Journal of Sound and Vibration 388 (2017) 154–170

Contents lists available at ScienceDirect

Journal of Sound and Vibration

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

Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks

Osama Abdeljaber [a], Onur Avci [a,*], Serkan Kiranyaz [b], Moncef Gabbouj [c], Daniel J. Inman [d]

[a] Department of Civil and Architectural Engineering, Qatar University, Doha, Qatar
[b] Department of Electrical Engineering, Qatar University, Doha, Qatar
[c] Department of Signal Processing, Tampere University of Technology, Tampere, Finland
[d] Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, USA

ARTICLE INFO

Article history:
Received 1 June 2016
Received in revised form
12 October 2016
Accepted 28 October 2016
Available online 9 November 2016

Keywords:
Vibration
Structural health monitoring
Structural damage detection
Neural networks
Convolutional neural networks

ABSTRACT

Structural health monitoring (SHM) and vibration-based structural damage detection have been a continuous interest for civil, mechanical and aerospace engineers over the decades. Early and *meticulous* damage detection has always been one of the principal objectives of SHM applications. The performance of a classical damage detection system predominantly depends on the choice of the features and the classifier. While the fixed and hand-crafted features may either be a sub-optimal choice for a particular structure or fail to achieve the same level of performance on another structure, they usually require a large computation power which may hinder their usage for real-time structural damage detection. This paper presents a novel, fast and accurate structural damage detection system using 1D Convolutional Neural Networks (CNNs) that has an inherent adaptive design to fuse both feature extraction and classification blocks into a single and compact learning body. The proposed method performs vibration-based damage detection and localization of the damage in real-time. The advantage of this approach is its ability to extract optimal damage-sensitive features automatically from the raw acceleration signals. Large-scale experiments conducted on a grandstand simulator revealed an outstanding performance and verified the computational efficiency of the proposed real-time damage detection method.

1. Introduction

Engineering structures have always been susceptible to various kinds of damage (deterioration, degradation, corrosion, fatigue, creep, shrinkage, etc.) during their service life due to environmental, operational and human-induced factors. With their relatively large size, damage inspection of civil infrastructure has been reported to be laborious and expensive. Yet, the civil structures need to be inspected regularly to remain operational, improve the lifecycle performance, avoid catastrophic failures and protect human lives.

When damaged, the material and geometric characteristics of a structural component change, affecting the stiffness and

* Corresponding author.
E-mail addresses: o.abdeljaber@qu.edu.qa (O. Abdeljaber), onur.avci@qu.edu.qa (O. Avci), mkiranyaz@qu.edu.qa (S. Kiranyaz), moncef.gabbouj@tut.fi (M. Gabbouj), daninman@umich.edu (D.J. Inman).

http://dx.doi.org/10.1016/j.jsv.2016.10.043
0022-460X/

stability of the structure. Conventional damage assessment methods, which depend on periodic visual inspection of structures are not efficient especially for complex structures as they require highly-trained labor and easy access to the monitored structural members. Detecting, locating and quantifying the structural damage in civil infrastructure have remained a constant challenge for researchers and engineers. Therefore, a significant amount of research has been conducted to develop automated local and global structural health monitoring (SHM) techniques [1].

Global (i.e. vibration-based) damage detection methods are used to assess the overall performance of the monitored structure by translating its vibration response into meaningful indices reflecting the actual condition of the structure. The ultimate goal of vibration-based methods is to identify the presence, severity, and location of the damaged areas by processing signals measured by a network of accelerometers. Vibration-based techniques can be classified into parametric (model-based) and nonparametric (signal-based) approaches. In parametric methods, system identification algorithms are utilized to determine the modal parameters such as natural frequencies and mode shapes from the measured response. Changes in these parameters with respect to the parameters identified for the undamaged case are used to recognize the structural damage. On the other hand, nonparametric approaches employ statistical means to identify damage directly from the measured signals.

During the last two decades, machine learning algorithms have been extensively used by researchers to develop a wide range of parametric and nonparametric vibration-based structural damage detection techniques. The vast majority of machine learning based damage detection methods available in the literature involve two processes, feature extraction and feature classification.

Parametric machine learning based methods available in the literature use different techniques to extract the modal parameters from the measured vibration response for several undamaged and damaged cases. In other words, the features extracted in these methods are simply the dynamic characteristics of the structure such as natural frequencies and mode shapes [2,3]. On the other hand, in nonparametric machine learning based damage detection methods, several signal processing and statistical analysis techniques have been utilized for feature extraction. Additionally, various classifiers have been used in both parametric and nonparametric methods to classify the extracted features. Section 2 presents a brief review on the feature extraction and classification techniques that have been used in both parametric and nonparametric structural damage detection methods.

Therefore, a classical signal-based structural damage detection approach typically consists of a continuous acquisition of signals by sensors, extraction of certain (hand-crafted) features and feature classification by a classifier. Accordingly, it is imperative to extract damage-sensitive features correlated with the severity of the damage in the monitored structure, and to have a well-configured and trained classifier that has the utmost ability to discriminate those features. This is why the choice of both features extracted and the classifier used usually depends on a *trial-and-error* process for a particular structural damage detection application and significantly varies in the literature. As discussed in Section 2, such fixed features/classifiers that are either manually selected or hand-crafted may not optimally characterize the acquired signal and thus cannot accomplish a reliable performance level for damage detection. In other words, which feature extraction is the optimal choice for a particular signal and classifier remains unanswered up to date. Furthermore, feature extraction usually turns out to be a computationally costly operation, which eventually may hinder the usage of such methods for a real-time SHM application.

This paper proposes a fast and highly accurate nonparametric vibration-based algorithm for structural damage detection based on adaptive 1-D Convolutional Neural Networks. The main objective is to identify and locate any structural damage in real-time by processing the raw vibration signals acquired by a network of accelerometers. With a proper adaptation over the traditional CNNs, the proposed approach can directly classify the accelerometer signal without requiring any feature extraction, pre- or post-processing. Consequently, this will lead to an efficient system in terms of speed, allowing a real-time application. Due to the CNNs ability to learn to extract the optimal features, with a proper training, the proposed system can achieve a superior damage detection and localization accuracy despite the noise-like and uncorrelated patterns of the accelerometer signal. Some samples of the latter are shown in Fig. 1. Furthermore, this study will demonstrate that simple CNN configurations can easily achieve a high detection performance compared to the complex ones commonly used for deep learning tasks over such complex and uncorrelated signals that can even defy a human expert inspector.

The proposed damage detection method is evaluated experimentally using Qatar University (QU) Grandstand Simulator. This structure is one of the largest lab structures used for testing machine learning based damage detection algorithms. Additionally, the structure is equipped with a large number of accelerometers compared to similar studies in the literature, which allowed the authors to test the CNN-based algorithm under several structural damage cases.

The rest of the paper is organized as follows. Section 2 includes a brief review of the recent applications of CNNs. Section 3 discusses the structural design and instrumentation of QU Grandstand Simulator. Overview, adaptation, and the back-propagation training of 1D CNNs are presented in Section 4. The proposed CNN-based damage detection algorithm is explained in Section 5. The experiments conducted to demonstrate the algorithm using QU Grandstand Simulator along with the results and performance evaluation are provided in Section 6. Finally, Section 7 concludes the paper and suggests potential topics for future research.

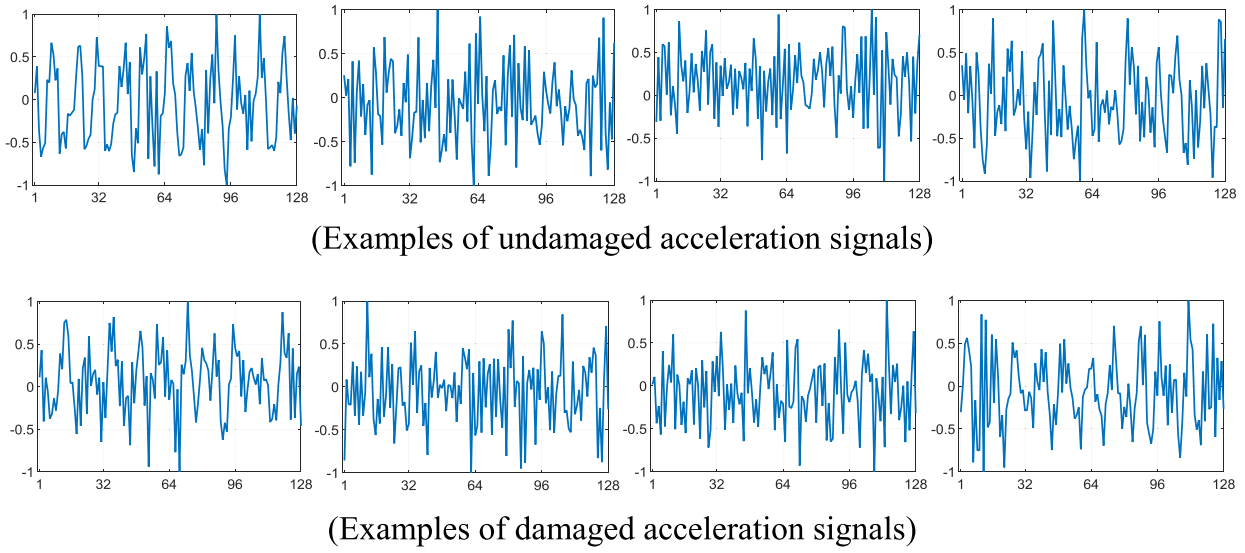

Fig. 1. Sample accelerometer signals from undamaged (top) and damaged (bottom) joints.

2. Related work

There have been numerous parametric and non-parametric structural damage detection methods proposed in the literature. Among many parametric methods, the most common classifiers are multi-layer feedforward artificial neural networks (ANNs) [4–10]. Additionally, researchers have implemented other classifiers in parametric methods such as online sequential extreme learning machine (OS-ELM) algorithm [11], probabilistic neural networks (PNNs) [12,13], and fuzzy neural networks (FNNs) [14]. On the other hand, several nonparametric and machine learning based damage detection methods that use a wide range of signal processing techniques have also been used for feature extraction. Examples of these techniques include simple statistical measures (maximum and variance of signals) [15], principle component analysis (PCA) [16–18], wavelet transform [19], autoregressive modeling [20,21], self-organizing maps [22–24], nonlinear autoregressive with exogenous inputs (NARX) neural networks [25]. Also, different classifiers have been used for feature classification in nonparametric methods such as ANNs [15–19], singular value decomposition [20], and support vector machine [21]. Previous studies have shown that nonparametric methods are able to distinguish damage cases that cannot be easily attributed to changes in modal parameters [26].

As mentioned earlier, the objective of this study is to address the aforementioned drawbacks and limitations of machine learning based nonparametric methods by using Convolutional Neural Networks (CNNs). CNNs have recently become the *de-facto* standard for "Deep Learning" tasks such as object recognition in large image archives as they achieved the state-of-the-art performances [27–29], with a significant performance gap. Typically, CNNs are feed-forward artificial neural networks (ANNs) that have both alternating convolutional and subsampling layers. As the convolutional layers basically model the cells in the human visual cortex [30], CNNs are developed primarily for 2D signals such as images and video frames. However, recently 1D CNNs have successfully been used for the classification of electrocardiogram (ECG) beats [31] achieving the state-of-the-art performance in terms of both accuracy and speed. Furthermore, in a recent study [32], 1D CNNs have achieved the fastest solution with an elegant accuracy for fault detection in high power engines. The main reason behind such superiority lies in the configuration of CNNs. Convolutional layers use linear filters, whose parameters are *optimized* during the training process. These filters extract crucial information (features), which characterize the object/pattern in an image/signal. The convolutional layers are followed by a feed-forward and fully connected layers, which are identical to the hidden layers of Multi-layer Perceptrons (MLPs) where the classification task is mainly realized. As a result, regardless of the variations in the signal characteristics and patterns, CNNs have the natural ability to learn the optimal features and the classifier parameters in a combined optimization process, the so-called Back-Propagation.

3. Qatar University grandstand simulator

Extensive analytical and experimental studies on structural health monitoring and vibration serviceability of stadia have been carried out at Qatar University. One of the main objectives is to develop efficient structural damage detection techniques that are suitable for monitoring of modern stadia. Before the application of the newly-developed techniques to real-life structures, it is pertinent to verify them experimentally in a controlled laboratory environment. Experiments performed using large-scale test structures provide an important link between the analytical work and the field deployment. For this purpose, a large grandstand simulator has been constructed at Qatar University (QU) which serves as a test bed for the

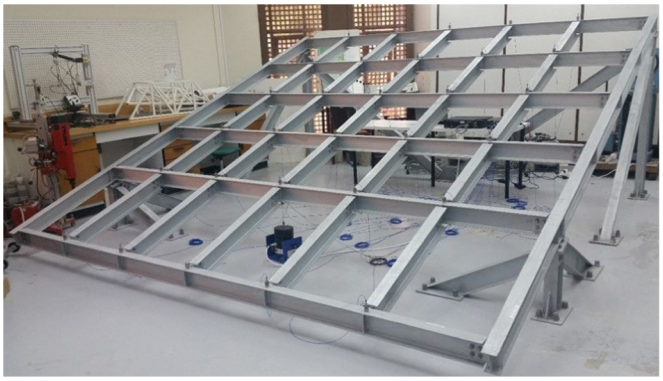

Fig. 2. Instrumented main steel frame of QU grandstand simulator.

project.

The QU grandstand simulator is still under construction. It is arguably the largest stadium structure built in a laboratory environment. The main hot-rolled steel frame of the test structure has been designed and constructed as shown in Fig. 1. Nonstructural elements of the grandstand simulator including risers, treads, seats, and handrails will be installed during the next phases of construction. In this study, QU grandstand simulator is utilized in its current form (steel frame only) to verify the proposed CNN-based damage detection algorithm.

3.1. Structural design and damage simulation

QU grandstand simulator was designed to host a total of 30 spectators with footprint dimensions of 4.2 m $\times$ 4.2 m. Several aspects were considered in the structural design of this test structure to guarantee its safety and compatibility with the specifications of modern grandstands [33]. As shown in Fig. 2, the steel frame consists of 8 main girders and 25 filler beams supported on 4 columns. The 8 girders are 4.6 m long, while the length of the 5 filler beams in the cantilevered portion is about 1 m and the length of the remaining 20 beams is 77 cm, each. The length of the two long columns is around 1.65 m.

Since the filler beams of the test structure are removable and interchangeable, many structural damage scenarios can be simulated either by loosening the bolts at beam-to-girder connections or by replacing some of the filler beams with damaged ones. As explained in detail in Section 6, the proposed algorithm is tested against very slight damage cases which were created by loosening one or more of the 30 beam-to-girder joints (steel connections).

3.2. Instrumentation

As shown in Fig. 2, the main steel frame of the grandstand simulator is equipped with a total of 30 accelerometers installed on the main girders at the 30 joints. 27 PCB model 393B04 accelerometers (Fig. 3a) and 3 B&K model 8344 accelerometers (Fig. 3b) were used. PCB model 080A121 magnetic mounting plates were used to attach the accelerometer to the steel structure. Also, a modal shaker (Model 2100E11) was used to apply vibration on the structure (Fig. 4). The signal is applied to the shaker through a SmartAmp 2100E21-400 power amplifier. Two 16-channel data acquisition devices were used to generate the shaker input and collect the acceleration output. The data acquisition devices and the shaker's power amplifier are shown in Fig. 5.

Fig. 3. The accelerometers. (a) PCB 393B04. (b) B&K 8344.

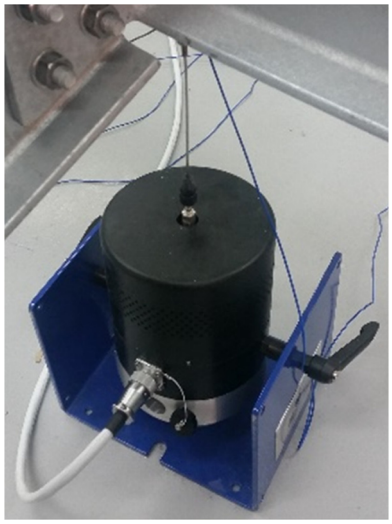

Fig. 4. The modal shaker (TMS 2100E11) attached to the test structure.

Fig. 5. The two data acquisition devices (DT9857E-16) along with the shaker's power amplifier (SmartAmp 2100E21-400).

4. Overview of CNNs

CNNs are biologically inspired feed-forward ANNs that present a simple model for the mammalian visual cortex. They are now widely used and have become the *de-facto* standard in many object and event recognition systems in an image or video. Fig. 6 illustrates a 2D CNN model with an input layer accepting 28×28 pixel images. Each convolution layer after the input layer alternates with a sub-sampling layer, which decimates the propagated 2D maps from the neurons of the previous layer. Unlike hand-crafted and fixed parameters of the 2D filter kernels, in CNNs they are trained (optimized) by the back-propagation (BP) algorithm. However, the kernel size and the sub-sampling factor, which are set to 5 and 2, respectively, for illustration purposes in Fig. 6, are the two major parameters of the CNN. The input layer is only a passive layer which accepts an input image and assigns its (R,G,B) color channels as the feature maps of its three neurons. With forward propagation over a sufficient number of sub-sampling layers, they are decimated to a scalar at the output of the last sub-sampling layer. The following layers are identical to the hidden layers of a MLP, fully-connected and feed-forward. These so called fully-connected layers end up with the output layer that produces the decision (classification) vector.

In this study an adaptive 1D CNN configuration is used in order to fuse feature extraction and learning (damage detection) phases of the raw accelerometer data. The adaptive CNN topology will allow the variations in the input layer dimension. Furthermore, as shown in Fig. 7 the compact CNN layer structure has now such hidden neurons of the

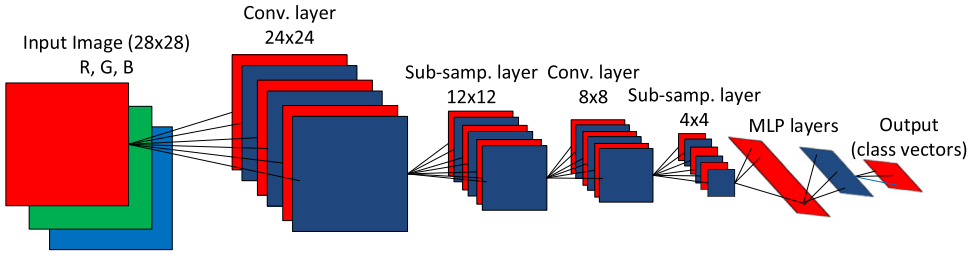

Fig. 6. Overview of a sample conventional CNN.

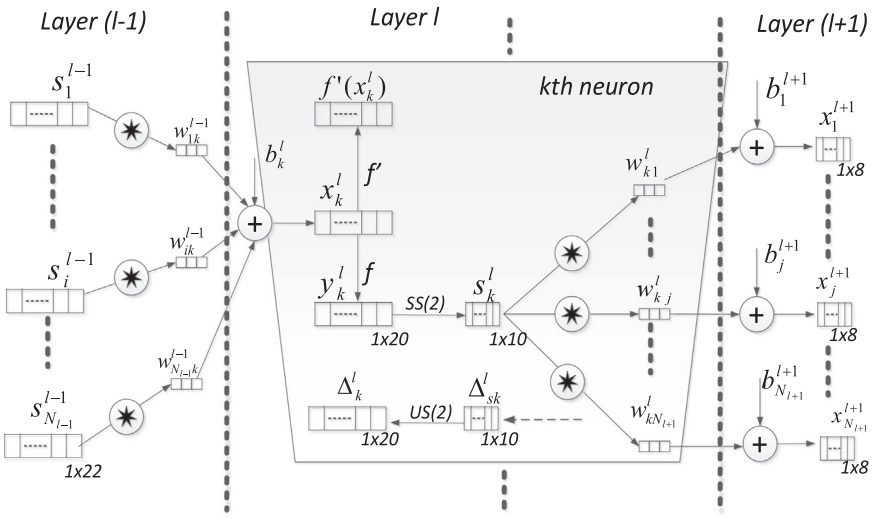

Fig. 7. The convolution layers of the proposed adaptive 1D CNN configuration.

convolution layers that can perform both convolution and sub-sampling operations. This is why the authors call the fusion of the convolution and the sub-sampling layers as the "CNN layer" to make the distinction but still call the remaining layers as the fully-connected layers that are identical to the hidden layers of a MLP. As a result, the adaptive 1D CNNs are composed of an input layer, hidden CNN layers, and fully-connected layers that end up with an output layer.

Further structural differences are visible between the traditional 2D and the proposed 1D CNNs. The main difference is the use of 1D arrays instead of 2D matrices for both kernels and feature maps. Accordingly, the 2D matrix manipulations such as 2D convolution (*conv2D*) and lateral rotation (*rot180*) have now been replaced by their 1D counterparts, *conv1D* and *reverse* operations. Moreover, the parameters for the kernel size and the sub-sampling are now scalars, K and ss for the 1D CNNs, respectively. However, the fully-connected layers are identical to their 2D counterpart and, therefore, it has the same BP formulation.

In 1D CNNs, the 1D forward propagation (FP) from the previous convolution layer, $l-1$, to the input of a neuron in the current layer, l, can be expressed as:

$$x_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} \text{conv1}D\left(w_{ik}^{l-1}, s_i^{l-1}\right) \tag{1}$$

where x_k^l is the input, b_k^l is a scalar bias of the k^{th}neuron at layer l, and $s_i^{l-1}$is the output of the i^{th} neuron at layer $l-1$. $w_{ik}^{l-1}$is the kernel weight from the i^{th} neuron at layer $l-1$ to thek^{th} neuron at layer l. The intermediate output of the neuron, y_k^l can then be expressed from the input x_k^l , as follows:

$$y_k^l = f\left(x_k^l\right) \text{and } s_k^l = y_k^l \downarrow ss \tag{2}$$

where $s_k^l$is the output of the neuron and $\downarrow ss$ represents the down-sampling operation with the factor, ss.

The adaptive CNN configuration requires automatic assignment of the sub-sampling factor of the output CNN layer (the last CNN layer). It is set to the size of its input array. For instance, in Fig. 7, assume that layer $l+1$ is the last CNN layer, then the parameter ss is automatically assigned to 8 which is the input array size. Such design allows the usage of any number of CNN layers.

The training methodology, Back-Propagation, will be briefly formulated in Appendix A, and further details can be found in [27] and [35].

5. The proposed damage detection and localization method

As discussed earlier, in this study, structural damage is simulated experimentally by loosening the bolts connecting the filler beams to the mean girders. The objective of the CNN-based algorithm is to detect the damage (if any) and identify the location of the damaged joint(s) accurately. The proposed algorithm requires designing and training a unique 1-D CNN for each one of the 30 joints instrumented with accelerometers. Each CNN is responsible for assessing the condition of one joint using only the raw acceleration signal measured at this joint. Therefore, the proposed damage detection algorithm is decentralized since each CNN is capable of detecting damage at its respective joint independently from the other CNNs.

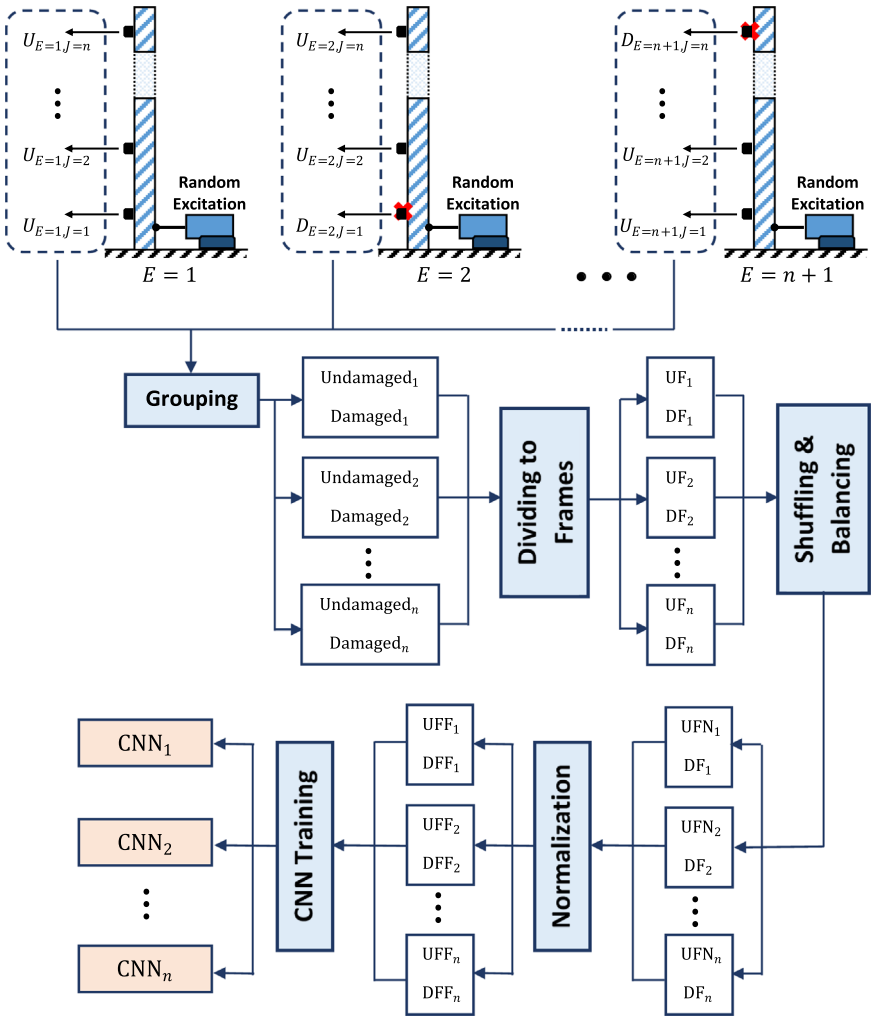

Fig. 8. Illustration of the data generation and the training process of CNNs. The damaged joints are marked with a red-cross. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.1. The training setup

Training of the CNNs requires generating a data set that consists of a number of undamaged/damaged acceleration signals for each joint. For a structure having a total of n joints (or accelerometers) as shown in Fig. 8, a unique CNN is assigned to each joint resulting in a total of n CNNs. To generate the training data set needed to train the CNNs, it is required to conduct $n + 1$ experiements. In the first experiment ($E = 1$), n acceleration signals are measured for the *undamaged* structure (i.e. when all joints are undamaged) under random shaker excitation. The resulting signals are denoted as $U_{E=1,J=1}$, $U_{E=1,J=2}$, …,$U_{E=1,J=n}$. The notation U indicates that the signal is measured at undamaged joint, while the subscripts E and J denote the experiment number and the joint number, respectively. Next, the remaining experiments are conducted one by one, in a sequential order.

In each experiment, $E = k + 1$, damage is introduced at the joint $J = k$ (by loosening its connection bolts in this study), and the n acceleration signals are measured under random excitation. The measured signals are denoted as $U_{E=k+1,J=1}$, …,$D_{E=k+1,J=k}$, …,$U_{E=k+1,J=n}$, where the notation D indicates that this signal was measured at the damaged joint k. After conducting the $n + 1$ experiments, the signals measured at each joint i are grouped together as follows in order to create the damaged/undamaged vectors required to train the corresponding CNN, CN N_i:

$$\text{Undamaged}_i = \left[U_{E=1,J=i} \;\; U_{E=2,J=i} \;\; \cdots \;\; U_{E=i,J=i} \;\; U_{E=i+2,J=i} \;\; \cdots \;\; U_{E=n+1,J=i} \right] \tag{3}$$

$$\text{Damaged}_i = \left[D_{E=i+1,J=i} \right] \tag{4}$$

One can notice that the undamaged data set for a particular joint i includes signals measured while the other joints are undamaged ($U_{E=1,J=i}$) as well as signals measured while one of the other joints are damaged. The data generation and

collection process was conducted this way in an attempt to minimize the effect of damaged joints on the classification efficiency of the CNNs at the other joints. In other words, the goal was to ensure that the effect of damaging a particular joint on the response of the other joints will not cause the CNNs to misclassify the undamaged joints as damaged.

The next step is to divide the aforementioned undamaged and damaged vectors to a large number of frames, where each frame contains a certain number of samples n_s. The result of this operation for joint i can be written as:

$$\mathrm{UF}_i = \left[\mathrm{UF}_{i,1} \;\; \mathrm{UF}_{i,2} \;\; \cdots \;\; \mathrm{UF}_{i,n_{\mathrm{uf}}}\right] \tag{5}$$

$$\mathrm{DF}_i = \left[\mathrm{DF}_{i,1} \;\; \mathrm{DF}_{i,2} \;\; \cdots \;\; \mathrm{DF}_{i,n_{\mathrm{df}}}\right] \tag{6}$$

where UF_i and DF_i are vectors containing the undamaged and damaged frames for the joint i, respectively, and n_{uf} and n_{df} are the total number of undamaged and damaged frames, respectively. Given the total number of samples in each acceleration signal n_T; n_{uf} and n_{df} can be computed as

$$n_{\mathrm{uf}} = n \times \frac{n_T}{n_s} \tag{7}$$

$$n_{\mathrm{df}} = \frac{n_T}{n_s} \tag{8}$$

From Eqs. (7) and (8), it is obvious that for a structure with a large number of joints (accelerometers) n, the number of undamaged frames for a particular joint will be significantly larger than the number of damaged frames. Training the CNN using extremely unbalanced undamaged/damaged frames may degrade the classification performance. Therefore, the frames corresponding to joint i are balanced according to the following procedure:

1. The n_{uf} frames in UF_i are randomly shuffled to yield a new vector, UFS_i.
2. The shuffled vector UFS_i is truncated by taking only the first n_{df} and removing the remaining frames resulting in a new undamaged vector UFN_i that contains a total of n_{df} frames.

Note that UF_i was shuffled in order to make sure that the undamaged frames from all experiments have an equal chance of being selected. Next, all frames in vectors UFN_i and DF_i are normalized between −1 to 1 resulting in the final vectors UFF_i and DFF_i that will be used to train the CNN, CNN_i.

To train CNN_i, first, it is required to specify its parameters such as the number of convolutional layers and neurons, the number of hidden fully-connected layers and neurons, the kernel size, K, and the sub-sampling factor, ss. Finally, the CNN training is carried out based on the data in UFF_i and DFF_i using back-propagation as explained in Section 4.1. The entire data generation and CNNs training process explained this section is illustrated in Fig. 8.

5.2. Testing of the CNNs

Once all CNNs are trained based on the procedure detailed in Section 5.1, they can be directly used to assess the condition of the structure. Each CNN_i is utilized to compute an index that reflects the likelihood of damage at joint i directly from the raw acceleration signal measured at its location. This can be done according to the following steps:

1. Induce damage at one or more locations (or keep the structure undamaged).
2. Apply a random shaker input.
3. Measure the acceleration signal at each joint.
4. Divide each acceleration signal to a number of frames each containing a total of n_s samples.
5. Normalize the frames between −1 to 1.
6. Feed the normalized frames measured at each joint to the corresponding CNN (CNN_i).
7. Compute the probability of damage (PoD_i) at the i^{th} joint as below:

$$\mathrm{PoD}_i = \frac{D_i}{T_i} \tag{9}$$

where T_i is the total number of frames processed by $CN\ N_i$, and D_i is the number of frames classified as “damaged”. As will be verified experimentally in Section 6, the PoD computed at damaged joints are expected to be significantly higher than the values for the undamaged joints. This gives a clear indication regarding both the presence and the location of a structural damage.

6. Experimental results

In this section, the efficiency of the proposed CNN-based damage detection algorithm is evaluated using the QU

grandstand simulator. The experimental work was carried out in two phases. In the first phase, for simplicity, only a single girder on the steel frame ($n = 5$ joints) was monitored. In the second phase, the performance of the damage detection approach was tested utilizing the entire structure ($n = 30$ joints).

6.1. Experimental setup

The horizontal girder at the middle of the test structure was considered for the first phase of the experimental work. As explained in Section 5.1, a total of 6 experiments were conducted to generate the data required for training. In each experiment, the acceleration signals were collected under a 0–512 Hz band-limited white noise shaker excitation at a sampling frequency of 1024 Hz. The signals were recorded for 256 s, so that each signal contains n_T=262144 samples. The shaker control and data acquisition operations were conducted using ME'ScopeVES software [34]. A Matlab [35] code (explained in Section 6.4) was used to group, divide into frames, balance, and normalize the data sets. The frame length n_s was taken as 128 samples, therefore, vectors UFF_i and DFF_i contain a total of 2048 frames for each joint i. Only 50% of these frames were used for the training process (i.e. 1024 undamaged frames and 1024 damaged frames for each CNN).

Five CNNs were trained for the five joints along the tested girder. All the CNNs were selected to have a compact configuration with only two hidden convolution layers and two hidden fully-connected layers. The aim of such a compact configuration was to accomplish a high computational efficiency required particularly for real-time detection. Besides, this also demonstrates that deep and complex CNN configurations are not really needed to achieve the desired detection performance. The structure and parameters of the 1D CNNs were obtained by trial-and-error. The 1D CNN configuration used in all experiments has (64, 32) neurons on the two hidden convolution layers and (10, 10) neurons on the two hidden fully-connected layers. The output (MLP) layer size is 2, which is the number of classes. Also, each CNN has a single input neuron which takes the input signal as the 128 time-domain samples of each frame in the training data set. The kernel size K, and the sub-sampling factor ss for all CNNs were set to 41 and 2, respectively.

For all experiments a two-fold stopping criteria for BP training was assigned:

1. The train classification error (CE) of 1%.
2. Maximum 100 BP iterations.

Whenever either criterion is met, the BP training stops. The learning factor, ε, was initially set as 0.001 and the global adaptation is performed at each BP iteration. At each iteration, if the train MSE decreases in the previous iteration, ε is increased by 5%; otherwise, reduced by 30%.

6.2. Phase-1 evaluation

After carrying out the training of the five CNNs, the performance of the damage detection algorithm is tested against eight structural cases. Case 1 of the first phase of the experimental demonstration corresponds to the undamaged girder, while Cases 2–6 are associated with a single damage (i.e. damage at a single joint). In Cases 7 and 8, double damages are

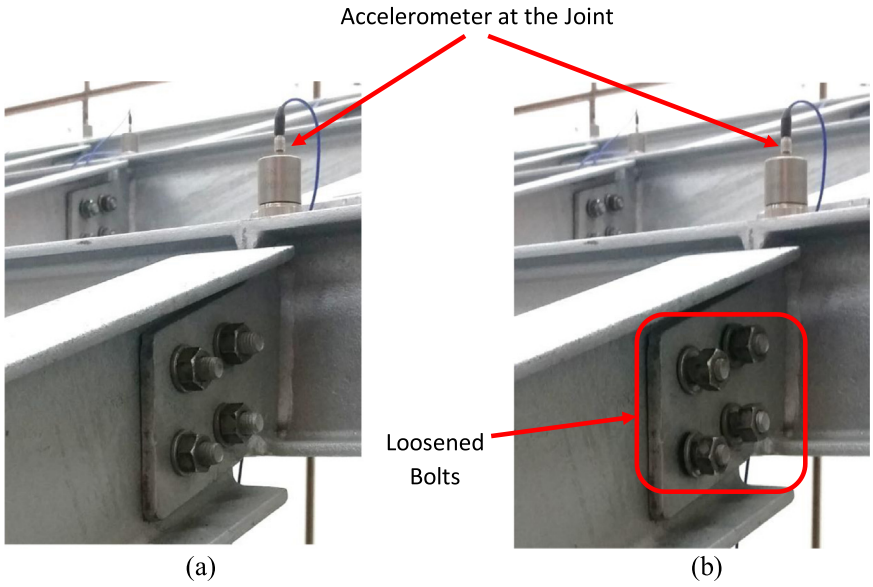

Fig. 9. (a) Undamaged joint. (b) Damaged joint.

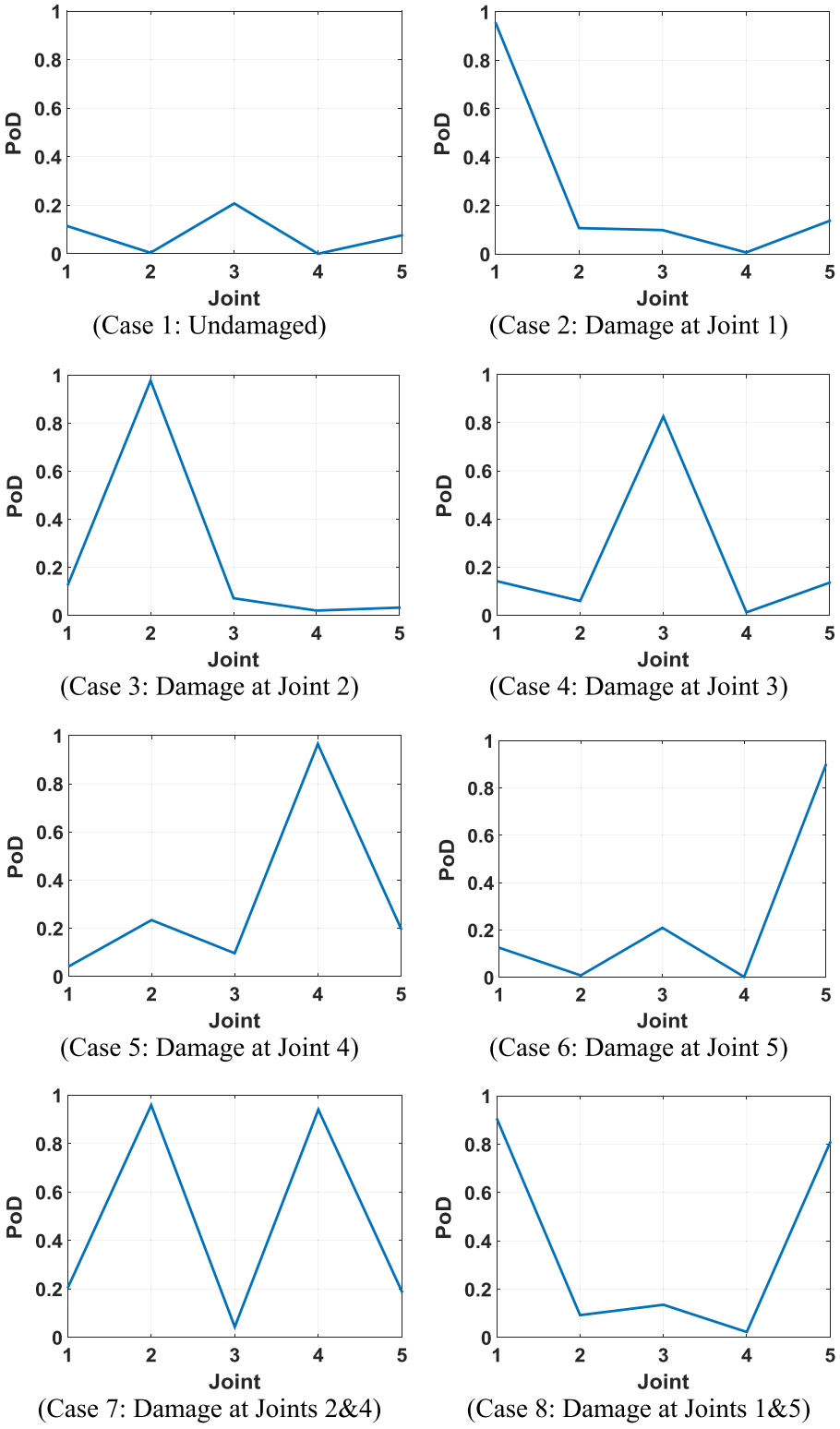

Fig. 10. PoD distributions for Cases 1–8 in the first phase of the experimental work.

applied to the girder (i.e. simultaneous damage at two joints). The damaged and undamaged joints are shown in Fig. 9. The testing process was conducted for each structural case as explained in detail in Section 5.2. By processing the raw signals using the CNN-based algorithm, the eight PoD distributions shown in Fig. 10 were obtained.

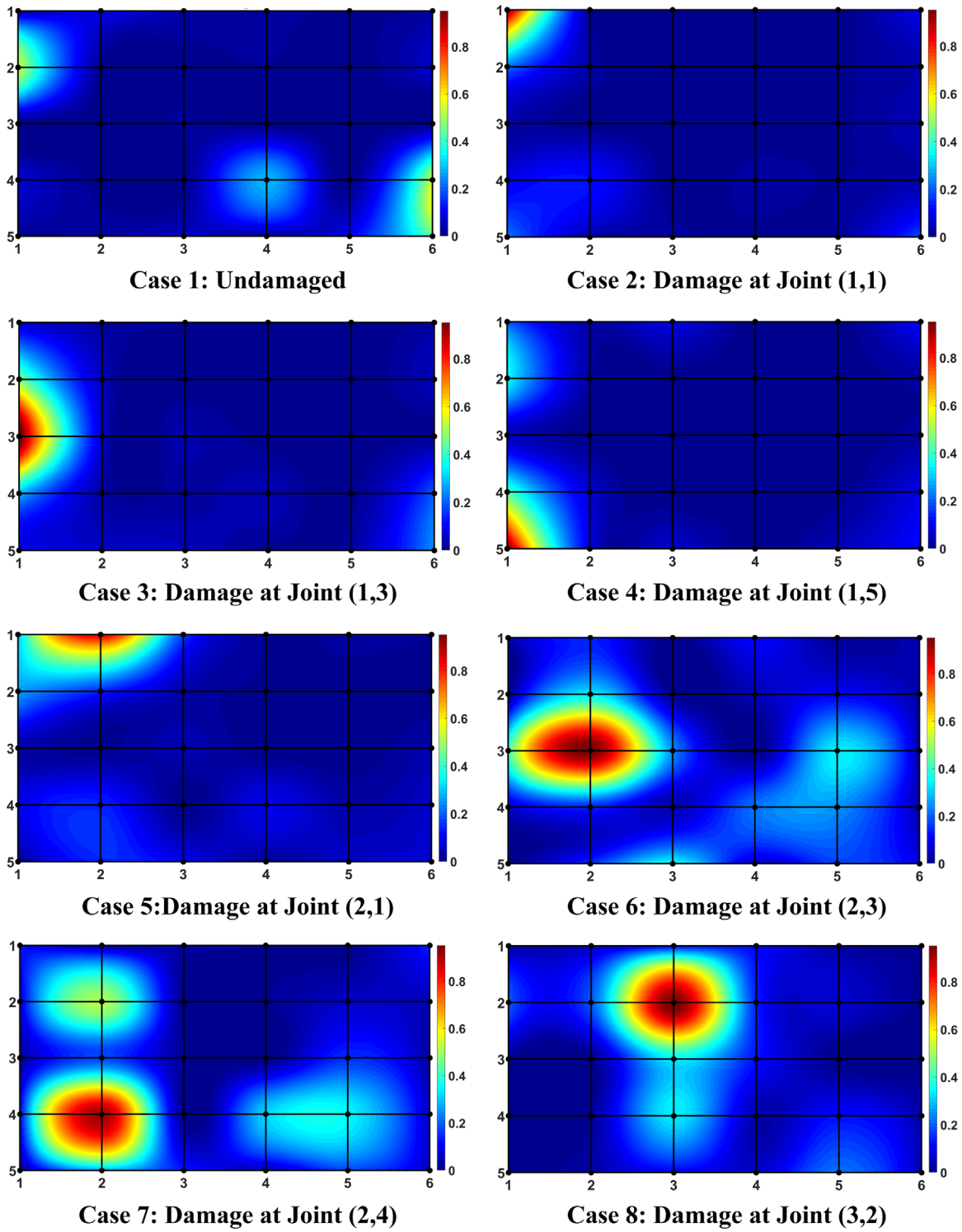

Fig. 11. PoD distributions for Cases 1–8 in the second phase of the experimental work.

The results of the first phase clearly indicates that the algorithm has successfully evaluated the condition of the monitored girder as undamaged. For Cases 2–6 (single damage cases), the algorithm has successfully assigned high PoD values (i.e. close to 1.0) to the damaged joints, while the computed PoD values for the intact joints were very low. Again, for both double damage cases (Cases 7 and 8), high PoD values were obtained for the two damaged joints, and much lower PoD values were assigned to the remaining joints.

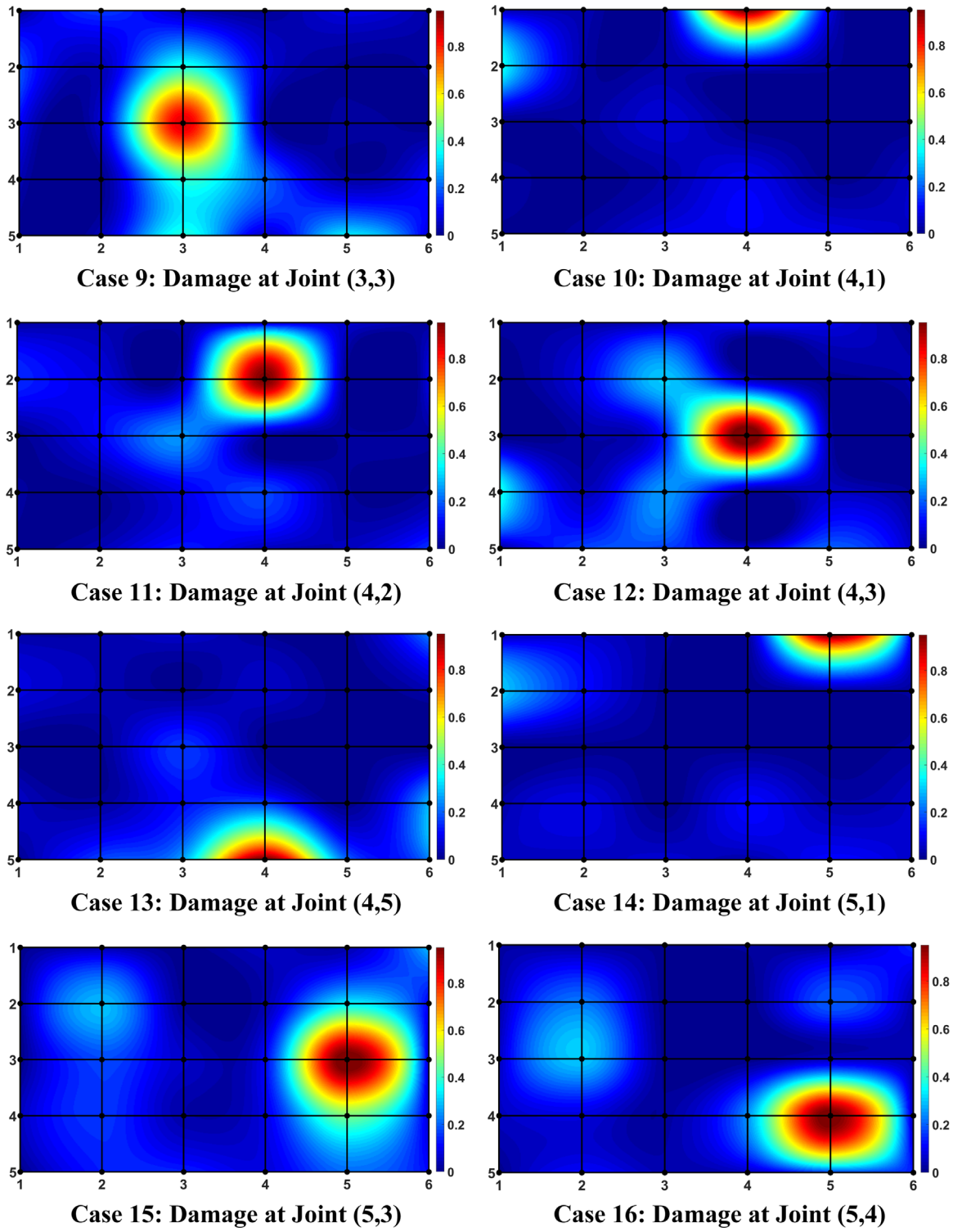

Fig. 12. PoD distributions for Cases 9–16 in the second phase of the experimental work.

6.3. *Phase-2 evaluation*

Similarly, for the second phase of the experimental work, the entire steel frame consisting of $n = 30$ joints was monitored. Therefore, a total of 31 experiments were needed to generate the training data. The same data generation and CNN

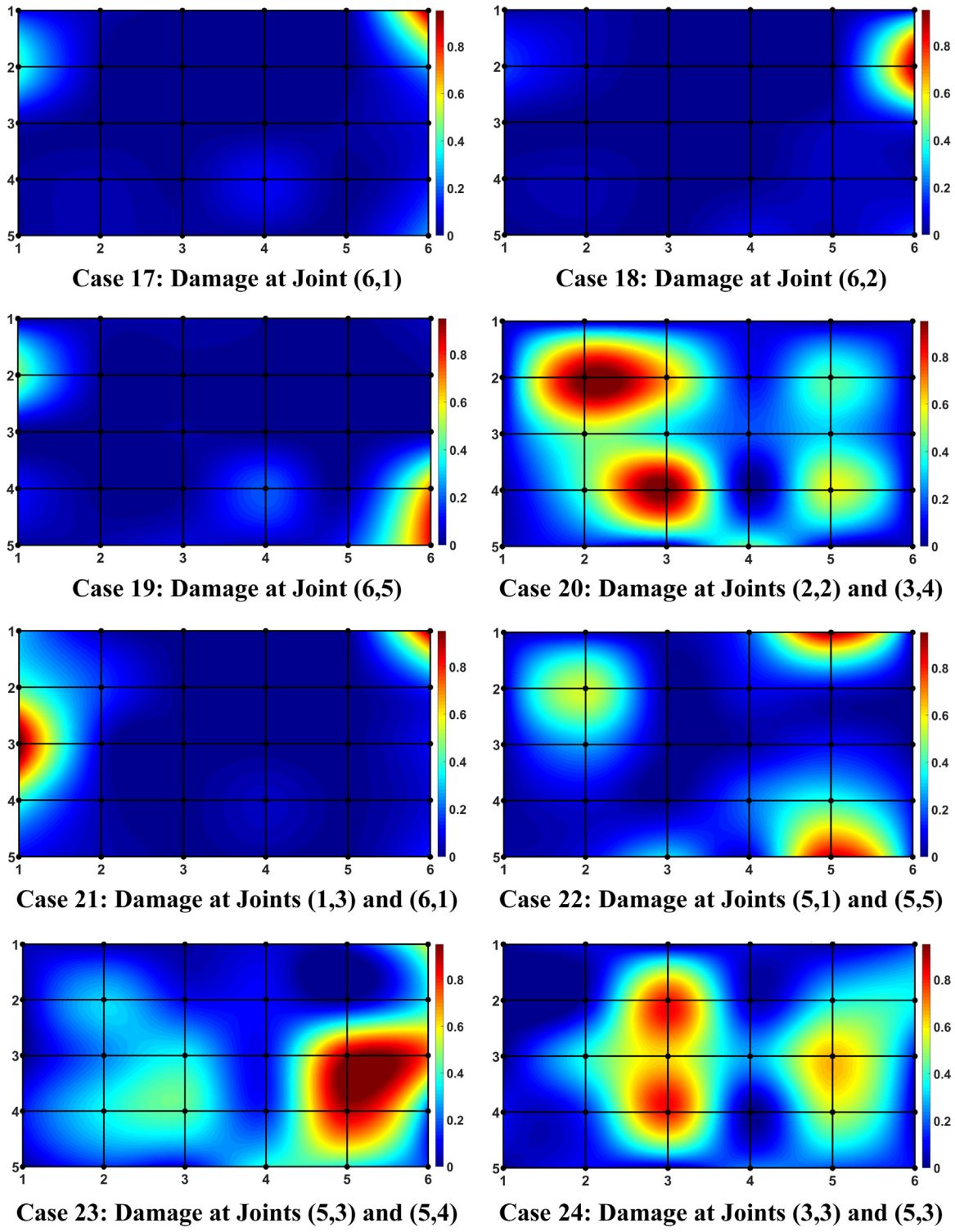

Fig. 13. PoD distributions for Cases 17–24 in the second phase of the experimental work.

training parameters used in the first phase were used again for this phase. The average classification error of the resulting 30 CNNs over the training data was found to be only 0.54%. To test the performance of the resulting 30 CNNs, the algorithm was tested against 24 structural cases (undamaged case + 18 single damage cases + 5 double damage cases). The PoD distributions obtained for 24 structural cases are presented in Figs. 10–13 in the form of "heat maps". For Case 1 (undamaged

case), the algorithm assigned low PoD values (less than 0.5) to all joints in the structure. For the single damage cases (Cases 2–19), the PoD maps correctly indicate significant damage at the damaged joint. Also, very good performance was achieved for the first three double damage cases (Cases 20–22), where the highest PoD values were assigned exactly to the two loosened joints.

In Case 23, the performance of the algorithm was tested when two adjacent joints are damaged. The results obtained for this case are satisfactory, however, high PoD value was incorrectly assigned to an undamaged joint (i.e. joint (6,3)). Also, for Case 24 where two joints along the structure's line of symmetry were damaged, the PoD map does not reflect the damage location accurately. Considering the very slight damage introduced to the structural system (only loosening the bolts of the connections), the results of the two phases clearly demonstrate an elegant performance of the proposed damage detection algorithm in assessing the condition of structures and locating single damages. While the performance of the proposed method was excellent under double damage cases for a structure monitored by a small number of accelerometers (first experimental phase), there is a room for improvement for the double damage cases on the entire test structure.

6.4. Computational complexity analysis

The adaptive 1D CNN classifier explained in Section 4.1 is implemented in C++ using MS Visual Studio 2013 in 64-bit. This program is capable of carrying out the forward and back-propagation operations required for training and using the CNNs. Also, a Matlab [35] code was written and used to extract vectors UFF_i and DFF_i directly from the signals collected in the experiments as detailed in Section 5.1. Another Matlab code was used to generate the PoD distribution directly from the raw acceleration signals using the trained CNNs as explained in Section 5.2. The experiments were conducted using a computer with I7-4910MQ at 2.9 GHz (8 cores) and 32-Gb memory.

As mentioned earlier, the key feature of the proposed CNN-based damage detection technique is that the computational time and effort required to classify the signals are significantly reduced. To illustrate this feature, the same CNN configuration has been used. The acceleration signal used for this illustration was acquired at a sampling frequency of 1024 Hz; therefore, it consists of 1024 samples. The signal was divided to eight frames each having 128 samples. Accordingly, the total time required for the classification of 1-s signal was only 22 ms. Note that this speed is about 45 × faster than the real-time requirement.

7. Conclusions and future work

This paper presented a novel damage detection approach with the adaptive implementation of 1D Convolutional Neural Network (CNNs). The proposed system was verified experimentally by monitoring the main steel frame of QU grandstand simulator. The results of the small and large-scale tests demonstrate a high performance level for real-time SHM and structural damage detection processes. Based on the experimental results presented in this study, the following conclusions can be drawn:

- This study presented the first attempt to implement a CNN-based approach in vibration-based structural damage detection. The results demonstrated the superior ability of the compact 1D CNNs to learn the extraction of optimal features automatically from the raw accelerometer data. The proposed approach not only achieves a high level of generalization but also eliminates the need for manual model or parameter tuning on any hand-crafted feature extraction. This fact makes the algorithm applicable for monitoring almost any civil infrastructure.
- Due to the simple structure of the 1D CNNs that requires only 1D convolutions (scalar multiplications and additions), a mobile and low-cost hardware implementation of the proposed approach is quite feasible. Moreover, since the CNN-based method is computationally inexpensive, it can be easily applied for real-time structural health monitoring of any engineering structure (e.g., civil, mechanical, or aerospace).
- The CNNs were capable of learning directly from the acceleration data measured under random excitations. In all training and testing steps, the excitation input was assumed to be unknown. Therefore, the proposed algorithm is very promising for monitoring civil structures under ambient vibration (i.e. damage detection using output-only data).
- The conventional centralized algorithms require the signals measured at all locations to be collected and transferred to a single processing unit. Transferring and synchronizing large amount of data can be problematic especially when a wireless sensor network is used for SHM. The proposed algorithm is decentralized, which means that a unique classifier (CNN) is assigned to each location. Each CNN processes only the locally-available data to assess the structural condition at its location. Hence, the proposed method offers an effective solution to overcome this problem.

As the future work, the following objectives will be addressed in order to further improve the performance of the CNN-based damage detection approach and to extend its application domain:

1. The performance of the proposed method under double damage cases can be further improved by following an alternative approach for generating the damaged/undamaged data at each location.
2. In this study, the CNN parameters (number of layers, number of neurons, kernel size, and sub-sampling factor) were

selected based on trial-and-error. It is expected that the classification performance of the CNNs can be enhanced by optimizing the CNN configuration and parameters.
3. Since the real damaged data needed to train the CNNs are rarely available for civil structures, the authors will investigate the possibility of training the CNNs based on real undamaged data together with simulated damaged data to detect real damages with a high accuracy.
4. In this work, the algorithm was tested against slight damage introduced at the structure's joints. In future studies, the authors will test the proposed method under more severe damage cases; e.g. beams with damaged flanges and or damaged webs.

Subsequently, the generated dataset along with the source code will be made publicly available in an open database repository.

Appendix A

A.1. Back-propagation for 1D CNN

The back-propagation (BP) steps are briefly formulated as follows. The BP of the error starts from the output fully-connected layer. Let $l = 1$ and $l = L$ be the input and output layers, respectively. Also, let N_L be the number of classes in the database. For an input vector p, and its corresponding target and output vectors, t_i^p and $\left[y_1^L,\cdots,y_{N_L}^L\right]$, respectively, the mean-squared error (MSE) in the output layer for the input p, E_p, can be expressed as:

$$E_p=\mathrm{MSE}\left(t_i^p,\left[y_1^L,\cdots,y_{N_L}^L\right]\right)=\sum_{i=1}^{N_L}\left(y_i^L-t_i^p\right)^2 \tag{10}$$

The objective of the BP is to minimize the contributions of the network parameters to this error. Therefore, it is required to compute the derivative of the MSE with respect to an individual weight (connected to that neuron, k) w_{ik}^{l-1}, and bias of the neuron k, b_k^l, so that the gradient descent method can be performed to minimize their contributions and hence the overall error in an iterative manner. Specifically, the delta of the k^{th} neuron at layer l, Δ_k^l will be used to update the bias of that neuron and all weights of the neurons in the previous layer connected to that neuron as:

$$\frac{\partial E}{\partial w_{ik}^{l-1}}=\Delta_k^l\, y_i^{l-1}\ \text{and}\ \frac{\partial E}{\partial b_k^l}=\Delta_k^l \tag{11}$$

So, from the first MLP layer to the last CNN layer, the regular (scalar) BP is simply performed as:

$$\frac{\partial E}{\partial s_k^l}=\Delta s_k^l=\sum_{i=1}^{N_{l+1}}\frac{\partial E}{\partial x_i^{l+1}}\frac{\partial x_i^{l+1}}{\partial s_k^l}=\sum_{i=1}^{N_{l+1}}\Delta_i^{l+1}\,w_{ki}^l \tag{12}$$

Once the first BP is performed from the next layer $l + 1$, to the current layer l, then it is possible to further back-propagate to the input delta Δ_k^l. Let zero order up-sampled map be $\mathrm{us}_k^l=\mathrm{up}\left(s_k^l\right)$, then one can write:

$$\Delta_k^l=\frac{\partial E}{\partial y_k^l}\frac{\partial y_k^l}{\partial x_k^l}=\frac{\partial E}{\partial \mathrm{us}_k^l}\frac{\partial \mathrm{us}_k^l}{\partial y_k^l}f'\left(x_k^l\right)=\mathrm{up}\left(\Delta s_k^l\right)\beta f'\left(x_k^l\right) \tag{13}$$

where $\beta=(ss)^{-1}$ since each element of s_k^l was obtained by averaging ss number of elements of the intermediate output, y_k^l. The inter BP of the delta error $\left(\Delta s_k^l \underset{\Sigma}{\leftarrow} \Delta_l^{l+1}\right)$ can be expressed as:

$$\Delta s_k^l=\sum_{i=1}^{N_{l+1}}\mathrm{conv1Dz}\left(\Delta_l^{l+1},\mathrm{rev}\left(w_{ki}^l\right)\right) \tag{14}$$

where $\mathrm{rev}(.)$ reverses the array and $\mathrm{conv1Dz}(.,.)$ performs full convolution in 1D with $K - 1$ zero padding. Finally, the weight and bias sensitivities can be expressed as:

$$\frac{\partial E}{\partial w_{ik}^l}=\mathrm{conv1D}\left(s_k^l,\Delta_l^{l+1}\right)\ \text{and}\ \frac{\partial E}{\partial b_k^l}=\sum_n\Delta_k^l(n) \tag{15}$$

As a result, the iterative flow of the BP algorithm can be written as the following:
1. Initialize all weights (usually randomly, $\mathrm{U}(-a,a)$)
2. For each BP iteration DO:
   a. For each item (or a group of items or all items) in the dataset, DO:
      i. **FP**: Forward propagate from the input layer to the output layer to find outputs of each neuron at each layer, $y_i^l,\ \forall\ i\in\left[1,N_l\right]\ \text{and}\ \forall\ l\in\left[1,L\right]$.

ii. **BP**: Compute delta error at the output layer and back-propagate it to first hidden layer to compute the delta errors, Δ_k^l, $\forall\ k \in \left[1, N_l\right]$ and $\forall\ l \in \left[2,\ L-1\right]$.
iii. **PP**: Post-process to compute the weight and bias sensitivities.
iv. **Update**: Update the weights and biases with the (accumulation of) sensitivities found in (c) scaled with the learning factor, ε:

$$w_{ik}^{l-1}(t+1) = w_{ik}^{l-1}(t) - \varepsilon \frac{\partial E}{\partial w_{ik}^{l-1}} \tag{16}$$

$$b_k^l(t+1) = b_k^l(t) - \varepsilon \frac{\partial E}{\partial b_k^l} \tag{17}$$

References

[1] S.R. Anton, D.J. Inman, G. Park, Reference-free damage detection using instantaneous baseline measurements, *AIAA J.* 47 (2009) 1952–1964, http://dx.doi.org/10.2514/1.43252.
[2] M. Mansouri, O. Avci, H. Nounou, M. Nounou, Iterated square root unscented Kalman filter for nonlinear states and parameters estimation: three DOF damped system, *J. Civil. Struct. Health Monit.* 5 (2015) 493–508.
[3] M.Mansouri, O.Avci, H.Nounou, M.Nounou, A comparative assessment of nonlinear state estimation methods for structural health monitoring, in: Proceedings of IMAC XXXIII Conference and Exposition on Structural Dynamics, Orlando, FL, USA, 2015.
[4] S.J.S. Hakim, H. Abdul Razak, S.A. Ravanfar, Fault diagnosis on beam-like structures from modal parameters using artificial neural networks, *Measurement* 76 (2015) 45–61, http://dx.doi.org/10.1016/j.measurement.2015.08.021.
[5] C. Ng, Application of Bayesian-designed artificial neural networks in Phase II structural health monitoring benchmark studies, *Aust. J. Struct. Eng.* 15 (2014) 27–37, http://dx.doi.org/10.7158/S12-042.2014.15.1.
[6] L.D. Goh, N. Bakhary, A.A. Rahman, B.H. Ahmad, Prediction of unmeasured mode shape using artificial neural network for damage detection, *J. Teknol. (Sci. Eng.)* 61 (2013) 57–66, http://dx.doi.org/10.11113/jt.v61.1624.
[7] M. Mehrjoo, N. Khaji, H. Moharrami, A. Bahreininejad, Damage detection of truss bridge joints using artificial neural networks, *Expert Syst. Appl.* 35 (2008) 1122–1131, http://dx.doi.org/10.1016/j.eswa.2007.08.008.
[8] M. Betti, L. Facchini, P. Biagini, Damage detection on a three-storey steel frame using artificial neural networks and genetic algorithms, *Meccanica* 50 (2014) 875–886, http://dx.doi.org/10.1007/s11012-014-0085-9.
[9] P.M. Pawar, K. Venkatesulu Reddy, R. Ganguli, Damage detection in beams using spatial fourier analysis and neural networks, *J. Intell. Mater. Syst. Struct.* 18 (2006) 347–359, http://dx.doi.org/10.1177/1045389X06066292.
[10] J.J. Lee, J.W. Lee, J.H. Yi, C.B. Yun, H.Y. Jung, Neural networks-based damage detection for bridges considering errors in baseline finite element models, *J. Sound Vib.* 280 (2005) 555–578, http://dx.doi.org/10.1016/j.jsv.2004.01.003.
[11] V. Meruane, Online sequential extreme learning machine for vibration-based damage assessment using transmissibility data, *J. Comput. Civil. Eng.* 30 (2015) 4015042, http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000517.
[12] X.T. Zhou, Y.Q. Ni, F.L. Zhang, Damage localization of cable-supported bridges using modal frequency data and probabilistic neural network, *Math. Probl. Eng.* 2014 (2014).
[13] S.F. Jiang, C.M. Zhang, C.G. Koh, Structural damage detection by integrating data fusion and probabilistic neural network, *Adv. Struct. Eng.* 9 (2006) 445–458, http://dx.doi.org/10.1260/136943306778812787.
[14] C.M. Wen, S.L. Hung, C.S. Huang, J.C. Jan, Unsupervised fuzzy neural networks for damage detection of structures, *Struct. Control Health Monit.* 14 (2007) 144–161, http://dx.doi.org/10.1002/stc.116.
[15] P. Chun, H. Yamashita, S. Furukawa, Bridge damage severity quantification using multipoint acceleration measurement and artificial neural networks, *Shock Vib.* 2015 (2015).
[16] U. Dackermann, J. Li, B. Samali, Dynamic-based damage identification using neural network ensembles and damage index method, *Adv. Struct. Eng.* 13 (2010) 1001–1016, http://dx.doi.org/10.1260/1369-4332.13.6.1001.
[17] U. Dackermann, W.A. Smith, R.B. Randall, Damage identification based on response-only measurements using cepstrum analysis and artificial neural networks, *Struct. Health Monit.* 13 (2014) 430–444, http://dx.doi.org/10.1177/1475921714542890.
[18] R.P. Bandara, T.H.T. Chan, D.P. Thambiratnam, The three-stage artificial neural network method for damage assessment of building structures, *Austr. J. Struct. Eng.* 14 (2013).
[19] Y.-Y. Liu, Y.-F. Ju, C.-D. Duan, X.-F. Zhao, Structure damage diagnosis using neural network and feature fusion, *Eng. Appl. Artif. Intell.* 24 (2011) 87–92, http://dx.doi.org/10.1016/j.engappai.2010.08.011.
[20] E. Figueiredo, G. Park, C.R. Farrar, K. Worden, J. Figueiras, Machine learning algorithms for damage detection under operational and environmental variability, *Struct. Health Monit.* 10 (2011) 559–572, http://dx.doi.org/10.1177/1475921710388971.
[21] A. Santos, E. Figueiredo, M.F.M. Silva, C.S. Sales, J.C.W.A. Costa, Machine learning algorithms for damage detection: kernel-based approaches, *J. Sound Vib.* 363 (2016) 584–599, http://dx.doi.org/10.1016/j.jsv.2015.11.008.
[22] O. Avci, O. Abdeljaber, Self-organizing maps for structural damage detection: a novel unsupervised vibration-based algorithm, *J. Perform. Constr. Facil.* http://dx.doi.org/10.1061/(ASCE)CF.1943-5509.0000801.
[23] O. Abdeljaber, O. Avci, Nonparametric structural damage detection algorithm for ambient vibration response: utilizing artificial neural networks and self-organizing maps, *J. Arch. Eng.* http://dx.doi.org/10.1061/(ASCE)AE.1943-5568.0000205.
[24] O.Abdeljaber, O.Avci, N.T.Do, M.Gul, O.Celik, F.N.Catbas, Quantification of structural damage with self-organizing maps BT – structural health monitoring, damage detection & mechatronics, in: A. Wicks, C. Niezrecki (Eds.), Proceedings of the 34th IMAC, a Conference and Exposition on Structural Dynamics 2016, vol. 7, Springer International Publishing, Cham, 2016, pp. 47–57. doi:10.1007/978-3-319-29956-3_5.
[25] L. Yan, A. Elgamal, G.W. Cottrell, Substructure vibration narx neural network approach for statistical damage inference, *J. Eng. Mech.* 139 (2011) 737–747, http://dx.doi.org/10.1061/(ASCE)EM.1943-7889.0000363.
[26] M. Gul, F. Catbas, Damage assessment with ambient vibration data using a novel time series analysis methodology, *J. Struct. Eng.* 137 (2010) 1518–1526, http://dx.doi.org/10.1061/(ASCE)ST.1943-541X.0000366.
[27] D.C. Cire/csan, U. Meier, L.M. Gambardella, J. Schmidhuber, Deep, big, simple neural nets for handwritten digit recognition, *Neural Comput.* 22 (2010) 3207–3220, http://dx.doi.org/10.1162/NECO_a_00052.
[28] D.Scherer, A.Müller, S.Behnke, Evaluation of pooling operations in convolutional architectures for object recognition, in: Proceedings of the 20th International Conference on Artificial Neural Networks: Part III, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 92–101. ⟨http://dl.acm.org/citation.cfm?Id=1886436.1886447⟩.
[29] S.Kiranyaz, M.A.Waris, I.Ahmad, R.Hamila, M.Gabbouj, Face segmentation in thumbnail images by data-adaptive convolutional segmentation networks, in: Proceedings of IEEE International Conference on Image Processing (ICIP), 2016, pp. 2306–2310. doi:10.1109/ICIP.2016.7532770, 2016.

[30] D.H. Hubel, T.N. Wiesel, Receptive fields of single neurones in the cat's striate cortex, *J. Physiol.* 148 (1959) 574–591 ⟨http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1363130/⟩.
[31] S. Kiranyaz, T. Ince, M. Gabbouj, Real-time patient-specific ECG classification by 1-D convolutional neural networks, *IEEE Trans. Biomed. Eng.* 63 (2016) 664–675, http://dx.doi.org/10.1109/TBME.2015.2468589.
[32] T. Ince, S. Kiranyaz, L. Eren, M. Askar, M. Gabbouj, Real-time motor fault detection by 1-D convolutional neural networks, *IEEE Trans. Ind. Electron.* 63 (2016) 7067–7075, http://dx.doi.org/10.1109/TIE.2016.2582729.
[33] O. Abdeljaber, A. Younis, O. Avci, N. Catbas, M. Gul, O. Celik, H. Zhang, Dynamic testing of a laboratory stadium structure, *ASCE Geotech. Struct. Eng. Congr.* (2016) 1719–1728, http://dx.doi.org/10.1061/9780784479742.147.
[34] ME'ScopeVES version 6.0.2015.1113. Scotts Valley, CA, Vibrant Technology.
[35] MATLAB version 8.1.0.604. Natick, MA, MathWorks.