IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 August 2022, accepted 3 September 2022, date of publication 8 September 2022, date of current version 19 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3205352

RESEARCH ARTICLE

A Real-Time Fault Detection Framework Based on Unsupervised Deep Learning for Prognostics and Health Management of Railway Assets

MINORU SHIMIZU[1], SURESH PERINPANAYAGAM[1], (Member, IEEE), AND BERNADIN NAMOANO[2]

[1]Integrated Vehicle Health Management Centre, Cranfield University, Bedfordshire, MK43 0AL Cranfield, U.K.
[2]Digital Engineering and Manufacturing Centre, Cranfield University, Bedfordshire, MK43 0AL Cranfield, U.K.

Corresponding author: Minoru Shimizu (minoru.shimizu@cranfield.ac.uk)

ABSTRACT Fault detection based on deep learning has been intensively investigated in the recent decade due to increasing availability of data and its ability to engineer features with deep neural network architectures. Despite much attention to its application, the major challenge is the lack of available labelled datasets to build the models since maintenance is usually conducted regularly to avoid significant defects. This paper aims to propose a successful real-time fault detection framework based on unsupervised deep learning using only healthy normal data. The approach is based on autoencoder architecture and a one-class support vector machine as a classifier. As a case study, large real-world datasets acquired from railway door systems have been employed. The five different types of deep learning models and a one-class classifier are trained and comprehensively validated based on performance metrics and sensitivity analysis. In addition, two experiments have been carried out to verify the model's adaptability and robustness to variational time-series data. The result shows a typical autoencoder is the least sensitive to a decision boundary set by the one-class classifier. However, the two experiments show that the fault detection accuracy for a bidirectional long short-term memory-based autoencoder is considerably higher than other autoencoder-based models at 0.970 and 0.966 as F1 score, meaning only this model is adaptable and robust to variational data. The experimental result allows us to obtain the understandability of the deep learning models. Furthermore, the regions of anomalies are localised with unsupervised models, which enables diagnosing the cause of failure.

INDEX TERMS Fault detection, PHM, signal processing, unsupervised deep learning, machine learning, data-driven approach, AE, Bi-LSTM, railway, door systems.

I. INTRODUCTION

Prognostics and Health Management (PHM) is a comprehensive technology which enables engineers to turn data and health states into information that will improve our knowledge of the system and provide a strategy to maintain the system in its originally intended function. While it has rooted in the aerospace industry, it is now explored in many applications, including manufacturing, automotive, railway and heavy industry [1]. PHM has significant benefits in reducing support and operating costs. An unexpected one-day stoppage in the machinery industry may cost as much as up to 100,000 to 200,000 euros [2]. Furthermore, most importantly, maintenance tasks are significant from a safety point of view. Inadequate maintenance can lead to a devastating incident. For example, on the 10 May 2002, a train travelling from London to Norfolk in the UK derailed at Potters Bar railway station, causing seven deaths and injuring over seventy people. The derailment was due to the failure of points; one of the main factors is that points had been poorly maintained and were out of adjustment in some respects [3]. Thus, an accident related to inappropriate maintenance could be a significant disaster that causes social anxiety and lead to the loss of social credibility of the industry.

The associate editor coordinating the review of this manuscript and approving it for publication was Li He.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

Fault detection serves an important role in PHM and has been investigated in recent decades. Researchers in such diverse disciplines as medicine, engineering and sciences have been developing methodologies to detect fault or anomaly conditions, pinpoint or isolate which component or object in a system or process is faulty, and decide on the potential impact of a failing or failed component on the health of the system [4].

In this area of study, the methodologies usually centre on model-based or data-driven approaches. Model-based approaches incorporate a physical understanding of the systems through mathematical representations and include system modelling. The output of the model is then compared with the actual output measurement throughout the residual analysis [5], [6]. However, the mechanical system contains many components interconnected with various uncertainties, which makes the modelling approach of limited value. On the other hand, data-driven approaches use statistical pattern recognition and machine learning to detect changes [5]. Data-driven approaches do not require mathematical modelling of the systems and have gained much attention with the increasing availability of data.

The data-driven approaches include traditional machine learning (ML) and deep learning (DL) approaches. The traditional ML approaches need several steps such as preprocessing data, and feature extraction before building a model. However, the manual feature extraction demands expert domain knowledge, which makes traditional ML approaches difficult. On the other hand, DL approaches enable fault detection models to be created without handcrafted features employing a deep network architecture, which is a remarkable advantage compared to traditional ML.

Fault detection techniques based on DL are categorised into supervised and unsupervised learning approaches. The supervised DL approaches require labelled datasets to train a model. So far, much fault detection research has been conducted based on supervised DL approaches, including deep neural network (DNN) [7], [8], two-dimensional convolutional neural network (2D CNN) [9], [10], [11], [12], one-dimensional convolutional neural network (1D CNN) [13], [14], gated recurrent units (GRU) [15], and long short-term memory (LSTM) [16], [17]. However, the requirement of a sufficient number of labelled datasets is a significant drawback of supervised approaches since faulty data is always insufficient due to conservative maintenance to avoid catastrophic incidents. In addition, only anticipated faults can be detected in the case of supervised learning approaches. Moreover, in general, supervised DL needs more training datasets than traditional ML approaches because of deep network architecture including many parameters to be learned.

Contrary to the supervised approaches, unsupervised DL approaches do not require labelled datasets. In the case of industrial data acquisition, healthy normal data is widely available, while faulty data is scarce. The unsupervised DL approach aims at extracting relevant characteristics of the input data itself. If healthy data is used as a training dataset, engineered features with unsupervised DL models can represent characteristics of healthy data. Then, these features can be used for classification tasks for the purpose of fault detection. Previous research has been proposed based on unsupervised learning approaches, such as stacked autoencoder [18], [19], denoising autoencoder (DAE) [20], sparse autoencoder (SAE) [21], variational autoencoder (VAE) [22], and deep belief network (DBN) [23]. However, in the literature described above, the following step of using the engineered features is a supervised classification model, which means there is still a need for labelled datasets even though features are created in an unsupervised manner. Despite the drawbacks, little research using only healthy data to build an entire fault detection system can be found in the literature [24], [25], [26].

In addition, it is crucial to build a reliable DL model based on the rationale behind a network architecture. However, it might be challenging to understand what the DL models mean because AI models are black boxes in nature, meaning that the inner mechanism to produce outputs in these methods is unknown [27]. Despite the significance of understandability and reliability of the models, the unsupervised DL algorithms found in [24], [25], and [26] are chosen and validated empirically applying fault detection accuracy. It is also pointed out in [28] that researchers have not explained the reasons as to why or how these DL architectures have been selected. In that case, it might be required to build DL architectures comprehensively by only using fault detection accuracy, which is impractical.

This paper aims to propose a successful real-time fault detection framework based on unsupervised DL using only healthy normal data. The approach is based on autoencoders (AEs) and a one-class support vector machine (SVM) as a classifier. As a case study, large real-world datasets acquired from railway door systems have been employed. The five different types of DL models and one-class SVM are trained with healthy normal data and comprehensively validated based on performance metrics and sensitivity analysis. In addition, two experiments have been carried out in order to verify the model's adaptability and robustness to variational time-series data. To our best knowledge, this is the first paper to propose a fault detection framework based on unsupervised DL for railway door systems. The main contributions of the paper are summarised as follows:

1) We propose a real-time fault detection framework for railway assets based on unsupervised DL approaches using only healthy normal data.
2) We comprehensively build and compare representative unsupervised DL models based on fault detection accuracy and sensitivity analysis.
3) We verify the model adaptability and robustness to variational time-series data and obtain understandability of the DL models.
4) We visualise reconstructed profiles generated by DL models and localise regions of anomalies, which enables diagnosing the cause of failure.

The remainder of this article is organised as follows. Section II provides the proposed methodology. The result and discussion are given in section III. Finally, section IV concludes this article.

II. PROPOSED METHODOLOGY

A. FAULT DETECTION WORKFLOW

The proposed real-time fault detection workflow for train doors is shown in Figure 1. The workflow is divided into two procedures, offline and online. In the offline, current and encoder signals acquired from railway assets are used as training datasets to train an unsupervised DL model and a one-class SVM model to build a fault detection model. In this approach, time-series current and encoder signals are pre-processed to be aligned and eliminate noise by a low pass filter, followed by segmentation into the opening and closing operations. Then current and encoder signals are standardised, which means signals are divided by the standard deviation of each signal, in order to make each signal have the same standard deviation. Once the training dataset is prepared, the unsupervised DL model is trained with the training dataset. The training data is then reconstructed with the DL model. The sum of squared errors (SSE) between input and reconstructed data is calculated. Finally, the one-class classification model, which is a one-class SVM model, is trained.

The DL model and the one-class classification model created offline are implemented on the online procedure to detect faults in real-time. The current and encoder signals are pre-processed, segmented, and standardised in the same manner as offline. Then the real-time input data is reconstructed with the DL model built offline, followed by SSE calculation and prediction with the one-class SVM model. The fault detection workflow can be executed once one door operation is terminated so that faulty behaviour can be detected as early as possible, which allows us to maintain machinery before breakdown.

The proposed method offers remarkable advantages in terms of practical fault detection applications available in the industry. Firstly, fault detection models can be built by using only healthy normal data acquired in the industry, where labelled datasets are always insufficient. Secondly, the proposed method enables fault detection models to be created without handcrafted features employing a deep network architecture. Thirdly, both anticipated and unanticipated faults can be detected and localised due to its unsupervised approaches. Fourthly, the fault detection model built offline can be improved by additional operational data that enable fault detection to be more accurate and reliable.

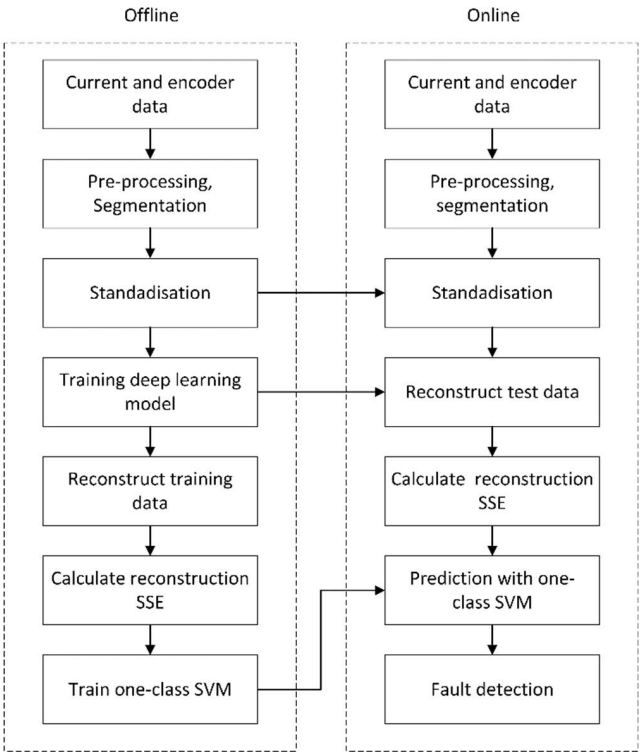

FIGURE 1. The proposed real-time fault detection workflow.

B. DATASET

1) DATA ACQUISITION

In this study, large real-world datasets acquired from railway door systems have been employed. An electric door is considered, which is composed of a voltage power source, a DC motor, a door control unit (DCU), a transmission and door leaves. In short, a DC motor, powered by a voltage source and controlled by DCU, can output the specified shaft angular velocity and torque, which are transmitted to transmission so that the door leaves can move in a pre-designed manner. The door data, which consists of current and encoder signals, is collected through the communication port from the DCU at a frequency of 50 Hz. The time lag is often observed between the motion profile and the current. To align the time-series, the dynamic time warping (DTW) method is used for the first alignment. The low pass filter is applied on a window of 0.25 seconds, representing five consecutive measurement time intervals to reduce noise carried by both current and encoder signals.

2) AN EXAMPLE OF THE SIGNAL PROFILE

An example of the signal profiles of the opening and closing operations is shown in Figure 2. In the opening profile, the speed and current increase steadily up to a maximum, followed by a slight curve, and then decrease to zero. The closing profile follows a similar pattern with two main differences in the current. One is that the peak in the closing profile is lower than the opening. The second is an abrupt change at the end of the closing profile, followed by a slight bump of the speed, which promotes pushing the door to its maximal reachable position where a locking process can be triggered [29].

In this research, current signals from the closing operation are used as an example for a fault detection purpose. It is also possible to employ opening operation instead since the

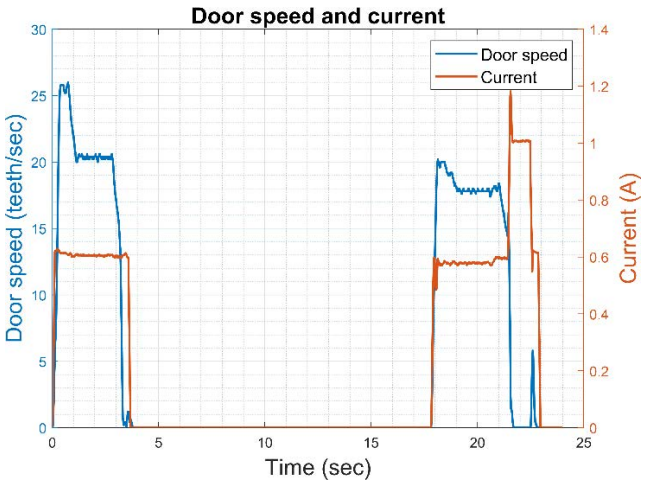

FIGURE 2. Door speed and current signals.

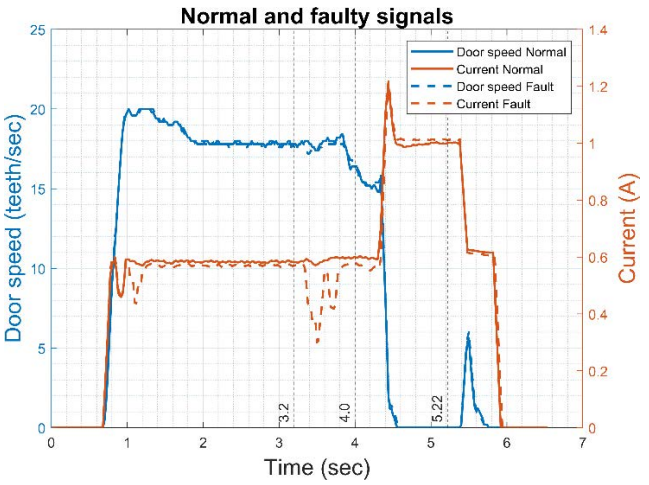

FIGURE 3. Normal and faulty signals of closing operation.

proposed method is unspecific to types of operations and fault modes. The example of the normal and faulty signals of closing operation is shown in Figure 3. The normal current signal has flat curves from 3.2 seconds to 4.0 seconds, while there are negative peaks and fluctuations in that of faulty data. It should be noted that concrete fault types are unidentifiable in this research. The experimental current signals of the representative fault modes for electro-mechanical actuators (MAEs) provided in [30] have been compared to the faulty signals in the railway door operation. However, none of the fault modes resembles the faulty signals in the railway door operation. It is possible, therefore, that the observed faulty behaviour could be accompanied by several fault modes because the train door system contains many components. The identification and diagnosis of fault modes are out of the scope of this research. In order to apply DL models with time-series data, it is crucial to make each current and door speed profile in the dataset be the same length in time. Thus, the profiles over 5.22 seconds are eliminated in the example dataset, which means each profile includes 262 data points.

3) TRAIN AND TEST DATASET

The acquired operational current and encoder signals of closing operation are split into training and test datasets as given in Table 1. It is noteworthy that the training dataset includes only normal data because the DL models and one-class classifier are trained with normal data in an unsupervised manner. Then, the test dataset is employed to validate the fault detection accuracy.

C. DEEP LEARNING MODELS

1) AUTO-ENCODER

Many unsupervised DL algorithms are based on the idea of an autoencoder (AE). The AE is a special case of a feed-forward neural network that is trained to attempt to copy its input to its output. The network consists of two parts: an encoder and a decoder, as shown in Figure 4. The encoder is a function that maps the input into lower-dimensional space,

TABLE 1. The train and test dataset.

	Normal	Anomaly	Total
Train	100	0	100
Test	100	100	200

where compressed data is used as an internal representation of the original input. The decoder is a reverse process of the encode that produces a reconstruction by using internal representation. The AE is trained the same as a feedforward neural network, where a back-propagation algorithm computes gradients of a loss function, and gradient descent is used to optimise parameters.

The encoding and decoding process can be represented using the following equations [31]:

$$\boldsymbol{h} = f(\boldsymbol{W_e}\boldsymbol{X} + \boldsymbol{b_e}) \tag{1}$$

$$\boldsymbol{Y} = g(\boldsymbol{W_d}\boldsymbol{h} + \boldsymbol{b_d}) \tag{2}$$

$$\boldsymbol{X} = (\boldsymbol{x^1}\ \boldsymbol{x^2}\ \boldsymbol{x^3}\ \cdots\ \boldsymbol{x^m}) \tag{3}$$

$$\boldsymbol{Y} = (\boldsymbol{y^1}\ \boldsymbol{y^2}\ \boldsymbol{y^3}\ \cdots\ \boldsymbol{y^m}) \tag{4}$$

where $\boldsymbol{X}$ and $\boldsymbol{Y}$ are n by m matrices containing n by 1-dimensional column vector $\boldsymbol{x^i}$ ($i = 1, \ldots, m$), and m observations, $\boldsymbol{W_e}$ and $\boldsymbol{W_d}$, and $\boldsymbol{b_e}$ and $\boldsymbol{b_d}$ are weight matrices and biases, respectively, and f and g are activation functions. If $\boldsymbol{X^{train}}$ is used as a training dataset, the AE is trained to optimise weight parameters and biases to minimise the loss function given in the following equation:

$$L\left(\boldsymbol{X^{train}}; \boldsymbol{Y^{train}}\right) = \sum_{i}^{n}\sum_{j}^{m}\left\|\boldsymbol{X_{ij}^{train}} - \boldsymbol{Y_{ij}^{train}}\right\|^2 + \lambda(\|\boldsymbol{W_e}\|^2 + \|\boldsymbol{W_d}\|^2) \tag{5}$$

where $\boldsymbol{Y^{train}}$ is a prediction calculated by equations (1) and (2), λ is a L2 regularisation hyperparameter, which controls the strength to force weight parameters to be small to avoid overfitting. As a definition of the loss function, the optimisation goal is to minimise the difference between $\boldsymbol{X^{train}}$ and

$\boldsymbol{Y}^{train}$ while preventing weight parameters from being large. The optimisation process makes $\boldsymbol{Y}^{train}$ identical to $\boldsymbol{X}^{train}$.

Once the AE is trained, the model can be used to obtain the difference between input and output data. The difference is called a reconstruction error. The reconstruction error is calculated as the sum of squared error (SSE), the mean squared error (MSE), and the mean of SSE (MSSE), given in the following equations:

$$SSE^i = \left\| \boldsymbol{x}^i - \boldsymbol{y}^i \right\|^2 \tag{6}$$

$$MSE^i = \frac{1}{n} SSE^i \tag{7}$$

$$MSSE = \frac{1}{m} \sum_{j=1}^{m} SSE^i \tag{8}$$

It is noteworthy that the reconstruction error differs depending on input data characteristics. For instance, suppose the input data resembles training data, meaning the characteristics of input data are equivalent to those of training data, the reconstruction error can be small enough to be the same as that of training data. However, the reconstruction error can be larger if input data have a different tendency from training data. In this research, the capability of AEs relating to the reconstruction error is employed for fault detection purposes.

The typical AE model is built to compare representative DL models, as given in Table 2. Firstly, the standardised current and encoder signals, described in Figure 1, are concatenated as a 1 by 524 column vector. Then, the concatenated data is used to train the AE model. The activation function for an encoder and a decoder layer is the sigmoid function. Notably, the dimension of the input layer, 1 by 524, is determined depending on current and door speed signals, whose profiles include 262 data points, respectively. Suppose the proposed AE architecture is employed in another example. Then, the input dimension is modifiable according to the signal profiles.

2) BIDIRECTIONAL LONG SHORT-TERM MEMORY AUTOENCODER

Long short-term memory (LSTM) is one of the most popular types of recurrent neural networks (RNNs), enabling information to be preserved over many time steps, and was initially proposed by Hochreiter and Schmidhuber [32]. The LSTM includes gating units, which control the flow of information in the LSTM [33]. The forward propagation process can be expressed as the following equations:

$$\tilde{\boldsymbol{s}}_t = tanh(\boldsymbol{W}_c \boldsymbol{h}_{t-1} + \boldsymbol{U}_c \boldsymbol{x}_t + \boldsymbol{b}_c) \tag{9}$$

$$\boldsymbol{f}_t = \sigma(\boldsymbol{W}_f \boldsymbol{h}_{t-1} + \boldsymbol{U}_f \boldsymbol{x}_t + \boldsymbol{b}_f) \tag{10}$$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i \boldsymbol{h}_{t-1} + \boldsymbol{U}_i \boldsymbol{x}_t + \boldsymbol{b}_i) \tag{11}$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o \boldsymbol{h}_{t-1} + \boldsymbol{U}_o \boldsymbol{x}_t + \boldsymbol{b}_o) \tag{12}$$

$$\boldsymbol{s}_t = \boldsymbol{s}_{t-1} \odot \boldsymbol{f}_t + \boldsymbol{i}_t \odot \tilde{\boldsymbol{s}}_t \tag{13}$$

$$\boldsymbol{h}_t = tanh(\boldsymbol{s}_t) \odot \boldsymbol{o}_t \tag{14}$$

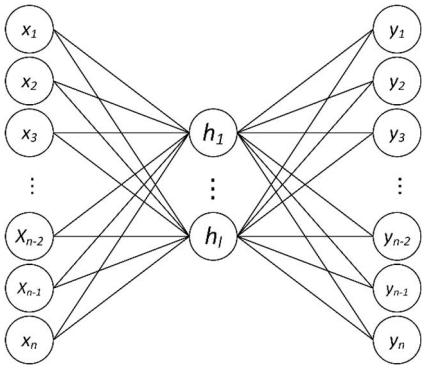

FIGURE 4. AE architecture.

TABLE 2. The AE architecture.

No.	Layer	Output shape	Learnable parameter #
1	Input	1 × 524	0
2	Encoder	1 × 10	5250
3	Decoder	1 × 524	5764
4	Output	1 × 524	0

where $\boldsymbol{x}_t$ and $\boldsymbol{h}_t$ are input and hidden state, $\boldsymbol{f}_t$, $\boldsymbol{i}_t$, and $\boldsymbol{o}_t$ are forget gate, input gate and output gate, $\boldsymbol{s}_t$ is state unit at time step t, correspondingly. The $\boldsymbol{b}_c$, $\boldsymbol{b}_f$, $\boldsymbol{b}_i$, and $\boldsymbol{b}_o$ are bias vectors, σ is an activation function, and $\boldsymbol{W}_c$, $\boldsymbol{W}_f$, $\boldsymbol{W}_i$, $\boldsymbol{W}_o$, $\boldsymbol{U}_c$, $\boldsymbol{U}_f$, $\boldsymbol{U}_i$, and $\boldsymbol{U}_o$ are weight matrices, respectively. The $\odot$ symbol denotes element-wise multiplication. The forget gate $\boldsymbol{f}_t$ determines if each element of $\boldsymbol{s}_{t-1}$ is remembered or forgotten. The input gate $\boldsymbol{i}_t$ determines if each element of the state unit $\boldsymbol{s}_t$ is updated by the latest information at the current time step. The output gate $\boldsymbol{o}_t$ determines if each element of the state unit is transferred to the hidden state [33]. The calculation flow can be described as an LSTM block diagram, as shown in Figure 5.

A remarkable advantage of the LSTM compared to RNNs is making the weight parameters controlled by gating units. In this case, the time scale of integration with past information can be changed dynamically because gating units are also determined by sequential input itself.

In addition, the LSTM also mitigates vanishing and exploding gradients. The state unit is essentially copied from time step to time step, given that forget gate and input gate are close to zero and one, respectively, as shown in equation (13). In this way, gradient expression does not accumulate over time, which enables preventing gradients from vanishing or exploding. That allows LSTM networks to learn long-term dependencies.

The LSTM captures the past information, which are $\boldsymbol{x}_1$, $\boldsymbol{x}_2$,..., $\boldsymbol{x}_{t-1}$ and the present input $\boldsymbol{x}_t$. However, in many applications, the whole input sequence needs to be used to output accurate predictions. Bidirectional LSTM (Bi-LSTM) can be applied to address that need. As the name suggests, Bi-LSTM combines an LSTM that moves forward through time beginning from the start of the sequence with another

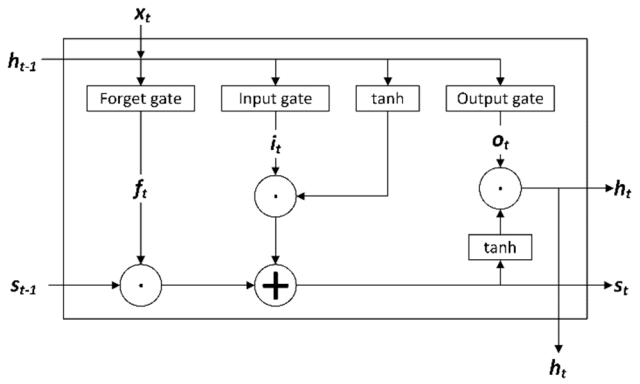

FIGURE 5. LSTM block diagram.

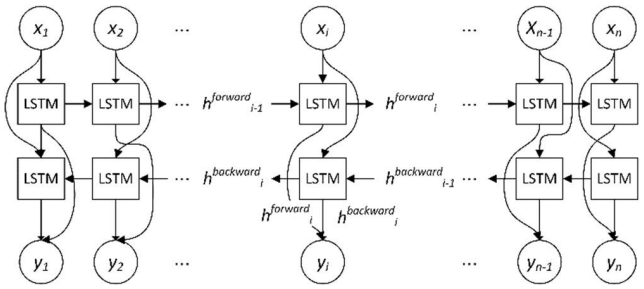

FIGURE 6. Bidirectional LSTM.

LSTM that moves backwards through time beginning from the end of the sequence, as described in Figure 6. The calculation is written in the following equations:

$$h^{BiLSTM} = [h^{forward}, h^{backward}] \tag{15}$$

$$y_t = tanh(W_y h_{BiLSTM} + b_y) \tag{16}$$

where $h^{forward}$ and $h^{backward}$ are hidden states of a forward LSTM and a backward LSTM, b_y and W_y are a bias vector and a weight matrix for activation of bidirectional LSTM y_t, respectively. The $h^{forward}$ and $h^{backward}$ can be calculated separately based on equations (9-14). This allows the bidirectional LSTM to compute activation y_t depending on both the past and future information.

A Bi-LSTM autoencoder (Bi-LSTM-AE) is an autoencoder whose encoder and decoder employ Bi-LSTM architectures. The proposed Bi-LSTM-AE architecture is described in Figure 7 and Table 3. The encoder architecture is constructed using a Bi-LSTM, a fully connected (FC) layer, and a ReLU layer, which consists of No. 2, 3 and 4 in Table 3. The Bi-LSTM layer includes forward LSTM and backward LSTM layers. By contrast, the decoder architecture is a reverse process of the encode that produces reconstructed data, which consists of No. 5, 6 and 7 in Table 3. The scaling layer rescales the output of Bi-LSTM to be ranged same as input data since the output from the Bi-LSTM is ranged from -1 to $+1$ due to tanh activation function. The scale factor is set as 5 in this research, meaning the output from Bi-LSTM is multiplied by 5 in the scaling layer. The hyperparameters are given in Table 4.

As for the FC layers, each column of an input matrix is fed into the fully connected layer. For example, in the case of the first fully connected layer in Figure 7, the calculation can be described as the following equation:

$$X = (x^1 \; x^2 \; x^3 \; \cdots \; x^{timestep}) \tag{17}$$

$$\begin{aligned} h &= \sigma(WX + b) \\ &= \sigma\{(Wx^1 \; Wx^2 \; \cdots \; Wx^{timestep}) + b\} \end{aligned} \tag{18}$$

where X is a 32 by timestep input matrix, W and b are a weight matrix and a bias, whose sizes are 10 by 32 and 10 by 1, σ is the Relu function, and h is 10 by timestep output matrix. Notably, each column vector x^i (i =1, …, timestep) of the input matrix is fed into the fc layer and then mapped with the same W and b, accordingly.

TABLE 3. Proposed Bi-LSTM-AE architecture.

No.	Layer	Output shape	Learnable parameter #
1	Input	2 × 262	0
2	Bi-LSTM	32 × 262	2432
3	FC layer	10 × 262	330
4	ReLU	10 × 262	0
5	FC layer	32 × 262	352
6	ReLu	32 × 262	0
7	Bi-LSTM	2 × 262	272
8	Scaling layer	2 × 262	0
9	Output	2 × 262	0

3) ONE DIMENSIONAL CONVOLUTIONAL NEURAL NETWORK AUTOENCODER

Convolutional neural networks (CNNs) are a specialised kind of neural network for processing grid-like data, such as image data, which can be a 2D grid of pixels [31].

CNNs consist of convolution layers which employ several filters called kernels. The kernels convolve input, and then convolved input is passed to a nonlinear activation function, such as hyperbolic tangent, sigmoid or rectified linear unit (ReLU) function. The convolution and activation operation can be expressed as follows:

$$z = f(x * k) + b \tag{19}$$

where x and k are input data and a l-by-l kernel, f is an activation function, b and z are a bias and a mapped feature matrix, respectively. The $*$ denotes convolution operation.

CNNs have been tremendously successful in practical applications such as image recognition and object detection, where a 2D grid of input data and 2D kernels are used. In that case, the CNNs can be called 2D CNNs. On the other hand, CNNs are also proven promising for 1D input data such as time-series data, where one-dimensional vector kernels are applied for convolution operation instead of 2D kernels, whose CNNs can be categorised as 1D CNN.

TABLE 4. Hyperparameters of Bi-LSTM-AE.

Layer	Parameter name	Parameter
Whole layers	Optimiser	Adam
	Max Epoch	2000
	Learning rate	0.001
	L2 regularisation	0.001
Bi-LSTM	Activation function for the hidden state	tanh
	Activation function for the gates	sigmoid

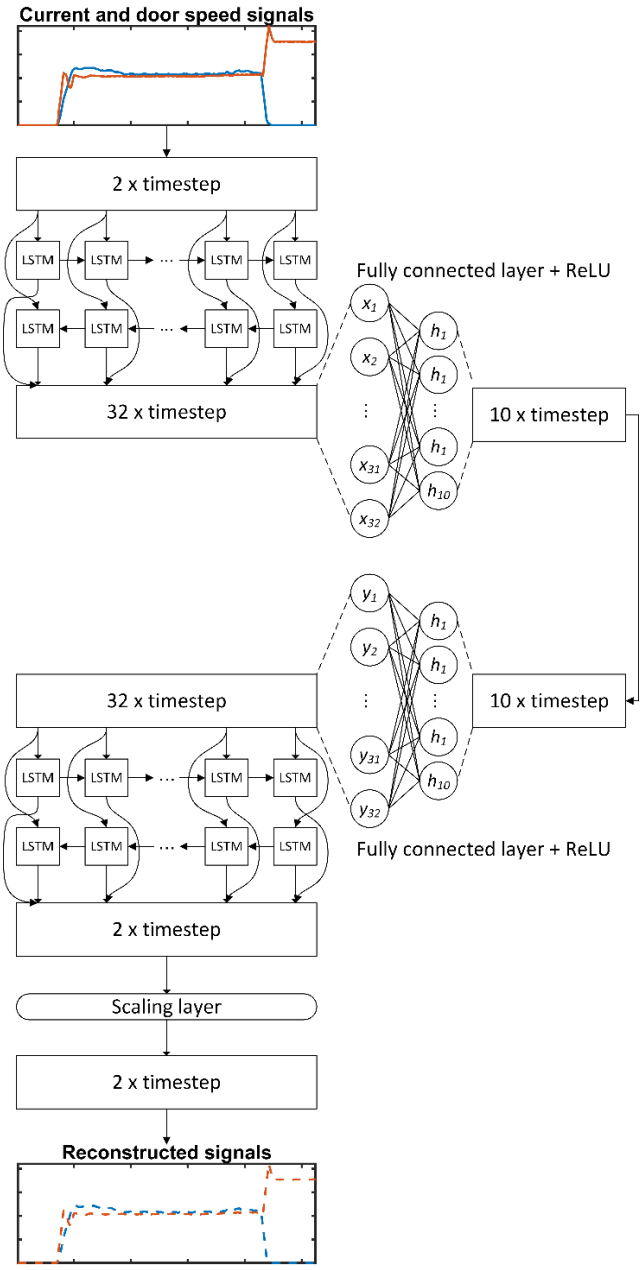

FIGURE 7. Proposed Bi-LSTM AE architecture.

A 1D CNN autoencoder (1D CNN-AE) is an autoencoder whose encoder and decoder employ 1D CNN architectures.

TABLE 5. Constructed 1D CNN-AE architecture.

No.	Layer	Output shape	Learnable parameter #
1	Input	262 × 2 × 1	0
2	1D CNN 1	85 × 1 × 10	190
3	ReLU	85 × 1 × 10	0
4	1D CNN 2	26 × 1 × 10	1620
5	ReLU	26 × 1 × 10	0
6	FC layer	26 × 1 × 6	126
7	ReLU	26 × 1 × 6	0
8	FC layer	6 × 1 × 10	140
9	ReLU	6 × 1 × 10	0
10	1D TCNN 1	26 × 1 × 10	1810
11	ReLU	26 × 1 × 10	0
12	1D TCNN 2	262 × 2 × 1	262
13	Output	262 × 2 × 1	0

The constructed 1D CNN-AE is described in Figure 8 and Table 5. The encoder architecture is constructed using a 1D CNN, an FC layer, and a ReLU layer, which are from No. 2 to No. 7 in Table 5. On the other hand, the decoder architecture is a reverse process of the encode, including 1D transposed convolutional neural networks (1D TCNNs), which are from No. 8 to 12 in Table 5. 1D TCNN enables upsampling the input data. That means the output dimensions are greater than the input dimension in 1D TCNN, initially proposed in [34]. The hyperparameters are given in Table 6.

4) STACKED AUTO ENCODER AND MULTILAYER AUTOENCODER

The stacked autoencoders (SAEs) are autoencoders which have multiple hidden layers. Instead of training whole SAEs, the same as training a neural network (NN) with multiple hidden layers, it is possible to train one shallow AEs, and then stack all hidden layers to be a single SAE, as described in Figure 9. This training is called greedy layer-wise training. In that case, the first autoencoder is trained to reconstruct training input data. Then the whole training dataset is encoded by using the first encoder. The encoded training dataset is utilised for training the second autoencoder to reconstruct the encoded training dataset. Finally, trained shallow AEs are stacked to be a single SAE.

The whole SAEs can be trained in the same manner as NN and greedy layer-wise training. The terminology of SAEs can refer to the models trained by both methods. For the sake of clarity, we define the SAEs trained in the same manner as NN as multilayer AEs (MLAE), whereas defining the SAEs trained by greedy layer-wise as SAEs in this research. The constructed SAE and MLAE architectures are equivalent to each other, as given in Table 7, though the training scheme differs.

D. ONE-CLASS SUPPORT VECTOR MACHINE

The one-class SVM is a variant of the SVM algorithm. Basically, the SVM is a binary linear classifier. The SVM is

TABLE 6. Hyperparameters of 1D CNN-AE.

Layer	Parameter name	Parameter
Whole layers	Optimiser	Adam
	Max Epoch	2000
	Learning rate	0.001
	L2 regularisation	0.0001
1D CNN 1	Filter size	9 x 1
	Number of filters	10
	Stride	3
1D CNN 2	Filter size	8 x 1
	Number of filters	20
	Stride	3
1D TCNN 1	Filter size	9 x 1
	Number of filters	10
	Stride	3
1D TCNN 2	Filter size	13 x 1
	Number of filters	2
	Stride	3

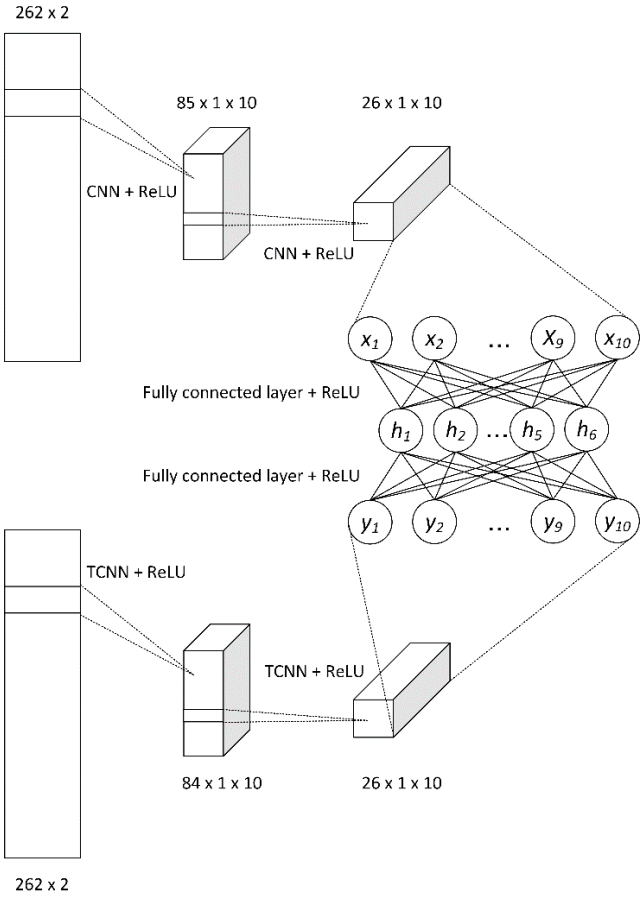

FIGURE 8. 1D CNN-AE architecture

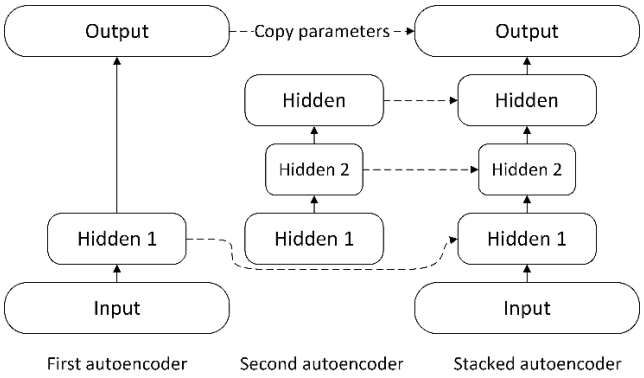

FIGURE 9. Structure of SAEs.

TABLE 7. Constructed SAE and MLAE architecture.

No.	Layer	Output shape	Learnable parameter #
1	Input	1 x 524	0
2	Encoder 1	1 x 50	26,250
3	Encoder 2	1 x 10	510
4	Decoder 1	1 x 50	550
5	Decoder 2	1 x 524	26,724
6	Output	1 x 524	0

also called maximum margin classifiers because the model constructs a decision boundary so as to have a maximum margin from training samples. The SVM is also capable of generating a nonlinear decision boundary by using a kernel function called the kernel trick. In that case, the original data is mapped into a higher dimensional feature space with a kernel function, and then the SVM model sets a linear classifier in this feature space. Then, the original data can be separated by a nonlinear function even though the decision boundary is a linear function in the feature space. The decision boundary in the feature space is also called a hyperplane.

The one-class SVM resembles the SVM for binary classification. The one-class SVM is trained with kernel trick using a dataset which has only one class label. The assumption is that all the training data sample belongs to the positive class and the origin of the feature space belongs to the negative class. Hence, the objective is to maximise separation between the origin and hyperplane in the feature space. However, it is not always practical to generate a decision boundary completely separating all the training data samples and the origin because the exact separation of the training data can lead to poor generalisation [35]. Therefore, the one-class SVM allows some of the training samples to be negative. The optimisation problem can be written as follows:

$$argmin_{\boldsymbol{w},\boldsymbol{\xi},b}\frac{1}{2}\|\boldsymbol{w}\|^2+\frac{1}{\nu m}\sum_{i=1}^{m}\xi_i+b \tag{20}$$

$$s.t.\ \boldsymbol{w}\Phi(\boldsymbol{x_i})+b+\xi_i\geq 0,\quad \xi_i\geq 0\, for\ all\ i=1,\ldots,m \tag{21}$$

where $\boldsymbol{w}$ and b are weight and bias, ξ_i is the slack variable, m is the number of training samples, and Φ is a function mapping $\boldsymbol{x_i}$ into the higher dimensional feature space. The variable $\nu \in (0, 1)$ corresponds to both an upper bound on the fraction of outliers and a lower bound on the fractions

of support vectors. It can be interpreted as the proportion of outlier fraction the one-class SVM allow to be negative. The lagrangian corresponding to the minimisation subject to the constraints is given as follows [36]:

$$L(\boldsymbol{w}, \boldsymbol{\xi}, b) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{1}{\nu m}\sum_{i=1}^{m}\xi_i + b - \sum_{i=1}^{m}\alpha_i\left(\boldsymbol{w}\Phi\left(\boldsymbol{x_i}\right) + b + \xi_i\right) - \sum_{i=1}^{m}\eta_i\xi_i \tag{22}$$

$$\frac{\partial L}{\partial \boldsymbol{w}} = 0 \rightarrow \boldsymbol{w} = \sum_{i=1}^{m}\alpha_i\Phi(\boldsymbol{x_i}) \tag{23}$$

$$\frac{\partial L}{\partial \xi_i} = 0 \rightarrow \alpha_i = \frac{1}{\nu m} - \eta_i,\ \eta_i \in (0, 1) \tag{24}$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{m}\alpha_i = 1 \tag{25}$$

where, $\alpha_i \geqq 0$ and $\eta_i \geqq 0$ are lagrange multipliers. As a result, the lagrangian can be written as follows:

$$argmin_{\alpha}\alpha_i\alpha_j K\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) \tag{26}$$

$$s.t.\ 0 \leqq \alpha_i \leqq 1, \quad \sum_{i=1}^{m}\alpha_i = \nu m \tag{27}$$

$$K\left(\boldsymbol{x_i}, \boldsymbol{x_j}\right) = \Phi\left(\boldsymbol{x_i}\right) \cdot \Phi\left(\boldsymbol{x_j}\right) \tag{28}$$

where K is called kernel function such as gaussian kernel, written as the following equation

$$K(\boldsymbol{x_i}, \boldsymbol{x_j}) = e^{\|\boldsymbol{x_i} - \boldsymbol{x_j}\|/2\sigma} \tag{29}$$

The σ is a parameter meaning variation of Gaussian distribution. The optimisation problem is solved for α_i, η_i, $i = 1, \ldots, m$, followed by computation of $\boldsymbol{w}$ and b to generate hyperplane in the feature space. Then, the nonlinear decision boundary is set in the original space.

For the sake of convenience, we call hyperparameter ν outlier fraction in the paper. In this research, outlier fraction ν is optimised so as to make fault detection accuracy highest in the test dataset. The outlier fraction ν needs to be searched in the vicinity of 0 to make the number of misclassified samples as small as possible. Otherwise, fault detection accuracy could be worse than expected. Thus, we chose the grid search in logarithmic scale, which enables candidates of ν to be exponentially distributed nearby 0, as given in Table 8.

E. VALIDATION

1) VALIDATION WORKFLOW

The proposed validation workflow is described in Figure 10. First, a dataset, which are time-series current and encoder signals of closing operation, are pre-processed to be aligned and eliminate noise by a low pass filter, followed by standardisation, which is the same procedure of health monitoring workflow described in Figure 1. Then, a pre-processed dataset is split into training and test dataset. By using the training dataset, the unsupervised DL model and the one-class SVM are trained, and the reconstruction SSE is calculated.

TABLE 8. The candidates of outlier fraction.

Hyper parameter	candidate
outlier hyperparameter ν	10^{γ} $\gamma \in$ [-2, -1.9, -1.8, -1.7, -1.6, -1.5, -1.4, -1.3, -1.2, -1.1, -1.0, -0.9, -0.8, -0.7, -0.6, -0.5, -0.4, -0.3, -0.2, -0.1,]

In test procedure, test data is modified according to experimental conditions, as described in detail in 4) of Section II.E. Then, the test data is reconstructed with the DL model built with training data, followed by SSE calculation and prediction with the one-class SVM model. Finally, the prediction result is validated with performance metrics.

It is notable that the fault detection accuracy differs according to a decision boundary set by one-class SVM. The decision boundary is determined with an outlier fraction, which is a hyperparameter of one-class SVM. In this research, the candidates of the outlier fraction are applied to set decision boundaries, as given in Table 8. Then, the highest classification accuracy is chosen among the candidates.

2) PERFORMANCE METRICS

A confusion matrix is used to analyse the performance of a fault detection system. A confusion matrix is a two-dimensional table of counts of how often each category is classified or misclassified as each other category. In the case of binary classification for fault detection, the confusion matrix has the following four elements: positive (faulty) sets are either detected or not; similarly, negative (normal) sets are either detected or not. These elements are true positive (TP), false negative (FN), true negative (TN) and false positive (FP), respectively, as given in Table 9. Once populated, this matrix is then used to derive performance metrics commonly used in the industry [37] as given in the following equations:

$$Precision\ (P) = \frac{TP}{TP + FP} \tag{30}$$

$$Recall\ (R) = \frac{TP}{TP + FN} \tag{31}$$

$$F1\ score = \frac{2PR}{P + R} \tag{32}$$

$$False\ positive\ rate\ (FPR) = \frac{FP}{TP + FP} = 1 - P \tag{33}$$

$$True\ positive\ rate\ (TPR) = Recall\ (R) \tag{34}$$

In the context of fault detection, the practitioners have two concerns [24]. The first one is the minimisation of false alarm occurrences. Too many of them will make the fault detection systems unreliable and impractical. The second one is that actual faulty samples should be detected as positive and collected as accurately as possible since undetected actual faulty behaviour might have catastrophic consequences in the

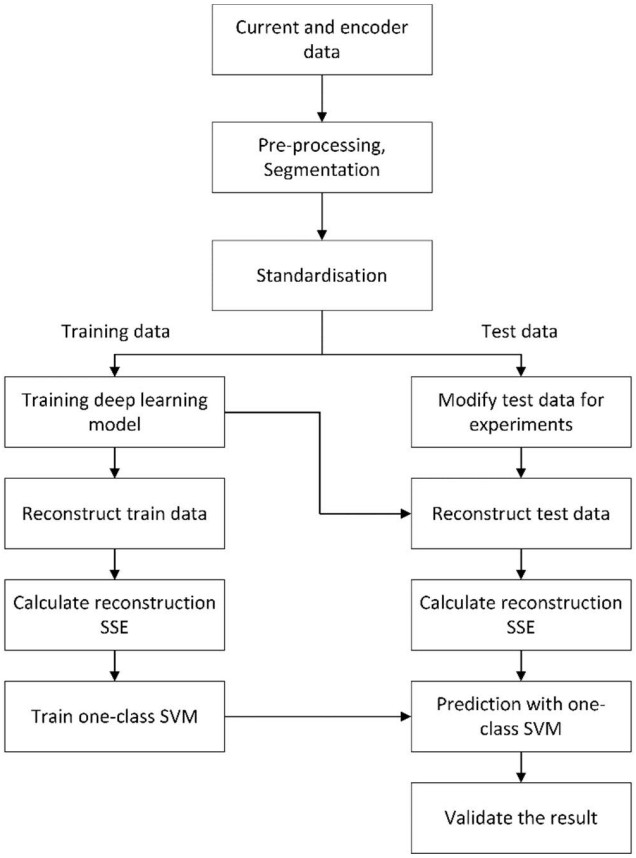

FIGURE 10. Validation workflow.

industry. The two corresponding indicators are the false positive rate (FPR) and recall (R). Precision (P) needs to be as high as possible to minimise FPR. Thus, the two practitioners' concerns can be expressed to maximise P and R.

In general, precision measures how many of the samples predicted as positive are actual positive. Recall, on the other hand, measures how many of the positive samples are captured by the positive predictions. There is a trade-off between optimising precision and optimising recall [38]. A perfect recall can be obtained given that all samples are predicted as positive class, and therefore the precision can be very low, which means there are too many false alarm occurrences. On the contrary, precision can be perfect if a model predicts only a single sample which is the most likely to be positive as positive and the rest as negative. In that case, however, recall can be very low. One way to take precision and recall into account and summarise them is the calculation of the harmonic mean of P and R, which is the F1 score given in equation (32). In this research, the F1 score is applied to evaluate fault detection accuracy, which is ranging from 0 to 1. A high F1 score means high fault detection accuracy and vice versa.

3) SENSITIVITY ANALYSIS

The practice of sensitivity analysis is widespread in technological disciplines. It means analysing how much the output of a model is affected as the model parameters are changed [33]. A receiver operating characteristic (ROC) curve is used in order to validate the model sensitivity against the threshold set by one-class SVM. ROC gives a comprehensive overview of the trade-off between FPR and true positive rate (TPR), as described in Figure 11. The ideal ROC curve has zero FPR and one TPR. A metric called area under the ROC curve (AUC) is the area underneath the entire ROC curve from (0, 0) to (1, 1). The AUC provides a single-number summary of the ROC curve, which becomes one given the ideal ROC curve. In this research, the ROC curve is plotted corresponding to the threshold set by one-class SVM. The threshold is determined by the outlier fraction, which is a hyperparameter of one-class SVM.

TABLE 9. Confusion matrix.

		Predicted Class	
		Faulty	**Normal**
Actual Class	**Faulty**	True Positive (TP)	False Negative (FN)
	Normal	False Positive (FP)	True Negative (TN)

4) EXPERIMENT

In order to verify the fault detection model's adaptability to variational time series data, two experiments are proposed. The first experiment is a time-shifted modification to the test dataset. In the first experiment, we have shifted the current and door speed signals of the test dataset over time dimension by 0.4 second, as shown in Figure 12.

On the other hand, we have expanded the current and door speed signals over time dimension by 0.4 second, in which door operation status is steady-state in the second experiment, as described in Figure 13. The steady-state means door speed and current signals are stable and constant. The assumption for both experiments is that normal and faulty profiles modified according to each experimental condition should also be predicted as normal and fault, respectively.

The test data with no modification is also used to validate fault detection accuracy as a default setting. This test is defined as the default test in this paper.

5) LOCALISATION OF REGIONS OF ANOMALIES

The reconstruction error is utilised to localise anomalous regions in time series data. The general idea is that if some region's reconstruction MSE calculated with the DL model is relatively larger than the MSE of the whole region, those regions are considered anomalous. The steps of localisation of regions of anomalies are the followings.

1. Define the time step window whose size is set to 0.3 second. the initial position of the left side of the windows is assigned to 0 second.
2. Calculate the mse of the window
3. If the mse of the window has over 15 times above the mse of the whole region, the window is labelled an anomalous region. In this research, the term 'the mse threshold' is used to refer to parameter 15.

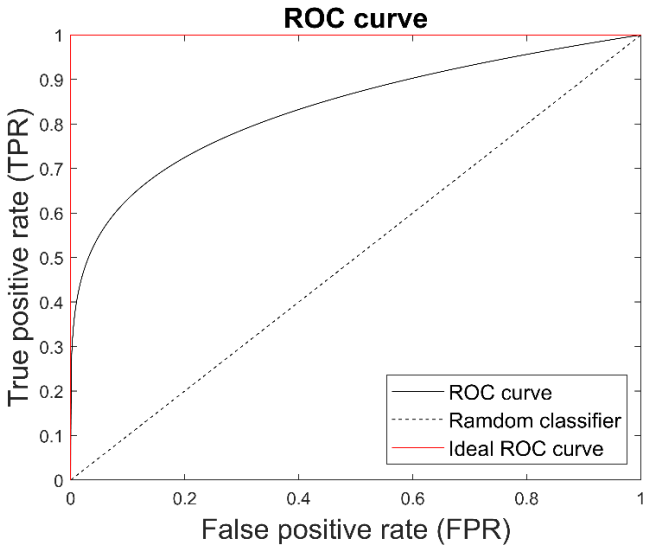

FIGURE 11. ROC curve.

4. Slide the window by 0.02 second
5. Repeat steps 2, 3 and 4 until the right side of the window reaches the end of the time series region
6. Identify windows which are labelled anomalous regions

III. RESULTS AND DISCUSSION

A. PERFORMANCE METRICS AND SENSITIVITY ANALYSIS

The fault detection performance metrics are given in Table 10. As for the default test, Bi-LSTM-AE has the highest F1 score among the models at 0.971, followed by AE and 1D CNN-AE at 0.962 and 0.961, respectively. Meanwhile, the F1 scores for SAE and MLAE are comparatively lower than those for other models at 0.947. However, overall, it is noteworthy that anomalies can be detected at more than 0.940 F1 score by using unsupervised DL models compared in this research. As shown in Figure 15 and Figure 16, normal profiles are well reconstructed with all AE-based models, while anomaly profiles are not well reconstructed. As a result, the reconstruction errors of anomaly data become larger than normal profiles, which is the reason why the unsupervised model enables detecting faulty behaviours by using reconstruction errors. However, the F1 scores of Bi-LSTM-AE, 1D CNN-AE and AE are relatively resembling, so it might be challenging to choose the appropriate model by using only performance metrics. This is because fault detection accuracy could be lower with other operational data, whose distribution is varied due to different operating conditions as training and test dataset.

Furthermore, it is also crucial that the fault detection models need to be less sensitive to the threshold set by one-class SVM, which means the fault detection accuracy needs to be unaffected by the variation of the threshold. As shown in Figure 14 and Table 11, the ROC curve for AE is the closest to the ideal ROC curve at 0.9993 for AUC, followed by Bi-LSTM-AE, 1D CNN-AE at 0.9962 and 0.9916 for AUC,

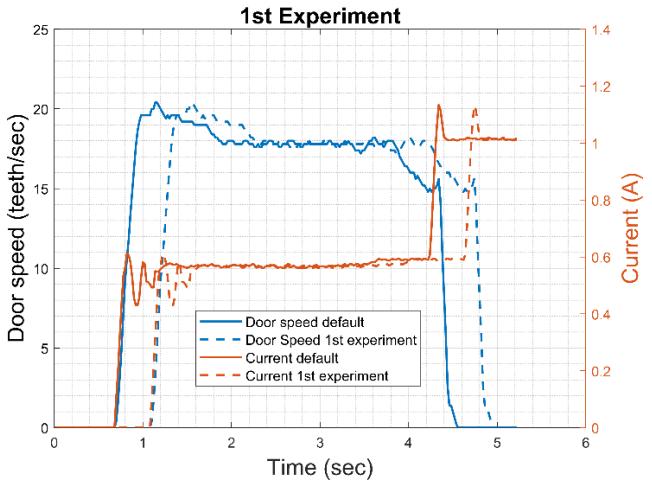

FIGURE 12. Signals for the first experiment.

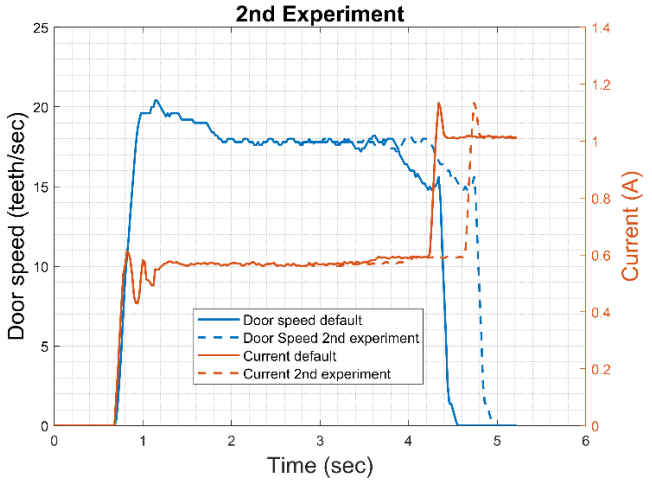

FIGURE 13. Signals for the second experiment.

respectively. By contrast, ROC curves for SAE and MLAE are relatively differentiated from the ideal ROC curve than other models at 0.9651 and 0.9607. The result shows that AE is the least sensitive to the threshold, which means the AE model is the most robust against the variation of the threshold. Hence, AE can be the most appropriate fault detection model in the default test despite a lower F1 score than Bi-LSTM-AE.

However, as for the first and second experiments, only Bi-LSTM-AE has adequate F1 scores at 0.970 and 0.966, correspondingly, while other models have 0.500 precision and 1.00 recall, as given in Table 10. The result for other models except for Bi-LSTM-AE means that entire test data are predicted as anomalies, which is impractical for fault detection purposes. The profiles for the first and second experiments, as shown in Figure 17 and Figure 18, are well reconstructed with Bi-LSTM-AE, whereas the reconstructed data is quite noisy with 1D CNN-AE. In addition, the AE, SAE and MLAE are not adaptable to modified experimental test data since reconstructed profiles are neither shifted nor expanded in time.

TABLE 10. Fault detection performance metrics.

Model	Default			First Experiment			Second Experiment		
	Precision	Recall	F1 score	Precision	Recall	F1 score	Precision	Recall	F1 score
Bi-LSTM-AE	0.943	1.00	0.971	0.943	1.00	0.970	0.934	1.00	0.966
1D CNN-AE	0.942	0.980	0.961	0.500	1.00	0.666	0.500	1.00	0.666
AE	0.926	1.00	0.962	0.500	1.00	0.666	0.500	1.00	0.666
SAE	0.916	0.980	0.947	0.500	1.00	0.666	0.500	1.00	0.666
MLAE	0.908	0.990	0.947	0.500	1.00	0.666	0.500	1.00	0.666

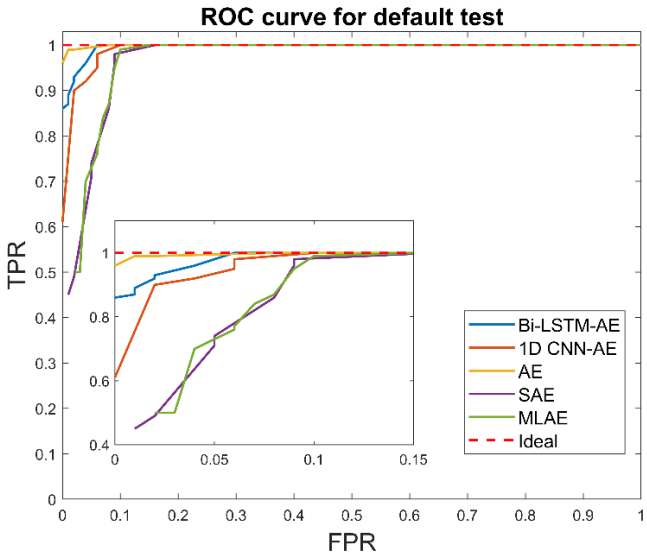

FIGURE 14. ROC curve for default test.

TABLE 11. AUC for default test.

Model	Bi-LSTM -AE	1D CNN -AE	AE	SAE	MLAE
AUC	0.9962	0.9916	0.9993	0.9651	0.9607

The result for Bi-LSTM-AE can be attributed to its parameter sharing and sequential network architecture, taking past and future information into consideration, as explained in 2) of Section II.C. The parameter sharing means that the same weight and bias parameters are employed in LSTM layers. That makes it possible to apply the model to samples of different forms and generalise across them [31]. In the case of separate parameters for each time index, the model cannot be generalised to samples whose characteristics are not observed in the training dataset. Those models need to learn all possible samples they would observe. That is the reason why the AE, SAE and MLAE are unadaptable to the profiles for the first and second experiments. On the other hand, parameter sharing allows the DL model to capture features which can occur at different time indices.

Furthermore, the sequential network architecture enables past and future time-series information to be taken into consideration by using memory cells and gating units. Certainly, 1D CNN is also sharing parameters as kernels. That enables 1D CNN-AE to reconstruct profiles more accurately than AE, SAE and MLAE for the first and second experiments, as shown in Figure 17 and Figure 18. However, 1D CNN does not take past and future information into consideration. The convolution operation capture features of a specific timestep range specified as filter size, while kernels keep sliding over the time dimension, taking no past and future information. The better results of Bi-LSTM-AE for the first and second experiments than 1D CNN-AE could attribute to its sequential network architecture.

B. RECONSTRUCTED ERROR DISTRIBUTION

The reconstruction errors of DL models for the default test, first and second experiments are described in Figure 19. The threshold is set by one class SVM as a decision boundary, above which observations are predicted as anomalies. The SSE distributions of the default setting, the first and second experiments for Bi-LSTM-AE are almost identical since each normal profile of the first and second experiments is well reconstructed, as seen in Figure 17 and Figure 18. The MSSE rates of the first and second experiments for Bi-LSTM-AE are 0.998 and 1.002, respectively, as given in Table 12, respectively. Thus, it is confirmed that the SSEs of the first and second experiments for Bi-LSTM-AE are approximately equivalent to the SSE of the default test, as shown in Figure 19. The result means that the thresholds for the first and second experiments can accurately separate normal and faulty test data. That is the reason why Bi-LSTM-AE has high faulty detection accuracies for the first and second experiments.

On the other hand, the SSE distributions of 1D CNN-AE, AE, SAE and MLAE for the first and second experiments differ considerably from those for the default test, as shown in Figure 19. The MSSE rates of those DL models are far larger than those of the Bi-LSTM-AE, as given in Table 12. It is attributed that the current and door speed profiles are not well reconstructed for the first and second experiments by using 1D CNN-AE, AE, SAE and MLAE, as shown in Figure 17 and Figure 18. Consequently, as described in Figure 19, whole observations of the test dataset are above the threshold and predicted as anomalies. That is the reason why other DL models, except for the Bi-LSTM-AE have 0.500 precision and 1.00 recall.

Therefore, the reconstruction distribution result shows the adaptability and robustness of the Bi-LSTM-AE model to time-series variation, as demonstrated in the first and second experiments.

C. LOCALISATION OF REGIONS OF ANOMALIES

The regions of anomalies can be localised with MSEs generated by unsupervised DL models, as described in Figure 20. The fundamental idea is that given some regions' MSE is

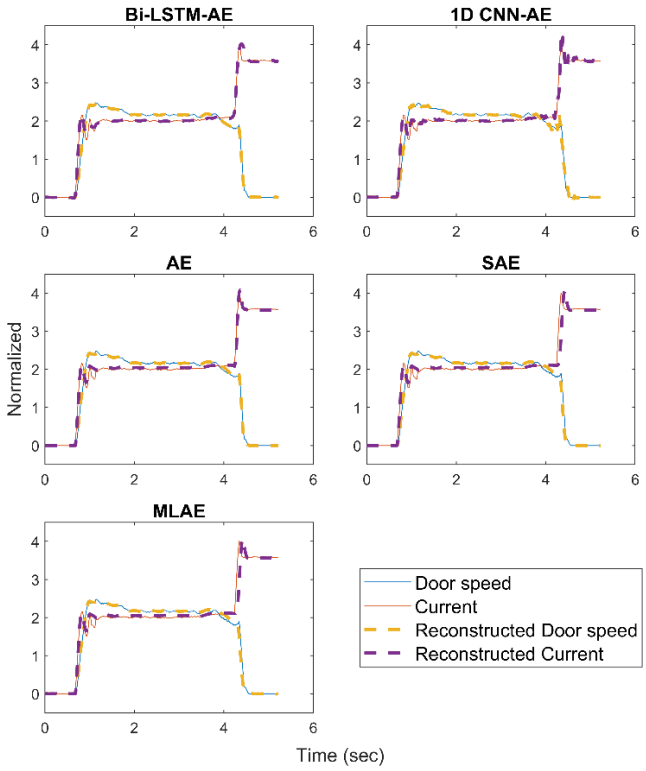

FIGURE 15. Reconstructed normal profiles for default test.

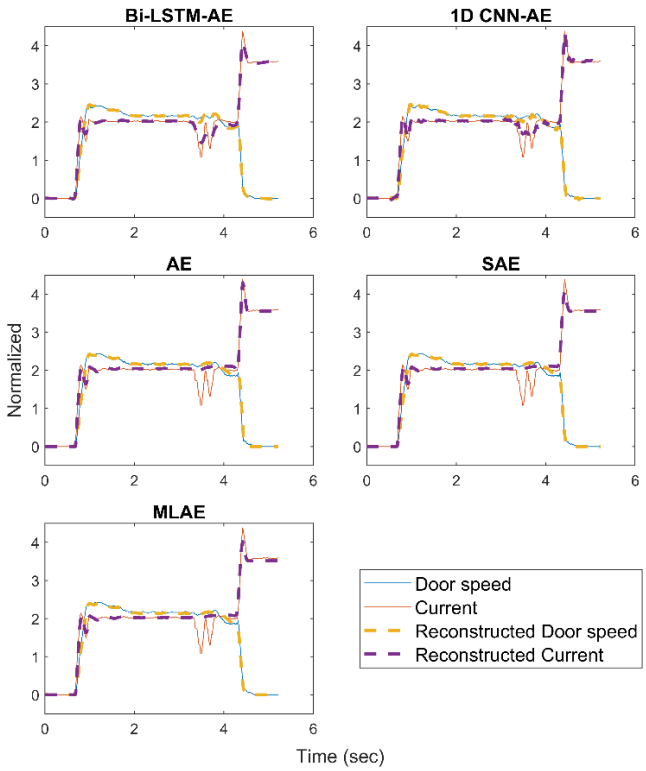

FIGURE 16. Reconstructed anomaly profiles for default test.

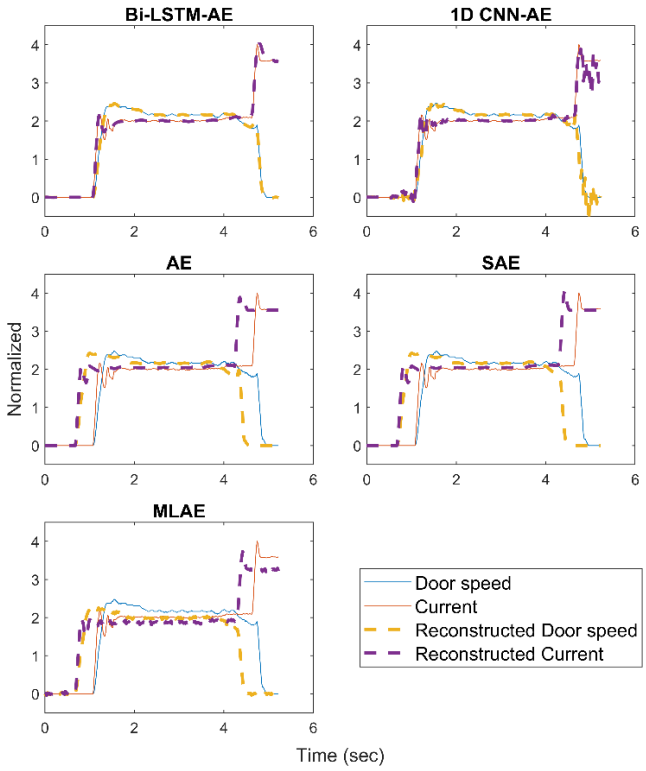

FIGURE 17. Reconstructed normal profiles for first experiment.

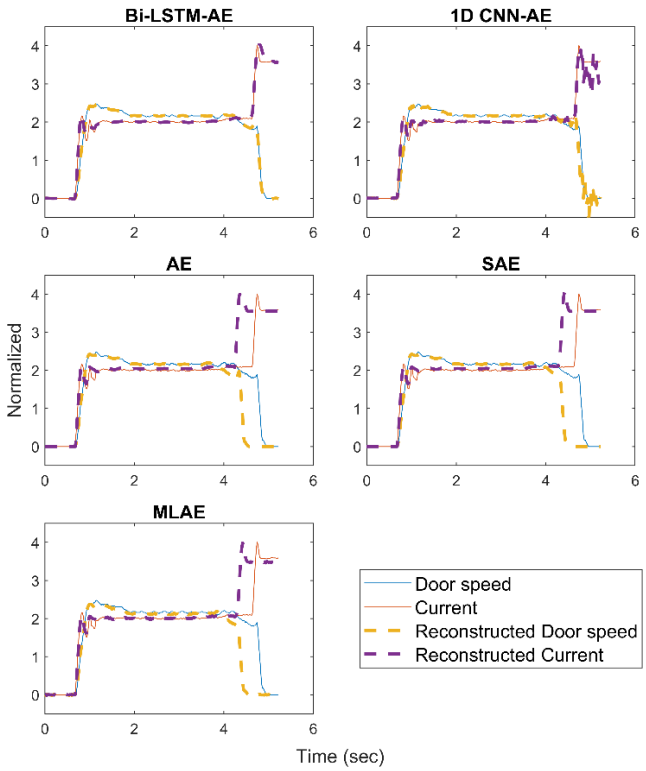

FIGURE 18. Reconstructed normal profiles for second experiment.

considerably larger, those regions are considered anomalous, which is intuitively straightforward. The localised anomalous regions are given in bold red lines in Figure 20.

The result reveals that the fault region can be localised with unsupervised DL-based fault detection approaches, which is also beneficial in order to diagnose the cause

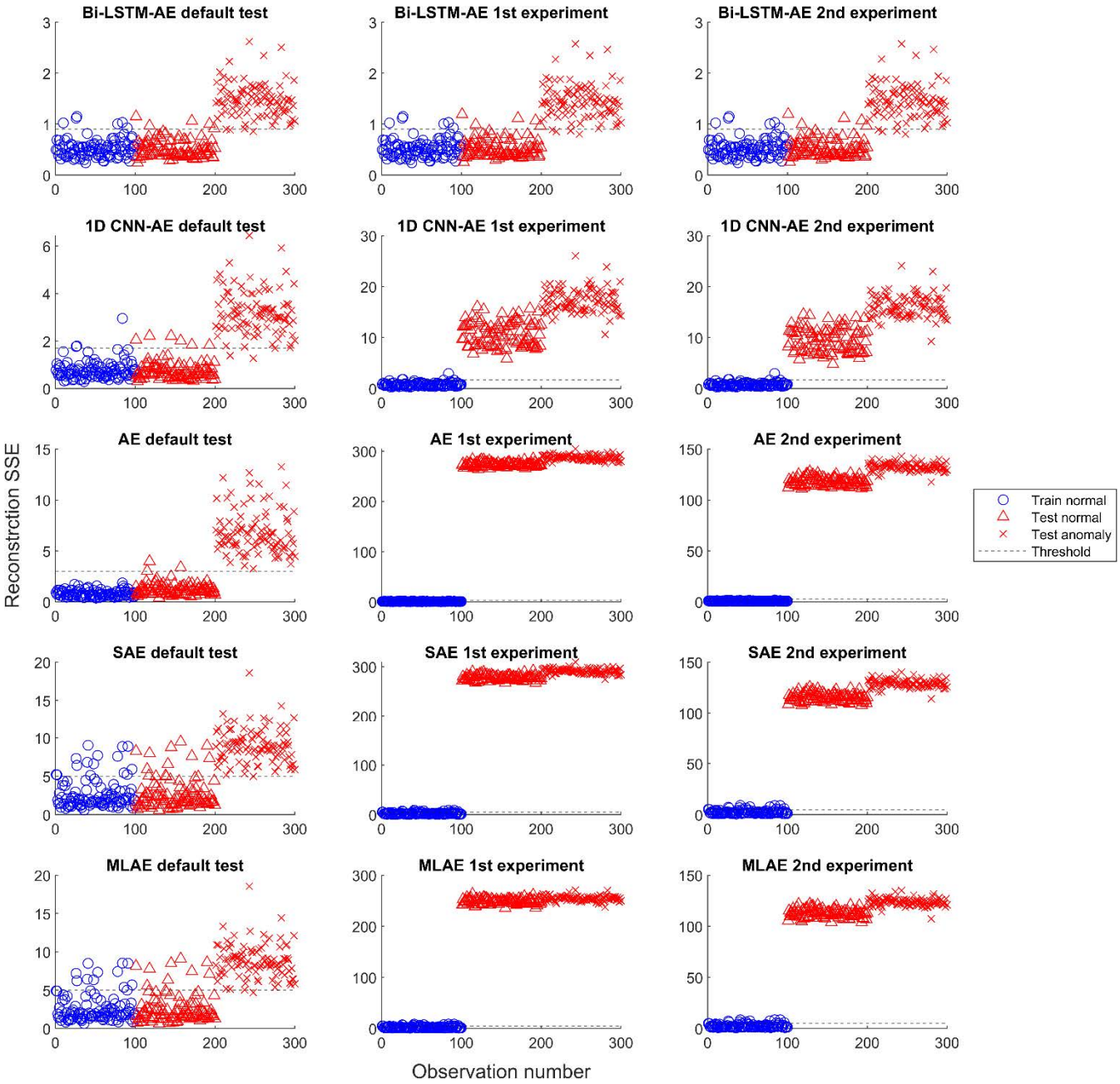

FIGURE 19. The SSE of DL models for default test, first and second experiments.

TABLE 12. MSSE and MSSE rate.

Model	MSSE	MSSE of normal data of test set		MSSE rate	
	Training data	First exp	Second exp	First exp/train data	Second exp/train data
Bi-LSTM-AE	0.529	0.528	0.530	0.998	1.002
1D CNN-AE	0.783	10.685	9.476	13.643	12.100
AE	0.820	273.122	118.125	332.919	143.987
SAE	2.759	277.694	115.549	100.650	41.881
MLAE	2.699	248.644	112.627	92.118	41.726

of failure. Furthermore, it is remarkable that unanticipated faults can also be detected and localised since the proposed method does not require labelled datasets. In the case of supervised learning techniques, on the other hand, fault needs to be detected and localised based on labelled training datasets. Consequently, only anticipated fault can

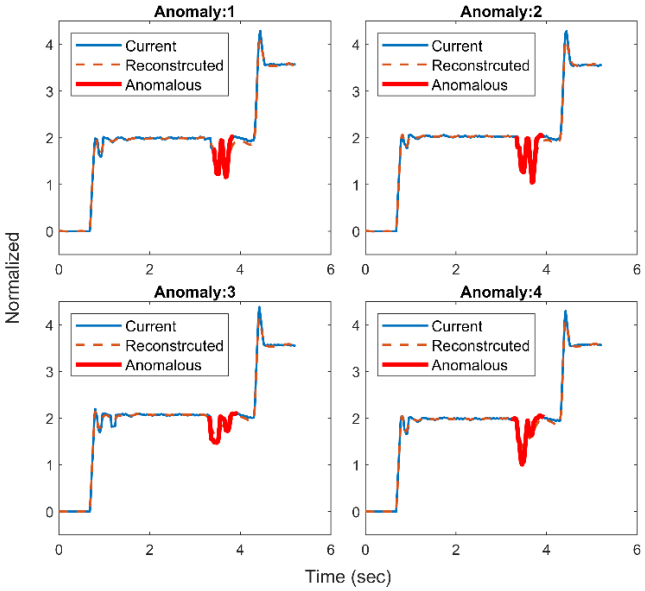

FIGURE 20. Localisation of regions of anomalies for current signals.

TABLE 13. Computational times.

Model	Testing time consumption (ms)
Bi-LSTM-AE	19.6
1D CNN-AE	31.5
AE	70.8
SAE	194.5
MLAE	216.3

be detected and localised with supervised learning-based approaches.

Thus, the unsupervised learning-based approach is beneficial not only for tackling the issue relating to the lack of available labelled datasets but also for detecting and localising unanticipated faults.

Certainly, it is important to bear in mind that there is a potential bias to test dataset because the MSE threshold is optimised empirically with test dataset. However, the possible bias could be avoidable if some amount of faulty data is available as a validation dataset to optimise the MSE threshold.

D. REAL-TIME PERFORMANCE

It is crucial for real-time applications to guarantee response times within a specified time. We have validated computational efficiency for each fault detection model, as given in Table 13. The training process could be done off-line and hence will not affect real-time fault detection performance. The result of this study shows that testing can be conducted in less than 250 milliseconds for five DL models. The term 'testing' is used to refer to the fault detection process for single door operation. This means that fault detection results can be obtained prior to another door operation since intervals between door operations are typically more than minutes. Thus, the fault behaviour can be detected in real-time once one door operation is terminated, which allows us to maintain machinery before breakdown.

Indeed, the proposed method entails a limitation if the fault detection needs to be executed on the edge device implemented on the train. In that case, the fault detection DL model could not be embedded on the device due to its computational resource constraints. The hardware device, whose specifications are Intel(R) Core (TM) i7-10750H CPU @ 2.60 GHz processor and 8 gigabytes RAM, is employed for this validation. However, it might be one of the possible options that fault detection procedure could be executed with cloud computing resources since the amount of sensor reading data is considerably low at approximately 97.0 megabytes as an assumption per day for a single train. Therefore, the railway data can be transferred to the cloud servers by a mobile network. Thus, the fault detection model could be executed on cloud servers with enough computing resources. Besides, there is a possibility to have edge devices, which have rich computing resources depending on the industrial situations.

Certainly, it might also be arguable that the proposed method is not a real-time application because the fault detection models require an entire single door operation for decision making. But we insist it is possible to guarantee that fault detection can be conducted between the last door operation and the next one. This satisfaction is enough for a practical real-time fault detection application because the door closing operation time is less than 6.0 sec, so there is no need to detect faulty behaviour while the door is being operated.

IV. CONCLUSION

This paper aims to propose a successful real-time fault detection framework based on unsupervised DL using only healthy normal data. The approach is based on AEs and one-class SVM as a classifier. As a case study, large real-world datasets acquired from railway door systems have been employed. The datasets include motor current and encoder signals with opening and closing operation status.

First, the time-series current and encoder signals are preprocessed to be aligned, noises are reduced by using a low pass filter technique, followed by segmentation and standardisation. Secondly, the AEs and one-class SVM are trained with healthy normal data and comprehensively validated based on performance metrics and sensitivity analysis. Lastly, two experiments have been carried out to verify the model's adaptability and robustness to variational time-series data.

The result shows a typical AE has the highest AUC at 0.9993, which means the AE is the least sensitive to the threshold set by one-class SVM with the default test dataset. However, as for the first and second experiments, F1 scores for Bi-LSTM-AE are considerably higher than other AE-based models at 0.970 and 0.966, respectively. It means that only Bi-LSTM-AE is adaptable and robust to variational time series data among the DL models due to its parameter sharing and sequential network architecture, taking

past and future information into consideration. The experimental results also enable obtaining the understandability and explainability of the DL models. Furthermore, the regions of anomalies are localised with Bi-LSTM-AE, which is also beneficial in diagnosing the cause of failure.

The proposed method offers remarkable advantages in terms of practical fault detection applications available in the industry. Firstly, fault detection models can be built by using only healthy normal data acquired in the industry, where labelled datasets are always insufficient. Secondly, the proposed method enables fault detection models to be created without handcrafted features utilising a deep network architecture. Thirdly, the proposed Bi-LSTM-AE is adaptable and robust to variational time-series data, which is crucial for the purpose of practical PHM applications. Fourthly, both anticipated and unanticipated faults can be detected and localised due to unsupervised approaches. Fifthly, the fault detection model built offline can be improved by additional operational data that enable fault detection to be more accurate.

For future work, a priori hyperparameter optimisation is still an open question. In this research, hyperparameters of DL models and a one-class classifier are set empirically based on fault detection accuracy. However, it is significant for practitioners to optimise hyperparameters prior to fault occurrence. The augmentation of faulty data could be one of the methods to address the issue. As for unsupervised fault detection, the augmented faulty data is used to validate the model accuracy. Compared to training the supervised model, the amount of faulty data necessary for unsupervised models could be considerably smaller, which is one of the advantages of unsupervised fault detection. In addition, it is required to obtain more understandability and explainability of unsupervised DL models in order to build a reliable fault detection model. The understandability and explainability of supervised DL have been investigated in recent years. The faulty representation could be derived from the supervised models, which have learned faulty data characteristics during the training process. However, unsupervised models have no information from faulty data by nature. Hence understandability and explainability of unsupervised DL models entail more difficulty than supervised DL models from a fault detection perspective. The black-box experiments demonstrated in this paper are practical because they can explain the DL models without being aware of the exact mechanism of network architectures. Therefore, understandability and explainability of unsupervised DL based on black box experiments could be the next research direction in the future.

ACKNOWLEDGMENT

The authors wish to thank Unipart Rail and Instrumentel for their support.

REFERENCES

[1] K. L. Tsui, N. Chen, Q. Zhou, Y. Hai, and W. Wang, "Prognostics and health management: A review on data driven approaches," *Math. Problems Eng.*, vol. 2015, pp. 1–17, May 2015, doi: 10.1155/2015/793161.

[2] Y. Peng, M. Dong, and M. J. Zuo, "Current status of machine prognostics in condition-based maintenance: A review," *Int. J. Adv. Manuf. Technol.*, vol. 50, no. 1, pp. 297–313, Jan. 2010, doi: 10.1007/S00170-009-2482-0.

[3] Health and Safety Executive. (Mar. 2003). *Train Derailment at Potters Bar—10 May 2002—A Progress Report by the HSE Investigation Board May 2003*. [Online]. Available: https://webarchive.nationalarchives.gov.uk/ukgwa/20081108234009/http%3A//www.rail-reg.gov.uk/upload/pdf/incident-pottersbar-may03progrep.pdf

[4] G. Vachtsevanos, F. Lewis, M. Roemer, A. Hess, and B. Wu, "Fault Prognosis," in *Intelligent Fault Diagnosis and Prognosis for Engineering Systems*. Hoboken, NJ, USA: Wiley, Mar. 2007, pp. 284–354, doi: 10.1002/9780470117842.CH6.

[5] O. F. Eker, F. Camci, and I. K. Jennions, "A new hybrid prognostic methodology," *Int. J. Prognostics Health Manage.*, vol. 10, no. 2, 2019, Accessed: Aug. 27, 2021. [Online]. Available: http://www.phmsociety.org/node/2559, doi: 10.36001/ijphm.2019.v10i2.2727.

[6] C. Roberts, H. P. B. Dassanayake, N. Lehrasab, and C. J. Goodman, "Distributed quantitative and qualitative fault diagnosis: Railway junction case study," *Control Eng. Pract.*, vol. 10, no. 4, pp. 419–429, Apr. 2002, doi: 10.1016/S0967-0661(01)00159-9.

[7] Y. Yang, P. Fu, and Y. He, "Bearing fault automatic classification based on deep learning," *IEEE Access*, vol. 6, pp. 71540–71554, 2018, doi: 10.1109/ACCESS.2018.2880990.

[8] L. Luo, L. Xie, and H. Su, "Deep learning with tensor factorization layers for sequential fault diagnosis and industrial process monitoring," *IEEE Access*, vol. 8, pp. 105494–105506, 2020, doi: 10.1109/ACCESS.2020.3000004.

[9] Z. Q. Chen, C. Li, and R. V. Sanchez, "Gearbox fault identification and classification with convolutional neural networks," *Shock Vib.*, vol. 2015, Oct. 2015, Art. no. 390134, doi: 10.1155/2015/390134.

[10] P. Zhang, G. Zhang, W. Dong, X. Sun, and X. Ji, "Fault diagnosis of high-speed railway turnout based on convolutional neural network," in *Proc. 24th Int. Conf. Autom. Comput. (ICAC)*, Sep. 2018, pp. 1–6, doi: 10.23919/IConAC.2018.8749078.

[11] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-time motor fault detection by 1-D convolutional neural networks," *IEEE Trans. Ind. Electron.*, vol. 63, no. 11, pp. 7067–7075, Nov. 2016, doi: 10.1109/TIE.2016.2582729.

[12] F. Aziz, A. U. Haq, S. Ahmad, Y. Mahmoud, M. Jalal, and U. Ali, "A novel convolutional neural network-based approach for fault classification in photovoltaic arrays," *IEEE Access*, vol. 8, pp. 41889–41904, 2020, doi: 10.1109/ACCESS.2020.2977116.

[13] S. Kim and J.-H. Choi, "Convolutional neural network for gear fault diagnosis based on signal segmentation approach," *Struct. Health Monitor.*, vol. 18, no. 6, pp. 1401–1415, 2019, doi: 10.1177/1475921718805683.

[14] S. Ham, S.-Y. Han, S. Kim, H. J. Park, K.-J. Park, and J.-H. Choi, "A comparative study of fault diagnosis for train door system: Traditional versus deep learning approaches," *Sensors*, vol. 19, no. 23, p. 5160, Nov. 2019, doi: 10.3390/s19235160.

[15] J. Wang, R. Zhao, D. Wang, R. Yan, K. Mao, and F. Shen, "Machine health monitoring using local feature-based gated recurrent unit networks," *IEEE Trans. Ind. Electron.*, vol. 65, no. 2, pp. 1539–1548, Jul. 2017, doi: 10.1109/TIE.2017.2733438.

[16] R. Zhao, R. Yan, J. Wang, and K. Mao, "Learning to monitor machine health with convolutional Bi-directional LSTM networks," *Sensors*, vol. 17, no. 2, p. 273, Jan. 2017, doi: 10.3390/S17020273.

[17] Q. Liu, T. Liang, Z. Huang, and V. Dinavahi, "Real-time FPGA-based hardware neural network for fault detection and isolation in more electric aircraft," *IEEE Access*, vol. 7, pp. 159831–159841, 2019, doi: 10.1109/ACCESS.2019.2950918.

[18] C. Lu, Z. Y. Wang, W. L. Qin, and J. Ma, "Fault diagnosis of rotary machinery components using a stacked denoising autoencoder-based health state identification," *Signal Process.*, vol. 130, pp. 377–388, Jan. 2017, doi: 10.1016/J.SIGPRO.2016.07.028.

[19] G. Liu, H. Bao, and B. Han, "A stacked autoencoder-based deep neural network for achieving gearbox fault diagnosis," *Math. Problems Eng.*, vol. 2018, Jul. 2018, Art. no. 5105709, doi: 10.1155/2018/5105709.

[20] W. Yan and L. Yu, "On accurate and reliable anomaly detection for gas turbine combustors: A deep learning approach," in *Proc. Annu. Conf. Prognostics Health Manage. Soc.*, Aug. 2019, pp. 440–447.

[21] Z. Chen and W. Li, "Multisensor feature fusion for bearing fault diagnosis using sparse autoencoder and deep belief network," *IEEE Trans. Instrum. Meas.*, vol. 66, no. 7, pp. 1693–1702, Jul. 2017, doi: 10.1109/TIM.2017.2669947.

[22] A. S. Yoon, T. Lee, Y. Lim, D. Jung, P. Kang, D. Kim, K. Park, and Y. Choi, "Semi-supervised learning with deep generative models for asset failure prediction," 2017, *arXiv:1709.00845*.
[23] J. Tao, Y. Liu, and D. Yang, "Bearing fault diagnosis based on deep belief network and multisensor information fusion," *Shock Vib.*, vol. 2016, Sep. 2016, Art. no. 9306205, doi: 10.1155/2016/9306205.
[24] G. Michau, Y. Hu, T. Palmé, and O. Fink, "Feature learning for fault detection in high-dimensional condition monitoring signals," *Proc. Inst. Mech. Eng. O, J. Risk Rel.*, vol. 234, no. 1, pp. 104–115, Aug. 2019, doi: 10.1177/1748006X19868335.
[25] S. Plakias and Y. S. Boutalis, "Exploiting the generative adversarial framework for one-class multi-dimensional fault detection," *Neurocomputing*, vol. 332, pp. 396–405, Mar. 2019, doi: 10.1016/J.NEUCOM.2018.12.041.
[26] H. J. Park, S. Kim, S. Y. Han, S. Ham, K. J. Park, and J. H. Choi, "Machine health assessment based on an anomaly indicator using a generative adversarial network," *Int. J. Precis. Eng. Manuf.*, vol. 22, no. 6, pp. 1113–1124, Jun. 2021, doi: 10.1007/S12541-021-00513-1/FIGURES/10.
[27] A. K. M. Nor, S. R. Pedapati, M. Muhammad, and V. Leiva, "Overview of explainable artificial intelligence for prognostic and health management of industrial assets based on preferred reporting items for systematic reviews and meta-analyses," *Sensors*, vol. 21, no. 23, p. 8020, Dec. 2021, doi: 10.3390/S21238020.
[28] S. Khan and T. Yairi, "A review on the application of deep learning in system health management," *Mech. Syst. Signal Process.*, vol. 107, pp. 241–265, Jul. 2018, doi: 10.1016/j.ymssp.2017.11.024.
[29] B. Namoamo, "Fault diagnosis in time series data with application to railway assets," Ph.D. dissertation, School Aerosp., Transp. Manuf., Eng. Manage. Manuf. Syst., Cranfield Univ., Cranfield, U.K., 2017.
[30] C. Ruiz-Carcel and A. Starr, "Data-based detection and diagnosis of faults in linear actuators," *IEEE Trans. Instrum. Meas.*, vol. 67, no. 9, pp. 2035–2047, Sep. 2018, doi: 10.1109/TIM.2018.2814067.
[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. Accessed: Jun. 21, 2022. [Online]. Available: https://www.deeplearningbook.org/
[32] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.
[33] S. J. Russell, *Artificial Intelligence: A Modern Approach*. London, U.K.: Pearson, 2021.
[34] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, May 2010, pp. 2528–2535, doi: 10.1109/CVPR.2010.5539957.
[35] M. Jordan, J. Kleinberg, and B. Schölkopf, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
[36] S. Mahadevan and S. L. Shah, "Fault detection and diagnosis in process data using one-class support vector machines," *J. Process Control*, vol. 19, no. 10, pp. 1627–1639, Dec. 2009, doi: 10.1016/J.JPROCONT.2009.07.011.
[37] I. K. Jennions, *Integrated Vehicle Health Management: The Technology*. Warrendale, PA, USA: Society of Automotive Engineers, 2013.
[38] S. G. Andreas and C. Müller, *Model Evaluation and Improvement | Introduction to Machine Learning With Python*. Sebastopol, CA, USA: O'Reilly Media, 2016.

MINORU SHIMIZU received the bachelor's degree in physics and the master's degree in chemistry and material science from the Tokyo Institute of Technology, Japan. He is currently pursuing the M.Sc. degree (by research) in transport systems at the Integrated Vehicle Health Management (IVHM) Centre, Cranfield University. Before he joined Cranfield University, he worked for several years as a Research Engineer in a construction and mining machinery manufacturer. He was responsible for the research and development of the IoT monitoring systems for construction and mining machinery to improve customer productivity. His research interests include fault detection, diagnosis, and prognosis based on data-driven approaches, such as machine learning and deep learning for time-series data.

SURESH PERINPANAYAGAM (Member, IEEE) received the bachelor's and master's degrees in aeronautical engineering from Imperial College London, and the Ph.D. degree in mechanical engineering from the Rolls-Royce University Technology Centre, Imperial College, under the auspices of the European FP7 Program. He worked on over 25 industry-led projects funded by industry partners, Aerospace Technology Institute (ATI), Engineering and Physical Sciences Research Council (EPSRC), U.K. Research and Innovation, and the European Commission. He worked at Ford Motor Company Development Centres, Dunton, U.K., and Merknick, Germany, and implemented an integrated data-centric engineering platform for new vehicle design and development. He is currently a Senior Lecturer in intelligent systems and a Chartered Engineer at the Integrated Vehicle Health Management (IVHM) Centre/DARTeC. His expertise are in data-centric engineering, digital twins and high-fidelity simulation, and artificial intelligence (AI) for intelligent systems.

BERNADIN NAMOANO received the M.Sc. degree in computer science from Polytechnique University, France, in 2013, and the Ph.D. degree in the field of condition monitoring applied to railway assets from Cranfield University, which was funded by the EPSRC and Unipartrail/Instrumentel. Before joining Cranfield University, in 2017, he worked as a Software Engineer in Paris, focusing on architectural design for big data, big time series data analysis, and software lifecycle implementation and maintenance. He is also working on the development of novel techniques for detecting, diagnosing and predicting faults using big temporal data. He awarded the EPSRC Doctoral Fellowship Prize.

https://dspace.org/

A real-time fault detection framework based on unsupervised deep learning for prognostics and health management of railway assets

Shimizu, Minoru; Perinpanayagam, Suresh; Namoano, Bernadin

2022-09-08
IEEE
https://doi.org/10.1109/ACCESS.2022.3205352
Shimizu M, Perinpanayagam S, Namoano B. (2022) A real-time fault detection framework based on unsupervised deep learning for prognostics and health management of railway assets. IEEE Access, Volume 10, pp. 96442-96458

Downloaded from a DSpace repository. https://dspace.org/