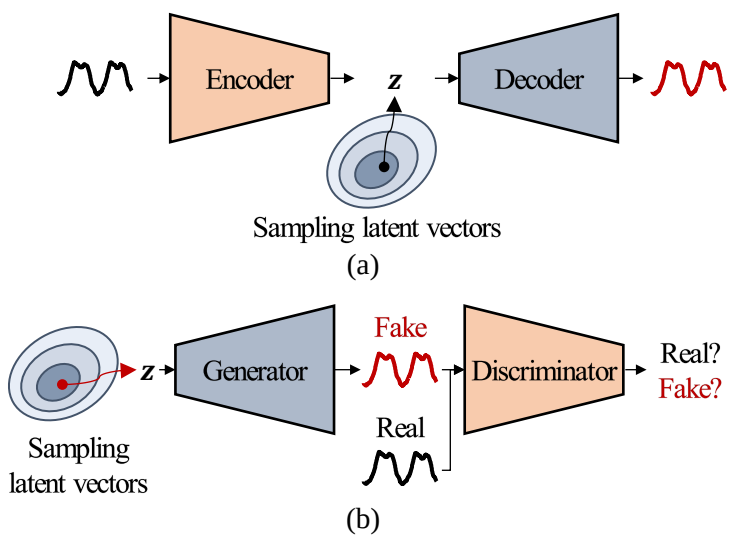


Figure 2-10 Architecture of generative networks: (a) VAE and (b) GAN

trained in an adversarial manner. When a latent vector is sampled, the generator produces a fake sample to deceive the discriminator; then, the discriminator tries to discriminate whether or not the input sample is fake [42]. Although VAE is usually more stable than GAN, GAN can generate clearer samples than VAE [43].

A small number of prior studies have examined VAE-based signal augmentation. Zhao et al. [44] proposed a VAE approach based on a 1D CNN. The encoder and the decoder were composed of several convolution and max-pooling layers. When validating the generation performance using a rolling bearing dataset, this approach generated new samples similar to the training data and enhanced the classification accuracy. Zhang et al. [45] used VAE to develop a semi-supervised fault-diagnosis scheme. The developed VAE model was composed of an encoder, an external classifier, and a decoder; the classifier predicted the label, given a latent vector. Che et al. [46] combined VAE and meta-learning for bearing fault diagnosis. That proposed VAE, which consisted of various fully connected layers, generated the minority fault signals; then, a metric-based meta-learning model was trained with those generated signals. Some studies have employed GAN to produce fault signals. A generative model based on an auxiliary classifier generative adversarial network (ACGAN) was proposed by Shao et al. [47]. The discriminator in ACGAN performed two classification tasks; the first task was to classify whether the input of the discriminator was real or not, and the other was to classify the label of the input data. Enhanced GAN was proposed to generate imbalanced vibration signals [48]. A deep convolutional generative adversarial network (DCGAN) with 1D CNN was used to construct the model. Later, Gao et al. [49] proposed an augmenting scheme based on WGAN-GP. The proposed network was more stable than both the DCGAN

and the ACGAN approaches. When tested with various classifiers, including logistic regression and random forest, the performance results of the classifiers were improved when trained by new samples produced by the proposed network. Suh et al. [50] developed a new generative network to oversample the fault data of an induction motor by using a Wasserstein generative adversarial network with gradient penalty (WGAN-GP) and DCGAN. When validating a CNN-based classifier with various imbalance ratios, the proposed approach improved the classification performance in most cases. A sparsity-constrained generative adversarial network (SC-GAN) was designed to augment the minority data [51]. In the work, a sparse auto-encoder was trained first and the encoder and the decoder became the discriminator and the generator, respectively; a fully connected layer was added at the end of the decoder. By imposing sparsity on the GAN, it could achieve more stable generation and learn the important frequency components of the input signal. Peng et al. [52] proposed WGAN with hierarchical feature matching (HFM) to produce bearing fault signals. Wasserstein loss was used to make the training procedure stable and HFM was developed so that the features of the generated signals of each condition were close to those of the true signals.

Although the previous VAE or GAN-based augmentation studies have shown outstanding generation performance, they have two limitations: 1) the lengths of the generated signals are not changeable, and 2) wrong samples can be produced if latent vectors are incorrectly sampled. Figure 2-11 describes these two limitations. First, the signals generated by the conventional models have the same length because the network architecture cannot be changed. The length of the generated signal and the input signal is the same. Longer or shorter signals cannot be generated using these

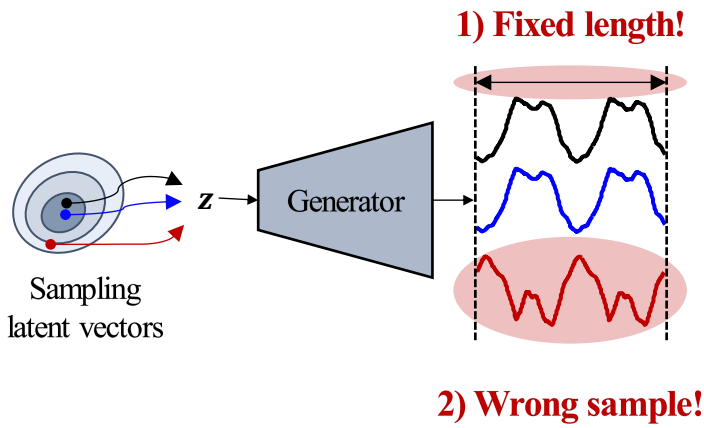


Figure 2-11 Limitations of the VAE or GAN-based models

methods. Second, incorrect samples can be produced if the latent vectors that will be entered into the generator are sampled improperly. As described in Section 2.4.2, latent vectors are sampled from a Gaussian or uniform distribution, and the generator produces a new sample, given the sampled latent vectors. When a GAN is trained with image samples, it has been found that each latent dimension is related to a visual property of an image, including rotation, thickness, etc. [53]. However, in the case of vibration signals, the physical meaning of the latent space has not been discovered yet. This makes it difficult to set up the standards for the sampling procedure to prevent generating invalid samples.

2.4.3 Health Classification

Deep-learning-based health classification studies have drawn much attention due to their high performance and automated feature extraction ability. The objective of

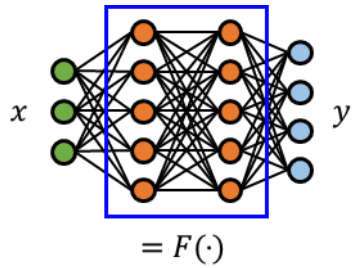


Figure 2-12 Health classification modeling

health classification is to predict the label condition of the given sample. Figure 2-12 shows the graphical expression of health classification modeling.

Let the input sample be *x*, the model be $F(\cdot)$ whose parameters are *W*, and the output class be *c*. Then, the health classification modeling becomes as follows:

$$P(y=c \mid x) = F(x;W) \tag{2.7}$$

This means that a classifier learns the probability distribution when the output class is equal to *c*, given a sample *x*. To learn the function $F(\cdot)$, the objective function should be defined first. When there are two classes, the objective function becomes binary cross-entropy. Also, in the case of more than three two classes, categorical cross-entropy is used as the objective function. Both functions are defined as follows:

$$L(y,\hat{y}) = -\frac{1}{B}\sum_{i=1}^{B} y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i) \tag{2.8}$$

$$L(y,\hat{y}) = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{C} y_{ij} \log \hat{y}_{ij} \tag{2.9}$$

where y_i is the true class, $\hat{y}$ is the output, B is the number of samples, and C is the number of classes. Using a training data, the parameters of a classifier are optimized by back-propagation [54]. After forward propagation to compute the output, the gradient of the parameters about the loss function is calculated in reverse order from the output layer to the input layer. After calculating all gradients, it is updated by the gradient descent rule. This is mathematically expressed as follows:

$$W^{k+1} = W^k - \eta \frac{\partial L}{\partial W^k} \tag{2.10}$$

where k is the iteration number and η is the learning rate. Other than the stochastic gradient descent rule, adaptive momentum estimation (Adam) has been widely used because Adam can reach the optimum point fast and stably [55].

Many prior studies have developed new deep-learning-based classifiers using various deep-learning algorithms. Oh et al. [56] suggested a rotor system diagnosis scheme by training a restricted Boltzmann machine with a proposed vibration imaging method. Wu et al. [57] and Long et al. [58] used a 1D CNN and 2D CNN to diagnose the states of rotating machinery, respectively. Also, Zhao et al. [59] developed a new fault diagnosis method for a planet bearing using a CNN. Islam et al. [60] utilized a CNN to construct a bearing classifier that extracted features automatically from the wavelet packet transformation of an acoustic emission signal. In addition, Nguyen et al. [61] and Bruin et al. [62] developed fault diagnosis models by using LSTM. They focused on the capability of LSTM, which can understand the sequential context in the time-series signals.

Despite the outstanding performance of the conventional methods, they can be

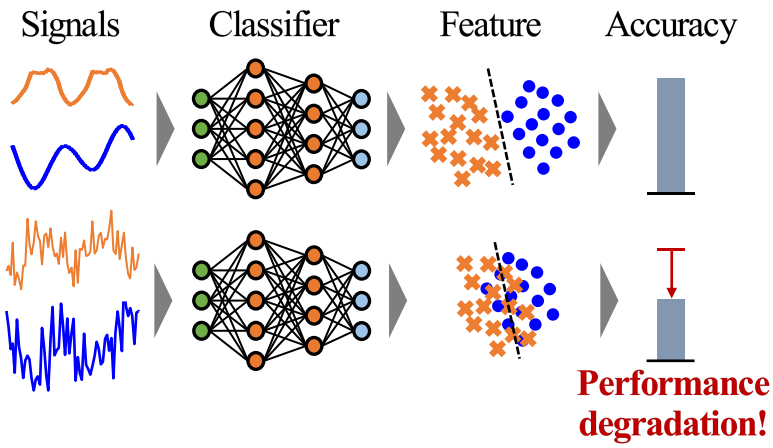


Figure 2-13 Limitation of the prior studies of health classification

affected by the noise in the input data. Figure 2-13 illustrates the noise issue in health classification. If the test data has a similar data distribution to that of the training data, the classifier can well predict the label of the test data. However, in a real industrial field, the measured signals can be corrupted by much noise due to mechanical or environmental causes. Thus, newly measured signals have different distributions due to the noise, which leads to performance degradation. Consequently, this noise issue should be addressed for noise-robust health classification.

2.5 Summary and Discussion

The objective of this doctoral dissertation is to develop a deep-learning-based methodology for macro- and micro-level fault diagnosis using signals of different resolutions. First, the concept of PHM and fault diagnosis are reviewed in Section

2.1. Since two types of signals of different resolutions are measured from a rotating machine, the characteristics of both signals are studied in Section 2.2. Section 2.3 presents the literature review of deep-learning algorithms that are utilized in this work. By using low- and high-resolution signals, anomaly detection and fault identification are conducted, respectively; Section 2.4 provides the literature review about both tasks. Section 2.4.1 explains the concept of anomaly detection and reviews the previous studies. When fault signals are insufficient compared to normal data, data augmentation is required to increase the number of fault samples; a literature review about data augmentation is presented in Section 2.4.2. Section 2.4.3 offers the theoretical background of health classification and the prior studies about it.

Sections of this chapter have been published or submitted as the following journal articles:

1) **Jin Uk Ko**, Kyumin Na, Joon-Seok Oh, Jaedong Kim, and Byeng D, Youn, "A new auto-encoder-based dynamic threshold to reduce false alarm rate for anomaly detection of steam turbines," *Expert Systems with Applications*, Vol. 189, pp. 116094, 2022.
2) **Jin Uk Ko**, Jinwook Lee, Taehun Kim, Yong Chae Kim, and Byeng D. Youn, "Frequency-learning generative network (FLGN) to generate vibration signals of variable lengths," *Expert Systems with Applications*, 2022
3) **Jin Uk Ko**, Joon Ha Jung, Myungyon Kim, Hyeon Bae Kong, Jinwook Lee, and Byeng D, Youn, "Multi-task learning of classification and denoising (MLCD) for noise-robust rotor system diagnosis," *Computers in Industry*, Vol. 125, pp. 103385, 2021.

Chapter 3

Ensemble Denoising Auto-encoder based Dynamic Threshold (EDAE-DT) for Anomaly Detection

In this chapter, an ensemble denoising auto-encoder-based dynamic threshold (EDAE-DT) is newly proposed to reduce false alarms in anomaly detection. An ensemble denoising auto-encoder (EDAE) is trained to model the normal data of a steam turbine. A deep neural network is selected as the base model of the EDAE because it is most widely used in the field of anomaly detection of engineered systems [18, 22, 26, 63]. An ensemble technique can reduce the generalization error by averaging the output of several models [64]. A denoising task can further improve the reconstruction performance by learning how to remove noise in the input [65]. After training the EDAE, the dynamic threshold (DT) calculates a variable threshold according to the output of the EDAE by computing the upper confidence limit from the joint distribution of that output and the residual. The threshold value is determined dynamically with respect to the output. After anomaly detection, to identify the part that is related to the anomaly, a new sensitivity is defined by using the maximum values of the residual and the threshold. Through this enhancement, the operators are able to investigate the specific parts related to the sensitive

condition parameters, which can reduce maintenance costs. In this research, two datasets from a thermal power plant are used to validate the proposed EDAE-DT. Each dataset consists of several operating parameters and has a sampling frequency of 1 [sample/min]. To confirm the effect of the ensemble and denoising techniques in the modeling process, the proposed EDAE approach is compared with AE and DAE methods; the AE method is used in [22, 66], and the DAE approach is used in [17, 26]. Since the performance of a deep-learning algorithm varies according to its architecture, the modeling performances of AE, DAE, and EDAE are compared with various numbers of hidden layers. Then, the anomaly detection performance of EDAE-DT is compared with previous anomaly detection methods; specifically, the N-sigma method [28, 29] and the MD-based method [17]. N-sigma is chosen for comparison because it is simple and intuitive; MD is selected since a threshold is determined statistically from a single health index extracted from multi-variate signals. For quantitative validation, several metrics are newly defined for the evaluation of anomaly detection performance. In addition, to validate the performance of the fault diagnosis, a confusion matrix is used by labeling the detected status as normal or anomaly. The validation results indicate that the proposed method detects anomalies with significantly fewer false alarms, as compared with conventional methods, while also detecting anomalies faster than experts.

The remainder of this chapter is structured as follows. Section 3.1 provides the theoretical background of the conventional deep-learning-based anomaly detection methods. Section 3.2 explains the proposed method in detail. Performance evaluation metrics are presented in Section 3.3. The validation data is provided in

Section 3.4, and Section 3.5 shows the validation results. Finally, Section 3.5.3 offers the conclusion of this study.

3.1 Background: Deep-learning-based Anomaly Detection

In this section, the conventional studies about deep-learning-based anomaly detection are reviewed. First, the studies to model the normal data are presented. Then, the prior studies to set a threshold are investigated.

3.1.1 Conventional Methods to Model the Normal Data

An auto-encoder (AE) is an unsupervised learning algorithm that is trained to reconstruct its input. It has been widely used to model the characteristics of normal data. The basic architecture of an AE is illustrated in Figure 3-1. Given multi-variate operation signals, including vibration, temperature, and pressure, they are entered into an encoder and are reconstructed in the decoder; the architectures of both parts are usually symmetric. The representation of the (l+1)-th layer becomes as follows:

$$\mathbf{a}^{l+1} = f\left(\mathbf{W}^{l}\mathbf{a}^{l} + \mathbf{b}^{l}\right) \tag{3.1}$$

where $f(\cdot)$ is the activation function; $\mathbf{a}^{l+1}$ is the output of the l-th layer; $\mathbf{W}^{l}$ and $\mathbf{b}^{l}$ are the weight and bias between the l-th layer and the (l+1)-th layer. To induce the non-linear dimensionality reduction of the input, the final layer of the encoder has smaller hidden nodes than those of the input layer [67]. In this way, the encoder extracts the essential information from the input data. Then, a decoder reconstructs the input

from the encoded information. The AE is trained by minimizing the mean squared error (MSE) or the mean absolute error (MAE). In this study, MAE is used because MAE is more sensitive to local variations of input data [68]. The mathematical expression of MAE is defined as follows:

$$L = \frac{1}{M}\sum_{i=1}^{M}\left|\mathbf{x}_i - \hat{\mathbf{x}}_i\right| \tag{3.2}$$

where $\mathbf{x}_i$ is the sample at time index i, and M is the number of samples. When an AE is successfully trained with normal data of a steam turbine, it can be said that the AE models the normal condition of the turbine.

Other than the AE, a denoising auto-encoder (DAE) is also widely employed. Its architecture is the same as AE; the difference is the input. Noise is added to the input, and the DAE predicts clean data, which is the data before the noise is added. Since the DAE has the ability to reconstruct the input data while removing the noise, the performance of the DAE is usually better than that of the AE.

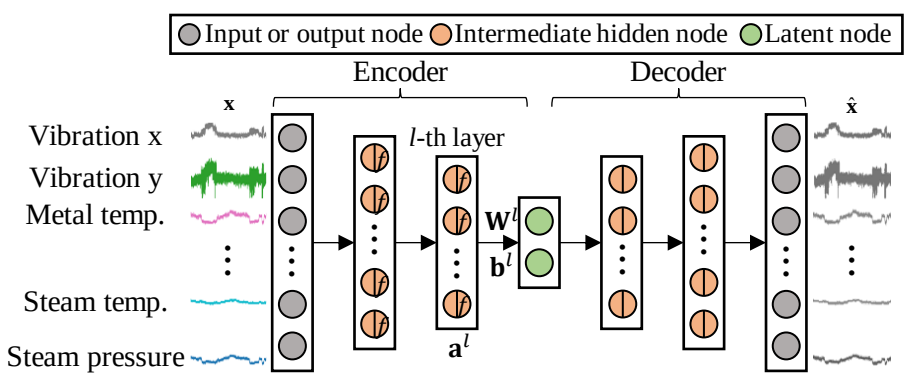


Figure 3-1 Architecture of an auto-encoder (AE)

3.1.2 Conventional Methods to Set a Threshold

Nelson's N-sigma method and the Mahalanobis-distance-based (MD) method are the most widely used methods to set a threshold. Both methods set a constant threshold; the N-sigma method is intuitive, and MD extracts a single health indicator from multi-variate signals. In the N-sigma method, mean (μ_k) and standard deviation (σ_k) are calculated from the L1 residual of the *k*-th parameter. Then, the threshold in N-sigma method is determined as below:

$$t_k = \mu_k + N \times \sigma_k \tag{3.3}$$

where *N* is selected heuristically or as the value that satisfies the confidence level (*p*).

The MD method is different from the N-sigma method. The residual of the *k*-th parameter is computed as the difference between the true and predicted output as follows:

$$r_k = y_{k,true} - y_{k,pred} \tag{3.4}$$

Then, a monitoring indicator at the *l*-th sample is calculated as follows:

$$t^l = \sqrt{\left(\mathbf{r}^l - \bar{\mathbf{r}}\right)^T \mathbf{S}^{-1} \left(\mathbf{r}^l - \bar{\mathbf{r}}\right)} \tag{3.5}$$

where $\bar{\mathbf{r}}$ and $\mathbf{S}$ are the mean vector and covariance matrix of the residual, respectively. After that, the probability distribution function of the indicator *t* is computed by kernel density estimation. Finally, the threshold is set as the point such that the cumulative distribution function of *t* becomes (1 - *p*).

3.2 Ensemble Denoising Auto-encoder-based Dynamic Threshold (EDAE-DT)

To detect anomalies of rotating machinery with fewer false alarms, an ensemble denoising auto-encoder-based dynamic threshold (EDAE-DT) is newly proposed. The overall procedure of EDAE-DT is described in Figure 3-2. Similar to the conventional anomaly detection procedure, the proposed method consists of training and testing steps. In the training procedure, operating signals of the normal condition are measured from the steam turbine. In the preprocessing step, the missing values are filled by linear interpolation of nearby values, outliers are removed by the 6-MAD (median absolute deviation) method, and moving average filtering is applied for smoothing. Finally, the signals are min-max scaled.

Next, EDAE is trained as follows. The learning rate and the number of latent nodes are selected by Bayesian optimization. Figure 3-3 illustrates the architecture of EDAE. The denoising auto-encoder (DAE) has a symmetric architecture around the latent layer. Dropout is applied to the intermediate hidden layers (orange circle), except the latent layer; the dropout rate is set as 0.1. Each model is trained with noisy input signals; they are added using different white-gaussian noise (ε) with the same signal-to-noise ratio for each model. That is, if x_n^i denotes the n-th parameter value at time index i, all clean signals (x) and corrupted signals ($\tilde{x}_i$) for the j-th model become like Eq. (3.6). Then, the loss function of the j-th model (L_j), the output of the EDAE (y^i), and the residual of j-th tag at i-th index (r_j^i) become like Eq. (3.7). In Eq. (3.7), B is the mini-batch size, $Q_j(\cdot)$ is the learned representation of model j, and M is the number of models; M is set as five in this work. r_j^i is non-negative since it is the L1 norm of the output and the target value. By learning how to denoise the input

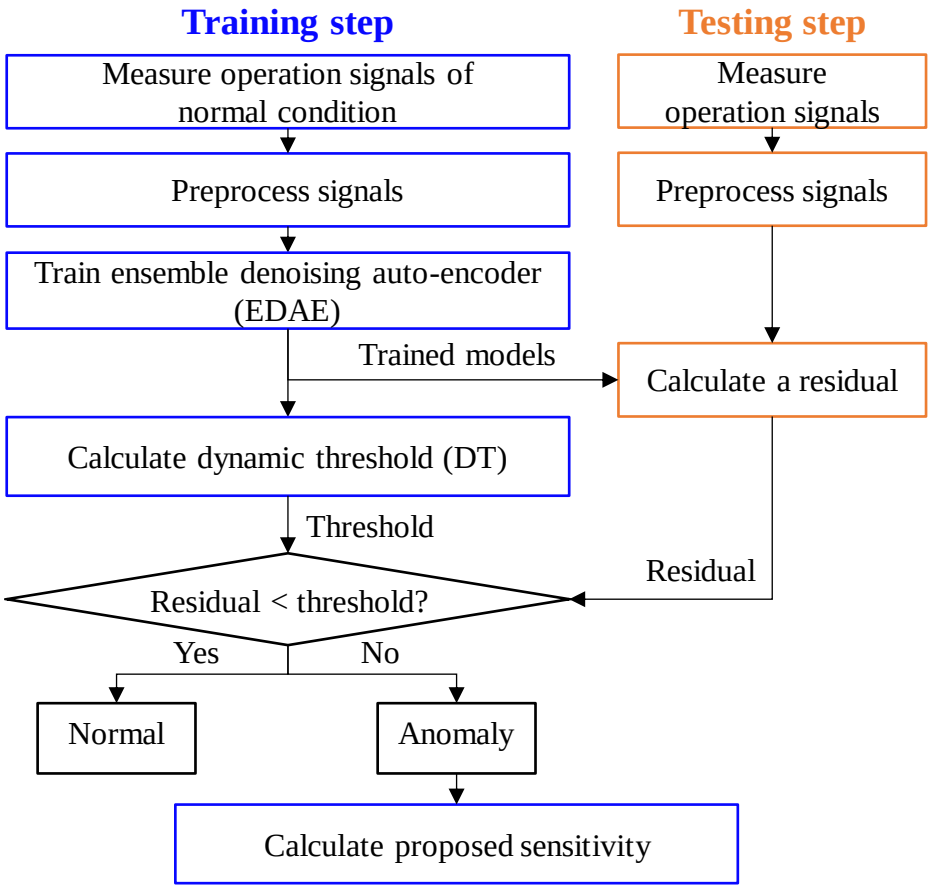


Figure 3-2 Procedure of EDAE-DT

signals, the reconstructing performance can be enhanced. Also, the ensemble technique can improve performance by reducing the uncertainty in the outputs.

$$\mathbf{x}^i = \left[x_1^i, x_2^i, ..., x_n^i \right]^T$$
$$\tilde{\mathbf{x}}_j = \mathbf{x} + \boldsymbol{\varepsilon}_j \tag{3.6}$$

$$
\begin{aligned}
L_j &= \frac{1}{B}\sum_{i=1}^{B}\left|Q_j\left(\tilde{\mathbf{x}}_j^i\right)-\mathbf{x}^i\right| \\
\mathbf{y}^i &= \frac{1}{M}\sum_{j=1}^{M}Q_j\left(\tilde{\mathbf{x}}^i\right) \\
r_j^i &= \left|y_j^i - x_j^i\right|
\end{aligned}
\tag{3.7}
$$

After training the EDAE, the dynamic threshold (DT) is computed; each step is described graphically in Figure 3-4. For each parameter, let the output of EDAE be

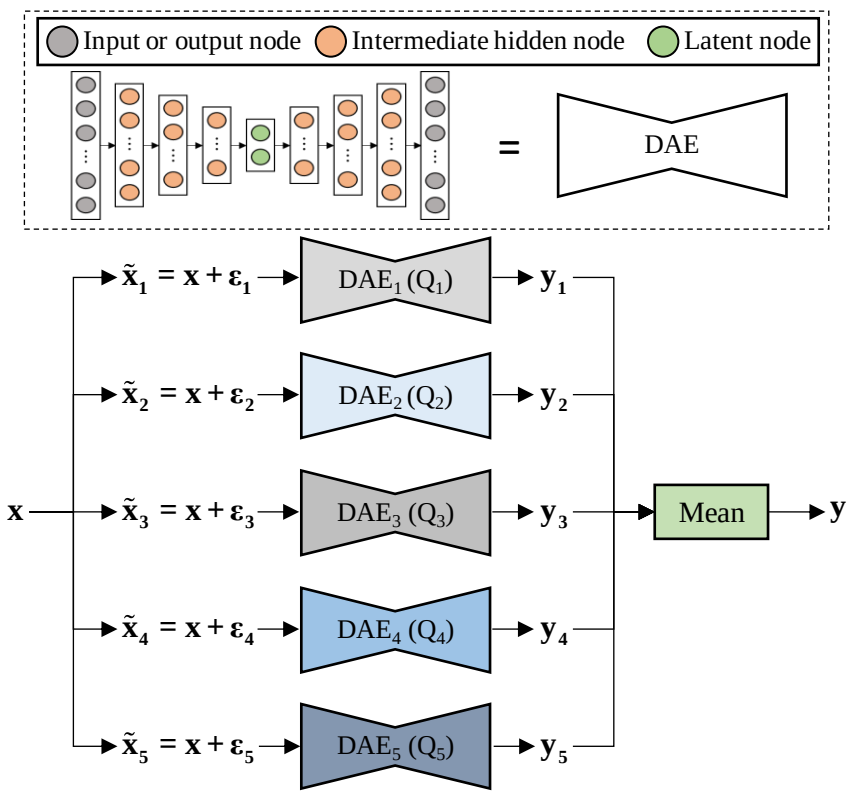


Figure 3-3 Architecture of EDAE

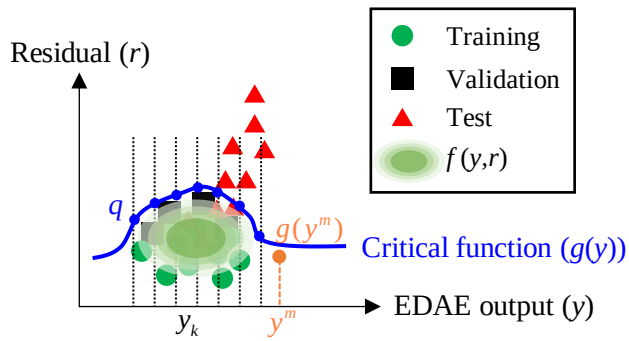


Figure 3-4 Concept of dynamic threshold (DT)

$y = [y_{tr};y_{val}]^{T,}$ and the corresponding residual be $r = [r_{tr};r_{val}]^{T}$. Both variables are obtained by Eq. (3.7); index i and j are ignored for convenience. The joint probability distribution $f(y,r)$ is obtained by kernel density estimation using a Gaussian kernel. The bandwidth is estimated by Scott's rule [69]. Then, for each y_k (black dotted line) of the regular grid size, the marginal distribution $h(\cdot)$ is obtained as shown in Eq. (3.8).

$$h\left(r; y = y_k\right) = \frac{f\left(y_k, r\right)}{\int_{-\infty}^{\infty} f\left(y_k, r\right) dr} \tag{3.8}$$

The denominator plays the role of normalization to make the integral of $h(\cdot)$ equal to 1. Next, using a pre-defined confidence level (p), the critical point (q_k) for each y_k is calculated so that q_k satisfies Eq. (3.9).

$$\int_{-\infty}^{q_k} h\left(r; y = y_k\right) dr = 1 - p \tag{3.9}$$

Those critical points are linearly interpolated, and the upper and lower tails are flattened as follows:

$$\begin{cases} g(y) = g(y_{upper}) \; if \;\; y \geq y_{upper} \\ g(y) = g(y_{lower}) \; if \;\; y \leq y_{lower} \end{cases} \tag{3.10}$$

Here, y_{lower} and y_{upper} are chosen as 0.8 times of minimum of y and 1.2 times of the maximum of y, respectively. The reason for flattening is that interpolation is usually inaccurate in those regions. Then, a critical function ($g(y)$) can be defined with respect to each parameter, which is the blue line in Figure 3-4. When y^m is the EDAE's output for unseen data, $g(y^m)$ becomes the threshold value. If the residual is greater than $g(y^m)$, the system is considered to have an anomaly, since the residual exceeds the confidence limit.

After detecting an anomaly by EDAE-DT, to find the condition parameters related to the anomaly, a sensitivity is newly defined as follows:

$$s_j = \frac{\max\left(r\left(y_{j,te}\right)\right) - \max\left(g\left(y_{j,total}\right)\right)}{\max\left(r\left(y_{j,te}\right)\right)} \tag{3.11}$$

where s_j is the sensitivity of the j-th parameter, $r(y_j)$ is the residual of y_j, $y_{j,total}$ is the union of ($y_{j,tr}$;$y_{j,val}$;$y_{j,te}$), and $g(\cdot)$ is the critical function. If s_j is positive, an alarm is produced for the parameter; it can be said that no alarm occurs if s_j is negative. Also, a greater value of s_j indicates a more sensitive parameter. This sensitivity value can give a clue to the plant operators about which parameters are relevant to the abnormality.

3.3 Performance Evaluation Metrics

As described in Section 2.4.1, deep-learning-based anomaly detection methods consist of two main steps: modeling of normal data and anomaly detection. Accordingly, two kinds of metrics are newly defined for quantitative validation; metrics for modeling performance and for anomaly detection performance. The proposed metrics are summarized in Table 3-1. RMSE (root mean squared error) of validation data is defined to quantify the modeling performance because it is the most widely used metric for regression [70, 71]. In the formulation, T is the time length of the validation data, N is the number of condition parameters; x_j^i and y_j^i are j-th parameters at time index i and the ensembled output of it, respectively. RMSE indicates how well an algorithm reconstructs the input data, which means modeling the normal data. The smaller this metric, the better the algorithm learns the normal data.

Definitions of false alarms and valid alarms are needed to define metrics for anomaly detection performance. Figure 3-5 describes the meaning of both alarms. In Figure 3-5, T_{tr}, T_{val}, and T_{te} are the time lengths of the training, validation, and test data, respectively; the black line is the residual, and the dotted blue line is the threshold. Training, validation, and test data might be continuous or not; however, they should be in the order of time, not overlapped. An alarm occurs when the residual exceeds the threshold. False alarms are alarms that arise in the training and validation periods; these are expressed in the orange regions. Valid alarms, expressed in the gray region, are alarms that occur in the test period, since a change due to an

anomaly will occur in the test period. In this context, three metrics – α, β, and δ – are newly proposed to quantify the anomaly detection performance. α and β are the numbers of false alarms and valid alarms per hour, respectively. In the formulation of Table 3-1, F and V, mathematically defined in Eq. (3.12), are sets of false alarms and valid alarms; in the equation, f_s is the sampling frequency. A small α means that anomaly detection is reliable; a large α denotes that anomaly detection is so sensitive that the operator might be confused by too many false alarms. In contrast, the larger the value of β, the better an algorithm is sensitive to the change of the system due to an anomaly. α is more crucial than β from the viewpoint of reliability only if β is greater than 0. δ is defined as the difference between the detection time of experts (T_e) and the first detection of a valid alarm (V_0). It is desirable for δ to be positive; otherwise, there is no reason to use that anomaly detection algorithm. As for the units of the metrics, RMSE has no dimension because input signals are normalized, and α, β are times per hour; the unit of δ is days.

Table 3-1 Defined evaluation metrics

Performance type	Notation	Definition	Formulation	Unit
Modeling performance	RMSE	Root mean squared error	$\sqrt{\frac{1}{TN}\sum_{i=1}^{T}\sum_{j=1}^{N}\left(y_j\left(t_i\right)-x_j\left(t_i\right)\right)^2}$	-
Anomaly detection performance	α	# of false alarms per hour	$\frac{N\left(F\right)}{T_{tr}+T_{val}}$	times / hour
	β	# of valid alarms per hour	$\frac{N\left(V\right)}{T_{te}}$	times / hour
	δ	How much faster than experts' detection	$T_e - V_0$	days

$$F=\left\{t_i \mid g\left(y_j\left(t_i\right)\right)\le r\left(y_j\left(t_i\right)\right) \text{ and } t_i\in P\right\} \text{ where } P=\left\{t_i \mid \left\{T_0\le t_i\le T_1\right\}\cup\left\{T_2\le t_i\le T_3\right\}, t_{i+1}-t_i=\frac{1}{f_s}\right\}$$
$$V=\left\{t_i \mid g\left(y_j\left(t_i\right)\right)\le r\left(y_j\left(t_i\right)\right) \text{ and } t_i\in Q\right\} \text{ where } Q=\left\{t_i \mid T_4\le t_i\le T_5, t_{i+1}-t_i=\frac{1}{f_s}\right\} \quad (3.12)$$

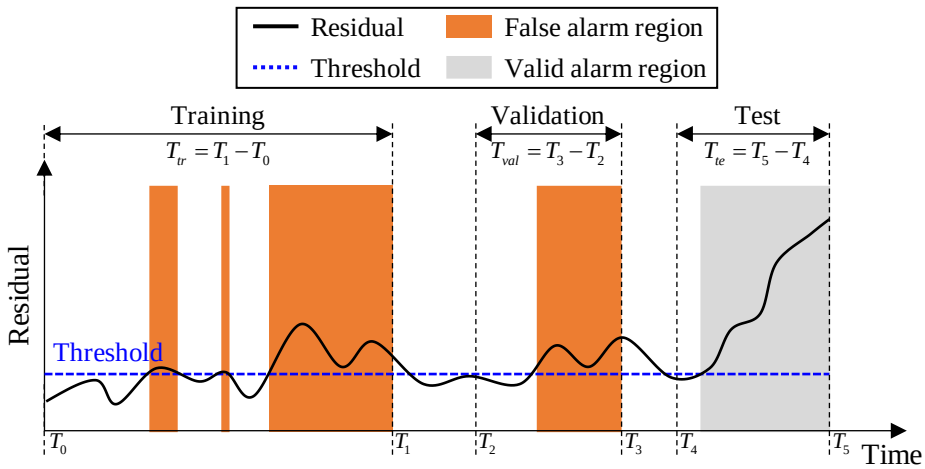


Figure 3-5 Definition of false alarms and valid alarms

3.4 Description of the Validation Datasets

Two datasets were collected from a steam turbine of domestic thermal power plant A. The power capacity of each generator of plant A is 500 [MW]. The operating signals of the steam turbine were measured by an OSIsoft PI system. The PI system organizes various signals from the entire power plant. 24 parameters related to the steam turbine were measured for this dataset; the meaning and the unit of each parameter are provided in Table 3-2, and the sensor locations are illustrated in Figure 3-6. Also, the sampling rate, data configuration, and several types of anomaly information for the two datasets are summarized in Table 3-3. The sampling rate is 1 [sample/min], and the number of parameters is 24 for the two datasets. Since EDAE should be designed to learn the characteristics of the normal condition well, the length of the training data is set as around 4-5 months, to be long enough to allow accurate modeling of the normal condition. The validation data is set to a length of

1 week, and the length of test data is around 1 day, including the experts' detection time T_e. The test period is inevitably short, since a turbine is usually stopped after any anomaly is detected. For dataset A_1, training, validation, and test data are successively constituted; however, the configuration is not successive for dataset A_2. The number of training samples is around 170,000 for both datasets. For case A_1, the shutdown began at 13:30:00 on 10/31/11, and the turbine was re-operated at 03:20:00 on 11/10/11. Here, and throughout, the date format is month/day/year; for instance, 10/31/11 is October 31, 2011. In the case of A_2, the turbine started to stop at 02:00:00 on 12/19/13; it was restarted at 19:40:00 on 12/19/13 after maintenance. The anomaly cause of A_1 is a high vibration in the x-direction at the 4th turbine bearing, detected by the operator at 12:20:00 on 10/31/11; the anomaly-related parameter is also analyzed as vibration in the x-direction at bearing #4 by experts. The anomaly reason for A_2 is leakage at the crossover pipe; this was discovered by the operator at 20:40:00 on 12/18/13. For A_2, among the pressure and temperature of crossover pipe signals, experts determined that the pressure of the crossover pipe is significantly related to the anomaly; on the other hand, the temperature is relatively slow to change due to the anomaly.

Figure 3-7 illustrates preprocessed-signal trends of anomaly-related parameters determined by experts; units are not expressed since they are normalized. The vertical black-dotted line is the anomaly detection time by the experts. For case A_1, the variation scale of the training and validation periods are similar to each other. However, vibration increased significantly in the test period near 14:00:00 on 10/30/11. That is, the change due to the anomaly is valid in the test period of A_1. The variations of training and validation data for case A_2 are also similar. However, the

variation in the test period is not valid, as compared to case A_1; this means that for A_2 it will be harder to detect the anomaly than it was for A_1.

Table 3-2 Condition parameter information of datasets A_1 and A_2

No.	Parameter	Unit	Notation
1	Vibration in the x-direction at bearing #1	mm	x_1
2	Vibration in the y-direction at bearing #1	mm	x_2
3	Vibration in the x-direction at bearing #2	mm	x_3
4	Vibration in the y-direction at bearing #2	mm	x_4
5	Vibration in the x-direction at bearing #3	mm	x_5
6	Vibration in the y-direction at bearing #3	mm	x_6
7	Vibration in the x-direction at bearing #4	mm	x_7
8	Vibration in the y-direction at bearing #4	mm	x_8
9	Vibration in the x-direction at bearing #5	mm	x_9
10	Vibration in the y-direction at bearing #5	mm	x_{10}
11	Vibration in the x-direction at bearing #6	mm	x_{11}
12	Vibration in the y-direction at bearing #6	mm	x_{12}
13	Vibration in the x-direction at bearing #7	mm	x_{13}
14	Vibration in the y-direction at bearing #7	mm	x_{14}
15	Vibration in the x-direction at bearing #8	mm	x_{15}
16	Vibration in the y-direction at bearing #8	mm	x_{16}
17	Vibration in the x-direction at bearing #9	mm	x_{17}
18	Vibration in the y direction at bearing #9	mm	x_{18}
19	Metal temperature of the crossover pipe	°C	x_{19}
20	Steam temperature of the crossover pipe	°C	x_{20}
21	Pressure of the crossover pipe	psi	x_{21}
22	Steam pressure of the hot reheater line	kg/cm^2g	x_{22}
23	Pressure of upstream of the low-pressure bypass valve	kg/cm^2g	x_{23}
24	Pressure of the hot reheater outlet line	kg/cm^2g	x_{24}

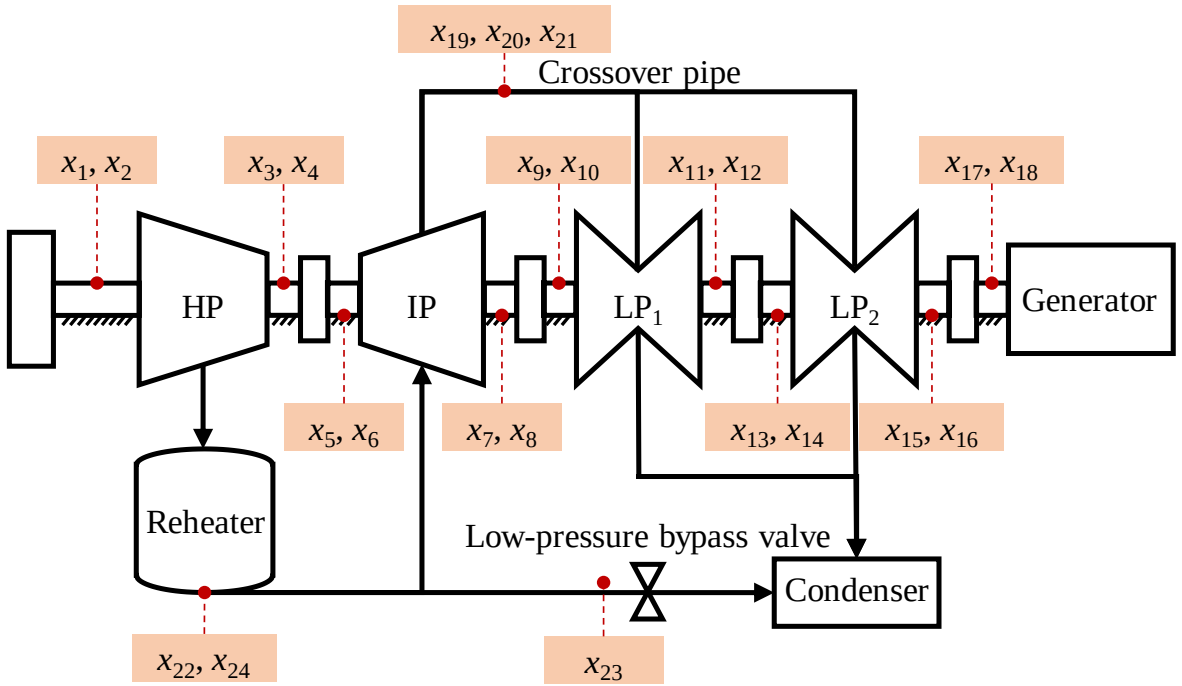


Figure 3-6 Sensor locations of a steam turbine

Table 3-3 Data description of datasets A_1 and A_2

Dataset		**A_1**	**A_2**
Sampling rate [sample/min]		1	
Number of condition parameters		24	
Data configuration	Training data	06/30/11 to 10/23/11	02/12/13 to 07/12/13
	Validation data	10/23/11 to 10/30/11	08/01/13 to 08/08/13
	Test data	10/30/11 to 10/31/11	12/18/13 to 12/19/13
Anomaly detection time by experts (T_e)		10/31/11 12:20:00	12/18/13 20:40:00
Start time of shutdown		10/31/11 13:30:00	12/19/13 02:00:00
Restart time after maintenance		11/10/11 03:20:00	12/19/13 19:40:00
Cause of anomaly		High vibration in the x-direction at bearing #4	Leakage at crossover pipe
Anomaly-related parameter by experts		Vibration in the x-direction at bearing #4 (x_7)	Pressure of crossover pipe (x_{21})

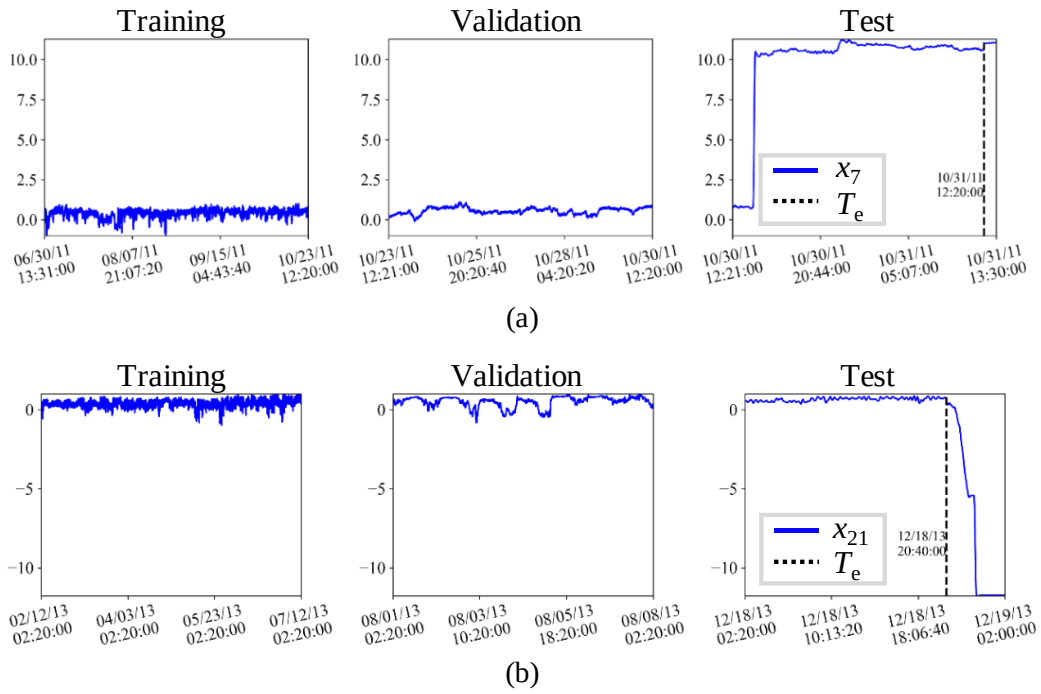


Figure 3-7 Trends of preprocessed anomaly-related condition parameters: (a) x_7 for A_1 and (b) x_{21} for A_2

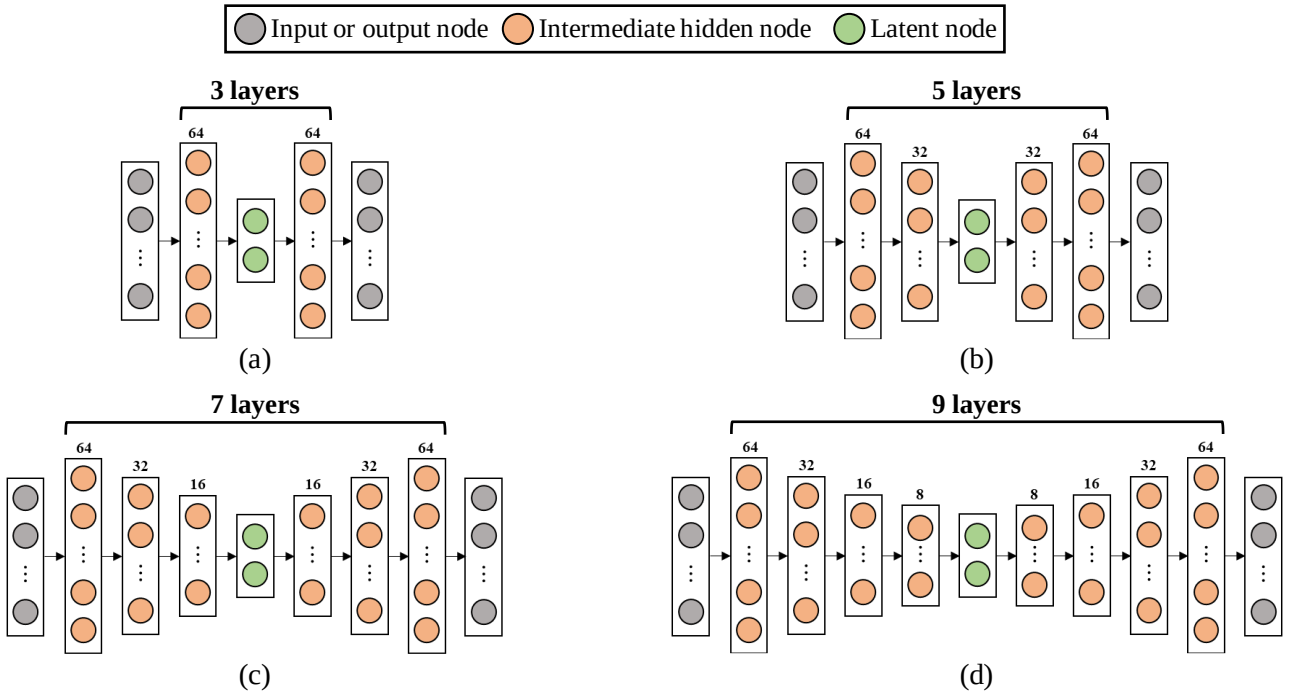


Figure 3-8 Architecture of four auto-encoders: (a) 3 layers, (b) 5 layers, (c) 7 layers, and (d) 9 layers

3.5 Validation of the Proposed Method

For dataset A_1, EDAE, AE, and DAE are trained with four different depths of hidden layers, as described in Figure 3-8; specifically, 3 layers, 5 layers, 7 layers, and 9 layers. The gray circles show the input or output nodes, the green circles represent the latent nodes, and the orange circles are the intermediate hidden nodes. For DAE and EDAE, noise with a signal-to-noise-ratio of 5 [dB] is added to the input signals. Training epochs and batch size are set as 60 and 64, respectively. EDAE is ensembled with five DAEs. The confidence level of each threshold method is set as 1e-3.

3.5.1 Case Study 1: Dataset A_1

For the four different architectures in Figure 3-8, modeling performances of AE, DAE, and EDAE are compared to each other. The critical hyper-parameters – the number of latent nodes and learning rate – are chosen by Bayesian optimization. The number of iterations is 12, and the acquisition function is chosen as *expected improvement* (EI). The convergence results of the optimization are summarized in Figure 3-9; the y-axis denotes the minimum validation loss until the iteration. The MAE of the validation data is converged in every case, which means a local optimum has been reached. Table 3-4 shows the Bayesian optimization results with respect to the different depths of the hidden layers. To make a bottleneck layer, the number of latent nodes is constrained to be less than the number of hidden nodes in the layer that is before the latent layer. The optimal learning rate decreases as a model becomes deeper because a small learning rate has the advantage of optimizing a complex neural network. Using the optimized critical hyper-parameters, the three algorithms are trained. The training and validation losses per epoch of the AEs,

DAEs, and EDAEs are organized in Figure 3-10. The red circles, blue triangles, green squares, and purple plus-shaped lines represent the results for 3 layers, 5 layers, 7 layers, and 9 layers, respectively. For each EDAE, the averaged losses of five DAEs are illustrated. The training losses are converged during the training procedure, and the validation loss is usually greater than the training loss. The difference between the converged training loss and the validation loss decreases in order for AE, DAE, and EDAE, respectively. This means that EDAE suffers the least from the overfitting issue.

RMSE values for trained AE, DAE, and EDAE are summarized in Figure 3-11. DAE shows a smaller RMSE value than AE, due to the denoising task. The RMSE values of EDAE are smaller than those of AE and DAE in every case. In particular, the RMSE value for an EDAE of 3 layers is the smallest. This means that the EDAE of 3 layers learns the normal condition better than other approaches. This is because a light neural network is enough to model the training data, whose input dimension is just 24. In the case of light data, a neural network with many hidden layers may have a severe overfitting problem. Consequently, the EDAE of 3 layers is selected for further study.

After training the EDAE of 3 layers using the training data, N-sigma, MD, and DT are obtained. Figure 3-12 represents the averaged anomaly detection metrics of those thresholds. When seeing the metric β, N-sigma produces 42.2934 valid alarms per hour, while DT generates slightly fewer valid alarms; the β value of MD is too small, as compared to the other two methods. Also, DT generates the first valid alarm faster than the experts by 0.78 days. While MD triggers the first valid alarm slower than experts, N-sigma triggers the alarm earlier than experts by 0.84 days, which is

slightly faster than DT. However, the metric for false alarms (α) shows that DT generates far fewer false alarms than either N-sigma or MD. Specifically, the α value of DT is about 32.92% of that of N-sigma. In summary, the results show that EDAE-DT can detect an anomaly faster than experts, while generating the fewest false alarms, as compared to the conventional methods.

Based on the newly proposed sensitivity in Eq. (3.11), the top three anomaly-sensitive parameters were selected; these are presented in Table 3-5. As you can see, vibration in the x-direction at bearing #4 (x_7) is the most sensitive parameter. This coincides with the sensitive parameter that is analyzed by the experts. The α value of that parameter is 0.0519 times per hour, which means that there is one false alarm every 20 hours, on average. On the other hand, the β value is 55.3907 times per hour, which shows that the most sensitive parameter can generate frequent valid alarms.

Figure 3-13 illustrates the critical function of x_7, and Figure 3-14 presents the output, residual, and dynamic threshold of x_7. In Figure 3-14(a), the first column shows the results of the training data, the second column shows those of validation, and the third column shows those of the test period. In the first row of Figure 3-14(a), the blue line is the true data, and the yellow line is the output. In the second row of Figure 3-14(a), the blue line is the residual, and the yellow plot is the dynamic threshold. The vertical black-dotted line denotes the time required for detection by experts. Since the residual is the L1 norm of the output and true data, the residual is not negative. Because a critical function produces a threshold for the EDAE's output, it is good for training and validation samples to be located under the critical function to mitigate the false alarm issue; this can be seen in Figure 3-13. Unlike the training and validation samples, test samples cross the function; this means that alarms occur

during the test period. Thus, it is confirmed that DT can effectively reduce false alarms, while generating valid alarms.

The diagnostic performances of N-sigma, MD, and DT are compared using classification metrics – precision, recall, accuracy, and F1 score – and a confusion matrix. The true labels of samples are annotated as binary; the samples during training and validation periods are labeled as normal, and the ones in the test period are labeled as an anomaly. The predicted label is obtained for each parameter as follows:

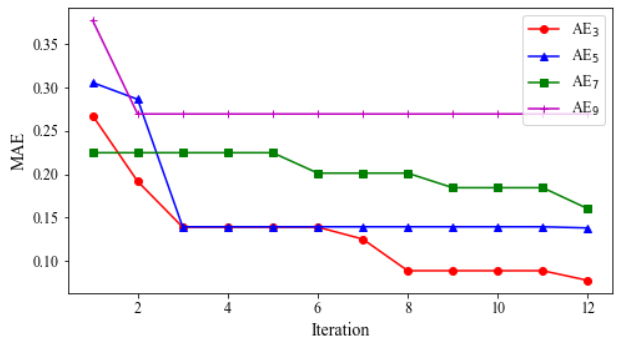
$$y_n^i = \begin{cases} 0 & if \quad r_n^i < t_n^i \\ 1 & otherwise \end{cases} \tag{3.13}$$

where y_n^i is a predicted label, r_n^i denotes a residual, and t_n^i is a threshold of the n-th parameter at time index i. Then, a single label at time index i (y^i) is calculated by averaging the predicted outputs of all parameters as follows:

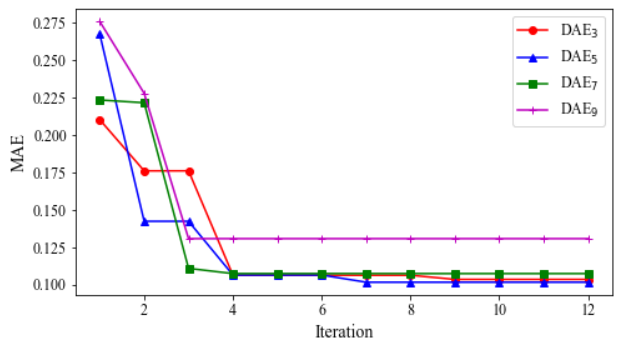
$$y^i = \begin{cases} 0 & if \quad \frac{1}{n}\sum_n y_n^i < 0.5 \\ 1 & otherwise \end{cases} \tag{3.14}$$

Table 3-6 shows the averaged diagnostic performance metrics over 10 independent trials for dataset A_1. As you can see, MD has inaccurate results, as compared to N-sigma and DT. The recall and accuracy of DT are greater than those of N-sigma, but the precision and F1 scores of DT are slightly smaller than those of N-sigma. That is, the diagnostic performances of DT and N-sigma are similar to each other. This is because 1) the labeling might be wrong due to the lack of exact label information, and 2) the number of faulty samples is far smaller than that of normal samples. The

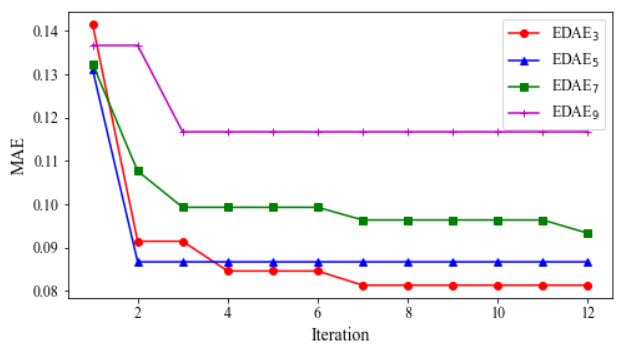
predicted labels for x_7 are described in Figure 3-15. A blue line denotes a residual, a yellow line is a threshold, a black dotted line denotes T_e, and a red circle is the predicted label of a sample. For MD, a health index is illustrated instead of the residual. Though MD classifies most of the samples during training and validation periods as normal, it also misclassifies the test samples as normal. This is consistent with the results of Figure 3-12, which denotes that the valid alarm rate of MD is the smallest. DT and N-sigma seem to have similar prediction results. The confusion matrices of the model used in Figure 3-12 are illustrated in Figure 3-16. The label of normal samples is 0, and that of the faulty ones is 1. The float value is the number of predicted samples over that of total samples, and the value in parentheses is that of predicted samples. From the confusion matrices, it can also be found that MD misclassifies the fault samples as normal. Also, DT and N-sigma have similar classification performance, which is also shown in Table 3-6 and Figure 3-15.



(a)



(b)

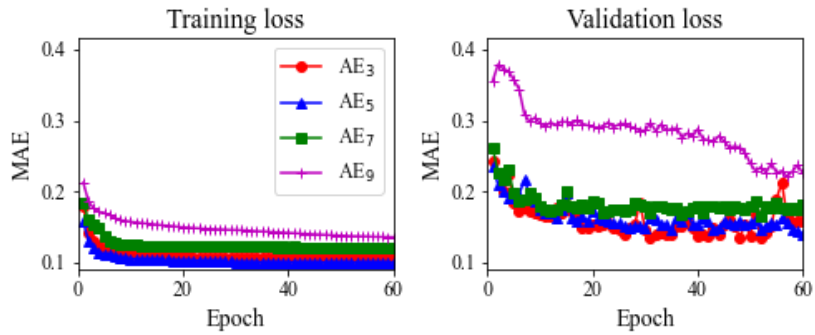


(c)

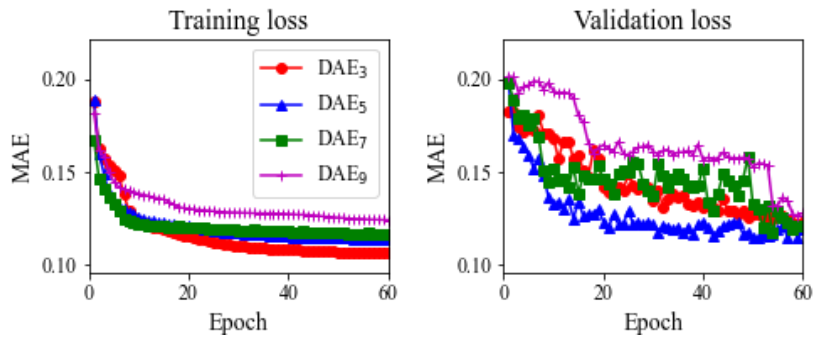
Figure 3-9 Convergence plots with dataset A_1: (a) AE, (b) DAE, and (c) EDAE

Table 3-4 Bayesian optimization results of AE, DAE, and EDAE for dataset A_1

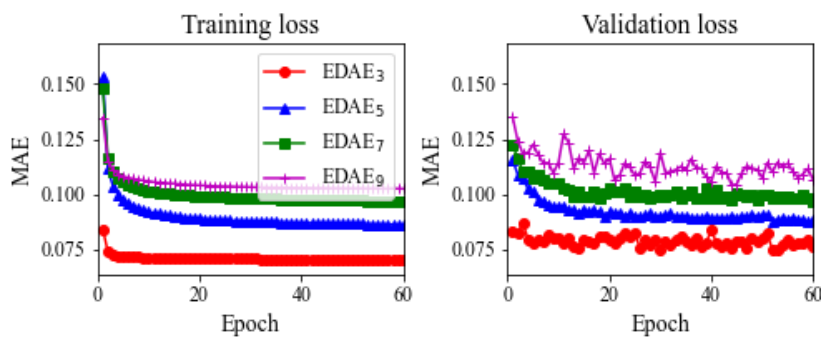
# of hidden layers	Hyper-parameters	AE	DAE	EDAE
3 layers	# of latent nodes	11	25	15
	Learning rate	0.004466	0.010000	0.000674
5 layers	# of latent nodes	25	18	10
	Learning rate	0.000873	0.001267	0.000120
7 layers	# of latent nodes	11	14	9
	Learning rate	0.000744	0.006773	0.000594
9 layers	# of latent nodes	2	4	7
	Learning rate	0.004984	0.001319	0.000120



(a) AE_3, AE_5, AE_7, and AE_9



(b) DAE_3, DAE_5, DAE_7, and DAE_9



(c) $EDAE_3$, $EDAE_5$, $EDAE_7$, and $EDAE_9$

Figure 3-10 Training and validation losses of auto-encoders for dataset A_1: (a) AEs, (b) DAEs, and (c) EDAEs

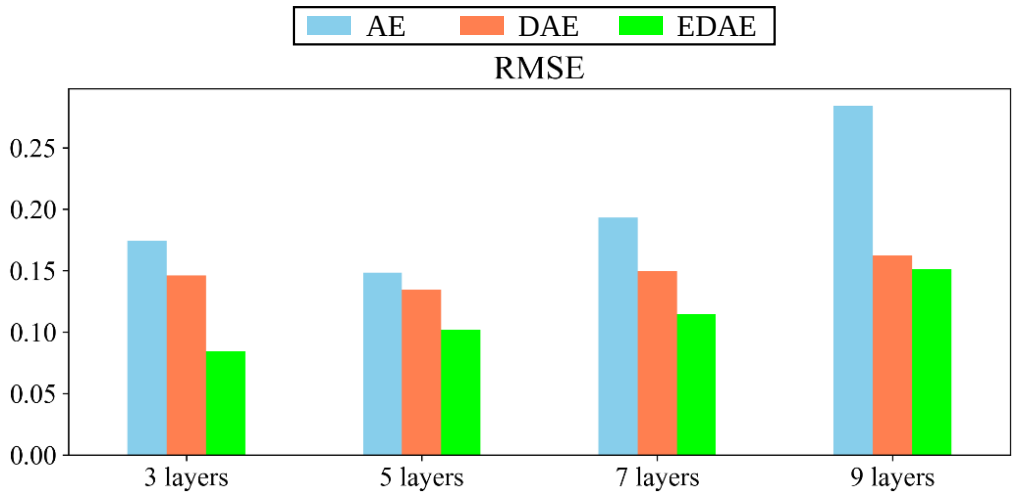


Figure 3-11 RMSE of AE, DAE, and EDAE with respect to four different architectures for dataset A_1

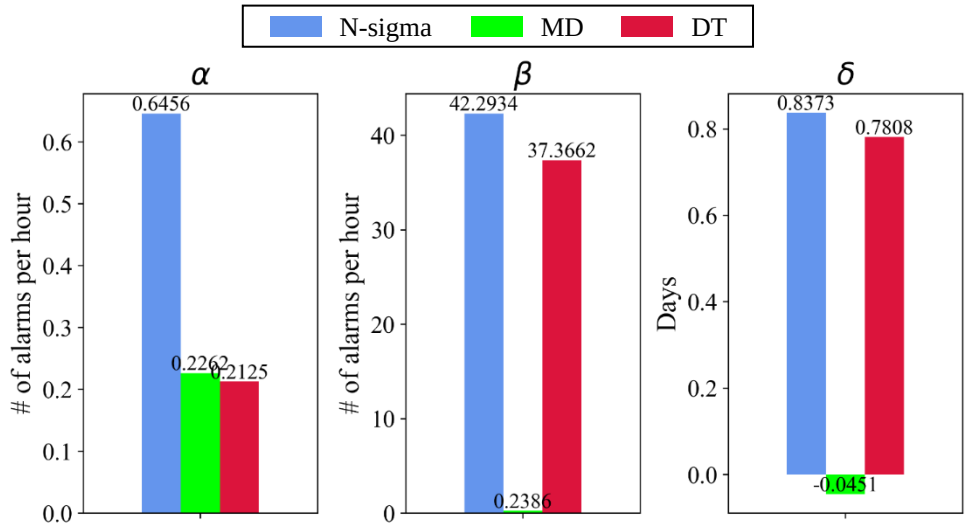


Figure 3-12 Averaged anomaly detection metrics of three thresholds for dataset A_1; N-sigma, MD, and DT

Table 3-5 Anomaly detection performance of the top three parameters of A_1

Condition parameters	***s***	***α* [times/hour]**	***β* [times/hour]**	***δ* [days]**
Vibration in the x-direction at bearing #4 (x_7)	0.9218	0.0519	55.3907	0.9174
Vibration in the x-direction at bearing #3 (x_5)	0.8716	0.0366	55.1921	0.9139
Vibration in the y-direction at bearing #5 (x_{10})	0.8395	0.04408	55.3510	0.9167

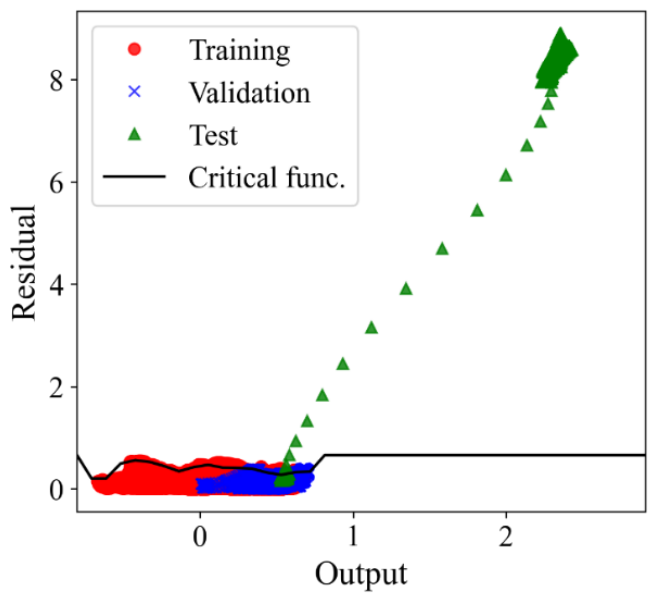


Figure 3-13 Critical function of x_7 for dataset A_1

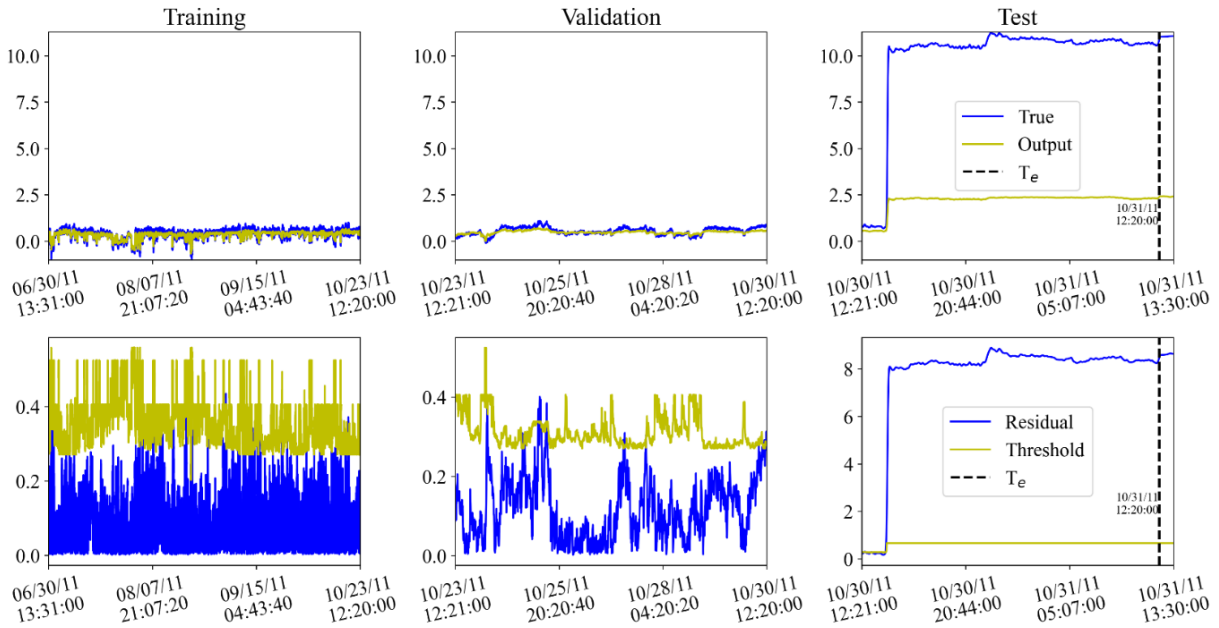
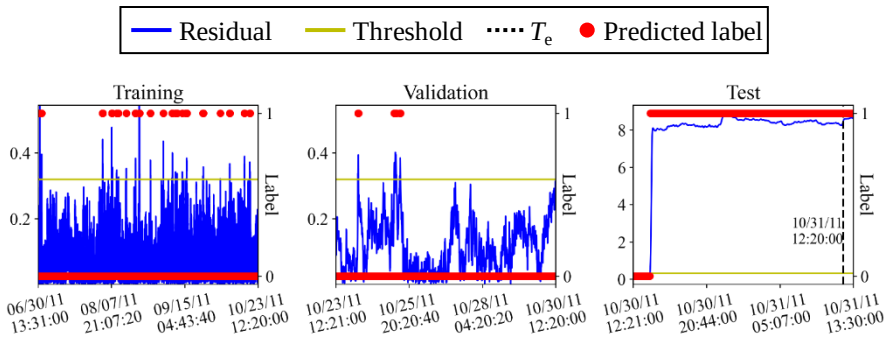


Figure 3-14 Output and residual results of EDAE for dataset A_1

Table 3-6 Averaged diagnostic performance of 10 trials for dataset A_1

Metrics	Precision	Recall	Accuracy	F1 score
N-sigma	**0.919**	0.992	0.993	**0.958**
MD	0.003	0.006	0.987	0.004
DT	0.918	**1.000**	**0.999**	0.957

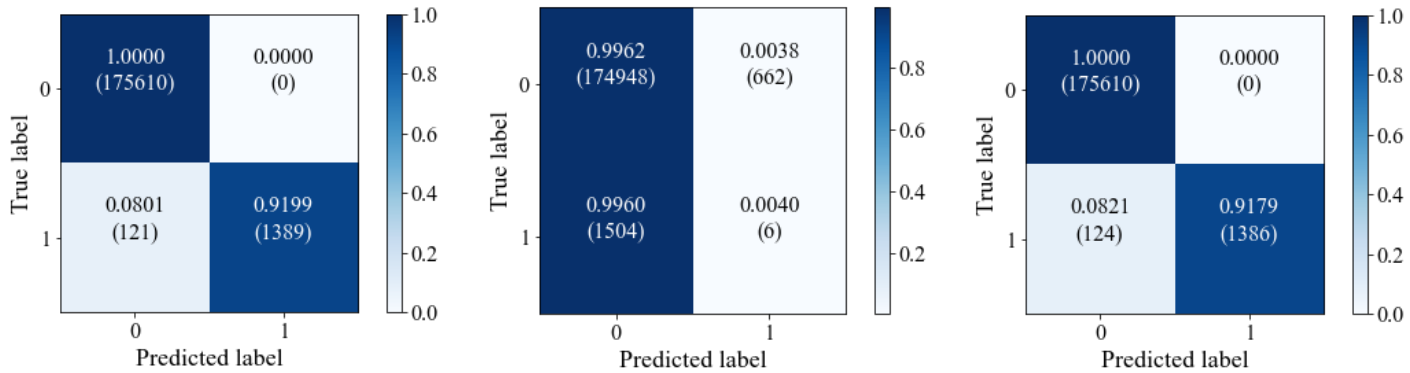


(a) N-sigma

(b) MD

(c) DT

Figure 3-15 Predicted label for x_7, as determined by the diagnostic methods: (a) N-sigma, (b) MD, and (c) DT



(a) N-sigma (b) MD (c) DT

Figure 3-16 Confusion matrices of the diagnostic methods for dataset A_1: (a) N-sigma, (b) MD, and (c) DT

3.5.2 Case Study 2: Dataset A_2

As in case A_1, the modeling performances of AE, DAE, and EDAE with respect to the different numbers of hidden layers are compared to each other. For AE, DAE, and EDAE, the critical hyper-parameters are selected by Bayesian optimization. The optimization settings are the same as in the case of A_1. The convergence plots are organized in Figure 3-17. The minimum objective function is converged during the optimization, which denotes that a local optimum has been found. Bayesian optimization results of AE, DAE, and EDAE are summarized in Table 3-7. In the same manner, as that used for A_1, the number of latent nodes is upper-bounded with the number of nodes of the previous layer to build a bottleneck architecture. As can be seen, the learning rate is generally decreased when the number of hidden layers increases. This is because a small learning rate is suitable for finding an optimal point in a more complex network. The three algorithms are trained with optimized hyper-parameters. The training and validation MAEs per epoch are summarized in Figure 3-18; the legend is the same as in Figure 3-10. The losses of five DAEs of each EDAE are averaged. The training and validation losses are converged in most cases. In addition, the difference between the training loss and validation loss of EDAE is smaller than those of AE and DAE.

Figure 3-19 illustrates the RMSE values of trained AE, DAE, and EDAE for four different architectures. As shown in the figure, for each architecture, RMSE values decrease in the order of AE, DAE, and EDAE. The RMSE value of EDAE for the 3-layer scenario is smaller than that of the others. This illustrates that an EDAE of three layers can model the normal data remarkably well, better than the other approaches. Deeper EDAEs show worse modeling performance than the

EDAE of 3 layers. The reason for this is that the deeper models suffer an overfitting problem. Therefore, the EDAE of 3 layers is analyzed in detail.

Three thresholds – N-sigma, MD, and DT – are calculated using the EDAE's output. Averaged anomaly detection metrics of those methods are described in Figure 3-20. First, the α value of N-sigma is the highest among the thresholds; this implies that the false alarm problem is the most severe when using N-sigma. MD falls in second place, and DT shows the smallest α. This indicates that the false alarm issue is not severe for DT, as compared to N-sigma and DT. The valid alarm rate β value of DT is 43.5976; this means DT triggers valid alarms about 43 times per hour. MD's β is 0.0423, which represents that MD triggers fewer valid alarms than N-sigma and DT; that is, MD is least sensitive to the change of multi-variate time-series data that arises due to an anomaly. Finally, δ values of DT and N-sigma are 0.5765; this describes that those methods detect an anomaly faster than experts by around 13 hours. In contrast, the δ value of MD is negative, which means that MD's detection is slower than experts. In summary, EDAE-DT produces the fewest false alarms, while triggering valid alarms faster than experts. Thus, DT is superior to the N-sigma and MD methods.

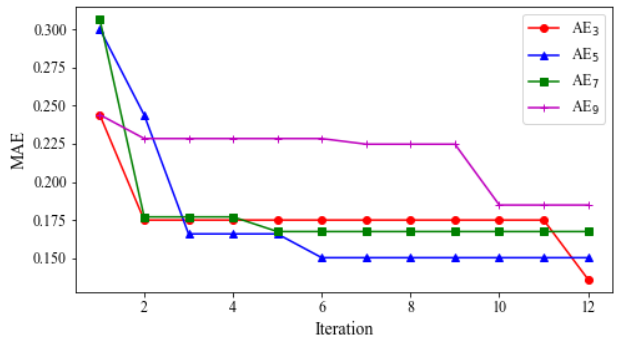
After validating the superior anomaly detection performance of EDAE-DT, parameters that are sensitive to the change of input due to the anomaly are selected, as outlined in Table 3-8. Those parameters are sorted in descending order based on the sensitivity. It turns out that the pressure of the crossover pipe has the largest sensitivity value, which matches the true anomaly cause shown in Table 3-3. Also, the false alarm rate of the parameter is around 0.052 times per hour, which is quite small. The valid alarm rate value is 57.9296 times per hour, which also describes

that the parameter is sensitive to the change of input data that happens due to an anomaly. Although there are temperature-related parameters at the crossover pipe (e.g., the metal temperature of the crossover pipe), these parameters are not selected as anomaly-sensitive parameters. The reason for this is that a change in temperature is slower than that of pressure when there is a sudden change in a system. Thus, pressure is a better choice for anomaly detection.

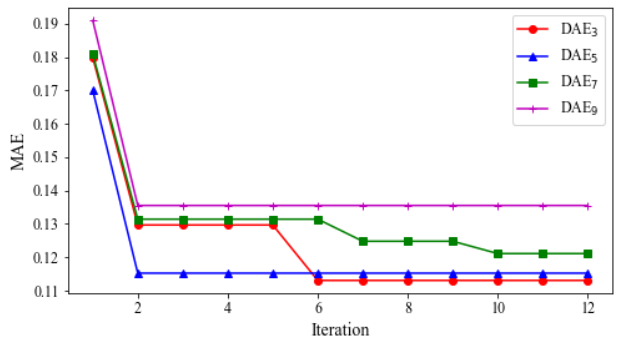
Figure 3-21 is the critical function of $x_{21,}$ and Figure 3-22 shows the output and residual of x_{21}. The legends are the same as those shown in Figure 3-13 and Figure 3-14, respectively. In the training and validation periods, the output of EDAE is similar to the true data; this indicates that EDAE can model the normal condition successfully. Furthermore, the greater error lies in the output of the test period. In Figure 3-21, the black line is the critical function. As you can see, the critical function exists over the training and validation points; this illustrates that false alarms can be diminished. Specifically, the residual of the test data increases gradually, crossing the critical function. Therefore, it can be validated that the dynamic threshold determined by the critical function can trigger valid alarms.

The performance of fault diagnosis of the N-sigma, MD, and DT approaches is compared through the use of classification metrics and a confusion matrix. The labeling method is the same as that used in the case of dataset A_1. The averaged performance metrics over 10 independent trials are summarized in Table 3-9. DT has the greatest recall, accuracy, and F1 score, as compared to other methods; the precision of DT is almost the same as that of N-sigma. Therefore, it can be said that DT has a more accurate diagnostic performance, as compared to the other methods. This matches with the facts found in Figure 3-20. However, the gaps in the diagnostic

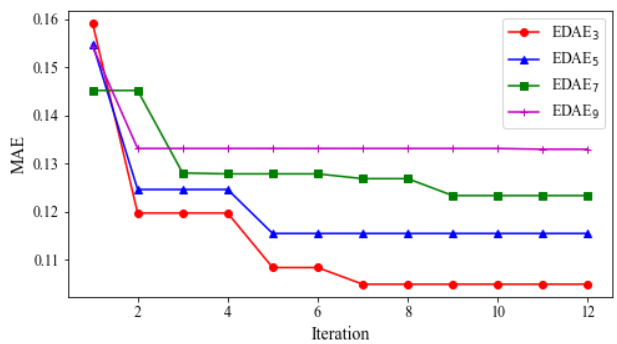
metrics between DT and N-sigma are not dramatic, as compared to the results in Figure 3-20. This is because 1) some samples might be wrongly labeled due to the absence of exact label information, and 2) the number of normal samples is much greater than that of faulty ones. Figure 3-23 describes the predicted labels for x_{21}; in the case of MD, a health index is plotted in place of the residual. MD mainly misclassifies the test samples as normal. Though DT and N-sigma have similar prediction results, the false alarm rate of DT is smaller than N-sigma when considering the training and validation samples. From the model employed in Figure 3-20, confusion matrices of those three methods are calculated in Figure 3-24. Likewise, for the results of case A_1, the classification performances of DT and N-sigma are similar to each other. Also, MD mainly predicts fault samples as normal, which denotes that its valid alarm rate is very small.



(a)



(b)

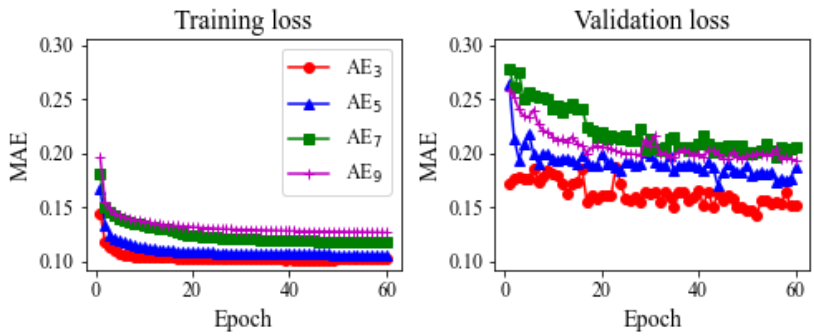


(c)

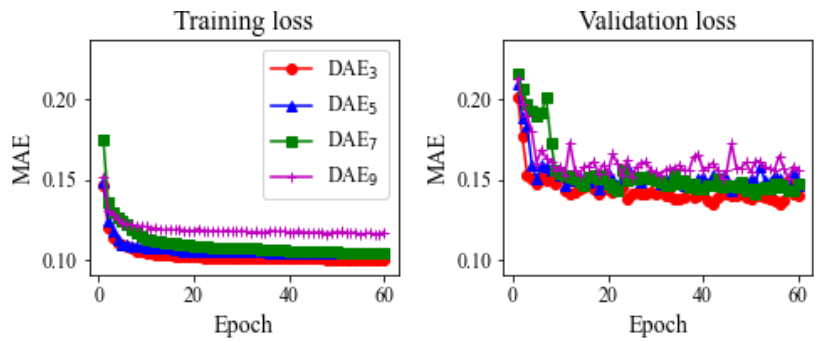
Figure 3-17 Convergence plots with dataset A_2: (a) AE, (b) DAE, and (c) EDAE

Table 3-7 Bayesian optimization results of AE, DAE, and EDAE for dataset A_2

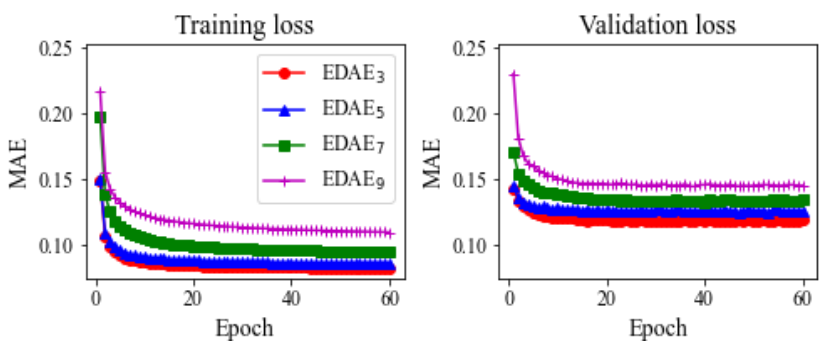
# of hidden layers	Hyper-parameters	AE	DAE	EDAE
3 layers	# of latent nodes	5	18	13
	Learning rate	0.009310	0.007051	0.000651
5 layers	# of latent nodes	15	11	15
	Learning rate	0.000565	0.001846	0.000257
7 layers	# of latent nodes	12	16	7
	Learning rate	0.000079	0.001746	0.000196
9 layers	# of latent nodes	6	7	6
	Learning rate	0.000031	0.002122	0.000201



(a) AE_3, AE_5, AE_7, and AE_9



(b) DAE_3, DAE_5, DAE_7, and DAE_9



(c) $EDAE_3$, $EDAE_5$, $EDAE_7$, and $EDAE_9$

Figure 3-18 Training and validation losses of auto-encoders for dataset A_2: (a) AEs, (b) DAEs, and (c) EDAEs

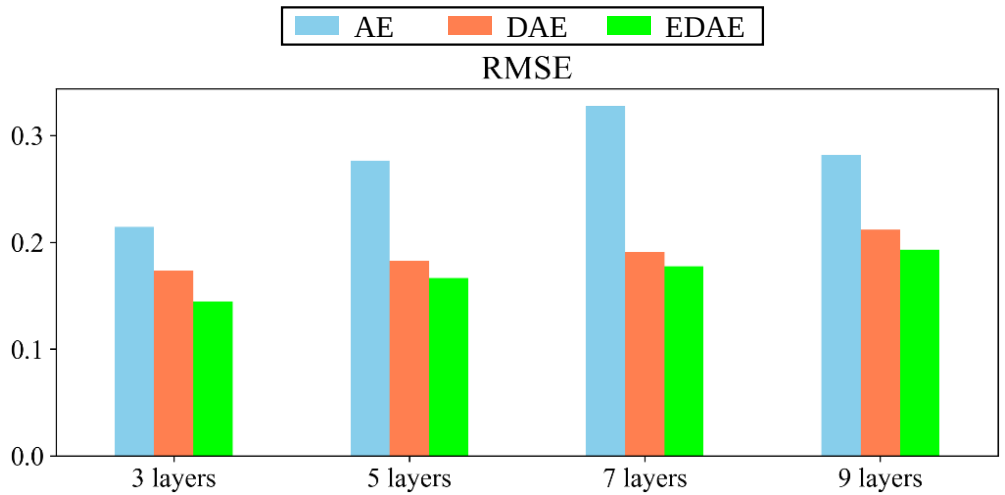


Figure 3-19 RMSE of AE, DAE, and EDAE with respect to four different architectures for dataset A_2

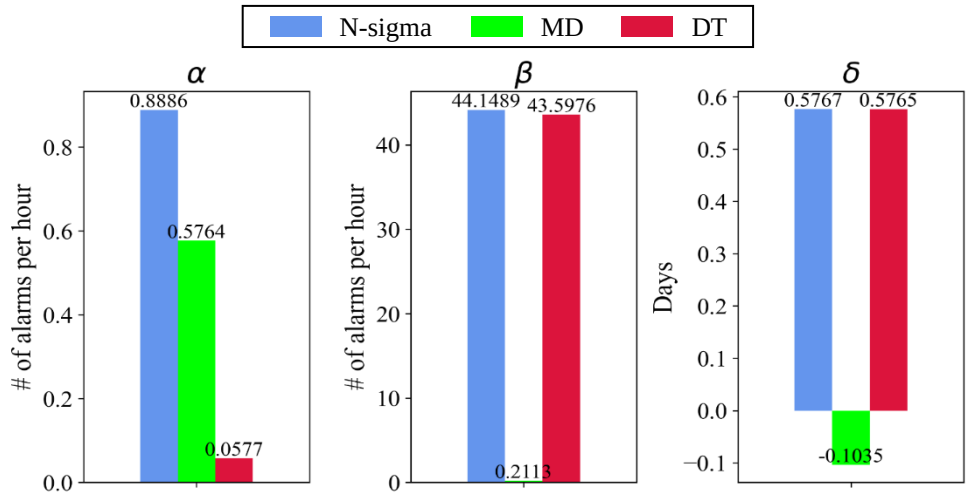


Figure 3-20 Averaged anomaly detection metrics of three thresholds for dataset A_2; N-sigma, MD, and DT

Table 3-8 Anomaly detection performance of the top three parameters of A_2

Condition parameters	***s***	***α* [times/hour]**	***β* [times/hour]**	***δ* [days]**
Pressure of the crossover pipe (x_{21})	0.9273	0.0520	57.9296	0.7639
Pressure of upstream of the low-pressure bypass (x_{23})	0.8216	0.0610	59.1549	0.7639
Pressure of the hot reheater outlet line (x_{24})	0.8184	0.0507	59.1127	0.7639

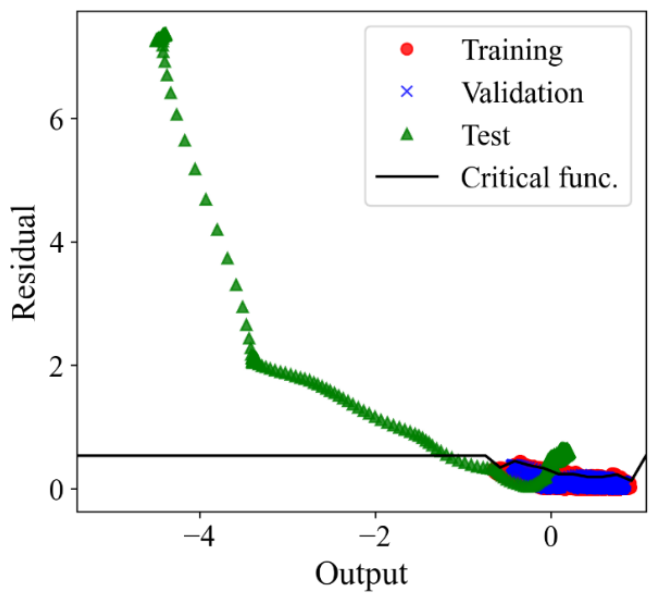


Figure 3-21 Critical function of x_{21} for dataset A_2

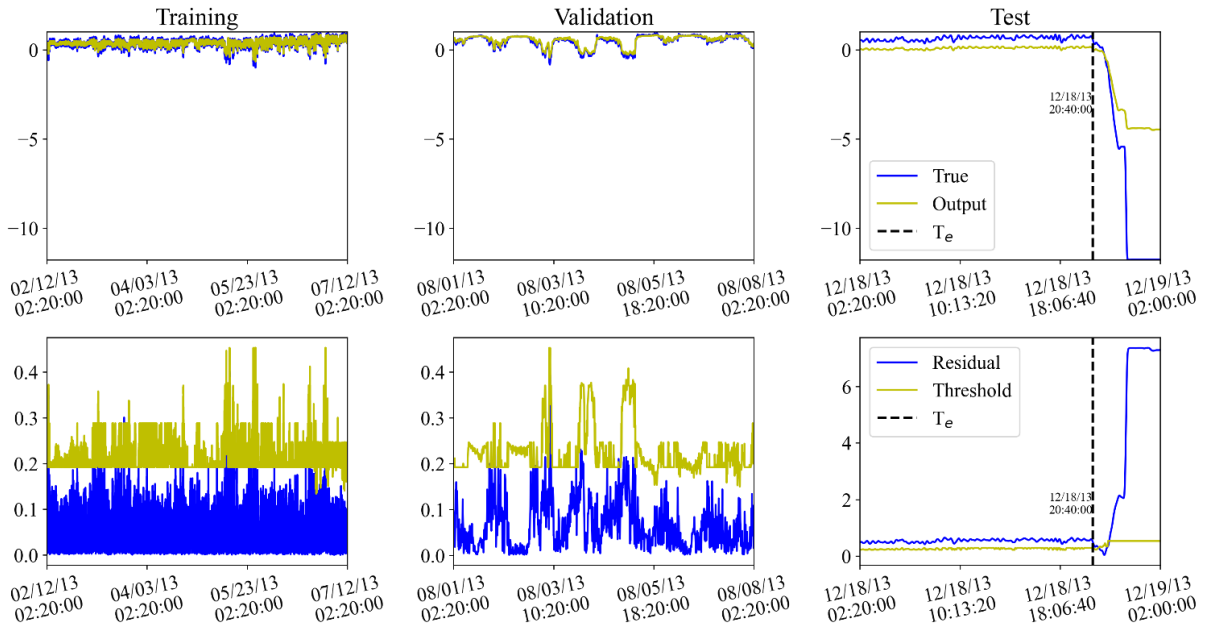
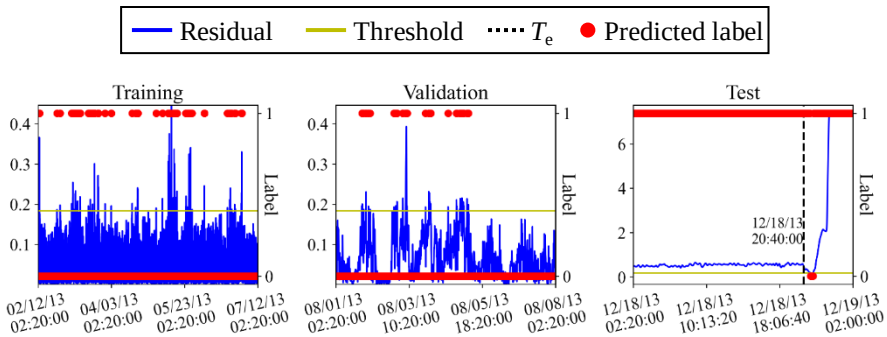


Figure 3-22 Output and residual results of EDAE for dataset A_2

Table 3-9 Averaged diagnostic performance of 10 trials for dataset A_2

Metrics	Precision	Recall	Accuracy	F1 score
N-sigma	**1.000**	0.967	0.999	0.983
MD	0.006	0.003	0.984	0.004
DT	0.997	**1.000**	**1.000**	**0.999**

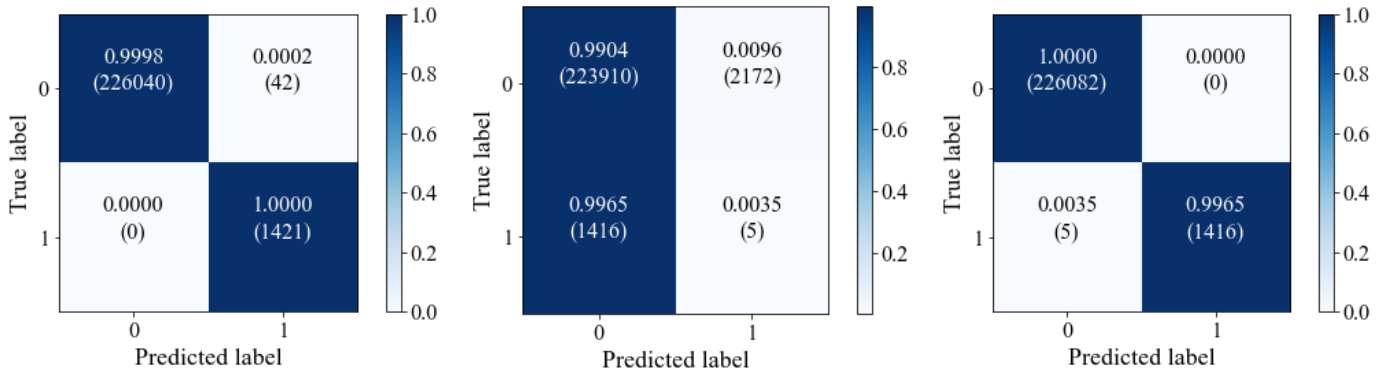


(a) N-sigma

(b) MD

(c) DT

Figure 3-23 Predicted label for x_{21}, as determined by the diagnostic methods: (a) N-sigma, (b) MD, and (c) DT



(a) N-sigma (b) MD (c) DT

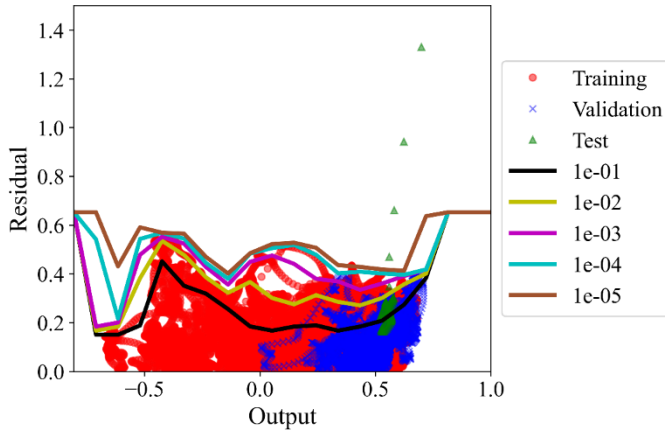
Figure 3-24 Confusion matrices of the diagnostic methods for dataset A_2: (a) N-sigma, (b) MD, and (c) DT

3.5.3 Analysis and Discussion

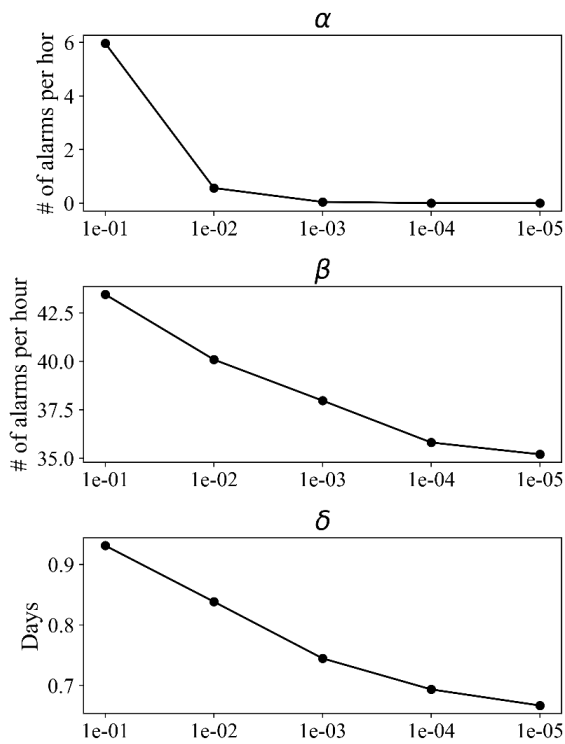
Since the confidence level (p) affects detection performance, the performance of the proposed method is investigated with respect to various confidence levels: p = [1e-01, 1e-02, 1e-03, 1e-04, 1e-05]. Figures 3-25 and 3-26 describe the critical functions and detection performance metrics of the datasets A_1 and A_2, respectively. As presented in Eq. (3.9), the critical points shift up in the residual direction as p gets smaller, which causes the critical function to move upward. This can be found in both Figures 3-25(a) and 3-26(a). Also, as can be seen from the detection performance results, α converges as p becomes greater than or equal to 1e-03 for both cases. β and δ decrease respectively when α increases; this means that the detection performance degenerates. This is because the threshold value increases as the critical function rises in the residual direction. As a result, it makes sense to set the confidence level as 1e-03 when making a trade-off among the three factors: minimizing α and maximizing β and δ.

The effect of the number of models (M) in EDAE is also analyzed. Though using more models in the ensemble technique usually presents better performance, the number of models cannot be increased infinitely because of computational cost. Figures 3-27 and 3-28 describe the modeling and anomaly detection performance results according to M = [3, 5, 7, 9, 11] for the datasets A_1 and A_2, respectively. In Figure 3-27, the EDAE of M =5 achieves the smallest RMSE value and the lowest false alarm rate (α); it also shows the greatest valid alarm rate (β). The δ value of the EDAE of M = 5 is slightly less than the greatest value, which is achieved by the EDAE of M = 7. When seeing Figure 3-28(a), the smallest RMSE value is obtained by the EDAE of M = 5. As can be seen from Figure 3-28(b), the EDAE of M = 5

presents better detection performance than other cases. It shows the greatest δ value, while achieving a small α value and a great β value. In summary, setting $M = 5$ is reasonable when considering the modeling and anomaly detection performance results.

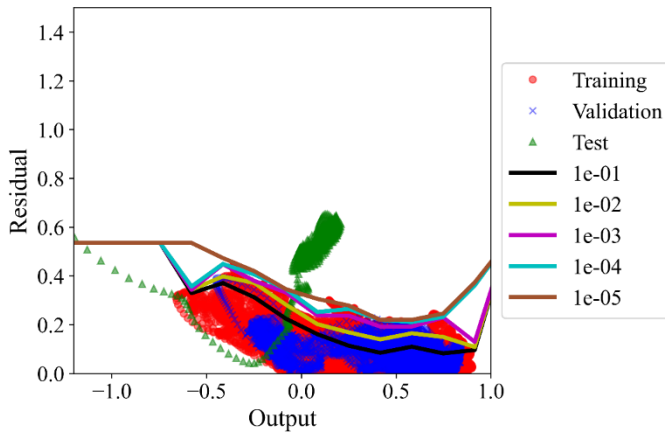


(a)

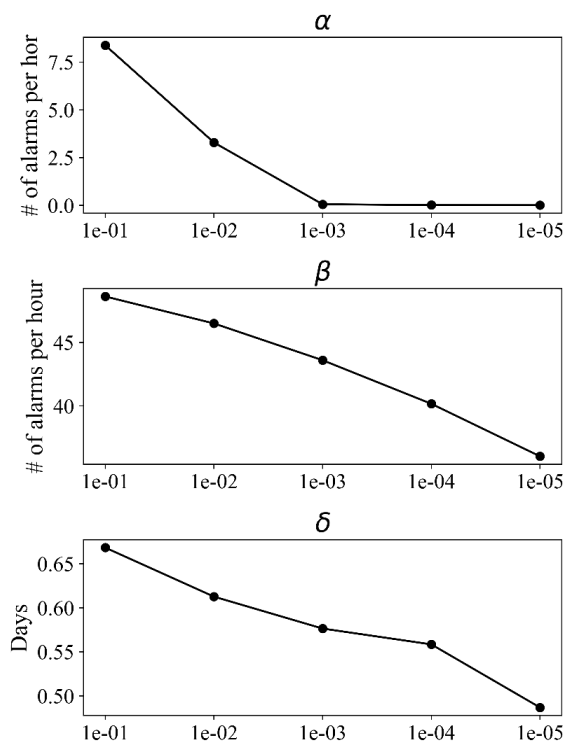


(b)

Figure 3-25 Anomaly detection performance with respect to the confidence level for dataset A_1: (a) critical functions and (b) detection performance metrics

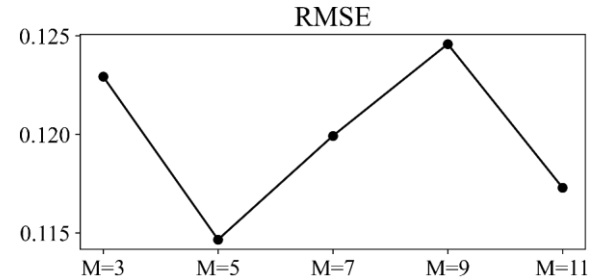


(a)

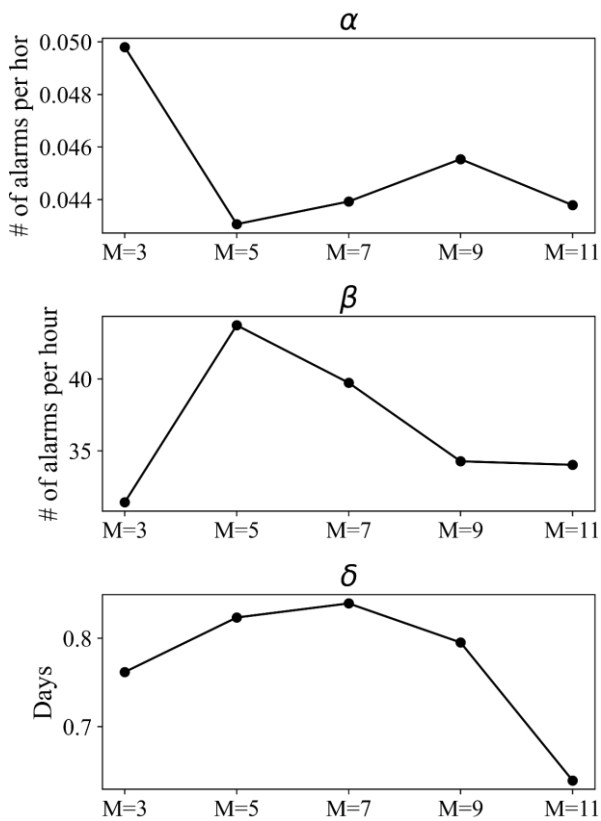


(b)

Figure 3-26 Anomaly detection performance with respect to the confidence level for dataset A_2: (a) critical functions and (b) detection performance metrics

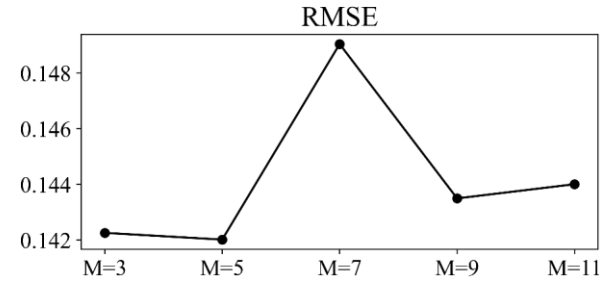


(a)

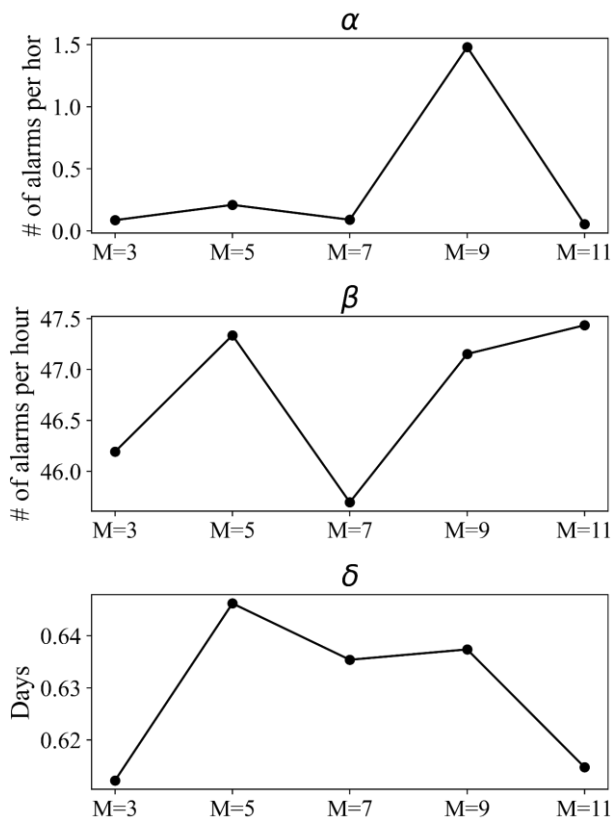


(b)

Figure 3-27 Performance according to the number of models in EDAE for dataset A_1: (a) modeling performance and (b) anomaly detection performance



(a)



(b)

Figure 3-28 Performance according to the number of models in EDAE for dataset A_2: (a) modeling performance and (b) anomaly detection performance

3.6 Summary and Discussion

This research proposes an EDAE-DT for accurate anomaly detection of a steam turbine. The EDAE approach can model the normal data successfully through its denoising task and its ensemble technique. The denoising task makes EDAE robust against noise, and the ensemble technique can improve the reconstruction performance. The DT method is developed to minimize false alarms in anomaly detection. By employing the joint probability distribution between the output of a model and the residual, a variable threshold is determined so that it satisfies the confidence limit according to the variation in the input. A sensitivity is newly defined by DT to find the condition parameters related to an anomaly. As a result, after an anomaly is detected, sensitive parameters can be identified. To quantitatively evaluate the anomaly detection performance, three performance metrics are newly defined. The proposed method is validated with two steam turbine datasets by using the metrics. Among the four different architectures, EDAE of 3 layers has a superior modeling performance than other auto-encoders. Also, the EDAE-DT approach generates much fewer false alarms, as compared to conventional methods, and alerts valid alarms faster than experts. It is also discovered that the most sensitive parameter, determined by the proposed sensitivity, matches with the true abnormal-related parameter. This can be helpful for the operators by localizing an area for inspection.

Sections of this chapter have been published as the following journal article:

1) **Jin Uk Ko**, Kyumin Na, Joon-Seok Oh, Jaedong Kim, and Byeng D, Youn, "A new auto-encoder-based dynamic threshold to reduce false alarm rate for anomaly detection of steam turbines," *Expert Systems with Applications*, Vol. 189, pp. 116094, 2022.

Chapter 4

Frequency-learning Generative Network (FLGN) for Data Augmentation

In this chapter, a new generative network called frequency-learning generative network (FLGN) is proposed 1) to generate signals of variable lengths at specific time ranges and 2) to ensure that there is little possibility of generating dissimilar samples. Though the proposed method completely differs from VAE and GAN, the proposed method is called a "generative network" since it is based on a neural network and tries to produce new signals. To generate signals at specific time ranges, the input becomes a time vector, and the output is set as the amplitude vector at that time. The key idea is the Fourier series, which indicates that a signal can be decomposed into several sinusoidal signals [72]. The proposed network is composed of three parts; a frequency extractor (*FE*), a phase extractor (*PE*), and a magnitude extractor (*ME*). Those extractors extract the stochastic frequency feature, the phase feature, and the magnitude feature, respectively. An attention block is utilized for each extractor so that it can focus on the important features. A deterministic frequency is learned in the form of a trainable parameter in a neural network. Then, a sine-basis is generated using the deterministic frequency parameter, the stochastic

frequency feature, and the phase feature. Then, the magnitude extractor extracts a magnitude feature using the sine-basis. Finally, a bias is added to the dot product of the sine-basis and the magnitude feature; this becomes the output of the proposed network. Through the research presented in this chapter, the proposed approach is verified by applying it to three datasets; a simulated signal, an RK4 dataset that was measured from a testbed of GE Bentley Nevada, and an open machinery fault database called MAFAULDA [73]. The generation performance is evaluated qualitatively and quantitatively. The validation results indicate that the proposed method can accurately generate signals for various time ranges, as desired. Furthermore, the proposed model can effectively learn the frequency components in the target signal. Specifically, when interpreting the proposed network by visualizing the attention score, it is found that the proposed model can focus on the characteristic frequency components.

The remainder of this chapter is organized as follows. Section 4.1 presents the theoretical background of the proposed method. Section 4.2 provides the proposed method in detail. The experimental implementation setting is offered in Section 4.3. Section 4.4 shows the descriptions of the validation datasets, and the validation results are presented in Section 4.5. Finally, the conclusion of this study and suggestions for future work are offered in Section 4.6.

4.1 Background: Fourier Series

Fourier series denotes that a periodic function is represented as the summation of sinusoidal waves [72]. Given a function $x(t)$, whose period is T, the Fourier series

expression of the function becomes as follows:

$$\begin{aligned} x(t) &= \frac{a_0}{2} + \sum_{n=0}^{\infty} a_n \cos\left(\frac{2\pi n}{T} t\right) + b_n \sin\left(\frac{2\pi n}{T} t\right) \\ &= \frac{c_0}{2} + \sum_{n=0}^{\infty} c_n \sin\left(\omega_n t + \phi_n\right) \end{aligned} \tag{4.1}$$

Here, c_n is the magnitude, ϕ_k is the phase, and c_0 is the bias. They are defined as follows:

$$\omega_n = \frac{2\pi}{T} n \tag{4.2}$$

$$c_n = \sqrt{a_n^2 + b_n^2} \tag{4.3}$$

$$\phi_n = \arctan\left(\frac{a_n}{b_n}\right) \tag{4.4}$$

$$\text{where } \begin{cases} a_n = \frac{2}{T}\int_0^T x(t)\cos\left(\frac{2\pi n}{T} t\right) dt \\ b_n = \frac{2}{T}\int_0^T x(t)\sin\left(\frac{2\pi n}{T} t\right) dt \\ c_0 = a_0 = \frac{1}{T}\int_0^T x(t)\, dt \end{cases} \tag{4.5}$$

The Fourier series can be interpreted as approximating a periodic function using frequency information such as magnitude, frequency, and phase. Inspired by this, the proposed method is developed. The motivation is graphically illustrated in Figure 4-1. To make a signal at a desired time range, the input is time. Then, feature extractors make magnitude, frequency, and phase. From this frequency information, another feature extractor computes sine-bases. Finally, the sine-bases are summed to yield the target signal.

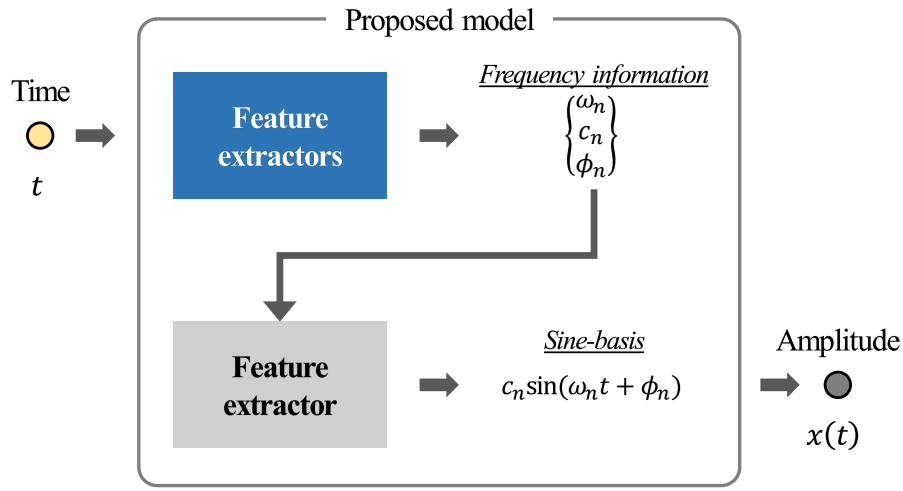


Figure 4-1 Motivation of the proposed method

4.2 Frequency-learning Generative Network (FLGN)

In this section, a novel generative model called frequency-learning generative network (FLGN) is explained in detail. The proposed method is developed to generate vibration signals of variable lengths and to minimize the risk of generating incorrect signals. The problem is formulated first with two assumptions: the target signal is stationary, and the training and test data have the same label conditions. Then, the detailed procedure of the proposed approach is described. Finally, the deep-learning settings to reflect signal processing knowledge are elucidated.

4.2.1 Problem Formulation

First, the problem that the proposed scheme is designed to address is formulated. The proposed method, which generates a signal at a desired time range, is developed

under the following assumptions:

1) It is assumed that the target signal is measured under a constant-speed condition. This means that the signal is stationary and that the frequency components remain constant.
2) The training and test data are assumed to have the same label conditions. For example, if the proposed model is trained with rubbing data, the proposed method will generate signals of the rubbing condition at different time ranges.

Let $D^{train} = \{t_i^{tr}, x(t_i^{tr})\}^{M}_{i=1}$ be the training dataset and $F(\cdot)$ be the proposed model. Here, t_i^{tr} is the time at the i-th index, $x(t_i^{tr})$ is the amplitude at the time t_i^{tr}, and M is the number of training samples. Then, the output of FLGN is $\hat{x}(t_i^{tr}) = F(t_i^{tr})$. When test data is $D^{test} = \{t_j^{te}\}^{M}_{j=1}$, the proposed method will generate $\hat{x}(t_j^{te})$. The time range of the test data can be changed as desired by the user.

4.2.2 Overall Procedure of FLGN

To produce signals of various lengths, a frequency-learning generative network (FLGN) approach is newly proposed in this research. Figure 4-2 illustrates the schematic diagram of the proposed method. The input is a time vector $\{t\}_{i=1}^{M}$, and the output is $\{\hat{x}(t)\}_{i=1}^{M}$, which is the amplitude vector at the corresponding time vector. There are three feature extractors; specifically, a frequency extractor (FE), a phase extractor (PE), and a magnitude extractor (ME). These all consist of

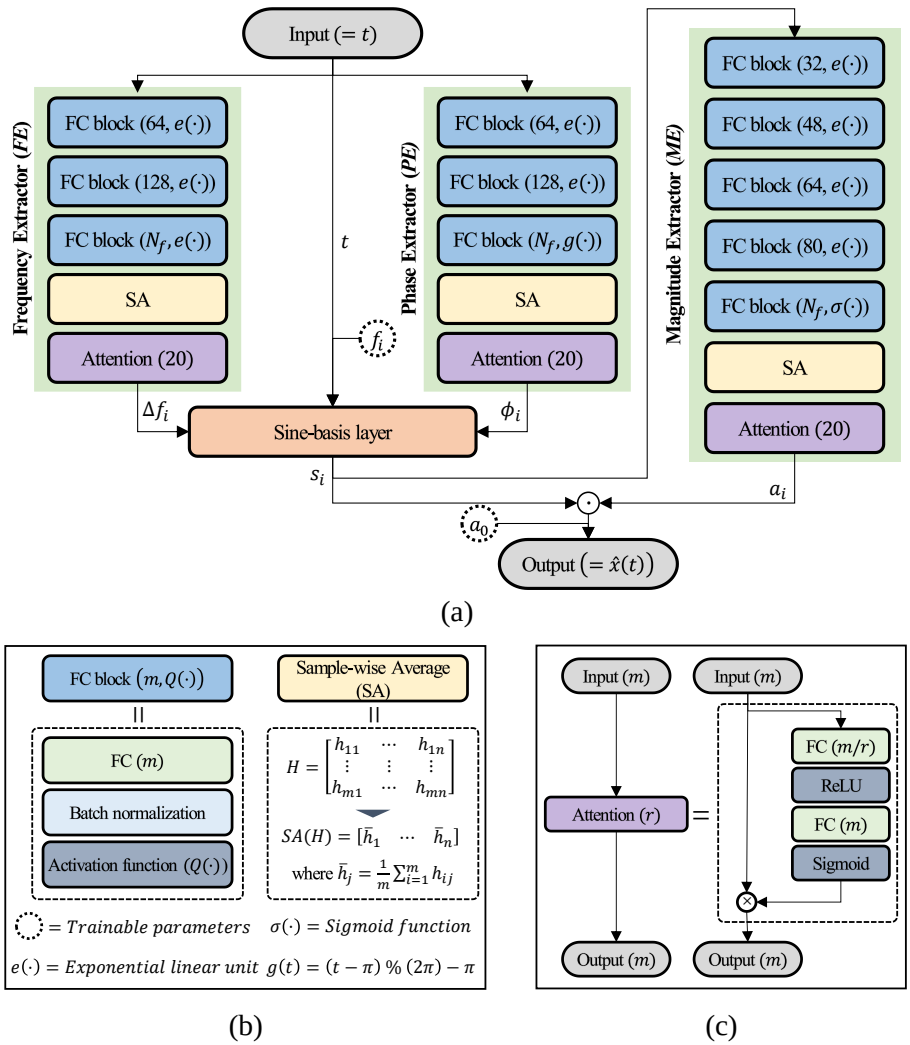


Figure 4-2 Schematic illustration of the proposed method: (a) architecture, (b) FC block, SA layer, and activation function, and (c) attention block

several fully connected (FC) blocks, one sample-wise average (SA) layer, and an attention block. FC block $(m, Q(\cdot))$ consists of 1) a fully connected layer where the

number of nodes is m, 2) a batch normalization layer, and 3) an activation function $Q(\cdot)$. The SA layer is used to sample-wisely average the feature, and an attention block is added to further focus on the important part of the averaged feature. The attention block is developed based on the squeeze and excitation network [74]. Using the attention blocks in each extractor, the proposed model can focus on the important frequency, phase, and magnitude features. f_i is a trainable parameter to learn the deterministic frequency and Δf_i is a feature used to learn the stochastic frequency; where $i = 1, \dots, N_f$ and N_f is the dimension of the frequency features. Δf_i is used because the frequency components can slightly change due to environmental disturbances, despite the constant-speed condition.

The input and output sizes of the main modules in FLGN are shown in Table 4-1. Let the length of the desired time range be B; then, the size of the input layer is $B \times 1$. *FE* and *PE* output a stochastic frequency feature (Δf_i) and a phase feature (ϕ_i), as presented in Eq. (4.6). The dimensions of both features become $1 \times N_f$; the first dimension is changed from B to 1 by the SA layer. $(f_i + \alpha \times \Delta f_i)$ will become the final frequency at the *i*-th index, which is constrained to be exist in the range of 0 and half of the sampling frequency (f_s). α is a hyper-parameter to control the effect of Δf_i. If the input signal has mostly deterministic frequency components, a small α will be more proper. $(f_i, \Delta f_i, \phi_i)$ are combined to make a sinusoidal basis (s_i) in the sine-basis layer, as shown in Eq. (4.7). The output size of the sine-basis layer is $B \times N_f$. Using the sine-basis, *ME* extracts a magnitude feature (a_i) like Eq. (4.8). The output dimension of *ME* is $1 \times N_f$. Finally, a bias (a_0) is added to the dot product of the sine-basis feature and the magnitude feature, as in Eq. (4.9). It is similar to the Fourier series, which approximates a signal as the

summation of sinusoidal signals. This becomes the final output $\left(\hat{x}(t)\right)$ of FLGN, whose dimension is $B \times 1$; this is same as the size of the input time vector. The objective function is mean squared error (MSE), as presented in Eq. (4.10). In the equation, *j* is the sample index, and *B* is the number of samples; for the training, *B* becomes the batch size. In Table 4-1, the input and output layers share the same size $B \times 1$. Since *B* can be determined as a user want, it is confirmed that the proposed method can generate signals of variable lengths.

$$\begin{aligned} f_i &= \left(Trainable\ parameter\right) \\ \triangle f_i &= FE\left(t\right) \\ \phi_i &= PE\left(t\right) \end{aligned} \tag{4.6}$$

$$s_i = \sin\left(2\pi\left(f_i + \alpha \times \triangle f_i\right)t + \phi_i\right) \tag{4.7}$$

$$a_i = ME\left(s_i\right) \tag{4.8}$$

$$\hat{x}\left(t\right) = a_0 + \sum_i a_i s_i = a_0 + \sum_i ME\left(s_i\right)\sin\left(2\pi\left(f_i + \alpha \times \triangle f_i\right)t + \phi_i\right) \tag{4.9}$$

$$L\left(x, \hat{x}\right) = \frac{1}{B}\sum_{j=1}^{B}\left(x_j - \hat{x}_j\right)^2 \tag{4.10}$$

Table 4-1 Input and output size of main modules in FLGN

Module	**Input size**	**Output size**
Input layer	$B \times 1$	$B \times 1$
FE	$B \times 1$	$1 \times N_f$
PE	$B \times 1$	$1 \times N_f$
Sine-basis layer	$1 \times N_f$	$B \times N_f$
ME	$B \times N_f$	$1 \times N_f$
Output layer	$1 \times N_f$	$B \times 1$

Table 4-2 Training procedure of FLGN

Input: Training data $D^{train} = \{t_i^{tr}, x(t_i^{tr})\}_{i=1}^{M}$; batch size B; N_f; deterministic frequency range (p, q); α; learning rate η; *training epochs*
Output: Model configuration of FLGN

I) Parameter initialization

Deterministic frequency parameters → $\varphi_f = (f_1, \dots, f_{N_f})$
Trainable parameters in *FE* and *PE* → φ_1
Trainable parameters in *ME* → φ_2
Total trainable parameters → $\theta = (\varphi_f, \varphi_1, \varphi_2)$
Initialize φ_f with uniform distribution $U(p, q)$
Initialize φ_1 with the *He uniform initialization* method
Initialize φ_2 with the *He normal initialization* method

II) Mini-batch training

while *validation loss does not converge* **do**
 for *epoch = 1* to *training epochs* **do**
  for *batch = 1* to $\left\lceil \frac{M-B}{\lfloor B/8 \rfloor} \right\rceil$ **do**
   Draw mini-batch samples $\{(t_1, x(t_1)), \dots, (t_B, x(t_B))\}$ from D^{train}
   Compute $\hat{x} = F(t; \theta)$
   Calculate loss function $L(x, \hat{x})$ in Eq. (4.10)
   Update parameters $\theta \leftarrow \theta - \eta \frac{\partial L}{\partial \theta}$
  end for
 end for
end while

The training procedure is described in Table 4-2. Given training data whose amplitude is min-max scaled, hyper-parameters, including batch size, N_f, frequency range (p, q), α, the learning rate (η), and the number of training epochs are chosen first. Next, the deterministic frequency parameters are initialized by uniform distribution $U(p, q)$. The parameters in *FE* and *PE* are initialized by the *He uniform*

initialization method [75], and those in *ME* are initialized by the *He normal initialization* method [75]. The proposed model is trained by mini-batch learning until the validation loss converges. The mini-batch samples are drawn to be overlapping; the stride is set as $\lfloor (B/8) \rfloor$. After the training is finished, the developed model can predict the amplitude in the test time range.

4.2.3 Deep-learning Implementation Details to Reflect Signals Processing Knowledge

First, because it is sometimes unknown which frequency is dominant, the deterministic frequency (f_i) is initialized with uniform distribution $U(p, q)$. (p, q) should satisfy the following condition: $0 \leq p < q < f_s/2$. In particular, the range can be chosen using prior knowledge about the frequency information in the target signal. For example, if it is known that frequency components of the target signal exist around 60 [Hz], the range can be selected as $(50, 70)$. N_f should be large enough to have the ability to learn most frequency components. For instance, if there are over 10 sub-harmonics of the fundamental frequency, setting N_f less than 10 makes it difficult to learn most of the frequency information. Also, the frequency $(f_i + \alpha \times \Delta f_i)$ is constrained to be between 0 and the Nyquist frequency $(f_s/2)$ to satisfy the Nyquist-Shannon sampling theorem [16]. In addition, the phase feature (ϕ_i) is restricted to exist between $-\pi$ and π. To do this, the following activation function Eq. (4.11) is applied to the end of the phase extractor.

$$g(h) = (h - \pi)\%(2\pi) - \pi \tag{4.11}$$

Here, "%" is the modulus operator. The function $g(h)$ is a periodic function, whose

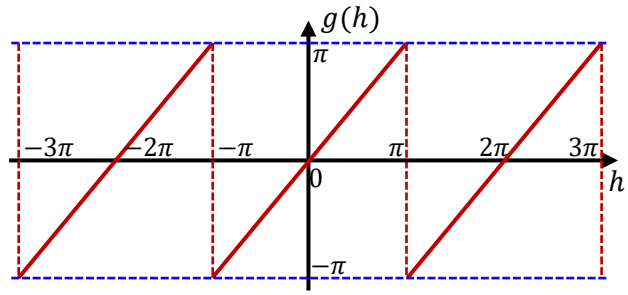


Figure 4-3 Trend of activation function $g(h)$

period is 2π; it is plotted in Figure 4-3. Finally, *L1 regularization* is applied to the deterministic frequency parameters (f_i) to make any useless frequency components zero [76]; the regularization scale is 1e-4. A regularizer combining *L1* and *L2* penalties is also applied to the magnitude extract for a similar reason; the scale value is 1e-4. This regularization can restrict most parameters to be small; that is, the magnitude features, except the features about the characteristic frequencies, will become small.

4.3 Experimental Implementation Setting

This section introduces the rest of the hyper-parameter conditions of the proposed method and the evaluation scheme. To rigorously validate its generation performance, FLGN is validated by both qualitative evaluation and quantitative evaluation.

4.3.1 Hyper-parameter Setting

The Adam optimizer is used to optimize the parameters of FLGN [55]. The initial learning rate is 5e-4, and the learning rate decay ratio is 1e-7. No bias is used at any fully connected layer. For an attention block, the reduction hyper-parameter r is chosen as 20. A batch normalization layer and an exponential linear unit (ELU) are employed with each fully connected layer. A new activation function $g(h)$, which is defined in Eq. (4.11), and a sigmoid function are used right before the SA layer in *PE* and *ME*, respectively. Batch size, N_f, and α are chosen differently for each dataset. In particular, the batch size should be large enough to contain the most frequency information. When training the proposed method with a mini-batch method, the batch size plays a role in the sequence length. Thus, if the batch size is too small, the mini-batch sample will not involve enough frequency components since the frequency resolution will be too big.

4.3.2 Evaluation Scheme

Given a time vector as the input, the proposed model generates the signal of that time range. Here, to test the proposed method, the generation performance is evaluated both qualitatively and quantitatively. For the qualitative evaluation, the true and the generated signals are visualized in the time domain and in the frequency domain, respectively. Magnitude spectrums of the true and the generated signals are compared in the frequency domain. If the generated signal is similar to the true one, both signals will also appear similar in both domains.

Similarity metrics and handcrafted features are computed for the quantitative evaluation. Root mean squared error (RMSE) and the correlation coefficient values

are calculated to identify how similar a generated signal is to the true signal. Features of both domains are also computed for quantitative evaluation. Time-frequency features are not considered because it is assumed that the signal is stationary [77]. If the generated and true signals are similar, those feature values will also be similar. The features of both domains, referring to [78], are summarized in Table 4-3. Here, X is the amplitude in the time domain, N is the length of X, f is the frequency, and $s(f)$ is the power spectrum function of X. RMS is relevant to the kinetic energy of the signal, and skewness and kurtosis can reflect the statistical characteristics of the signal. Shape factor, impulse factor, and crest factor describe how much the signal is similar to a sinusoidal waveform. The frequency center and root mean squared frequency (RMSF) indicate the fundamental frequency of the signal. Finally, the root variance frequency (RVF) shows how spread out the frequency components are.

The generation performance is further investigated using an auto-encoder. If the signals produced by the FLGN method are similar to the true signals, the auto-encoder that is trained only with the true signals will successfully reconstruct the generated signals. Figure 4-4 graphically illustrates the evaluation based on the auto-encoder. An FLGN model is trained, and the auto-encoder is trained using the true signals; its objective is to reconstruct the true data. Finally, the latent space is visualized, and RMSE and correlation coefficient between the signals reconstructed from the true and generated signals are computed. If the produced signals are similar to the true signals, both signals will be reconstructed successfully. Thus, the latent vectors of both signals will be close to each other and RMSE will be small, and the correlation coefficient will be near 1.

Table 4-3 Time-domain and frequency-domain features

Domain	Feature	Notation	Definition
Time domain	RMS	X_{rms}	$\sqrt{\frac{\sum_{i=1}^{N} X_i^2}{N}}$
	Skewness	X_{skew}	$\frac{\sum_{i=1}^{N}(X_i - X_{mean})^3}{(N-1)s^3}$
	Kurtosis	X_{kurt}	$\frac{\sum_{i=1}^{N}(X_i - X_{mean})^4}{(N-1)s^4}$
	Shape factor	X_{sf}	$\frac{X_{rms}}{Mean(\lvert X \rvert)}$
	Impulse factor	X_{if}	$\frac{Max(X)}{Mean(\lvert X \rvert)}$
	Crest factor	X_{cf}	$\frac{Max(X)}{X_{rms}}$
Frequency domain	Frequency center	X_{fc}	$\frac{\int_0^\infty f \times s(f)df}{\int_0^\infty s(f)df}$
	RMSF	X_{rmsf}	$\left(\frac{\int_0^\infty f^2 \times s(f)df}{\int_0^\infty s(f)df}\right)^{1/2}$
	RVF	X_{rvf}	$\left(\frac{\int_0^\infty (f - X_{fc})^2 \times s(f)df}{\int_0^\infty s(f)df}\right)^{1/2}$

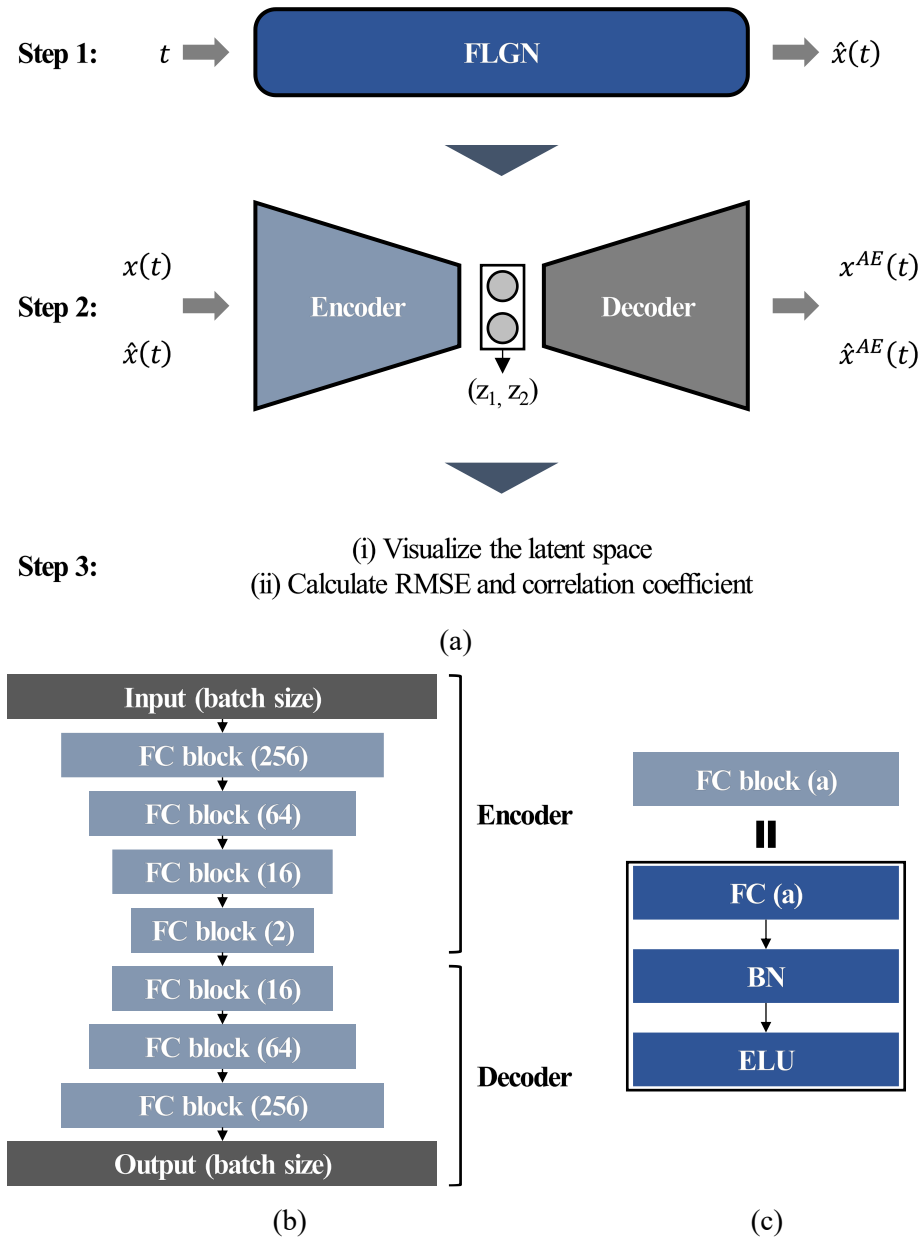


Figure 4-4 Performance evaluation using an auto-encoder: (a) procedure, (b) architecture of the auto-encoder, and (c) FC block in the auto-encoder

4.4 Description of the Validation Datasets

The developed method is validated on three datasets. The first dataset is a simulated signal (D_1), which contains impulsive signals and the signal of a low-frequency component. Two types of periodic impulsive signals with frequencies of 10 [Hz] and 25 [Hz] are involved to simulate the repeated impulsive fault. The low frequency is 5 [Hz], and white-Gaussian noise is added. The sampling frequency is determined as 2000 [Hz]. Figure 4-5 describes the time-domain and frequency-domain trends. The mathematical expression of the simulated signal is presented in Eq. (4.12); where "$*$" is the convolution operator. The characteristic frequencies are 5 [Hz], 330 [Hz], and 500 [Hz].

$$\begin{gathered} x_1(t) = e^{-200t}\sin\left(2\pi \times 500 \times t\right) \\ x_2(t) = 4e^{-150t}\sin\left(2\pi \times 330 \times t\right) \\ x_3(t) = 0.3\sin\left(2\pi \times 5 \times t\right) \\ \varepsilon \sim N\left(0, 0.1^2\right) \\ y(t) = x_1(t) * \sum_{k=0}^{\infty}\delta\left(t - \frac{k}{10}\right) + x_2(t) * \sum_{k=0}^{\infty}\delta\left(t - \frac{k}{25} + \frac{1}{200}\right) + x_3(t) + \varepsilon \end{gathered} \tag{4.12}$$

The second dataset is the RK4 dataset (D_2), which was measured from a GE Bently-Nevada testbed. The testbed setup is presented in Figure 4-6(a). The time-domain and frequency-domain trends are described in Figure 4-7. For this dataset, vibration signals were measured using two proximity sensors located at 90-degree intervals. The sampling frequency is 8500 [Hz], and the experiment was conducted in a steady-state condition of 3600 [rpm]; thus, the fundamental frequency (f_0) is 60 [Hz]. There are five health conditions in this dataset, including normal, misalignment, unbalance, oil whirl, and rubbing. Detailed information about the experiment can be found in [79]. Among the five health states, rubbing and oil whirl conditions are

examined in this study. As shown in Figure 4-7, the sub-harmonic components at $2f_0$ and $3f_0$ exist in the rubbing signal and the oil whirl signal contains the $0.5f_0$ component.

MAFAULDA (D_3) [73] is used as the third validation dataset. This dataset was measured from a Machinery Fault Simulator (MFS) testbed. Figure 4-6(b) shows the setup of the MAFAULDA testbed, and Figure 4-8 presents the time-domain and the frequency-domain trends of the imbalance and the horizontal misalignment conditions. There is a disc and a shaft that is supported by two rolling bearings; accelerometers are located at two points. The sampling frequency is 51200 [Hz], which is the highest among the three validation datasets. The dataset includes various fault conditions with different levels of fault severity and rotating speed; the rotating speed range is 700 ~ 3,600 [rpm]. More information about the testbed is described in [80]. In this research, imbalance and horizontal misalignment signals of 1,800 [rpm] are examined. Among the three datasets (D_1, D_2, and D_3), only the MAFAULDA signals are wavelet-denoised and low-pass filtered to remove unnecessary frequency components [81]; the cutoff frequency is set as 1000 [Hz]. The fundamental frequency (f_0) for both conditions is 30 [Hz]. For the imbalance condition, the fundamental frequency is dominant. The horizontal misalignment signal has many frequency components, including sub-harmonic components at $2f_0$, $3f_0$, $4f_0$, and $7f_0$. Since the signals of D_3 are much noisier than the others, it can be estimated that generating those signals will be the most difficult task.

The training, validation, and test data configurations are summarized in Table 4-4. For the simulated signal, the signal from 0 [sec] to 4 [sec] is used as the training data; the signal from 4 [sec] to 5 [sec] is employed as the validation data. Signals

from 0 [sec] to 1.50 [sec] and from 1.50 [sec] to 2.50 [sec] are used as the training and validation data for the RK4 dataset. In the case of MAFAULDA, training, and validation samples are determined as the samples from 0 [sec] to 1.00 [sec] and from 1.00 [sec] to 1.20 [sec], respectively. For each dataset, three test data samples with different time ranges are utilized to verify the proposed model. The size of each test data sample is chosen differently to verify the generation performance related to signals of variable lengths.

Table 4-5 shows the hyper-parameters of each dataset. For D_1 and D_2, batch size and N_f are selected as 512 and 1000, respectively. The frequency range is from 0 to 1000 [Hz], and the training epochs is chosen as 800. Since D_3 has the largest sampling rate among the three datasets, the batch size and N_f are set to be greater than their values for the other datasets. In particular, for each dataset, the batch size is set large enough to contain most sub-harmonic components of the fundamental frequency. The important hyper-parameter α is chosen by the grid search method [19]. α is selected from [0.001, 0.01, 0.1, 1.0, 10.0] to achieve the smallest MSE. Details of this process are summarized in Section 4.5.4.

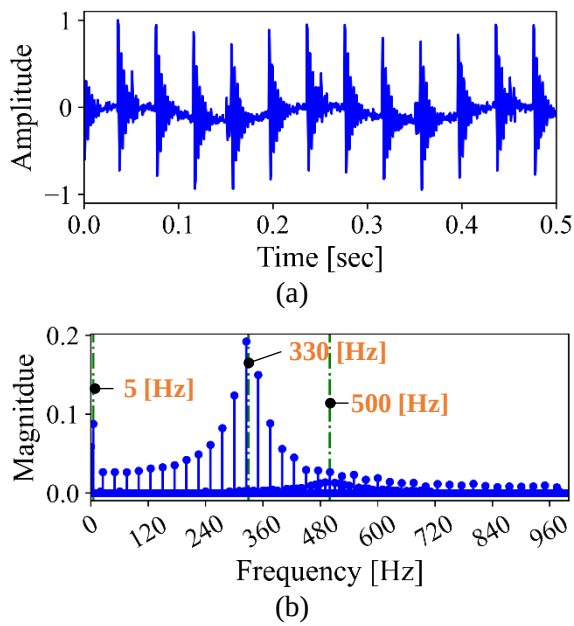
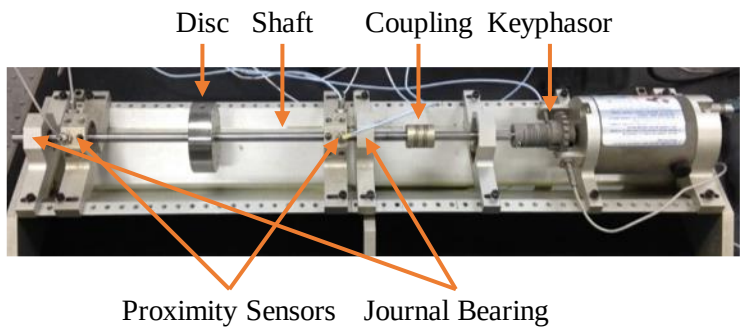
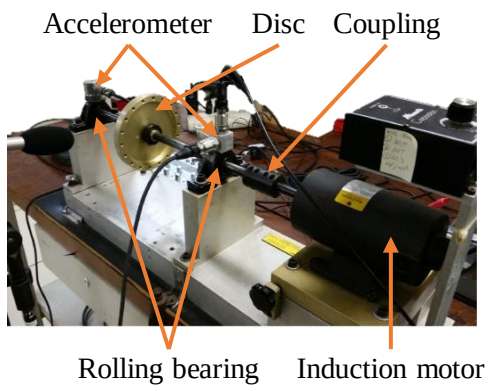


Figure 4-5 Time-domain and frequency-domain plots of the simulated signal: (a) time-domain and (b) magnitude spectrum



(a)



(b)

Figure 4-6 Testbed setups: (a) RK4 dataset and (b) MAFAULDA

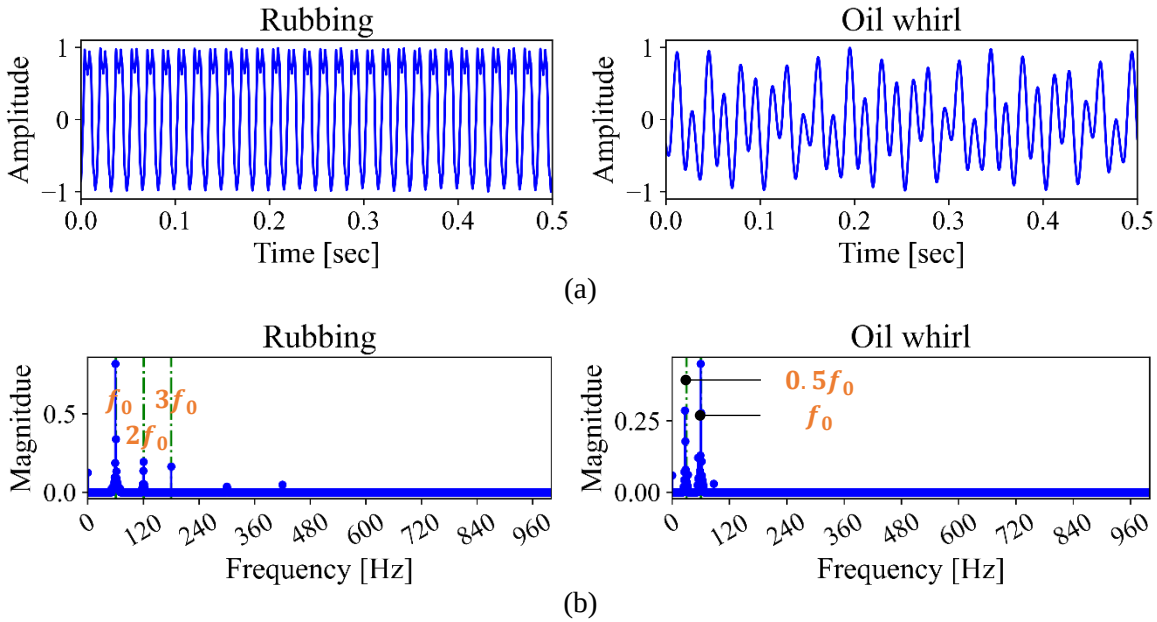


Figure 4-7 Time-domain and frequency-domain plots of the rubbing and oil whirl signals of the RK4 dataset: (a) time-domain trend and (b) magnitude spectrum

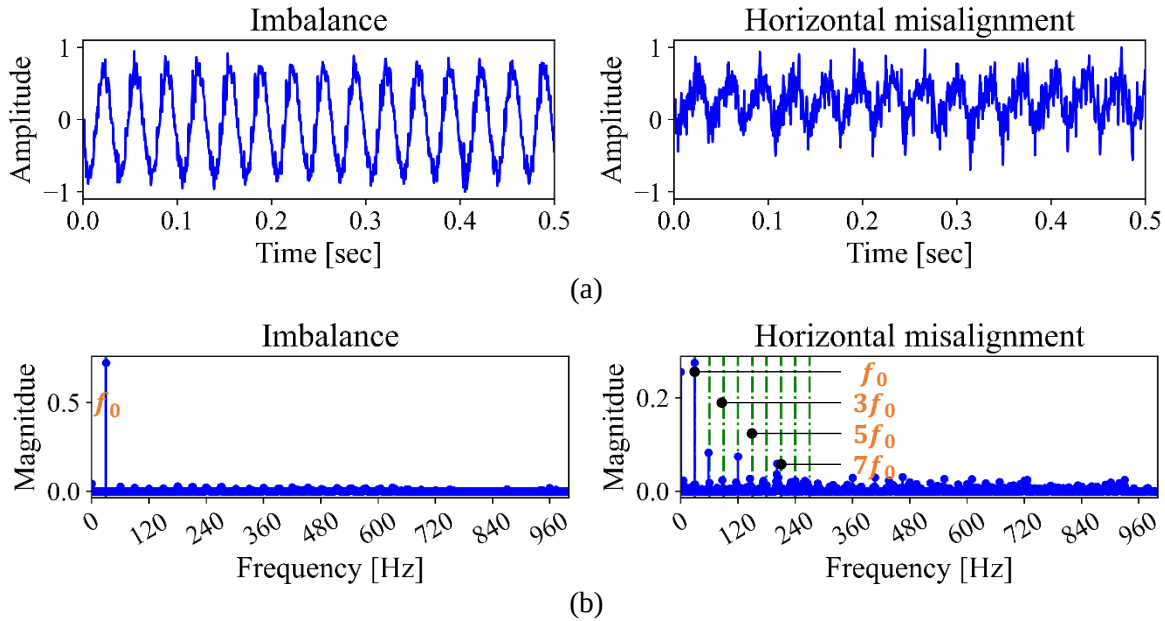


Figure 4-8 Time-domain and frequency-domain plots of the imbalance and horizontal misalignment signals of MAFAULDA: (a) time-domain trend and (b) magnitude spectrum

Table 4-4 Configuration of the training, validation, and test data of each dataset

Dataset	**Start time ~ end time [sec] (Number of samples)**				
	Training (Tr)	Validation (Val)	$Test_1$ (Te_1)	$Test_2$ (Te_2)	$Test_3$ (Te_3)
Simulated signal (D_1)	0.00 ~ 4.00 (8000)	4.00 ~ 5.00 (2000)	5.00 ~ 7.00 (4000)	7.00 ~ 8.50 (3000)	8.50 ~ 9.50 (2000)
RK4 dataset (D_2)	0.00 ~ 1.50 (12750)	1.50 ~ 2.50 (8500)	2.50 ~ 3.50 (8500)	3.50 ~ 4.25 (6375)	4.00 ~ 4.50 (4250)
MAFAULDA (D_3)	0.00 ~ 1.00 (51200)	1.00 ~ 1.20 (10240)	1.20 ~ 2.20 (51200)	2.20 ~ 2.70 (25600)	2.70 ~ 2.95 (12800)

Table 4-5 Hyper-parameters of each dataset

Dataset	**Batch size**	N_f	(p, q)	α	**Training epochs**
Simulated signal (D_1)	512	1000	(0, 1000)	0.1	800
RK4 dataset (D_2)	512	1000	(0, 1000)	0.01	800
MAFAULDA (D_3)	2048	1200	(0, 1000)	0.01	800

4.5 Validation of the Proposed Method

Three datasets are employed to validate the proposed approach. One is a simulated signal, which contains periodic impulsive signals and a signal with a low-frequency component. The second dataset is RK4 data, which was measured from a testbed of GE Bentley Nevada. The third dataset is a machinery fault database (MAFAULDA), which is an open dataset offered by [73]. The proposed FLGN is validated using the evaluation schemes presented in Section 4.3.2. The validation results show that the signals generated by FLGN are very similar to the true signals. Also, the results show that frequency components are successfully learned by the proposed method.

4.5.1 Case Study 1: Simulated Signal

The training and validation loss curves are analyzed first to confirm whether the training process is finished correctly; the curves are shown in Figure 4-9. In the figure, the blue line is the loss curve of the training data, and the red-dotted line is that of the validation data. The y-axis of the figure is limited to exist between 0 and 0.20. As shown in the figure, the validation loss decreases as the training loss decreases; further, both losses converge when the training is almost over. Since the validation loss does not increase while the training loss decrease, it can be concluded that an overfitting problem does not occur. Moreover, it seems that the losses slowly decrease during 10 ~ 100 epochs. This infers that FLGN does not initially learn the dominant frequency; however, it can learn the correct frequency as the training proceeds.

In addition to the loss curves, the validation batch samples are compared. Figure

4-10 describes the validation samples at the 20^{th}, 400^{th}, and 780^{th} epochs. In the figure, the blue line denotes the true sample, and the red line is the generated sample. Initially, the generated signal is very different from the true sample; the impulsive components are not captured in the generated signal. At the 400^{th} epoch, though a more similar signal is generated, there are still some errors in the generated signal. However, the errors decrease further as training progresses, and the sample generated by FLGN at the 780^{th} epoch is almost identical to the true sample.

For the three test data samples (Te_1, Te_2, and Te_3) in Table 4-4, the generated signal is compared with the true signal (as shown in Figure 4-11) by visualizing them in the time domain and in the frequency domain, respectively. The blue line means the true signal, and the red line denotes the produced signal. As can be seen from the time-domain results, the generated signal is almost the same as the true signal in all cases. Two periodic impulse signals and the low-frequency component of 5 [Hz] are learned well. Also, it can be found that FLGN can generate signals well even when the lengths of Te_1, Te_2, and Te_3 are varied. This cannot be achieved by conventional VAE or GAN-based models; these prior models can only produce a signal that has the same size as that of the final hidden layer of the generator. The magnitude spectrum results of the generated and true signals are also similar in all cases. Furthermore, the characteristic frequencies – 5 [Hz], 330 [Hz], and 500 [Hz] – are learned well by FLGN.

Next, the similarity metrics – RMSE and correlation coefficient – are computed and shown in Figure 4-12. For each data sample, the metrics are calculated based on the true signal and the generated signal. The blue line with triangles presents the RMSE curve, and the red line with circles denotes the curve of the correlation

coefficient. The RMSE value of the training data is the smallest, and that of Te_3 is the greatest. Specifically, the further away from the time range of the training data, the greater the RMSE value. This is natural because the performance of a deep-learning algorithm usually degrades as the input data becomes more dissimilar to the training data. For the correlation coefficient, meanwhile, the coefficient remains at about 1.0 for all data. This means that the generated signals are highly correlated with the true signals. This can be interpreted to mean that the generated signals are nearly identical to the true ones.

The features in Table 4-3 are calculated and shown in Figure 4-13. Here, the red bar with downward lines means the feature of the true signal, and the gray bar with upward lines presents that of the generated signal. First, since RMS, skewness, and kurtosis are similar to each other, the generated signal has kinetic energy and statistical characteristics that are similar to the true signal. The shape factor, impulse factor, and crest factor of the generated signals are also similar to those of the true signals. This indicates that the sinusoidal characteristics are similar. Finally, since the frequency center, RMSF, and RVF of both signals are almost the same, it can be argued that the frequency components are also similar.

The performance of the developed method is verified using an auto-encoder; the architecture of the auto-encoder is illustrated in Figure 4-4(b). Figure 4-14 presents the visualization of the latent vectors, which are encoded from the true and generated signals. The red circle denotes the latent vectors of the true signals, and the blue x-marker means those of the generated signals. As can be seen from the figure, the latent vectors of the generated signals are close to those of the true signals. This means that the produced signals are similar to the true signals. This can also be

found in Table 4-6, which shows the RMSE and the correlation coefficient between the signals reconstructed from the true and the generated signals. RMSE values are small, and the coefficient values are around 1. This quantitatively verifies that the proposed method can produce signals that are similar to the target signals.

In summary, the signal produced by FLGN is similar to the true signal when comparing the results in the time domain and in the frequency domain. The generated signal is highly correlated to the true one, and the handcrafted features of both signals are significantly similar to each other. Also, the proposed method can generate signals of variable lengths well. Therefore, it can be said that the proposed FLGN produces a signal of variable length that is similar to the target signal.

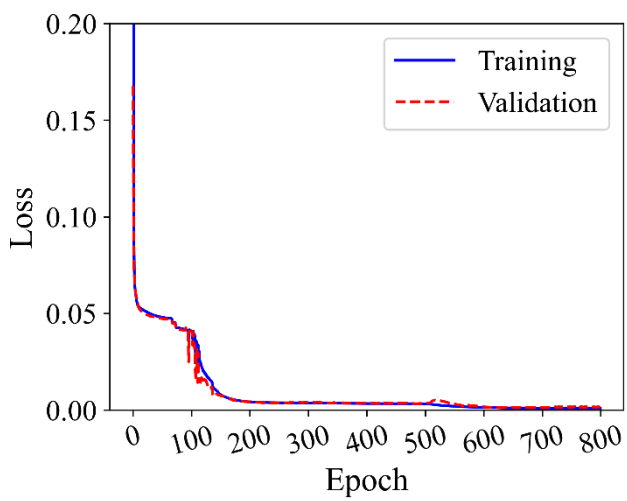


Figure 4-9 Training and validation loss curves in Case 1

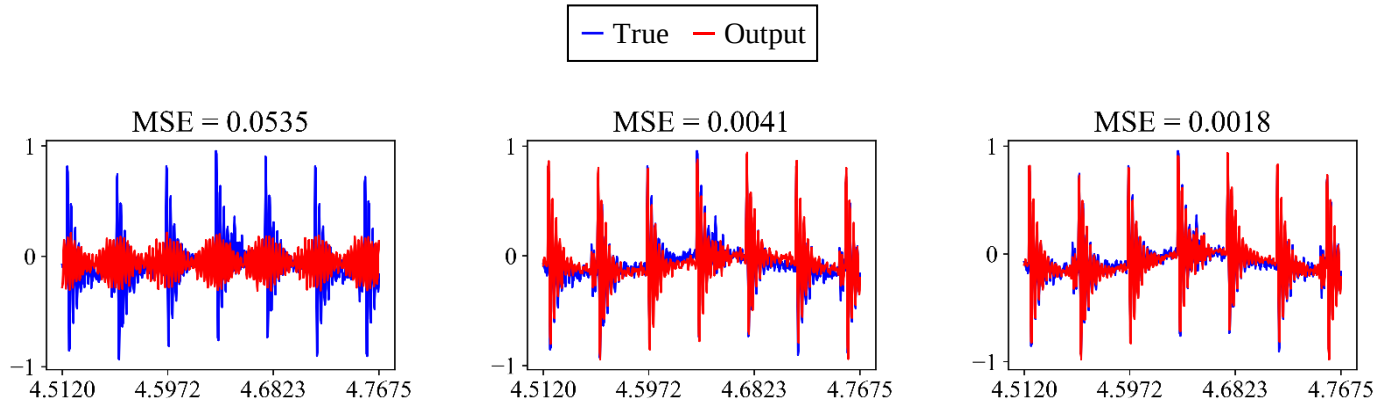


Figure 4-10 Time-domain visualization of validation batch samples for epochs in Case 1: (a) 20th epoch, (b) 400th epoch, and (c) 780th epoch

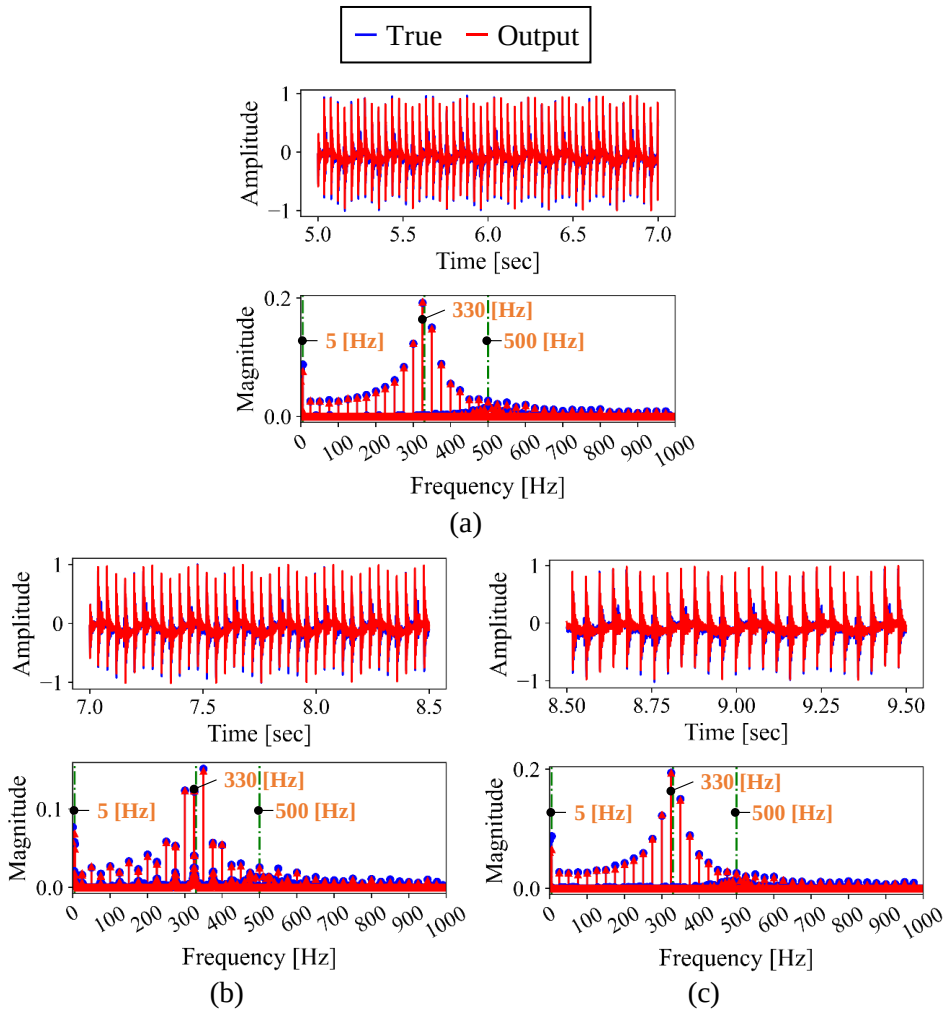


Figure 4-11 Time-domain trend and magnitude spectrum of each test data in Case 1: (a) Te_1, (b) Te_2, and (c) Te_3

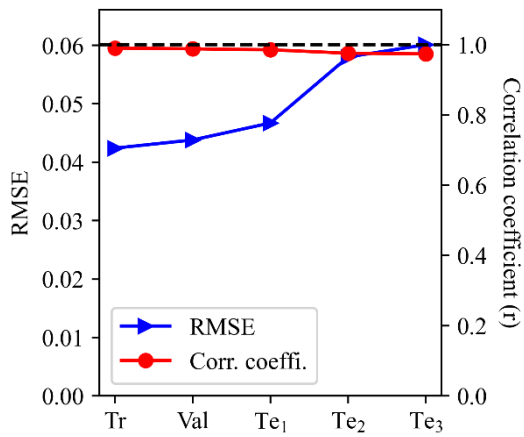


Figure 4-12 Similarity metric curves in Case 1

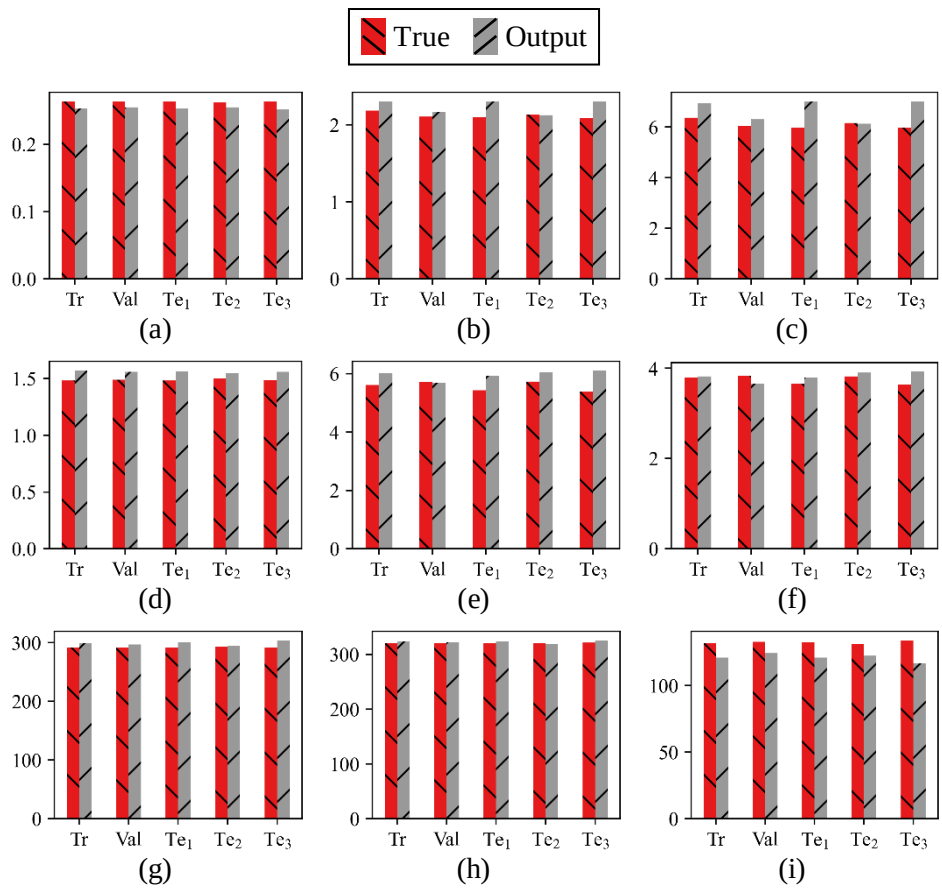


Figure 4-13 Time-domain and frequency-domain features in Case 1: (a) RMS, (b) skewness, (c) kurtosis, (d) shape factor, (e) impulse factor, (f) crest factor, (g) frequency center, (h) RMSF, and (i) RVF

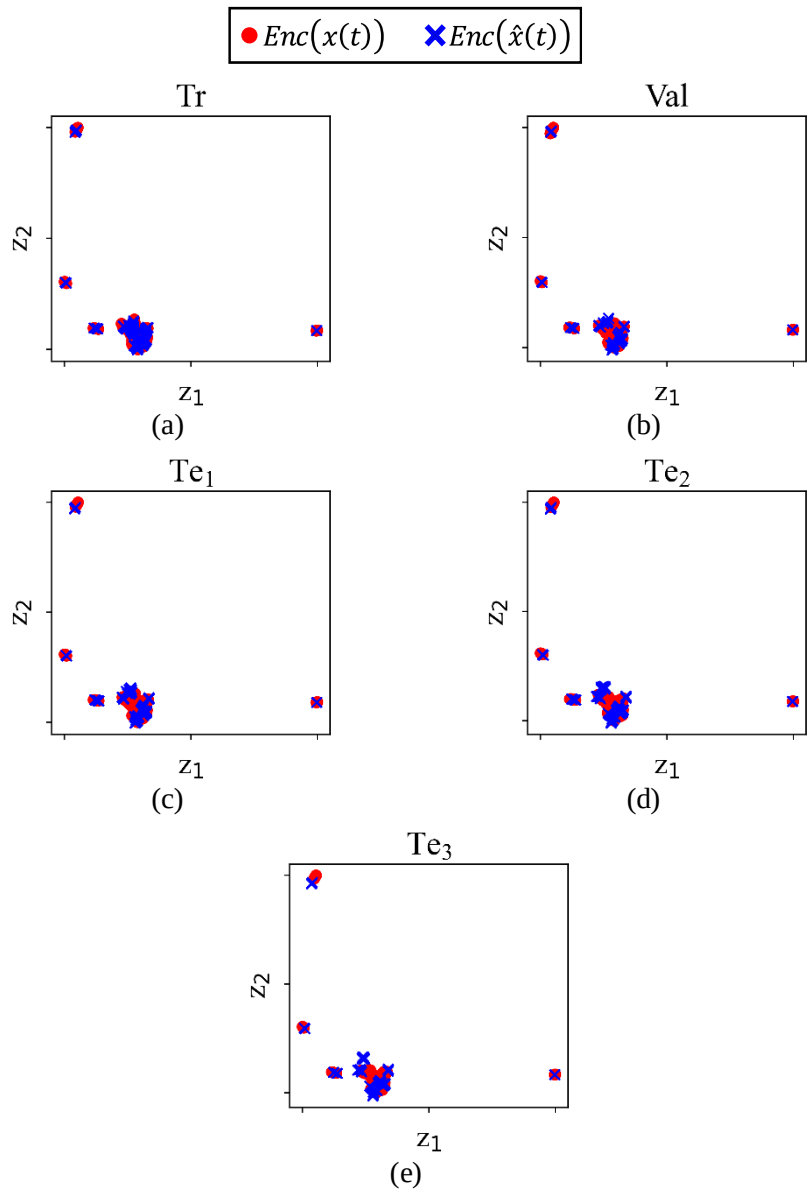


Figure 4-14 Visualization of the latent vectors in Case 1: (a) Tr, (b) Val, (c) Te_1, (d) Te_2, and (e) Te_3

Table 4-6 RMSE and correlation coefficient between the signals reconstructed from the true and generated signals in Case 1

Data	RMSE	Correlation coefficient
Tr	0.0160	0.9994
Val	0.0271	0.9983
Te_1	0.0349	0.9974
Te_2	0.0445	0.9957
Te_3	0.0548	0.9939

4.5.2 Case Study 2: RK4 Testbed Dataset

Figure 4-15 shows the training and validation loss curves for the rubbing and oil whirl conditions. The range of the y-axis is constrained to exist between 0 and 0.20. For the rubbing condition, the training and validation losses decrease gradually until the 200^{th} epoch; then, both losses remain constant until the 350^{th} epoch. After that, both losses converge at the end of the training procedure. The losses of the oil whirl condition decrease with fluctuation until the 200^{th} epoch; after that, both losses decrease and converge gradually. For both conditions, since the gap between the final training and validation losses is small enough, it can be concluded that any overfitting issue is not severe.

The batch samples of validation data are compared with the generated samples while the training procedure progresses. Figure 4-16 presents both signals at the 20^{th}, 400^{th}, and 780^{th} epochs of the rubbing and oil whirl conditions. The legend is the same as that of Figure 4-10. The generated signals are not similar to the true signals initially for either condition. However, as the training procedure proceeds, the

generated signals are almost the same as the true ones. This means that FLGN is trained well for both conditions.

The results of the three test data samples – Te_1, Te_2, and Te_3 – are described in Figure 4-17. The legend is the same as that of Figure 4-11 of Case 1. As you can see from the results of the rubbing condition, the generated signal is very similar to the true signal in the time domain. The magnitude spectrum results are also similar to each other. The fundamental frequency (f_0 = 60 [Hz]) and the fault-related frequencies ($2f_0$ and $3f_0$) are identical for all cases. For the oil whirl condition, the signals produced by FLGN are similar to the true signals in both domains. In particular, the generated signals have characteristic frequency components at f_0 and $0.5f_0$. However, unlike the rubbing condition, the error between the generated and true signals is greater. This is because the signal of the oil condition has a greater spectral smearing effect than that of the rubbing condition; consequently, it is more difficult to learn the frequency components for the oil whirl condition.

The RMSE and correlation coefficient between the true signal and the generated signal are calculated and presented in Figure 4-18. The legend is identical to that of Figure 4-12 of Case 1. In both conditions, the training data has the smallest RMSE value, and the RMSE value increases as the input time range moves farther away from the training time range. The reason for this phenomenon is the same as that described in Case 1; that is, the performance of a neural network often deteriorates as the test data becomes increasingly different from the training data. Examining the correlation coefficient curves, we find that the coefficient of the rubbing condition is almost 1.0, and that of the oil whirl condition is greater than 0.8. This means that the generated signal is significantly correlated to the true signal.

To further validate the similarity between the true data and the generated data, the features in Table 4-3 are computed to examine the similarity between both signals. Figures 4-19 and 4-20 show the results of the rubbing and oil whirl conditions, respectively. The true and generated signals share similar RMS values, which means that the energy of the signals is similar. The skewness and kurtosis of both signals are also similar. This indicates that the statistical properties are also similar. Also, since the shape factor, impulse factor, and crest factor of the produced signals are similar to those of the true ones, it can be confirmed that the produced signals have sinusoidal properties that are similar to those of the true signals. Furthermore, both signals have very similar frequency center, RMSF, and RVF values. This means that the dominant frequency information is almost identical when comparing the true and generated signals.

The proposed method is evaluated using an auto-encoder. Figures 4-21 and 4-22 show the visualization of the latent vectors of the rubbing and oil whirl conditions, respectively; the legend is the same as that of Figure 4-14. For most cases, the latent vectors of the signals generated by the FLGN method overlap those of the true signals. Specifically, for the oil whirl condition, the latent vectors of the produced signals are close to those of the true signals, even if some of the latent vectors spread out. The RMSE and correlation coefficient values of both conditions are summarized in Tables 4-7 and 4-8, respectively. As can be seen from the tables, RMSE values are small, and the correlation coefficient values are close to 1. This means that the generated signals are statistically similar to the true signals.

In summary, we validated the generation performance of FLGN for Case 2 in various ways. The validation results show that the proposed model is able to produce

signals of different lengths well. For both conditions, the generated signals are significantly correlated to the true ones and have similar handcrafted features.

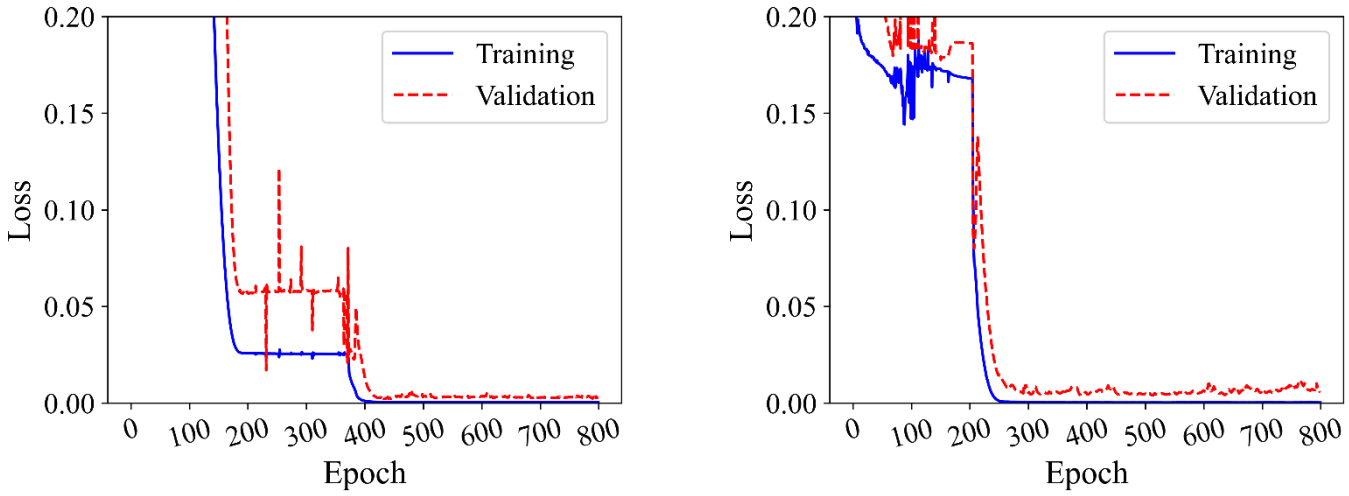


Figure 4-15 Training and validation loss curves in Case 2: (a) rubbing and (b) oil whirl

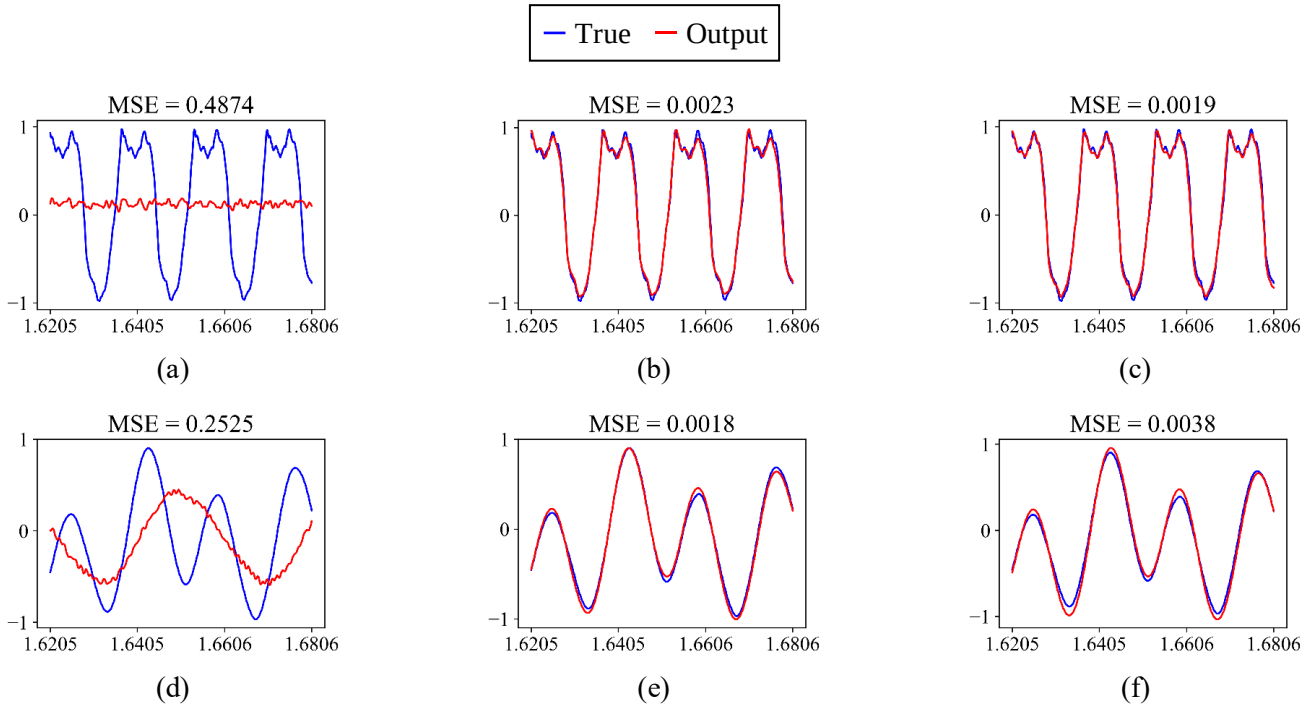
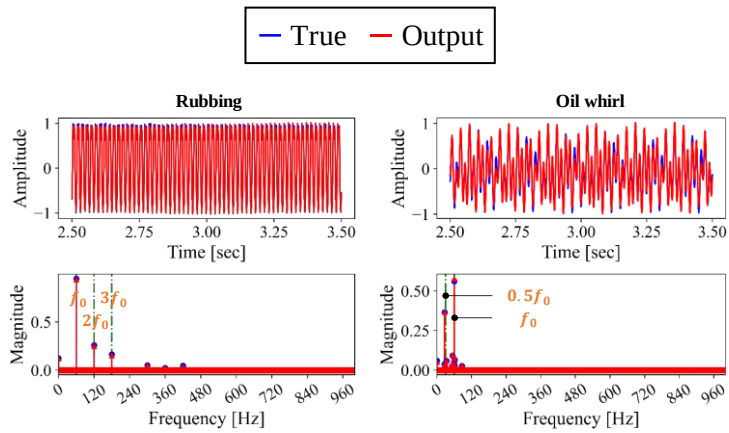
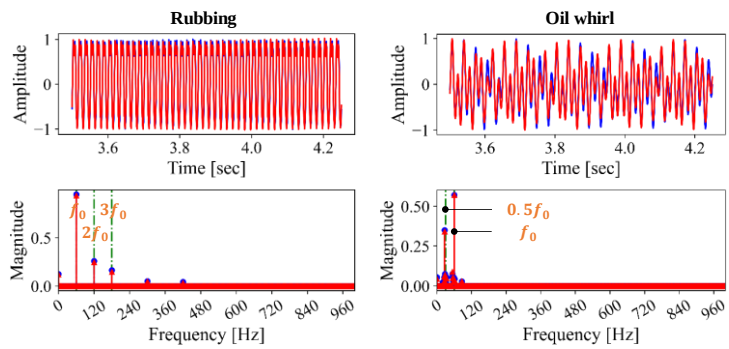


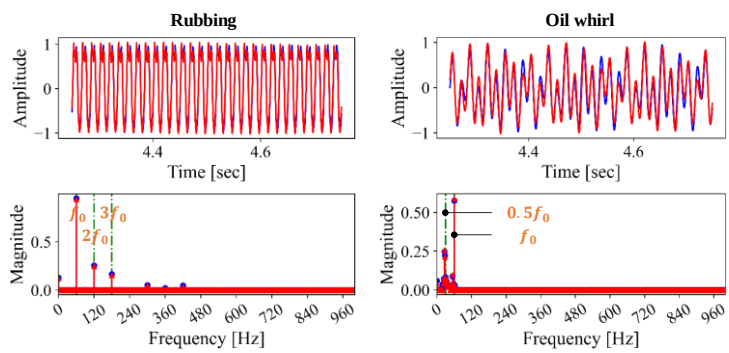
Figure 4-16 Time-domain visualization of validation batch samples for various epochs in Case 2: (a-c) 20th, 400th , and 780th epochs of rubbing and (d-f) 20th , 400th , and 780th epoch of oil whirl



(a)



(b)



(c)

Figure 4-17 Time-domain trend and magnitude spectrum of each test data in Case 2: (a) Te_1, (b) Te_2, and (c) Te_3

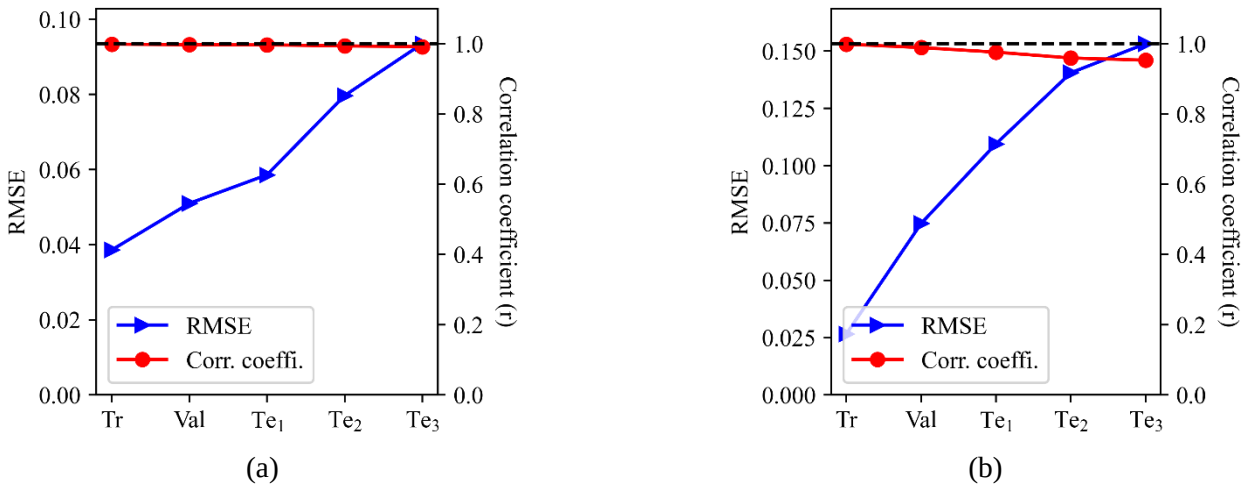


Figure 4-18 Similarity metric curves in Case 2: (a) rubbing and (b) oil whirl

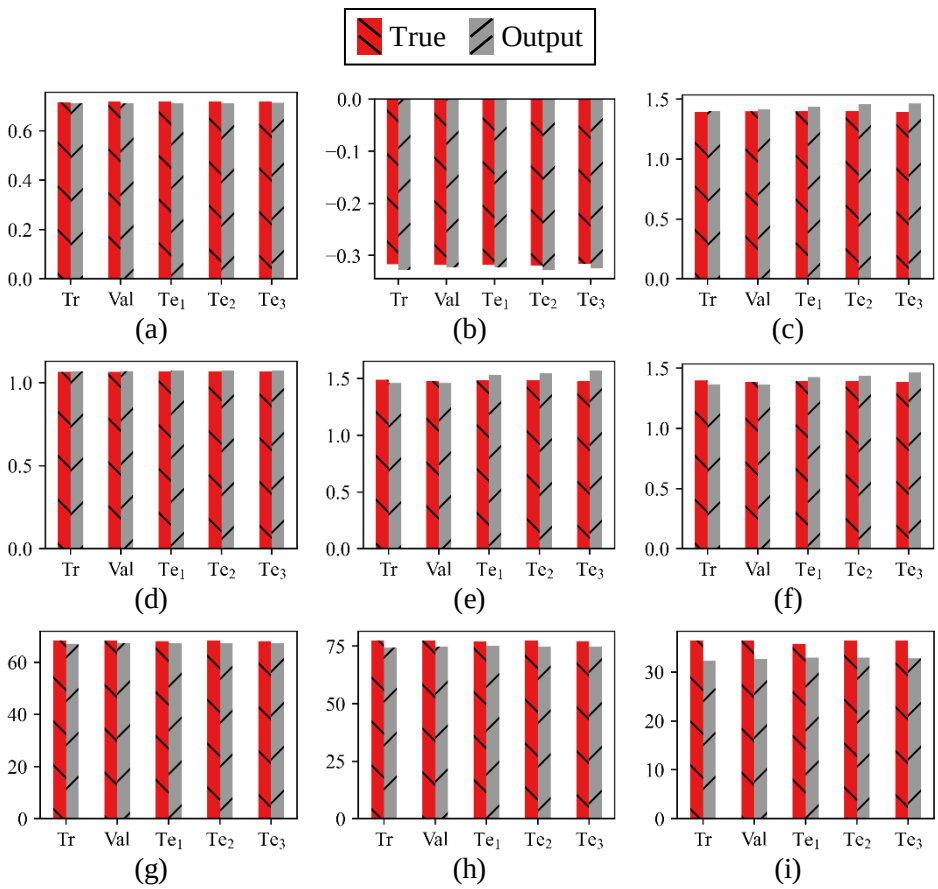


Figure 4-19 Time-domain and frequency-domain features of the rubbing condition in Case 2: (a) RMS, (b) skewness, (c) kurtosis, (d) shape factor, (e) impulse factor, (f) crest factor, (g) frequency center, (h) RMSF, and (i) RVF

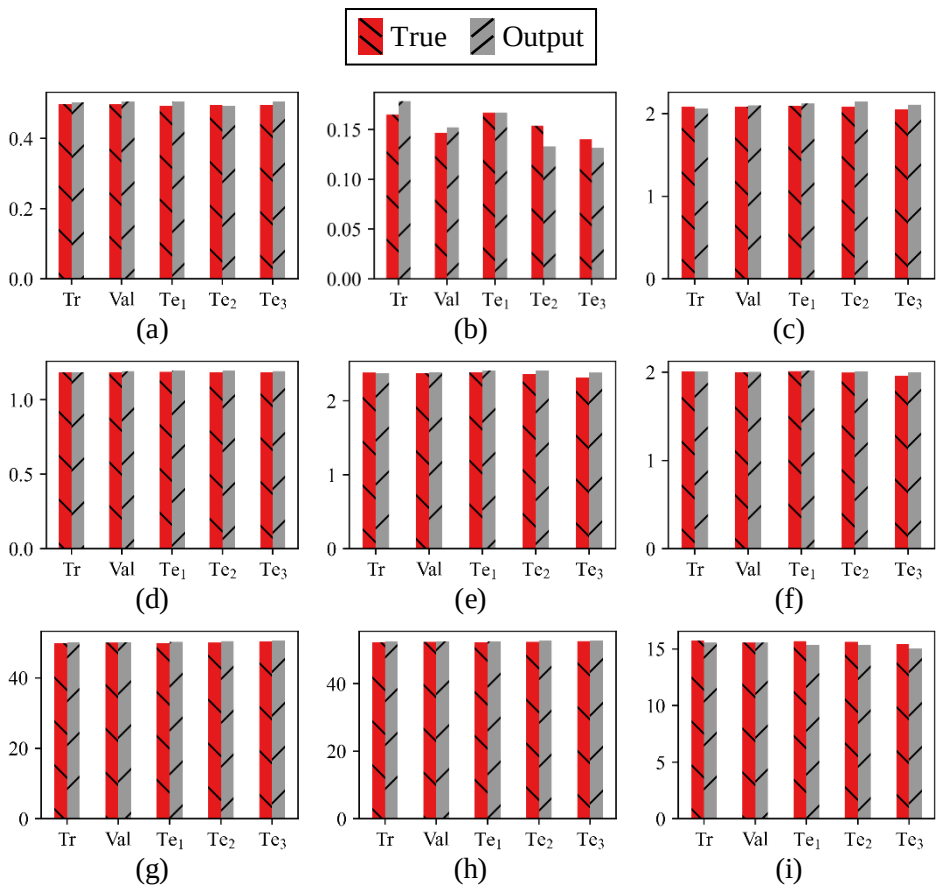


Figure 4-20 Time-domain and frequency-domain features of the oil whirl condition in Case 2: (a) RMS, (b) skewness, (c) kurtosis, (d) shape factor, (e) impulse factor, (f) crest factor, (g) frequency center, (h) RMSF, and (i) RVF

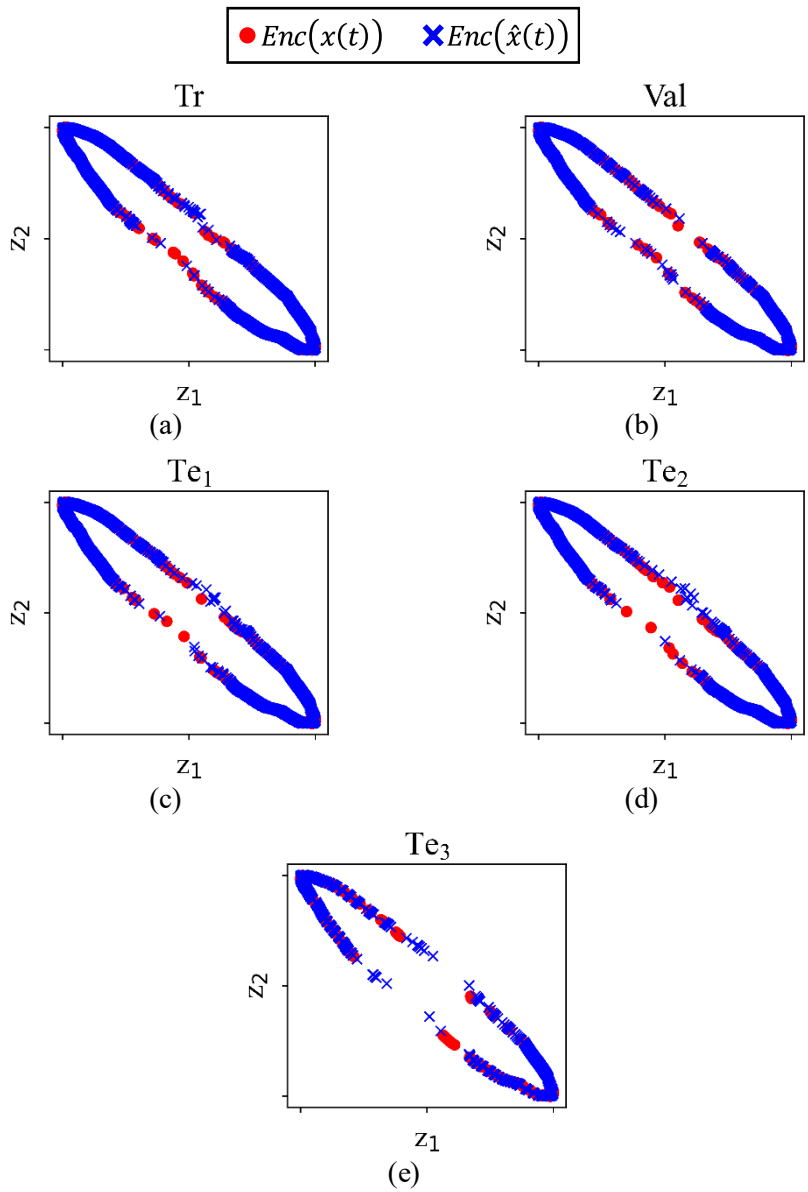


Figure 4-21 Visualization of the latent vectors of the rubbing condition in Case 2: (a) Tr, (b) Val, (c) Te_1, (d) Te_2, and (e) Te_3

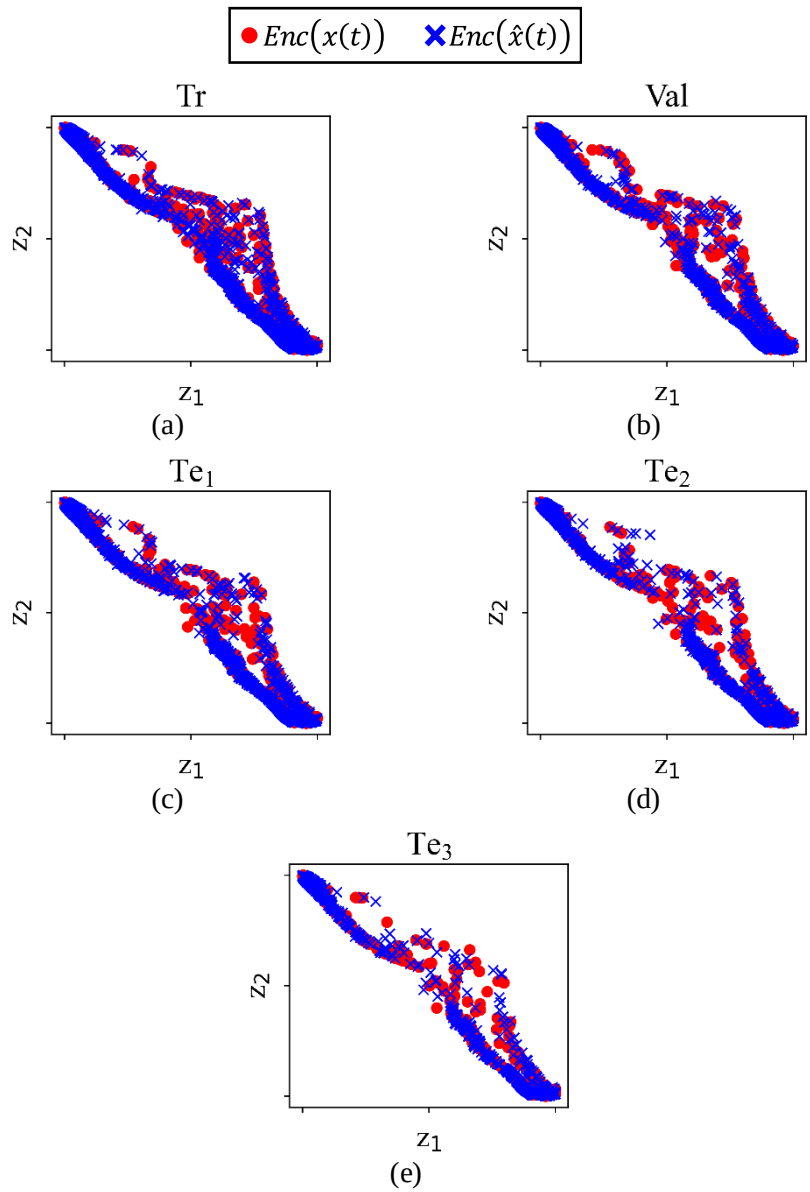


Figure 4-22 Visualization of the latent vectors of the oil whirl condition in Case 2: (a) Tr, (b) Val, (c) Te_1, (d) Te_2, and (e) Te_3

Table 4-7 RMSE and correlation coefficient between the signals reconstructed from the true and generated signals of the rubbing condition in Case 2

Data	RMSE	Correlation coefficient
Tr	0.0270	0.9989
Val	0.0368	0.9982
Te_1	0.0426	0.9980
Te_2	0.0600	0.9959
Te_3	0.0694	0.9951

Table 4-8 RMSE and correlation coefficient between the signals reconstructed from the true and generated signals of the oil whirl condition in Case 2

Data	RMSE	Correlation coefficient
Tr	0.0471	0.9937
Val	0.1066	0.9786
Te_1	0.1506	0.9641
Te_2	0.1885	0.9517
Te_3	0.2082	0.9473

4.5.3 Case Study 3: MAFAULDA

In this case, the signals of the imbalance and horizontal misalignment conditions are studied. The training and validation loss curves are shown in Figure 4-23. As shown, the training and validation losses decrease smoothly for the imbalance condition. For the horizontal misalignment condition, both losses also converge; however, there is much fluctuation. This is because the signal of the horizontal misalignment condition has more complex frequency components than that of the imbalance condition; this can be confirmed by examining the results in Figure 4-8. Since the difference between the training and validation losses is low enough at the end of the training procedure, it can be concluded that the overfitting problem is not severe in either condition.

Figure 4-24 describes the generated and true signals at the 20^{th}, 400^{th}, and 780^{th} epochs for the imbalance and horizontal misalignment conditions. The legend is identical to that of Figures 4-10 and 4-16. Also, the results are similar to those shown for Case 1 and Case 2. Though the generated signals are not similar to the true signals initially, they become similar to the true samples as the training procedure progresses. Even if there is much noise in the true signal, as found for Case 1 and Case 2, FLGN can effectively produce similar signals to the true signals.

The results of the test data are presented in Figure 4-25. The legend is identical to that of Figures 4-11 and 4-17. In the imbalance condition, the generated and true signals are similar to each other, when comparing them in the time domain. Also, both signals have an identical fundamental frequency component (f_0 = 30 [Hz]). In the horizontal misalignment condition, although the true signal has many sub-harmonic signals at $n \times f_0$ $(n = 2, \dots, 9)$, it is found that FLGN can learn most sub-

harmonic signals and that the generated signals are similar to the true signals.

Next, the similarity metrics are computed to quantitatively evaluate the degree to which the generated signal is similar to the true signal. Figure 4-26 presents the results for the imbalance and horizontal misalignment conditions. The legend is the same as that of Figures 4-11 and 4-16. For the imbalance condition, though the RMSE value of the test data is larger than that of the training data, the correlation coefficient remains greater than 0.9. This means that the generated signal is highly correlated to the true signal. This is also found in the results of the horizontal misalignment condition. The coefficient is larger than 0.7, while the RMSE value of the test data also increases compared to the training data. The gap between the test and training data is wider in the horizontal misalignment case. This is because there are more sub-harmonic components and noise components than in the imbalance condition. The phenomenon where the RMSE value increases from the training data to test data is also recognized, and the reason is estimated to be the same.

The handcrafted features of the imbalance and horizontal conditions are calculated and shown in Figures 4-27 and 4-28, respectively. Like Case 1 and Case 2, most features of the produced signals are similar to the true ones. However, in some features, including the impulse factor, frequency center, RMSF, and RVF, the gap between the generated and the true signals is greater than those of Case 1 and Case 2. This is because dataset D_3 has more sub-harmonic and noise components.

The generation performance of the FLGN method is verified through the use of an auto-encoder. The latent vectors of the imbalance and horizontal misalignment conditions are visualized in Figure 4-29 and Figure 4-30, respectively. The legends

are identical to those of Figures 4-21 and 4-22. Although the latent vectors of the true signals are more complicated than those of Case 1 and Case 2, those of the generated signals are similar to the true signals. This is also discovered in Tables 4-9 and 4-10, which present the RMSE and correlation coefficient values of both conditions. The RMSE values are less than 0.22, and the correlation coefficient values are greater than 0.96 for both conditions. This proves that the signals generated by the proposed approach are similar to the true signals.

In conclusion, when validating the proposed method by applying it to the MAFAULDA, not only does the proposed method have the ability to learn the frequency information well, but it can also generate signals of variable lengths well.

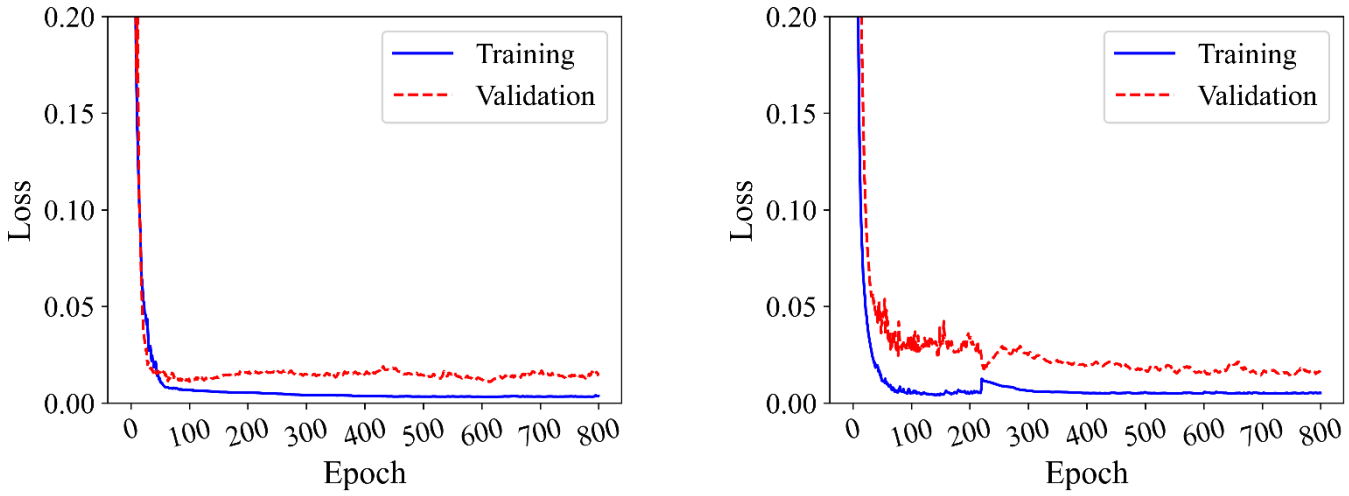


Figure 4-23 Training and validation loss curves in Case 3: (a) imbalance and (b) horizontal misalignment

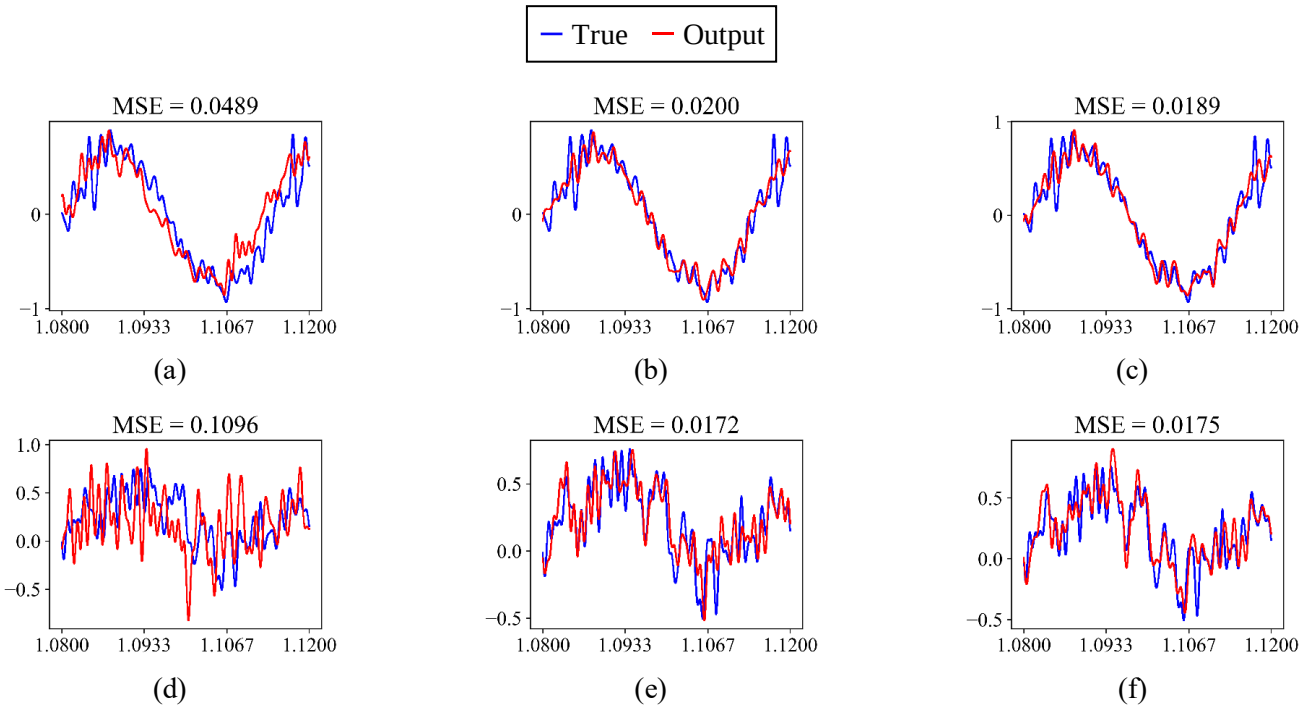
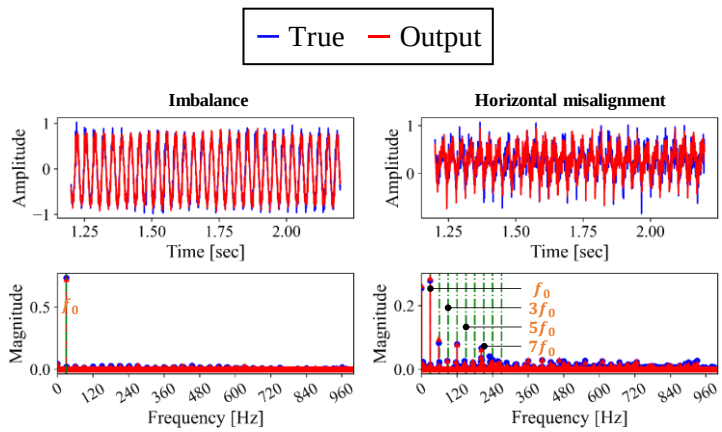
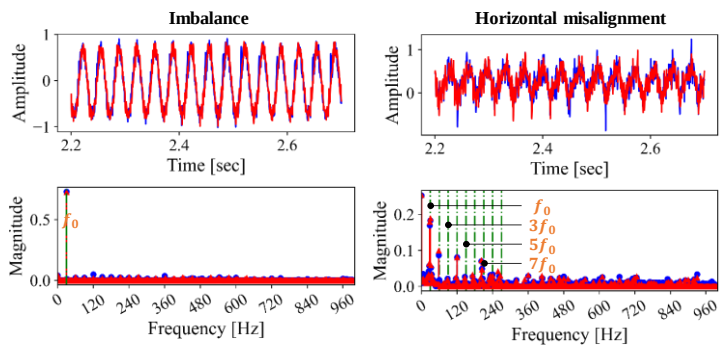


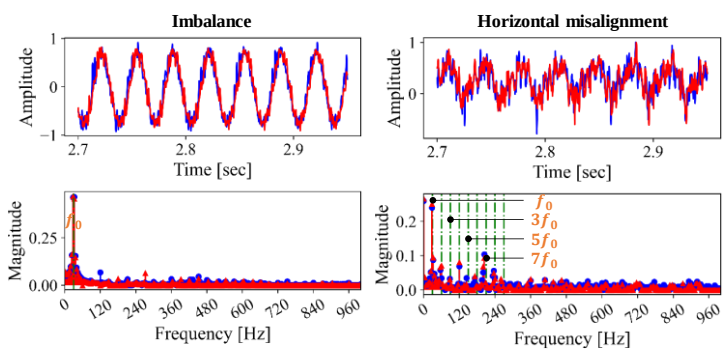
Figure 4-24 Time-domain visualization of validation batch samples for various epochs in Case 3: (a-c) 20th, 400th , and 780th epochs of imbalance and (d-f) 20th , 400th , and 780th epoch of horizontal misalignment



(a)



(b)



(c)

Figure 4-25 Time-domain trend and magnitude spectrum of each test data in Case 3: (a) Te_1, (b) Te_2, and (c) Te_3

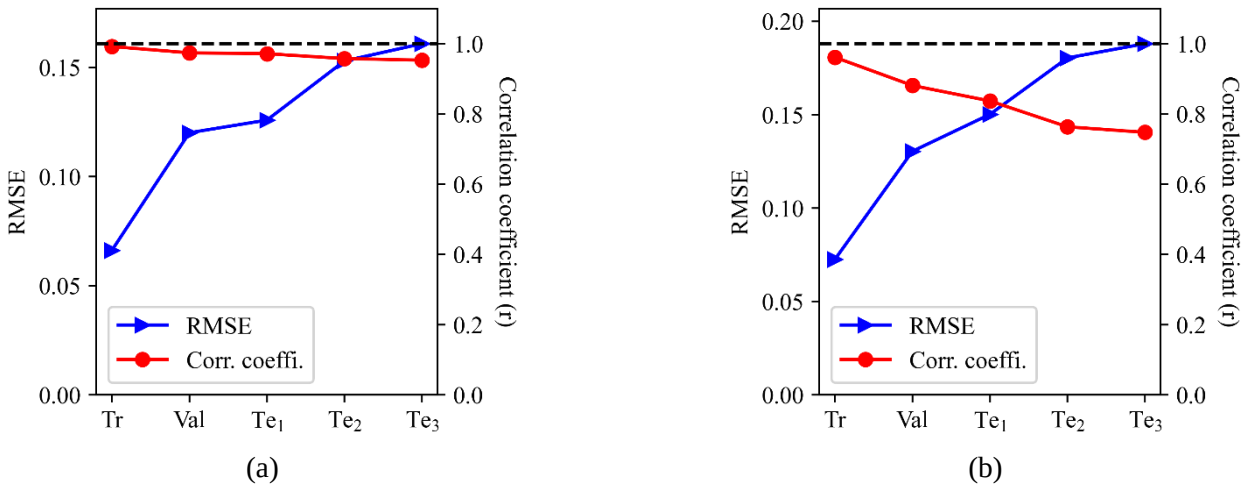


Figure 4-26 Similarity metric curves in Case 3: (a) imbalance and (b) horizontal misalignment

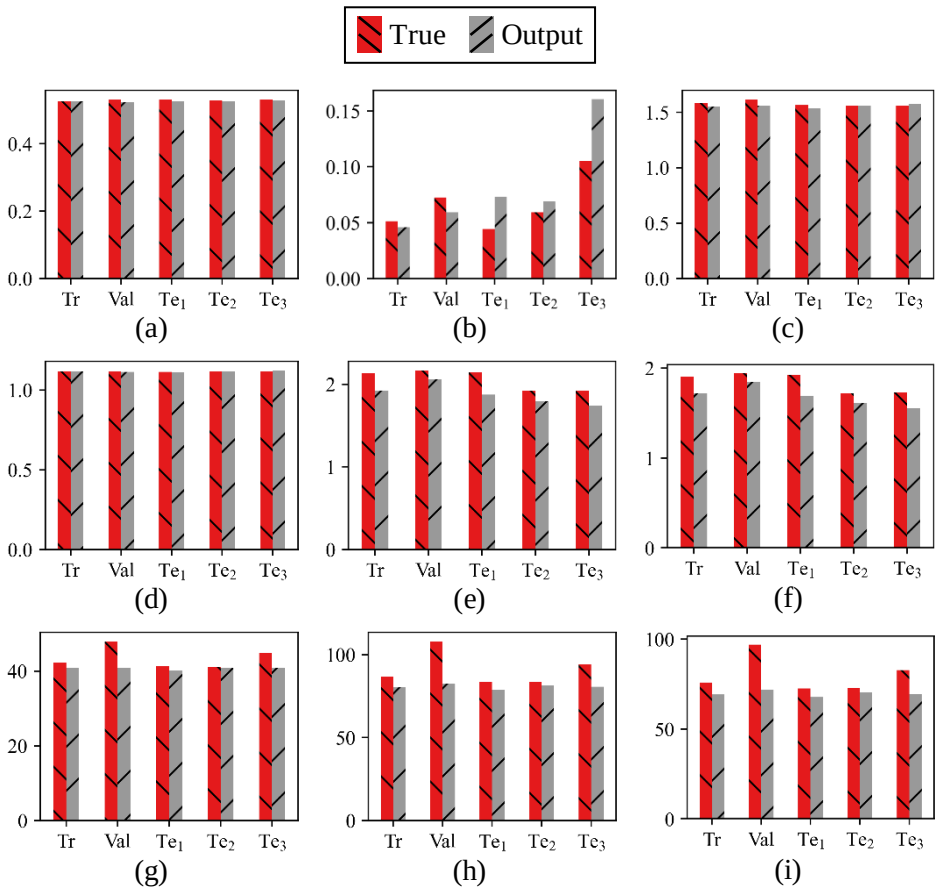


Figure 4-27 Time-domain and frequency-domain features of the imbalance condition in Case 3: (a) RMS, (b) skewness, (c) kurtosis, (d) shape factor, (e) impulse factor, (f) crest factor, (g) frequency center, (h) RMSF, and (i) RVF

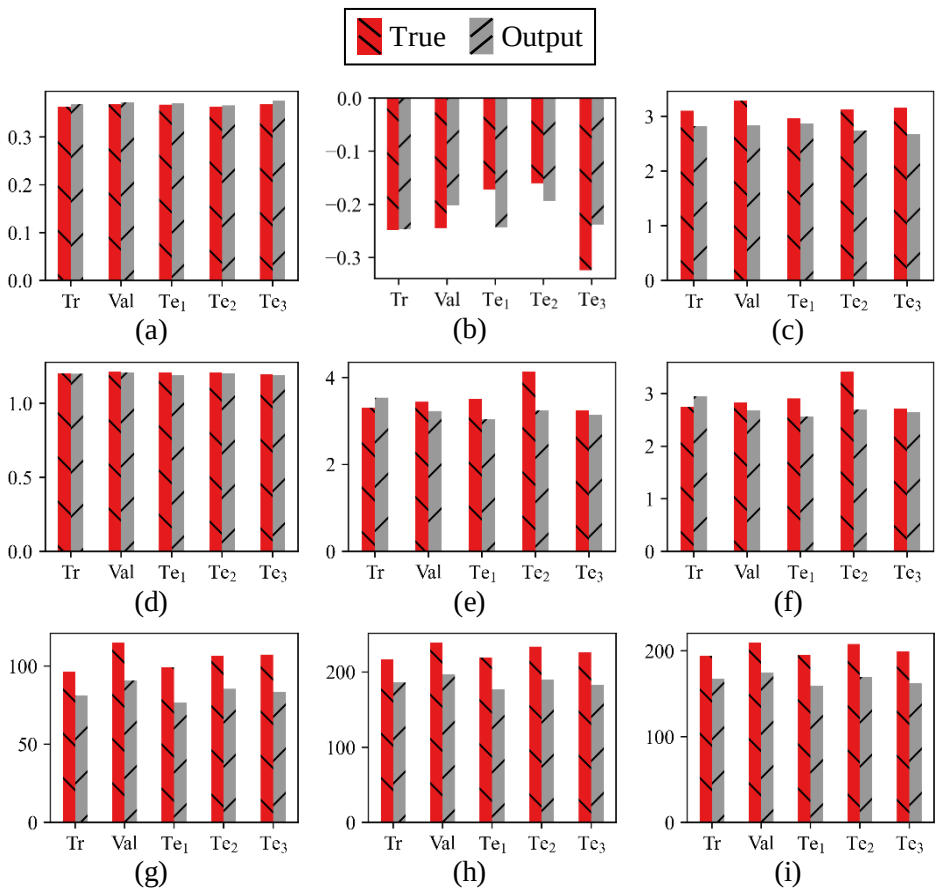


Figure 4-28 Time-domain and frequency-domain features of the horizontal misalignment condition in Case 3: (a) RMS, (b) skewness, (c) kurtosis, (d) shape factor, (e) impulse factor, (f) crest factor, (g) frequency center, (h) RMSF, and (i) RVF

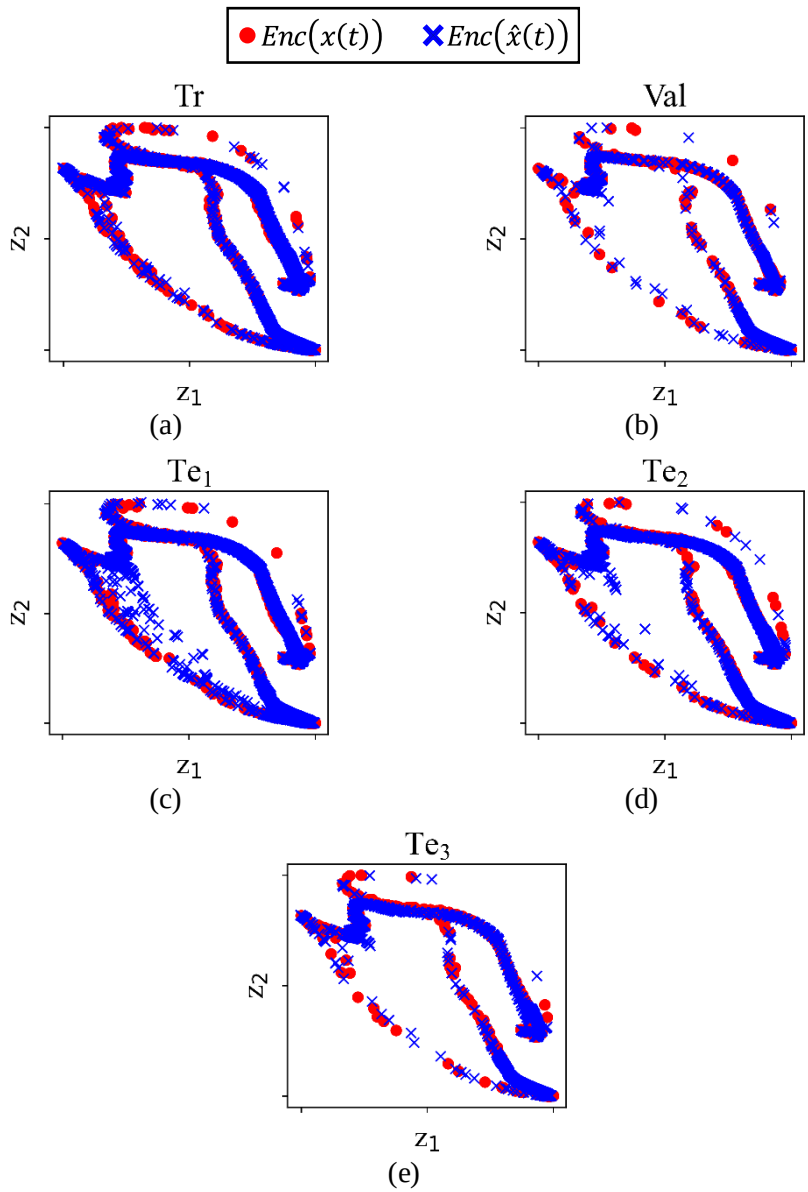


Figure 4-29 Visualization of the latent vectors of the imbalance condition in Case 3: (a) Tr, (b) Val, (c) Te_1, (d) Te_2, and (e) Te_3

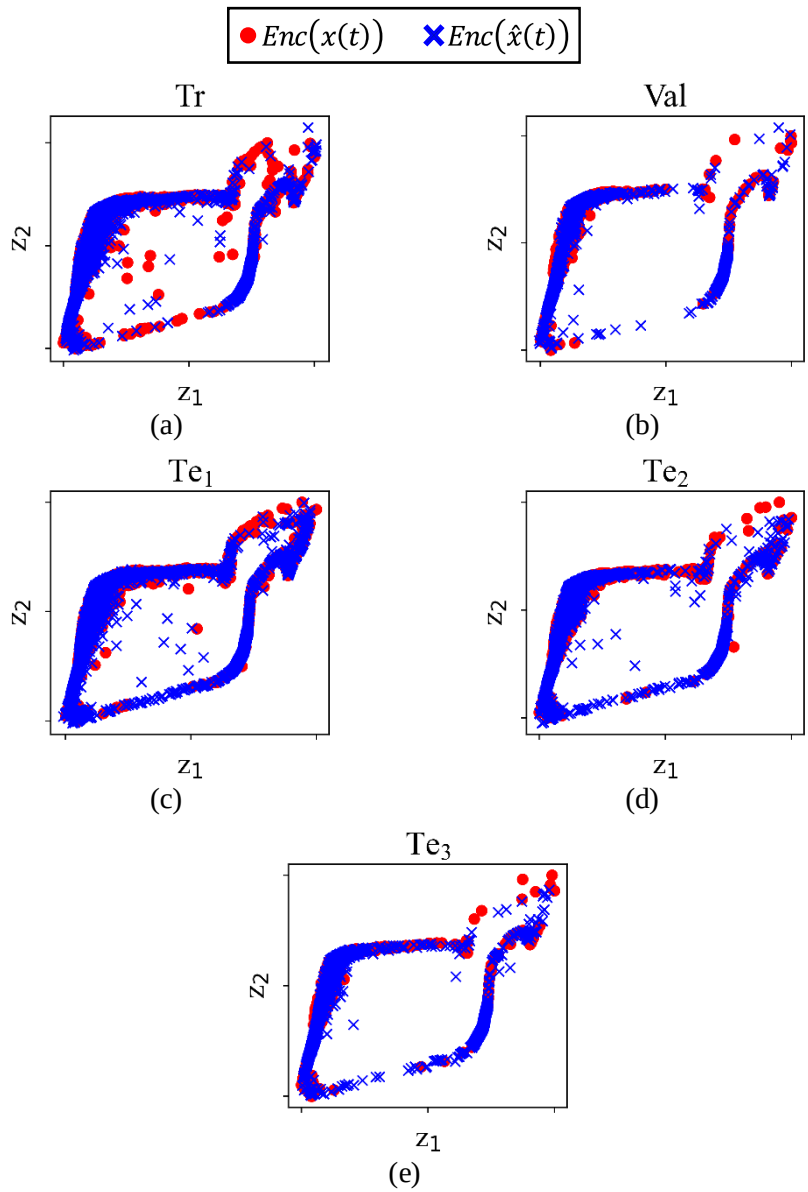


Figure 4-30 Visualization of the latent vectors of the horizontal misalignment condition in Case 3: (a) Tr, (b) Val, (c) Te_1, (d) Te_2, and (e) Te_3

Table 4-9 RMSE and correlation coefficient between the signals reconstructed from the true and generated signals of the imbalance condition in Case 3

Data	RMSE	Correlation coefficient
Tr	0.0281	0.9985
Val	0.0582	0.9951
Te_1	0.0653	0.9933
Te_2	0.1631	0.9676
Te_3	0.1849	0.9620

Table 4-10 RMSE and correlation coefficient between the signals reconstructed from the true and generated signals of the horizontal misalignment condition in Case 3

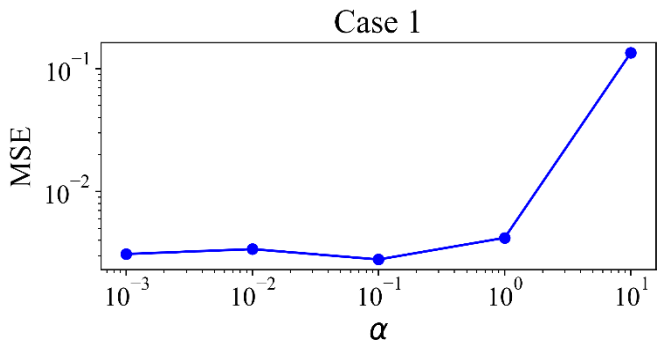
Data	RMSE	Correlation coefficient
Tr	0.0678	0.9922
Val	0.1539	0.9836
Te_1	0.1582	0.9765
Te_2	0.1917	0.9741
Te_3	0.2149	0.9716

4.5.4 Analysis and Discussion

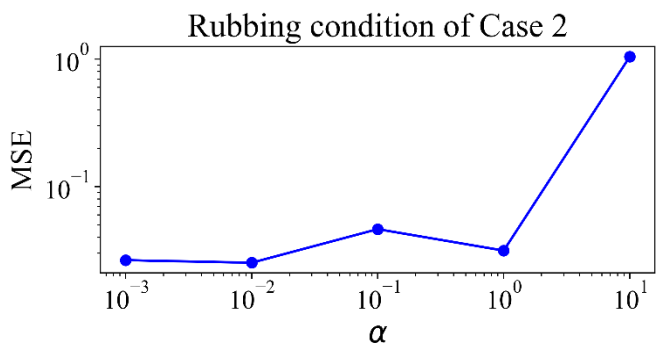
Grid search [19] is performed to tune α for each dataset. The grid is [0.001, 0.01, 0.1, 1.0, 10.0] and the value that achieves the smallest MSE is chosen. When computing the final frequency that combines the deterministic frequency and the stochastic frequency feature, α controls the relative effect of the stochastic frequency feature. If α is small, the deterministic frequency will become dominant; otherwise, the stochastic frequency will be important. The results for Te_3 are described in Figure 4-31. The x-axis and y-axis are described in the log scale. In Case 1, an α of 0.1 shows the smallest MSE value. In Case 2 and Case 3, an α of 0.01 achieves the smallest MSE value. Also, a small α results in better performance in general; α of 10.0 shows the largest MSE in all cases. This means that the deterministic frequency is more important than the stochastic frequency. This is because the validation datasets follow the assumption that the signals are stationary; thus, it is unnecessary to impose great weight on the stochastic frequency.

To interpret the proposed network, the attention score in *ME* is visualized with the frequency $(f_i + \alpha \times \Delta f_i)$ and the magnitude (a_i). Figure 4-32 shows the results of Te_3 of Case 1 and Case 2, and Figure 4-33 present those of Case 3. The frequency components $(f_i + \alpha \times \Delta f_i, a_i)$ are compared with the magnitude spectrum of the true signals, which are offered in Figures 4-5(b), 4-7(b), and 4-8(b). In Case 1, the magnitude spectrum is similar to the spectrum obtained by FFT. The attention score is high near the characteristic frequencies – 5 [Hz], 330 [Hz], and 500 [Hz]. This denotes that the proposed model is able to focus on the characteristic frequencies well. This result is also found in Case 2. The learned frequency features are similar to the magnitude spectrum of the true signals. When seeing the attention

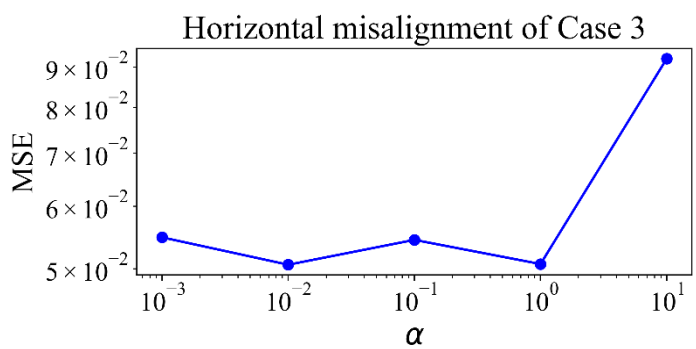
score, the sub-harmonic components – 60 [Hz], 120 [Hz], and 180 [Hz] – are dominant, and there are few components other than those sub-harmonic components. The reason why the unimportant components have a very small magnitude is because of the strong regularization applied to *ME*. Because the attention score is very high at the sub-harmonic frequencies, it can be argued that the proposed method also concentrates on the sub-harmonic components well in Case 2. For Case 3, the magnitude spectrum of the proposed method is similar to the true spectrum, which is shown in Figure 4-8(b). As can be seen from the attention score, the network focuses well on the sub-harmonics, including 30 [Hz], 60 [Hz], 90 [Hz], and 120 [Hz]. But, the sub-harmonic at 210 [Hz] is less concentrated, and other frequency components except the sub-harmonics are focused. This is because the proposed model is distracted by the noise components.



(a)



(b)



(c)

Figure 4-31 Grid search results for Te_3: (a) Case 1, (b) Case 2 (Rubbing), and (c) Case 3 (Horizontal misalignment)

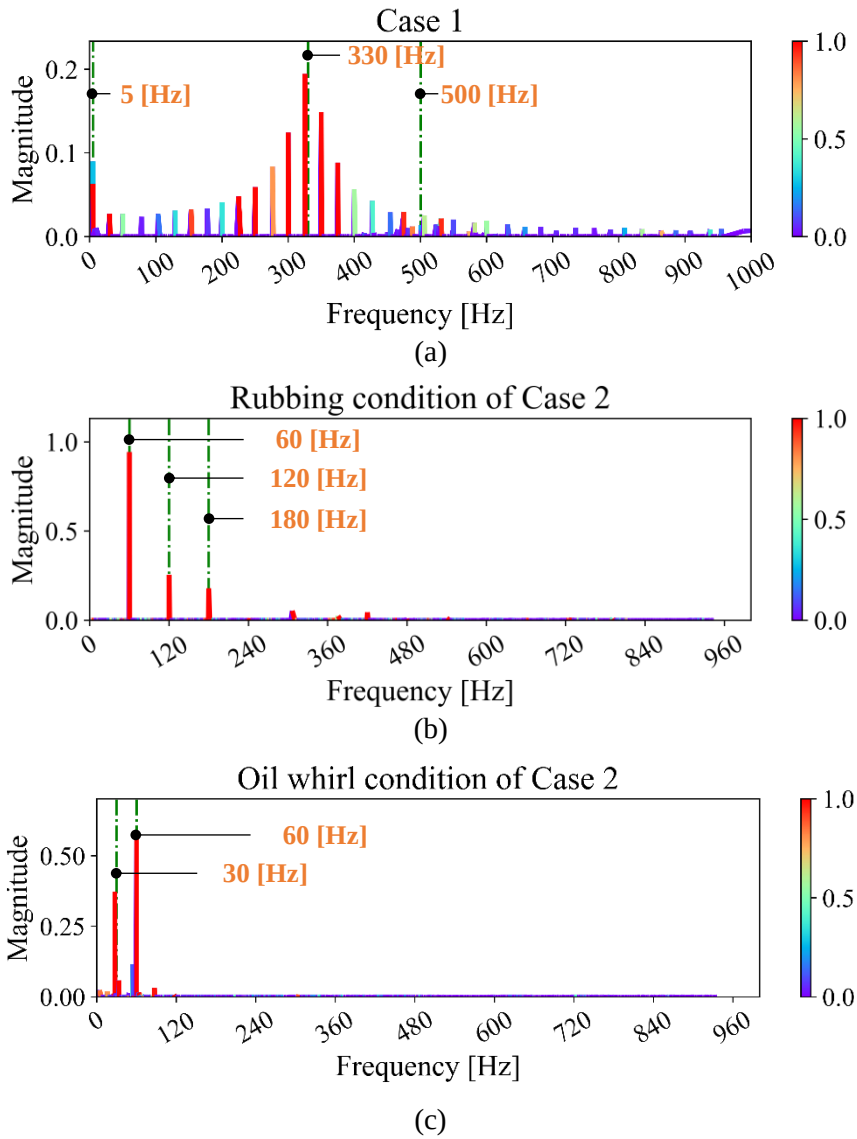


Figure 4-32 Visualization of the attention score for Te_3: (a) Case 1, (b) rubbing condition in Case 2, and (c) oil whirl condition in Case 2

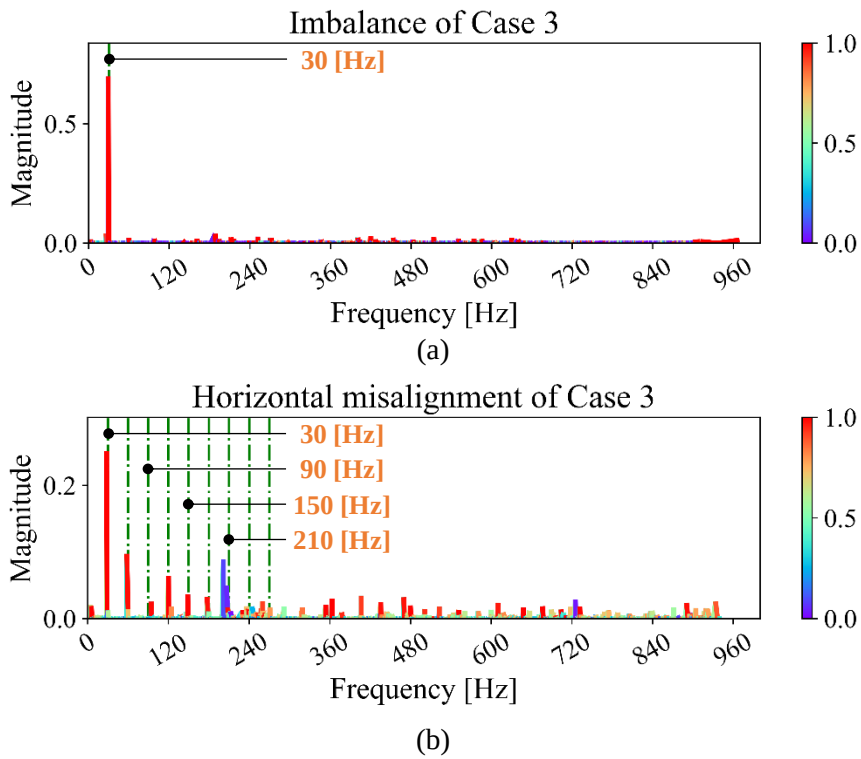


Figure 4-33 Visualization of the attention score for Te_3 in Case 3: (a) imbalance condition and (b) horizontal misalignment condition

4.6 Summary and Discussion

In this research, a new generative network called FLGN is newly proposed to generate signals of various lengths. Unlike VAE or GAN-based models, the proposed method is a new generative model that is designed and trained based on signal processing knowledge; therefore, the proposed approach has the capability to learn the frequency information of the training data. The proposed method consists of three extractors – the frequency extractor, the phase extractor, and the magnitude extractor. Those extractors can extract the frequency, phase, and corresponding magnitude in the training signal. Three datasets – a simulated signal, the RK4 dataset, and MAFAULDA – are utilized to validate the proposed model. The proposed method is evaluated both qualitatively and quantitatively. The validation results denote that the proposed approach can generate signals that are sufficiently similar to the true signals. Specifically, the fundamental frequency and its sub-harmonics are very similar to each other. The hyper-parameter study of α indicates that a small α achieves better performance for a stationary signal. Also, when interpreting the network by visualizing the attention score, it can be found that the proposed method can focus on the characteristic frequency components.

Sections of this chapter have been published or submitted as the following journal article:

1) **Jin Uk Ko**, Jinwook Lee, Taehun Kim, Yong Chae Kim, and Byeng D. Youn, "Frequency-learning generative network (FLGN) to generate vibration signals of variable lengths," *Expert Systems with Applications*, 2022

Chapter 5

Multi-task Learning of Classification and Denoising (MLCD) for Health Classification

This section proposes a multi-task learning of classification of denoising (MLCD) scheme to make a classifier robust against noisy data. The proposed method is a learning scheme that simultaneously learns classification and denoising, with hyper-parameters optimized by the Bayesian method [21]. Among various hyper-parameter optimization methods, including grid search [19] and random search [20], we chose the Bayesian method because it outperforms conventional methods [21]. Classification is chosen as the primary task because this study focuses on the diagnosis of a rotor system; that is, classifying the condition of the system. MLCD proposes simultaneous learning of these tasks rather than learning classification after denoising. By enabling an explicit denoising capability while classifying the health condition, MLCD improves the diagnostic performance by adding a regularization effect from learning the auxiliary task (denoising) and decreases the computational time required, as compared with the computational time that would be required to learn classification sequentially, after denoising. To validate the effect of MLCD on noisy signals, MLCD is integrated with two popular deep-learning algorithms;

LSTM and 1D CNN. The two MLCD-based algorithms, MLCD-LSTM and MLCD-1D CNN, are compared with LSTM and 1D CNN, respectively, by using rotor testbed data; these are ablation tests to validate the effect of MLCD. The performance of each algorithm is maximized by choosing critical hyper-parameters through Bayesian optimization. The results of the case study support that MLCD-LSTM and MLCD-1D CNN show improved test accuracy for various noisy inputs, respectively. By visualizing the intermediate features and the t-distributed Stochastic Neighboring Embedding (t-SNE) [25] results of the high-level features, it was found that MLCD-based algorithms extract noise-robust and various features, which also contain the sinusoidal characteristic of the input signals.

5.1 Background: Multi-task Learning

Multi-task learning (MTL) is a learning strategy that forces an algorithm to solve more than two tasks simultaneously [82]. Among the tasks, the main task is called the primary task. The other tasks used to help the primary task are called auxiliary tasks. By learning the auxiliary tasks simultaneously, the performance of the primary task can be improved because the auxiliary tasks prevent the algorithm from being overfitted to the primary task [83]. The neural network structure of MTL is shown in Figure 5-1, where there are three types of layers: the input layer, the shared layers, and the task-relevant layers. A shared representation for all tasks is learned in the shared layers, while a representation specific to each task is learned in the task-relevant layers. Note that T_1 indicates a primary task and $\{T_i\}_{i=2}^{m}$ denote auxiliary tasks. Examples of tasks include classification, regression, and denoising [84].

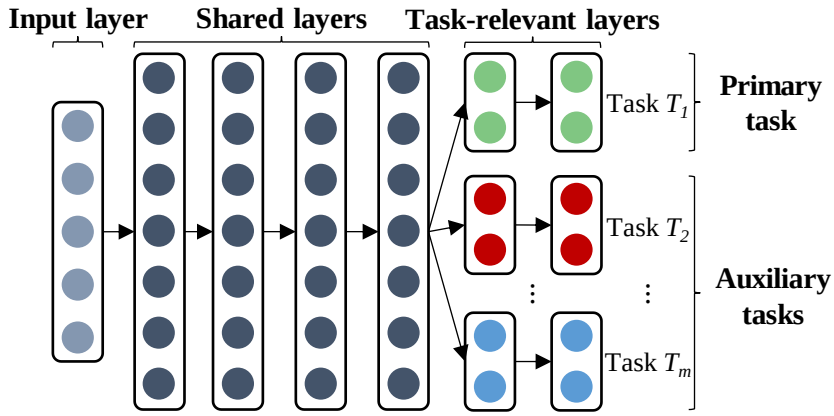


Figure 5-1 Architecture of a neural network with multi-task learning

To develop a noise-robust fault diagnosis algorithm, the proposed scheme defines the primary task as classification and the auxiliary task as denoising of the input signals. Given input signals $\{\tilde{x}_n\}_{n=1}^N$ and target label vectors $\{y_n\}_{n=1}^N$, classification seeks to find the function $y_n = f(\tilde{x}_n)$. The denoising task predicts clean samples $\{x_n\}_{n=1}^N$, given noisy samples $\{\tilde{x}_n\}_{n=1}^N$ where $\tilde{x}_n = x_n + \varepsilon$; ε is noise. That is, the step of denoising seeks to find the relationship $x_{\mathrm{n}} = g(\tilde{x}_n)$.

5.2 Multi-task Learning of Classification and Denoising (MLCD)

This section delineates the proposed multi-task learning of classification and denoising (MLCD) scheme to make a classifier robust against noisy signals. The overall procedure of the MLCD scheme and its integration with LSTM and 1D CNN are presented in this section.

5.2.1 Overall Procedure of MLCD

To solve the noise issue, multi-task learning of classification and denoising (MLCD) is newly proposed. In the proposed method, an algorithm learns the classification and denoising simultaneously. In the final layers of classification and denoising, softmax and linear activations are selected, respectively. Then, the outputs of the classification $\left(\hat{y}^{clf}\right)$ and denoising $\left(\hat{y}^{den}\right)$ are expressed as follows:

$$\hat{y}^{clf} = \frac{1}{\sum_{k=1}^{K} e^{z_k^{clf}}} \begin{bmatrix} e^{z_1^{clf}} \\ \vdots \\ e^{z_K^{clf}} \end{bmatrix} \tag{5.1}$$

$$\hat{y}^{den} = \begin{bmatrix} z_1^{den} \\ \vdots \\ z_D^{den} \end{bmatrix} \tag{5.2}$$

where K denotes the number of classes; D denotes the dimension of the input signals; $z_i = w_i^T h + b_i$ is the linear summation of the previous layer (h) with weight vector (w_i) and bias (b_i) corresponding to the i^{th} node of the final layer; $\{z_k^{clf}\}_{k=1}^{K}$ and $\{z_j^{den}\}_{j=1}^{D}$ denote the final linear projections in the classification and denoising, respectively. The designed objective function (L_{MLCD}) is defined as follows:

$$\begin{aligned} L_{MLCD} &= L_{clf}\left(\hat{y}^{clf}; W_{shd}, W_{clf}\right) + \beta \times L_{den}\left(\hat{y}^{den}; W_{shd}, W_{den}\right) \\ &= -\frac{1}{B}\sum_{n=1}^{B}\sum_{k=1}^{K} y_{nk}^{clf} \log \hat{y}_{nk}^{clf} + \beta \times \frac{1}{B}\sum_{n=1}^{B}\left\|\hat{y}_n^{den} - x_n\right\|_1 \end{aligned} \tag{5.3}$$

where L_{clf} and L_{den} are loss functions of classification and denoising; cross entropy loss and mean absolute error, respectively; W_{shd}, W_{clf}, and W_{den} denote the trainable

parameters in the shared, classification-relevant, and denoising-relevant layers, respectively; β is a weighting hyper-parameter; $y_n^{clf} = \left[y_{n1}^{clf}, \ldots, y_{nk}^{clf}, \ldots, y_{nK}^{clf}\right]^T$ is a true one-hot vector corresponding to an input $\tilde{x}_n$; B is batch size; $\|\cdot\|_1$ denotes the L1 norm. Then, the parameter updating rules become as follows:

$$W_{shd}^{k+1} = W_{shd}^{k} - \eta\left(\frac{\partial L_{clf}}{\partial W_{shd}^{k}} + \beta\frac{\partial L_{den}}{\partial W_{shd}^{k}}\right) \tag{5.4}$$

$$W_{clf}^{k+1} = W_{clf}^{k} - \eta\frac{\partial L_{clf}}{\partial W_{clf}^{k}} \tag{5.5}$$

$$W_{den}^{k+1} = W_{den}^{k} - \eta\frac{\partial L_{den}}{\partial W_{den}^{k}} \tag{5.6}$$

where k is the k^{th} iteration during training; η is the learning rate. η and β are critical hyper-parameters because η regulates the extent of training, and β controls the relative importance between the tasks. For most studies, hyper-parameters are chosen heuristically for simplicity. However, a manual hyper-parameter setting cannot ensure the maximal performance of an algorithm; for example, too large η can cause the training not to converge, and too large β can ignore the learning of classification. Therefore, in this study, the critical hyper-parameters are chosen by Bayesian optimization [21]. Bayesian optimization finds the solution by using a surrogate model and Bayesian updating. After choosing optimal hyper-parameters, the total parameters (W_{shd},W_{clf},W_{den}) are trained as expressed in Eqs. (5.4), (5.5) and (5.6).

MLCD improves generalization performance for two reasons. First, learning the denoising task gives hints for classification so that the algorithm learns more

meaningful features in the shared layers. This enables an MLCD-based algorithm to extract more diverse and meaningful features rather than similar and simple ones. Second, L_{den} plays a role as the regularization term for classification, which prevents the algorithm from being overfitted to classification. Thus, the final features at the classification-relevant layer will be distinguished better according to the classes. For these reasons, MLCD-based algorithms can achieve improved generalization performance.

The entire procedure of the developed fault diagnosis approach is described in Figure 5-2. There are three main parts to the method: data acquisition, data preprocessing, and fault classification with denoising. First, raw vibration signals are measured from a rotor system with perpendicularly located proximity sensors. These raw signals are not suitable to use directly because the number of sample points per cycle is not synchronized. In addition, the anisotropic characteristics of faults might not be involved well in the raw signals depending on the directions of the sensors. Thus, in the data preprocessing step, the raw signals are processed to be used as input to the deep-learning-based algorithms. Finally, the preprocessed data are used to train the deep-learning algorithm, such as LSTM or 1D CNN.

Among many candidates, LSTM and 1D CNN are employed in this research; these are the two most widely used algorithms in fault diagnosis studies. LSTM learns the sequential context in the input through several gates and cell states; 1D CNN learns meaningful representation by sliding filters – whose heights are equal to those of the input – in the time direction. The critical hyper-parameters are chosen by Bayesian optimization. The Bayesian method, which finds the optimal solution by surrogate function and Bayesian updating, can provide superior results, as

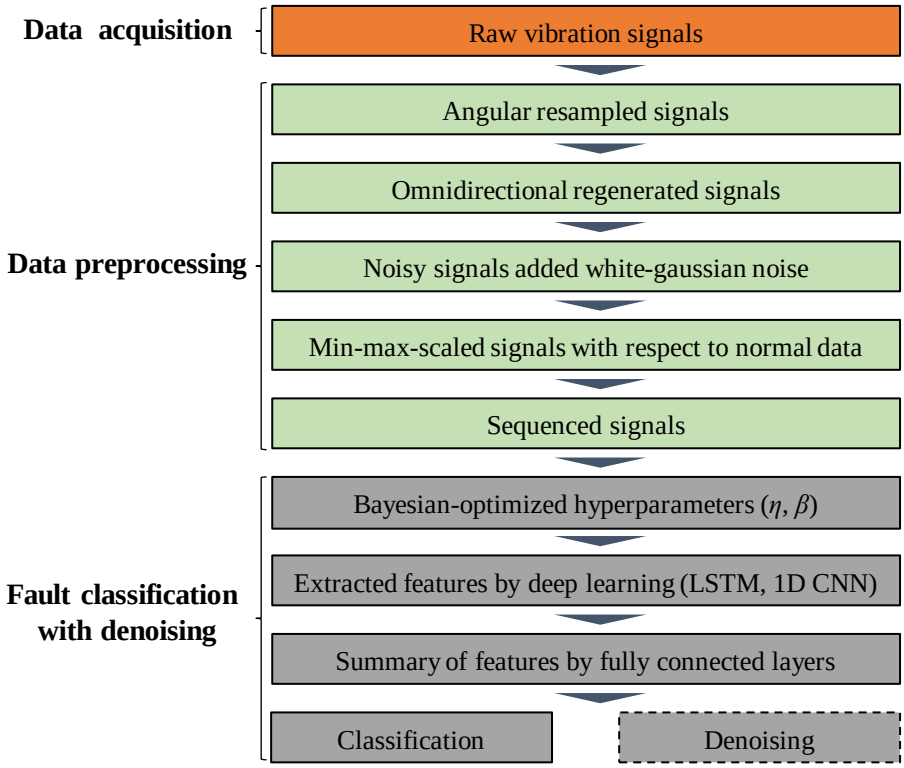


Figure 5-2 Overall procedure of the newly proposed method

compared to traditional methods like random search and grid search. After finding optimal hyper-parameters, the algorithm is trained to learn both tasks. Features are extracted in the shared layers automatically, and they are summarized through the fully connected layers of task-relevant layers for both tasks. Note that classification and denoising are learned simultaneously in training; however, only classification is turned on during testing.

5.2.2 Integration with LSTM: MLCD-LSTM

Based on the related works outlined in Section 5.1, LCD-LSTM is developed in this research by integrating MLCD with LSTM. The architecture of MLCD-LSTM is shown in Figure 5-3. The ODR signals are generated along the half circumference in 15˚ intervals; this adds up to 12 signals. The sequence length of the input is 64, which is the same as the number of sample points of two revolutions. Thus, the input dimension becomes 12-by-64. Then, two LSTM layers of 24 hidden nodes are stacked. All hidden states of the second LSTM layer are used for denoising, while only the final hidden state of LSTM is used for the classification. The outputs of LSTM are connected to the task-relevant fully connected layers ($FC1_C$, $FC1_D$); the number of hidden nodes at each FC1 is 256. These two FC1 layers are connected to the final layers ($FC2_C$, $FC2_D$), which give the output corresponding to the classification and denoising, respectively. The fully connected layers of the denoising (dotted black line) are inactivated in the test procedure. Note that the LSTM algorithm, which is compared with MLCD-LSTM, has the same architecture except for the denoising part. The batch size and training epoch are both set as 100. The hyperbolic tangent function and the rectified linear unit (ReLU) are selected as the activation functions of the LSTM layers and the fully connected layers, respectively.

5.2.3 Integration with 1D CNN: MLCD-1D CNN

MLCD-1D CNN is developed by applying MLCD to 1D CNN. The architecture of MLCD-1D CNN is described in Figure 5-4. The shapes of the input and output are the same as those of MLCD-LSTM. A total of four 1D convolutional layers and two max-pooling layers are used for the shared part. The number of filters in each

convolutional layer is 8, 16, 32, and 32, respectively. The stride of filters in each convolutional layer is 1. The size of the max-pooling is set to 2. The output of the final pooling layer is connected to two intermediate fully connected layers ($FC1_C$, $FC1_D$) of 128 hidden nodes. These two FC1 layers are connected to the final layers ($FC2_C$, $FC2_D$), which produce the final output of the classification and denoising tasks, respectively. Similar to MLCD-LSTM, the fully connected layers for the denoising task are not activated during the testing procedure. The architecture of the 1D CNN algorithm, which is compared with MLCD-1D CNN, is the same as that of MLCD-1D CNN, except for the denoising part. The batch size and training epoch are both chosen as 100. The leaky-rectified linear unit (LeakyReLU) and ReLU are chosen as the activation functions of the convolutional layers and the fully connected layers, respectively.

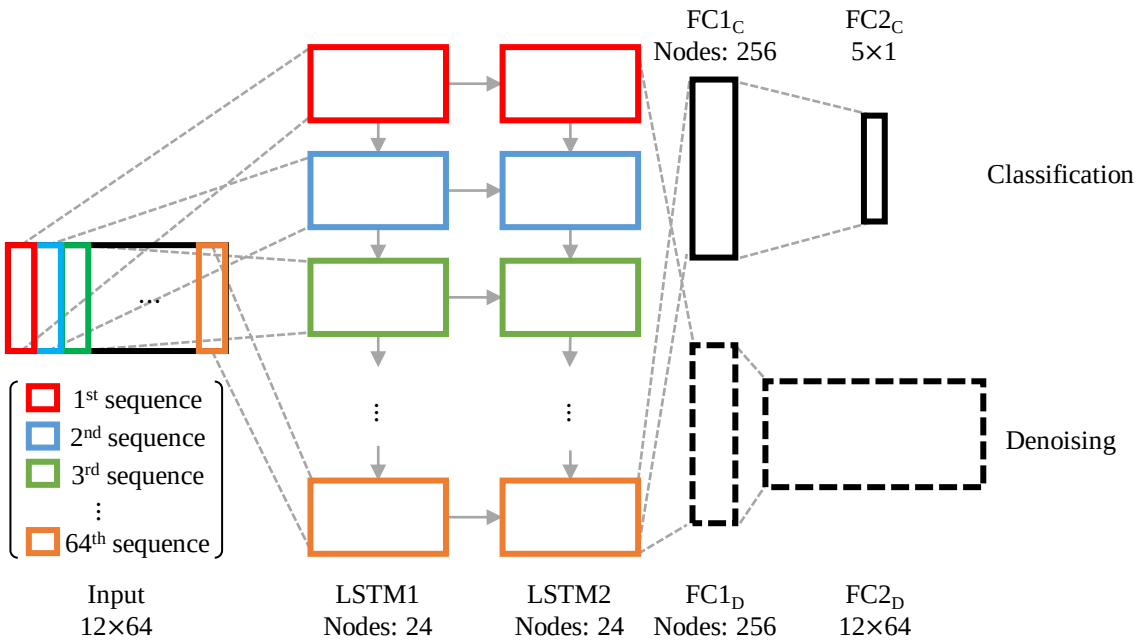


Figure 5-3 Architecture of MLCD-LSTM

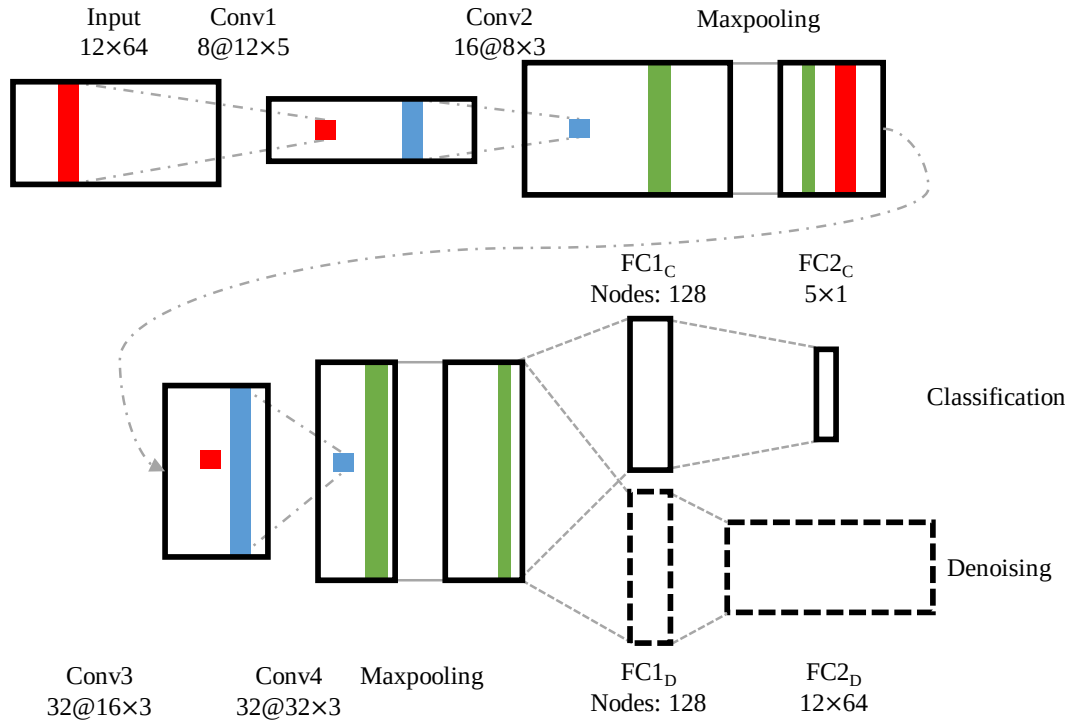
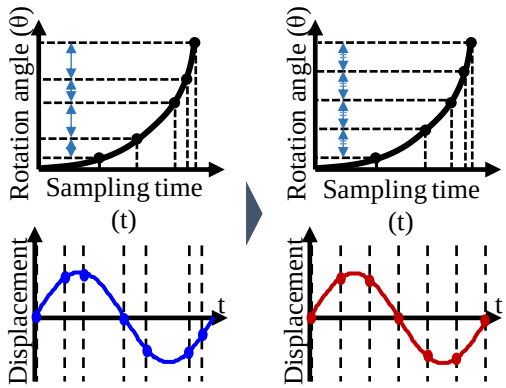


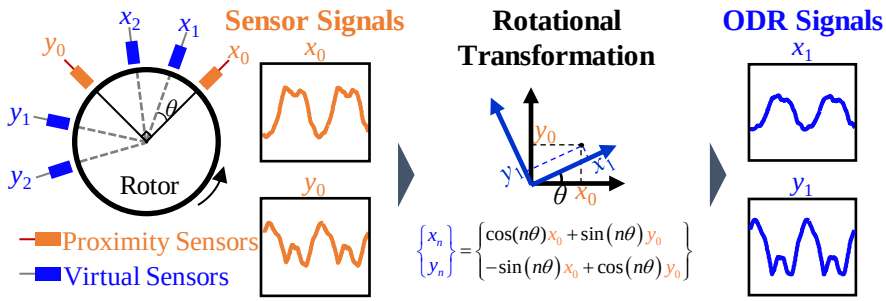
Figure 5-4 Architecture of MLCD-1D CNN

5.3 Preprocessing Techniques

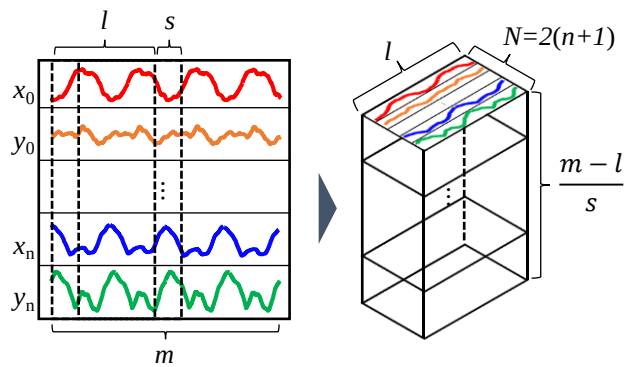
The details of the preprocessing step are illustrated in Figure 5-5. First, the raw signals are angular resampled to synchronize the number of sample points in a cycle by rearranging the rotation angle of a rotor equally [85, 86], as shown in Figure 5-5 (a). The rotation angle is obtained from the tacho signal. Then, to capture the directional characteristics of the fault, omnidirectional regeneration (ODR) signals [79] are generated from the resampled signals by rotational transformation, as shown in Figure 5-5 (b). The ODR signals can be considered as signals that are measured at several circumferential positions; thus, they contain more information about the system than the raw signals. Next, white gaussian noise is added to the ODR signals. The noisy ODR signals and the clean ODR signals are considered noisy and clean samples, respectively. The noisy signals of all labels are scaled with respect to the normal data to preserve the relative magnitude information. The noisy signals become the input of an MLCD-applied classifier, and the clean signals are the target output of the classifier. Finally, to make the signals be entered into LSTM and 1D CNN, the m noisy ODR and clean ODR signals of each class are sampled with a given sequence length (l) and stride (s); then, the number of final samples becomes $(m - l)/s$. The sequenced signals of all classes are concatenated and shuffled.



(a)



(b)



(c)

Figure 5-5 Graphical explanations of preprocessing: (a) angular resampled signals, (b) omnidirectional regeneration signals, and (c) sequenced signals

5.4 Description of the Validation Datasets

The RK4 dataset was obtained to validate the performance of the proposed MLCD method. Figure 4-6(a) shows the experimental setup of a GE Bently-Nevada RK4 testbed, which has been used in many fault diagnosis studies of rotor systems [56, 79, 87-91]. The experimental settings, including the sampling rate and rotating speed, are the same as in Section 4.4. Five health states, including normal and four fault states – unbalance, misalignment, rubbing, and oil whirl – were acquired, since those faults are the most common types of faults of a rotor system [92]. Each state was measured three times. There are some differences among data sets, though the state (label) remains the same. The raw signals are angular resampled so that there are 32 samples in each cycle. The ODR signals were generated by rotating the resampled signals from 0˚ to 90˚ at 15˚ intervals. Four levels of white gaussian noise of signal-to-noise ratio (SNR) – 10, 1, 0, -1 [dB] – were added, where the SNR in decibels is defined as follows:

$$SNR_{dB} = 10\log_{10}\left(\frac{P_{signal}}{P_{noise}}\right) \tag{5.7}$$

P_{signal} and P_{noise} denote the power of the signal and noise, respectively. The noisy (blue line) and clean signals (red dotted line) of each dataset are illustrated in Figures 5-6, 5-7, and 5-8. As the SNR gets smaller, the clean signals are more distorted by the noise. After scaled about normal signals, they were sampled with a sequence length of 64 and stride of 8; then, the number of training samples of each state became 7048. More information about the testbed and data is provided in [77].

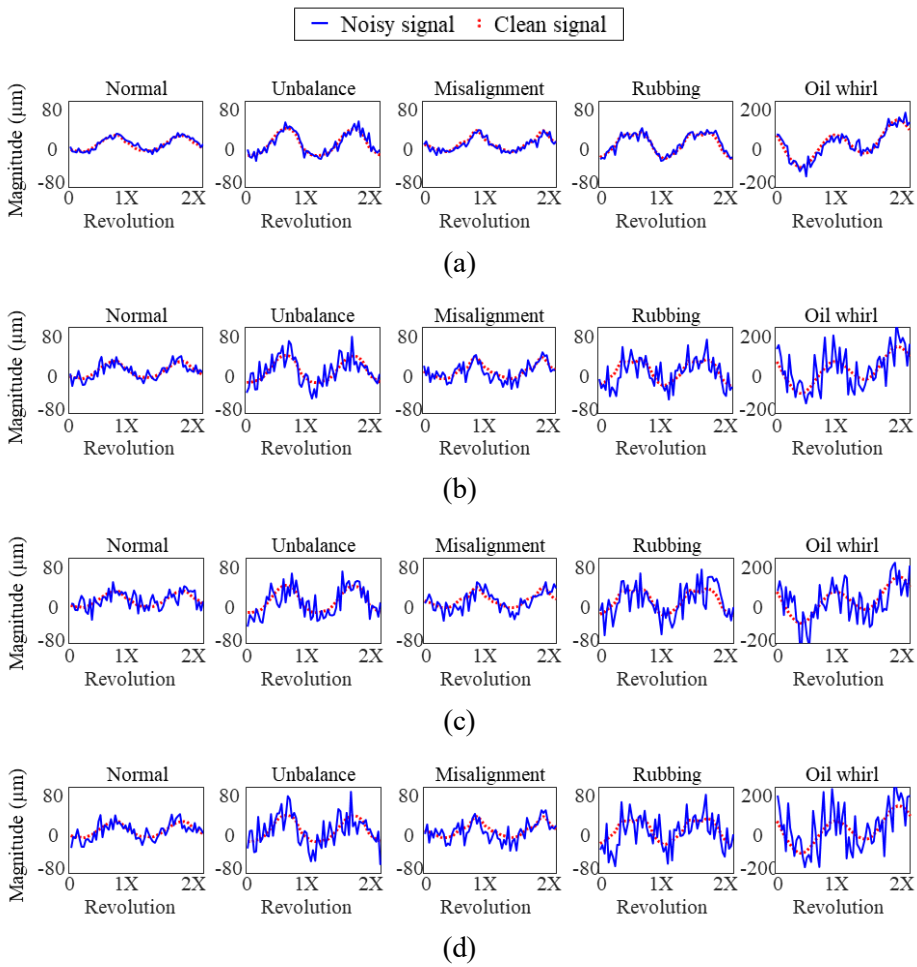


Figure 5-6 Signal trends of set 1: (a) SNR of 10 [dB], (b) SNR of 1 [dB], (c) SNR of 0 [dB], and (d) SNR of -1 [dB]

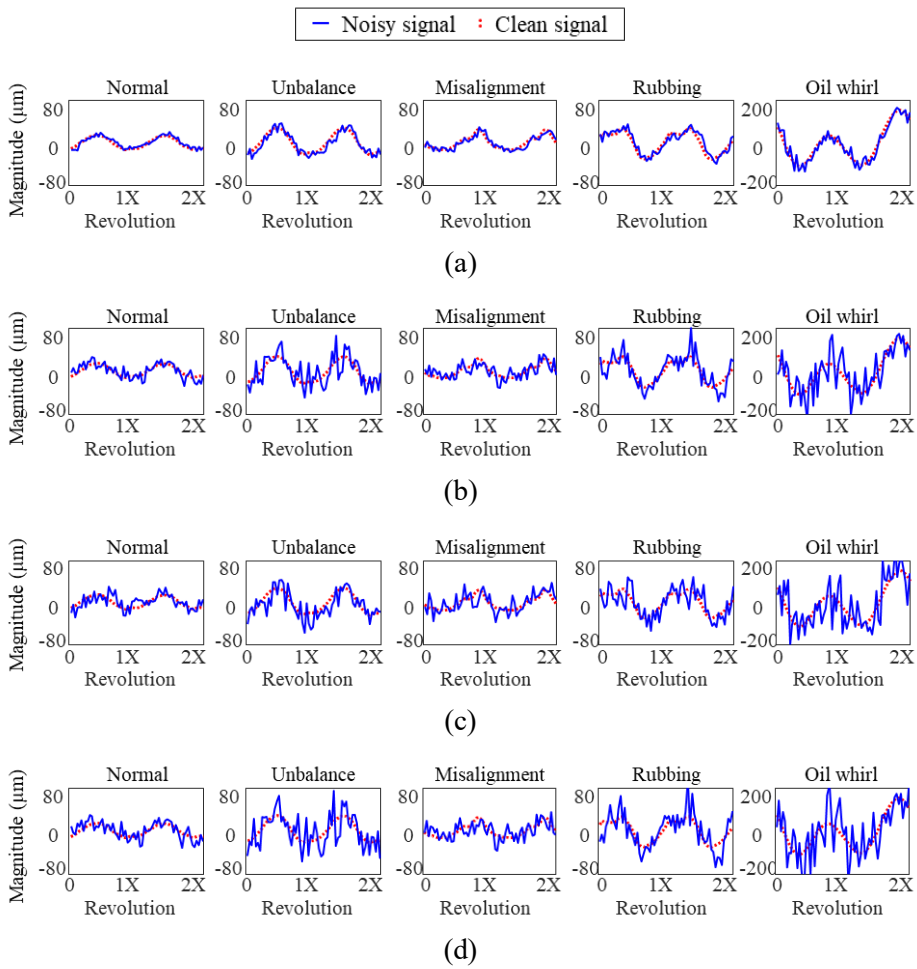


Figure 5-7 Signal trends of set 2: (a) SNR of 10 [dB], (b) SNR of 1 [dB], (c) SNR of 0 [dB], and (d) SNR of -1 [dB]

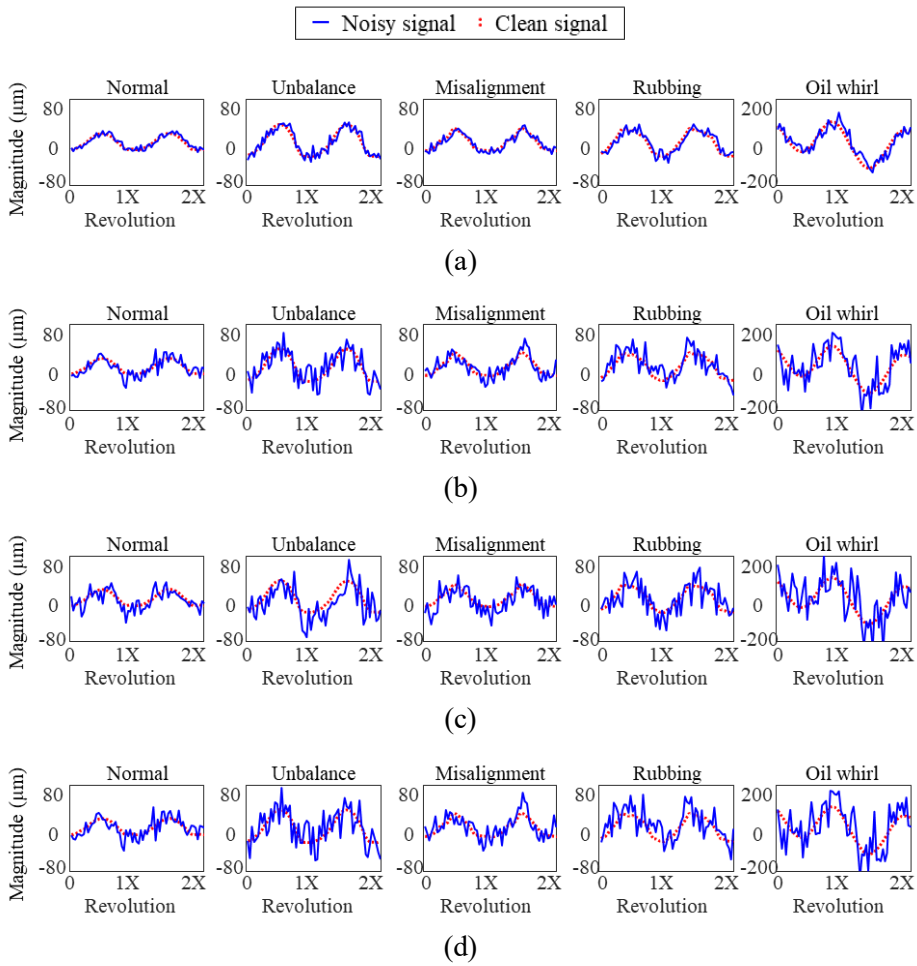


Figure 5-8 Signal trends of set 3: (a) SNR of 10 [dB], (b) SNR of 1 [dB], (c) SNR of 0 [dB], and (d) SNR of -1 [dB]

5.5 Validation of the Proposed Method

This section provides the results and analysis of the proposed MLCD method when validating the proposed method with the RK4 dataset. Two developed models – MLCD-LSTM and MLCD-1D CNN – are evaluated in terms of test accuracy and intermediate feature analysis.

5.5.1 Case Study 1: MLCD-LSTM

The optimized hyper-parameters of LSTM and MLCD-LSTM are summarized in Table 5-1. Figure 5-9 compares the average classification accuracy of 10 repeated tests of each case with the optimized hyper-parameters. *x* [dB] → *y* [dB] denotes that the algorithm is trained with an input where the noise of SNR of x [dB] is added and tested with the same data set added by SNR of y [dB] noise. The error range is one standard deviation from the mean accuracy. A large variance in accuracy means a large uncertainty in the results when there is a small disturbance in the input. When set 1 is used for training, MLCD-LSTM shows 10% to 25% better performance in all cases, as compared to LSTM. In addition, the overall variances of MLCD-LSTM are lower than those of LSTM. In particular, when an SNR of 1 [dB] is used, the proposed MLCD method decreases the variances significantly, as compared with the results from LSTM. If set 2 is used as training data, MLCD-LSTM also shows better test accuracy than LSTM. For the cases of SNRs of 10, 1, and 0 [dB], MLCD-LSTM shows similar test accuracy to LSTM. However, when an SNR of -1 [dB] is used, MLCD-LSTM shows a test accuracy of around 80%; whereas that of LSTM is around 40%, which is half of that of MLCD-LSTM. In this case, the variance of MLCD-LSTM is less than that of LSTM, which means the uncertainty in the

prediction is decreased through the use of the proposed MLCD method. Finally, when set 3 is used, the average test accuracies of MLCD-LSTM are greater than those of LSTM in all cases. In particular, when an SNR of 1 or -1 [dB] is used for training, the test accuracy of MLCD-LSTM is around 90%; whereas the performance of LSTM is around 80% and 60%, respectively. Comparing the variances of MLCD-LSTM and LSTM, MLCD-LSTM reduces variance significantly when an SNR of -1 [dB] is used for training. Although MLCD-LSTM slightly increases the variance when an SNR of 0 [dB] is trained, MLCD-LSTM shows greater test accuracy than LSTM. Overall, MLCD-LSTM improved generalization performance, as compared to LSTM. This is because learning the auxiliary task prevents the algorithm from being overfitted toward classification by giving a regularization effect, as discussed in Section 5.1.

Figure 5-10 provides a visualization of features at the $FC1_C$ of set 3 by t-SNE for three cases: SNR of 0 [dB] → -1 [dB] in (a) and (b), SNR of 1 [dB] → -1 [dB] in (c) and (d), and SNR of 10 [dB] → -1 [dB] in (e) and (f). Testing with an SNR of -1 [dB] is selected since it is the most difficult situation for a fault diagnosis algorithm. The better an algorithm trains, the better the features at the $FC1_C$ are classified. As you can see from Figures 5-10(a) and (b), while LSTM cannot distinguish normal, misalignment, and rubbing states, MLCD-LSTM diagnoses those states much better because the extracted features are distinctive according to the states. Figures 5-10(c) and (d) show that MLCD-LSTM also classifies normal, misalignment, and rubbing states much better than LSTM. In the case of an SNR of 10 [dB] → -1 [dB], as shown in Figures 5-10(e) and (f), given normal, misalignment, and rubbing states, the extracted features of LSTM are severely overlapped. However, MLCD-LSTM can

extract more distinguishable features from those states than LSTM. In particular, the normal state is diagnosed well from the misalignment and rubbing states by MLCD-LSTM. To summarize, since the features at the $FC1_C$ are better distinguished, as compared with LSTM, MLCD is shown to improve the fault diagnosis performance, given noisy input signals.

From the analysis of t-SNE, it is discovered that LSTM mostly confuses the rubbing state with others. To understand this fact a little more, the intermediate features at the shared layers – LSTM1 and LSTM2 in Figure 5-3 – are visualized in Figure 5-11 for the case of set 3 and an SNR of 0 [dB] → -1 [dB]. When a test sample in Figure 5-11(a) is given, the features of MLCD-LSTM are shown in Figure 5-11(b) and (c), and those of LSTM are shown in Figures 5-11(d) and (e). Three facts can be discovered from the results. First, it can be found that the noise is removed more and more as it passes through more layers in MLCD-LSTM. Second, compared to the features of LSTM, those of MLCD-LSTM are quite similar to sinusoidal waves. In particular, as you can see from Figures 5-11(c) and (e), most features of MLCD-LSTM are more similar to the true rubbing signal in Figure 5-8 than those of LSTM. Third, when comparing Figure 5-11(b) with (d) and Figure 5-11(c) with (e), respectively, while most features of LSTM overlap with each other, those of MLCD-LSTM show more various trends than those of LSTM. This indicates that MLCD enables the algorithm to extract more meaningful features, as compared to single-task learning of classification. Therefore, when significant noise exists in the input, MLCD-LSTM can understand the sinusoidal characteristic of the input signals better and extract a wider variety of features than LSTM.

Table 5-1 Bayesian optimization results of LSTM

	SNR [dB]	Algorithm	η (10^{-3})	β	Validation accuracy
Set 1	10	LSTM	0.0267	-	1.0000
		MLCD-LSTM	0.9482	0.4578	1.0000
	1	LSTM	3.5488	-	1.0000
		MLCD-LSTM	3.0938	1.3642	0.9979
	0	LSTM	0.3487	-	1.0000
		MLCD-LSTM	11.3084	48.9001	0.9997
	-1	LSTM	0.1000	-	1.0000
		MLCD-LSTM	12.2952	58.4489	1.0000
Set 2	10	LSTM	0.1000	-	1.0000
		MLCD-LSTM	6.6706	12.0396	1.0000
	1	LSTM	3.8200	-	1.0000
		MLCD-LSTM	2.3300	1.6878	1.0000
	0	LSTM	2.0691	-	1.0000
		MLCD-LSTM	8.8012	44.7087	0.9989
	-1	LSTM	32.0522	-	1.0000
		MLCD-LSTM	2.5600	1.4894	1.0000
Set 3	10	LSTM	0.1000	-	1.0000
		MLCD-LSTM	1.9932	0.1000	1.0000
	1	LSTM	0.1000	-	1.0000
		MLCD-LSTM	8.5376	10.6941	1.0000
	0	LSTM	0.3140	-	1.0000
		MLCD-LSTM	2.5695	0.1000	0.9994
	-1	LSTM	15.4718	-	1.0000
		MLCD-LSTM	1.6954	0.1000	0.9971

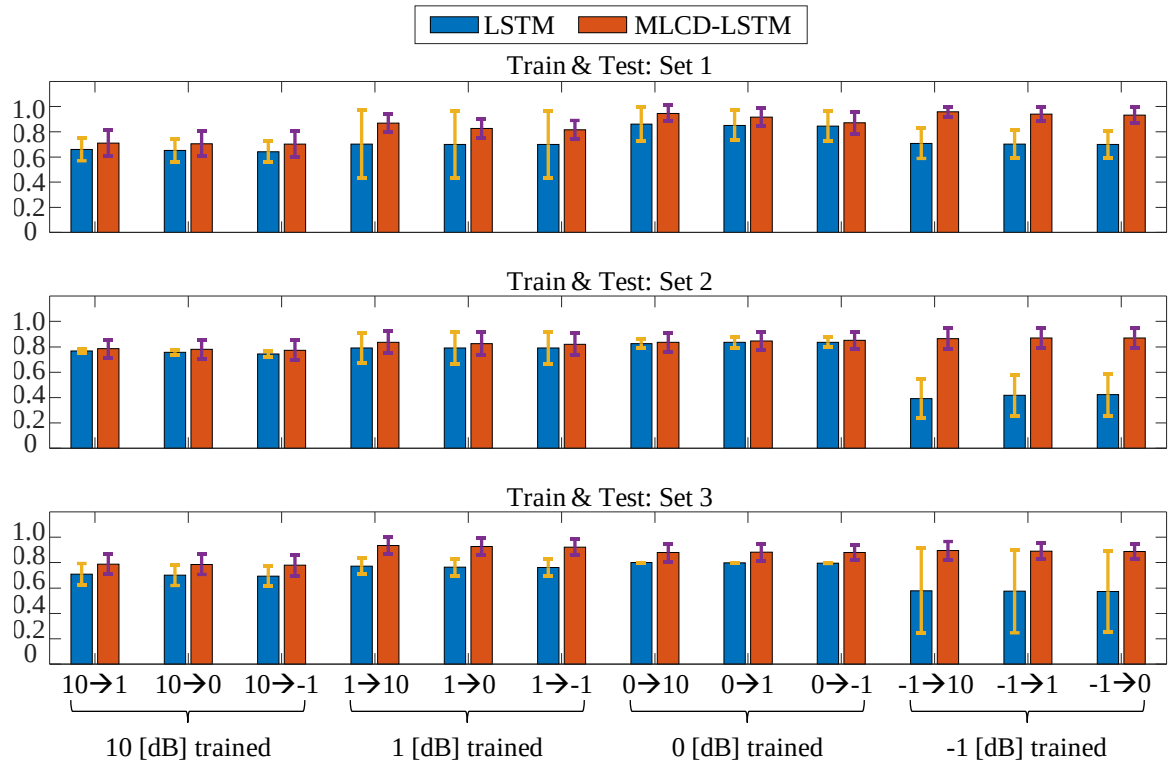


Figure 5-9 Average test results of LSTM and MLCD-LSTM

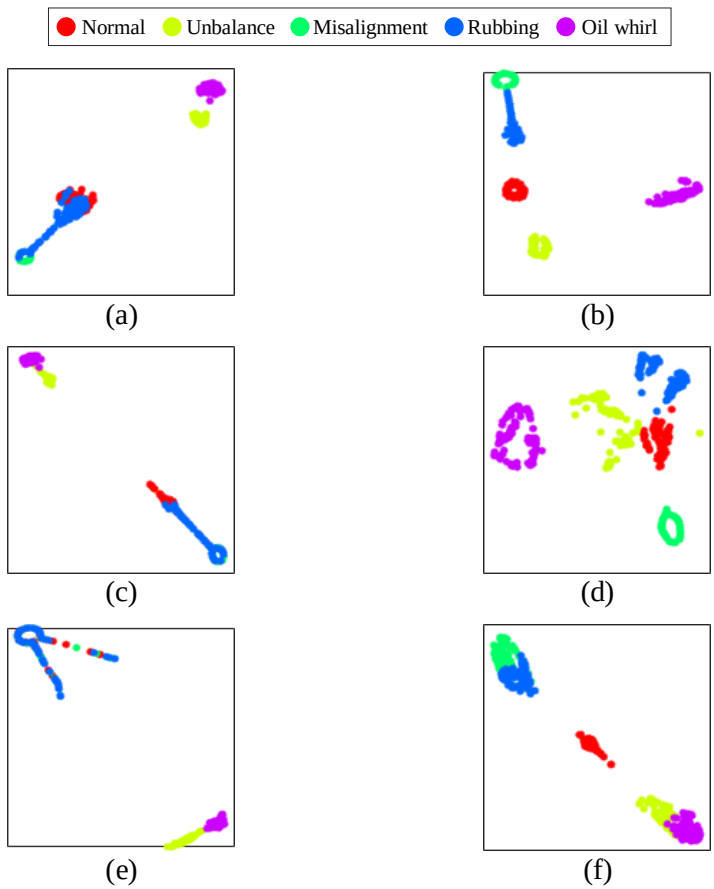


Figure 5-10 t-SNE visualization of features at $FC1_C$ with set 3: (a) LSTM, SNR of 0 [dB] → -1 [dB], (b) MLCD- LSTM, SNR of 0 [dB] → -1 [dB], (c) LSTM, SNR of 1 [dB] → -1 [dB], (d) MLCD- LSTM, SNR of 1 [dB] → -1 [dB], (e) LSTM, SNR of 10 [dB] → -1 [dB], and (f) MLCD- LSTM, SNR of 10 [dB] → -1 [dB]

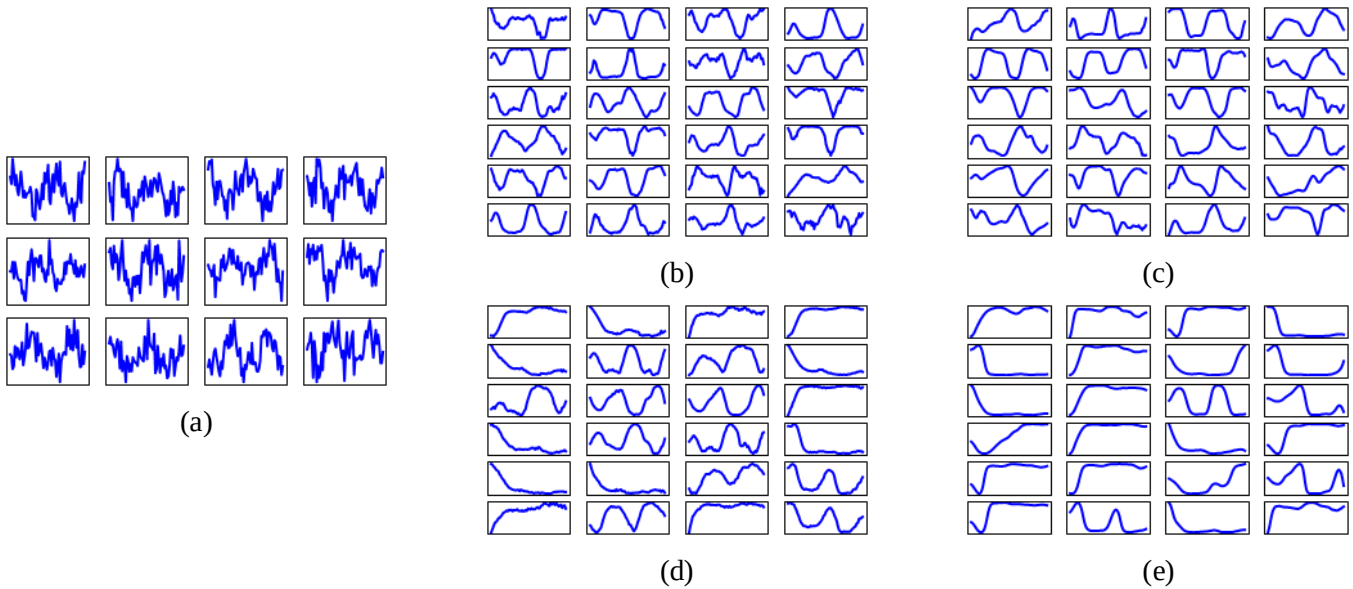


Figure 5-11 Visualization of intermediate features at the shared layers of LSTM and MLCD-LSTM with a rubbing test sample: (a) test sample, (b) after the first shared layer of MLCD-LSTM, (c) after the second shared layer of MLCD-LSTM, (d) after the first shared layer of LSTM, and (e) after the second shared layer of LSTM

5.5.2 Case Study 2: MLCD-1D CNN

The optimized hyper-parameters of 1D CNN and MLCD-1D CNN are tabulated in Table 5-2. Using the optimal hyper-parameters, each algorithm was trained and tested 10 times. The test results of MLCD-1D CNN and 1D CNN are shown in Figure 5-12. The results describe that the test accuracy of the proposed MLCD-1D CNN is better than that of 1D CNN in most cases. For set 1, MLCD-1D CNN shows better test accuracy than 1D CNN in all cases. In particular, when an SNR of 0 [dB] is trained, the mean test accuracy of MLCD-1D CNN is much greater than that of 1D CNN, while the variance of MLCD-1D CNN is far smaller than that of 1D CNN. This indicates that the MLCD method improves the generalization performance. In the case of set 2, there is little difference in the test accuracy between MLCD-1D CNN and 1D CNN when an SNR of 10 [dB] is used. However, when an SNR of 1 or 0 [dB] is used for training, the MLCD-1D CNN shows slightly better accuracy and less variance than 1D CNN. When set 3 is used, both MLCD-1D CNN and 1D CNN show almost 100 % test accuracy for the case of an SNR of 10 [dB]. For the case of an SNR of 1 [dB], MLCD-1D CNN shows slightly better performance than 1D CNN. However, when an SNR of 0 [dB] or -1 [dB] is used for training, the average accuracy of MLCD-1D CNN is almost 100%, while that of 1D CNN is under 80%. Moreover, the variance of test accuracy decreases considerably when the proposed MLCD method is used. In summary, MLCD-1D CNN shows an enhanced generalization performance because learning the denoising task gives a regularization effect to the classification task, as discussed in Section 5.1.

The features at the $FC1_C$ layer of set 1 are analyzed by using t-SNE in Figure 5-13 for three cases: SNR of 0 [dB] → -1 [dB] in (a) and (b), SNR of 1 [dB] → -1

[dB] in (c) and (d), and SNR of 10 [dB] → -1 [dB] in (e) and (f). Since the algorithms are more affected by the noise as the SNR level becomes smaller, the test case is chosen as an SNR of -1 [dB]. From Figures 5-13(a) and (b), it is found that MLCD-1D CNN classifies all states well, while 1D CNN confuses the misalignment and rubbing states. Moreover, MLCD-1D CNN clusters the features of each label better than 1D CNN: the features of some labels – normal, unbalance, and oil whirl – of 1D CNN are not clustered well. Figure 5-13(c) shows that it is difficult for 1D CNN to diagnose normal and rubbing conditions since the features of 1D CNN of normal and rubbing are close to each other. However, as shown in Figure 5-13(d), the features of normal and rubbing states of MLCD-1D CNN are clustered further apart than those of 1D CNN. Figures 5-13(e) and (f) show that while 1D CNN confuses normal, misalignment, and rubbing states, MLCD-1D CNN can extract more distinctive features from those states, which are located further from each other. In addition, MLCD-1D CNN also learns better-clustered features for the unbalance and oil whirl states. In short, it can be said that the generalization performance of MLCD-1D CNN is improved because the features at the $FC1_c$ are classified better than those of 1D CNN.

To understand the results better, the intermediate features at the first two convolutional layers – Conv1 and Conv2 in Figure 5-4– are visualized in Figure 5-14 for the case of set 1 and an SNR of 0 [dB] → -1 [dB]. Along with LSTM cases, the rubbing state is chosen since it is the hardest state for 1D CNN to diagnose accurately. There are three findings from the analysis. First, for both 1D CNN and MLCD-1D CNN, more noise in the input signal is removed as it passes through more convolutional layers; however, the extent of denoising is greater for MLCD-1D CNN.

This is because a higher level of representation for classification and denoising is learned as the input goes through more convolutional layers. Second, the features of MLCD-1D CNN are more similar to sinusoid waves, which means that MLCD-1D CNN can learn about the waveform of the input signal better than 1D CNN. Interestingly, the 8^{th} features (from upside to downside and from left to right) in Figure 5-14(b) and the 13^{th} and 16^{th} features in Figure 5-14(c) are quite similar to the true rubbing signal (red dotted line) in Figure 5-6(d). Lastly, when checking the similarity of features at the shared layers, the features of MLCD-1D CNN are more diverse than those of 1D CNN. In particular, almost half of the features at Conv2 of 1D CNN are similar to a w-shape, as shown in Figure 5-14(e). Consequently, given noisy input, MLCD-1D CNN learns the characteristic of the signal waveform better and generates noise-robust and more diverse features, as compared to 1D CNN.

Table 5-2 Bayesian optimization results of 1D CNN

	SNR [dB]	Algorithm	η (10^{-3})	β	Validation accuracy
Set 1	10	1D CNN	0.1000	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
	1	1D CNN	0.1000	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
	0	1D CNN	9.5624	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
	-1	1D CNN	0.1000	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
Set 2	10	1D CNN	0.1000	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
	1	1D CNN	0.1000	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
	0	1D CNN	0.1000	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
	-1	1D CNN	0.1842	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
Set 3	10	1D CNN	0.1000	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
	1	1D CNN	0.0244	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000
	0	1D CNN	4.0523	-	1.0000
		MLCD-1D CNN	0.413	15.2443	1.0000
	-1	1D CNN	1.6411	-	1.0000
		MLCD-1D CNN	0.1000	10.0000	1.0000

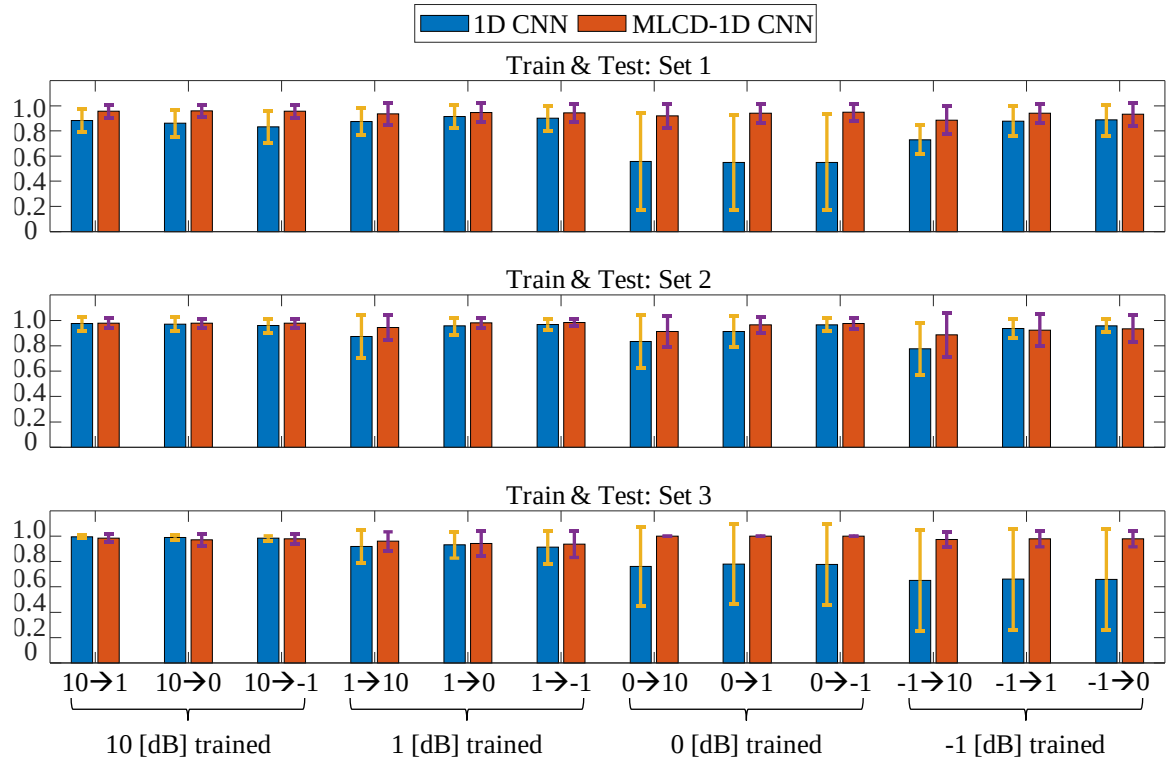


Figure 5-12 Average test results of 1D CNN and MLCD-1D CNN

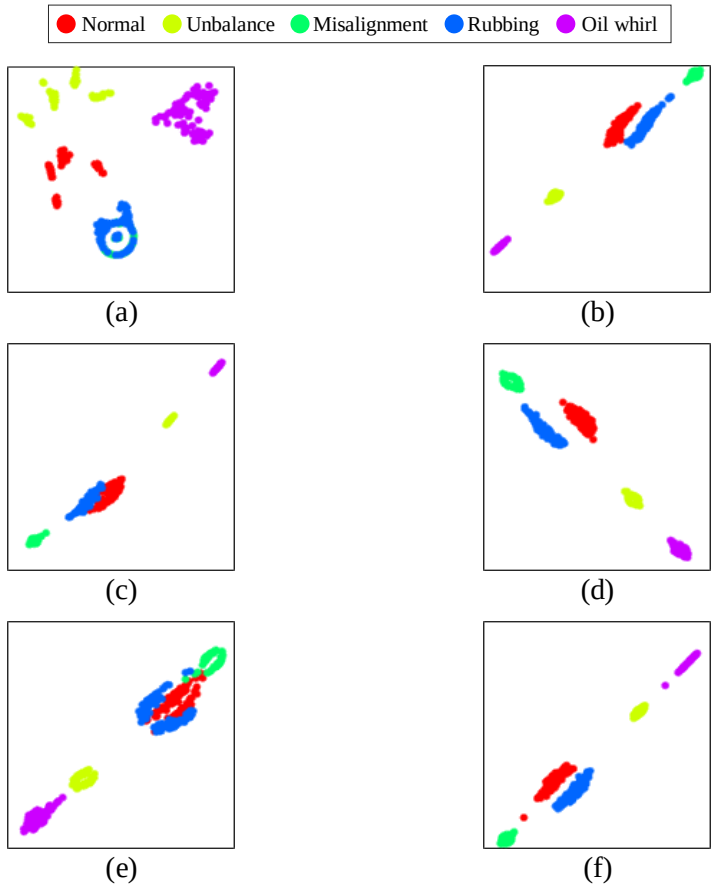


Figure 5-13 t-SNE visualization of features at $FC1_C$ with set 1: (a) 1D CNN, SNR of 0 [dB] → -1 [dB], (b) MLCD-1D CNN, SNR of 0 [dB] → -1 [dB], (c) 1D CNN, SNR of 1 [dB] → -1 [dB], (d) MLCD-1D CNN, SNR of 1 [dB] → -1 [dB], (e) 1D CNN, SNR of 10 [dB] → -1 [dB], and (f) MLCD-1D CNN, SNR of 10 [dB] → -1 [dB]

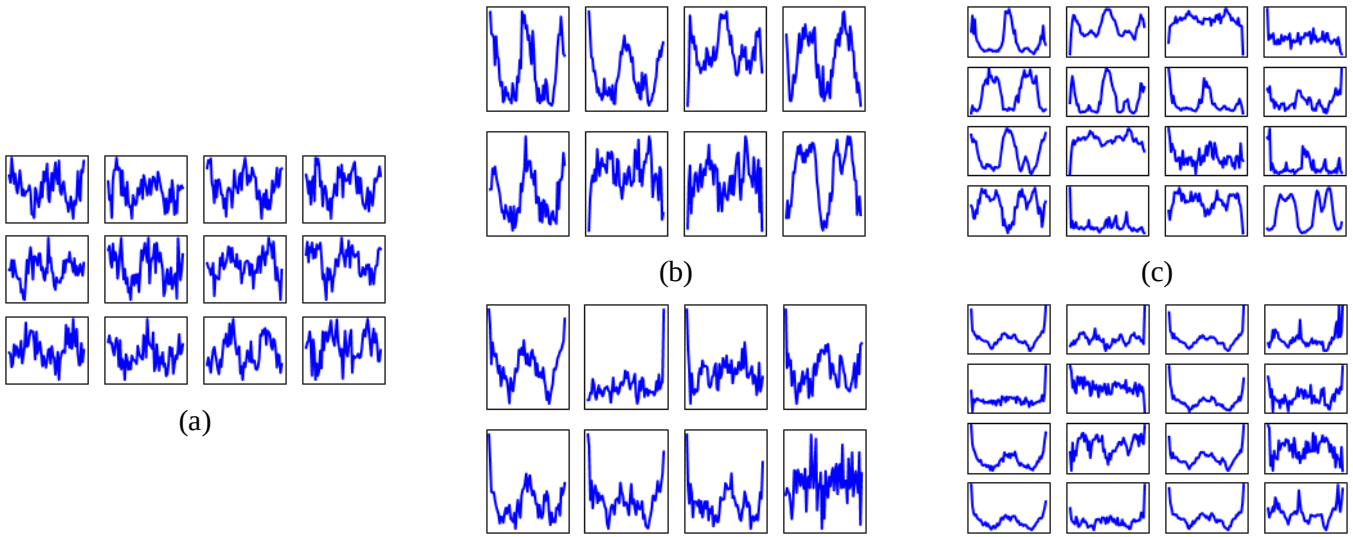


Figure 5-14 Visualization of intermediate features at the shared layers of 1D CNN and MLCD-1D CNN with a rubbing test sample: (a) test sample, (b) after the first shared layer, MLCD-1D CNN, (c) after the second shared layer, MLCD-1D CNN, (d) after the first shared layer, 1D CNN, and (e) after the second shared layer, 1D CNN

5.6 Summary and Discussion

In this research, we proposed a new training scheme called MLCD for noise-robust fault diagnosis. The key idea of MLCD is to improve the generalization performance of fault diagnosis through multi-task learning of classification and denoising using optimal hyper-parameters that are chosen by Bayesian optimization. MLCD was integrated with LSTM and 1D CNN; then, MLCD-LSTM and MLCD-1D CNN were newly developed. For each RK4 testbed data set, each algorithm was trained and tested with different SNR levels repeatedly. From the results and analysis, two conclusions can be made. First, the visualization of intermediate features shows that MLCD-based algorithms extract more meaningful features where the greatest amount of noise is removed and learn the representation of the signal waveform better. Second, when the high-level features at $FC1_C$ are visualized in two-dimensional space by t-SNE, the features of MLCD-based algorithms are classified better according to the five states. This means that the generalization performance of fault diagnosis is improved despite noisy input. In future work, tasks other than denoising will be researched to find the optimal combination with classification for noise-robust fault diagnosis.

Sections of this chapter have been published as the following journal article:

1) **Jin Uk Ko**, Joon Ha Jung, Myungyon Kim, Hyeon Bae Kong, Jinwook Lee, and Byeng D, Youn, "Multi-task learning of classification and denoising (MLCD) for noise-robust rotor system diagnosis," *Computers in Industry*, Vol. 125, pp. 103385, 2021.

Chapter 6

Conclusion

6.1 Contributions and Significance

This doctoral dissertation proposes a deep-learning-based methodology for macro- and micro-level fault diagnosis using operation and vibration signals. The proposed methodology consists of three novel studies: (1) an ensemble denoising auto-encoder-based dynamic threshold (EDAE-DT) to reduce false alarms by considering the fluctuation in the normal data; (2) a frequency-learning generative network (FLGN) to generate signals of variable lengths by learning the frequency information; and, (3) multi-task learning of classification and denoising (MLCD) approach to improve classification performance against noise by concurrently learning the denoising capability. The research in this dissertation provides the following contributions to the area of deep-learning-based fault diagnosis of rotating machinery.

Contribution 1: Development of a new anomaly detection technique that reduces false alarms by considering the fluctuations in the normal data.

This doctoral dissertation proposes an *ensemble denoising auto-encoder-based dynamic threshold (EDAE-DT)* to reduce false alarms in anomaly detection. Concretely, EDAE is a new modeling method that is able to learn the normal data well by using an ensemble technique with five DAEs. Together, the ensemble technique and denoising task enable the modeling performance to be improved. DT is developed to set a variable threshold by considering the joint distribution of the output of the EDAE and the residual. After calculating the joint distribution, it is discretized, and critical points are determined as the point where the upper tail of the marginal distribution becomes a confidence level; a critical function is obtained by linearly interpolating the critical points. This critical function computes a threshold value with respect to each output value. In summary, by 1) improving the modeling performance and 2) setting a threshold dynamically, the EDAE-DT achieves accurate anomaly detection, while generating the lowest number of false alarms of available methods.

Contribution 2: Suggestion of an innovative generative network to generate stationary signals of variable lengths by using the Fourier series.

This doctoral dissertation proposes a novel method called *frequency-learning generative network (FLGN)* to generate signals of variable lengths. FLGN is an innovative generative network, which is completely different from the prior VAE or GAN-based models. The FLGN approach consists of three feature extractors – a stochastic frequency extractor, a phase extractor, and a magnitude extractor – and a

sine-basis layer. A deterministic frequency is learned in the form of a trainable parameter; the stochastic frequency, phase, and magnitude are extracted in the form of features. The frequency and phase are used to construct a sine-basis, and that basis is entered into the magnitude extractor. The output of FLGN is obtained by adding a bias to the dot product of the magnitude vector and sine-basis vector. The proposed FLGN generates signals that are similar to the true signals, even if the lengths of the signals change. It is also found that the FLGN learns the characteristic frequency components in the training data well. In particular, through the use of an attention block at each extractor, it is discovered that the proposed FLGN approach focuses well on the characteristic frequencies.

Contribution 3: Suggestion of a new training scheme to make a classifier robust against noise by using multi-task learning.

This doctoral dissertation develops a new training scheme called *multi-task learning of classification and denoising (MLCD)* to make a classifier robust against noise. The proposed MLCD scheme learns the classification task, while learning the denoising task simultaneously. The multi-task learning technique enables improved generalization performance of a classifier. MLCD can be applied to any deep-learning algorithm regardless of its architecture. In this research, it is integrated with LSTM and 1D CNN. The MLCD-applied classifier has improved classification performance even if there is a large amount of noise in the input signal. MLCD also results in the classifier having less uncertainty in its output. Furthermore, not only does an MLCD-applied classifier have the ability to remove the noise in the input

signal, but the classifier also extracts meaningful and sinusoidal features. Overall, the MLCD-applied classifier extracts more discriminative features, as compared to a classifier without MLCD.

6.2 Suggestions for Future Research

This doctoral dissertation proposes an innovative methodology for macro- and micro-level fault diagnosis of rotating machinery using operation and vibration signals. Even if the proposed studies solve the limitations of the conventional approaches, there are still several research topics that need to be addressed further to enhance the performance of the resulting fault diagnosis. The following suggestions are specific recommendations for future research.

Suggestion 1: Validation of the proposed methods with signals under variable-speed conditions

The studies in this doctoral dissertation research were validated with signals that were obtained under constant-speed conditions. This means that the signals were stationary; their frequency information did not change with respect to time. However, some rotating machines, including motors and wind turbines, rotate under variable-speed conditions. Therefore, in future work, the proposed method should be validated with non-stationary signals under variable-speed conditions to broaden the applicability of the proposed studies.

Suggestion 2: Improvement of classification performance for extremely imbalanced data

In real industrial fields, fault samples are usually insufficient compared to normal samples; sometimes, fault samples might be extremely scarce. Although research thrust 2 augments fault samples given short signals, the generation performance will be decreased if the samples are extremely insufficient. Thus, an advanced fault diagnosis method should be developed for improved classification performance under extremely imbalanced data.

Suggestion 3: Development of a fault diagnosis scheme considering the domain discrepancy issue

Even when studying the same type of rotating machinery, measured signals can have various distributions according to the machines' various operating conditions. The performance of an algorithm is decreased if the test data has a different distribution than the training data; this is called the domain discrepancy issue. Domain adaptation is a research area that seeks to solve the domain discrepancy issue. Therefore, to make the proposed methodology work well on various mechanical systems, a novel fault diagnosis approach should be developed to mitigate the domain discrepancy issue through the use of domain adaptation techniques.

References

[1] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, D. Siegel, Prognostics and health management design for rotary machinery systems—Reviews, methodology and applications, Mechanical Systems and Signal Processing, 42 (2014) 314-334.

[2] B. Zhao, X. Zhang, H. Li, Z. Yang, Intelligent fault diagnosis of rolling bearings based on normalized CNN considering data imbalance and variable working conditions, Knowledge-Based Systems, 199 (2020) 105971.

[3] 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, Mechanical Systems and Signal Processing, 138 (2020) 106587.

[4] G. Genta, Dynamics of rotating systems, Springer Science & Business Media, 2005.

[5] P.W. Kalgren, C.S. Byington, M.J. Roemer, M.J. Watson, Defining PHM, a lexical evolution of maintenance and logistics, 2006 IEEE Autotestcon, IEEE, 2006, pp. 353-358.

[6] G. Pang, C. Shen, L. Cao, A.V.D. Hengel, Deep learning for anomaly

detection: A review, ACM Computing Surveys (CSUR), 54 (2021) 1-38.

[7] D.P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114, (2013).

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial networks, Communications of the ACM, 63 (2020) 139-144.

[9] S. Albawi, T.A. Mohammed, S. Al-Zawi, Understanding of a convolutional neural network, 2017 International Conference on Engineering and Technology (ICET), IEEE, 2017, pp. 1-6.

[10] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Computation, 9 (1997) 1735-1780.

[11] J. Snoek, H. Larochelle, R.P. Adams, Practical bayesian optimization of machine learning algorithms, Advances in Neural Information Processing Systems, 25 (2012).

[12] G.R. Terrell, D.W. Scott, Variable kernel density estimation, The Annals of Statistics, (1992) 1236-1265.

[13] D.-T. Hoang, H.-J. Kang, A survey on deep learning based bearing fault diagnosis, Neurocomputing, 335 (2019) 327-335.

[14] R.K. Pandit, D. Infield, SCADA-based wind turbine anomaly detection using Gaussian process models for wind turbine condition monitoring

purposes, IET Renewable Power Generation, 12 (2018) 1249-1255.

[15] Y. Xu, X. Tang, G. Feng, D. Wang, C. Ashworth, F. Gu, A. Ball, Orthogonal on-rotor sensing vibrations for condition monitoring of rotating machines, Journal of Dynamics, Monitoring and Diagnostics, 1 (2022) 29-36.

[16] C.E. Shannon, Communication in the presence of noise, Proceedings of the IRE, 37 (1949) 10-21.

[17] J. Chen, J. Li, W. Chen, Y. Wang, T. Jiang, Anomaly detection for wind turbines based on the reconstruction of condition parameters using stacked denoising autoencoders, Renewable Energy, 147 (2020) 1469-1480.

[18] A. Arranz, A. Cruz, M.A. Sanz-Bobi, P. Ruíz, J. Coutiño, DADICC: Intelligent system for anomaly detection in a combined cycle gas turbine plant, Expert Systems with Applications, 34 (2008) 2267-2277.

[19] G. Montavon, G. Orr, K.-R. Müller, Neural networks: tricks of the trade, Springer, 2012.

[20] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, The Journal of Machine Learning Research, 13 (2012) 281-305.

[21] J. Snoek, H. Larochelle, R.P. Adams, Practical bayesian optimization of machine learning algorithms, Advances in Neural Information Processing Systems, 2012, pp. 2951-2959.

[22] A. Dhini, B. Kusumoputro, I. Surjandari, Neural network based system for

detecting and diagnosing faults in steam turbine of thermal power plant, 2017 IEEE 8th International Conference on Awareness Science and Technology (iCAST), IEEE, 2017, pp. 149-154.

[23] X. Liu, S. Lu, Y. Ren, Z. Wu, Wind Turbine Anomaly Detection Based on SCADA Data Mining, Electronics, 9 (2020) 751.

[24] A. Likas, N. Vlassis, J.J. Verbeek, The global k-means clustering algorithm, Pattern Recognition, 36 (2003) 451-461.

[25] L. Van der Maaten, G. Hinton, Visualizing data using t-SNE, Journal of Machine Learning Research, 9 (2008).

[26] C. Lu, Z.-Y. Wang, W.-L. Qin, J. Ma, Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification, Signal Processing, 130 (2017) 377-388.

[27] G. Lee, M. Jung, M. Song, J. Choo, Unsupervised anomaly detection of the gas turbine operation via convolutional auto-encoder, 2020 IEEE International Conference on Prognostics and Health Management (ICPHM), IEEE, 2020, pp. 1-6.

[28] L.S. Nelson, The Shewhart control chart—tests for special causes, Journal of Quality Technology, 16 (1984) 237-239.

[29] W.A. Shewhart, Economic control of quality of manufactured product, Macmillan And Co Ltd, London, 1931.

[30] X. Zeng, M. Yang, Y. Bo, Gearbox oil temperature anomaly detection for wind turbine based on sparse Bayesian probability estimation, International Journal of Electrical Power & Energy Systems, 123 (2020) 106233.

[31] Y. Kim, K. Na, B.D. Youn, A health-adaptive time-scale representation (HTSR) embedded convolutional neural network for gearbox fault diagnostics, Mechanical Systems and Signal Processing, 167 (2022) 108575.

[32] X. Li, W. Zhang, Q. Ding, J.-Q. Sun, Intelligent rotating machinery fault diagnosis based on deep learning using data augmentation, Journal of Intelligent Manufacturing, 31 (2020) 433-452.

[33] J. Lee, M. Kim, J.U. Ko, J.H. Jung, K.H. Sun, B.D. Youn, Asymmetric inter-intra domain alignments (AIIDA) method for intelligent fault diagnosis of rotating machinery, Reliability Engineering & System Safety, 218 (2022) 108186.

[34] Y. Han, B. Tang, L. Deng, An enhanced convolutional neural network with enlarged receptive fields for fault diagnosis of planetary gearboxes, Computers in Industry, 107 (2019) 50-58.

[35] L. Meng, M. Zhao, Z. Cui, X. Zhang, S. Zhong, Empirical mode reconstruction: Preserving intrinsic components in data augmentation for intelligent fault diagnosis of civil aviation hydraulic pumps, Computers in Industry, 134 (2022) 103557.

[36] S. Ma, F. Chu, Ensemble deep learning-based fault diagnosis of rotor bearing systems, Computers in Industry, 105 (2019) 143-152.

[37] J.U. Ko, J.H. Jung, M. Kim, H.B. Kong, J. Lee, B.D. Youn, Multi-task learning of classification and denoising (MLCD) for noise-robust rotor system diagnosis, Computers in Industry, 125 (2021) 103385.

[38] T. Zhang, S. Liu, Y. Wei, H. Zhang, A novel feature adaptive extraction method based on deep learning for bearing fault diagnosis, Measurement, 185 (2021) 110030.

[39] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42 (2011) 463-484.

[40] Z. Wu, H. Zhang, J. Guo, Y. Ji, M. Pecht, Imbalanced bearing fault diagnosis under variant working conditions using cost-sensitive deep domain adaptation network, Expert Systems with Applications, 193 (2022) 116459.

[41] J.M. Johnson, T.M. Khoshgoftaar, Survey on deep learning with class imbalance, Journal of Big Data, 6 (2019) 1-54.

[42] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, Advances in Neural Information Processing Systems, 27 (2014).

[43] I. Goodfellow, Nips 2016 tutorial: Generative adversarial networks, arXiv preprint arXiv:1701.00160, (2016).

[44] D. Zhao, S. Liu, D. Gu, X. Sun, L. Wang, Y. Wei, H. Zhang, Enhanced data-driven fault diagnosis for machines with small and unbalanced data based on variational auto-encoder, Measurement Science and Technology, 31 (2019) 035004.

[45] S. Zhang, F. Ye, B. Wang, T.G. Habetler, Semi-supervised bearing fault diagnosis and classification using variational autoencoder-based deep generative models, IEEE Sensors Journal, 21 (2020) 6476-6486.

[46] C. Che, H. Wang, R. Lin, X. Ni, Deep meta-learning and variational autoencoder for coupling fault diagnosis of rolling bearing under variable working conditions, Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, (2022) 09544062221101834.

[47] S. Shao, P. Wang, R. Yan, Generative adversarial networks for data augmentation in machine fault diagnosis, Computers in Industry, 106 (2019) 85-93.

[48] R. Wang, S. Zhang, Z. Chen, W. Li, Enhanced generative adversarial network for extremely imbalanced fault diagnosis of rotating machine, Measurement, 180 (2021) 109467.

[49] X. Gao, F. Deng, X. Yue, Data augmentation in fault diagnosis based on

the Wasserstein generative adversarial network with gradient penalty, Neurocomputing, 396 (2020) 487-494.

[50] S. Suh, H. Lee, J. Jo, P. Lukowicz, Y.O. Lee, Generative oversampling method for imbalanced data on bearing fault detection and diagnosis, Applied Sciences, 9 (2019) 746.

[51] L. Ma, Y. Ding, Z. Wang, C. Wang, J. Ma, C. Lu, An interpretable data augmentation scheme for machine fault diagnosis based on a sparsity-constrained generative adversarial network, Expert Systems with Applications, (2021) 115234.

[52] Y. Peng, Y. Wang, Y. Shao, A novel bearing imbalance Fault-diagnosis method based on a Wasserstein conditional generative adversarial network, Measurement, 192 (2022) 110924.

[53] A. Voynov, A. Babenko, Unsupervised discovery of interpretable directions in the GAN latent space, International Conference on Machine Learning, PMLR, 2020, pp. 9786-9796.

[54] R. Hecht-Nielsen, Theory of the backpropagation neural network, Neural networks for perception, Elsevier, 1992, pp. 65-93.

[55] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980, (2014).

[56] H. Oh, J.H. Jung, B.C. Jeon, B.D. Youn, Scalable and unsupervised feature engineering using vibration-imaging and deep learning for rotor system

diagnosis, IEEE Transactions on Industrial Electronics, 65 (2017) 3539-3549.

[57] C. Wu, P. Jiang, C. Ding, F. Feng, T. Chen, Intelligent fault diagnosis of rotating machinery based on one-dimensional convolutional neural network, Computers in Industry, 108 (2019) 53-61.

[58] L. Wen, X. Li, L. Gao, Y. Zhang, A new convolutional neural network-based data-driven fault diagnosis method, IEEE Transactions on Industrial Electronics, 65 (2017) 5990-5998.

[59] D. Zhao, T. Wang, F. Chu, Deep convolutional neural network based planet bearing fault classification, Computers in Industry, 107 (2019) 59-66.

[60] M.M. Islam, J.-M. Kim, Automated bearing fault diagnosis scheme using 2D representation of wavelet packet transform and deep convolutional neural network, Computers in Industry, 106 (2019) 142-153.

[61] M.-T. Nguyen, V.-H. Nguyen, S.-J. Yun, Y.-H. Kim, Recurrent neural network for partial discharge diagnosis in gas-insulated switchgear, Energies, 11 (2018) 1202.

[62] T. De Bruin, K. Verbert, R. Babuška, Railway track circuit fault diagnosis using recurrent neural networks, IEEE Transactions on Neural Networks and Learning Systems, 28 (2016) 523-533.

[63] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, R.X. Gao, Deep learning and its applications to machine health monitoring, Mechanical Systems and

Signal Processing, 115 (2019) 213-237.

[64] R. Nisbet, J. Elder, G. Miner, Handbook of statistical analysis and data mining applications, Academic Press, 2009.

[65] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, Deep learning, MIT Press Cambridge, 2016.

[66] M.-A. Lutz, S. Vogt, V. Berkhout, S. Faulstich, S. Dienst, U. Steinmetz, C. Gück, A. Ortega, Evaluation of anomaly detection of an autoencoder based on maintenace information and scada-data, Energies, 13 (2020) 1063.

[67] J. Gehring, Y. Miao, F. Metze, A. Waibel, Extracting deep bottleneck features using stacked auto-encoders, 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 3377-3381.

[68] C.J. Willmott, K. Matsuura, Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance, Climate Research, 30 (2005) 79-82.

[69] D.W. Scott, Multivariate density estimation: theory, practice, and visualization, John Wiley & Sons, 2015.

[70] S.P. Neill, M.R. Hashemi, Fundamentals of ocean renewable energy: generating electricity from the sea, Academic Press, 2018.

[71] A. Stetco, F. Dinmohammadi, X. Zhao, V. Robu, D. Flynn, M. Barnes, J. Keane, G. Nenadic, Machine learning methods for wind turbine condition

monitoring: A review, Renewable Energy, 133 (2019) 620-635.

[72] G.P. Tolstov, Fourier series, Courier Corporation, 2012.

[73] F.M.L. Ribeiro, Machinery Fault Database (MAFAULDA) - Multivariate time-series acquired by sensors on a SpectraQuest's Machinery Fault Simulator (MFS) Alignment-Balance-Vibration (ABVT), 2018.

[74] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7132-7141.

[75] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026-1034.

[76] A.Y. Ng, Feature selection, L 1 vs. L 2 regularization, and rotational invariance, Proceedings of the twenty-first International Conference on Machine Learning, 2004, pp. 78.

[77] B.C. Jeon, J.H. Jung, B.D. Youn, Y.-W. Kim, Y.-C. Bae, Datum unit optimization for robustness of a journal bearing diagnosis system, International Journal of Precision Engineering and Manufacturing, 16 (2015) 2411-2425.

[78] Z. Xia, S. Xia, L. Wan, S. Cai, Spectral regression based fault feature extraction for bearing accelerometer sensor signals, Sensors, 12 (2012) 13694-13719.

[79] J.H. Jung, B.C. Jeon, B.D. Youn, M. Kim, D. Kim, Y. Kim, Omnidirectional regeneration (ODR) of proximity sensor signals for robust diagnosis of journal bearing systems, Mechanical Systems and Signal Processing, 90 (2017) 189-207.

[80] M.A. Marins, F.M. Ribeiro, S.L. Netto, E.A. Da Silva, Improved similarity-based modeling for the classification of rotating-machine failures, Journal of the Franklin Institute, 355 (2018) 1913-1930.

[81] R.M. Souza, E.G. Nascimento, U.A. Miranda, W.J. Silva, H.A. Lepikson, Deep learning for diagnosis and classification of faults in industrial rotating machinery, Computers & Industrial Engineering, 153 (2021) 107060.

[82] R. Caruana, Multitask learning, Machine learning, 28 (1997) 41-75.

[83] M.L. Seltzer, J. Droppo, Multi-task learning in deep neural networks for improved phoneme recognition, 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 6965-6969.

[84] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature, 521 (2015) 436-444.

[85] L.F. Villa, A. Reñones, J.R. Perán, L.J. De Miguel, Angular resampling for vibration analysis in wind turbines under non-linear speed fluctuation, Mechanical Systems and Signal Processing, 25 (2011) 2157-2168.

[86] F. Bonnardot, M. El Badaoui, R. Randall, J. Daniere, F. Guillet, Use of the acceleration signal of a gearbox in order to perform angular resampling

(with limited speed fluctuation), Mechanical Systems and Signal Processing, 19 (2005) 766-785.

[87] M. Kim, J.H. Jung, J.U. Ko, H.B. Kong, J. Lee, B.D. Youn, Direct Connection-Based Convolutional Neural Network (DC-CNN) for Fault Diagnosis of Rotor Systems, IEEE Access, 8 (2020) 172043-172056.

[88] S.-M. Lee, Y.-S. Choi, Fault diagnosis of partial rub and looseness in rotating machinery using Hilbert-Huang transform, Journal of Mechanical Science and Technology, 22 (2008) 2151-2162.

[89] B. Pang, G. Tang, C. Zhou, T. Tian, Rotor fault diagnosis based on characteristic frequency band energy entropy and support vector machine, Entropy, 20 (2018) 932.

[90] W. Sun, J. Chen, J. Li, Decision tree and PCA-based fault diagnosis of rotating machinery, Mechanical Systems and Signal Processing, 21 (2007) 1300-1317.

[91] X. Zhu, D. Hou, P. Zhou, Z. Han, Y. Yuan, W. Zhou, Q. Yin, Rotor fault diagnosis using a convolutional neural network with symmetrized dot pattern images, Measurement, 138 (2019) 526-535.

[92] L.-L. Jiang, H.-K. Yin, X.-j. Li, S.-W. Tang, Fault diagnosis of rotating machinery based on multisensor information fusion using SVM and time-domain features, Shock and Vibration, 2014 (2014).

국문 초록

회전기계 내 저해상도 및 고해상도 신호를 활용한 딥러닝 기반 거시적 및 미시적 고장 진단 방법론

서울대학교 대학원
기계항공공학부
고 진 욱

회전기계는 제조 및 발전과 같이 다양한 산업 현장에서 널리 사용되고 있다. 회전기계의 예기치 못한 고장은 막대한 경제적, 인적 손실을 야기할 수 있다. 이러한 상황을 예방하기 위해서, 회전기계의 건전성 상태를 정확히 관리하는 것을 목표로 하는 고장 진단 연구가 주목을 받고 있다. 고장 진단 기법들은 목표 시스템의 이상을 정확히 감지하고 건전성 상태를 식별하는 것을 목표로 한다. 최근에는 딥러닝 기반 연구들이 자동적으로 유의미한 특성인자를 추출하는 능력 덕분에 뛰어난 진단 성능을 보이고 있다.

회전기계에서는 해상도가 서로 다른 운전 신호 및 진동 신호가 취득된다. 저샘플링 주파수로 취득되는 운전 신호는 실시간으로 얻어지고, 시스템을 전반적으로 관리할 수 있는 다양한 종류의 상태

변수를 포함하고 있다. 진동 신호는 고샘플링 주파수로 측정되고 실시간이 아니라, 고장이 발생하면 취득된다. 해상도가 다른 두 신호를 활용해서 고장 진단의 두 가지 하위 테스크인 이상 감지 및 고장 식별이 수행된다. 운전 신호를 가지고 수행되는 이상 감지는 시스템의 이상을 가능하면 빨리 감지하는 것을 목표로 한다. 이것은 거시적 수준의 고장 진단으로 여겨진다. 이상 감지 수행 시, 정상 데이터는 비지도 학습 방식으로 모델링 되고, 잔차 신호가 계산된 후에 기준치가 결정된다. 잔차 신호가 기준치를 초과하면, 해당 시스템은 이상이 있다고 판단된다. 고장 식별은 진동 신호를 사용해서 시스템의 건전성 상태를 분류하는 것을 목표로 한다. 이것은 미시적 수준의 고장 진단으로 여겨진다. 지도학습 방식을 활용해 딥러닝 기반 진단기를 학습시킨다. 그러므로 많은 양의 라벨 데이터가 학습에 필요하다. 실제 산업 현장에서는 고장 데이터가 부족하기 때문에, 부족한 고장 데이터를 증량하기 위한 데이터 증량 기법이 필수적이다. 최근에는 변분적 오토인코더나 적대적 생성 신경망을 활용한 증량 기법이 널리 연구되고 있다.

이상 감지와 고장 식별은 각자 따로 연구되었다. 만약 두 테스크가 통합된다면, 거시적 및 미시적 고장 진단이 수행될 수 있다. 하지만, 딥러닝 기반 거시적 및 미시적 고장 진단 기법을 개발하는 데 해결해야 할 세 가지 문제점이 있다. 첫째, 기존 이상 감지 기법들은 시스템에 아무 이상이 없어도 오감지를 빈번하게 발생시켰다. 기존 방법들은 정상 데이터를 부정확하게 모델링하거나 기준치를 잘못 설정해서 정상 데이터에 존재하는 변동을 고려하지 못한다. 둘째, 기존 생성 신경망 기반 모델들은 구조적 특징에 기인한 한계점을 갖고 있다. 다양한 길이의 신호가 만들어질 수 없고, 잠재 벡터가 잘못 샘플링되면 잘못된

샘플이 생성될 수 있다. 건전성 분류와 관련된 마지막 이슈는 분류기의 성능이 입력 데이터의 노이즈에 영향을 받을 수 있다는 점이다. 노이즈는 데이터 분포를 왜곡할 수 있기 때문에, 분류기가 노이즈가 있는 데이터를 올바르게 분류하는 것은 어렵다.

이러한 현황을 바탕으로, 본 박사학위 논문에서는 회전기계 내 운전 및 진동 신호를 활용한 딥러닝 기반 거시적 및 미시적 고장 진단 기법을 제안한다. 첫 번째 연구는 오감지를 줄이는 이상 감지를 위해서, 새로운 모델링 및 기준치 설정 기법들을 제안한다. 제안하는 모델링 방법은 오토인코더에 앙상블 및 디노이징 기법을 적용하여 개발됐다. 또한, 결과값과 잔차 신호 사이의 결합분포를 사용해서 동적 기준치를 설정하는 기법도 개발됐다. 이를 통해, 제안하는 방법은 정상 데이터의 변동을 고려하여 오감지를 상당히 줄일 수 있다. 두 번째 연구에서는 다양한 길이의 신호를 만들기 위한 새로운 생성 모델을 제안한다. 제안하는 네트워크는 입력과 출력이 시간 및 진폭이고, 학습 데이터의 주파수 정보를 학습하도록 설계됐다. 제안하는 모델은 나이키스트 이론과 같은 신호 처리 지식을 반영하기 위해서 신중히 설계됐다. 학습 후에, 제안하는 방법은 원하는 시간대의 다양한 길이의 신호를 만들 수 있다. 또한, 제안하는 네트워크는 어텐션 블록 덕분에 특성 주파수 성분에 집중할 수 있다. 세 번째 연구는 분류와 디노이징 테스크를 동시에 배우는 학습 기법을 제안한다. 제안하는 기법에서는 두 가지 테스크를 동시에 학습하기 위해서 다중 테스크 학습 기법이 사용된다. 제안하는 기법은 네트워크 종류에 상관없이 어떠한 딥러닝 알고리즘에 적용될 수 있다. 제안하는 방법으로 학습된 분류기는 건전성 상태를 잘 분류할 뿐만 아니라, 입력 신호의 노이즈도 제거할 수 있다.

주요어: 거시적 및 미시적 고장 진단
회전기계
저해상도 운전 신호
고해상도 진동 신호
딥러닝
건전성 예측 및 관리

학 번: 2017-20541