Anomaly Detection for Construction Vibration Signals using Unsupervised Deep Learning and Cloud Computing

Qiuhan Meng, PhD Candidate
Department of Civil and Environmental Engineering
The Hong Kong Polytechnic University
Hung Kong, China
Email: qiuhan.m.meng@connect.polyu.hk

Songye Zhu*, PhD, Professor
Department of Civil and Environmental Engineering
Research Institute for Artificial Intelligence of Things
The Hong Kong Polytechnic University
Hung Kong, China
Email: songye.zhu@polyu.edu.hk
*Corresponding author

Anomaly Detection for Construction Vibration Signals using Unsupervised Deep Learning and Cloud Computing

Qiuhan Meng[a] and Songye Zhu [a,b,*]

[a] Department of Civil and Environmental Engineering, The Hong Kong Polytechnic University, Hong Kong, China

[b] Research Institute for Artificial Intelligence of Things, The Hong Kong Polytechnic University, Hong Kong, China

ABSTRACT:

In-operation construction vibration monitoring records inevitably contain various anomalies caused by sensor faults, system errors, or environmental influence. An accurate and efficient anomaly detection technique is essential for vibration impact assessment. Identifying anomalies using visualization tools is computationally expensive, time-consuming, and labor-intensive. In this study, an unsupervised approach for detecting anomalies in construction vibration monitoring data was proposed based on a temporal convolutional network and autoencoder. The anomalies were autonomously detected on the basis of the reconstruction errors between the original and reconstructed signals. Considering the false and missed detections caused by great variability in vibration signals, an adaptive threshold method was applied to achieve the best identification performance. This method used the log-likelihood of the reconstruction errors to search for an optimal coefficient for anomalies. A distributed training strategy was implemented on a cloud platform to speed up training and perform anomaly detection without significant time delay. Construction-induced accelerations measured by a real vibration monitoring system were used to evaluate the proposed method. Experimental results show that the proposed approach can successfully detect anomalies with high accuracy; and the distributed training can remarkably save training time, thereby realizing real-time anomaly detection for online monitoring systems with accumulated massive data.

KEYWORDS:

Anomaly detection; cloud computing; distributed training; unsupervised deep learning; vibration-based monitoring

1. Introduction

Vibration-based monitoring is an effective technique for identifying structural or machine statuses through vibration responses. An in-operation vibration monitoring system is inevitably susceptible to sensor faults, system failure, and environmental influence. In this case, inaccurate values (often known as data anomalies) may be recorded, such as spikes, drift, and biased values. Moreover, some anomalies are related to emergencies, such as power and equipment failure. These anomalies can easily interfere with the analysis of actual vibration levels, leading to false assessment results and disturbing the alarm functions of vibration monitoring systems. For example, contractors are often required to suspend construction activities and take remedial actions, if the monitored vibration levels exceed prescribed control limits and alarm signals are issued. Frequent false alarms triggered by data anomalies instead of real vibration impact may considerably disturb the construction schedule and cause economic losses to construction projects. A reliable and efficient method for diagnosing data anomalies is crucial to ensure the serviceability of typical vibration monitoring systems so that users can respond to true alarms promptly and properly and prevent potential adverse impacts.

Numerous studies on detecting anomalies in machinery and structural vibration signals have been conducted [1-5]. In comparison, vibrations induced by construction activities are time-variant and nonstationary. For example, vibration amplitudes and frequencies vary considerably during different construction works (including no construction conditions). Extremely harsh environments and electromagnetic interference on construction sites may trigger signal anomalies more frequently, which may substantially interfere real-time alarm functions. These issues challenge the development of an accurate and reliable anomaly diagnosis for construction-induced vibration monitoring data.

The anomaly detection for time-series data has been previously realized by data visualization tools, wherein perceptible changes in waveform shapes or trends were considered abnormal behaviors [6]. The most common time-series visualization techniques include line charts, scatter plots, and maps [7]. Given that an appropriate overview of anomalous data requires expertise in data analysis and knowledge of the dynamic characteristics of vibration signals, human intervention and judgment bias are inevitable in this approach. Moreover, a full-scale monitoring system would produce vast amounts of data during its service life. Therefore, identifying data anomalies manually in real-time through visualization techniques is difficult.

The above limitation can be overcome using artificial Intelligence techniques. Various supervised machine learning methods have been successfully implemented for anomaly detection in vibration monitoring data. For example, Kerschen et al. [8] detected faulty sensor data through principal component analysis, and its efficiency was verified through an experimental application. Yang and Nagarajaiah [9] proposed a principal component pursuit algorithm to find noise outliers in the ambient vibration response of structural health monitoring (SHM) data. Gul and Catbas [10] used autoregressive models in conjunction with the Mahalanobis distance to detect outliers in the SHM data for different laboratory structures. Fu et al. [11] detected sensor faults by distributed similarity test based on the similarity of the power spectral density, and then trained an artificial neural network to classify three abnormal behaviors in wireless smart sensor networks.

Supervised deep learning (DL) methods, such as convolutional neural network (CNN) and long short-term memory (LSTM), have recently been proved effective for anomaly detection in vibration-based monitoring. Ni et al. [12] used a one-dimensional (1D) CNN to detect anomalies in SHM data. Zhang and Lei [13] proposed a CNN-based data anomaly detection method for vibration signals collected on a bridge. Li et al. [14] established a two-stage CNN model to identify anomaly patterns for a long-term SHM system. Liu et al. [15] used the LSTM network to identify sensor faults based on statistical features of the monitoring data. Lindemann et al. [16] employed LSTM and discrete wavelet transform to detect anomalies in vibration singles of various pumps. In addition to studies using time-series data, studies based on computer vision and DL have been reported in the latest publications [17, 18]. In particular, acceleration signals were converted into images in the data preprocessing stage and then used to train CNN for anomaly detection.

However, supervised learning approaches require a large amount of labeled data, and labeling all anomalous patterns for massive data is impractical, if not impossible [19]. Moreover, abundant normal data and limited anomalous data are often available from vibration monitoring systems. The imbalance in training data may result in the poor performance of various supervised learning approaches [20].

Alternatively, unsupervised learning methods, which typically train on abundant data without labels and use hidden feature representations to detect data anomalies, represent more useful and promising solutions.

The existing unsupervised methods for anomaly detection can be classified into one-class approaches, clustering analysis, and reconstruction-based methods [21]. One-class approaches, such as one-class support vector machine (SVM) [14, 22], assume that all training data belong to one class and distinguish anomalous data from normal data. However, one-class SVM performs poorly in cases when anomalies form clusters by themselves. It is also unsuitable for multi-class anomaly detection in large datasets. Clustering algorithms aim to separate data based on inner similarity. They assume that normal data belong to one specific cluster, whereas anomalous data should not belong to any cluster. The underlying mechanism of a clustering technique is to identify clusters instead of anomalies [23]. This technique does not perform well when data belong to one or more clusters (e.g., overlapping clusters or fuzzy boundaries). The main idea of reconstruction-based methods is that networks learn from normal data with few anomalies and make reliable predictions. Anomalous sequences deviating from normal data cannot be reconstructed identically because of the lack of training. Hence, anomalies can be identified through great discrepancies between input and output. The reconstruction errors are usually represented by distances or densities [21]. The commonly used indices include Euclidean distance [24], Mahalanobis distance [25], and maximum likelihood estimation [26]. A predetermined threshold is often applied to determine whether data are anomalous. Incorrect or unreliable anomaly threshold may cause false positive (i.e., the model indicates normal data as anomalous data) or false negative cases (i.e., the anomalous data are neglected by the model). Determining an appropriate threshold to identify anomalies remains an outstanding concern.

Autoencoder (AE) is a generally used DL method that can reconstruct data sequences in an unsupervised manner. Tremendous efforts have been devoted to adopting AE in damage detection [27-29] and defect detection [30, 31]. Mao et al. [32] proposed a combination of the generative adversarial network and AE to detect and eliminate anomalies in accelerations by encoding time series into images in advance. Nevertheless, outliers with very short durations are often visible in the time domain but may disappear in other domains [33], and thus it is difficult to select appropriate encoding methods suitable for different types of outliers in practical applications. Such time-to-image transformation requires extra operation and specialized knowledge and brings more operation complexity.

Moreover, high sampling-frequency data captured by in-operation monitoring systems are expected to be processed in real time. Increasing sensor numbers also lead to larger data streams. Such large data streams and complex DL models with many hyperparameters require high computational efficiency [34, 35], which is barely achievable by a single computing machine with limited hardware support. Besides, new anomalies that are unexpected in the training phase may be generated in practical situations, and thus the anomaly detection model should be continuously updated. There is a strong demand to deploy DL models to multiple machines for simultaneous training and computation. Research about distributed training has made remarkable achievements over the past decades. Xing et al. [36] presented a comprehensive summary of deployment, communication mode, and design principles of distributed learning platforms and algorithms. Zhang et al. [37] compared the scale and availability of three popular distributed learning platforms. Their performances were evaluated by an image classification task on the MNIST dataset. Verbraeken et al. [38] reviewed advanced machine learning algorithms and emphasized the current situation, application, and future development of distributed machine learning.

Cloud computing with distributed training has been developed to increase data scalability and reduce computational time. Cloud computing refers to a group of on-demand computing resources (typically storage and computing power) deployed on networked machines at different locations [39]. Both hardware and services require minor operation from an individual user. Users can access the cloud computing platform through the Internet by paying for the resources they use only, which can help reduce unnecessary hardware expenses. Early in 2005, Apache developed the Hadoop distributed system, wherein Hadoop distributed file system (HDFS) can provide expandable storage space for massive data and MapReduce realizes distributed computing. Recently, cloud computing has been implemented in several SHM applications. Yu and Lin [40] proposed a structural damage detection method based on MapReduce. Similarly, Cai and Mahadevan [41] improved the computation efficiency of SHM diagnosis through MapReduce and Apache Spark.

To the best of the authors' knowledge, real-time anomaly detection using unsupervised DL for practical vibration monitoring systems has never been reported in the previous literature. In light of this gap, the present study proposed a novel anomaly detection method for construction vibration monitoring signals by combining a temporal convolutional network (TCN) and AE. Compared with those existing methods in the literature, the TCN-AE model was trained in an unsupervised manner without any need for labeled training data, thereby alleviating data preparation requirements and eliminating human intervention. An adaptive threshold method for anomalies was adopted, and the optimal coefficient corresponding to the best performance was determined.

This paper is organized as follows. Section 2 introduces the background theory of AE, TCN, adaptive threshold with adjustment coefficient, and distributed training implemented in this study. Section 3 briefly introduces the vibration data source collected by a vibration monitoring system operated in a construction project in Hong Kong, the anomalous patterns in the training dataset, and the algorithmic setup. In Section 4, the anomaly detection results and the performance of distributed training are presented. Section 5 summarizes the major contributions of this study and directions for future work.

2. Anomaly Detection using Unsupervised Deep Learning and Cloud Computing

2.1. Overview of TCN-AE

2.1.1. Basic Framework of AE

AE has been widely used for dimension reduction [42] and denoising [43]. Fig. 1 shows the schematic framework of AE, consisting of an encoder network and a decoder network. The encoder transforms input x into a hidden representation $h(x)$, and the decoder maps the hidden vector to a reconstruction of input, denoted as $\hat{y}$. AE aims to minimize the reconstruction error between input and output vectors. This objective can be described as

$$\arg\min \sum_{i=1}^{n} \|x_i - \hat{y}_i\|^2, \tag{1}$$

where n is the number of samples. In this study, logcosh is used as a loss function to estimate the reconstruction error. The mathematical expression of logcosh is written as

$$\mathrm{logcosh} = \sum_{i=1}^{n} \log(\cosh(\hat{y}_i - x_i)). \tag{2}$$

The logcosh is the logarithm of the hyperbolic cosine of the prediction error, which is insusceptible to occasional outliers and has the advantages of robustness, no tunable hyperparameter, and fast convergence [44]. The previous literature [45] demonstrated that the logcosh loss function not only outperformed two typical loss functions, i.e., mean square error (MSE) and mean absolute error (MAE), but also improved the performance of the network.

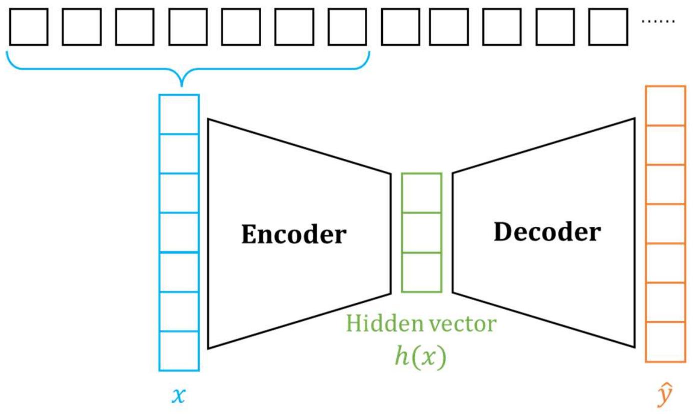


Fig. 1. Basic framework of AE.

2.1.2. Architecture of TCN

TCN was developed by Bai et al. [46] and successfully applied to anomaly detection in time-series data later [47]. Conceptually, TCN is based on a convolutional network with numerous advantages, such as few hyperparameters, strong adaptability, and computational efficiency. Unlike CNN, TCN uses casual convolution, dilated convolution, and residual blocks. Given an input time-series signal $x = (x_0, x_1, \ldots, x_T)$, an output sequence $y = (y_0, y_1, \ldots, y_T)$ is predicted at each time step. The casual convolution at time t $(t < T)$ is conducted based on the elements from time t and before, thereby preserving the order of sequential input data [46]. The convolution works as a sliding window that covers the input sequence and summarizes the weighted average of x in each time step. It can be defined as

$$y(n) = \sum_{i=0}^{k-1} x(n-i)w(i), \tag{3}$$

where y is the output of convolution, $w(i)$ represents convolutional filter weights, and k specifies the filter length. Zero padding is applied only on the left side of the input vector to realize casual convolution. As a result, the output signal has the same length as the input.

The dilated convolution adds a parameter called dilation rate d [48], where dilation rate d refers to the interval between elements in the input signal used for output sequence computation. For a network with M layers, the dilation rate is generally set to $d = 2^{M-1}$. The dilated convolution can be written as

$$y(n) = \sum_{i=0}^{k-1} x(n-di)w(i). \tag{4}$$

Fig. 2 shows an example of dilated casual convolution with a dilation rate $d = (1,2,4)$ and a kernel size of $k = 3$. If $d = 1$, the regular convolution operation is obtained. In the stacked architecture, long-term temporal information is learned by the network with the increasing dilation rate.

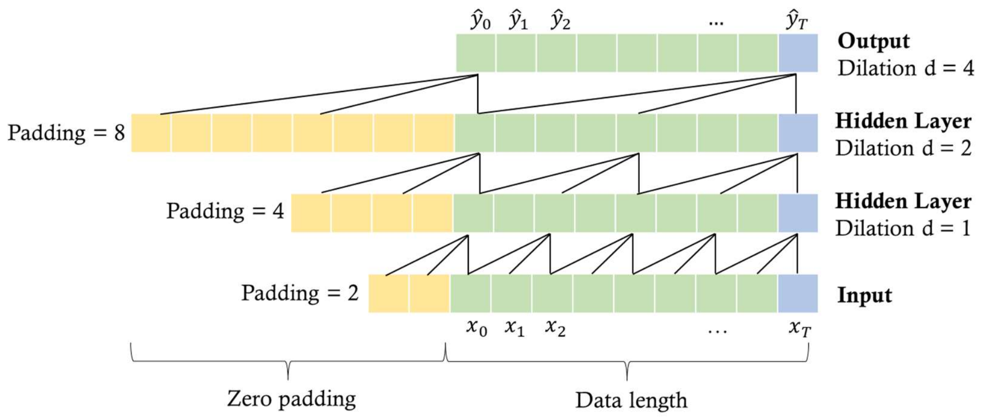


Fig. 2. Dilated casual convolution with dilated rates $d = (1,2,4)$ and kernel size $k = 3$.

As presented in Fig. 3, the structure of the baseline TCN is a stack of residual blocks. A residual block contains two subblocks, each comprising a dilated convolution layer, a weight normalization layer, an activation layer, and a dropout layer sequentially. Furthermore, an additional 1×1 convolution layer, known as a skip connection, is applied to ensure the same input and output dimension of each residual block.

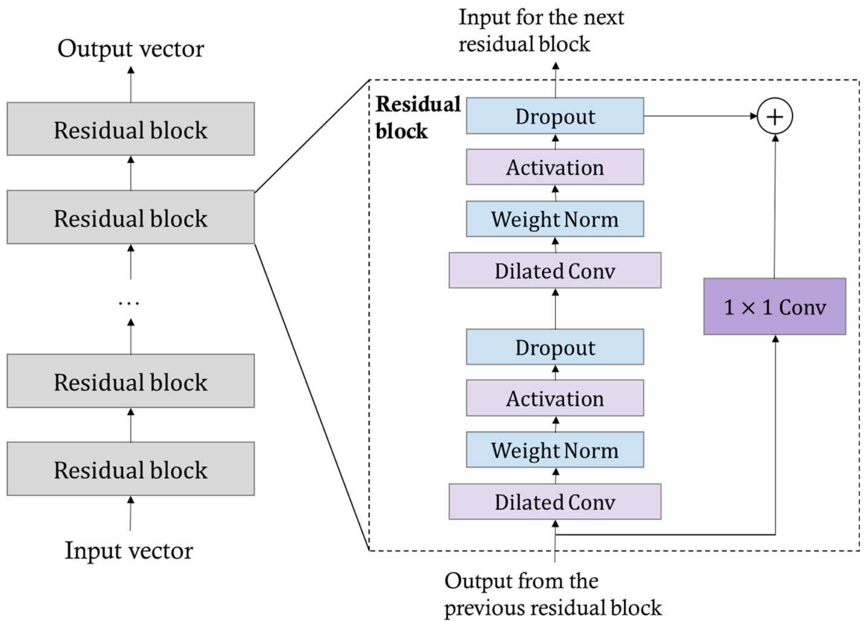


Fig. 3. TCN residual blocks.

2.1.3. Architecture of TCN-AE

Thill et al. [49] developed TCN-AE, which combines an AE and a novel TCN model. Fig. 4 demonstrates the architecture of TCN-AE with dilated rates $d = (1, 2, 4)$. The input time series of length T passes through a TCN-encoder with three dilated convolutional layers. The red box represents the kernel of the dilated convolution. Each dilated convolutional layer is followed by a skip connection of a 1×1 convolution. The outputs of the skip connections are concatenated and then compressed as an encoded representation. Afterward, the encoded representation is downsampled by a pooling layer, generating the final output of the TCN-encoder. In the TCN-decoder, the input is upsampled to the original length T. Similar to the TCN-encoder, a stack of three dilated convolutional layers followed by skip connections is applied to the data sequence after upsampling. The reversed dilated rates $d = (4, 2, 1)$ are used for the TCN-decoder. Finally, the output convolutional layer with an activation function reconstructs the representation sequence to a decoded time series.

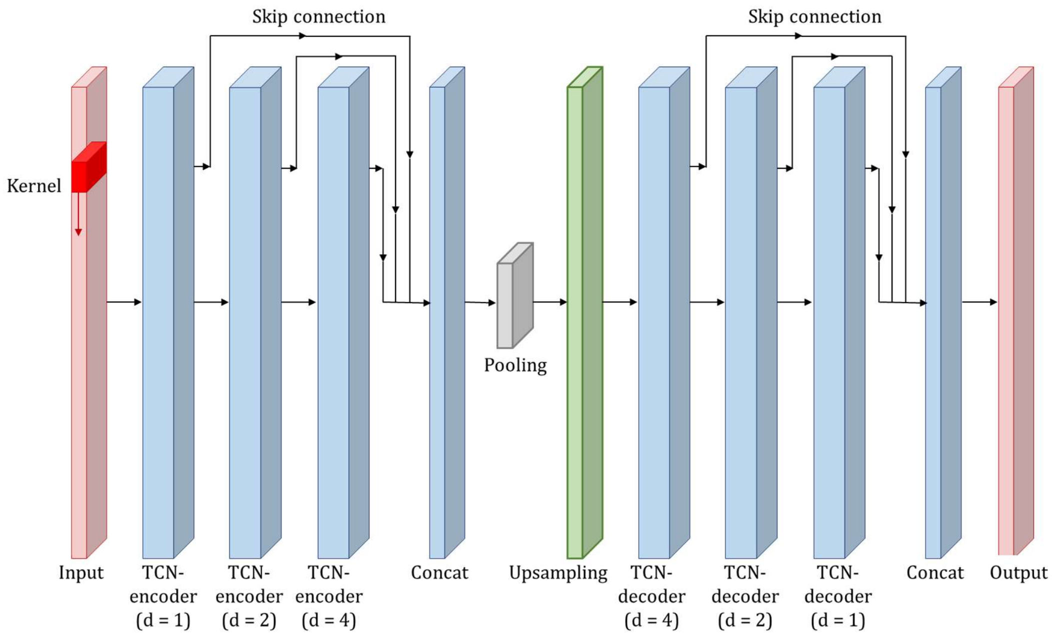


Fig. 4. Architecture of TCN-AE with dilated rates $d = (1, 2, 4)$.

2.2. Adaptive Threshold with Adjustment

After the TCN-AE is trained, reconstruction error is used as an indicator for judging data anomalies. As mentioned, the TCN-AE aims to minimize the difference between the input and output signals. The TCN-AE is trained on normal data sequence with few anomalies in this study. Consequently, high reconstruction errors are expected when anomalies exist in the data sequence. Minor differences correspond to normal data, whereas huge differences indicate abnormal events. The learning and reconstruction processes operate in unsupervised manners.

The reconstruction errors do not have a constant baseline. Thus, an adaptive threshold was applied to find the optimal threshold for anomalies automatically. An anomaly is usually a temporal pattern rather than one exact point of the signal; therefore, a rolling window of length l slides over the error sequence to generate an error matrix $E = (e_0, e_1, \dots, e_{T-l+1})$. The reconstruction error between input x and output $\hat{y}$ is calculated by the Euclidean distance

$$e = \sqrt{\sum_{i=0}^{n} (x_i - \hat{y}_i)^2}. \tag{5}$$

The length of the rolling window is set to 500 in this study, which corresponds to 2 s for each window. In order to smoothen the noise and erroneous events that may occur abruptly, a Gaussian filter is applied to the error matrix.

The occurrence of anomalies is decided by an anomaly score. It is calculated according to the log-likelihood of the error matrix, which can be expressed as

$$a_s = L(\theta|e) = \prod_{i=0}^{n} f(e_i;\theta) = \sum_{i=0}^{n} \ln f(e_i;\theta), \tag{6}$$

where θ is the parameter of the probability distribution, and f is the density. Given the complexity of time-series data, the kernel density estimation (KDE) is utilized to estimate the probability density [50], which can be expressed as

$$\hat{f}_h(e) = \frac{1}{n}\sum_{i=0}^{n} K\left(\frac{e - e_i}{h}\right), \tag{7}$$

where K is the kernel function, and h is the bandwidth. The Gaussian kernel is selected in this study because it can provide the density distribution with optimal smoothness [51].

The adaptive threshold is adjusted dynamically based on the anomaly score. If the anomaly score is larger than the threshold, the data within the time window will be classified as anomalies. The threshold is set based on Harrell-Davis (HD) quantile estimator [52] and median absolute deviation (MAD) approach [53]. The HDMAD is defined as

$$HDMAD = \mathrm{c} \times Q_{HD}(p = 0.5)\{|a_{s_i} - M_{HD}|: 1 \leq i \leq n\}, \tag{8}$$

where M_{HD} is the HD median of the anomaly score, and c is a constant term of 1.4826, suggested by Simmons et al. [54]. With this approach, the anomaly threshold can be calculated by

$$T = M_{HD} - \alpha \times HDMAD, \tag{9}$$

where α is the adjustment coefficient that usually equals 3. In this study, the value of α is adjusted between 2 to 5 [53], and the adjustment step is set to 0.4 each time. The coefficient that maximized the F_1-score is taken as the optimal coefficient.

The calculation of the F_1-score is derived from true positive (TP), false positive (FP), and false negative (FN). A binary anomaly flag [0, 1] is determined for each time series point, in which flag = 1 means anomalous data while flag = 0 means normal. Therefore, TP indicates that anomalous data are correctly identified by the adaptive threshold, while FP indicates that normal data are incorrectly detected as anomalies, and FN indicates missed detection of anomalous data. On the other hand, true negative (TN) indicates other normal data points are detected as normal. Noted that labels are not passed to the DL model at any time but are only used for calculating the F_1-score.

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{12}$$

2.3. Distributed Training

Distributed deep neural networks mainly focus on the process of training in parallel. Distributed training has two methods: data parallelism and model parallelism. Model parallelism is applied to a model too large to fit into a single machine, whereas data parallelism is appropriate for extremely large datasets to achieve fast training. Fig. 5 shows the distributed training paradigm of data parallelism. Before training begins, a complete model is replicated on all GPUs. The training dataset is divided into several pieces with the same number of GPUs. Each GPU trains a replica of the model locally and computes the gradient of the loss function on a portion of the training data. An efficient ring-allreduce strategy is used to average gradients and communicate gradients among GPUs to reduce communication costs [55]. Then, the model is updated synchronously until it converges. Despite the simple and straightforward concept of the data parallelization approach, the implementation is rather complicated. Distributed learning is achieved in this study by adopting an open-source toolkit, Horovod [55].

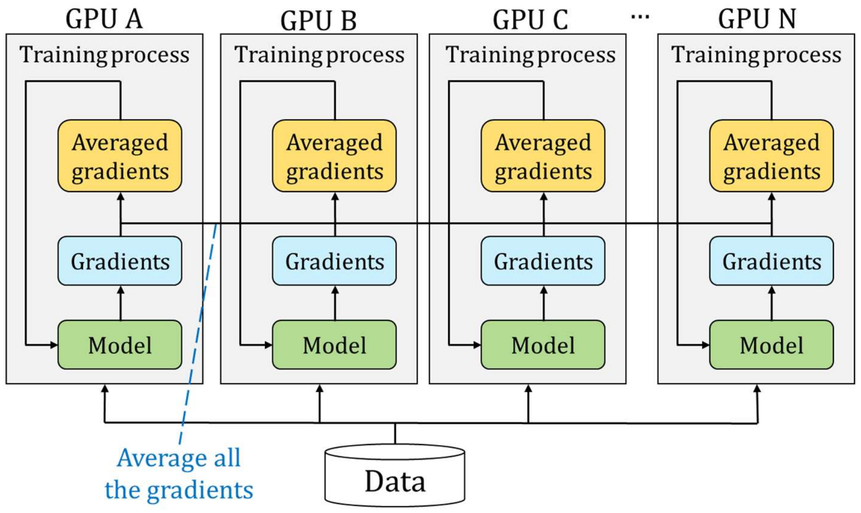
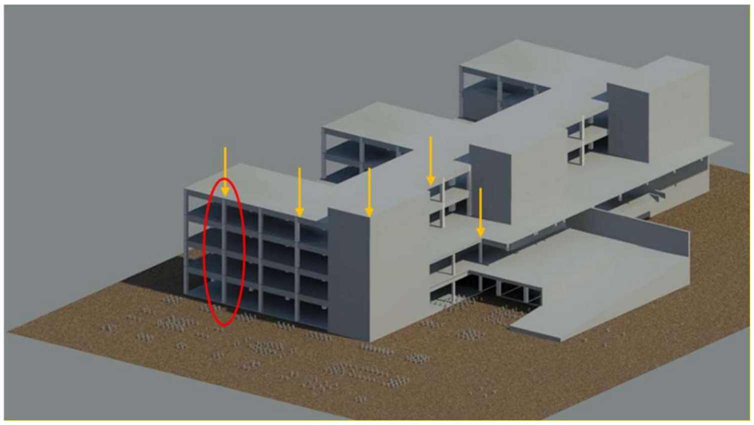


Fig. 5. Data parallelism approach.

3. Experimental Setup

3.1. Construction-induced Vibration Database

A real-time sensory system for construction-induced vibration impact monitoring was deployed on a selected construction site in Hong Kong, where foundation works involving mini piling were conducted. As shown in Fig. 6(a), five monitoring points were installed on five columns in the existing adjacent building to monitor the vibration impact in the most influenced area. Three high-fidelity accelerometers (KD1000) were installed at each monitoring point for triaxial vibration measurement. The accelerometers have a sensitivity of 10 mV/g, a measurement range of 500 g, and a sampling frequency of up to 10 kHz. Each accelerometer was equipped with a magnetic base. As shown in Fig. 6(b), three accelerometers were mounted on a steel block through magnetism in three orthogonal directions. The steel block was attached to the column through the epoxy resin. A fixed monitoring station was deployed inside the building, which comprised other measurement units such as a data logger (NI USB-6343), a charge amplifier (KD500), and a computer.



(a) (b)

Fig.6. Installation of accelerometers and five measurement points. (a) Five measurement points, (b) Three accelerometers mounted on a steel block.

The acceleration was recorded with a sampling frequency of 250 Hz, which can fulfil the requirements of measuring vibrations induced by construction activities. The monitoring system has 15-channel of acceleration signals, corresponding to 13,500,000 data points every hour. The data in the vertical direction of point 1 (red circle) were used for illustration in this study. Notably, the measured vibrations with and without construction activities might differ by order of magnitude. Even the construction-induced vibrations attenuated rapidly with increasing separation distances from vibration sources. Therefore, the collected vibration data varied considerably in a wide range. All data were thus normalized by scaling into the range between zero and one before the training. The normalization of the data can eliminate the measurement distance effect but preserve other information and relationships of the original input data, enhancing model accuracy and convergence speed [56]. The equation of min-max normalization is given as

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{13}$$

where x is the original data and x' is the normalized data.

Previous studies have shown that anomaly detection can be realized with AE trained on normal data [57, 58]. One hour of acceleration with few anomalies was used as training data, corresponding to 900,000 data points. The training samples were extracted using a 10 s window with 90% overlap. In each training epoch, 10% of training data that was selected randomly was used as the validation set. The test dataset contained another set of one-hour acceleration data collected at different time, which was deliberately selected to ensure that it contains three types of outliers, namely, spike, shift, and trend.

Fig. 7 demonstrates the normal monitoring data with and without ongoing construction works. Great amplitude changes can be observed, which indicates that the vibration data generated by construction activities is time-variant. Three types of anomalies exist in the monitoring data. Fig. 8(a) shows the spike type, a sudden change in the measurement value, corresponding to high-frequency large-amplitude vibration contents. It is one of the most common data anomalies in the construction-induced vibration monitoring system. Another common anomalous pattern, namely shift, is shown in Fig. 8(b), which might be caused by systematic bias. Typical acceleration data should fluctuate around the zero amplitude of the monitored object. As shown in Fig. 8(c), the noticeable monotonic upward or downward changes in the time domain can be considered as a trend type, which may be attributed to sensor faults.

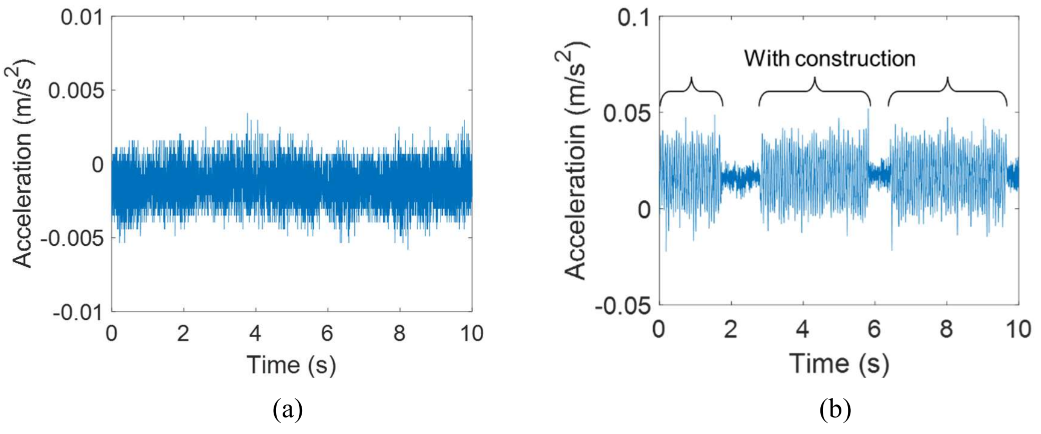


Fig. 7. Examples of normal data. (a) Without construction activity, (b) With construction activity.

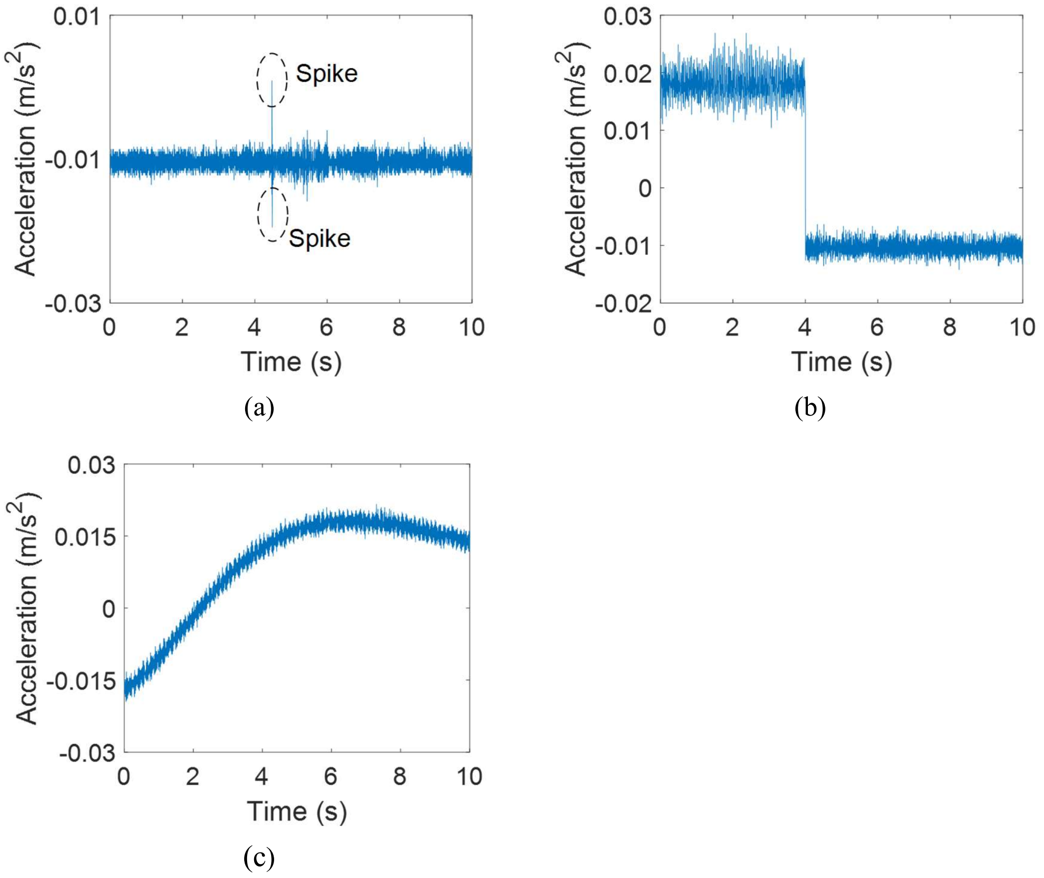


Fig. 8. Examples of data anomaly. (a) Spike, (b) Shift, (c) Trend.

3.2. Algorithmic Setup

All experiments were conducted on the MatPool cloud platform instead of on a single computer used on-site. Each virtual machine is configured with Ill(IXeon(R) E5-2678 v3 CPU @ 2.50 GHz, 30 GB of memory, and NVIDIA RTX A2000 GPUs. The number of GPUs varied between 1 and 4. The Keras [59] and TensorFlow [60] frameworks were adopted in the experiments. Fig. 9 shows an overview of the framework of TCN-AE with its parameters. The one-dimensional input acceleration data of length T was passed to a stack of $L = 5$ dilated convolution layers with $n_{filters} = 32$ filters of size $k = 5$, resulting in a feature map with dimension $T \times 32$. The dilation rates $d = 1, 2, 4, 8, 16$ were used for the encoder. Each dilated convolutional layer was followed by a 1×1 convolution with $n_{filters} = 5$ filters, which reduced the dimension to $T \times 5$. Afterward, an average pooling layer with a pool size $s = 8$ was set to downsample the output of the encoder. Subsequently, the compressed hidden vector was passed to the decoder and then upsampled to the original length T. The TCN architectures of the encoder and decoder were the same, and the dilation rates $d = 16, 8, 4, 2, 1$ were used for the decoder.

During the training process, 10% of the training data was used for validation. The Adam optimizer proposed by Kingma and Ba [61] was applied with an initial learning rate of 0.002. Other parameters of the model include linear activation and Glorot normal weight initializer [62]. The batch size was set to 128 for the model. It was trained over 50 epochs. An early stopping with the patience of 5 epochs was applied to avoid the overfitting problem of the model.

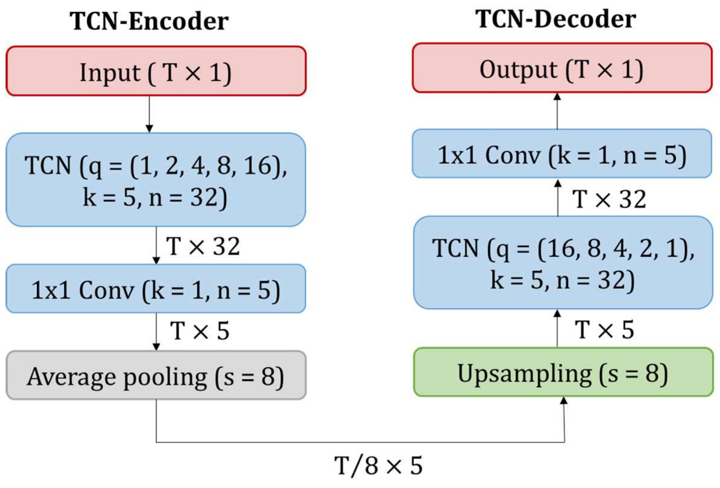


Fig.9. Framework of TCN-AE with its parameters

4. Results and Discussion

4.1. Anomaly Detection with Adaptive Threshold

Fig. 10 demonstrates acceleration signals with and without construction activities reconstructed by the TCN-AE. In this case, TCN-AE could successfully reconstruct the normal signal. Moreover, it was not affected by the state of ongoing construction activity. In contrast, TCN-AE had difficulties in reconstructing the accelerations with anomalies, as shown in Fig. 11.

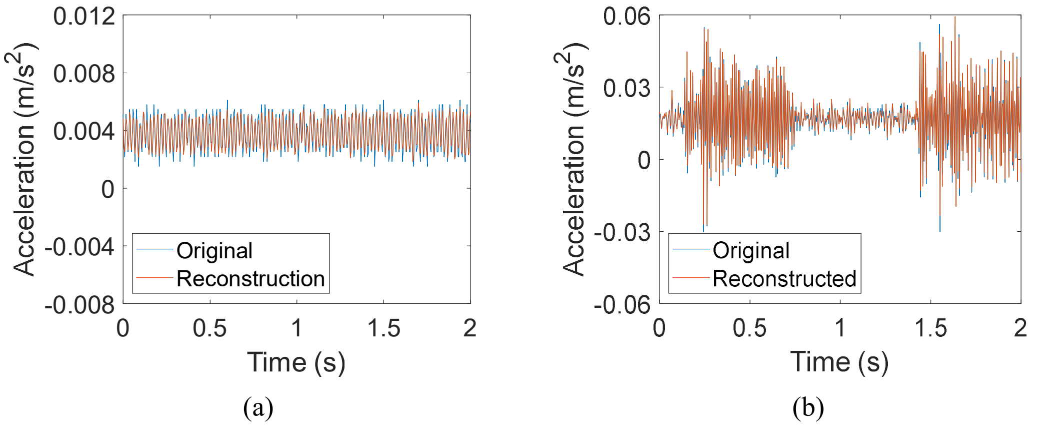


Fig. 10. Normal data reconstructed by the TCN-AE. (a) Without construction activity, (b) With construction activity.

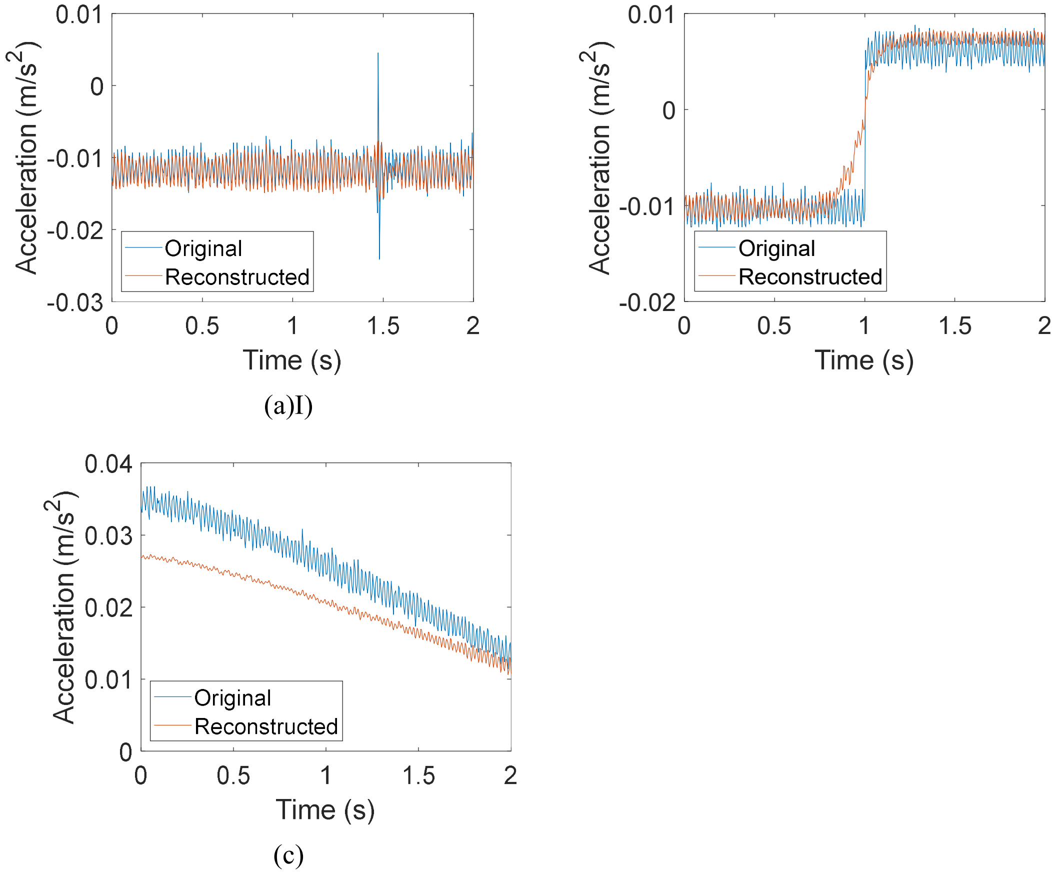


Fig. 11. Anomalous data reconstructed by the TCN-AE. (a) Spike, (b) Shift, (c) Trend.

The log-likelihood value of Euclidian distance was calculated as the anomaly score. Given the complexity and noise in the measured acceleration, KDE was used to estimate the distribution of anomaly scores. Fig. 12 shows the KDE estimated distribution with Gaussian kernel and different bandwidths. The blue curve is the histogram of anomaly scores. The comparison shows that the densities estimated by KDE with bandwidth $h = 0.2$ and $h = 0.5$ are oversmoothed by hiding much of the underlying structure, whereas the KDE with bandwidth $h = 0.05$ is considered optimally smoothed because its density is close to the true density. Consequently, the bandwidth for the following experiments was set to 0.05.

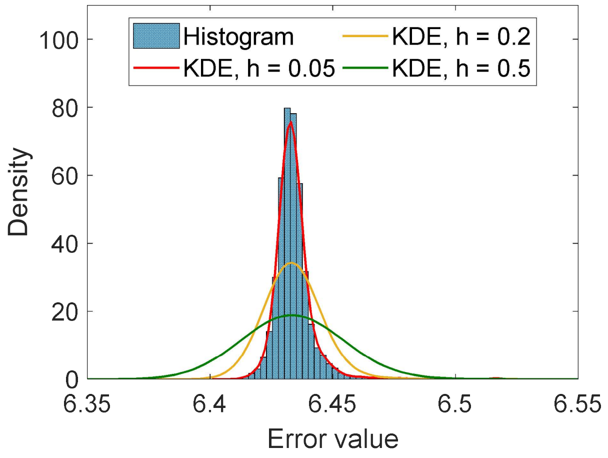


Fig.12. KDE estimated error distribution with different bandwidths.

High anomaly scores are considered anomalies. The adjustment of the optimal threshold was performed by varying coefficient α from 2 to 5. By setting $\alpha = 2$, the anomaly detection obtained an F_1-score of 0.8798. The α increased by 0.4 each step and stopped at $\alpha = 5$. The highest F_1-score equals 0.9427 with an optimal coefficient of 3.2. Fig. 13 depicts the anomaly score and adaptive thresholds when $\alpha = 3.2$. Table 1 summarizes the coefficients and model performances in terms of precision, recall, and F_1-score at each step.

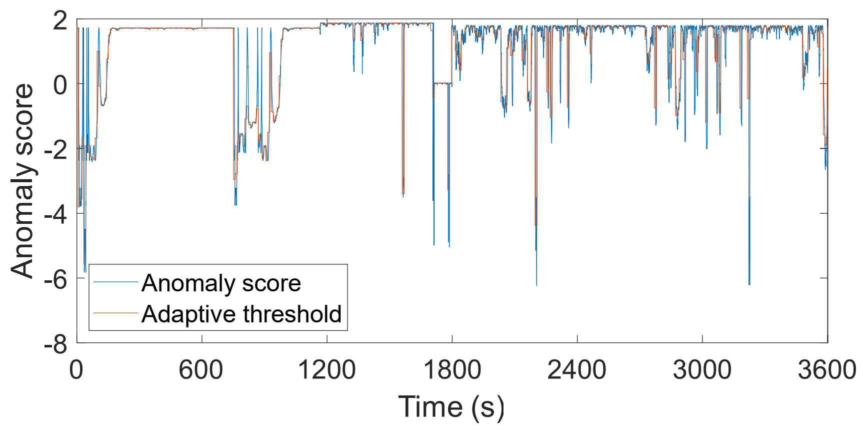


Fig. 13. Adaptive threshold ($\alpha = 3.2$) and anomaly score.

Table 1
Coefficient and performance for anomalies detection.

Step	α	Precision	Recall	F_1-score
1	2.0	0.8797	0.8798	0.8798
2	2.4	0.9296	0.9403	0.9185
3	2.8	0.9317	0.9437	0.9313
4	3.0	0.9372	0.9437	0.9319
5	**3.2**	**0.9475**	**0.9539**	**0.9427**
6	3.6	0.9417	0.9478	0.9374
7	4.0	0.9391	0.9363	0.9374
8	4.4	0.9116	0.8933	0.9024
9	4.8	0.8879	0.9190	0.8497
10	5.0	0.8436	0.8467	0.8389

As shown in Fig. 14, most of the anomalies were successfully identified by the adaptive threshold, in which red shaded areas represent anomaly windows containing anomalies. Fig. 15 demonstrates examples of FN and FP cases. Specifically, FN indicates missed detection of subsistent anomalies. As demonstrated in Fig. 15(a), the spile anomaly has not been detected. On the contrary, there is no anomaly in the shaded area shown in Fig. 15(b), which is counted as FP. The precision of all cases is lower than the recall, which indicates that for most cases, normal data are incorrectly presumed to be anomalous.

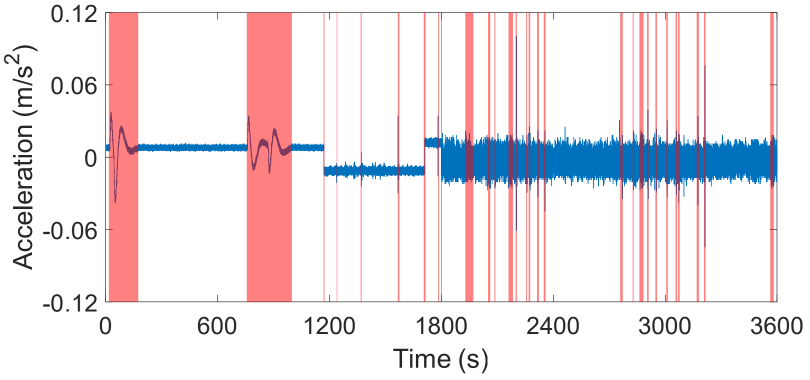
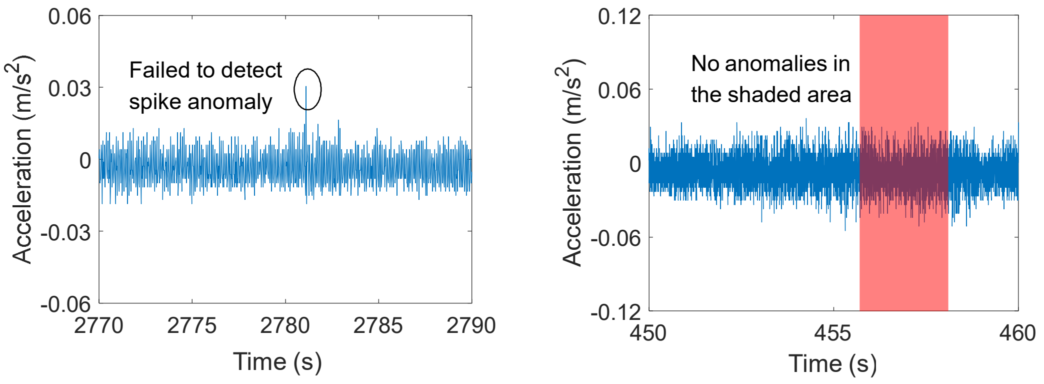


Fig. 14. Acceleration with the detected anomalies marked by the red shaded area.



(a) (b)

Fig. 15. Examples of FN and FP cases. (a) FN, (b) FP.

4.2. Performance of Distributed Training

The computation efficiency of the TCN-AE is another crucial concern directly affecting the online application feasibility. In this study, distributed training was conducted with Horovod on an hour of acceleration data from one channel (900,000 data points) for 20 epochs. The configurations of all cases are summarized in Table 2. In particular, the computing time was compared among four distributed strategies: a single GPU (case 1), a stand-alone cluster with four GPUs (cases 2 and 3), a four-node cluster with one GPU each (case 4), and a two-node cluster with two GPUs each (case 5). Direct training was performed in case 1, given that a single GPU was used.

Table 2
Configurations of distributed training.

	Machines	GPUs/Machine	Batch size/GPU	Global batch size	Learning rate
Case 1	1	1	128	128	0.002
Case 2	1	4	32	128	0.002
Case 3	1	4	128	512	0.004
Case 4	4	1	32	128	0.002
Case 5	2	2	64	128	0.002

The comparison was based on a batch size of 128, corresponding to 320,000 data points per iteration. Two alternatives were considered for the training on the stand-alone cluster with four GPUs. Case 2 had a global batch size of 128 distributed over four GPUs, which resulted in a batch size of 32 for each GPU. The batch size of 128 was set to each GPU in case 3, thereby obtaining a global batch size of 512. The learning rate set in case 3 was doubled to compensate for the large batch size. The batch size per GPU was set to 32 and 64 in cases 4 and 5, respectively.

Fig. 16 compares the training time of each case. The comparison between case 1 and other cases shows that the training progress runs faster with more GPU counts. Case 3 with a global batch size of 512 consumes less time than case 2 with a global batch size of 128, indicating that a large batch size would reduce the number of iterations required to complete an epoch of training. Besides, GPUs were deployed on four and two machines in case 4 and case 5, respectively. Compared with training on one machine, the training time was not affected by cross-communication between different machines, thus validating the effectiveness of the adopted distributed training strategy.

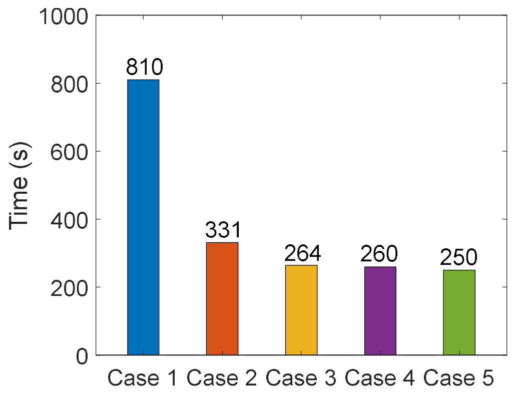


Fig. 16. Comparison of training time.

The experimental results indicate that processing the 1 h single-channel acceleration took 250 s by distributing the training on the two-node cluster with two GPUs each (case 5), which corresponded to 4 s for training on 1 min of single-channel data. When applied to the online vibration monitoring system, the newly generated data would be used to keep the model up to date. For the construction-induced vibration system containing 15 channels (as described in Section 3.1), training the model on all data generated in 1 min takes approximately 60 s, which realizes the real-time training on data of multiple channels. Furthermore, a well-trained model took a very short time to make predictions; thus, the anomaly detection was almost synchronous with model training. If the number of sensors (i.e., channels) is further increased, the training time can be further reduced by employing an increased number of GPUs or machines to satisfy real-time anomaly detection.

Fig.17 compares the training and validation losses measured by logcosh for all cases. Both the training and validation losses greatly depended on batch size. For a fixed number of epochs, case 3 with the largest global batch size of 512 converged at the highest training and validation losses, whereas other cases trained on the same global batch size converged toward a similar minimum. This finding indicates that case 3 required more epochs to converge to an optimal minimum value. Overlarge batch size would reduce the generalization capability of the model [63]. Even though increasing the learning rate can compensate for large batch sizes, the effect of the two-fold increased learning rate was not apparent in case 3.

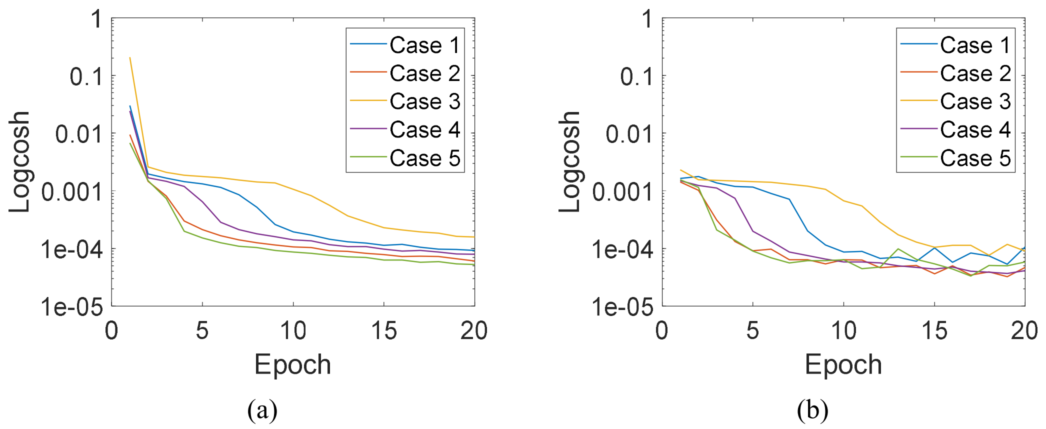


Fig. 17. Training and validation losses for all distributed training cases. (a) Training, (b) Validation.

5. Conclusions

In this study, a novel unsupervised DL approach for detecting data anomalies in construction vibration monitoring signals was proposed based on TCN-AE. Compared with most existing anomaly detection approaches, TCN-AE reconstructed time-series data in an unsupervised manner, and no additional labels were required in the training process. Anomaly scores were defined by the log-likelihood values of the reconstruction errors between input vibrations and reconstructed output. Data with larger anomaly scores were detected as anomalies. The anomaly threshold was defined by HD quantile with MAD, and the adaptive threshold method with an adjustment coefficient was adopted. Moreover, distributed training, which has not been implemented for anomaly detection for vibration-based monitoring systems in previous literature, was adopted. Major conclusions are summarized as follows.

The construction vibration signals from the monitoring system of a construction project were utilized to validate the efficacy of the proposed method. The experiments show that the TCN-AE could reconstruct normal data successfully. However, reconstruction errors became large when anomalous data existed. An adaptive threshold was applied for anomaly detection, and the experimental results show that a coefficient of 3.2 can achieve the best detection performance (the optimal F_1-score equals 0.9427). By utilizing distributed training on 2 GPUs over 2 virtual machines, the computation efficiency was significantly improved by 224%, as expected. In specific, computational time was reduced from 810 s to 250 s when training one-hour data, which corresponded to 900,000 data points. It is possible to extend the proposed distributed training method to real-time anomaly detection for many channels of data.

Noted that this study regarded all types of anomalous data as anomalies. The classification of different types of anomalies has not been achieved. In addition, the vibration responses in this study contained very few anomalies. More comprehensive experiments should be conducted in future research to examine the effectiveness of the proposed method when applied to vibration data containing more types of anomalies or high anomaly ratios.

Funding Acknowledgment

The authors are grateful for the financial support from the Hospital Authority of Hong Kong, the Research Grants Council of Hong Kong (Grant Nos. R5020-18, and T22-502/18-R), the Hong Kong Branch of the National Rail Transit Electrification and Automation Engineering Technology Research Centre (Grant No. K-BBY1), and the Hong Kong Polytechnic University (Grant Nos. ZE2L, ZVX6). The first author is also grateful for the financial support from the Hong Kong Polytechnic University through the GBA Startup Postdoc Programme 2022.

References

[1] K. Vos, Z. Peng, C. Jenkins, M.R. Shahriar, P. Borghesani, W. Wang, Vibration-based anomaly detection using LSTM/SVM approaches, Mechanical Systems and Signal Processing, 169 (2022) 108752.

[2] X. Xu, Y. Ren, Q. Huang, Z.-Y. Fan, Z.-J. Tong, W.-J. Chang, B. Liu, Anomaly detection for large span bridges during operational phase using structural health monitoring data, Smart Materials and Structures, 29 (2020) 045029.

[3] Z. Ma, Y. Luo, C.-B. Yun, H.-P. Wan, Y. Shen, An MPPCA-based approach for anomaly detection of structures under multiple operational conditions and missing data, Structural Health Monitoring, (2022) 14759217221100708.

[4] I.-S. Jung, M. Berges, J.H. Garrett Jr, B. Poczos, Exploration and evaluation of AR, MPCA and KL anomaly detection techniques to embankment dam piezometer data, Advanced Engineering Informatics, 29 (2015) 902-917.

[5] H. Lee, G. Li, A. Rai, A. Chattopadhyay, Real-time anomaly detection framework using a support vector regression for the safety monitoring of commercial aircraft, Advanced Engineering Informatics, 44 (2020) 101071.

[6] D.J. Hill, B.S. Minsker, Anomaly detection in streaming environmental sensor data: A data-driven modeling approach, Environmental Modelling and Software, 25 (2010) 1014-1022.

[7] W. Aigner, S. Miksch, H. Schumann, C. Tominski, Time & time-oriented data, Visualization of Time-Oriented Data, Springer2011, pp. 45-68.

[8] G. Kerschen, P. De Boe, J.-C. Golinval, K. Worden, Sensor validation using principal component analysis, Smart materials and structures, 14 (2004) 36.

[9] Y. Yang, S. Nagarajaiah, Blind denoising of structural vibration responses with outliers via principal component pursuit, Structural Control and Health Monitoring, 21 (2014) 962-978.

[10] M. Gul, F.N. Catbas, Statistical pattern recognition for Structural Health Monitoring using time series modeling: Theory and experimental verifications, Mechanical Systems and Signal Processing, 23 (2009) 2192-2204.

[11] Y. Fu, C. Peng, F. Gomez, Y. Narazaki, B.F. Spencer Jr, Sensor fault management techniques for wireless smart sensor networks in structural health monitoring, Structural Control and Health Monitoring, 26 (2019) e2362.

[12] F. Ni, J. Zhang, M.N. Noori, Deep learning for data anomaly detection and data compression of a long-span suspension bridge, Computer-Aided Civil and Infrastructure Engineering, 35 (2020) 685-700.

[13] Y. Zhang, Y. Lei, Data Anomaly Detection of Bridge Structures Using Convolutional Neural Network Based on Structural Vibration Signals, Symmetry, 13 (2021) 1186.

[14] S. Li, L. Jin, Y. Qiu, M. Zhang, J. Wang, Signal Anomaly Detection of Bridge SHM System Based on Two-Stage Deep Convolutional Neural Networks, Structural Engineering International, (2021) 1-10.

[15] G. Liu, L. Li, L. Zhang, Q. Li, S. Law, Sensor faults classification for SHM systems using deep learning-based method with Tsfresh features, Smart Materials and Structures, 29 (2020) 075005.

[16] B. Lindemann, N. Jazdi, M. Weyrich, Anomaly detection and prediction in discrete manufacturing based on cooperative LSTM networks, 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE), IEEE, 2020, pp. 1003-1010.

[17] Z. Tang, Z. Chen, Y. Bao, H. Li, Convolutional neural network-based data anomaly detection method using multiple information for structural health monitoring, Structural Control and Health Monitoring, 26 (2019) e2296.

[18] Y. Bao, Z. Tang, H. Li, Y. Zhang, Computer vision and deep learning–based data anomaly detection method for structural health monitoring, Structural Health Monitoring, 18 (2019) 401-421.

[19] S. Ahmad, A. Lavin, S. Purdy, Z. Agha, Unsupervised real-time anomaly detection for streaming data, Neurocomputing, 262 (2017) 134-147.

[20] M.A. Mazurowski, P.A. Habas, J.M. Zurada, J.Y. Lo, J.A. Baker, G.D. Tourassi, Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance, Neural networks, 21 (2008) 427-436.

[21] M. Goldstein, S. Uchida, A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data, PloS one, 11 (2016) e0152173.

[22] A. Dalvi, Performance of one-class Support Vector Machine (SVM) in detection of anomalies in the bridge data, University of Cincinnati, 2017.

[23] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM computing surveys, 41 (2009) 1-58.

[24] S. Rajasegarar, C. Leckie, M. Palaniswami, J.C. Bezdek, Distributed anomaly detection in wireless sensor networks, 2006 10th IEEE Singapore international conference on communication systems, IEEE, 2006, pp. 1-5.

[25] H. Sarmadi, A. Karamodin, A novel anomaly detection method based on adaptive Mahalanobis-squared distance and one-class kNN rule for structural health monitoring under environmental effects, Mechanical Systems and Signal Processing, 140 (2020) 106495.

[26] Y.-M. Zhang, H. Wang, H.-P. Wan, J.-X. Mao, Y.-C. Xu, Anomaly detection of structural health monitoring data using the maximum likelihood estimation-based Bayesian dynamic linear model, Structural Health Monitoring, 20 (2021) 2936-2952.

[27] X. Ma, Y. Lin, Z. Nie, H. Ma, Structural damage identification based on unsupervised feature-extraction via Variational Auto-encoder, Measurement, 160 (2020) 107811.

[28] C.S.N. Pathirage, J. Li, L. Li, H. Hao, W. Liu, P.J.E.s. Ni, Structural damage identification based on autoencoder neural networks and deep learning, 172 (2018) 13-28.

[29] Z. Shang, L. Sun, Y. Xia, W. Zhang, Vibration-based damage detection for bridges by deep convolutional denoising autoencoder, Structural Health Monitoring, 20 (2021) 1880-1903.

[30] D.-M. Tsai, P.-H. Jen, Autoencoder-based anomaly detection for surface defect inspection, Advanced Engineering Informatics, 48 (2021) 101272.

[31] A. Mujeeb, W. Dai, M. Erdt, A. Sourin, One class based feature learning approach for defect detection using deep autoencoders, Advanced Engineering Informatics, 42 (2019) 100933.

[32] J. Mao, H. Wang, B.F. Spencer Jr, Toward data anomaly detection for automated structural health monitoring: Exploiting generative adversarial nets and autoencoders, Structural Health Monitoring, 20 (2021) 1609-1626.

[33] G.R. Garcia, G. Michau, M. Ducoffe, J.S. Gupta, O. Fink, Temporal signals to images: Monitoring the condition of industrial assets with deep learning image processing algorithms, Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, 236 (2022) 617-627.

[34] Y. Li, S. Peng, Y. Li, W. Jiang, A review of condition-based maintenance: Its prognostic and operational aspects, Frontiers of Engineering Management, 7 (2020) 323-334.

[35] E. Della Valle, S. Ceri, F. Van Harmelen, D. Fensel, It's a streaming world! Reasoning upon rapidly changing information, IEEE Intelligent Systems, 24 (2009) 83-89.

[36] E.P. Xing, Q. Ho, P. Xie, D. Wei, Strategies and principles of distributed machine learning on big data, Engineering, 2 (2016) 179-195.

[37] K. Zhang, S. Alqahtani, M. Demirbas, A comparison of distributed machine learning platforms, 2017 26th international conference on computer communication and networks (ICCCN), IEEE, 2017, pp. 1-9.

[38] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, J.S. Rellermeyer, A survey on distributed machine learning, ACM Computing Surveys, 53 (2020) 1-33.

[39] L. Qian, Z. Luo, Y. Du, L. Guo, Cloud computing: An overview, IEEE international conference on cloud computing, Springer, 2009, pp. 626-631.

[40] L. Yu, J.-C. Lin, Cloud computing-based time series analysis for structural damage detection, Journal of Engineering Mechanics, 143 (2017) C4015002.

[41] G. Cai, S. Mahadevan, Big data analytics in uncertainty quantification: Application to structural diagnosis and prognosis, ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering, 4 (2018) 04018003.

[42] C.S.N. Pathirage, J. Li, L. Li, H. Hao, W. Liu, P. Ni, Structural damage identification based on autoencoder neural networks and deep learning, Engineering structures, 172 (2018)

13-28.

[43] X. Lu, Y. Tsao, S. Matsuda, C. Hori, Speech enhancement based on deep denoising autoencoder, Interspeech, 2013, pp. 436-440.

[44] Q. Wang, Y. Ma, K. Zhao, Y. Tian, A comprehensive survey of loss functions in machine learning, Annals of Data Science, 9 (2022) 187-212.

[45] P. Chen, G. Chen, S. Zhang, Log hyperbolic cosine loss improves variational auto-encoder, The Seventh International Conference on Learning Representations (ICLR) 2019New Orleans, 2018.

[46] S. Bai, J.Z. Kolter, V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, arXiv preprint arXiv:1803.01271, (2018).

[47] Y. He, J. Zhao, Temporal convolutional networks for anomaly detection in time series, Journal of Physics: Conference Series, IOP Publishing, 2019, pp. 042050.

[48] F. Yu, V. Koltun, Multi-scale context aggregation by dilated convolutions, arXiv preprint arXiv:1511.07122, (2015).

[49] M. Thill, W. Konen, H. Wang, T. Bäck, Temporal convolutional autoencoder for unsupervised anomaly detection in time series, Applied Soft Computing, 112 (2021) 107751.

[50] B.W. Silverman, Density estimation for statistics and data analysis, Routledge2018.

[51] D.W. Scott, Multivariate density estimation: theory, practice, and visualization, John Wiley & Sons2015.

[52] F.E. Harrell, C. Davis, A new distribution-free quantile estimator, Biometrika, 69 (1982) 635-640.

[53] C. Leys, C. Ley, O. Klein, P. Bernard, L. Licata, Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median, Journal of experimental social psychology, 49 (2013) 764-766.

[54] J.P. Simmons, L.D. Nelson, U. Simonsohn, False-positive psychology: undisclosed flexibility in data collection and analysis allows presenting anything as significant, (2016).

[55] A. Sergeev, M. Del Balso, Horovod: fast and easy distributed deep learning in TensorFlow, arXiv preprint arXiv:1802.05799, (2018).

[56] D. Singh, B. Singh, Investigating the impact of data normalization on classification performance, Applied Soft Computing, 97 (2020) 105524.

[57] Y. Hu, T. Palmé, O. Fink, Fault detection based on signal reconstruction with auto-associative extreme learning machines, Engineering applications of artificial intelligence, 57 (2017) 105-117.

[58] N. Chouhan, A. Khan, Network anomaly detection using channel boosted and residual learning based deep convolutional neural network, Applied Soft Computing, 83 (2019) 105612.

[59] A. Gulli, S. Pal, Deep learning with Keras, Packt Publishing Ltd2017.

[60] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, {TensorFlow}: A System for {Large-Scale} Machine Learning, 12th USENIX symposium on operating systems design and implementation (OSDI 16), 2016, pp. 265-283.

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

[62] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, Proceedings of the thirteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 249-256.

[63] N.S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, P.T.P. Tang, On large-batch training for deep learning: Generalization gap and sharp minima, arXiv preprint arXiv:1609.04836, (2016).