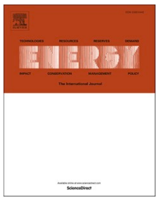
Energy 282 (2023) 128180

Contents lists available at ScienceDirect

Energy

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



Fault data seasonal imbalance and insufficiency impacts on data-driven heating, ventilation and air-conditioning fault detection and diagnosis performances for energy-efficient building operations

Fangliang Zhong [*], John Kaiser Calautit, Yupeng Wu

Department of Architecture and Built Environment, University of Nottingham, United Kingdom

ARTICLE INFO

Handling Editor: Neven Duic

Keywords:
HVAC
Fault detection and diagnosis (FDD)
Fault impact
Deep learning
Building performance simulation
Climate conditions

ABSTRACT

The heating, ventilation and air-conditioning fault impacts vary with different seasonal climatic conditions, but the fault data may not be available under some seasons in real buildings due to the frequency and span of fault occurrences. This study evaluates the fault detection and diagnosis (FDD) performance differences of the proposed convolutional and recurrent neural networks under limited seasonal fault data scenarios and an ideal scenario covering climatic conditions from multiple seasons. The fault and normal data were gathered from fault simulations using a verified prototype building EnergyPlus model and two real fault datasets. Four different data experiments based on the simulated dataset were implemented to assess FDD performance differences, and two sets of further experiments based on each real fault dataset were conducted to verify the findings from previous experiments. The results show that the FDD architectures, trained on sufficient fault data under a certain season (s), indicate poor generalization ability to identify faults under unseen seasons. Moreover, the coverage of fault data under different seasons is more crucial in enhancing FDD performances than the amount of fault data under each season. These findings will help researchers consider this practical issue when evaluating new or existing data-driven FDD methods.

1. Introduction

The building sector was responsible for 31% of global final energy usage in 2020 [1]. As one of the main components of modern buildings, heating, ventilation and air-conditioning (HVAC) systems can account for 44% and 51% of building energy consumption in commercial [2] and residential buildings [3], respectively, in order to maintain indoor thermal comfort and air quality for occupants. However, the occurrence of faults, such as sensor faults and actuator faults, in HVAC systems leads to not only more than 20% of HVAC system energy demand increase [4], but also indoor environmental conditions degradations [5]. Therefore, timely and accurate detection, diagnosis and maintenance of faults are essential to keep the energy-efficient operation of HVAC systems, and comfortable and healthy indoor environments. The implementation of fault detection and diagnosis (FDD) methods are estimated to be able to save the HVAC energy consumption by 10–40% [6].

1.1. Fault detection and diagnosis background

With the recent advancement in machine learning algorithms and computing power, and the increasing data availability through building management systems (BMS), data-driven FDD methods for HVAC systems have been widely investigated. Compared with the conventional FDD methods [7], i.e., rule-based and model-based methods, data-driven methods offer more flexibility and easier implementation in practice [8]. Rule-based methods [9] rely on diagnostic rules to indicate faulty system operations, and the derivation of the rules requires deep expertise in system knowledge. Model-based methods [10] identify the faults by the deviations between the actual measurements of system conditions and energy consumption, and the predicted results from the physical or statistical models of HVAC systems. Hence, the applications of these two types of methods are specific to a certain system and challenging to implement. In contrast, data-driven methods [11] can automatically capture the underlying behaviours of faults through relevant features from HVAC system operational data, which addresses the limitations of conventional methods.

* Corresponding author.
E-mail addresses: Fangliang.Zhong@outlook.com (F. Zhong), john.calautit1@nottingham.ac.uk (J.K. Calautit), Yupeng.wu@nottingham.ac.uk (Y. Wu).

https://doi.org/10.1016/j.energy.2023.128180
Received 7 January 2023; Received in revised form 14 June 2023; Accepted 18 June 2023
Available online 28 June 2023

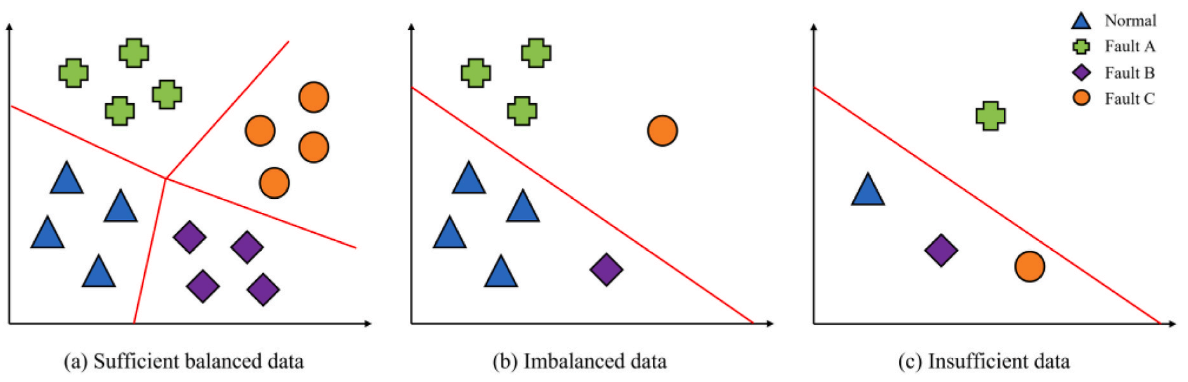


Fig. 1. An illustration of FDD classification under (a) sufficiently balanced data; (b) imbalanced data; (c) insufficient data (the red line(s) in each case represent the possible separation boundaries for different data classes predicted by the FDD model). (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

Among the various types of data-driven methods, supervised classification machine learning algorithms [12] are feasible to solve HVAC system fault detection and diagnosis problems considering the availability of labelled fault data. A data label is the ground truth of the HVAC system operating condition at a certain time step. Depending on the number of label classes defined in the dataset, the FDD problem for HVAC systems can be regarded as either a binary anomaly detection [13] problem or a multi-class classification problem [14]. A variety of supervised classification machine learning algorithms have been applied to identifying HVAC system faults. Sun et al. [15] integrated a support vector machine (SVM) with wavelet de-noising technique, and max-relevance and min-redundancy algorithm, in order to identify refrigerant charge faults for multi-split variable refrigeration flow (VRF) systems. Apart from SVM, Liu et al. [16] applied principal component analysis (PCA)-based exponentially-weighted moving average approach to diagnose refrigerant charge faults for VRF systems. Bode et al. [17] employed a series of machine learning algorithms, including logistic regression, k-nearest-neighbor, random forest, SVM, naive Bayes, etc., to detect heat pump faults, and test the transferability of FDD models from experimental fault data to real fault data. Du, Jin and Yang [18] developed an FDD model for variable air volume (VAV) systems using a wavelet neural network. Liu et al. [19] proposed a diagnostic Bayesian network-based FDD method for a solar aided heat pump system under insufficient data and domain knowledge. Allen, Rubaai and Chawla [20] developed an operation monitoring system for a VAV terminal based on fuzzy logic and artificial neural network.

1.2. Deep learning fault detection and diagnosis studies

Recently, deep neural networks, that consist of multiple learning layers to capture the deeper-levels of information from the input data [21], have gained increasing attention in the area of HVAC FDD. Due to the time series nature of HVAC operational data, convolutional neural network (CNN) [22] and recurrent neural network (RNN) [23] have performed well in supervised fault classification problems for HVAC systems. CNN is powerful in extracting meaningful features from time series data. Cheng et al. [24] proposed multiscale CNNs to extract discriminative features from multiscale raw sensor data of air handling units (AHU), and the proposed architecture demonstrated outstanding F1 scores [25]. Eom et al. [26] applied CNNs to the refrigerant charge amount prediction under both heating and cooling operation modes. Fan et al. [27] applied a two-dimensional (2D) CNN transfer learning framework in order to address the integration issue of HVAC fault data from different systems. Similarly, Li et al. [28] employed deep transfer learning strategies based on a benchmark CNN architecture to enhance FDD performances for different types and working conditions of chillers.

RNN is robust in learning temporal patterns in time series data as it not only takes into account the information at the current time step, but also stores the information from the previous time step. Zhang et al. [29] used long short term memory (LSTM), a type of RNN architecture, to identify the AHU faults according to the sparse slow features. Taheri et al. [30] compared the FDD performances of seven deep RNN architectures and found that the architecture of two LSTM layers with a deep transition output performed best among the seven architectures. In addition, Shahnazari et al. [31] adopted RNN to identify the faults using the FDD estimation filters based on predictive models built on the normal operational data for a VAV system.

Taking advantage of both CNN and RNN, the architectures combining CNNs and RNNs have shown encouraging results in supervised fault classification problems. Wang et al. [32] indicated that the combined one-dimensional (1D) CNN and gated recurrent unit (GRU), a type of RNN, outperformed other deep neural networks in identifying the faults of a chiller system. Qin et al. [33] applied the CNN-RNN combination to detect the high-speed train bogie faults, and the framework showed the highest f1 score of 0.97 compared with other CNNs or RNNs only methods. Canizo et al. [34] established a novel multi-head CNN-RNN framework for an elevator system, and the proposed framework could keep high detection performances under imbalanced normal and fault data distributions. However, there is still a lack of studies examining the application of this CNN-RNN combination in the HVAC systems FDD field, considering the number of input features from HVAC time-series data is significantly higher than those from the above-mentioned cases. Hence, this study employed the CNN-RNN combination to detect and diagnose the faults in HVAC systems.

Although the proposed FDD methods based on these deep learning algorithms are robust, the performances of these methods are inclined to degrade when applied to HVAC FDD problems in real buildings. Two main obstacles to maintaining good performances, i.e., the imbalanced distributions of normal and fault data, and the insufficient amount of available fault data (Fig. 1), have already been mentioned in previous literature [35]. On the one hand, the normal operational time of an HVAC system is inherently longer than the faulty operational time, leading to imbalanced data sample distributions between normal and fault conditions. Hence, the bias towards the majority group, i.e., normal class, is imposed on FDD models trained on the imbalanced data (Fig. 1b). On the other hand, the amount of fault data is limited in most buildings and particularly lacking in newly built buildings where the fault occurrence rate is relatively low or retrofit buildings that are newly equipped with essential sensors and BMS to record the faults (Fig. 1c). The lack of operational data may lead to the poor generalization issue for data-driven FDD models.

1.3. Research gap on fault detection and diagnosis practical issues

Apart from the aforementioned two issues, there is another issue that has not yet been realized. According to Zhong et al. [36], HVAC fault

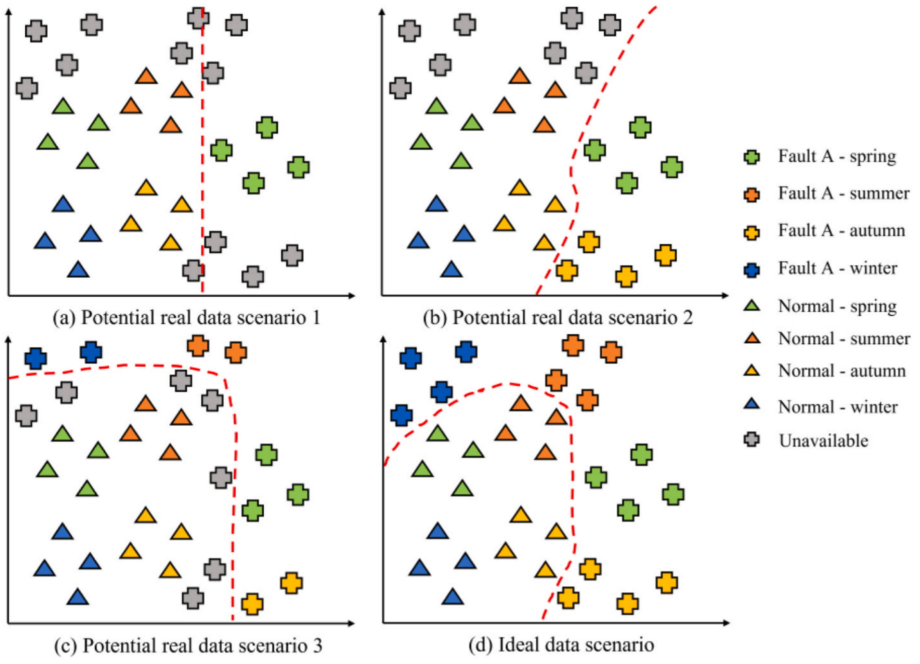


Fig. 2. Potential examples of FDD classification for real data scenarios and the ideal data scenario (the red dash line in each case represents the possible separation boundary predicted by the FDD model to differentiate the normal data from the fault A data). (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

impacts vary with changing climate conditions, as HVAC system operating conditions change with outdoor weather conditions. Lu et al. [37] indicated that HVAC fault impact patterns are correlated to system seasonal working conditions. However, the acquisition of fault data under all the seasons from real buildings is often difficult due to the frequency and the span of occurrences for different faults. This issue results in several possible data scenarios in reality: (1) the fault data under some classes only cover a certain season but is not available under other seasons (Fig. 2a); (2) the fault data under some classes covers two or more seasons, but is still lacking in remaining seasons (Fig. 2b); (3) the fault data under some classes cover all the seasons, and the total amount of data seems to be sufficient, but the amount of data under each season is limited (Fig. 2c). These possible data scenarios may induce potential performance degradations for HVAC FDD applications in real buildings, as a sufficient amount of fault data under one or several but not all the seasons cannot guarantee that the FDD models can fully learn different fault patterns in other season(s). Thus, the potential impact of this issue on the HVAC FDD performances was investigated in this study.

1.4. Aim and objectives

This study aims to evaluate the performance differences of a data-driven FDD method, namely, the combined CNN-RNN framework, under the limited seasonal fault data scenarios and the ideal fault data scenario covering climatic conditions from four seasons. The fault and normal data were collected from three sources: (1) fault simulations using a widely-used building performance simulation (BPS) program, EnergyPlus [38] based on a verified prototype medium office building model [39]; (2) experimental fault implementations from ASHRAE research project 1312 (ASHRAE RP1312) [40]; (3) experimental fault implementations using a flexible research platform (FRP) [41]. Based on the simulated fault dataset, three data scenarios representing different extents of the season coverage of fault data, i.e., one-season training scenario, dual-seasons training scenario and full-season quarter-size training scenario, and an ideal full-seasons full-size training scenario, were established for FDD evaluation experiments. According to the evaluation results, the performance differences among different data scenarios were compared, and the FDD performance degradation caused by fault data seasonal imbalance was identified. Moreover, two sets of further experiments based on the real fault datasets, including similar data scenarios to those from the experiments using the simulated dataset, were performed to verify the practical issue. As for the practical applications, the proposed architecture provides a general FDD method and can be extended to different case studies by modifying the number of CNN and RNN layer blocks or adding extra layer types to the baseline structure. The evaluation experiments for the seasonal data practical issue proposed in this study can be replicated following the architecture structure and hyperparameter configurations, and extended to more sophisticated quantitative experiments with specific seasonal data imbalance ratios. It should be noted that the values of FDD performance metrics may vary when reproducing experimental results due to randomness in data sampling and shuffling and differences in FDD training environment settings, but similar general trends should be observed.

2. Method

The workflow for evaluating the potential practical issue of limited fault data under one or multiple seasons is illustrated in Fig. 3. Firstly, the HVAC fault data was obtained from EnergyPlus fault simulations using a verified prototype building model and two real fault datasets. The descriptions for the simulated and real datasets are provided in sections 2.1 and 2.2, respectively. The collected data from each source was divided into four subsets according to the four seasons of a year, and the proportion of samples for each class in each subset was preserved as that in the original dataset. In the second stage, the 1D time-series data was then normalized and transformed into 2D data through a sliding window, as described in section 2.3. To ensure the detection and diagnosis ability of the proposed data-driven method is not the bottleneck when evaluating FDD performances under different data scenarios, an advanced deep learning architecture, i.e., combined CNN-RNN architecture, was employed in the third stage and explained in detail in section 2.4. Two types of cutting-edge RNNs, LSTM and GRU, that solve the vanishing and exploding gradient issues [42] for simple RNNs, were

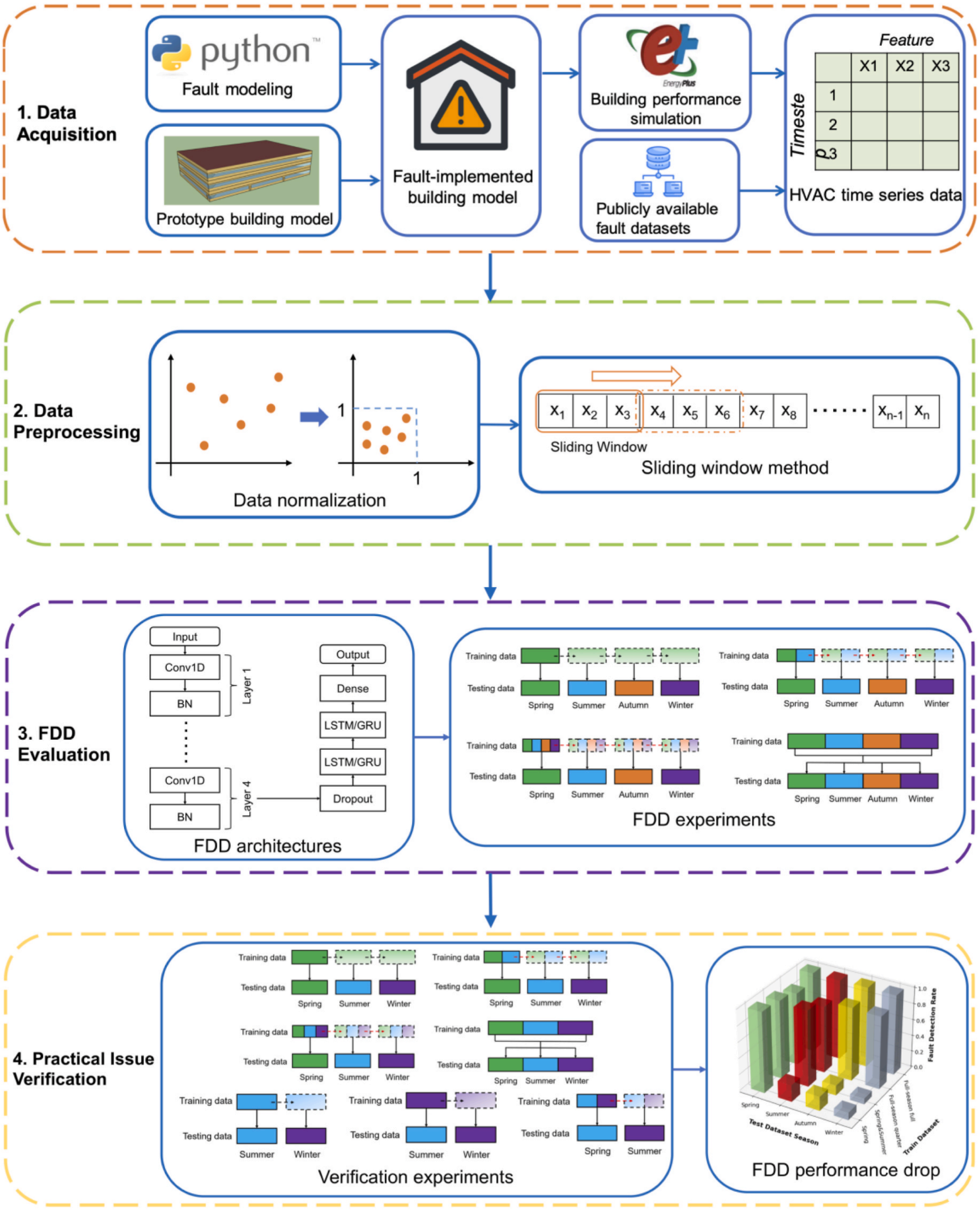


Fig. 3. Research workflow for evaluating the impact of different fault data scenarios on FDD performances.

respectively combined with CNNs to detect and identify HVAC faults. Based on the simulated dataset, the proposed architectures were assessed under three data scenarios based on the potential practical issue of limited seasonal fault data, and an ideal data scenario covering the climatic conditions of four seasons. After the initial evaluation, the architectures were assessed using similar data scenarios established based on the other two real fault datasets to verify the practical issue. The FDD evaluated performance metrics are described in section 2.5, and the details for the evaluation experiments are provided in section 2.6.

2.1. Description of simulated fault dataset

The prototype medium office building model from the department of energy (DOE) [39] was adopted to generate HVAC normal and fault data in this study. The prototype commercial building models were widely used for modeling the building energy consumption [43], assessing the HVAC fault impacts [44] and predicting the power usage [45] of commercial buildings under various climatic conditions. The medium office building is a three-level building (Fig. 4a) with a core and four perimeter zones on each level (Fig. 4b). The building is equipped with a VAV system coupled with three packaged air conditioners. Each air

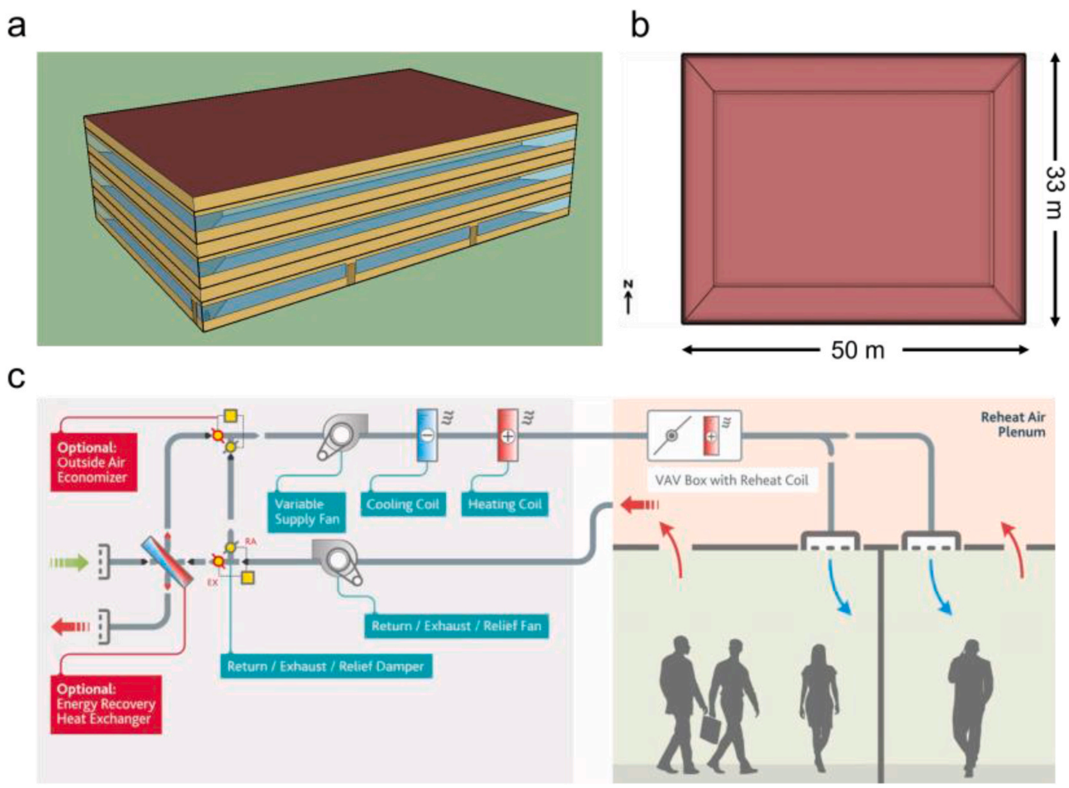

a
b
c
33 m
50 m
N
Optional: Outside Air Economizer
Variable Supply Fan
Cooling Coil
Heating Coil
VAV Box with Reheat Coil
Reheat Air Plenum
Return / Exhaust / Relief Fan
Return / Exhaust / Relief Damper
Optional: Energy Recovery Heat Exchanger
RA
EX

Fig. 4. (A) DOE prototype medium office building model; (b) The medium office building floor plan; (c) The layout of a typical VAV system with reheat coils and OA economizer [49].

Table 1
Simulated HVAC fault and normal data descriptions.

Class label	Fault type	Fault intensity	Sample size
0	Normal		45,696 per season
1	No overnight setback	Setpoint setback is off	576 per season
2	Cooling coil SA temperature sensor positive bias	+2 °C	576 per season
3	Cooling coil SA temperature sensor negative bias	−2 °C	576 per season
4	Heating coil SA temperature positive sensor bias	+2 °C	576 per season
5	Heating coil SA temperature negative sensor bias	−2 °C	576 per season
6	OA temperature sensor positive bias	+2 °C	576 per season
7	OA temperature sensor negative bias	−2 °C	576 per season
8	Thermostat positive offset	+2 °C	576 per season
9	Thermostat negative offset	−2 °C	576 per season
10	Supply fan motor efficiency degradation	15%	576 per season
11	Supply fan stuck	Stuck at the 50% speed	576 per season
12	OA damper stuck	Stuck at the 25% position	576 per season
13	OA damper leakage	Leaking percentage 40%	576 per season
14	VAV damper leakage	Leaking percentage 40%	576 per season

conditioner has a direct expansion (DX) cooling coil and a gas-fueled heating coil and serves each floor of the building. Regarding the air distribution, each zone is equipped with a VAV terminal unit which has an electric reheat coil and an outdoor air (OA) damper to adjust the OA fraction for the supply air (SA) (Fig. 4c). The simulated fault dataset was generated using fault simulations on EnergyPlus under the marine climate of London. The faults were modeled using Python programming language [46] according to fault modeling methods provided in Refs. [47,48] and introduced into the verified prototype medium office building model. 14 faults, including control, sensor, packaged air conditioner and VAV terminal faults, were simulated at a 5-min interval for 48 h under each season and the normal operating conditions of the VAV system were simulated at the same interval for adjacent 272 h in total before and after fault occurrences. Thus, each fault case accompanied by the adjacent normal system operation produced 3840 data samples in total for 320 h per season, and each one-season dataset has 53,760 samples for 4480 h. Overall, the ratio of the total amount of fault data to normal data is 0.18. The fault intensities, class labels and sample sizes for fault and normal classes are provided in Table 1. The more in-depth explanations for each fault can be found in Ref. [36].

2.2. Description of real fault datasets

The two real fault datasets were gathered from ASHRAE RP1312 [40] and National Renewable Energy Laboratory (NREL)'s fault tests [41], and are used to verify the practical issue identified from the fault data experiments based on the simulated dataset. The ASHRAE RP1312 carried out experimental measurements for an AHU in the Iowa Energy Resource Station under both fault and normal operating conditions. The Energy Resource Station is comprised of AHU–1, responsible for the common areas of the test facility; AHU–A and AHU–B, which are identical and each serves four mirror zones, as shown in Fig. 5a. Thus, a set of comparison experiments were carried out in these two AHUs. The fault tests were implemented using the AHU-A, and the comparative normal operating conditions were monitored using the AHU–B. The measurements for 6 faults, implemented under multiple seasons, i.e., spring, summer and winter, were extracted from the original raw dataset. The

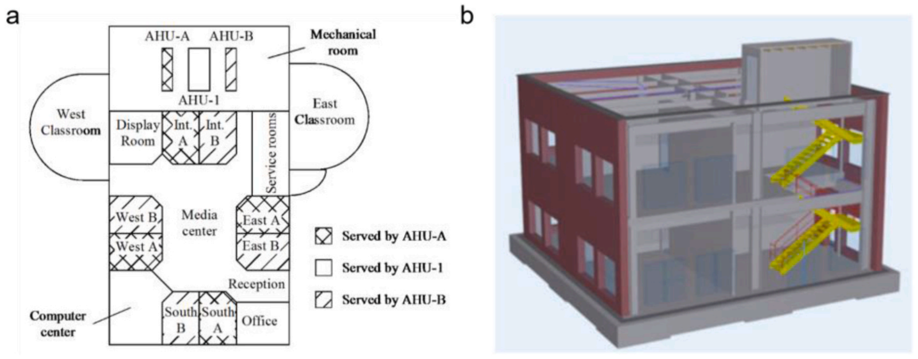


Fig. 5. (A) The layout of Iowa Energy Resource Station; (b) a 3D model of the FRP [50].

Table 2
Real HVAC fault and normal data descriptions.

ASHRAE RP1312 Dataset			
Class label	Fault type	Fault intensity	Sample size (spring/ summer/winter)
0	Normal		11,883/12,664/8076
1	OA damper stuck fully closed	Stuck at the 0% position	722/722/722
2	EA damper stuck fully open	Stuck at the 100% position	1440/1126/1319
3	EA damper stuck fully closed	Stuck at the 0% position	722/721/721
4	Cooling coil valve stuck fully closed	Stuck at the 0% position	715/714/N/A
5	Cooling coil valve stuck fully open	Stuck at the 100% position	1077/612/682
6	Return fan complete failure	Off	721/721/N/A
FRP Dataset			
Class label	*Fault type*	*Fault intensity*	*Sample size (summer/ winter)*
0	Normal		576/576
1	HVAC setback delayed onset	3 h	96/96
2	HVAC setback early termination	3 h	96/96
3	HVAC no setpoint setback	Off	96/96
4	Thermostat positive bias	2.2 °C	96/96
5	Thermostat negative bias	2.2 °C	96/96

Table 3
Sliding window specifications for the simulated dataset.

Window Length	Window stride	Number of windows in each one-season dataset
6	6	8960
12	12	4480
24	24	2240

faults are indicated to record at a 1-min interval from 6:00 to 18:00 for both AHUs. The fault types and their intensities, and sample sizes for fault and normal classes are presented in Table 2. The total amounts of data samples under spring, summer and winter are 17,280, 17,280 and 11,520, respectively. Besides, the ratio of the total amount of fault data to normal data is 0.41.

Regarding the NREL's fault dataset, the fault tests were performed in an FRP (Fig. 5b). The FRP is equipped with a DX rooftop unit (RTU) and a reheat VAV system. 32 fault tests were conducted in the FRP, and five of them, including HVAC setpoint setback faults and thermostat biases were taken into account in the verification. The reason for selecting these five faults is that only these faults were tested under multiple seasons, i.e., summer and winter. The fault data was recorded at a 15-min interval, and each fault test lasted for 24 h. The fault intensities and sample sizes for fault and normal classes are also shown in Table 2. Due to the limited amount of samples under each fault class, the fault amount under each season was extracted equally from the original raw dataset, in order to avoid the original large imbalance ratio between fault data under different seasons. The total amount of data samples is 1056 for each season, and the ratio of the total amount of fault data to normal data is 0.17.

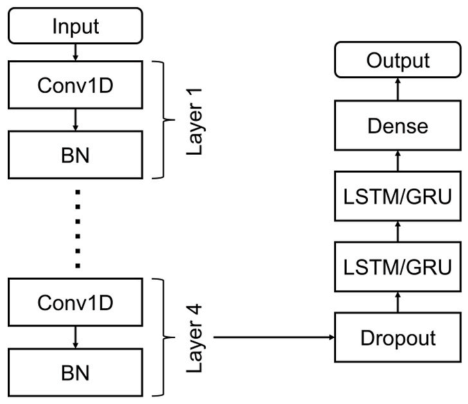


Fig. 6. The proposed combined CNN-RNN architectures, one using LSTMs as recurrent layers and another using GRUs as recurrent layers.

2.3. Data preprocessing techniques

The outliers in the raw data were removed, and the data for each feature was normalized using the maximum and minimum values of each feature to convert the data into the range between zero and one:

$$X_i^{'} = \frac{X_i - min(X_i)}{max(X_i) - min(X_i)} \tag{1}$$

where X_i is the original data sequence of each feature, $X_i^{'}$ is the normalized data sequence of each feature, $max(X_i)$ and $min(X_i)$ are the maximum and minimum values of the data sequence of each feature.

After the data normalization, the 1D time series data was transformed into 2D data using a sliding window since a single data sample was not informative enough to indicate a fault occurrence due to the dynamic operation of HVAC systems. The number of windows W_n was calculated using the following equation (2):

$$W_n = \frac{S_n - W_l}{W_s} + 1 \tag{2}$$

Where S_n is the number of data samples within the data sequence, W_l is

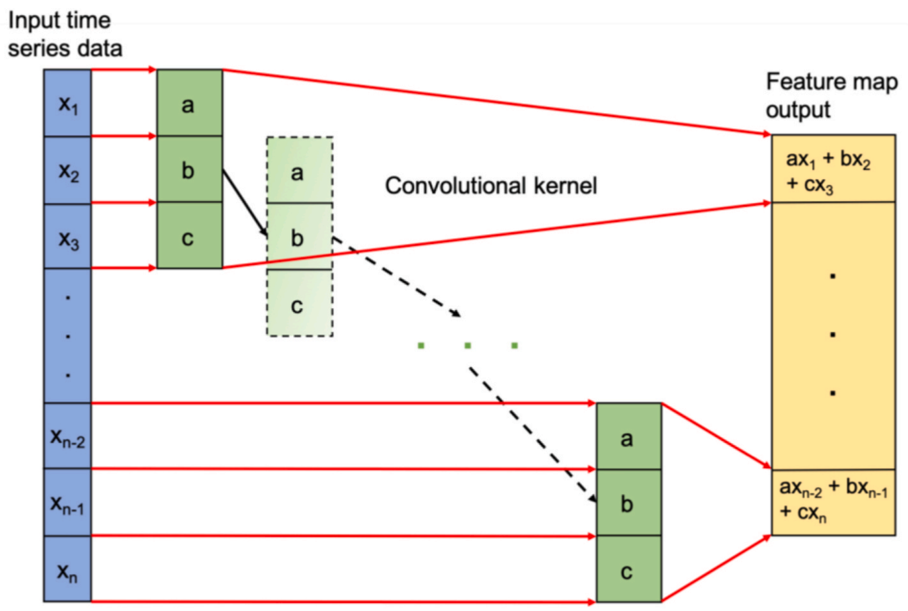


Fig. 7. An example of 1D convolution operation.

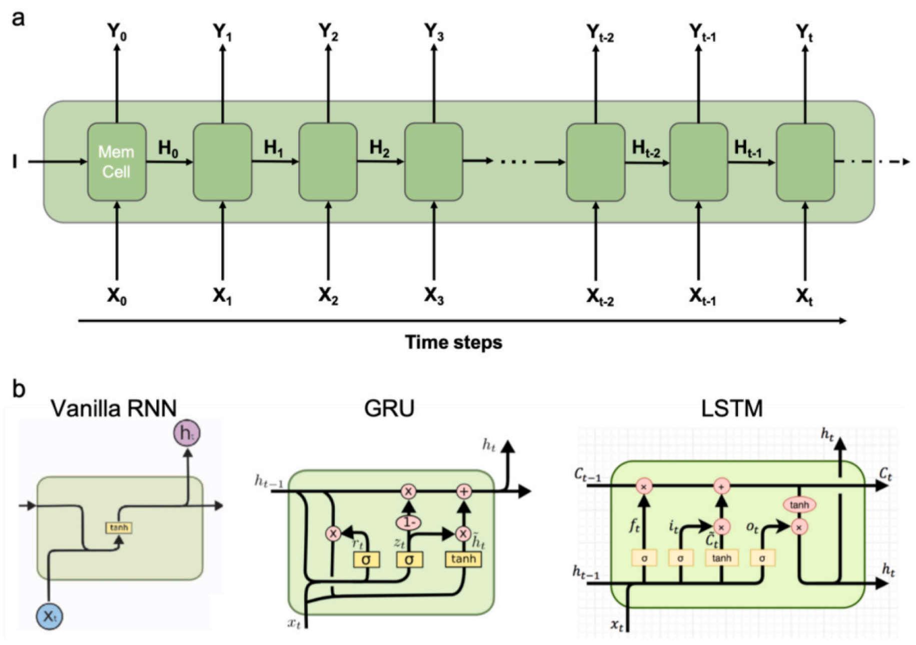


Fig. 8. (A) A typical recurrent layer; (b) The basic structures for vanilla RNN, GRU and LSTM.

the window length, W_s is the window sliding stride. The candidates of the window length considered in a parametric study were 6, 12 and 24 and also used for the window stride correspondingly, which means that there is no overlapping between adjacent windows. The resulting dataset sizes under each season for the simulated dataset are presented in Table 3.

2.4. Proposed fault detection and diagnosis architecture

The proposed architecture combines the convolutional and recurrent layers, as illustrated in Fig. 6.

Compared with the traditional shallow machine learning algorithms, such as SVM and fully-connected neural networks, the combined CNN-RNN architecture has the following advantages: (1) the convolutional part of the architecture can extract discriminative features from HVAC time series data, which contains a large number of sensor signal sequences and complex dynamic fault patterns [24], particularly in the present study; (2) the recurrent part can learn the temporal patterns in HVAC operational datasets that are usually high-dimensional [30]; (3) the CNN-RNN architecture results in an end-to-end learning, which mitigates the potential performance degradation due to the separation of feature extraction and classification [34]. The preliminary explanations for CNN and RNNs are provided in section 2.4.1. The structure of the proposed architecture and the used hyperparameters are presented in section 2.4.2.

2.4.1. Preliminary theory for the architecture

2.4.1.1. Convolutional neural network. Convolution layers are able to capture insightful features from input data through the convolutional

Table 4
Architecture hyperparameters for the experiments using the simulated dataset.

Hyperparameters	Values
Number of kernels in the input convolutional layer $K_{n,1}$	150, 200, 250 and 300
Number of kernels in the second convolutional layer $K_{n,2}$	300, 350, 400, 450, 500, 550 and 600
Number of kernels in the third convolutional layer $K_{n,3}$	2 $K_{n,2}$
Number of kernels in the third convolutional layer $K_{n,4}$	2 $K_{n,3}$
Kernel size	3, 5, 7 and 9
Stride	1
Padding	'same'
Convolutional dropout rate	0.15, 0.2 and 0.25
Number of units in recurrent layers	64, 128, 256 and 512
Recurrent dropout rate	0.1
Window length	6, 12 and 24
Batch sizes	32, 56, 64, 112 and 128
Training data size ratio	80%, 70% and 60%
Cross validation strategy and number of resplitting iterations	Stratified shuffle split for 10 times
Optimizer	SGD
Epochs	500 (with early stopping)
Learning rate	0.0001–0.002 (exact value depending on experiments)

kernel. As a result, the output feature map is generated by sliding the kernel over the input data or the feature map generated by the preceding convolution layer, as illustrated in Fig. 7. The output of a 1D convolution can be usually denoted as [24]:

$$(I * K)_i = \sum_{j=0}^{N-1} I_j K_{i-j} \tag{3}$$

where I represents the input time series sequence of length N and K is the convolutional kernel.

There are several key hyperparameters regarding the convolutional layer. The number of kernels represents the number of output feature maps from the current convolutional layer. The kernel size defines the scale of the convolution operation, i.e., the length of the 1D convolutional window in this case (green windows in Fig. 7). The stride is the step that the kernel slides over the input feature map. The padding parameter can be either 'valid' or 'same'. 'Valid' means that no padding is applied to input feature map(s). 'Same' leads to adding zeros around the edges of each input feature map so that the dimensionality of each output feature map can be kept the same as that of the corresponding input feature map. Apart from these, the activation function, which is used to transform the weighted sum of the input into the output of each node from each layer, for the convolutional layer is usually rectified linear activation function (ReLU), as expressed in Eqn. (4).

$$f(x) = max\ (0, x) \tag{4}$$

2.4.1.2. Recurrent neural network. Recurrent layers are capable of storing useful information from previous time steps and employing them for future classifications and predictions. As presented in Fig. 8a, a recurrent layer consists of a single memory cell that repeatedly computes the output Y_t for each time step. At each time step, the memory cell takes the input time series (starting from X_0 to X_t) at that time step and generates an output (starting from Y_0 to Y_t) for that time step. Besides, a hidden state vector (starting from H_0 to H_t) is generated at each time step and fed as an extra input to the memory cell at the next time step. However, vanilla RNNs usually suffer from vanishing gradient issue, and therefore two variants of the vanilla RNN, GRU and LSTM, have been adopted as solutions to this issue (Fig. 8b).

The LSTM was proposed by Hochreiter and Schmidhuber [51] in 1997 to resolve the vanishing gradient issue. Instead of using an activation function of the hidden state in the memory cells, three gates were employed in the memory cells: the input, forget and output gates. Each gate uses a sigmoid activation function to control the values passing through the gate. The forget gate f_t determines the degree of the information from the previous state to be forgotten. The output gate o_t controls which part of the cell is output to the hidden state h_t. The mathematical operations for the input i_t, forget f_t and output gates o_t, the candidate cell $\widetilde{c}_t$, the updated cell state c_t and the output of the current hidden unit h_t are expressed as follows:

$$i_t = \sigma(W_i[x_t + h_{t-1}] + b_i) \tag{5}$$

$$f_t = \sigma\left(W_f[x_t + h_{t-1}] + b_f\right) \tag{6}$$

$$o_t = \sigma(W_o[x_t + h_{t-1}] + b_o) \tag{7}$$

$$c_t^{\sim} = tanh(W_c[x_t + h_{t-1}] + b_c) \tag{8}$$

$$c_t = c_{t-1} \times f_t + c_t^{\sim} \times i_t \tag{9}$$

$$h_t = o_t \times tanh(c_t) \tag{10}$$

where W_i, W_f, W_o and W_c are the weighting vectors, b_i, b_f, b_o, b_c are the bias vectors, the subscripts i, f, o and c denote the input gate, forget gate, output gate and the memory cell, respectively. σ denotes the sigmoid function and $tanh$ is the hyperbolic tangent function.

GRU was proposed by Cho, van Merriënboer and Bahdanau [52] in 2014. Compared with LSTM, the GRU has only two gates and less parameters, and thus the training of GRU is more efficient and requires less computational power. The GRU consists of two gates, i.e., the update gate and the reset gate. Similar to the gates used in LSTM, the sigmoid activation function is used for each gate. The update gate determines the degree of the cell state to be updated with the candidate state. The reset gate determines whether the previous hidden state should be kept or not. The mathematical operations for the update gate, reset gate, the candidate cell and the output of the current hidden unit are represented as follows:

$$z_t = \sigma(W_z[x_t + h_{t-1}] + b_z) \tag{11}$$

$$r_t = \sigma(W_r[x_t + h_{t-1}] + b_r) \tag{12}$$

$$h_t^{\sim} = tanh(W_h[x_t + r_t \times h_{t-1}] + b_h\Big) \tag{13}$$

$$h_t = (1 - z_t) \times h_{t-1} + z_t \times\ h_t^{\sim} \tag{14}$$

where W_z, W_r and W_h are the weighting vectors, b_z, b_r and b_h are the bias vectors, the subscripts z, r, and h denote the update gate, the reset gate and the hidden unit, respectively.

2.4.2. Structure of the proposed architecture

The proposed architecture consists of four stacked 1D convolutional layers and two stacked recurrent layers. Each convolutional layer is followed by a batch normalization (BN) layer [53] to accelerate the architecture training. In addition, the only dropout layer [54] is added after all convolutional layers and BN layers to avoid the overfitting issue and the disharmony between the dropout and BN techniques [55]. The dropout rates of 0.15, 0.2 and 0.25 were attempted in this study. Besides, ReLu [56] is applied to each convolutional layer as the activation function, and the 'same' padding [56] is employed for the convolutions. The candidates for the number of convolutional kernels in each layer and the kernel size for the experiments using the simulated dataset are shown in Table 4. As for the recurrent layers, two types of cutting edge RNNs are employed in this part of the architecture, namely, LSTMs and GRUs. Both types of recurrent layers are able to learn long-term temporal correlations. 'Tanh' activation function [54] is adopted for the recurrent layers. The regularization of the recurrent layers is implemented by applying 'recurrent dropout' to the second recurrent layer,

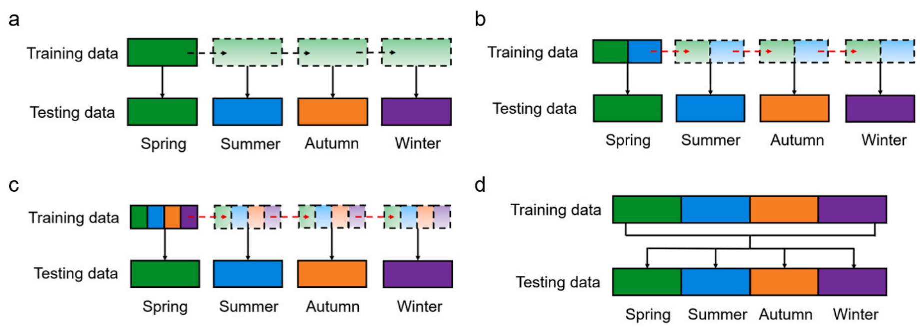


Fig. 9. The proposed data experiments under the simulated fault dataset: (a) Training data: a one-season dataset; testing data: each season's dataset (spring as an example here). (b) Training data: a mixed dataset of two seasons of half size; testing data: each season's dataset (a mixed dataset of spring and summer as an example here). (c) Training data: a mixed dataset of four seasons of quarter size; testing data: each season's dataset. (d) Training data: a mixed dataset of four seasons of full size; testing data: each season's dataset.

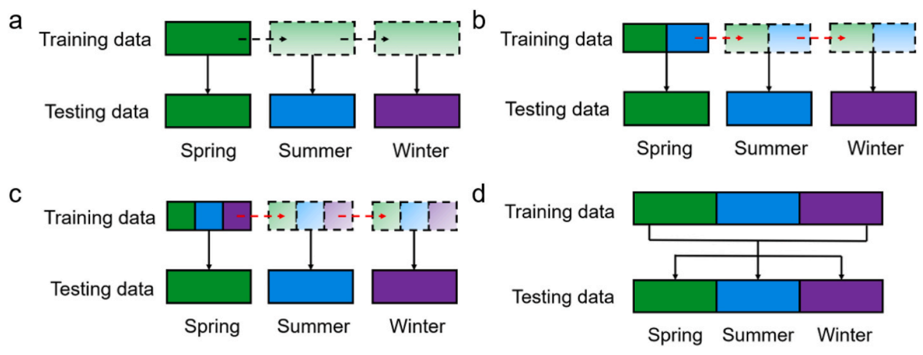


Fig. 10. The proposed data experiments under the ASHRAE RP1312 fault dataset: (a) Training data: a one-season dataset; testing data: each season's dataset (spring as an example here). (b) Training data: a mixed dataset of two seasons of half size; testing data: each season's dataset (a mixed dataset of spring and summer as an example here). (c) Training data: a mixed dataset of three seasons of quarter size; testing data: each season's dataset. (d) Training data: a mixed dataset of three seasons of full size; testing data: each season's dataset.

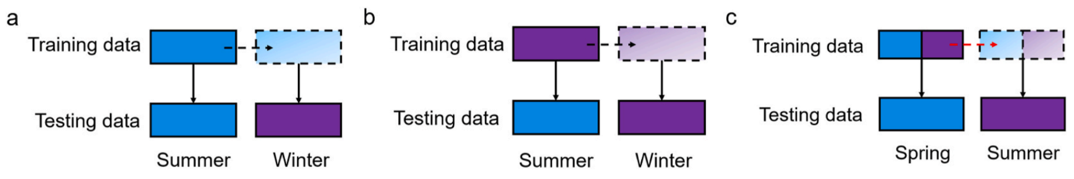


Fig. 11. The proposed data experiments under the NREL's fault dataset: (a) Training data: the summer dataset; testing data: each season's dataset. (b) Training data: the winter dataset; testing data: each season's dataset. (c) Training data: a mixed dataset of summer and winter of half size from each; testing data: each season's dataset.

Table 5
Baseline configurations for the two architectures used in the general hyper-parameter analysis.

Architecture	Combined CNN-LSTM	Combined CNN-GRU
Number of kernels for Conv 1	150	200
Number of kernels for Conv 2	500	450
Number of kernels for Conv 3	1000	900
Number of kernels for Conv 4	2000	1800
Kernel size	3	3
Convolutional dropout rate	0.15	0.2
Number units for Recurrent 1	512	512
Number units for Recurrent 2	256	256
Recurrent dropout rate	0.1	0.1

which drops the connections between the recurrent units [57], and the dropout rate is set to 0.1. Apart from this, the candidates for the number of recurrent units in each layer are 64, 128, 256 and 512. At last, a dense layer with the activation function of 'softmax' [56] is used after the recurrent layers to produce the probability for each class.

The architecture was trained using back-propagation [58] to update the weights and biases of the neurons. The back-propagation method propagates backward the loss between the predicted output and the actual output from the output layer to the input layer. The gradient of the loss is then calculated with respect to the weight and bias of each neuron, as follows:

$$\frac{\partial \mathscr{L}}{\partial W^{[l]}} = \frac{\partial \mathscr{L}}{\partial z^{[l]}} \cdot \left[a^{[l-1]}\right]^T \tag{15}$$

$$\frac{\partial \mathscr{L}}{\partial b^{[l]}} = \frac{\partial \mathscr{L}}{\partial z^{[l]}} \tag{16}$$

$$z^{[l]} = W^{[l]} \cdot a^{[l-1]} + b^{[l]} \tag{17}$$

where $\mathscr{L}$ is the loss function, $W^{[l]}$ is the weight vector of the l th layer, $b^{[l]}$ is the bias vector of the l th layer, $z^{[l]}$ is the weighted sum of the activations from the previous layer with a bias, $a^{[l-1]}$ is the activations from the previous layer. The gradient descent algorithm updates the weights and biases of each neuron in order to miminize the loss. In this study, the stochastic gradient descent (SGD) algorithm [59] was used as the optimizer since it performs a parameter update for each training sample and can reach fast convergence.

$$\theta = \theta - \eta \cdot \nabla_\theta J\left(\theta; x^{(i)}; y^{(i)}\right) \tag{18}$$

where θ is the model parameters, η is the learning rate, $J(\theta; x^{(i)}; y^{(i)})$ is the cost function for each training sample $x^{(i)}$ and its corresponding label $y^{(i)}$.

Due to the multi-class classification nature of the FDD problem, categorical cross entropy was used as the loss function. Besides, the maximum number of epochs was set to 500 and the early stopping

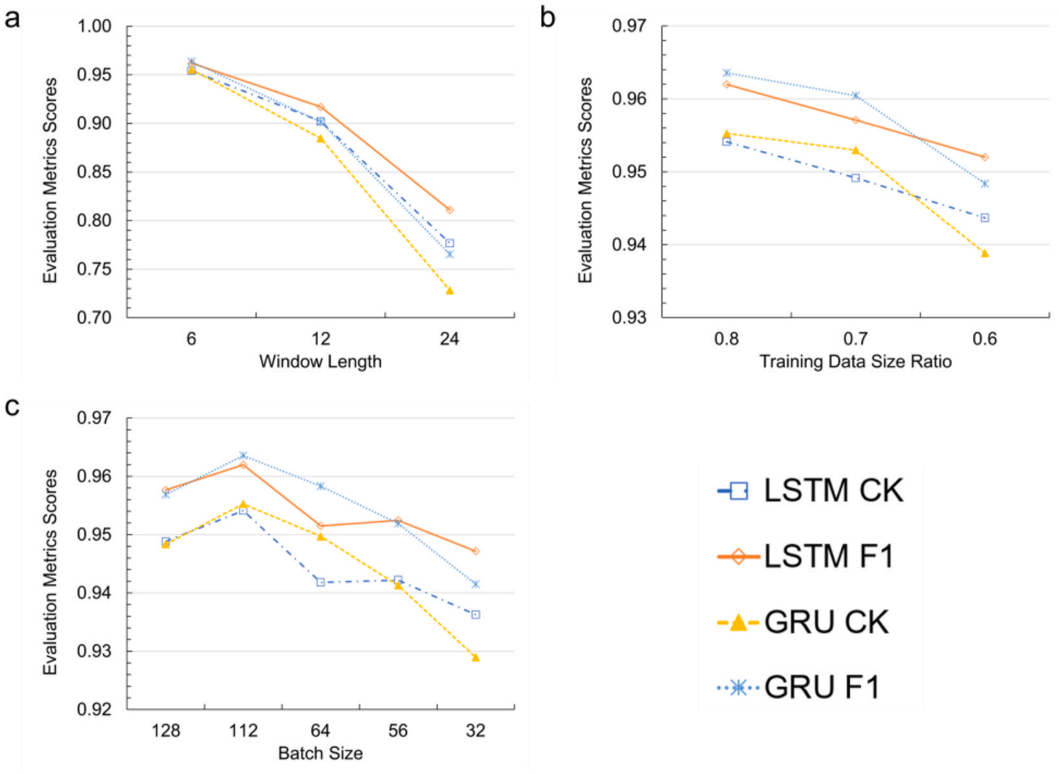


Fig. 12. FDD performance comparisons among different (a) window lengths; (b) training data size ratios; (c) batch sizes.

Table 6
FDD highest test metrics scores under each season for the architectures trained on one-season, dual-seasons and full-seasons datasets. (The values in italics are the larger scores between the two architectures under the same test season and training scenario, and the values in bold are the highest scores under this test season for one-season, dual-seasons and full-seasons scenarios, respectively).

Training data	Architecture	Spring		Summer		Autumn		Winter	
		F1	CK	F1	CK	F1	CK	F1	CK
Spring	CNN-LSTM	**0.975**	**0.971**	*0.081*	*0.020*	*0.074*	0.037	0.072	0.027
	CNN-GRU	0.975	0.970	0.079	0.009	0.072	*0.041*	*0.078*	*0.048*
Summer	CNN-LSTM	0.068	0.010	**0.972**	**0.966**	0.082	0.014	0.067	−0.007
	CNN-GRU	*0.075*	*0.027*	0.968	0.961	*0.089*	*0.049*	*0.079*	*0.028*
Autumn	CNN-LSTM	0.080	0.018	*0.059*	*0.015*	0.983	0.980	0.064	0.007
	CNN-GRU	*0.087*	*0.030*	0.057	0.006	**0.984**	**0.982**	*0.070*	*0.011*
Winter	CNN-LSTM	0.078	0.023	0.054	0.016	0.079	0.023	0.979	0.975
	CNN-GRU	*0.082*	*0.026*	*0.055*	0.016	*0.084*	*0.033*	**0.982**	**0.979**
Spring&Summer	CNN-LSTM	0.937	0.925	0.895	0.873	0.099	0.055	0.081	0.037
	CNN-GRU	**0.942**	**0.933**	*0.901*	*0.880*	*0.109*	*0.067*	*0.087*	*0.046*
Summer&Winter	CNN-LSTM	*0.122*	0.104	**0.908**	**0.890**	0.115	0.092	0.948	0.940
	CNN-GRU	0.118	0.104	0.902	0.884	**0.123**	**0.101**	**0.949**	**0.941**
Quarter size	CNN-LSTM	*0.862*	*0.841*	*0.790*	*0.759*	*0.862*	*0.842*	*0.858*	*0.836*
	CNN-GRU	0.850	0.827	0.771	0.737	0.846	0.823	0.845	0.821
Full size	CNN-LSTM	0.977	0.973	0.951	0.940	0.972	0.968	0.973	0.969
	CNN-GRU	**0.978**	**0.974**	**0.958**	**0.948**	**0.977**	**0.973**	**0.978**	**0.974**

technique was employed to stop the training if the model performances converged before the defined maximum number of epochs. The best learning rate for each experiment was found by setting a learning rate scheduler that gradually grows from 0.00001 to 0.1 during a training practice. The candidates for batch sizes and training data size ratios are presented in Table 4. The validation and testing data size was defined as 50% of the remaining dataset, respectively. In addition, the stratified shuffle split method [60] was used as the cross validation strategy, which splits the dataset by preserving the original proportion of data samples from each class in the split sets. The split was repeated ten times, and thus the architecture was trained, validated and tested on ten different data distributions. All the experiments were conducted using a workstation with an AMD Ryzen 9 3900X 12-core processor, an Nvidia RTX 3080 graphic card and 32 GB RAM.

2.5. Fault detection and diagnosis evaluation metrics

Due to the imbalanced multi-class classification nature of the FDD problem, some machine learning evaluation metrics, such as accuracy, may produce misleading high scores that incorrectly reflect good model performances, as data-driven FDD models have a strong capability of identifying the majority class, i.e., the HVAC normal conditions. Therefore, F1 score [25] and Cohen's Kappa (CK) score [61] were used to evaluate the general classification capability of the proposed FDD architectures. The average metric scores of the ten-times resplitting iterations were used to evaluate the architectures' performances in

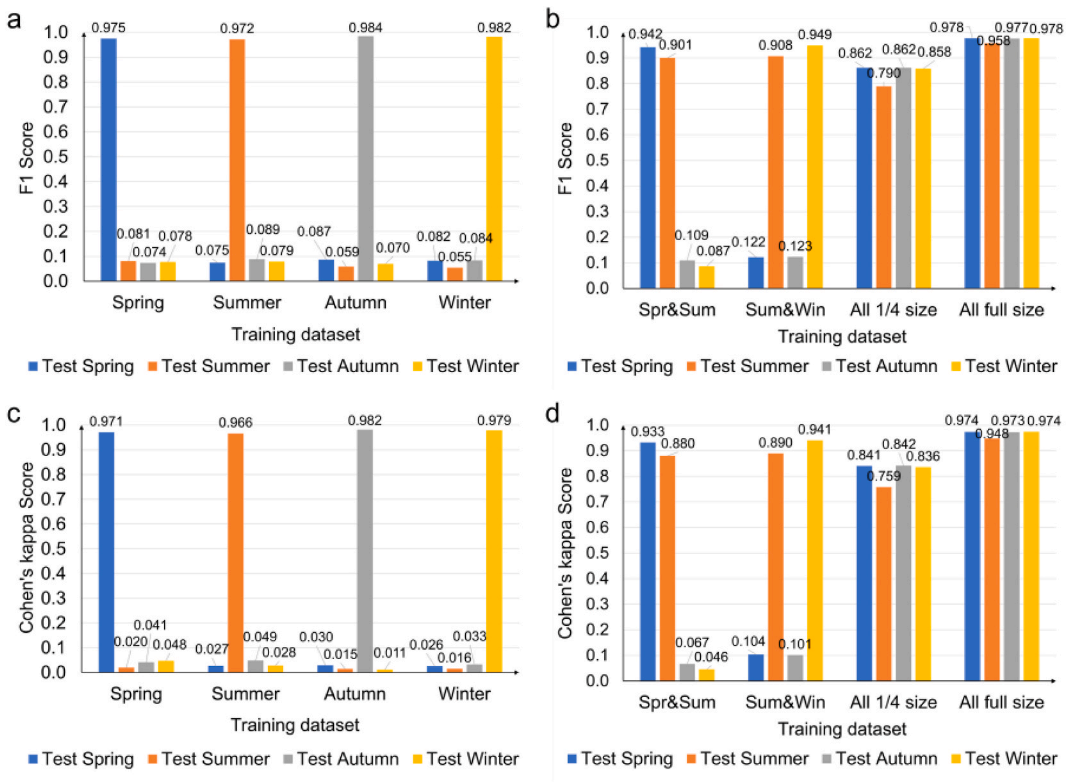


Fig. 13. Highest F1 scores among architectures trained on (a) one-season datasets, (b) dual-seasons datasets and full-seasons datasets; highest CK scores among architectures trained on (c) one-season datasets, (d) dual-seasons datasets and full-seasons datasets ('All 1/4 size' represents the full-season quarter size scenario; 'All full size' represents the full-season full size scenario).

Table 7
Average fault detection rate, fault diagnosis rate, false positive rate and false negative rate among 10 cross-validation times for the best architecture in experiments 1–4 under the simulated dataset.

Test dataset	Fault detection rate	Fault diagnosis rate	False positive rate	False negative rate
Exp 1: CNN-LSTM trained on the spring dataset				
Spring	0.999	1.000	0.009	0.001
Summer	0.208	0.144	0.178	0.792
Autumn	0.176	0.128	0.117	0.824
Winter	0.091	0.111	0.058	0.909
Exp 2: CNN-GRU trained on the mixed spring-summer dataset				
Spring	0.970	0.999	0.019	0.030
Summer	0.947	1.000	0.033	0.053
Autumn	0.122	0.411	0.066	0.878
Winter	0.071	0.254	0.029	0.929
Exp 3: CNN-LSTM trained on the full-seasons quarter-size mixed dataset				
Spring	0.902	0.998	0.034	0.098
Summer	0.829	1.000	0.046	0.164
Autumn	0.909	0.993	0.029	0.091
Winter	0.900	0.996	0.037	0.100
Exp 4: CNN-GRU trained on the full-seasons full-size mixed dataset				
Spring	0.999	1.000	0.008	0.001
Summer	0.996	1.000	0.017	0.004
Autumn	0.998	1.000	0.009	0.002
Winter	0.994	1.000	0.007	0.006

different data experiments. The F1 score for each class is calculated using the precision and recall scores for each class as follows:

$$F1 = 2\cdot\frac{Precision\cdot Recall}{Precision + Recall} \tag{19}$$

where precision and recall are calculated using the following equations:

$$Precision = \frac{TP}{TP + FP} \tag{20}$$

$$Recall = \frac{TP}{TP + FN} \tag{21}$$

Where *TP* represents the number of true positive predictions, *FP* represents the number of false positive predictions, *FN* represents the number of false negative predictions [62].

The Cohen's Kappa score *k* is calculated by the following equation:

$$k = \frac{p_o - p_e}{1 - p_e} \tag{22}$$

Where p_o represents the observed proportional agreement of the true and predicted labels, p_e represents the probability of random agreement of both true and predicted labels.

In addition, the fault detection rate, fault diagnosis rate, false positive rate and false negative rate were calculated based on the confusion matrices. The fault detection rate is defined as the ratio of the amount of detected faults to the total amount of faults. The fault diagnosis rate represents the ratio of the amount of correctly diagnosed faults (the fault type is correctly predicted) to the total amount of detected faults. The false positive rate is the ratio of the amount of the actually negative (normal conditions) samples that are predicted as positive samples (faults) to the total amount of negative samples. The false negative rate is the ratio of the amount of the actually positive (faulty conditions) samples that are predicted as negative samples (normal conditions) to the total amount of positive samples.

2.6. Fault detection and diagnosis evaluation experiments setup

The experiments for assessing the hypothetical fault detection and diagnosis practical issue consist of three parts: (1) some pre-experiments

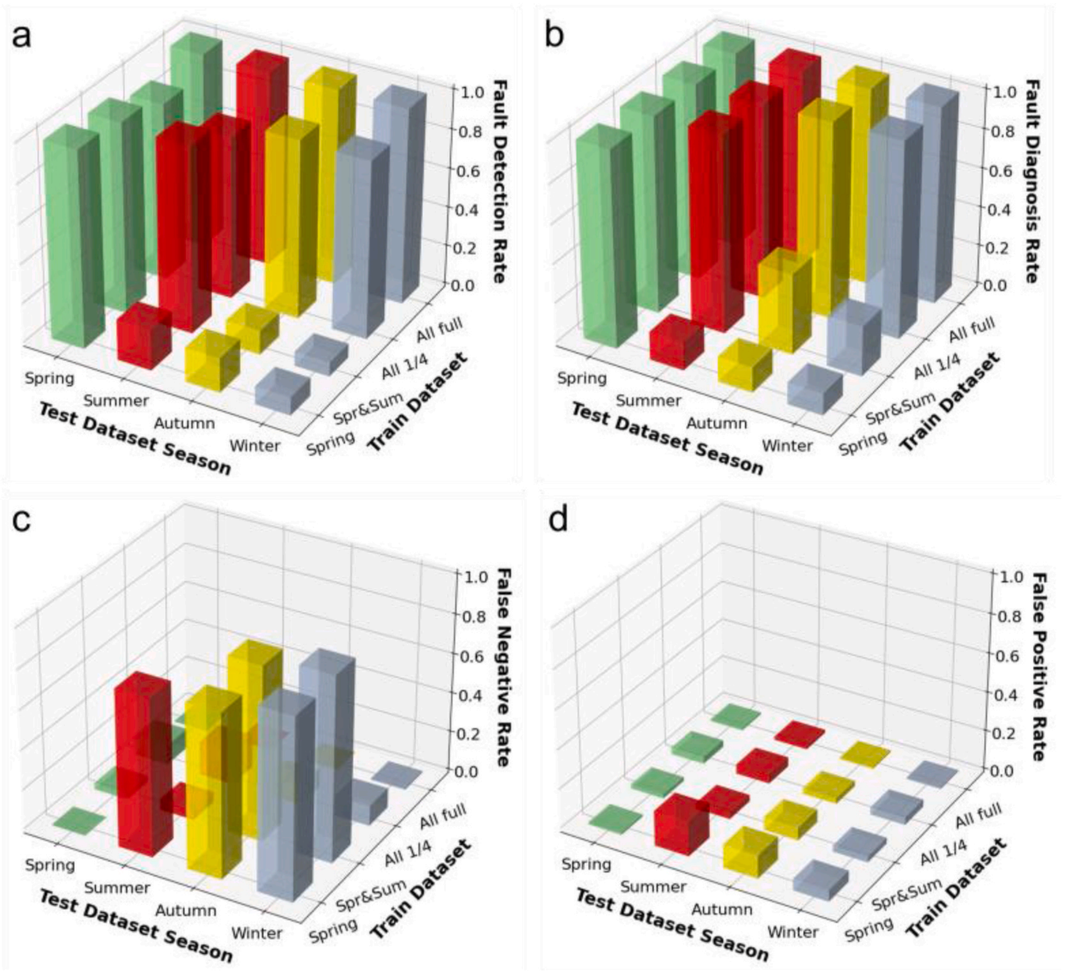


Fig. 14. Fault detection rate, fault diagnosis rate, false negative rate and false positive rate tested under each season for experiments 1–4 using the simulated dataset ('All 1/4' represents the full-season quarter size scenario; 'All full' represents the full-season full size scenario).

were conducted to find the optimal values for window length, training data size ratio and batch size; (2) four data experiments, considering the potential practical issue of limited seasonal fault data and an ideal data scenario, were implemented using the simulated fault dataset; (3) similar experiments to those based on the simulate dataset were carried out using the real fault datasets, in order to verify the findings obtained from experiments in part 2.

2.6.1. *Hyperparameter tuning analysis*

A hyperparameter analysis was first conducted to determine the values of some general parameters, including window length, training data size and batch size. The candidates for these hyperparameters, as listed in Table 4, were selected for evaluating the resultant FDD performances. The general hyperparameter tuning was performed on a dataset of a certain season. The objective of this analysis was to first fix these general hyperparameters to the optimal values, in order to save the computational resource for searching the remaining deep neural network hyperparameters in the data experiments.

2.6.2. *Fault detection and diagnosis evaluation data scenarios*

The simulated dataset includes four equal-sized subsets for four seasons, respectively. The sample amount of each fault class is the same in each subset. Based on the simulated dataset, four data experiments were proposed to assess the influence of the practical issue of limited fault data under one or multiple seasons on the proposed FDD architectures' performances:

(1) The architectures were trained on a certain season's training set and tested on each season's full-size dataset, as illustrated in Fig. 9a.
(2) The architectures were trained on a mixed training set that contains half data from each of two different seasons and is of the same total size as the one-season training set in experiment 1, and tested on each season's full-size dataset, as illustrated in Fig. 9b.
(3) The architectures were trained on a mixed training set that includes a quarter of data from each season and is also of the same size as the one-season training set in experiment 1, and tested on each season's full-size dataset, as illustrated in Fig. 9c.
(4) The architectures were trained on a mixed training set that contains full-size data from each season and is four times the size of the one-season training set in experiment 1, and tested on each season's full-size dataset, as illustrated in Fig. 9d.

It should be noted that the amount of normal and fault data in each scenario is stratified, which means that the imbalanced ratio of each fault class to the normal class is maintained, and the proportion of each fault class in each mixed dataset remains the same as 0.02 as that in each one-season's dataset. It should also be noted that the effect of imbalanced ratios on the FDD performances is not evaluated in this study since this has already been studied in Ref. [35].

2.6.3. *Fault detection and diagnosis practical issue verification*

To verify the findings from the previous experiments detailed in section 2.6.2, similar data scenarios were proposed according to the

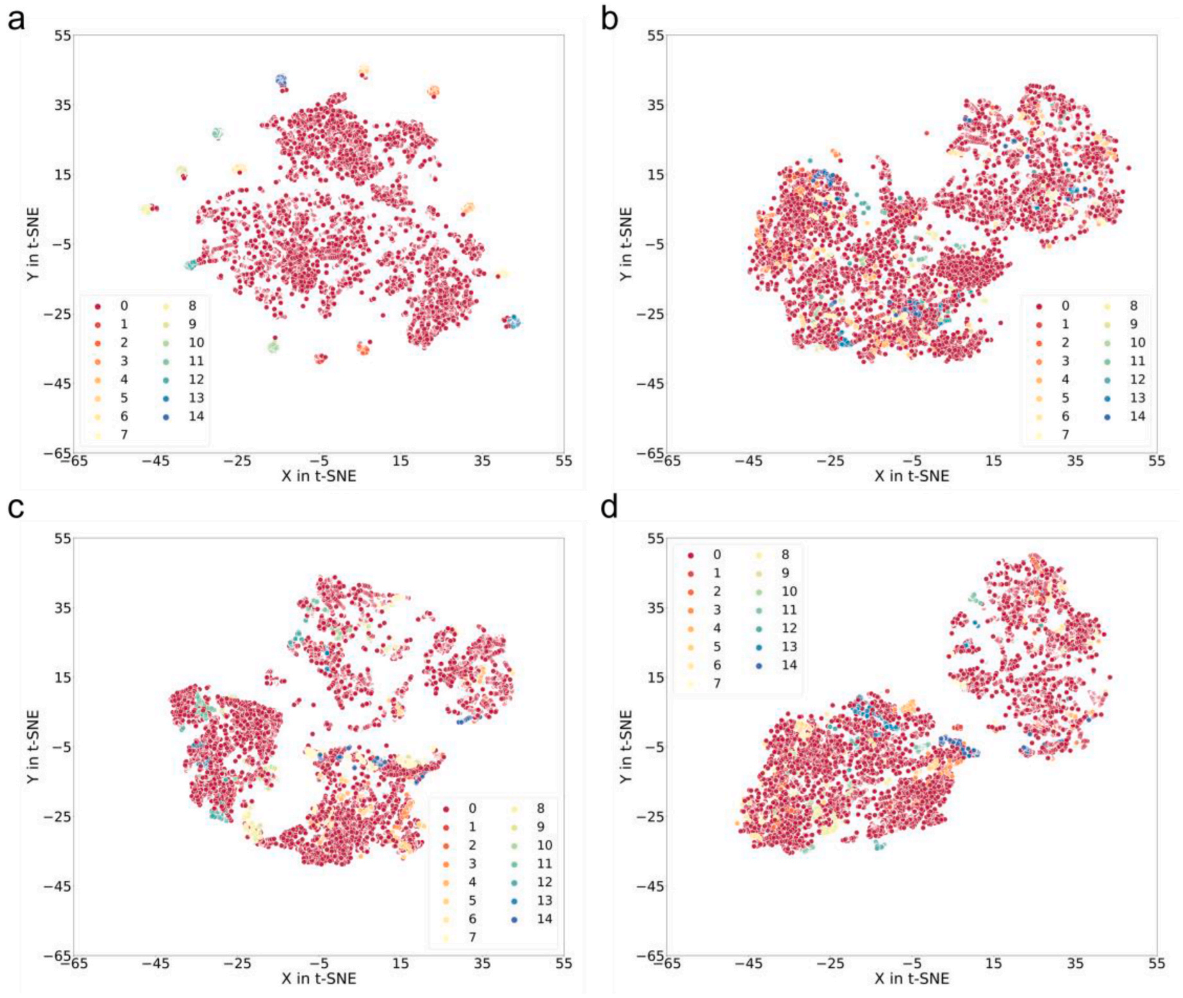


Fig. 15. Scatterplots for the hidden features extracted by the best architecture CNN-LSTM tested under (a) spring; (b) summer; (c) autumn; (d) winter; in experiment 1 using the *t*-SNE method.

ASHRAE RP1312 dataset. The only difference between this set of experiments and the previous set of experiments in section 2.6.2 lies in the coverage of seasons. The seasons considered in this set of experiments are only spring, summer and winter due to the data availability in this dataset, as illustrated in Fig. 10. In addition, the NREL's dataset is fully used to produce three data scenarios, including two one-season training scenarios using the summer and winter datasets, respectively, and a mixed dual-seasons scenario of summer and winter, as presented in Fig. 11.

3. Results and discussions

The hyperparameter analysis for the window length, training data size ratio and batch size is presented in section 3.1. The results for four experiments, considering the potential practical issue and an ideal data scenario, based on the simulated fault dataset are discussed in section 3.2. The verification results, based on the real fault datasets, for findings from the simulated data experiments are explained in section 3.3.

3.1. General hyperparameter analysis results

This analysis was implemented using the combined CNN-LSTM and the combined CNN-GRU, respectively. The baseline configurations for these two architectures are presented in Table 5. The FDD performances, in terms of F1 and CK scores, for window lengths, training data size ratios, and batch sizes, are illustrated in Fig. 12. The window length of 6 contributed to the highest scores, while the window lengths of 12 and 24 led to lower and significantly lower scores. This indicates that the smaller window length is more robust in capturing the fault patterns, as the smaller window length leads to a larger number of windows fed into the FDD model training when there is a limited amount of fault data available for training. As for the training data size ratio, the performance scores reduced with the decrease in the ratio. Besides, an adequate amount of data needs to be left for validating and testing the model, and thus the training data size ratio is set to 0.8 without considering higher ratios. Regarding the batch size, the peak of the performance scores can be observed at the batch size of 112. On one hand, the batch size of 128 included a higher variability in feature patterns and a higher number of windows for each class, but failed to outperform the batch size of 112, which may be due to the less times of model weights adjustments per epoch when the batch size of 128 is used. On the other hand, the smaller batches than 112 indeed increased the times for updating the model weights per epoch, however, they were likely to contain lower variability in feature patterns and some of them might not be able to take into account sufficient amount of data windows for each class. To conclude, the window length of 6, the training data size ratio of 0.8 and the batch size of 112 are used for further experiments.

3.2. Fault detection and diagnosis evaluation experiments using simulated fault dataset

The results obtained from the four data experiments are discussed in

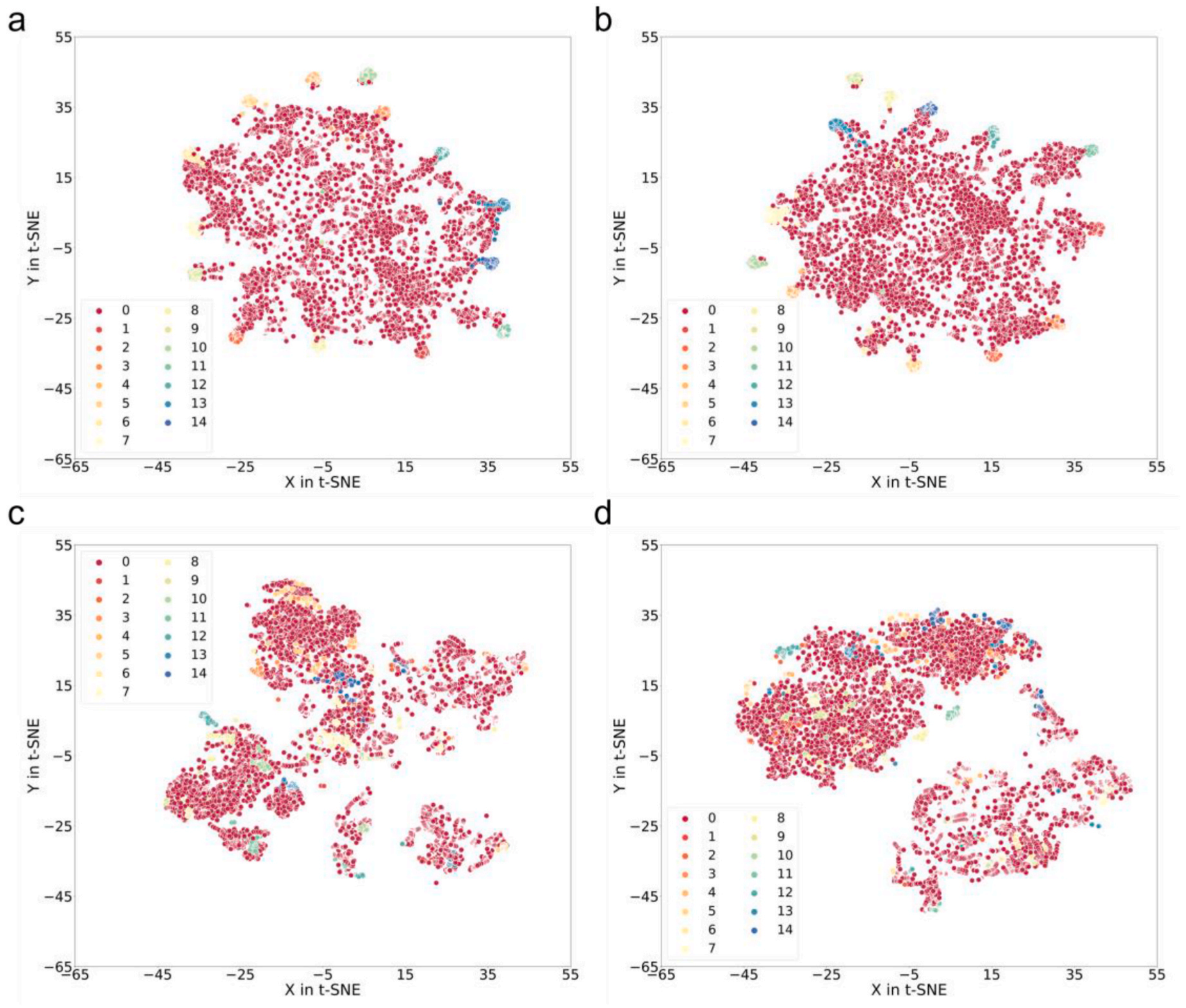


Fig. 16. Scatterplots for the hidden features extracted by the best architecture CNN-GRU tested under (a) spring; (b) summer; (c) autumn; (d) winter; in experiment 2 using the *t*-SNE method.

subsections 3.2.1-3.2.4. The highest F1 and CK scores tested under each season for both architectures, CNN-LSTM and CNN-GRU, under each experiment are displayed in Table 6. Based on Table 6, the higher F1 and CK scores between the two architectures are shown in Fig. 13. In addition, the average fault detection, fault diagnosis, false positive and false negative rates among 10 cross-validation iterations for the best architecture, trained on the spring only, mixed spring and summer, full-seasons quarter-size and full-seasons full-size datasets, are shown in Table 7 and illustrated in Fig. 14.

3.2.1. *Experiment 1 results: one-season dataset evaluation*

Experiment 1 was conducted based on each one-season dataset. Comparing the test performances of the two architectures under the season same as the training data's in Table 6, the CNN-GRU architecture outperformed another architecture CNN-LSTM under autumn and winter, but indicated slightly poorer scores under spring and summer. In general, the FDD architectures only indicated good test metrics under the seen season from the training, but showed significantly poor scores under every unseen season (Fig. 13a and c).

The best architecture CNN-LSTM indicated significantly high fault detection and diagnosis rates (Fig. 14a and b), and low false positive and negative rates when evaluated under the spring full-size dataset (Fig. 14c and d), from which the training data was extracted. However, the fault detection and diagnosis rates tested on the datasets of other three seasons are considerably lower than those tested on the spring dataset (Fig. 14a and b). The t-stochastic neighbor embedding (*t*-SNE) dimensionality reduction method was used to visualize the 2D distribution of hidden features extracted by this architecture. The good separation of features from different classes indicates the robust FDD ability of the architecture. As shown in Fig. 15a, the fault classes, labelled as 1 to 14 (see Table 1), are separated into clusters with obvious boundaries among them when tested under the spring full-size dataset, and only a small amount of features from fault classes overlapped with those from the normal class, labelled as 0. As for the hidden features from the test datasets of the other three seasons (Fig. 15b–d), there are no obvious boundaries among different classes. These findings confirm the poor generalization ability of the FDD architecture trained on sufficient fault data from a certain season in identifying the faults under the unseen seasons, which agrees with the finding from previous fault impact analysis studies [36,37] that HVAC fault impacts vary with different climatic conditions. In addition, the fault detection and diagnosis rates decreased along with the season transitions from summer, to autumn, and to winter (Fig. 14a and b). Meanwhile, the false positive rate decreased (Fig. 14d) and the false negative rate increased with the season transitions (Fig. 14c). This phenomenon indicates that the model is becoming more biased towards the majority class, i.e., the normal class, from summer to winter.

3.2.2. *Experiment 2 results: dual-seasons dataset evaluation*

Experiment 2 was carried out on two mixed datasets of dual seasons, i.e., spring and summer, and summer and winter. Comparing the performances of the two architectures in Table 6, the CNN-GRU

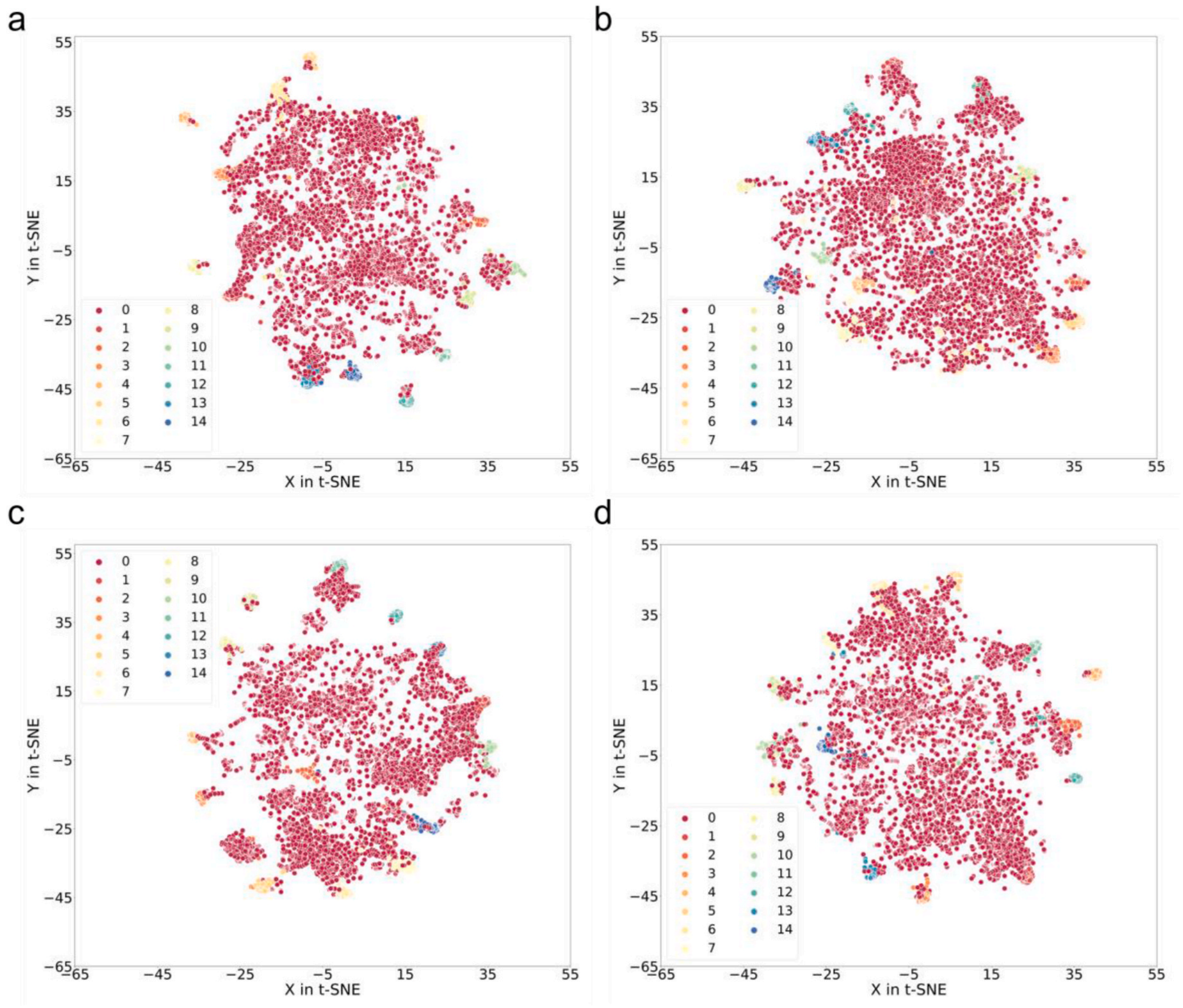


Fig. 17. Scatterplots for the hidden features extracted by the best architecture CNN-LSTM tested under (a) spring; (b) summer; (c) autumn; (d) winter; in experiment 3 using the *t*-SNE method.

architecture trained on the mixed dataset of spring and summer outperformed the corresponding CNN-LSTM architecture when tested under each season. Besides, the CNN-GRU architecture trained on the mixed dataset of summer and winter also indicated higher performance scores when tested under autumn and winter, but showed slightly poorer scores than the CNN-LSTM architecture when tested under summer. The highest performance scores, tested under the seen seasons from training, decreased slightly in experiment 2 (Fig. 13b and d), compared with those from experiment 1 (Fig. 13a and c). This can be attributed to the less amount of fault data under each season contained in each dual-seasons' dataset from experiment 2 than that included in each one-season's dataset from experiment 1.

As shown in Table 7, the fault detection and diagnosis rates are higher than 0.94 and around 1, respectively, when tested under seen seasons, i.e., spring and summer, but are lower than 0.12 and 0.41, respectively, when tested under unseen seasons, i.e., autumn and winter. The considerable performance differences between seen and unseen seasons can also be observed in Fig. 16. The feature clusters for most fault classes tested under seen seasons are separated from each other and from the normal class (Fig. 16a and b), and those for the classes tested under unseen seasons are mixed up (Fig. 16c and d). This conforms with the finding from experiment 1 that the FDD architectures have limited ability to detect and diagnose the faults from the unseen seasons. Compared with the performance indicators for the CNN-LSTM architecture trained on the spring only dataset in experiment 1, both fault detection and diagnosis rates tested on the summer full-size dataset increased substantially for this CNN-GRU architecture trained on the mixed spring and summer dataset (Fig. 14a and b) while the false negative rate tested on the same season's dataset decreased considerably (Fig. 14c).

3.2.3. *Experiment 3 results: full-season quarter-size dataset evaluation*

Experiment 3 was implemented on a mixed dataset that contains a quarter of each season's dataset. Comparing the performances of two architectures in Table 6, the CNN-LSTM architecture outperformed the CNN-GRU architecture when tested under each season. Compared with the highest performance scores from the previous two experiments, the scores for the unseen seasons from the previous two experiments increased substantially in experiment 3, while the scores for the seen seasons from the previous experiments decreased slightly in experiment 3 due to the gradual decrease in the data amount of each seen season, i.e. full-size to half-size, and to quarter size across experiments 1–3. For example, the highest F1 score tested under spring decreased from 0.975 (Fig. 13a) to 0.942 (Fig. 13b), and further to 0.862 (Fig. 13b) for each best architecture trained on the spring, mixed spring and summer, and full-seasons quarter-size datasets, respectively.

By including the fault data under unseen seasons from experiments 1 and 2 into the training set of experiment 3, the architecture CNN-LSTM showed significant improvements in increasing fault detection and diagnosis rates (Fig. 14a and b), and mitigating false negative rate (Fig. 14c), when tested under the autumn and winter full-size datasets. In addition, the reduction of data amount for the seen seasons from the

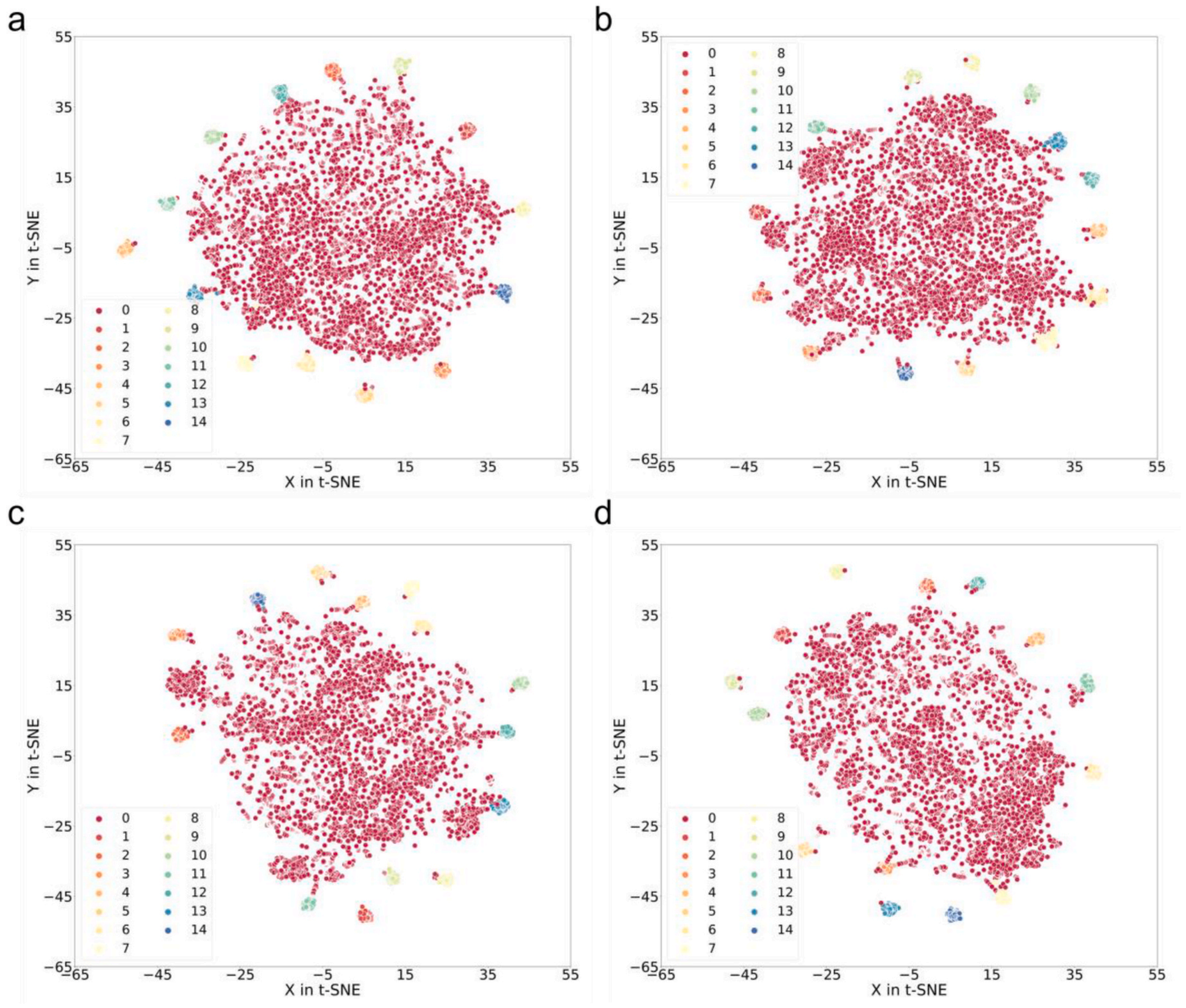


Fig. 18. Scatterplots for the hidden features extracted by the best architecture CNN-GRU tested under (a) spring; (b) summer; (c) autumn; (d) winter; in experiment 4 using the *t*-SNE method.

Table 8
Average fault detection rate, fault diagnosis rate, false positive rate and false negative rate among 10 cross-validation times for the best architecture in experiments 5–8 under the ASHRAE RP1312 dataset.

Test dataset	Fault detection rate	Fault diagnosis rate	False positive rate	False negative rate
Exp 5: CNN-LSTM trained on the spring datasetowhead				
Spring	0.9975	0.9994	0.0003	0.0025
Summer	0.6388	0.6297	0.0868	0.3612
Winter	0.8152	0.7877	0.2154	0.1848
Exp 6: CNN-GRU trained on the mixed spring-summer datasetowhead				
Spring	0.9910	0.9980	0.0039	0.0090
Summer	0.9978	0.9980	0.0015	0.0022
Winter	0.9520	0.7659	0.3073	0.0480
Exp 7: CNN-LSTM trained on the full-seasons quarter-size mixed datasetowhead				
Spring	0.9909	0.9978	0.0035	0.0091
Summer	0.9955	0.9974	0.0018	0.0045
Winter	0.9977	0.9992	0.0007	0.0023
Exp 8: CNN-LSTM trained on the full-seasons full-size mixed datasetowhead				
Spring	0.9953	0.9991	0.0013	0.0047
Summer	0.9998	0.9998	0.0009	0.0002
Winter	0.9999	1.0000	0.0005	0.0001

Table 9
Average fault detection rate, fault diagnosis rate, false positive rate and false negative rate among 10 cross-validation times for the best architecture in experiments 9–11 under the NREL's dataset.

Test dataset	Fault detection rate	Fault diagnosis rate	False positive rate	False negative rate
Exp 9: CNN-LSTM trained on the summer dataset				
Summer	0.9979	0.9925	0.0240	0.0021
Winter	0.8208	0.2158	0.4958	0.0021
Exp 10: CNN-LSTM trained on the winter dataset				
Summer	0.8317	0.2267	0.7892	0.1683
Winter	0.9983	0.9996	0.0153	0.0017
Exp 11: CNN-GRU trained on the mixed summer and winter dataset				
Summer	0.9554	0.9206	0.0608	0.0446
Winter	0.9933	0.9664	0.0250	0.0067

previous experiments slightly degraded fault detection rates (Fig. 14a), but had no prominent impact on fault diagnosis rates (Fig. 14b). According to the *t*-SNE visualization results, the boundaries among fault classes tested under autumn and winter (Fig. 17c and d) became apparent in comparison with those tested under the same seasons in experiment 2 (Fig. 16c and d). In addition, some fault clusters overlapped with the normal cluster tested under spring and summer, but most fault clusters are still separated from each other (Fig. 17a and b). As a result, this implies that the coverage of fault data under different seasons is more crucial in enhancing the FDD performances than the amount of fault data used for training under each single season.

3.2.4. *Experiment 4 results: full-season full-size dataset evaluation*

Experiment 4 was performed on a mixed dataset that contains full-

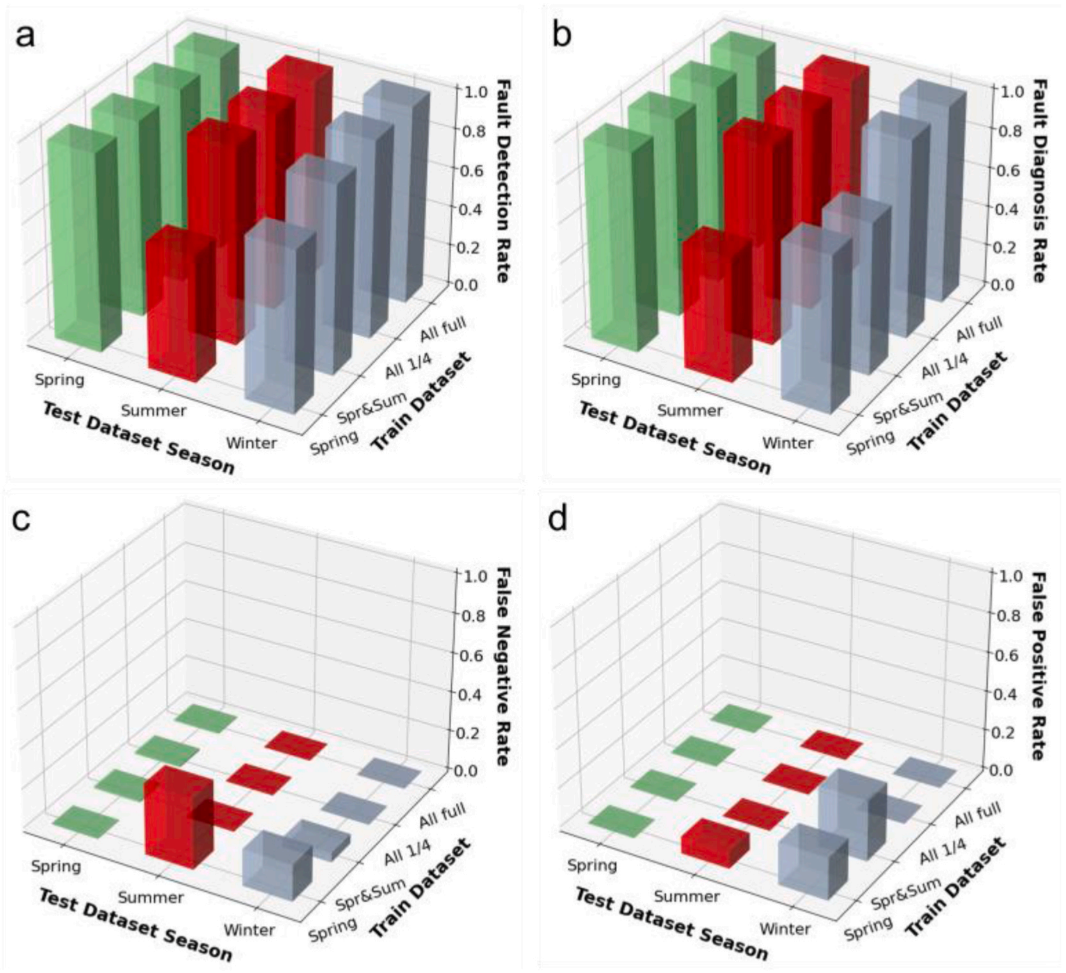


Fig. 19. Fault detection rate, fault diagnosis rate, false negative rate and false positive rate tested under each season for experiments 1–4 using the ASHRAE RP1312 dataset ('All 1/4' represents the full-season quarter size scenario; 'All full' represents the full-season full size scenario).

size datasets of each season. As presented in Table 6, the CNN-GRU architecture slightly outperformed the CNN-LSTM architecture when tested under each season. The highest F1 scores tested under each season are within the range of 0.96–0.98 (Fig. 13b), which demonstrates excellent FDD performances under all the seasons. Compared with the highest F1 scores from experiment 3, the F1 scores under this experiment increased by around 0.12 to 0.17 (Fig. 13b). With the increased amount of fault data under each season compared with experiment 3, the best architecture CNN-GRU in this experiment is capable of capturing the fault patterns under all the seasons. As shown in Table 7, the architecture indicated robust performances when tested under all the seasons, as the fault detection and diagnosis rates under each season are both around 1. Besides, both false negative and positive rates are less than 0.02 under all seasons. The *t*-SNE visualization results (Fig. 18) also demonstrate that the boundaries among fault classes are considerably apparent and all the fault classes' clusters are separated from the normal class cluster.

3.3. Identified practical issue verification using real fault datasets

To verify the findings obtained from the data experiments using the simulated fault dataset, data experiments 5–8 based on the ASHRAE RP1312 dataset and data experiments 9–11 based on the NREL's dataset were evaluated by the CNN-LSTM and CNN-GRU architectures. The same performance indicators as those used in experiments 1–4 are presented in Tables 8 and 9, respectively. Comparing the fault detection and fault diagnosis rates tested on each season's full-size dataset in experiment 5, the best architecture CNN-LSTM trained on the spring dataset performs well on the spring full-size dataset with both indicators of around 1, but shows lower values of both indicators tested under summer and winter (Fig. 19a and b). Similarly, in experiments 9 and 10, the fault detection and fault diagnosis rates tested under the unseen season are averagely 0.17 and 0.77 lower than those tested under the seen season. Apart from this, experiment 6 demonstrates that the fault detection rate tested under the unseen season, i.e., winter, is averagely 0.04 lower than that tested under the seen seasons, i.e., spring and summer, and the fault diagnosis rate under the unseen season is averagely 0.23 lower than that tested under the seen seasons. Therefore, the limited FDD ability for the faults from the unseen seasons is also observed in real fault data experiments.

However, a different observation in experiments 5–8 from experiments 1–4 is the variations in each performance indicator with the changes of fault data amount under spring are slight, as shown from the green bars in Fig. 19a–d. Besides, the decrease of fault data amount in the training set from full-size (exp. 9 and 10) to half-size (exp. 11) led to fault detection rate reductions of 0.04 and 0.01 and fault diagnosis rate reductions of 0.07 and 0.03 under summer and winter, respectively. Therefore, the FDD performances can be improved by increasing the amount of fault data under a season, but the improvements are also limited by other factors, such as the number of faults and the ratio of fault to normal data included in the training set. If the number of faults considered for the FDD task is low or the ratio of fault to normal data is high, the increase in the amount of fault data under a season may have a limited effect on the FDD performance enhancement.

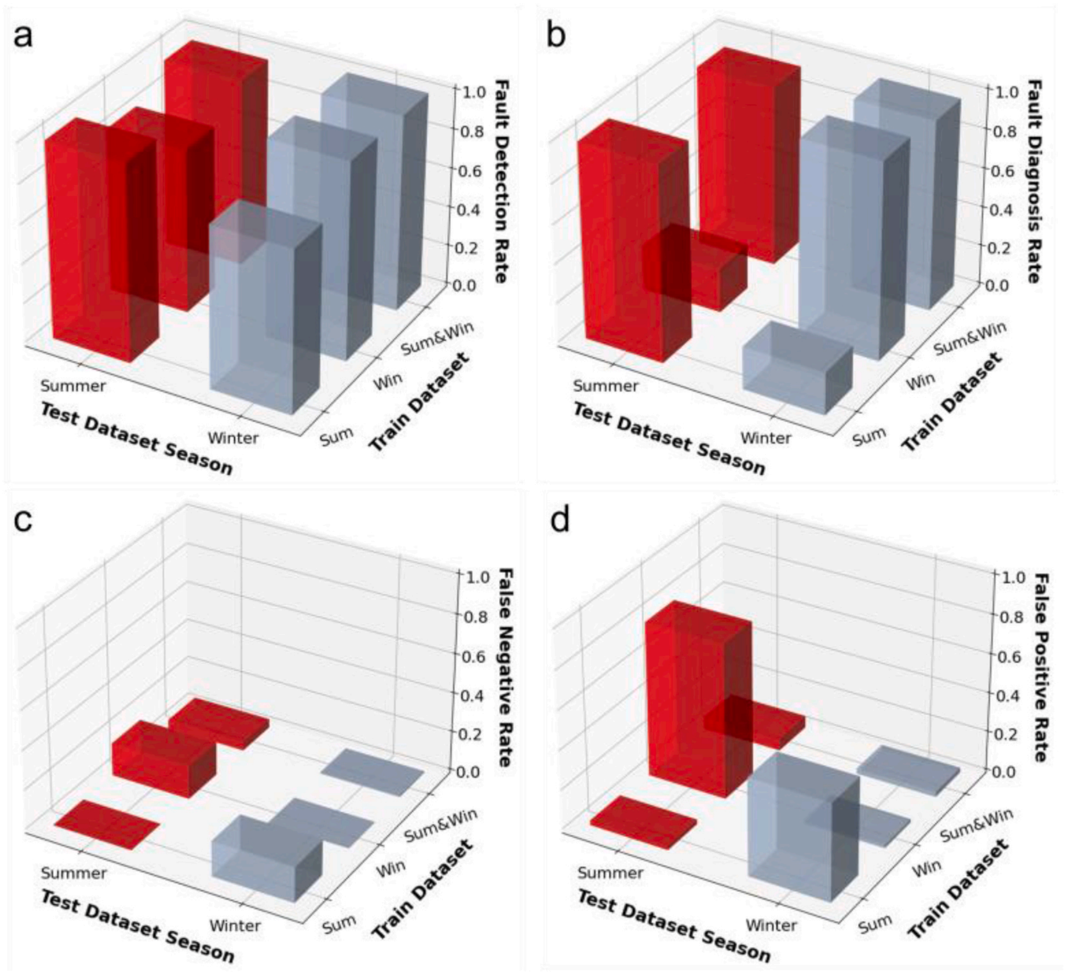


Fig. 20. Fault detection rate, fault diagnosis rate, false negative rate and false positive rate tested under each season for experiments 1–4 using the NREL's dataset.

Moreover, the FDD performance enhancement led by expanding the seasonal coverage of fault data is also prominent in real fault data experiments. As presented in Fig. 19a and b, the fault detection and fault diagnosis rates tested under the unseen seasons, i.e., summer and winter, for the best architecture trained on the spring only dataset (exp. 5) are lower than those tested under the same seasons for the best architecture trained on the full-seasons quarter dataset (exp. 7). Similarly, the best architecture trained on the mixed summer and winter dataset (exp. 11) indicated higher fault detection and diagnosis rates (Fig. 20a and b) and lower false negative and positive rates (Fig. 20c and d) than those tested under the unseen season, i.e., winter in experiment 9 and summer in experiment 10, for the best architecture trained on the summer only, and winter only dataset, respectively.

To summarize, as demonstrated by the simulated and real fault data experiments, the FDD architectures have limited generalization ability to detect and identify the faults from unseen seasons, and including fault data from a wider coverage of seasons into the training set has a significant impact on improving the FDD performances. In addition, the FDD performances can be improved by increasing the fault data amount under a season, but the improvements are affected by the number of faults and the ratio of fault to normal data included in the training set.

4. Conclusion and future works

This study evaluated the FDD performance differences of the combined CNN-RNN framework under the limited seasonal fault data scenarios and an ideal fault data scenario covering climatic conditions from four seasons. The fault and normal data were gathered from fault simulations using a verified prototype building EnergyPlus model and two fault datasets from real buildings. Four data experiments using the simulated fault dataset were proposed based on the practical issue related to the fault data season coverage and the amount of fault data within each season, in order to assess the FDD performances under different data scenarios. Moreover, two sets of further experiments based on each of the real fault datasets were implemented to verify the practical issue. The main findings of this study are as follows:

(1) The smaller window length is more robust in capturing the fault patterns, as the smaller window length leads to a larger number of windows fed into the FDD model training when there is a limited amount of fault data available for training.
(2) The FDD architectures trained on sufficient fault data under a certain season(s) show poor generalization ability to detect and identify the faults under the other unseen seasons.
(3) The FDD performance under a season can be enhanced by increasing the amount of fault data under this season, but the enhancement is limited by other factors, such as the number of faults and the imbalance ratio of fault to normal data included in the training set.
(4) The coverage of fault data under different seasons is more crucial in enhancing the FDD performances than the amount of fault data under a certain season(s).

Based on the findings of this study, the importance of fault data covered under different seasonal climatic conditions is highlighted. The results of this study will help guide FDD researchers to develop new data-driven FDD methods or revise existing methods considering this practical issue, which is highly intended to lead to FDD performance gaps between development tests and applications in real buildings. Future work can explore FDD performance differences tested on fault data under climatic conditions of the contemporary climate period and future climate periods. In addition, data-driven generative techniques can be employed to produce fault data under the seasons when the fault data is unavailable.

Credit author statement

Fangliang Zhong: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing - original draft, Writing - review & editing, **John Kaiser Calautit**: Conceptualization, Methodology, Project administration, Resources, Supervision, Writing - review & editing, **Yupeng Wu**: Supervision, Project administration, Writing - review & editing.

Declaration of competing interest

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

Data availability

The authors do not have permission to share data.

References

[1] International Energy Agency (IEA). World energy outlook 2020. Paris, France: IEA; 2020.
[2] United States Energy Information Administration (EIA). Commercial building energy consumption survey (CBECS) 2012. Washington, D.C., U. S.: EIA; 2016.
[3] United States Energy Information Administration (EIA). Residential energy consumption survey (RECS) 2015. Washington, D.C., U. S.: EIA; 2018.
[4] Wu S, Sun J. Cross-level fault detection and diagnosis of building HVAC systems. Build Environ 2011;46(8):1558–66.
[5] Lu X, Fu Y, O'Neill Z, Wen J. A holistic fault impact analysis of the high-performance sequences of operation for HVAC systems: modelica-based case study in a medium-office building. Energy Build 2021;252. 111448.
[6] Schein J, Bushby ST, Castro NS, House JM. A rule-based fault detection method for air handling units. Energy Build 2006;38(12):1485–92.
[7] Zhao Y, Li T, Zhang X, Zhang C. Artificial intelligence-based fault detection and diagnosis methods for building energy systems: advantages, challenges and the future, vol. 109. Renewable Sustainable Energy Rev.; 2019. p. 85–101.
[8] Mirnaghi MS, Haghighat F. Fault detection and diagnosis of large-scale HVAC systems in buildings using data-driven methods: a comprehensive review. Energy Build 2020;229:110492.
[9] Zhao Y, Li T, Fan C, Lu J, Zhang X, Zhang C, Chen S. A proactive fault detection and diagnosis method for variable-air-volume terminals in building air conditioning systems. Energy Build 2019;183:527–37.
[10] Manservigi L, Bahlawan H, Losi E, Morini M, Spina PR, Venturini M. A diagnostic approach for fault detection and identification in district heating networks. Energy 2022;251:123988.
[11] Liu J, Li G, Liu B, Li K, Chen H. Knowledge discovery of data-driven-based fault diagnostics for building energy systems: a case study of the building variable refrigerant flow system. Energy 2019;174:873–85.
[12] Liu B. Supervised learning. In: Web data mining. New York: Springer Publishing; 2011. p. 63–132.
[13] Hu Y, Chen H, Xie J, Yang X, Zhou C. Chiller sensor fault detection using a self-adaptive principal component analysis method. Energy Build 2012;54:252–8.
[14] Wu B, Cai W, Cheng F, Chen H. Simultaneous-fault diagnosis considering time series with a deep learning transformer architecture for air handling units. Energy Build 2022;257:111608.
[15] Sun K, Li G, Chen H, Liu J, Li J, Hu W. A novel efficient SVM-based fault diagnosis method for multi-split air conditioning system's refrigerant charge fault amount. Appl Therm Eng 2016;108:989–98.
[16] Liu J, Li G, Chen H, Wang J, Guo Y, Li J. A robust online refrigerant charge fault diagnosis strategy for VRF systems based on virtual sensor technique and PCA-EWMA method. Appl Therm Eng 2017;119:233–43.
[17] Bode G, Thul S, Baranski M, Muller D. Real-world application of machine-learning-based fault detection trained with experimental data. Energy 2020;198:117323.
[18] Du Z, Jin X, Yang Y. Fault diagnosis for temperature, flow rate and pressure sensors in VAV systems using wavelet neural network. Appl Energy 2009;86:1624–31.
[19] Liu Z, Liu Y, Zhang D, Cai B, Zheng C. Fault diagnosis for a solar assisted heat pump system under incomplete data and expert knowledge. Energy 2015;87:41–8.
[20] Allen WH, Rubaai A, Chawla R. Fuzzy neural network-based health monitoring for HVAC system variable-air-volume unit. IEEE Trans Ind Appl 2016;52(No. 3).
[21] LeCun Y, Bengio Y, Hinton G. Deep learning, Nature 2015;521:436–44.
[22] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions, proceedings of 2015 IEEE conference on computer vision and pattern recognition (CVPR). Boston, MA: United States; June , 15523970.
[23] Schuster M, Paliwal KK. Bidirectional recurrent neural networks. IEEE Trans Signal Process 1997;45(No. 11).
[24] Cheng F, Cai W, Zhang X, Liao H, Cui C. Fault detection and diagnosis for Air Handling Unit based on multiscale convolutional neural networks. Energy Build 2021;236(No):110795.
[25] Sasaki Y. The truth of the F-measure. 2007. https://www.cs.odu.edu/~mukka/cs795sum09dm/Lecturenotes/Day3/F-measure-YS-26Oct07.pdf.
[26] Eom YH, Yoo JW, Hong SB, Kim MS. Refrigerant charge fault detection method of air source heat pump system using convolutional neural network for energy saving. Energy 2019;187(No):115877.
[27] Fan, C., He, W., Liu, Y., Xue, P. and Zhao, Y., A novel image-based transfer learning framework for cross-domain HVAC fault diagnosis: from multi-source data integration to knowledge sharing strategies, Energy Build., Vol. 262, No. 111995, 2022.
[28] Li G, Chen L, Liu J, Fang X. Comparative study on deep transfer learning strategies for cross-system and cross-operation-condition building energy systems fault diagnosis. Energy 2023;263:125943.
[29] Zhang H, Li C, Wei Q, Zhang Y. Fault detection and diagnosis of the air handling unit via combining the feature sparse representation based dynamic SFA and the LSTM network. Energy Build 2022;269:112241.
[30] Taheri S, Ahmadi A, Mohammadi-Ivatloo B, Asadi S. Fault detection diagnostic for HVAC systems via deep learning algorithms. Energy Build 2021;250:111275.
[31] Shahnazari H, Mhaskar P, House JM, Salsbury TI. Modeling and fault diagnosis design for HVAC systems using recurrent neural networks. Comput Chem Eng 2019;126:189–203.
[32] Wang Z, Dong Y, Liu W, Ma Z. A novel fault diagnosis approach for chillers based on 1-D convolutional neural network and gated recurrent unit. Sensors 2020;20 (No. 2458).
[33] Qin N, Liang K, Huang D, Ma L, Kemp AH. Multiple convolutional recurrent neural networks for fault identification and performance degradation evaluation of high-speed train bogie. IEEE Transact Neural Networks Learn Syst 2020;31(No. 12).
[34] Canizo M, Triguero I, Conde A, Onieva E. Multi-head CNN–RNN for multi-time series anomaly detection: an industrial case study. Neurocomputing 2019;363: 246–60.
[35] Fan C, Li X, Zhao Y, Wang J. Quantitative assessments on advanced data synthesis strategies for enhancing imbalanced AHU fault diagnosis performance. Energy Build 2021;252:111423.
[36] Zhong F, Calautit JK, Wu Y. Assessment of HVAC system operational fault impacts and multiple faults interactions under climate change. Energy 2022;258:124762.
[37] Lu X, Fu Y, O'Neill Z, Wen J. A holistic fault impact analysis of the high-performance sequences of operation for HVAC systems: modelica-based case study in a medium-office building. Energy Build 2021;252:111448.
[38] U.S. Department of Energy (DOE). EnergyPlus version 9.3 engineering reference. Washington, D.C., United States: DOE; 2020.
[39] Deru M, Field K, Studer D, Benne K, Griffith B, Torcellini P, Liu B, Halverson M, Winiarski D, Rosenberg M, Yazdanian M, Huang J, Crawley D, U.S.. Department of Energy commercial reference building models of the national building stock. Golden, United States: National Renewable Energy Laboratory; 2011.
[40] Wen J, Li S. Tools for evaluating fault detection and diagnostic methods for air-handling units, American society of heating, refrigeration and air-conditioning engineers. Atlanta, United States: ASHRAE); 2011.
[41] Kim J, Leach M. Curated modeled fault data set. Golden, United States: National Renewable Energy Laboratory; 2019. https://doi.org/10.25984/1862681.
[42] Skansi S. Introduction to deep learning. Cham, Switzerland: Springer; 2018.
[43] Zhang, Y., Tennakoon, T., Chan, Y. H., Chan, K. C., Fu, S. C., Tso, C. Y., Yu, K. M., Huang, B. L., Yao, S. H., Qiu, H. H., Chao, C. Y. H., Energy consumption modelling of a passive hybrid system for office buildings in different climates, Energy, Vol. 239, No. 121914, 2022.
[44] Li Y, O'Neill Z. An innovative fault impact analysis framework for enhancing building operations. Energy Build 2019;199:311–31.
[45] Huang, S., Ye, Y., Wu, D. and Zuo, W. An assessment of power flexibility from commercial building cooling systems in the United States, Energy, Vol. 221, No. 119571, 2021.
[46] van Rossum G, Drake FL. *Python 3.6.9 reference manual*, CreateSpace, scotts valley. 2019.
[47] Cheung H, Braun JE. Development of fault models for hybrid fault detection and diagnostics algorithm. Golden, United States: National Renewable Energy Laboratory; 2015.
[48] Kim J, Frank S, Braun JE, Goldwasser D. Representing small commercial building faults in EnergyPlus, Part I: model development. Buildings 2019;9(No. 233).
[49] Integrated environmental solutions (IES), ApacheHVAC user guide. Glasgow, United Kingdom: IES; 2016.
[50] Buckberry HL, Bhandari M. ORNL MAXLAB flexible research platforms, proceedings of 2012 ACEEE summer study on energy efficiency in buildings. Pacific Grove, United States; August .
[51] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9: 1735–80.

[52] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y., Learning phrase representations using RNN encoder–decoder for statistical machine translation, arXiv:1406.1078vol. 3.

[53] Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift, proceedings of the 32nd international conference on machine learning. Lille, France; 2015. p. 448–56. July 7-9.

[54] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 2014; 15:1929–58.

[55] Li X, Chen S, Hu X, Yang J. Understanding the disharmony between dropout and batch normalization by variance shift. In: Proceedings of 2019 IEEE/CVF Conference on computer Vision and pattern recognition; 2019. Long Beach, United States, June 15-20, 2019, No. 19262863.

[56] Goodfellow I, Bengio Y, Courville A. Deep learning. Cambridge, Massachusetts, United States: the MIT Press; 2016.

[57] Gal Y, Ghahramani Z. A theoretically grounded application of dropout in recurrent neural networks, NIPS'16: proceedings of the 30th international conference on neural information processing systems. Barcelona, Spain; 2016. p. 1027–35. December 5-10.

[58] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature 1986;323:533–6.

[59] Ruder S. An overview of gradient descent optimization algorithms. 2016. arXiv preprint arXiv:1609.04747.

[60] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B. Scikit-learn: machine learning in Python. J Mach Learn Res 2011;12:2825–30.

[61] Cohen J. A coefficient of agreement for nominal scales. Educ Psychol Meas 1960;20 (No. 1):37–46.

[62] Frank S, Lin G, Jin X, Singla R, Farthing A, Granderson J. A performance evaluation framework for building fault detection and diagnosis algorithms. Energy Build 2019;192:84–92.