Deep Learning for Structural Health Monitoring

By

Arya Panji Pamuncak

A thesis submitted to the University of Warwick

for the degree of Doctor of Philosophy in Engineering

University of Warwick, School of Engineering

June 2021

Table of Content

List of Figures

List of Tables

Acknowledgements

First and foremost, I wish to express my deepest gratitude to my supervisor, Dr. Irwanda Laory. I would like to thank him for providing his valuable support and guidance for my research throughout the last four years. I would also express my sincere gratitude to my second supervisor, Prof. Weisi Guo, for both his excellent guidance and encouragement, especially during my difficult time. This research would have not been possible without their support and advice.

Furthermore, I would also like to show my gratitude to LPDP (Indonesian Endowment Fund for Education) for the financial support they have provided during my research at the University of Warwick.

I am also grateful for all colleagues in the Structural Health Monitoring Research Group, especially Reza, Andre, Yanjie, Augusta, and Ahmed. Thank you for the useful advice and discussion that all of you have given for my research. In addition, I wish to thank the staffs for their professional advice during the experimental work.

Finally, my special thanks go to my beloved family for their endless supports. I am truly grateful for my parents, Mr Priyageng Sun Pangudi and Mrs Gardenia Priyageng, for their helps and supports over the years. In addition, I would like to thank Nini Lea, Eyang Lexi, Om Cahyo, and Tante Andri, that have accompanied me and my family in the UK, so we do not really feel homesick. Last but not least, I would like to thank my wife Rinda Amalia and my son Arkana. Both of them have given me unlimited encouragement and supported me unconditionally throughout my study.

Declaration

This thesis is submitted to the University of Warwick in support of my application for the degree of Doctor of Philosophy. The thesis is my own work and has not been submitted for a degree at another university.

Part of this thesis is based on materials provided in the following list of publications.

List of Publications

<u>Journal Paper (published)</u>

A. Pamuncak, W. Guo, A. Soliman Khaled, and I. Laory, “Deep learning for bridge load capacity estimation in post-disaster and -conflict zones,” *R. Soc. Open Sci*., vol. 6, no. 12, p. 190227, Dec. 2019.

A. P. Pamuncak, M. R. Salami, A. Adha, B. Budiono, and I. Laory, “Estimation of structural response using convolutional neural network: application to the Suramadu bridge,” Eng. Comput., 2021.

<u>Journal Papers (submitted)</u>

A. Adha, A. Pamuncak, W. Qiao, and I. Laory, “Automated building classification framework based on FEMA building typology using Convolutional Neural Network,” International Journal of Disaster Risk Rediction., 2021.

<u>Conference Papers</u>

A Pamuncak, W Guo, A Jesus and I Laory. A deep learning-based framework for bridge load rating estimation. The 9th European Workshop in Structural Health Monitoring (EWSHM 9). Manchester, UK, 2018.

M.R. Salami, A Jesus, A.P. Pamuncak, I. Zarkasi, B. Suhendro, B. Budiono, P. Suprobo, I. Laory. Long-Term Structural Health Monitoring of Suramadu Bridge. The 8th International Conference on Structural Health Monitoring of Intelligent Infrastructure (SHMII), Brisbane, 2017.

Abstract

In recent years, Structural Health Monitoring (SHM) has attracted significant attention due to its potential in providing effective maintenance strategy for infrastructure. However, interpreting the SHM data remains a challenging task. Model-based interpretation which utilises a behaviour model in the interpretation, requires experts in both developing the model and understanding the change in the model. On the other hand, data-based/model-free interpretation methods reduce the complexity since no physical model is utilised. However, expertise is required in performing data-based interpretation methods due to the need of feature extraction. This thesis is motivated to develop a deep learning-based data interpretation method that can learn features automatically, thereby minimising the required expertise.

In this thesis, a deep learning-based method for estimating load capacity of bridges from bridges' images is developed. Data labelling is performed using information from National Bridge Inventory (NBI) database. Parametric study is performed to further investigate the method.

A deep learning-based method that utilises correlation between two or more sensor measurements is proposed. This method employs raw measurement data from sensors. The proposed method is implemented for estimating structural responses by using measurements from other sensor as the input. The proposed method is compared with other machine learning methods and the method outperforms the other methods.

Two damage detection approaches utilising deep learning techniques are discussed: novelty detection and multiclass classification. Both frameworks successfully predict the presence of the damage that could not be detected by a frequency-based method.

An approach that combines deep learning with Moving Principal Component Analysis (MPCA) as an existing damage detection method is introduced. Experimental data collected from a laboratory-scale bridge are employed as a case study to validate the method. A series of investigation on parameters used in both MPCA and deep learning architecture are conducted in order to observe the method.

key words: Structural Health Monitoring, Deep learning, Convolutional neural network, Data-based data interpretation, Damage detection, Machine learning, Moving Principal Component Analysis

Abbreviations

1-D	One-dimensional
2-D	Two-dimensional
AASHTO	American Association of State Highway and Transportation Officials
ANN	Artificial Neural Network
ASCE	American Society of Civil Engineers
CCF	Cross Correlation Function
CDF	Cumulative Distribution Function
CEEMDAN	Complete Ensemble Empirical Mode Decomposition with Adaptive Noise
CNN	Convolutional Neural Network
CorCNN	Correlation – Convolutional Neural Network
DAQ	Data Acquisition
EM	Elastomagnetic
FEA	Finite Element Analysis
FEM	Finite Element Method
FFT	Fast Fourier Transform
FHWA	Federal Highway Administration
FRF	Frequency Response Function
GSM	Gapped Smoothing Method
HHT	Hilbert-Huang Transform
ICA	Independent Component Analysis
IMPWH	Indonesian Ministry of Public Works and Housing

LHS	Latin Hypercube Sampling
LReLU	Leaky Rectified Linear Unit
LRFR	load and resistance factor rating
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
MLP	Multi-layer Perceptron
MLR	Multiple Linear Regression
MPCA	Moving Principal Component Analysis
NBI	National Bridge Inventory
PCA	Principal Component Analysis
PDF	Probability Density Function
PNN	Probabilistic Neural Networks
PS	Pressure Sensitive
PSD	Power Spectral Density
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
ROI	Region of Interest
RRA	Robust Regression Analysis
RTD	Resistance Temperature Detector
SGD	Stochastic Gradient Descent
SHM	Structural Health Monitoring
SVM	Support Vector Machine
SVR	Support Vector Regression
VI	Visual Instrument
WPD	Wavelet Packet Decomposition

Chapter 1 Introduction

1.1 Motivation

Infrastructure systems such as bridges, are valuable assets that play a critical role in the economy of regions by providing connections between communities. Currently, many in-service bridges have been operating for several decades, and they might have experienced deterioration that could reduce their service lives. Bridge failures might affect the economy severely as well as threaten public safety. Hence, it is important to maintain and recognise the structures' condition in keeping the economy running and public safe. Traditionally, visual inspections were performed for condition assessment. However, this approach is expensive, both on time and resources, as well as subjective. As a potential solution, Structural Health Monitoring (SHM) has attracted remarkable attention in recent years due to its capability in providing efficient framework for monitoring structures [1], [2]. In SHM, structures are equipped with a number of sensors thus the structures' condition can be monitored continuously, allowing early damage detection. Therefore, maintenance strategy can be timely performed in the initial stage of damage to prevent further damage on the structure.

Aside from providing effective maintenance strategy, SHM also offers long term economic benefits as a result from the reduced maintenance and operational cost. According to [3], implementing SHM on bridges approaching near the end of their service lives provides an annual return of 5:1 in respect to the investment cost of SHM. In addition, Carrión et al. [4] estimated that the implementation of SHM might provide total economic benefit between 37,333 USD and 51,667 USD annually. Moreover, according to Comisu et al. [5], SHM might decrease the maintenance cost and overall life cycle cost of a bridge by 25% and 10% respectively. The promising operational and economic benefits provided by SHM lead to the wide implementation of SHM in various structures all over the globe. However, despite having numerous data from SHM, interpreting the data still remains a challenging task.

Based on the presence of a behaviour model, data interpretation for SHM can be categorised into model-based and data-based interpretation approaches. In model-based data interpretation, a physical model representing the monitored structure, usually a Finite Element Method (FEM) model, is employed to estimate structural responses. Data interpretation is then performed by comparing the actual data obtained from sensors and the estimation from the model. In this approach, deviation from the model prediction indicates anomaly from normal condition thus the deviation is then employed using inverse method to further analyse the structure's condition. This approach suffers from the fact that creating a model that can perfectly mirror a structure is challenging and requires high level of expertise.

On the other hand, the data-based interpretation methods reduce the complexity in the interpretation since no physical model is involved in the process. The methods require no information about the structure's geometry or material since the interpretation is made by analysing the statistical pattern generated from the data. Generally, the data-based interpretation approach involves three main steps: data collection, feature extraction, and interpretation. In data collection, data, consisting of both structural responses and measurements of surrounding environment, are collected using a sensor system installed on a structure. Then, feature extraction is performed on the data to extract important information that can be useful for interpretation. Finally, based on the comparison on the extracted features, interpretation on the structure's condition is performed.

Despite the promising potential for SHM application, the data-based interpretation method also has a limitation. In applying this method, feature extraction might be considered as one of the most important steps as the selection of features might affect the accuracy of the interpretation. However, high level of domain knowledge is required in determining the important information to be extracted from raw data. Furthermore, the feature extraction steps are often performed based on the researcher's professional experience.

Therefore, the following research questions arise: How can we develop a method that can automatically learn features from data which might improve the data-based interpretation for SHM? How can we further enhance the existing data-based interpretation methods?

1.2 Objectives

This thesis aims to establish and develop a method for data-based interpretation of SHM, that can automatically learn important information from the data. Therefore, it can contribute in SHM field by reducing the level of knowledge required in performing data-based interpretation on SHM data. The objectives of the thesis are as follows:

- Develop a method for data interpretation on SHM data, either in the form of visual (images) or time history data.
- Develop a method that can automatically learn important information from SHM data.
- Improve existing data-based interpretation methods by incorporating deep learning.
- Test and validate the methods using laboratory and full-scale case studies.

1.3 Outline

The outline of the thesis is as follows:

Chapter 2 presents a review of literature on SHM including the data interpretation methods. In addition, a review of methodologies for data-based interpretation employing various approaches is presented.

Chapter 3 introduces a deep learning-based method for estimating bridges' load capacity. The method employs images of bridges as the input and generate estimation of bridge capacity in the form of load rating and design load.

Chapter 4 presents a data-based approach that implements deep learning for interpretation of time-series data. The method utilises correlation between sensor measurements and can be implemented for damage detection, sensor calibration, and estimation of missing data. The method is validated using data obtained from a a full-scale bridge.

Chapter 5 presents two damage detection frameworks employing deep learning. The first framework is based on novelty detection in detecting damage and it utilises a deep learning model that is trained in unsupervised way. On the other hand, the second framework employs deep learning models that are trained through supervised learning in order to predict both the presence and severity of damage on a bridge. In addition, parametric study on the model parameters is discussed in this chapter.

Chapter 6 presents a data-based interpretation method that combines deep learning and an existing methodology for damage detection purpose. This chapter includes the investigation on the impact of parameters to the damage detection performance.

Chapter 7 outlines a summary of conclusions on the thesis. This chapter also provides an assessment of how well the research objectives are achieved. Finally, areas of future works are also detailed in this chapter.

Chapter 2 Literature Review

Summary

In this chapter, a review of literature on SHM a is presented (section 2.1). The data interpretation methods for SHM that consist of model-based and data-based interpretation approaches are then discussed in section 2.2. Section 2.3 discusses further the methodologies that are used in data-based interpretation method including principal component analysis, moving principal component analysis, random forest, artificial neural networks, and multiple linear regression. The limitations of the data-based interpretation method are summarised in section 2.4 and deep learning techniques that provide potential solution for the limitation on current data-based method are reviewed in section 2.5. Section 2.6 reviews the convolution neural network and its implementation in the application of SHM. Finally, section 2.7 provides the summary and identifies the research gaps in the implementation of deep learning for SHM application.

2.1 Structural Health Monitoring (SHM)

There are some definitions on SHM. According to Housner et al. [6], SHM is the process utilising both sensing technologies and structural characteristics analysis for damage detection on structures. In addition, according to Aktan et al. [7], SHM should be considered within the field of condition assessment. SHM is also often defined as the process of utilising advanced damage detection strategy for civil, mechanical, and aerospace engineering structures [8]. Radzienski et al. referred SHM as the process of employing damage detection methods by using structural features [9]. To sum up, the main objectives of SHM lie in both damage detection and structural assessment.

There are three main areas of research on SHM such as damage identification, structural identification, and measurement system design. Generally, damage refers to the change in material, and/or structural geometry which might influence the structural performance [8]. Hence, the damage identification task involves performing comparison between two different conditions, one is generally representing the undamaged state. According to Rytter [10], the damage identification steps conducted using SHM system can be categorised as:

- Detection – identify the presence of damage.
- Location – verify the location of the damage.
- Classification – identify the type of damage.
- Assessment – collect information on the damage extent.
- Prognosis – estimate the residual life and asses the structural safety.

On the other hand, Structural identification is aimed to provide structural parameters and numerical models of the monitored structures [11]. The main objective of structural identification is to reduce the gap between the behaviour model and the real system in order to create a model that can help in assessing the structure’s condition reliably. Structural identification task involves creating/updating a structure’s numerical model based on actual response collected using SHMs.

Measurement system design deals with the optimal selection and placement of sensors [12]. The main aim of the measurement system design is to provide a cost-effective monitoring system while maintaining the accuracy of the system. Although it is beneficial to have a large number of sensors installed on a structure, installing these sensors and the supporting devices (i.e., cable, construction material, and data acquisition) requires high level of investment especially for the implementation on large structures. Therefore, optimal number of sensors have to be decided in order to minimise the installation cost. Moreover, in some scenarios, the sensors might even need to be installed inside the structure (i.e., embedded inside the concrete). Thus, in some cases, relocating sensors are not feasible hence careful planning is required in deciding where to install the sensors. On the other hand, the accuracy of modal parameter identification relies heavily on the sensors' placement in the structure. In this case the number of sensors have to be sufficient enough to obtain data that can accurately represent the structures. Therefore, measurement system design is necessary to find the optimal sensor number and location in order to provide efficient and accurate monitoring system.

According to [13], the process in SHM consists of four main steps: operational evaluation, data collection and cleansing, feature extraction, and interpretation using the extracted features. The operational evaluation is performed to gather information including the damage definition, the operational and environmental condition of the monitored structures, the challenges in data collection considering the surrounding environment, and the feasibility in implementing the system [13]. Data collection step is conducted by using sensors that are installed permanently on the structure. At present, SHM gain advantages from the advance of sensor and data technologies, allowing the possibility to deploy numerous sensors on structures for obtaining monitoring data. Various types of instruments such as accelerometer [14]–[17], strain gauges [18], fibre optic sensor [14], Global Positioning System (GPS) [19], video camera [20], elastomagnetic (EM) sensor [21], and laser-based sensor [22] have been employed for bridge monitoring application.

After data collection step, data cleansing is performed on the collected data to remove corrupt data before conducting feature extraction step. Feature extraction refers to the process of generating useful information from raw data. This process is aimed to retrieve damage sensitive parameters that can be used to identify the change in the structural condition. Finally, in the last step, the extracted features are utilised for the data interpretation. This step requires the use of behaviour or/and statistical models depending on the interpretation method that is applied.

In recent years, SHM has grown rapidly for monitoring long-span bridges as numerous SHM systems have been deployed on bridges all around the world. Some well-known bridges that have been installed with SHM system are Tamar Bridge [17], [23] and Humber Bridge [23] in the UK; Tsing Ma Bridge [18] and Stonecutters Bridge [24] in Hong Kong; Rainbow Bridge [25], Yokohama Bay Bridge [15], [25], and Higashi Kobe Bridge [26] in Japan; and Sutong Bridge [27] and Jiangyin Bridge [19] in China. While the aforementioned bridges employ wire-based SHM, some bridges such as Jindo Bridge [28], Yongjong Grand Bridge [29], Golden Gate Bridge [16], Caihong Bridge [30], and Jinzhou Bridge [30] adopt wireless sensor network technology for SHM. The wireless sensor network removes the need of cable installation since the data transmission is performed using wireless communication. Hence, the cost for the sensor installation required for cable installation can be reduced. In civil engineering application, the implementation of SHM is not only limited for bridge monitoring, but also for monitoring of dams [31]–[33], [33]–[36], tunnels [37]–[41], stadiums [42]–[46], and buildings [47]–[53]. In addition, apart from civil engineering application, SHM systems have been widely adopted in broad range of applications including mechanical [54]–[60], and aerospace engineering [61]–[69].

The data generated by SHM systems can be in the form of measurement data or images. The advance of computer vision field has opened great opportunity for vision-based SHM. Vision-based SHM combines sensors, image processing, and computer vision for SHM application. Vision-based SHM involves non-contact measurement thus offering simple installation, potentially providing a cost-effective alternative from conventional sensors. In vision-based SHM, visual data can be processed individually or in combination with measurement data. This is generally performed by analysing the pixels from the images. Previous research has implemented vision-based SHM for a broad range of application including defect detection and structural response monitoring.

The purpose of defect detection using vision-based SHM is to find structural damage such as for crack, spalling, rust, and loose bolt from images. Most research and application utilising vision-based SHM for bridge monitoring utilises zoomed images of a bridge part (i.e., bridge’s deck, connection, and tower) as the input. The images are generally obtained by using a camera that can either be installed in the site or attached to a moving vehicle (i.e., car, unmanned aerial vehicle, etc). Traditionally, the method requires extraction of features using hand-crafted technique from the image pixels. Then analysis is performed on the extracted information to determine the presence of damage. Various hand-crafted feature extraction methods have been employed. For example, Abdel-Qader et al. extracted PCA data from images of concrete deck from different condition for detecting the crack on the concrete surface [70]. Liu et al. utilised the edges from images as features for crack detection on concrete structures [71]. The author then utilised the features to train an SVM model in supervised way. In addition, Abdel-Qader et al. investigated four edge detection methods such as Fast Fourier Transform (FFT), fast haar transform, Canny detector, and Sobel [72]. In the research, edges were extracted from images of concrete and the information was employed for crack detection. It was shown that the fast Haar transform outperformed the other hand-crafted techniques. Hu et al. employed the textures and shape descriptors extracted from pavement images for crack detection [73]. In the research, SVM models were trained using the extracted features, and comparison with traditional edge-based methods was performed. It was shown that the proposed method managed to detect all cracks while tackling the illumination problem.

In addition to defect detection, vision-based SHM has been implemented for monitoring structural responses. For this type of application, generally the method is initiated by performing calibration on the camera. The purpose of the calibration is to find the scaling factor for conversion from pixels units to engineering units. The next step is performed by selecting Region of Interest (ROI). The ROI can be either a part of structures or an additional object installed on a structure (i.e., pre-installed markers, LED lamp, or panel). From the ROI, features are extracted, and object tracking is performed on the extracted features. In the final step, the movement of the features obtained through the visual tracking are translated into engineering units using the scale factor obtained during the calibration process.

Some research employing vision-based SHM for monitoring structural response has been reported. In 2015, Feng et al. proposed a vision-based method for measuring displacement of a bridge structure remotely without using artificial marker [74]. In the research, a conventional displacement sensor was used for comparison, and it was found that the proposed method achieved comparable measurement accuracy. In addition, it was also shown that the proposed method could further be used to measure high-frequency components. In 2018, Lydon et al. developed a multi-point displacement monitoring method for SHM using computer vision techniques [75]. In this research, multiple cameras were employed for monitoring bridge's displacement in several testing points. In addition to measuring displacement data, the method also utilised a camera to identify the load. During the field testing, two cameras were used to measure displacement and one camera was utilised to identify the vehicle type associated with the displacement. The vehicle information might potentially allow the calculation of global and local responses of the bridge to a passing vehicle. Feng et al. developed a vision-based cable force estimation method in 2017 [76]. In the research, cable vibration measurement was initially performed by using vision-based technique on images collected from a video camera. Fourier transform was then executed on the vibration data, and by using the frequency information, the cable force data were estimated. The results obtained from the proposed method were compared with a load cell reading and it was found that the proposed method produced maximum 5.6% of deviation from the costly conventional sensor. The promising potential of vision-based SHM has attracted significant attention for research community. Research on this method has been performed further for more advance application such as model updating [77], dynamic analysis [78], modal identification [79]–[81], and damage detection [82], [83].

2.2 SHM Data Interpretation methods

Currently SHMs have been widely employed for monitoring numerous structures all around the world, and from these systems, significant amount of data has been generated over time. The next important part of the process is how to gain important information from the data to understand the condition of the monitored structures. In general, the data interpretation method for SHM can be categorised into model-based and data-based methods [8], [84], [85]. The classification is based on the utilisation of physical models in interpreting the data. For more detail explanation on the weaknesses and strengths of each methods, the reader is referred to [86].

2.2.1 Model-based interpretation method

Model-based method utilises a behaviour model of the monitored structure, typically in the form of Finite Element Analysis (FEA) model, that is combined with inverse techniques in interpreting the data [87]. The model-based method generally involves two stages. In the initial stage, a model is calibrated by using data from SHM in order to increase its accuracy. At this stage, the structure is assumed to be in its healthy condition where no damage is present. In the next stage, the new response data collected from the SHM are compared with the prediction from the model and the deviation from the model is then used to update the model parameters [88]. This process is also known as the model updating technique which aims to minimise residual between actual data and model predictions [89]. Then, by interpreting the updated parameters, the structure's condition is determined.

In the initial stage of SHM implementation, the research was focused on model-based interpretation. In model-based method, most research has been performed on vibration analysis [90]–[93]. In this case, structural conditions are determined by identifying changes in stiffness in relation to the change in the vibration characteristic such as mode shapes or natural frequencies. In addition, research utilising model-based technique that employs static observation has also been reported [94], [95]. In the research, model updating has been executed by using displacement and strain measurements instead of vibration characteristic.

Implementing model-based method could be challenging since the method requires a model that can perfectly resemble the monitored structure. On the other hand, creating such models demands high level of expertise thus could be time and resource consuming. Moreover, understanding the updated parameter might also require high level of knowledge in the field. In addition, some assumptions are generally taken when building the models which might introduce uncertainties leading to inaccurate models. Finally, finding the type and optimal number of parameters can be a problem in employing model-based interpretation method [96]. The models might require the use of large number of parameters since in most scenarios the potential damage type and location is unpredictable.

2.2.2 Data-based interpretation method

Data-based interpretation methods employs advance statistic techniques on the monitoring data in order to learn the pattern from the data [8], [84]. This method requires no geometrical or material information of the structure being monitored since the features required for the interpretation are learned from the data, removing the need of implementing a behaviour model in the process. In this method, machine learning techniques have been widely adopted [84].

Data-based methods involve three main steps:

- **data collection** - structural responses and several types of loads (i.e. traffic and temperature data) are measured and collected using sensors.
- **feature extraction –** damage sensitive features that provide useful information in determining the health state of the structure are extracted from the data.
- **Prediction** – the structure’s condition is determined from either classification or regression by using the extracted features as the input.

In SHM application, machine learning methods can be employed in three ways: supervised, unsupervised, and semi-supervised. In supervised learning, each observation is given its corresponding label, generally by the structure’s condition when the data are collected, and a model is trained to predict the structure’s condition based on the input features. On the other hand, unsupervised learning requires no labelled data. In general, this method is applied for novelty detection where a threshold level is defined by using a statistical technique and anomaly is detected when an observation exceeds the threshold. Finally, semi-supervised learning utilises the combination of both labelled and unlabelled data in the process.

2.3 Methodologies for Data-based Interpretation

2.3.1 Principal Component Analysis (PCA)

PCA is one useful technique that can be used for data dimensionality reduction while preserving the important information from the original datasets [97]–[100]. This is achieved by transforming the data into a new and smaller set of uncorrelated variables that are generally called as principal components. The principal components are formed by a linear combination between the original variables and these components orthogonal to each other.

Consider a dataset $U(t)$ that contain some time series measurements from S sensors:

$$U(t) = \begin{bmatrix} u_1(t_1) & u_2(t_1) & \cdots & u_s(t_1) \\ u_1(t_2) & u_2(t_2) & \cdots & u_s(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ u_1(t_{N_m}) & u_2(t_{N_m}) & \cdots & u_s(t_{N_m}) \end{bmatrix} \tag{2-1}$$

where N_m is the total number of data points, u_i $(i = 1, 2, \dots, S)$ represents the response of i-th sensor, and t_j $(j = 1, 2, \dots, N_m)$ denotes the response at time step j. PCA is initialised by performing normalisation on the original dataset as follow:

$$U'(t) = \begin{bmatrix} \frac{u_1(t_1) - \mu_1}{\sigma_1} & \frac{u_2(t_1) - \mu_2}{\sigma_2} & \cdots & \frac{u_s(t_1) - \mu_s}{\sigma_s} \\ \frac{u_1(t_2) - \mu_1}{\sigma_1} & \frac{u_2(t_2) - \mu_2}{\sigma_2} & \cdots & \frac{u_s(t_2) - \mu_s}{\sigma_s} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{u_1(t_{N_m}) - \mu_1}{\sigma_1} & \frac{u_2(t_{N_m}) - \mu_2}{\sigma_2} & \cdots & \frac{u_s(t_{N_m}) - \mu_s}{\sigma_s} \end{bmatrix} \tag{2-2}$$

where μ_i and σ_i $(i = 1, 2, \dots, S)$ are the mean value and the standard deviation of the measurement from i-th sensor, respectively. Normalisation plays an important role in PCA to avoid dominant variables because of different measurement units used by the variables in a dataset [101]. After the transformation, all variables will be in the same scale. The normalisation will produce variables with zero mean and unit variance.

The next step on the PCA method is computing the $M \times M$ covariance matrix of the normalised dataset. the covariance matrix C is given by:

$$C = \frac{1}{M} U'^T U' \tag{2-3}$$

where M equals to the number of sensors used in the process. After the covariance matrix has been generated, the principal components are then calculated by obtaining the eigenvalue and eigenvector from the covariance matrix as follows:

$$(C - \lambda_i I)\psi_i = 0 \tag{2-4}$$

λ_i and ψ_i are the eigenvalue and its corresponding eigenvector, respectively. From this step, a total of M eigenvectors each having its corresponding eigenvalue are extracted. The eigenvectors are also called as principal components. These principal components are sorted based on the eigenvalues in descending order where the first principal component contains the highest variance from the dataset thus keeping the important information from the measurement. The second principal component has the second highest variance, and it is orthogonal to the first component. From this process, the first few principal components will have the most variance while the remaining components might only contain the noise from the measurement [84], [98]. Therefore, PCA analysis is generally conducted on the principal components that contain the most information.

Since it is easier to interpret and analyse a small set of uncorrelated variables, PCA has been broadly implemented in wide range of domain including engineering, biology, chemistry, geology, and social sciences. PCA has been implemented in SHM field, mainly for both dimensionality reduction and removal of environmental effects. For data dimensionality reduction, Ni et al. [102] proposed a damage detection method that combined neural network and PCA-compressed frequency response function (FRF) data. In this research, PCA was employed to reduce both the data dimension and noise from the FRF data collected from a building model at both undamaged and damaged condition. In this study, the performance of the proposed method was compared with the performance of a neural network that employed original FRF data as the neural network's input and it was shown that the PCA-based method outperformed the latter in term of prediction accuracy and robustness to measurement noise. In addition, Kromanis et al. [103] applied PCA for reducing feature dimension from dataset containing temperature and structural response measurements. This research was aimed to evaluate four machine learning approaches in predicting thermal responses. In this research, time histories of temperature and bridge responses were processed for noise and outlier removal. Then PCA was employed in the processed data for feature reduction and machine learning models were trained using the reduced features. The application of PCA managed to reduce the computational resource without significantly decreased the prediction accuracy of the prediction models. Furthermore, Mao et al. [104] employs PCA for automated modal identification. In the research PCA was applied for features reduction as well as noise removal. Modal validation criteria were calculated and PCA was utilised to transform the multidimensional criteria into a lower dimensional space. Next, k-means clustering was used on the first principal component generated from PCA to extract modal parameters. The method was validated using data from a laboratory model and SHM data. It was shown that the method successfully removed spurious modes and separated the close modes.

The data dimensionality reduction leads to the needs of smaller computation resources, potentially enabling the execution of data processing in the sensor nodes. In 2008, Park et al. [105] proposed a wireless micro electro-mechanical sensor that utilised PCA for damage detection on structures. The sensor system combined a micro-fibre composite patch and a low cost AD5933 data acquisition (DAQ) where the PCA operation for dimensionality reduction and noise reduction was executed. The features extracted from PCA were then clustered using k-mean algorithm to determine the structure’s condition. The proposed sensor was tested on aluminium structure. In the experiment, data from both healthy and damaged state were collected, and it was shown that the proposed wireless sensor managed to detect the damage by performing on-board computation.

Some research that employed PCA to remove environmental effects from data has also been reported. in 2012, Magalhães et al. [106] performed data analysis on two years of monitoring data obtained from an arch bridge. The analysis was performed by using the natural frequencies derived from accelerometer data and PCA was employed to reduce the impact of parameters such as humidity or wind load. Four damage scenarios were simulated by using a numerical model, and the proposed method suceesfully detected these conditions. Furthermore, Hu et al. [107] implemented PCA on three years of monitoring data collected from a footbridge to remove the environmental effect on the measurement for damage detection purpose. In this research, a damaged state dataset was generated using a behaviour model, and the proposed method managed to differentiate between healthy and damaged state data. Similarly, Concepcion et al. [108] employed PCA to compensate the temperature effect to the acceleration measurement. In this research, a wireless sensor network containing six nodes, each equipped with a temperature sensor and accelerometer, was installed on a concrete bridge. Temperature and acceleration data were collected on the bridge undamaged and damaged condition. Then, PCA was used to remove the temperature influence on the acceleration measurements. In 2015, Comanducci et el. [109] utilised PCA for removal of wind effects for damage detection purpose. In the research, acceleration data were generated using a numerical model by using wind speed data obtained from real measurement data. Modal properties were calculated using stochastic subspace identification procedure [110] and 12 frequencies were used in the research. PCA was applied to remove the effect of wind load from the time history of these frequencies. Damage was simulated by introducing various damage levels on the simulation model. Damage detection was performed in both noise-free and noisy environment and it was shown that the method could successfully detect the damage on different damage levels while producing robust performance even in the presence of noise.

In addition to measurement data, PCA has also been utilised on image processing for damage detection. In 2006, Abdel-Qader et al. [70] implemented PCA on images of concrete deck for crack detection. In this study, a database containing images of bridges obtained from different conditions was used. The PCA method then was utilised under three different approaches to detect crack from the image. Furthermore, Ho et al. [111] proposed a method based on image processing for detecting damage on cable surface of cable-stayed bridges. In the research, image data showing surface of cable bridges were collected by using a climbing robot. Initially, images were collected and labelled according to the presence of damage for training data. In detecting surface damage, PCA was applied on the input image to calculate the Mahalanobis distance for pattern classification. This method achieved 75% accuracy in detecting 40 damage cases, and the accuracy could be improved by increasing the amount of training data. Finally, due to the capability in the dimensionality reduction, PCA has also been exploited for data visualisation [112]. In this case, a dataset that previously consists of more than two variables can be visualised in 2-D plane by using PCA.

2.3.2 Moving Principal Component Analysis (MPCA)

PCA has a primary drawback: inclusion of all available data in performing the analysis. This drawback could lead to high resources required to perform the calculation. In addition, the inclusion of the undamaged and damaged state data when performing PCA might affect the capability in damage detection. In a scenario when damage just recently occurs, the undamaged state dataset might still heavily affect the measurement hence additional time delay is required before the eigenvector will be altered due to the damage [99]. To tackle this problem, MPCA that employs a moving window of fixed size in which PCA calculation is performed was introduced by Posenato et al. [113] in 2008. Figure 2.1 shows the steps executed in MPCA.

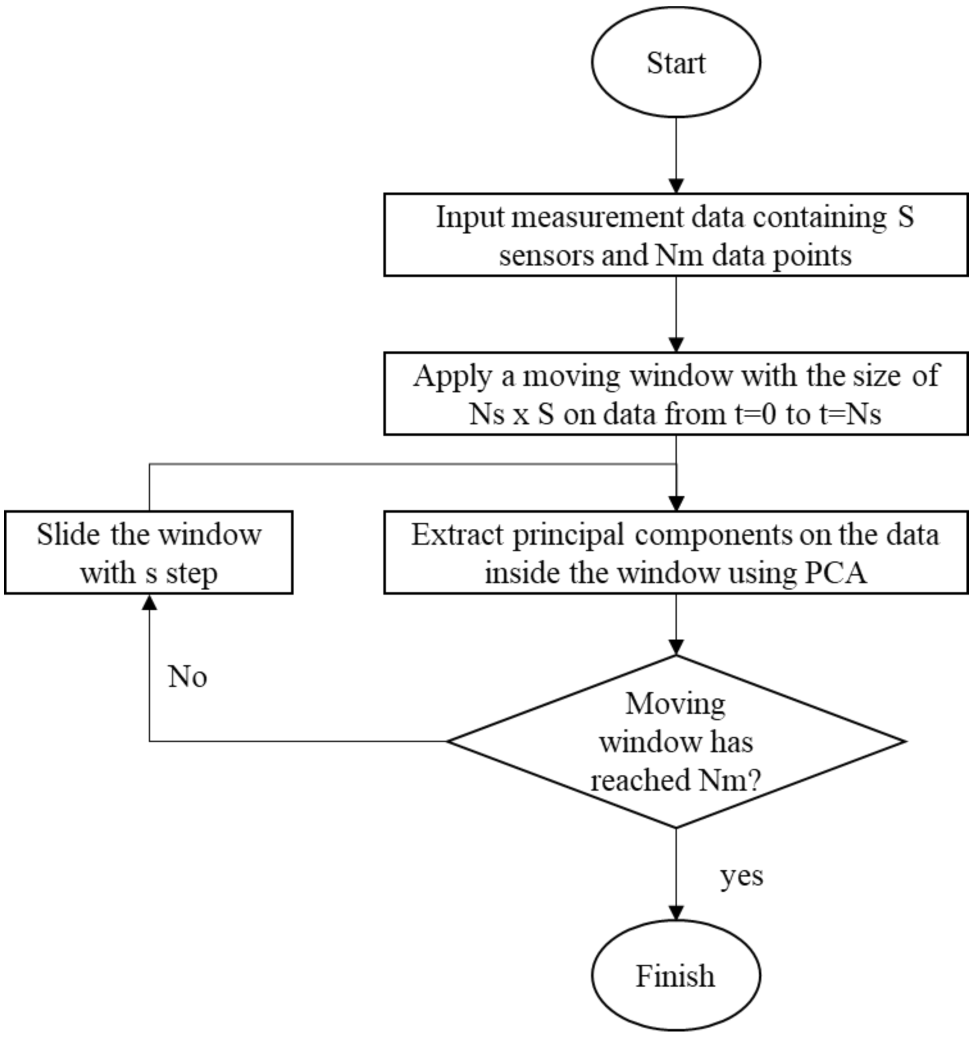

Start
Input measurement data containing S sensors and Nm data points
Apply a moving window with the size of Ns x S on data from t=0 to t=Ns
Slide the window with s step
Extract principal components on the data inside the window using PCA
No
Moving window has reached Nm?
yes
Finish

Figure 2.1 Steps performed in MPCA for time series data

In MPCA, a moving window of a fixed size N_S is sliding through time series. As the window slides, an individual PCA operation is conducted on the time series observation inside the window. The process involves normalisation on the data inside the window, calculation of covariance matrix from the normalised data, solving the eigenvalues and eigenvectors from the covariance matrix, and sorting the eigenvectors based on the eigenvalues. This way, the PCA calculation is only executed on the data inside the window instead of on the whole dataset. There are two main advantages that are achieved from the execution of PCA on the moving window. Firstly, it increases the computation efficiency due to the smaller number of data points when compared to performing PCA on the whole observations. Furthermore, it is possible to separate the damaged state data from the healthy state data by using the moving window. Therefore, MPCA also improves the capability in discriminating between features of undamaged and damaged state which results to improved damage detection performance.

MPCA is generally analysed on the first few eigenvectors since they contain most of the measurement variance. On the other hand, the remaining eigenvectors hold less important information that might be contributed by the measurement noise.

In the application of structural damage detection, MPCA is usually implemented in two phases: training and monitoring phases. In the training phase, it is assumed that there is no damage in the structure. The purpose of the training phase is to define a threshold level for separating the anomaly, that occurs due to damage, from normal measurement. For this purpose, all eigenvectors that are obtained during training phase are calculated and stored. Then, the mean value μ_i and the standard deviation σ_i of each eigenvector ψ_i are collected. The threshold value for eigenvector ψ_i is then determined as $\mu_i \pm 6\sigma_i$ [84].

During the monitoring phase, by using MPCA, new eigenvectors are obtained as the moving window slides through time series. These new eigenvectors are compared with the baseline threshold that has been defined in the training phase. Due to the presence of damage, the structural responses might be affected thus altering the covariance matrix, eigenvalues, and eigenvectors. Therefore, if the new eigenvector exceeds the threshold that has been defined in the training phase, then an anomaly that might indicate the presence of damage on the structure can be detected.

The size of the sliding window is considered as one of the most important parameters in performing MPCA. In selecting the window size, the periodic variability of the measurements has to be taken into account in order to reduce the variation that might not be caused by the structural damage. According to Laory et al. [84], [98], at least the window size should be the same with the longest periodic behaviour. This way, the periodic variability in the principal components obtained from MPCA can be avoided.

Some research that implements MPCA for SHM application has been reported. In 2008, Posenato et al. [113] utilised MPCA for damage detection using displacement data obtained from a numerical model. In this research, four damage scenarios were simulated, and the MPCA-based method was able to detect all scenarios. In 2011, Laory et al. [98] compared the performance of damage detection methods utilising MPCA and regression analysis. From this research, it was found that MPCA outperformed regression analysis in term of damage detectability although the latter produced shorter time delay in detecting damage. Cavadas et al. [114] proposed a damage detection method using combination between influence line and MPCA methods in 2013. Influence lines time histories were simulated using various loading data and anomaly was then detected by performing MPCA operation on the generated dataset. In this study, regression analysis was also performed as a comparison, and it was shown that the MPCA could detect and locate the damage despite requiring higher time for detection compared to the regression analysis method. Malekzadeh et al. [115] employed MPCA for damage detection using fiber optic sensor. In the research, strain data collection was conducted using 12 fibre bragg grating sensors installed on a laboratory scale bridge on six scenarios, where five scenarios represent the bridge's damaged state. A Low pass filter was applied to remove frequency above 2Hz and MPCA was employed on the filtered data. Damage index of each sensor was calculated using the first and second principal components of the corresponding sensor and damage detection is performed by monitoring all damage indexes. It was shown that the method utilising MPCA successfully detected damage on most damage scenarios.

In addition, MPCA has been combined with other methods for SHM purpose. Laory et al. [116] proposed a strategy for monitoring design of SHM system by combining MPCA and genetic algorithm methods. The approach was executed by performing genetic algorithm as a stochastic search with the objective of optimising multiple criteria such as the number of non-detectable damage scenarios, time delay for detection, and damage detectability. These criteria were extracted by using MPCA and regression analysis and decision was performed by using multi-criteria decision-making methods. Furthermore, in 2013, Laory et al. [84] combined MPCA with regression analysis methods such as Robust Regression Analysis (RRA), Support Vector Regression (SVR), and Random Forest to perform damage detection. The performances of the combined methods were compared with the performances produced when each method was executed individually, and it was shown that the combined methods outperformed the individual method in term of damage detectability and detection delay.

Some research performs data processing before the implementation MPCA. Zhu et al. [97] developed a damage detection method that combined Independent Component Analysis (ICA) and MPCA to extract thermal-induced strain from strain measurement. In this research, PCA was performed initially for feature reduction. Then ICA was executed on the reduced dataset to separate between traffic and temperature-related strain. MPCA operation was then conducted on the separated thermal-induced strain data for damage detection. The proposed method successfully improved the traditional MPCA method by reducing the delay in detecting anomaly while increasing damage detectability. In addition, Zhang et al. [99] proposed an improved MPCA-based damage detection method by combining space window with the traditional MPCA method. In the research, all sensors were clustered based on their location. MPCA was then operated on these clusters instead of doing on whole sensor data. The proposed method was verified using a numerical model and it was found that performing MPCA on the sensor cluster located near the damage could improve damage detectability compared to performing MPCA on the whole dataset.

2.3.3 Random Forest

In 2001, Breiman et al. proposed a computationally efficient technique for large datasets [117]. The approach is one of ensemble learning methods in which several randomly generated tree predictors are employed either for classification or regression problem. In Random Forest, each of the tree predictor produces output and the final output is made by aggregating. In regression problem where the prediction value is continuous, the final output is obtained by taking average value from the output of all tree predictors. On the other hand, for classification problem where the output is grouped into a categorical response, the output is produced by voting from the predictions made by the tree predictors. The use of multiple predictors instead of only one predictor offers higher performance and robustness to noise.

For regression problem, ensemble regression trees are utilised as the predictors in Random Forest. A single regression tree is constructed by splitting a dataset (root node) recursively into more homogeneous groups, which are called as nodes. Each node will either splits to create two descendant nodes or forms an end point which is referred to as terminal node as illustrated in Figure 2.2. For regression problem, at each node, the value of one input variable will be compared with a certain point. generally, if the input value is less than the split-point, then go to the left node. Otherwise, the right node is picked when the input value is higher. In the regression tree, each observation will take the appropriate path down until it arrives at the respective terminal node which determine the prediction result.

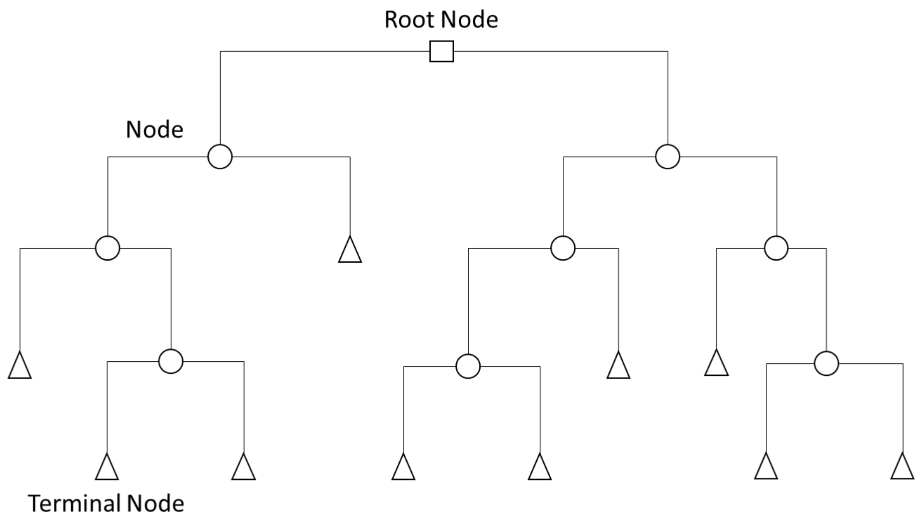


Figure 2.2 Example of a basic regression tree

Consider an original dataset $L(t) = (x_j(t), y(t))$, where x is the input containing explanatory variable(s), y is the dependant variable, $t = 1,2,..,N$ is the number of data points, and $j = 1,2,\ldots,p$ is the number of input variables. Random Forest approach is initiated by generating B sub-datasets that are randomly created from copying random data points from the original dataset L until each sub-dataset has equal number of samples as the original dataset. Due to the random sampling process, some data points might be copied more than once, while some might never be used in each sub-dataset. The samples which are not copied into a sub-dataset is employed as validation set of the corresponding sub-dataset.

The next step in the Random Forest is constructing B regression trees based on the B sub-dataset. Each generated sub-dataset will be employed in training its corresponding regression tree and performance evaluation of each tree is conducted using the corresponding validation set. As mentioned above, when given input data, each individual regression tree will generate prediction by following the path of the input data until it reaches the terminal node. Then, the final output of the Random Forest regression is obtained by averaging the output values from all regression trees as described in (2-5).

$$y(x) = \frac{1}{B}\sum_{b=1}^{B} y_b(x) + e \tag{2-5}$$

On the other hand, for random forest classifiers, the output is obtained differently. Each tree votes for a category and the Random Forest model picks a category with the highest number of votes as the output.

In designing Random Forest, some parameters should be taken into consideration such as the number of trees, the minimal node size, the number of randomly sampled variables considered at each split (generally referred to as mtry), and the splitting rule [118]. These parameters are also called as hyperparameters. Similar to hyperparameters on other machine learning techniques, the optimal values of these hyperparameters are problem dependent and the choice might affect the performance of the Random Forest model. In addition, the hyperparameter optimisation often involves trade-off between two or more objectives. Therefore, it is necessary to find the best value of hyperparameters that yields the lowest prediction error while avoiding overfitting.

The application of Random Forest technique for SHM field has been reported. The technique has been utilised to solve both regression and classification problems. In 2013, Laory et al. [84] combined MPCA with three regression analysis including Random Forest, SVR, and RRA. The Random Forest method was employed to perform regression between two elements in an eigenvector extracted from original dataset through MPCA. Furthermore, in 2014 Laory et al. [119] compared five machine learning approaches such as Random Forest, SVR, Artificial Neural Network (ANN), Multiple Linear Regression (MLR), and regression tree using monitoring data from Tamar Bridge as the case study. In this research, prediction models based on these five methods were trained to estimate the bridge’s natural frequency using traffic, temperature, and wind loading as the predictors. In the research, both the Random Forest and SVR outperformed the other techniques. In addition, it was also found that both traffic and temperature loading affected the natural frequencies significantly thus having these parameters is important when utilising frequency-based damage detection method. In addition, Xu et al. [120] compared ANN and Random Forest techniques for estimating structural responses. In the research, monitoring data from Forth Road Bridge were utilised. In the research PCA was used for data dimensionality reduction and Autoregressive Integrated Moving Average was employed to estimate the traffic condition. The traffic loading along with other data were employed as predictors to train machine learning models for estimating strain measurements. It was shown that the Random Forest model produced higher accuracy compared to the ANN model.

In performing classification function, Random Forest technique involves supervised learning where labelled data are utilised in the training. The data are firstly categorised into a number of classes and each data point is labelled according to its corresponding class. In 2018, Huang et al. [121] utilised Random Forest Algorithm for damage detection on benchmark data. This study employed American Society of Civil Engineers (ASCE) benchmark data which contained accelerometer measurements that were collected from seven conditions. Initially, feature extraction from the labelled acceleration data was performed using Cross Correlation Function (CCF) and Wavelet Packet Decomposition (WPD) and by using PCA, the extracted features were filtered based on the importance. The filtered features were then fed into Random Forest model for supervised training. The result was then compared with the result from an SVM model as a non-ensemble machine learning technique. The Random Forest algorithm produced significant increase in prediction accuracy compared to the SVM model. In 2020, Lei et al. [122] proposed a method for bridge evaluation utilising Random Forest classifier. Measurement data from a real bridge were collected and used to calibrate a behaviour model. Then, in total 672000 bridge models were created using Latin Hypercube Sampling (LHS) from the calibrated model. The data that were generated from these models were labelled into three classes including “Open”, “Restrict”, and “Close”, and then divided into 70% training and 30% testing dataset for supervised learning of Random Forest classifier. It was shown that the proposed method successfully achieved prediction accuracy between 89% and 97%, depending on the bridge component and system. Finally, Wang et al. proposed a method implementing Random Forest classifier for monitoring bolt connection on structures [123]. In the research, Random Forest models were utilised for classification of joint condition using features extracted from vibration data. In total, eight scenarios including the healthy state were implemented in the research and the random forest classifier generated prediction accuracy higher than 90%.

In addition to processing measurement data, Random Forest classifier has also been implemented to process visual data for damage detection. Van Der Horst et al. [124] developed a Random Forest-based road damage detection method using laser scan data. In this research, data were collected using a laser scanner installed on a car. By using K-means clustering, features were extracted from the images and these features were used as inputs for the Random Forest classifier that determined if damage has occurred on the road.

Aside from being implemented as either a classifier or regressor, Random Forest has also been implemented for feature extraction. In 2014, Zhou et al. [125] implemented Random Forest for feature selection in performing damage detection. In this research, WPD was initially applied on acceleration data to generate features. Then, by using Random Forest model, these features were ordered according to the variables' importance and the less important variables were omitted. The remaining features were then applied on K-nearest neighbour model to classify the structure's condition. In this research, the model with fewer features produced higher prediction accuracy. Another application of Random Forest in SHM field was also reported in 2019 [126]. Unlike the previously mentioned research that detected damage on structures, this research proposed a fault detection method on wireless sensors. There are some problems that might occur on the wireless sensor network such as data loss fault, gain fault, offset fault, and spike fault. In this research, Random Forest was implemented to detect the type of fault on the sensor. In total, six different types of failure were studied and six classifiers including Random Forest, Convolutional Neural Network (CNN), ANN, Support Vector Machine (SVM), Probabilistic Neural Networks (PNN), and Stochastic Gradient Descent (SGD) were implemented. It was shown that in all fault conditions, the Random Forest classifier managed to achieve the highest performance metric.

2.3.4 Artificial Neural Networks (ANNs)

Artificial neural networks (ANNs) are one of the computational models that are inspired biologically from how human brain process information [127]. The aim of the implementation of ANN models is to formulate the nonlinear relation between input and output variables by utilising layers of connecting neurons. In ANNs, the nodes in one layer are fully connected to the nodes in both previous and next layers. Each connection has its own associated weigh which is iteratively updated during the training step. Generally, ANNs consist of input layer, hidden layer(s), and output layer as it can be seen in Figure 2.3.

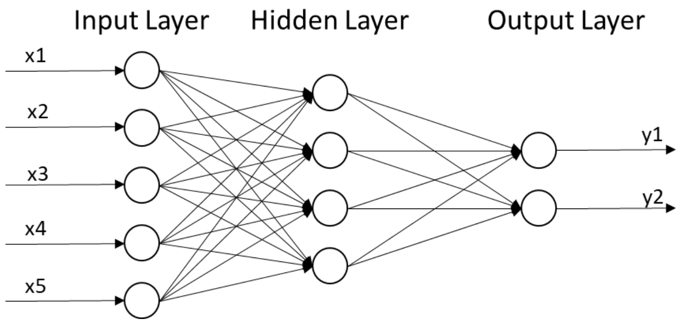


Figure 2.3 General architecture of ANNs consisting of one hidden layer between the input and output layers

In ANNs, the output of each neuron can be expressed as:

$$q^j = f_1\left(\sum_{i=1}^{i=r} P_i w_{ij}^1 + b_j^1\right), (j = 1,2,\dots,s) \tag{2-6}$$

where P is the input of the neuron from the previous layer, q is the output of the neuron, w is the scaled weigh of the neuron, b is the bias, and f is the activation function adopted in the layer. The activation function introduces non-linearity to the model. Commonly used activation functions for ANNs are sigmoid and tanh that are given by (2-7) and (2-8) respectively.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2-7}$$

$$f(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{2-8}$$

Sigmoid activation function uses a real-valued function as an input and produces an output in the range between 0 and 1 [128]. The outputs obtained from sigmoid are not zero-centred which might slow down the optimisation process. In addition, sigmoid also suffers from vanishing gradient. On the other hand, tanh produces zero-centred outputs in the range of -1 and 1 [129]. Similar to sigmoid, tanh also suffers from vanishing gradient problem.

During the training process, the training parameters, including the weighs and biases on ANN models, are modified iteratively to reduce the error that is produced from the discrepancies between the prediction and actual values. Commonly, backpropagation algorithm is implemented to minimise the error. In each training step, a set of inputs, also known as a mini batch, is feed into the models and the outputs are compared to the desired values. When there is small difference between these values, then no adjustment is performed on the training parameters on the models. On the other hand, when large error is produced, this error will propagate backward to modify the training parameters. The iterative training process will be finished either when the model has achieved a predefined level of error or when the training process has reached a predefined number of training iterations.

ANN has been broadly implemented in wide range of domains and for SHM application, some research utilising ANN has been performed for solving both regression and classification problems. In 2006, Lam et al. [130] employed ANN for identifying as well as locating damage on a structure. In the research, data were collected from a steel truss finite element model and Ritz vector changes were calculated from the data in the feature extraction stage. The features were utilised as the input of the ANN model, and it was shown that the ANN model could accurately predict all three damage scenarios. Furthermore, in 2008 Li et al. [131] utilised ANN for damage detection on a simulated three-span continuous beam. In the research, beams containing single and multi-damage cases were simulated and displacement data at the mid-point of each beam were collected. In the research the changes of variance produced from structural displacement were employed as the input for the ANN model. The ANN model produced identification accuracy between 80-88% by using the developed method. In 2019, Finotti et al. [132] proposed damage detection method using combined machine learning and statistical analysis. In the research 10 statistical indicators including signal peak, mean, kurtosis, standard deviation, skewness, crest factor, mean square, root mean square, K-factor, and variance were extracted from raw vibration measurement and used as features for training SVM and ANN classifier. Three case studies were implemented to validate the method and it was found that the ANN model generated prediction accuracy above 87.1% on all cases.

In addition to the implementation in wired-based SHM, ANNs have been implemented on wireless sensor networks. Xie et al. [133] investigated the implementation of ANN on a wireless sensor network for damage detection on a bridge. In the experiment, 400 wireless sensors were installed to measure acceleration on a bridge. Then, the data were grouped into four classes: healthy, damage level 1, damage level 2, and damage level 3. In the feature extraction stage, the natural frequency on each accelerometer was derived from the accelerometer measurement and the data were fed into an ANN model. Comparison was performed with other machine learning methods such as SVM, Decision Tree, and Logistic Regression and it was found that the ANN model produced the highest performance in term of prediction accuracy, recall, and noise robustness. Moreover, in 2019, Concepcion et al. [108] combined PCA and ANN for detecting damage on a reinforced concrete bridge. In the research, a wireless sensor network containing six rechargeable sensor nodes, each composed of a Raspberry Pi 3 Model B, temperature sensor and accelerometers, was deployed on a bridge platform to collect acceleration and temperature data. PCA was employed on the collected data to remove the influence of the temperature on the vibration. The compensated acceleration data were then utilised as the input of ANN models and the data were divided into 70% training, 15% validation, and 15% testing sets. Supervised learning with three labels each representing the structure's condition was performed using the training set. In this research, prediction accuracy up to 99.8% was produced by implementing the ANN model.

In addition, research implementing ANN regressor for damage detection has been reported. Jeyasehar et al. [134] utilised ANN for damage detection on prestressed concrete beams. In the research, some parameters including natural frequency, deflection, crack width, and ultimate load were utilised as damage sensitive features. In total five ANNs each having various type of input features were trained using backpropagation algorithm. It was found that all trained ANN models could predict expected damage level with deviation below 10%. Neves et al. [135] performed unsupervised learning using ANN for damage detection on a railway bridge. In the research, a numerically simulated railway bridge was used for the investigation. Six acceleration measurements were conducted and train passages with various load and speed were simulated. Six ANN models were trained to predict acceleration measurements in six locations. The input features of each ANN model consisted of the acceleration measurements from all locations, each having five datapoints, the train axle load, and the train speed. The output of these models was the prediction of future acceleration measurements. The output from the prediction model was then compared with the actual value to calculate the root mean square error (RMSE). Threshold level was determined using Gaussian process and anomaly was detected when the RMSE was exceeding the threshold. In addition, Mousavi et al. [136] performed damage localisation and quantification by using an ANN regressor. In the research, vibration data were collected from a laboratory truss bridge under an excitation. Features were extracted using Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (CEEMDAN) [137] and the extracted features were further transformed using Hilbert-Huang Transform (HHT) [138]. ANN models were trained using data from the healthy state. The model employed the features extracted using CEEMDAN as the input and the data transformed using HHT as the output. The proposed method detected damage by measuring the residuals generated between the output of ANN models and the actual data that were transformed using HHT. By comparing the residuals created from the sensors installed on different locations, the method was able to both quantify and localise the damage.

Research in compensating environmental effects to structural responses using ANNs has been reported. The reported research was mostly performing regression using ANN. In 2010, Zhou et al. [139] implemented ANN to find the correlation between natural frequencies and temperature using measurement data from Ting Kau Bridge SHM. In the research, some ANN models were trained by using combination of temperature data including the mean temperatures, the effective temperatures, and the principal components obtained from PCA application. Results showed that, given sufficient principal components, the model trained using the principal components generate the highest prediction accuracy. In 2014, Kromanis et al. [103] evaluated four machine learning techniques such as ANN, MLR, SVR, and robust regression for estimating thermal response. In the research, regression models were utilised to predict structural responses from the variation in the temperature data. In this study, all data were initially processed to remove the noise and outliers. Then PCA was performed for dimensionality reduction purpose. From the reduced features, pairs that consist of measurement from one temperature sensor and one tiltmeter/strain gauge were generated. These pairs were then used for the regression model that used temperature measurement as the input and the other measurement as the output. Furthermore, in 2018, Zhang et al. [140] developed a method for temperature compensation on cable force measurement. In this research, cable force measurement was conducted by using EM sensors. Each cable force measurement was accompanied by a temperature measurement and the actual force was obtained from the load cell reading on the hydraulic jack used to stretch the cable. The uncompensated EM sensor and temperature sensor readings were then used in the supervised training of ANN models and during the training process, the output of the ANN model was compared with the load cell measurement. Comparison was performed with temperature compensation based on polynomial fitting, and it was shown that higher performance was achieved by using the ANN model.

2.3.5 Multiple Linear Regression (MLR)

Linear regression offers powerful statistic tools for data analysis. The basic principle of linear regression is employing the linear relationship between a dependant variable y and an explanatory variable x [84], [103], [141]. MLR represents linear regression with two or more explanatory variables. The linear regression is formulated as:

$$y(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p \tag{2-9}$$

where β (β_1, β_2, …, β_p) are the regression coefficients that correspond to the explanatory variables x ($x_1, x_2, x_3, \ldots, x_p$). The regression coefficients are also known as weights. These coefficients are obtained by using the least squares method to minimise the error function, generally the mean square error. In a simple regression with one explanatory variable, only β_0 and β_1 are employed.

The weights in linear regression model determine the impact of the explanatory variables to the prediction [142]. As shown in (2-9), each explanatory variable x_i (x_1, x_2, x_3, …, x_p) is multiplied by its corresponding weight β_i (β_1, β_2, …, β_p) before producing a prediction of $y(x)$. When a variable x_i is connected to a positive β_i, the variable contributes positively to the prediction. On the other hand, if a variable is connected to a negative weight, then it will decrease the prediction value. The magnitudes of the weights define the influence of the corresponding explanatory variables. If a variable has a corresponding weight with a large magnitude, then the variable affects the prediction greatly. On the other hand, the variables whose corresponding weights have zero magnitude, produce no impact to the prediction.

In many engineering scenarios, a dependant variable is usually predicted using multiple parameters hence MLR is generally employed instead of linear regression [103]. MLR has been widely used for SHM application mainly for damage detection and environmental effects removal. Sohn et al. [143] proposed MLR-based damage detection using monitoring data from Alamosa Canyon Bridge. In the research, MLR models were trained to predict natural frequencies of the bridge using thermometer readings as the predictors. Observation collected from a test performed in 1996 was utilised for training data while dataset obtained in 1997 was used as the testing data. A threshold level was defined using 95% confidence interval and the measurements outside the interval were detected as abnormal data. In addition, Peeters et al. [144] also applied linear regression analysis in order to detect the presence of damage using data from Z24-Bridge (Switzerland) as the case study. In the research, data from the bridge were collected for one year before damage was introduced to the bridge. Linear regression models were trained to generate eigenfrequencies as the output by using temperature measurements as the input. In this research, simulation errors that were produced from the discrepancies between the actual and prediction values were compared with a threshold defined using 95% confidence interval. The method successfully detected the introduced damage from the presence of abnormal data exceeding the threshold. Furthermore, Deng et al. [145] implemented linear regression for condition assessment of Runyang Bridge. Similar to the previously mentioned research, in this research, linear regression models were trained using temperature data as the input to predict structural responses such as natural frequencies and displacement. For each prediction models, a threshold level defined using 95% interval was set for detecting anomaly.

On the other hand, MLR has also been implemented for investigating the environmental influence on the structural responses. Liu and DeWolf [146] observed the influence of temperature on natural frequencies of a concrete bridge in 2006. In this research, estimation of the first three natural frequencies of a curved concrete box girder bridge was generated using linear regression models that employed temperature measurements as the input. It was concluded that natural frequencies are sensitive to temperature hence temperature should be taken into consideration in performing damage detection. Ding and Li [147] investigated the environmental effects including wind and temperature loading on the variation of displacement measurement on Runyang Bridge. In the research linear regression was applied to analyse the correlation between the environmental effects and the bridge response. It was shown that for Runyang Bridge, temperature influences displacement measurement significantly while the wind loading has weak correlation with the displacement. Furthermore, In 2014, Kromanis et al. [103] compared four machine learning techniques including MLR, ANN, SVR, and robust regression for compensating thermal effect from measurement of structural responses. In this study, regression models were employed to estimate structural responses by using temperature data as the input of the models. Initially, data were processed to remove outlier and noise. This process was followed by performing PCA on the processed data for feature reduction. Using the selected features, regression models were trained in supervised way with the objective of estimating structural response using temperature data.

Other applications of MLR in SHM is for identifying correlation between monitored parameters. In 2003, Li et al [148] employed MLR for fatigue assessment of a bridge. In the research, strain gauge data collected from Tsing Ma Bridge monitoring system were used for the analysis. In total six strain gauge measurements were picked in the analysis. Initially, all time histories were transformed into frequency domain data by using FFT. Then, MLR was applied on the transformed data. In this case, the regression model estimated one strain gauge measurement in frequency domain using the other five transformed strain data. Other statistical methods including the median, mean, and mode methods were utilised as a comparison and it was shown that the MLR produced the closest response to the original data. Mata et al. [149] also implemented MLR to observe correlation between the structural response and temperature. In this research MLR models were trained to estimate the value of displacement and temperature using water height data as the input. The residuals obtained from the differences between the predictions and the actual values for these two parameters were then calculated. The time-frequency analysis was then performed on the residuals in order to identify the correlation between these parameters.

2.4 Limitation on conventional methodologies for data-based interpretation

Despite the potential, there are some challenges in implementing the data-based interpretation methods. In the method, feature selection is one of the most important aspects and the quality of the features could significantly influence the interpretation results. However, finding the useful features for one problem requires high level of knowledge on the problem under investigation. In addition, most of the conventional methods relies on hand-crafted features where useful information from the data is extracted manually based on the researchers' experience. This could result in biased estimates or losing important information from the data. In addition, it might be difficult to explore new features that might improve the interpretation.

In addition, the data-based interpretation methods might suffer from the sensitivity of the hand-crafted features. In damage detection, some features might be sensitive to other parameters aside from damage. For example, some research employs data-driven method that utilises manually extracted natural frequencies [134], [150]–[152]. However, it has been reported that the parameter is also sensitive both to the traffic condition and surrounding environmental condition. It has been shown that the temperature could cause daily frequency variation of 5%, and over 10% seasonal frequency variation [18], [153], [154]. The temperature induced shifts in the frequencies are often falsely identified as damage in the structure [155], [156]. In addition, frequencies might also be affected by the traffic loading. The variation in daily traffic loading could cause daily frequency shift over 5% [157]. Therefore, using natural frequencies as features without considering the sensitivity to other parameters might affect the accuracy of damage detection. It is desirable to develop a feature extraction method that can adaptively extract useful information by considering impact from other measurements.

Furthermore, modern SHMs generally employ large number of sensors and cameras in monitoring the structure's condition comprehensively. As a result, the system generates large amount of data over time. Due to the huge size of monitoring data, it is extremely challenging to perform feature extraction from the monitoring data manually. In addition, the more sensors/cameras applied in the system the larger the data dimension which might provide further obstacle in implementing conventional feature-based interpretation. This might impact the quality of the extracted features which might further influence the interpretation results.

2.5 Deep Learning

During the past decade, scalable machine learning techniques that can process large amounts of high-dimensional data have transformed many sectors, such as automated driving, medical imaging, and natural language processing. In particular, deep learning methods have replaced conventional feature-based machine learning by automating the feature extraction process and reducing the requirement of human domain expertise [158], [159]. Deep learning achieves this by employing many layers of processing stages hierarchically [142], [158], [160]. There are some deep learning architectures that have been reported such as CNN, restricted Boltzmann machine, deep belief network, and autoencoder. Among the architectures, CNNs are considered as the most successful deep learning model [158], [160].

2.6 Convolutional Neural Networks (CNNs)

Currently, deep learning has gained popularity in the image processing field, particularly due to the rise of Convolutional Neural Networks (CNNs). CNNs have architectures which are inspired by part of mammalian brain which is known as the primary visual cortex in processing visual input [142]. Introduced in the early 1990's, CNN rose into prominence in 2012 when a CNN-based prediction model produced the best performance in ImageNet competition. Following this success, CNN has revolutionised the field of computer vision and become a state of the art for recognition and detection purpose [158]. In addition, it has been utilised in a wide range of applications such as image classification for a large number of classes [161]–[163], traffic sign recognition [164], medical object classification [165]–[167], face recognition [168], [169], and damage detection in structures [170]–[174]. In addition to automatic feature extraction, CNN can produce excellent performance for complex image recognition task due to the capability of CNN in exploiting the local spatial correlation between pixels in the image [158]. Unlike other image recognition algorithm, CNN depends on the spatial separation instead of the spatial position, hence, combination of local features is more important than the location of features which might be varied on images.

In general, the CNN architecture combines both feature extraction and classifier into one body as it can be seen in Figure 2.4. In the feature extraction stage, a series of combinations between convolution and pooling layers is generally applied.

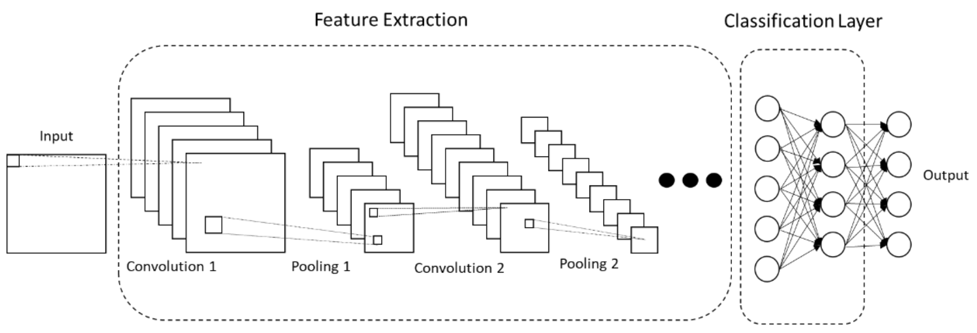


Figure 2.4 General architecture of CNNs that combines both feature extraction and classification in one body

2.6.1 Convolution Layers

In the convolution layer, a number of filters are moving around the input data while performing convolution to extract features from the data. This convolution operation sums the products obtained from element-by-element multiplication between the kernel and input array and is explained in (2-10):

$$f(i) = \sum_{n=1}^{vk} S(i+n)K(n) \tag{2-10}$$

where S is the input array, K is the kernel, and vk is the kernel size. Every filter employs kernel of equal weight to extract features from the input data. This is also known as weight sharing that provides efficient computation due to the use of fewer parameters. Depending on the type of the dimension on the input data, either two two-dimensional (2-D) CNN or one-dimensional (1-D) CNN is implemented. 2-D CNN is usually employed for processing images. This technique employs two-dimensional filters that move in two directions. On the other hand, 1-D CNN is primary utilised for time-series data or texts. This technique utilises one-dimensional filters that move in one direction. Figure 2.5 illustrates the operation performed in 2-D convolution layers.

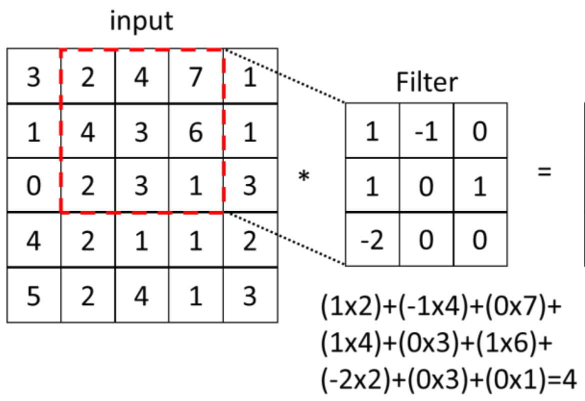
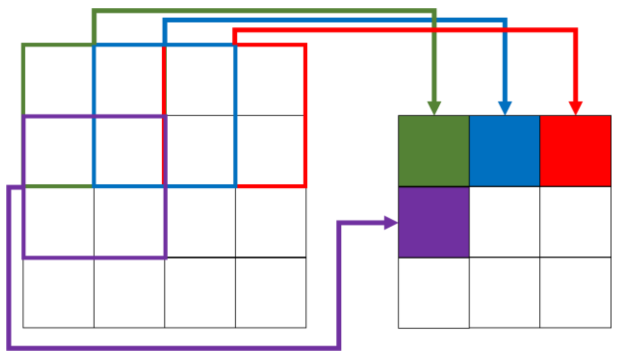


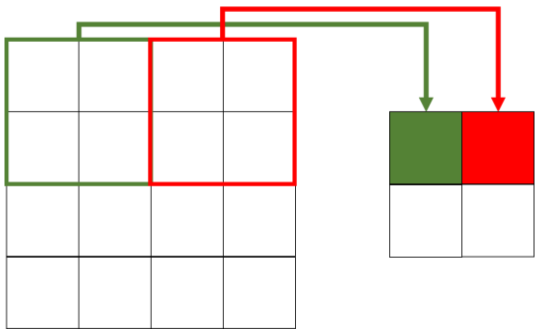
Figure 2.5 Illustration of 2-D convolution operation on a 5x5 input using a 3x3 filter. The red square in the results matrix represents the product of convolution operation between the filter and the red area in the input matrix.

There are some parameters that are involved in the utilisation of convolution layers such as filter size, filter number, stride, and padding. The filter size corresponds to the size of the receptive fields that are used to extract features. This parameter determines the number of weighs inside a filter. The filter number controls the number of filters utilised in each convolution layer. Each filter produces its own feature map which represents unique information from the input.

As it has been mentioned previously, CNNs employ filters that convolve around the input. Stride defines how far the filter moves over the input matrix. Figure 2.6 describes an example of the implementation of stride in the convolution layers. As shown in Figure 2.6 (a), when stride of 1 is implemented, the kernel only shifts for one unit over the input matrix. As shown in Figure 2.6 (a), initially, the filter performs convolution on the green area in the input matrix resulting to the green square in the output matrix. Then, the filter moves to the blue area in the input matrix and the product of the convolution operation is shown as the blue square at the output matrix. On the other hand, when stride of 2 is implemented, the filter directly moves to the red area in the input after performing convolution operation on the green area as described in Figure 2.6 (b). This results to smaller size of output matrix compared to the size of the output matrix when stride of 1 is implemented.



(a)



(b)

Figure 2.6 Example of stride implementation in 2-D convolutional layers utilising 2x2 filter; (a) Stride of 1; (b) stride of 2.

Padding is applied to increase the effective size of the input before entering the convolution layers. In CNN architecture, the first convolution layer reduces the size of the original input and the successive convolution layers might further decrease the size. Padding can be used to tackle this problem by adding fillers around the boundary of the input. In general, zero padding which adds zero on the boundary of the input is utilised. Figure 2.7 shows the utilisation of zero padding on a 3x3 input matrix. As it can be seen from the figure, padding increases the size of the input matrix from 3x3 into 5x5.

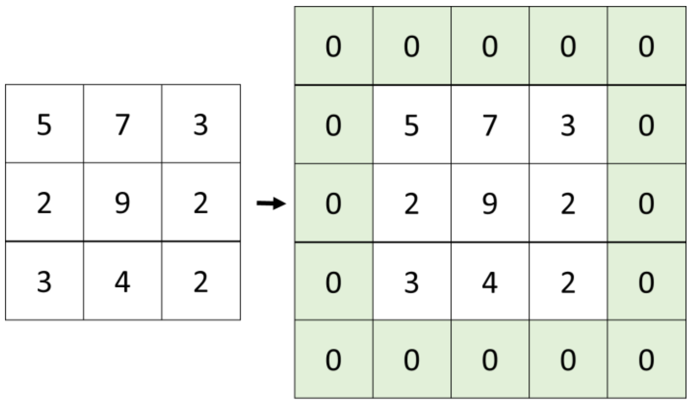


Figure 2.7 Example of zero padding implementation on a 3x3 input. Additional fillers (green boxes) are added on the boundary of the input.

2.6.2 Activation Functions

After completing the convolution process, non-linear activation function such as sigmoid, tanh, ReLU (Rectified Linear Unit), or LReLU (Leaky Rectified Linear Unit) is applied. The implementation of activation function in neural networks is crucial since it introduces non-linearity to the CNN models.

Figure 2.8 shows the comparison of activation functions that can be implemented for CNNs. Initially, sigmoid and tanh activation functions were commonly used in neural networks. Figure 2.8 (a) and Figure 2.8 (b) illustrates the sigmoid and tanh activation functions, respectively. As it has been mentioned in section 2.3.4, both the sigmoid and tanh activation functions suffer from vanishing gradient problem. In backpropagation algorithm, the error is transmitted backward, and the gradient information in each layer is employed in updating the weighs on the corresponding layer. Due to the vanishing gradient problem that occurs on both sigmoid and tanh activation functions, it is challenging to retrieve the gradient information. This situation becomes more severe when implementing deep learning models that employ many processing layers. As a result, it is challenging for the deep models to determine the direction where the parameters should be updated to reduce the error. To tackle this problem, ReLU activation function can be implemented. ReLU has been widely adopted as an activation function for deep learning models recently, due to the efficiency and simplicity [175]. The function is given as:

$$f(x) = \begin{cases} x, x \geq 0 \\ 0, x < 0 \end{cases} \tag{2-11}$$

As it can be seen in Figure 2.8 (c), ReLU activation function retains gradients for those in the right side of the function. This property helps in tackling the vanishing gradient problem that occurs in the sigmoid and tanh activation function. In addition, it also offers better training speed and cheaper computation compared to both sigmoid and tanh activation functions [161]. However, as it can be seen in the figure, in ReLU activation function, the gradient of those in the left side of the function is always zero. As a result, those on the left side of the function will never get activated. The Leaky ReLU is one of the improvements of ReLU. Unlike ReLU that produces output of 0 when x<0, LReLU still gives a small amount of information for the given input [175]:

$$f(x) = \begin{cases} x, x \geq 0 \\ \dfrac{x}{a}, x < 0 \end{cases} \tag{2-12}$$

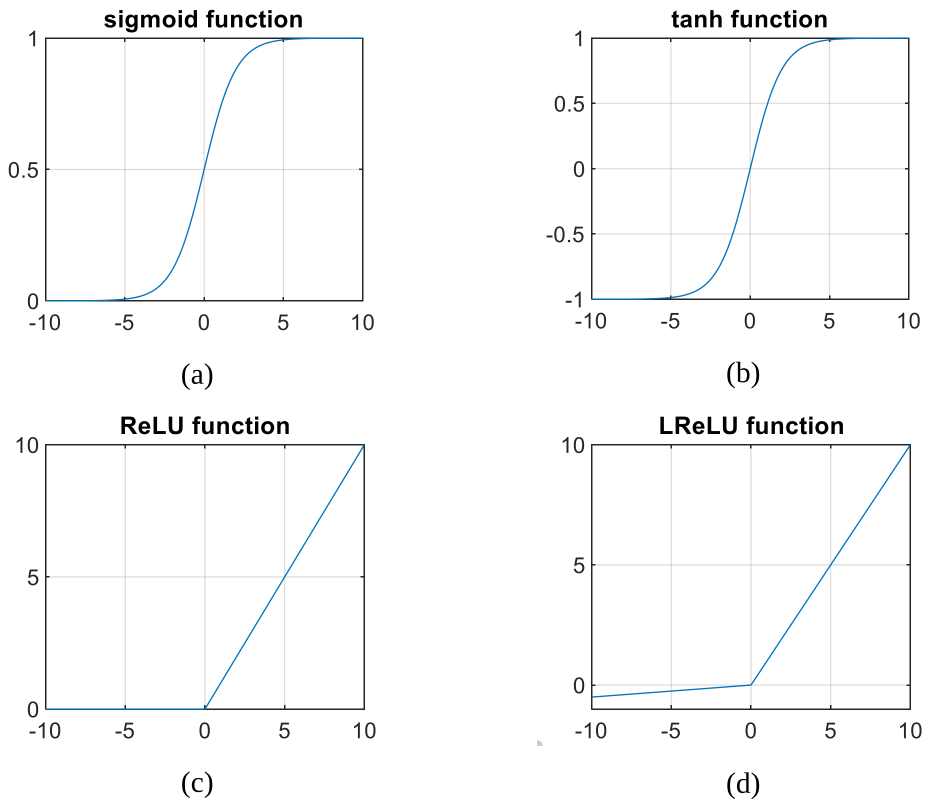


Figure 2.8 Comparison of activation function for CNNs; (a) Sigmoid function; (b) Tanh function; (c) ReLU function; (d) LReLU function.

As it can be seen in Figure 2.8 (d), the LReLU activation function keeps the gradients for those in the right side of the function while maintaining small bits of information for those in the left side of the function.

2.6.3 Pooling Layers

The pooling layers are typically included in CNN architectures. This layer performs sub-sampling operation that reduces the dimension of feature maps generated from the previous layer. Pooling layer operates by replacing regions in the feature maps with a summary statistic value that represents the regions. By implementing pooling layers, CNN architectures can produce more efficient computation. Figure 2.9 illustrates the operation performed on the pooling layer.

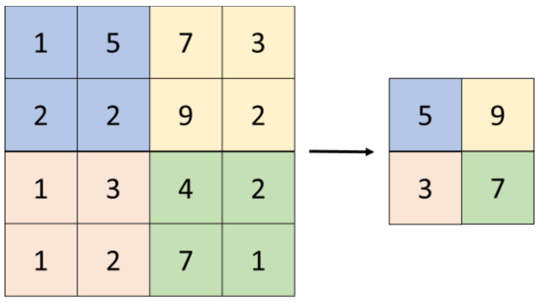


Figure 2.9 Example of 2x2 max pooling on a 4x4 input matrix. After the pooling layer, the dimension of the input is reduced into 2x2.

In general, there are two methods that are used for pooling: max pooling and average pooling [176]. Max pooling replaces a region in the feature maps with a maximum value of that region. On the other hand, average pooling modifies a region in the feature maps with the average value of that region. Parameters involved in the pooling layers include stride and filter size.

Finding the pooling method that works better for a new problem is still a subject of research interest. In addition, for civil engineering application, research investigating the comparison of pooling methods has not been yet reported. However, research implementing max pooling on CNN models for civil engineering application has been conducted [170], [174], [177], [178]. In the feature extraction stage of CNN architectures, generally the combination between convolution layer and pooling layer is repeated depending on the problem complexity.

2.6.4 Regression/Classification Stage

After the feature extraction stage, all extracted features are flattened and fed into a classifier or regressor. Typically, this stage consists of a standard Multi-layer Perceptron (MLP) which is made of a number of fully connected layers as it can be seen from Figure 2.4. The process performed in the fully connected layers is similar to the process in ANN that has been mentioned in (2-6). Activation functions such as sigmoid, tanh, ReLU, and LReLU can be implemented in the hidden layer of the regressor or the classifier.

In the example described in Figure 2.4, the output layer consists of four nodes, each of which represents the number of classes for a problem under study. By utilising the features extracted from the previous layer, the CNN model will generate output depending on whether regression or classification is performed. The regression layer generates continuous values as the output. On the other hand, the classification layer produces probabilistic values in the output layer. For this purpose, softmax activation function is implemented as shown in (2-13):

$$y_i = \frac{\exp(u_i)}{\sum_{i=1}^{n} \exp\,(u_i)} \tag{2-13}$$

2.6.5 Implementation of CNN in Structural Health Monitoring

Most of the research implementing CNNs for SHM is aimed to perform damage detection on structures. Recent research efforts have been made in automatic defect detection by utilising CNNs in image processing. In 2017, Cha et al. employed CNN technique for crack detection on concrete surfaces [174]. In this research, 40,000 images were used to train a CNN model and 55 testing images were utilised to observe the model performance. Comparative study with the traditional Canny and Sobel edge detection method was conducted to observe the performance of the proposed method. The proposed system produced better capability in sensing thin crack compared to the conventional method. Furthermore, Protopapadakis et al. proposed an automatic robotic inspector for tunnel condition monitoring in 2016 [173]. In his work, CNN was employed for visual inspection of the robot. By utilising CNN, high-level discriminative features for complex non-linear pattern classification were produced. These features later were used to calculate real-time 3D information to identify the crack position and orientation. In addition, studies for crack detection on pavement using CNN have been conducted [170]–[172]. In these studies, pavement images were employed to train neural networks. However, unlike [171], [172] which trained the neural networks from scratch, in [170] transfer learning using a pre-trained VGG 16 network was performed. Therefore, in the research a pre-trained model was finetuned for the new prediction task. Both studies managed to detect the presence of crack in the pavement images.

Recently, research utilising transfer learning using pre-trained CNN models for damage identification in structures using visual data has been attempted. In 2021, Ali et al. evaluated the performance of five CNN-based models for crack detection on concrete structures [179]. In this research, the performance of a CNN model trained from scratch was compared with other pre-trained models including VGG-16, VGG-19, ResNet-50, and Inception V3 models, that were fine-tuned using transfer learning technique. It was found that compared to the other models, the proposed model managed to produce comparable accuracy with the least processing time. In addition, Kim et al. proposed a crack detection method utilising a shallow CNN model [180]. In this research, Lenet-5, a shallow pre-trained CNN model that was originally built for handwritten digit recognition, was fine-tuned in order to be used for crack detection. For this purpose, two convolutional layers were added to the original LeNet-5 and all activation functions in the original model were replaced by ReLU function. For comparison, transfer learning was conducted on other pre-trained models such as VGG16, Inception, and Resnet. The proposed model achieved prediction accuracy of 99.8% while requiring the minimum computing time compared to the other models trained using transfer learning approach.

In addition to processing image data, CNN has shown promising capabilities in processing one dimensional signals. In 2017, Jing et al. [181] proposed a 1-D CNN-based gearbox fault detection method. Seven labels representing seven conditions were implemented in the research. The proposed method was compared with other machine learning methods such as ANN, SVM, and Random Forest and the proposed network outperformed other methods. In addition, in 2018, Zhang et al. [182] proposed a CNN-based fault detection method to detect bearing fault by using vibration data. Comparison was made between the proposed method and other machine learning methods such as SVM, MLP, and ANN. To improve the robustness and stability of the method, ensemble learning was applied. It was shown the proposed method outperformed other methods both in noisy and noise-free environments. Abdeljaber et al. [177] utilised a steel frame stadium structure to experimentally verify the CNN-based damage detection method employing vibration data that was proposed in their research. The proposed method achieved a high level of generalisation in damage detection. Moreover, Abdeljaber et al. [183] employed CNN technique for damage detection on four story steel structure in 2018. Out of nine damage scenarios considered, they utilised only two datasets representing the structure in its healthiest and most damaged condition in the training step. By implementing a metric called “Probability of Damage”, the proposed method managed to quantify the severity of damage in all damage scenarios.

While all the above-mentioned research utilises raw measurement data for damage detection, research implementing CNN for damage detection by using compressed data has also been reported. In 2020, Azimi and Peckan investigated the applicability of damage detection on structures using extremely compressed data [184]. Some types of sensor store data in a compressed form rather than in the form of raw measurement signals. The research was carried out in two stages. In the first stage, a CNN model was trained to detect damage by using compressed data in the form histogram of events. In the next stage, more compressed data were extracted from the histogram. This was performed by calculating the mean, standard deviation, and scale factor from the histogram. By using these data, new histograms were reconstructed, and transfer learning was performed CNN model that was previously trained by using the original histograms. The method was validated using data from a benchmark dataset and it was shown that the CNN model trained using the reconstructed histograms might achieve prediction accuracy of 91.9%.

For damage detection on bridge structures, numerous studies employing 1-D CNN have been reported. Lin et al. [178] implemented 1-D CNN for damage detection using data collected from a numerical model of a simply supported beam. In this study, damage assessment was conducted using the numerical model only. The numerical model was split into 10 parts and damage was simulated at one part at a time by reducing its stiffness. A CNN model was then trained using simulated acceleration data. It was shown that the prediction model successfully detected the damage and its location. In addition, Zhang et al. proposed a vibration-based damage detection method utilising CNN [185]. In this research, acceleration data were utilised as the input of CNN model. Three case studies were investigated, and in each case study, acceleration data were collected from the structure's undamaged and damaged condition. In each case study, the data were labelled based on the location of damage and CNN models were trained to predict the damage location. The proposed method managed to achieve high prediction accuracy in determining the damage location on all case studies.

Most of the above-mentioned research relies heavily on the availability of labelled data to perform supervised training on CNN models. However, in practice, these labelled data might not be available. One potential solution that has been performed to tackle this problem is to train CNN models using labelled data generated from a calibrated physical model. In 2020, a damage detection method combining Gapped Smoothing Method (GSM) and 2-D CNN was proposed [186]. The main idea of this research is to train a CNN model by using labelled data collected from a calibrated FEM model of a structure and apply real monitoring data on the trained CNN model. This study utilised Bo Nghi Bridge for the case study. In this research, an FEM model of a single bridge girder was employed to generate training data. In the model, the girder was divided into 50 elements and various damage scenarios were introduced by applying several levels of stiffness reduction for each element in the model. From these scenarios, acceleration data were generated, and the GSM method was employed to extract the damage indices from the first three vibration modes. The extracted data were labelled based on the damage location and the labelled data were used to train a CNN model. For generating testing data, a more complex FEM model of the bridge consisting of four girders was used. 100 damage scenarios were utilised in the testing data and the CNN model trained using data generated from an FEM of a single girder might produce average detection accuracy of 82% on data generated from a more complex model. Despite the promising potential, this method requires developing a behaviour model of the monitored structure, leading to the need of structural engineering expertise.

Aside from the implementation of CNN for damage detection method, CNN has also been implemented for data compression in SHM application. In 2020, Ni et al. proposed a data compression method to reduce the size of SHM data by using CNN [187]. In this research, data compression method was carried out in two stages such as anomaly detection and signal reconstruction. In the first stage, a CNN model was trained to differentiate between normal data and anomalies. This step is crucial since abnormal data might affect the reconstruction of the data significantly. In this research, a CNN-based model was trained to detect the anomalies in supervised way by using data that were labelled based on the normality. In the first stage, the CNN model was evaluated based on how accurate the model in differentiating between normal and abnormal data. In the second stage, only normal data were used both to train a CNN-based autoencoder and to evaluate the compression method. In this research, the autoencoder consisted of an encoder and a decoder, both having five convolutional layers. In evaluating the compression method, the correlation coefficient between the actual and reconstructed signal was employed. The method was applied on four sensor measurements obtained from a real bridge monitoring system. In the first stage, the CNN model managed to accurately predict the normality of data with prediction accuracy of 99.5%. In addition, for the second stage, the CNN-based autoencoder produced coefficient correlation in the range between 0.843 to 0.95.

2.7 Summary and Conclusion

In this chapter, a literature review on SHM and the current methodologies for SHM data interpretation has been presented. In particular, existing methodologies for data-based interpretation have been discussed in this chapter. Data-based interpretation approaches interpret the structure’s condition based on the pattern of the data, therefore they require no behaviour model of the monitored structure. However, conventional data-based interpretation methods require feature extraction which involves high level of understanding in the related domain. In addition, feature extraction might be challenging in SHM due to the high dimensionality of data as a result of the implementation of numerous sensors to monitor large structures. Deep learning techniques that have the potential in tackling the feature extraction problem as well as improving the current data-based interpretation approaches have been employed in some literatures. These literatures have been reviewed in this chapter and the research gaps have been identified as follows:

- Most studies on the utilisation of deep learning techniques for SHM that employ images as the input are limited to damage detection application. Further research is required in order to explore the potential of deep learning in processing images for SHM application.
- No research has provided comparison between the performance of deep learning-based damage detection method and the performance of damage detection approaches implementing conventional machine learning

techniques. Most studies have only provided the results from the deep learning implementation for SHM without providing the comparison with other machine learning methods.

- No work has been found on implementing deep learning method using different types of sensors. Most previous research implementing deep learning models only employ a single type of sensor as the input. On the other hand, generally various types of sensors are deployed in SHM and it is challenging to combine these sensors in performing data interpretation.
- No study has investigated the impact of hyperparameters of the deep learning models in the implementation of deep learning for SHM. Most research only presents results from a single model architecture without performing observation on the effect of the hyperparameters.
- Limited research is available on combining deep learning with existing damage detection for SHM. Further works are required to observe the potential of combining deep learning techniques with existing approaches.
- Most previous studies of the utilisation of deep learning in SHM have not been validated using real monitoring data.

Chapter 3 Deep Learning for Bridge Load Capacity Estimation in Post-Disaster and -Conflict Zones

Summary

In this chapter, A deep learning-based method for estimating the load carrying capacity of bridges is proposed and evaluated. Section 3.2 presents the methodology performed including the data collection (section 3.2.1), the prediction model training and testing (section 3.2.2), the performance evaluation (section 3.2.3), and the parametric study carried out in the research (section 3.2.4). Section 3.3 provides the result and discussion on the parametric study conducted in the research. In addition, an optimisation approach that enhances the performance of the proposed method is discussed in section 3.3. Finally, section 3.4 provides conclusions and summary of the chapter.

3.1 Introduction

3.1.1 Bridge Load Testing

Although government agencies generally manage database of infrastructure such as bridges, many unprecedented scenarios such as disaster or conflict might result to the loss of this information. It is important to understand the bridges condition to ensure public safety and improve the maintenance efficiency especially since many bridges are aging and deteriorating. One parameter that can be utilised for bridge assessment is the load carrying capacity. This parameter defines the load level that can be safely applied to a bridge during its lifetime [188]. The importance of this parameter in the bridge maintenance is inevitable since the presence of overweight vehicles might reduce the bridge service life [189].

There are two parameters that represent the load carrying capacity of a bridge such as load rating and design load. According to American Association of State Highway and Transportation Officials (AASHTO) “Manual for condition evaluation and load and resistance factor rating (LRFR) of highway bridges”, load rating is generally implemented as a parameter which defines the load carrying capacity of a bridge. Eurocode also provides guidelines on the load rating for steel (EN 1993-1-1: Eurocode 3), concrete (EN 1992-1-2: Eurocode 2), and composite structures (EN 1994-1-2: Eurocode 4). On the other hand, design load can be employed as another parameter which illustrates a bridge’s load carrying capacity. According to [188], design load represents the assumption of live load used when a bridge is designed. The design load is not affected by the bridge’s condition since this load level is only implemented in the design phase of a bridge.

In general, load rating is collected from visual inspection and load testing [95]. Although visual inspection can be performed rapidly, it suffers from subjectivity heavily thus load test involving the use of loaded trucks are generally conducted. However, the test is both time and resource consuming as the test often requires a pre weighed truck, instrumentation, and bridge closure. The challenges in the load test are summarised in Figure 3.1.

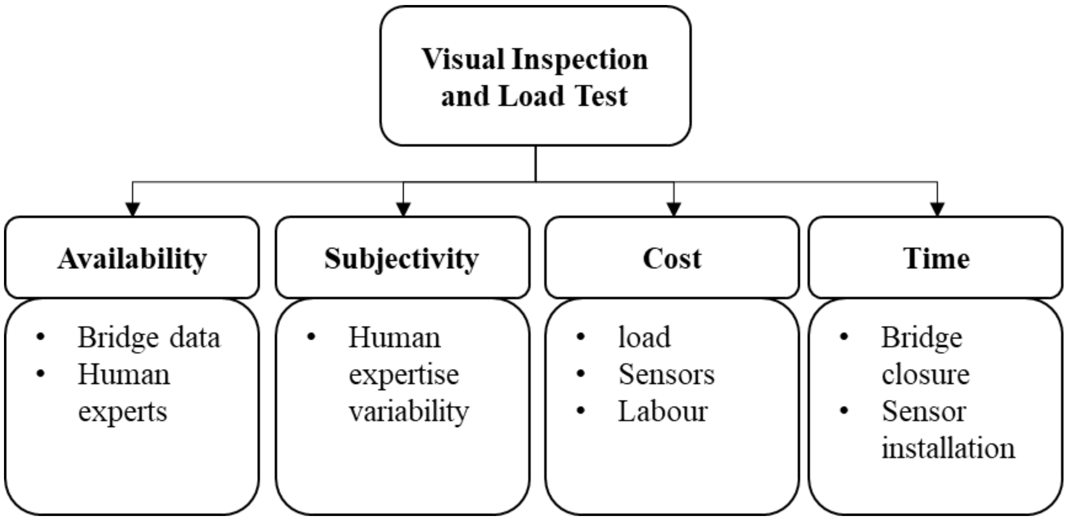


Figure 3.1 Challenges of visual inspection and load test

As it can be seen in Figure 3.1, there are clear limitations related to the aforementioned activities such as availability of resources, subjectivity, cost, and time. This is exasperated in post-disaster zones, where expertise is expensive and sparse, and the need to rapidly understand the capacity of bridges is critical for recovery. In this situation, end users (i.e., aid workers and military) have to prioritise which bridges to inspect with limited resources. Therefore, we are motivated to classify broad bridge capacity values as a precursor to closer condition inspection.

3.1.2 Summary of Novelty and Contribution

This chapter is based on a published work [184]. This study is primarily aimed at addressing the global development challenges where bridge capacity data are missing and hindering reconstruction. Current methods rely on human expertise through either visual inspection or testing. However, this is expensive, slow, and reliant on the availability of expertise. In this chapter, by utilising deep learning as an automatic structure identification method, we have created a cheap, scalable, transparent, and verifiable bridge load identification tool. In time, this method can be standardized and used on smartphones by non-experts. The propose method will require no physical model of a bridge and thus can provide solution for countries that are lacking in civil engineering experts or bridge documentation. In addition, the trained CNN can benefit other researchers through transfer learning.

3.2 Methodology

In this research an alternative method for estimation of load carrying capacity of bridges is proposed. This method utilises bridge images obtain via crowdsourcing as the input and provide an estimation about the load carrying capacity in the form of either load rating or design load. The main activities that have been performed in this project include:

1. Data collection of bridge images and load ratings of the bridges
2. CNN training and testing
3. Evaluation of performance and optimisation

3.2.1 Data collection

In this part of our study, a bridge database and corresponding crowdsourced bridge images were collected. Figure 3.2 shows the data collection steps performed in this study. As it can be seen in the figure, there are two data sources employed in this study: The National Bridge Inventory (NBI) Database published by American Federal Highway Administration (FHWA) [190] and a website called Bridgehunter [191]. The NBI database provides information such as inventory number, location, features, design load, construction, condition, and load rating about all bridges in the U.S. Both the load rating and design load information provided by the NBI database is employed for labelling purpose.

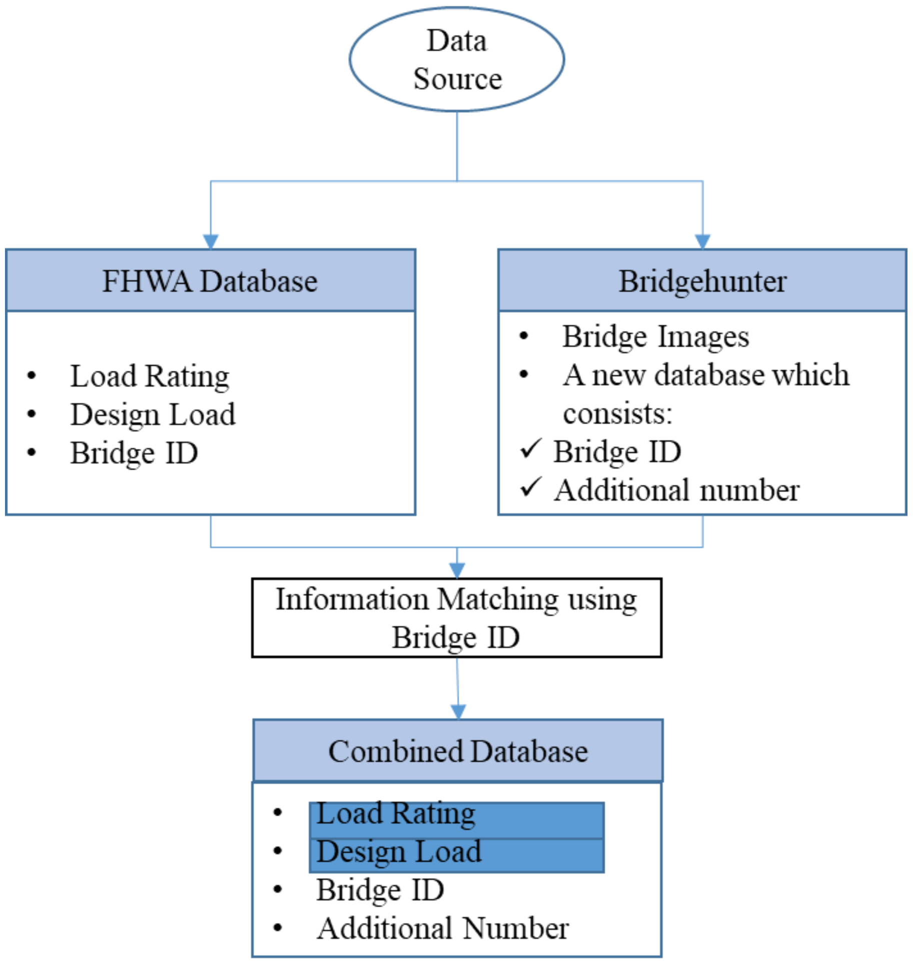


Figure 3.2 Details of data collection steps consisting of database collection and images collection through web scraping

On the other hand, bridge images were collected through web scraping on a website called www.bridgehunter.com [191]. For this purpose, an interface program for collecting images from the site was developed using Python. In this research, 54458 bridge images from 6753 bridges have been successfully collected and stored. Figure 3.3 shows the images obtained from web scraping. In addition, inventory number and the state ID of these bridges were also collected and stored in a new database (referred to as the bridgehunter database).

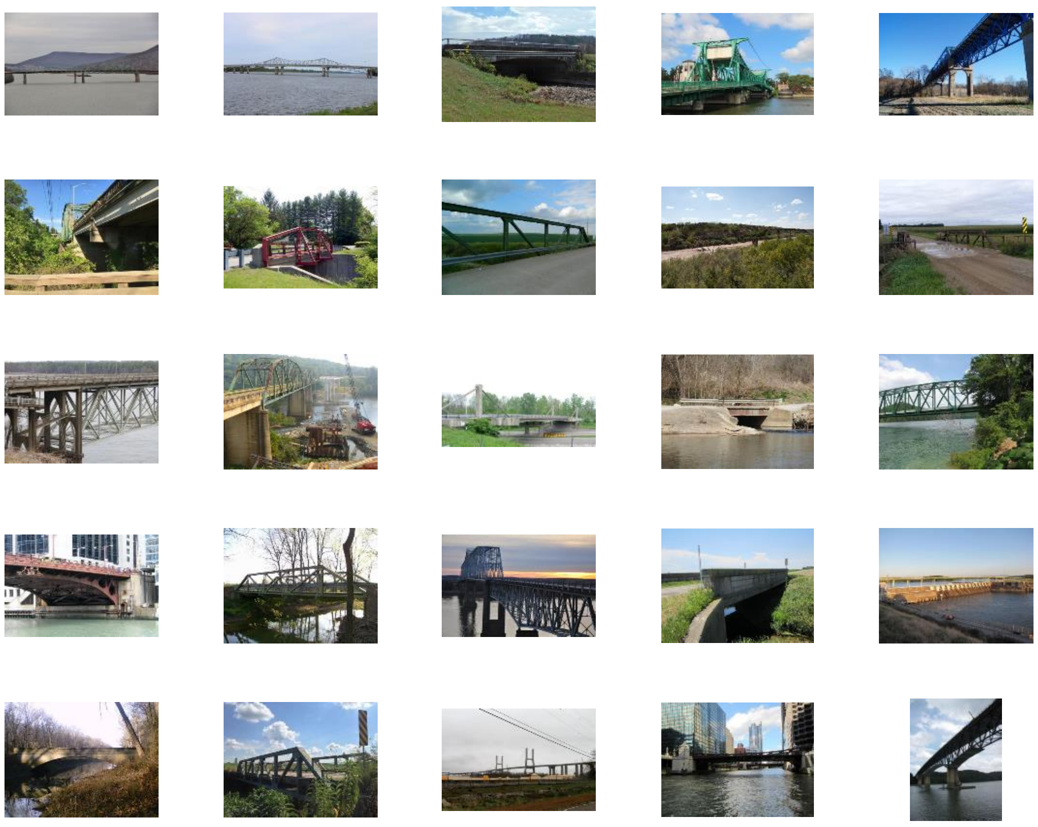


Figure 3.3 Some bridge images collected from web scraping

The next step performed in the data collection is combining the NBI database with the bridgehunter database. This step is essential especially for labelling purpose. As it has been previously mentioned, the both the NBI and bridgehunter database contains both the inventory number and the state ID of the bridges. As it can be seen in Figure 3.2, this information was employed as matching information to combine the databases. State ID was used since different bridges from different states, might have similar inventory number in the database. Hence, by using this method, information about load rating and design load for each image has been obtained from NBI database. From this step a combined database consisting of the load rating, design load, and bridge ID has been generated. In addition, an additional ID number was assigned on each bridge as an identifier of the crowdsourced images. From this step, all bridge images were stored in their respective folders that have been named according to this additional ID number.

Due to the difference in the database formatting among states, it is challenging to directly use the information from the NBI database on the bridgehunter database. In this case, the data in the NBI database has to be processed before it can be used as a matching information with the data from the bridgehunter database. For example, in NBI database, each bridge from state 10 is assigned with 8-9 characters as the bridge ID. These IDs match perfectly with the corresponding IDs in the bridgehunter database. In this case, the load carrying capacity information might be collected directly from the NBI database. On the other hand, the bridge ID from state 13 consists of 15 characters where the first seven characters are zero. However, in the bridgehunter database, the first seven zero characters in the bridge ID are removed. Therefore, data processing was performed to remove the first seven zero characters from the bridge IDs on the NBI database so they can match the bridge IDs in the bridgehunter database.

In addition, some bridges in the bridgehunter website have no information about the bridge ID which make it difficult to obtain the information of the load carrying capacity from the NBI database. Table 3.1 shows the number of samples according to the availability of the load carrying capacity information. Notice from Table 3.1, there are some samples that have no label. This condition occurs due to the unavailability of either load rating or design load data for the corresponding bridges in NBI database. Due to this problem, some samples can only be used for developing either load rating or design load prediction model.

Table 3.1 Summary of image attributes used for load rating and design load prediction

Remark	Total Number of images	Images showing a full picture of a bridge	images showing only parts of a bridge
All Data	54458	N/A	N/A
Images for Design Load Estimation	18821	7837	10984
Images for Load Rating Estimation	43180	13510	29670

Figure 3.4 shows the proposed load carrying capacity estimation method. First, each image is labelled using the information of either the load rating or the design load. Then, the labelled images are fed into the CNN model that has been trained using supervised training process. The CNN model then produces an output of load rating or design load estimation of the bridge. The output of the prediction is then compared with the actual value for validation purpose.

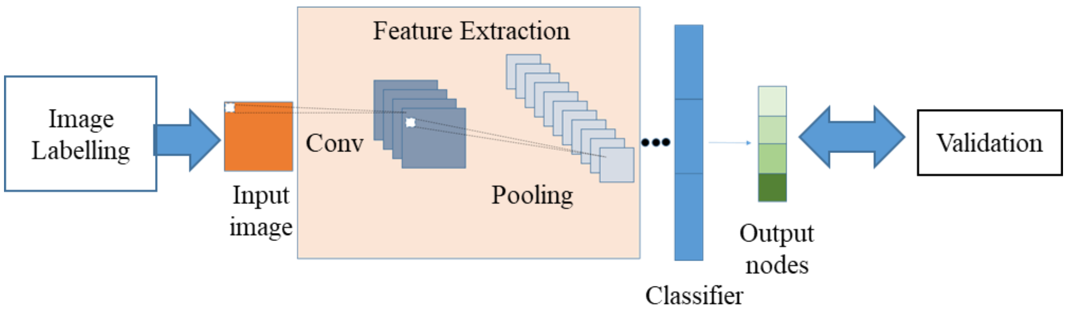


Figure 3.4 Proposed method for estimation of bridge's load capacity from images

3.2.2 CNN model training and testing

Instead of training a CNN model from scratch, in this research transfer learning method was performed. In this way, a pre-trained CNN model is finetuned in order to suit our application. The implementation of transfer learning using pre-trained models such as AlexNet, VGG-16, and GoogLeNet has shown a great potential in solving wide domain of image classification purpose [166]. It has been shown that, certain models created using transfer learning technique can produce better performance than models that are built from scratch [165], [166]. In this research AlexNet [161] was utilised as a pre-trained model for transfer learning. Table 3.2 shows the architecture of AlexNet.

AlexNet is a CNN-based pre-trained model that has been trained to classify 1000 objects. The model combines both feature extraction and classification function in one body. The feature extractor, from layer 1 to layer 8 in Table 3.2, automatically obtains important features for load capacity prediction and the classifier part of the model, from layer 9 to layer 12 on Table 3.2, performs prediction of load capacity using the extracted features. For the purpose of transfer learning, the output layer of the CNN model was modified according to the number of classes in the dataset we were working with. It was performed by modifying the number of connections at the last fully connected layer to match the number of classes in the dataset. For example, when performing classification of five classes, then the output layer is changed from the original 1000 nodes into five nodes.

Table 3.2 Details of layers in AlexNet architecture

Layer	**Type**	**Number of Kernel**	**Kernel Size**	**Stride**	**Padding**	**Activation**
0	Input	3	227 x 227	-	-	-
1	Convolution	96	11 x 11 x 3	4	0	Relu
2	Max Pooling	-	3 x 3	2	0	-
3	Convolution	256	5 x 5 x 48	1	2	Relu
4	Max Pooling	-	3 x 3	2	0	-
5	Convolution	384	3 x 3 x 256	1	1	Relu
6	Convolution	384	3 x 3 x 192	1	1	Relu
7	Convolution	256	3 x 3 x 192	1	1	Relu
8	Max Pooling	-	3 x 3	2	0	-
9	Fully Connected	4096	-	-	-	Relu + Dropout
10	Fully Connected	4096	-	-	-	Relu + Dropout
11	Fully Connected	1000	-	-	-	-
12	Softmax	-	-	-	-	-

To obtain training and testing data, the original dataset was randomly split into 80% and 20% for training and testing data respectively. In addition, pre-processing of images was conducted to modify the size of these images so they might be fed into the CNN model. After this pre-processing step, the size of all images was altered to 227 x 227 x 3 to match the input size of AlexNet.

In this research, the transfer learning process was performed using the pre-trained Alexnet model provided in Matlab. The transfer learning step was performed by modifying the last layer of the model using Matlab. The training process for one model took approximately one day. This computation time was reduced into only 10 minutes by implementing Nvidia GTX 1060 for GPU computation.

3.2.3 Performance evaluation

To evaluate the performance of the CNN models, common performance metrics for classification problem such as accuracy, precision, recall, and F1 score were measured for each prediction model. The accuracy, precision, recall, and F1 score are given by (3-1), (3-2), (3-3), and (3-4) respectively [192]:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}} \tag{3-1}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3-2}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3-3}$$

$$\text{F1 score} = 2 \times \frac{\text{precision} \,.\, \text{recall}}{\text{precision} + \text{recall}} \tag{3-4}$$

where TP is true positive, FN is false negative, TN is true negative, and FP is false positive. Accuracy defines the proportion of the samples that can be correctly predicted by the model. In addition, precision and recall improve the assessment of the model performance. Precision provides information on the proportion of the true detection reported by the models. It shows how accurate the model on predicting the relevant events. According to (3-2), the precision shows the proportion of the samples that are actually positive out of those that are predicted as positive. On the other hand, recall represents the fraction of relevant events that can be correctly predicted. As illustrated in (3-3), recall gives information about the number of positive samples that can be correctly predicted as positive. Out of those samples that are actually positive, recall represents the fraction of samples that are correctly predicted as positive. Finally, F1 score is a parameter that combines both precision and recall metrics as it is shown in (3-4).

3.2.4 Dataset Modification

In this research, dataset variation and its impact to the prediction performance was studied. Variations investigated in this research were the variation in number of classes, bridge images completion, and colour. The performance of the prediction model trained using these datasets was investigated.

3.2.4.1 Variation in Prediction Class

As it has been mentioned previously, the proposed system utilises a bridge image to provide estimation about either its load rating or design load and this estimation is given in range of loading. This range of loading is defined by prediction classes and classification task is performed by training prediction models. In this part of the research, CNN-based prediction models were trained by varying the number of classes for classification. This was performed to find dataset configuration which produces the highest performance in predicting either load rating or design load of bridges. In addition, the effect of imbalanced dataset was also investigated. For this purpose, a number of datasets were generated. In our study, LRx represents datasets that are labelled with load rating information while DLx represents datasets that are labelled using design load information.

Data distribution of samples according to their load rating can be seen in Figure 3.5. In this research, estimation of load rating was performed by first discretising the load rating value to obtain class labels. By using these labels, multiclass classification was performed for the estimation. To create dataset variation for load rating prediction, the class range was modified. In addition, the class interval modification took into account the number of samples obtained in each class. If one class was only formed by small number of images, this class was combined with the class adjacent to this class. As an example, in both dataset LR1 and LR2, due to small number of samples with 0-5 ton of load rating, the 0-5 ton class was merged with the 5-10 ton class. From this modification, datasets were created and can be seen in Table 3.3.

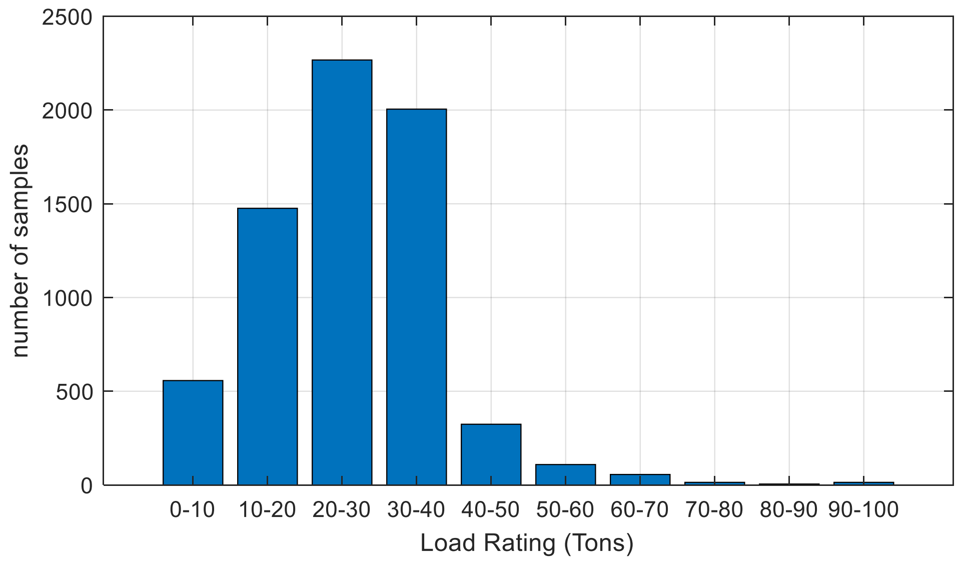


Figure 3.5 Distribution of samples based on the bridge's load rating

Table 3.3 Description of datasets for load rating prediction. In total, eights datasets are generated.

Dataset	Total Number of Classes	Class Description
LR1 and LR2	7	1. 0-10 ton 2. 10-15 ton 3. 15-20 ton 4. 20-25 ton 5. 25-30 ton 6. 30-35 ton 7. >35 ton
LR3 and LR4	5	1. 0-10 ton 2. 10-20 ton 3. 20-30 ton 4. 30-40 ton 5. >40 ton
LR5 and LR6	3	1. 0-15 ton 2. 15-30 ton 3. >30 ton
LR7 and LR8	3	1. 0-20 ton 2. 20-40 ton 3. >40 ton

As it is shown in Figure 3.5, the data obtained for this research creates an imbalanced dataset. Therefore, for each dataset in Table 3.3, a balanced dataset is produced by having a down-sampling process of the majority class in the respective imbalanced dataset. These balance and imbalanced dataset were used to train prediction model and the impact was observed.

Unlike the load rating data, the design load information has already been discretised in the NBI database according to [193]. In developing design load prediction model, several datasets were utilised by varying the number of classes used for the prediction. These datasets are shown in Table 3.4. Dataset A was obtained as the original dataset for design load prediction. The class number was slightly modified from the class number given in [193]. It was done to sort these classes in an ascending order. This dataset consists of 12 classes and as it is shown in Table 3.4, this dataset is imbalanced where a lot of samples are obtained for class 2, 3, and 5 and only few samples obtained in other classes especially for class 7, 8, and 11 which only have less than 100 samples. Hence, this dataset was not utilised in the research and other datasets were generated to overcome this problem. In DL1, class 7, 8, 11, and 12 were removed from the dataset due to the limited number of samples. Therefore, this dataset only classified 8 classes. In addition, to deal with data imbalance, in DL2 down-sampling of majority classes was performed. In this dataset, 1000 of samples from each majority classes (category 2, 3, and 5) were randomly picked. Due to this process, this dataset only had 5774 samples.

Unlike DL1 and DL2 that removed samples from class 7, 8, 11, and 12, both DL3 and DL4 combined these samples into one class. Therefore, these datasets were used to classify 9 classes. DL3 utilised all samples while down-sampling process was applied on DL3 to generate DL4 in order to create more balanced dataset. Finally, both datasets DL5 and DL6 were created by combining samples from class 5, 6, and 9 into one class. It was performed due to the similar load level applied to these classes. Similar to previous datasets, DL5 had imbalanced sample distribution while DL6 was generated as the balance version of DL5.

Table 3.4 Description of datasets for design load prediction. Due to the small number of samples in some classes such as class 7, 8, 11, and 12, six datasets are generated (DL1, DL2, DL3, DL4, DL5, and DL6) from the original dataset A.

Class	**Remark**	**Dataset**						
		A	**DL1**	**DL2**	**DL3**	**DL4**	**DL5**	**DL6**
1	10 ton	928	928	928	928	928	928	928
2	15 ton (3000 front 12000 rear)	4674	4674	1000	4674	1000	4674	1000
3	20 ton (4000 front 16000 rear)	1913	1913	1000	1913	1000	1913	1000
4	27 ton (3000 front 12000 mid & rear)	460	460	460	460	460	460	460
5	36 ton (4000 front 16000 mid & rear)	3991	3991	1000	3991	1000	5067	1000
6	Equal to HS20 with the inclusion of military loading	491	491	491	491	491	0	0
7	Pedestrian	3	0	0	0	0	0	0
8	Railroad	56	0	0	0	0	0	0
9	Equal to HS20 with an addition of road calculation	585	585	585	585	585	0	0
10	45 ton or greater	310	310	310	310	310	332	332
11	Greater than HL93	22	0	0		0	0	0
12	Other bridges outside AASHTO standard	107	0	0	188	188	0	0
Total		13540	13352	5774	13540	5962	13540	5962

3.2.4.2 Design Load Prediction and Load Rating Prediction Models Comparison

In order to create a fair comparison between load rating prediction model and design load prediction model, proprietary datasets were generated. In these datasets, the load rating class intervals were modified to match those in the design load. These generated datasets can be seen in Table 3.5. In both DL7 and LR9, all images which can be labelled for either load rating or design load prediction were used. However, it can be seen in the table that there is a discrepancy between the number of images available for load rating prediction and the number of images applicable for design load prediction. In addition, it can be seen that both DL7 and LR9 are imbalanced dataset. Therefore, both DL8 and LR10 which have equal number of samples in every class were created. In order to reduce a prediction bias toward a majority class in imbalanced dataset, both DL9 and LR11 were generated. As it can be seen from Table 3.5, these datasets only have 300 samples on each class due to small number of samples in class 6 available for design load prediction.

Table 3.5 Description of datasets for comparison of load rating and design load prediction models

Class	**Number of samples**			
	DL7	**LR9**	**Both DL8 and LR10**	**Both DL9 and LR11**
1	496	2098	496	300
2	2496	1153	1153	300
3	1201	1358	1201	300
4	287	2483	287	287
5	3171	2802	2802	300
6	186	920	186	186

3.2.4.3 Image Completion Variation

The images obtained from bridgehunter website contain both images showing a complete view of bridges and images showing incomplete bridge (only shows bridge connection, bridge railing, bridge deck, or bridge column). These variation in the samples could affect the performance of the prediction model therefore in this part of the research this impact was studied. Figure 3.6 shows samples that insufficiently represent a bridge. In order to filter these images, a CNN-based prediction was trained to classify whether or not an image is showing view of a bridge completely. From Table 3.1, it can be seen that the number of images which shows complete bridges is lower than the number of samples which shows incomplete bridge. This can be seen as one of the limitations of this research since these images were obtained from the web thus it is challenging to control their quality. Several datasets were created: dataset which only contained samples showing complete bridges, dataset which only contained images showing incomplete bridge, and dataset which contained all bridge images. To make a fair comparison, these datasets were configured to have similar number of samples hence down-sampling process was implemented. Table 3.6 shows the generated datasets.

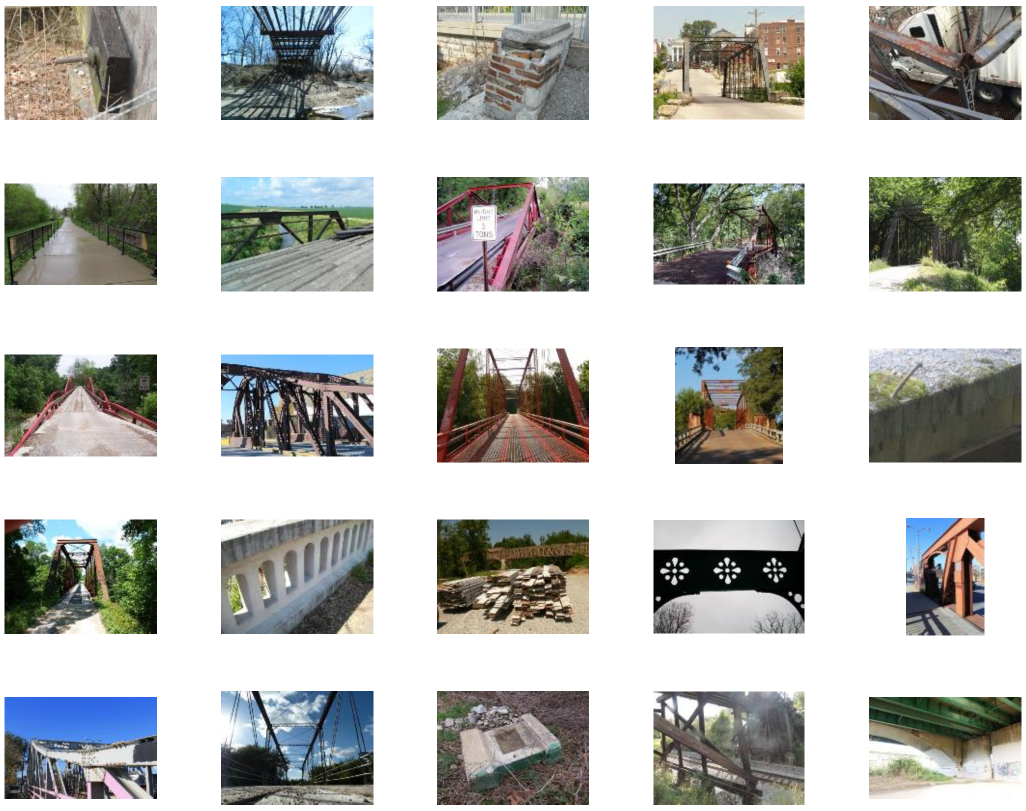


Figure 3.6 Images that only capture part of bridges

In Table 3.6, datasets A (DL10, DL11, and DL12) represent datasets which consist of both images showing complete and incomplete bridges, datasets B (DL13, DL14, and DL15) only include images showing complete bridges, and datasets C (DL16, DL17, and DL18) only consist of images with incomplete view of bridge. The number described in each dataset shows data distribution on the datasets. Dataset DL10, DL13, and DL16 were datasets created directly from image filtering process based on the perspective of the images. However, as it can be seen in Table 3.6, there is a variation in the number of images inside these datasets. To create a fair comparison, the number of images in dataset A, B, and C had to be equal therefore DL11, DL14, and DL17 were generated. As it can be seen in Table 3.6, these datasets have imbalanced sample distribution and due to the equal number of samples in DL13 and DL14, basically DL14 is similar to DL13. To prevent prediction bias in imbalanced datasets, dataset DL12, DL15, and DL18 were produced. As it is shown in Table 3.6, these datasets have almost equal number of samples in each class.

Table 3.6 Description of datasets for investigation of image completion to the performance of prediction model

Class	**Dataset consisting of all images (A)**			**Dataset consisting of only image with full view of bridges (B)**			**Dataset consisting only images of incomplete bridge (C)**		
	DL10	**DL11**	**DL12**	**DL13**	**DL14**	**DL15**	**DL16**	**DL17**	**DL18**
1	1456	496	300	496	496	300	960	496	300
2	6389	2496	300	2496	2496	300	3893	2496	300
3	2635	1201	300	1201	1201	300	1434	1201	300
4	706	287	287	287	287	287	419	287	287
5	5406	2460	300	2460	2460	300	2946	2460	300
6	641	312	300	312	312	300	329	312	300
7	962	399	300	399	399	300	563	399	300
8	379	186	186	186	186	186	193	186	186

3.2.4.4 Image Colour Variation

This section was conducted in order to investigate the effect of image colour to the performance of prediction model. The grayscale images were obtained by simply converting colourful images into their grayscale versions. Colourful images are formed by a number of pixels and each colour pixel has combination of RGB colour space. In this research, Luminance was implemented as the grayscale algorithm. Luminance is the standard grayscale algorithm, and it has been utilised frequently by image processing software for computer vision tasks [194]. In addition, it has been implemented in other studies [195]–[197]. In MATLAB, this algorithm is performed using "rgb2gray" function. Grayscale conversion was performed by calculating the luminance which is defined by [198]:

$$\mathrm{Y} = 0.299\mathrm{R} + 0.587\mathrm{G} + 0.114\mathrm{B} \tag{3-5}$$

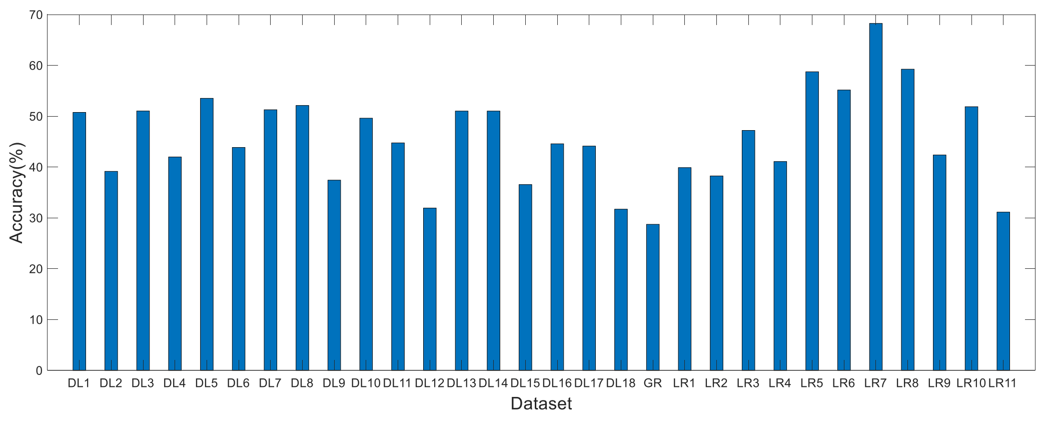
where Y is the grayscale value, R, G, and B are the red, green, and blue intensity respectively. From these grayscale images, a new dataset was created. This dataset was then employed to train a CNN-based prediction model. Finally, comparison between this prediction model and a prediction model created in the previous section was made to observe the effect of image colour to the prediction performance.

3.3 Results and Discussion

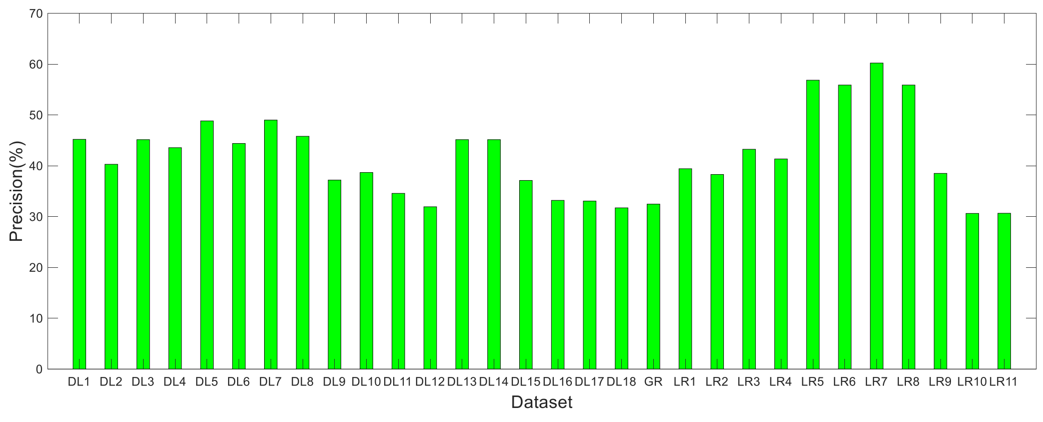
3.3.1 Variation in number of classes

3.3.1.1 Load Rating

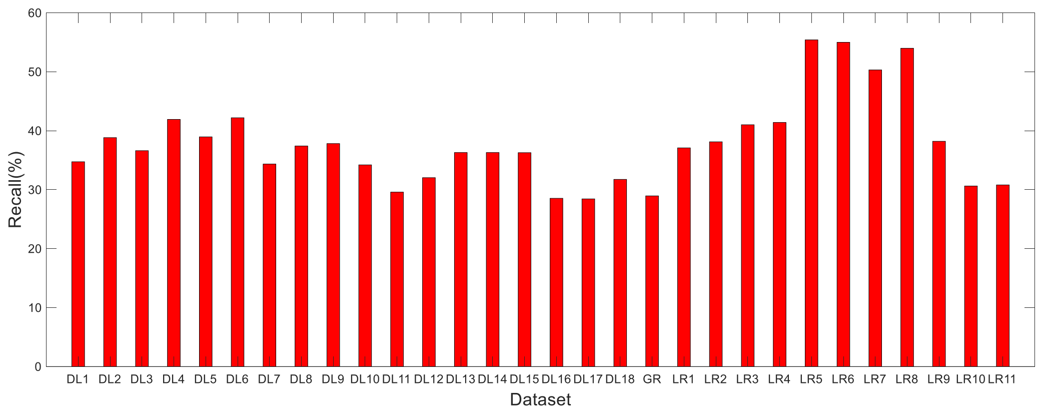
Figure 3.7 shows the performance of CNN models for load rating prediction trained using eight datasets with modification in class interval. From the figure, it can be seen that an increase in the class interval yields a better performance. Maximum accuracy of 68.26% and precision of 60% is achieved on prediction model trained using dataset LR7. However, the recall and F1 score produced by this network is slightly lower than those achieved by networks trained using dataset LR5, LR6, and LR8. In the figure, the minimum accuracy is obtained from prediction model trained using dataset LR2. In addition, balancing the data using down-sampling method yields to lower accuracy and precision. This decrease in accuracy from the effect of balanced dataset can be seen from Figure Figure 3.7. In the figure, it can be seen that for every imbalanced dataset (LR1, LR3, LR5 and LR7), the corresponding balanced dataset (LR2, LR4, LR6, and LR8) always produced lower accuracy. On the other hand, in term of recall and F1 score no similar trend occurs. However, the number of classes should be taken into account when evaluating the network performance. In the figure, accuracy higher than 60% are achieved on models trained using datasets that only have 3 classes. Therefore, accuracy higher than 60% is only achieved for three-classes prediction.



(a)

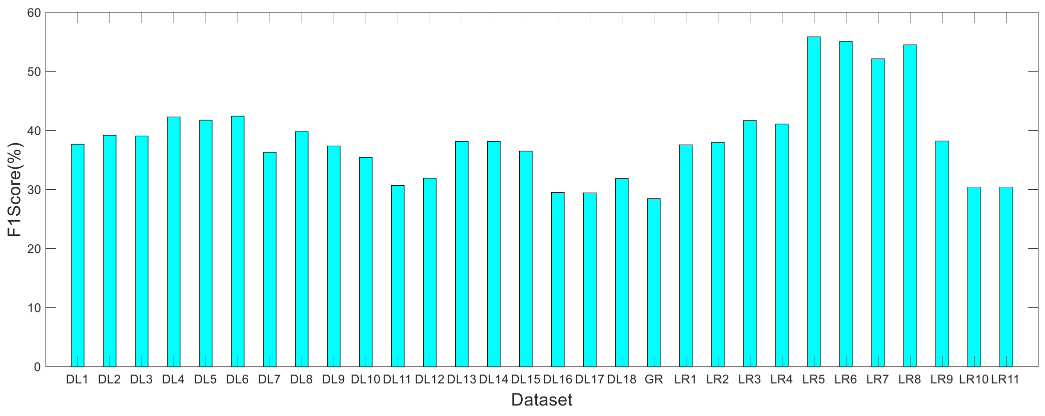


(b)



(c)

Figure 3.7 Performance metrics of models trained for load rating or design load prediction; (a) accuracy; (b) precision; (c) recall; (d) F1 score.



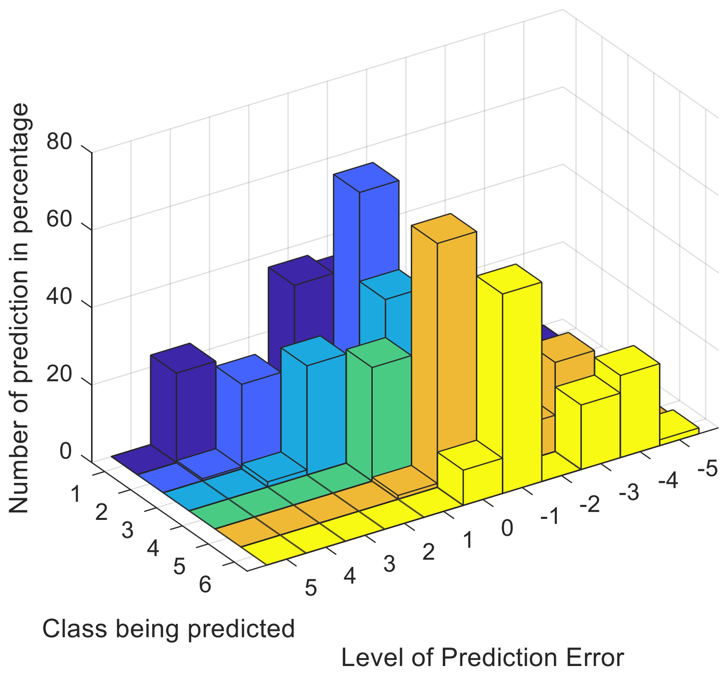
(d)

Figure 3.7 Continued.

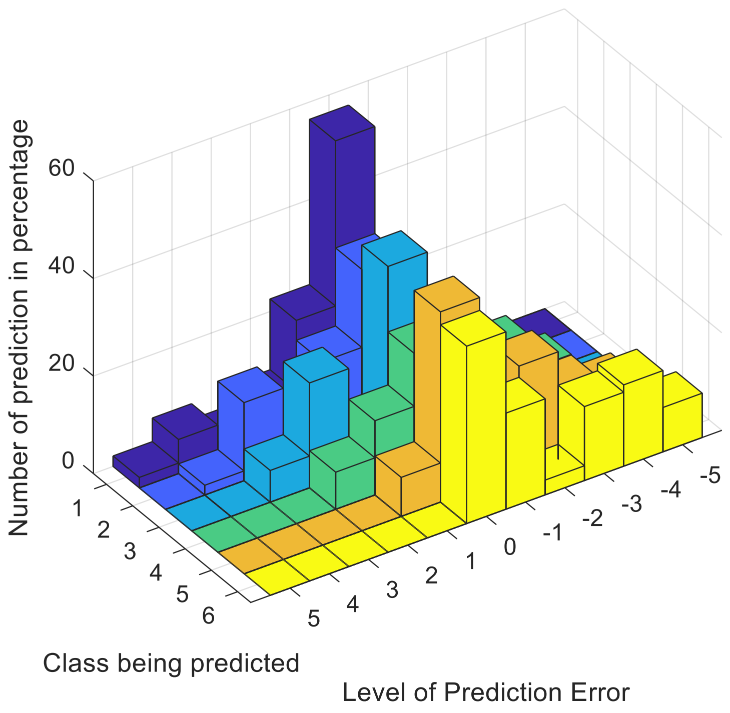
3.3.1.2 Design Load

Figure 3.7 shows the performance of models for design load prediction which are trained using various datasets. In the figure, it can be seen that for imbalanced dataset scenario, prediction model trained using DL5 produces the highest performance. Compared to networks trained using DL1 and DL3, this model achieves higher accuracy, precision, recall and F1 score. On the other hand, on balanced dataset scenario, model trained using DL6 performs best. It can be seen from the higher accuracy, precision, recall, and F1 score that are produced by this model compared to those produced by models trained using DL2 and DL4. In Figure 3.7, the effect of balanced dataset can also be observed. From the figure, it can be seen that balancing the dataset have negative relation with both accuracy and precision. This can be seen from every pair of balanced and imbalanced data (DL1-DL2, DL3-DL4, and DL5-DL6). However, it is also shown in Figure 3.7 that balancing data can improve both the recall and F1 score.

In order to observe the effect of balanced dataset to the model’s prediction, the prediction made by the networks was investigated. For this purpose, models trained using DL5 and DL6 were investigated. Figure 3.8 shows the prediction of samples in every category which is made by CNN models trained using these datasets. From the figure, it can be seen that although models trained using DL6 produces lower accuracy than those trained using DL5, the bias toward majority classes is minimised. In the figure, it can be seen that in imbalanced dataset, most prediction for samples from minority class is made toward majority class. On the other hand, for balanced dataset, almost in every class, maximum prediction on one class is made in the true class.



(a)



(b)

Figure 3.8 Error distribution generated by prediction models trained using (a) imbalanced dataset and (b) balanced dataset

The higher accuracy obtained from models trained using DL5 might be affected by the number of samples in the majority class. Although some misprediction produced in the minority classes, the number of samples in majority class is much larger than the number of samples in minority classes as shown in Table 3.4. Therefore, the true prediction made in the majority class contributes more than the misprediction hence higher accuracy can be achieved. In addition, unlike misprediction that is produced toward majority class in imbalanced dataset scenario, from Figure 3.8 it can be seen that for balanced dataset, misprediction mostly occurs to the adjacent category. In this case, the prediction does not deviate too much from the true class. There are exceptions such as for samples in 15, 20, and 45 tons classes. This might occur due to the data distribution. Although down-sampling has been performed to balance the dataset, the number of samples in the down-sampled classes is still twice as much as the number of samples in some categories such as the 27 tons, 36 tons with military inclusion, and 45 tons classes. Therefore, the bias prediction toward majority classes still occurs.

3.3.1.3 Comparison Between Models for Design Load Prediction and Models for Load Rating Prediction

The performance of prediction models for comparison between the performance of models for design load estimation and load rating estimation can be seen in Figure 3.7. In the figure, it can be seen that for equal number of samples in the dataset, the models created for design load prediction perform better compared to models trained for load rating prediction. Comparisons are made between DL7 and LR9 (case 1), between DL8 and LR10 (case 2), as well as between DL9 and LR11 (case 3).

In case 1, all images available for either load rating prediction or design load prediction are utilised. In Figure 3.7, it can be seen that DL7 produce higher accuracy and precision compared to LR9 and significant differences on these parameters are produced between these models. However, in case 1 the load rating prediction model produces slightly higher recall which leads to a slightly higher F1 score compared to the design load prediction model.

With equal number of images in each class (case 2), it can be seen that the design load prediction model produces higher performance than the load rating prediction model. This can be seen in Figure 3.7 from the higher accuracy, precision, recall, and F1 score obtained from the design load prediction model. Another interesting feature that can be observed in this figure is the higher accuracy achieved by the load rating prediction model compared to its performance on case 1. This event might occur due to the more imbalanced dataset used in case 2 compared to the dataset for case 1 which can be seen in Table 3.6. In imbalanced dataset scenario, more predictions tend to be made on a majority class which can lead to the increase in accuracy. However, the increase in accuracy doesn't represent a better performance since in case 2, the load rating prediction model produce lower precision, recall, and F1 score.

In balanced datasets scenario (case 3), it can be seen that the design load prediction model achieves higher accuracy, precision, recall, and F1 score compared to the load rating prediction model. In this case, with similar number of data and minimum bias from imbalanced dataset, the design load prediction model still manages to outperform the load rating prediction model. Note that lower performance is produced in this scenario due to the small number of samples utilised in the datasets for case 3. Therefore, in all scenarios the design load prediction models outperform the load rating prediction models.

3.3.2 The Impact of Image Completion to the Prediction Performance

The performance of the prediction models trained using datasets with various image quality on three different scenarios can be seen from Figure 3.7. For case 1 (DL10, DL13, and DL16), it can be seen that the models trained using dataset consisting of images which represent a bridge (DL13) perform the best among all. This can be seen from the highest accuracy, precision, recall, and f1 score produced by this prediction model compared to the values from the other models. This performance is achieved using fewer number of samples in the dataset compared to other datasets. The number of data samples used in the training process might explain the slight difference between the accuracy achieved by model trained using dataset DL10 and model trained using dataset DL13. In this case dataset DL10 has more images compared to dataset DL13 as shown in Table 3.6.

In case 2 (DL11, DL14, and DL17) where all datasets have equal number of data samples in every class, it can be seen that prediction models trained using good quality images (DL14) produce the highest performance among all. Furthermore, since all datasets have equal number of data samples, significant difference in the models' performance can be identified in this scenario. In addition, it is also shown that by using equal number of data samples, the model trained using images that do not completely show bridges achieves the lowest performance in term of accuracy, precision, recall, and F1 score.

In case 3 (DL12, DL15, and DL18) which is balanced datasets scenario, it can be seen that the model trained using images with full view of bridges (DL15) also achieves the highest accuracy, precision, recall, and F1 score among all prediction models. Similar with the previous section, decrease in performance is found in these prediction models. This might be due to the decrease of images number that are used to train these models because of down-sampling process used to create balanced datasets.

In this comparison, the result shows that in all 3 cases the models trained using good quality data (DL13, DL14, and DL15) produce the highest performance. From this comparison, it can be concluded that to obtain satisfactory performance, image quality plays an important role. This can be one factor that limits this research since it is challenging to obtain images with the right angle from web scraping.

3.3.3 The Impact of Image Colour to the Prediction Performance

Performances of prediction model trained using colourful images (DL6) and the performance of grayscale trained prediction model (GR) are provided in Figure 3.7. It can be seen that the implementation of grayscale images can worsen the performance of the load rating or design load estimation model. All parameters achieved by the model trained using grayscale image are lower than the parameters of prediction model trained using colourful images. This can be described by the effect of colour in detecting a material where colourful image can give more information about the material that forms an object. Hence, it is more suitable to use colourful images for this kind of application.

The impact of colour to the bridge classification has not been specifically reported from previous research. However, some research has reported a decrease in detection accuracy when implementing grayscale images for object classification. For example, in 2014, Chatifled et al. [199] reported a 3% decrease in the model accuracy after converting all input images into grayscale. In addition, another research on object detection utilising CIFAR dataset found that removing colour from images leads to 12% increase in the error rate [200]. Furthermore, for building classification application, McKee and Weber [201] showed that worse performance is produced by classifier trained using grayscale images as the input compared to the performance of models trained using colourful images. Therefore, despite reducing the data dimension that leads to cheaper computation as well as reducing variance in the data, eliminating colour might not always be beneficial in every application since it might discard useful information.

3.3.4 Proposed Optimisation Method for Performance Improvement

The result from prediction model shows unsatisfactory performance which can be inferred from the low accuracy. In order to analyse this condition, a prediction model for design load prediction trained using DL6 is taken as a case study. Figure 3.9 describes the training and validation process on a model trained using this dataset. From the figure, it can be seen that although the training accuracy reach 97% during the training process, overfitting occurs where no increase in the validation accuracy is produced after several iterations. Improvement of validation accuracy only occurs in the first 500 iteration before fluctuating validation accuracy is achieved. This leads to a significant difference between training and testing accuracy of the prediction model. This condition happens on all prediction models. Due to this situation, early stopping is implemented to stop the training process whenever no increase in validation accuracy is achieved. In iterative algorithm, Early stopping offers regularisation to combat overfitting by determining whether to stop ongoing iteration [202]. However, no significant improvement is obtained only by using early stopping method. As it can be seen in Figure 3.9, early stopping can only slightly improve the accuracy into 41%, which is only 3% improvement from 38% produced without early stopping.

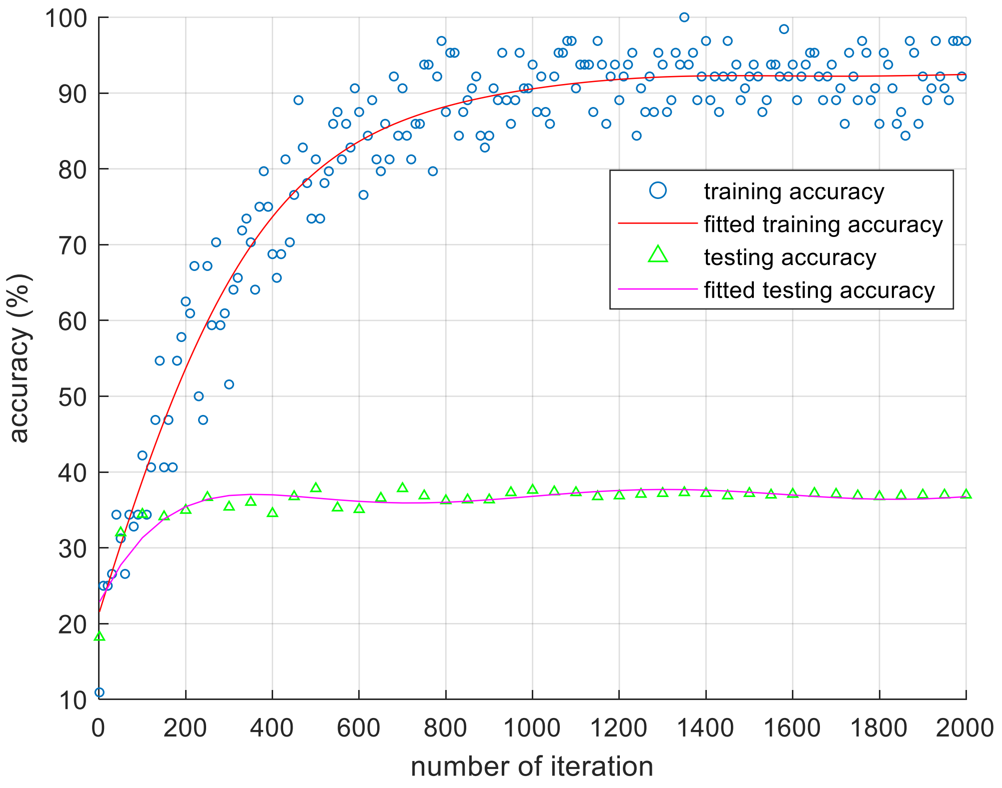
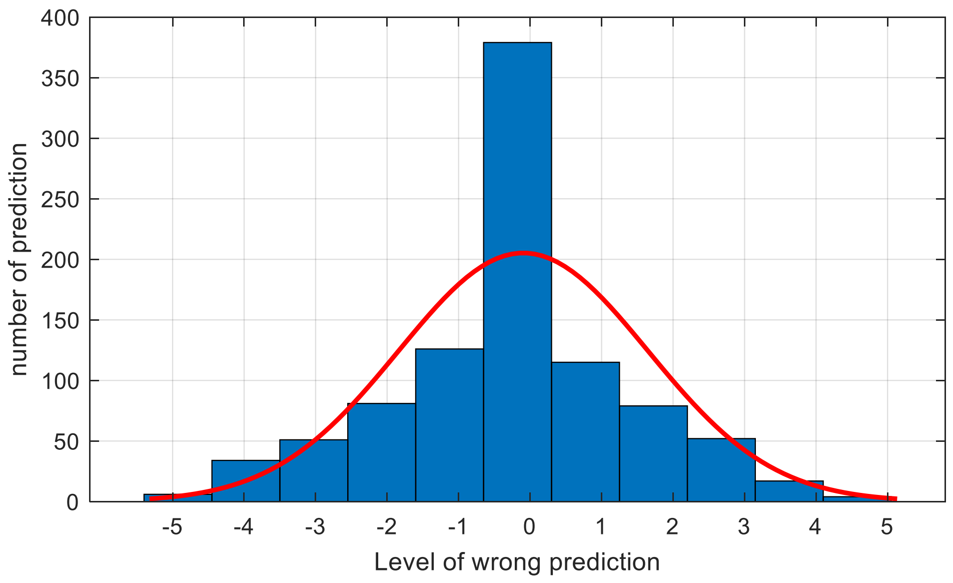
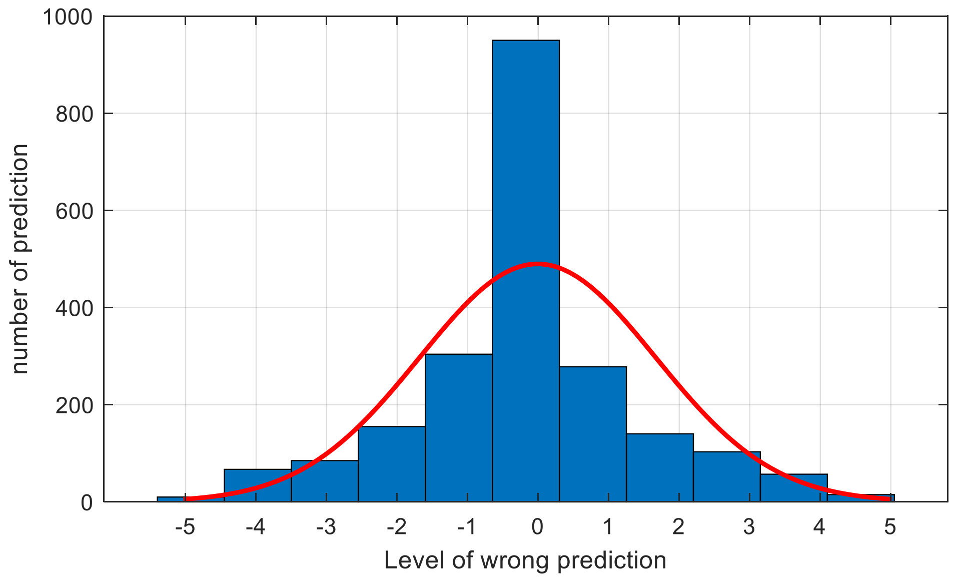


Figure 3.9 Training and validation accuracy of neural network for design load prediction

In order to observe the prediction error made by the model, an error probability distribution from the prediction is created. This error distribution is created by comparing the actual class and the prediction made using testing data. This error probability distribution for the design load prediction model is shown in Figure 3.10(a). From the figure, it can be seen that the error follows normal distribution. In this case, most predictions are made into the true class (40%) and most error occurs when predicting a bridge into a class adjacent to the true class (13.35% of the predictions are one class lower than the true classes). Both design load and load rating prediction models produce similar trend in the error distribution as it is shown in Figure 3.10.



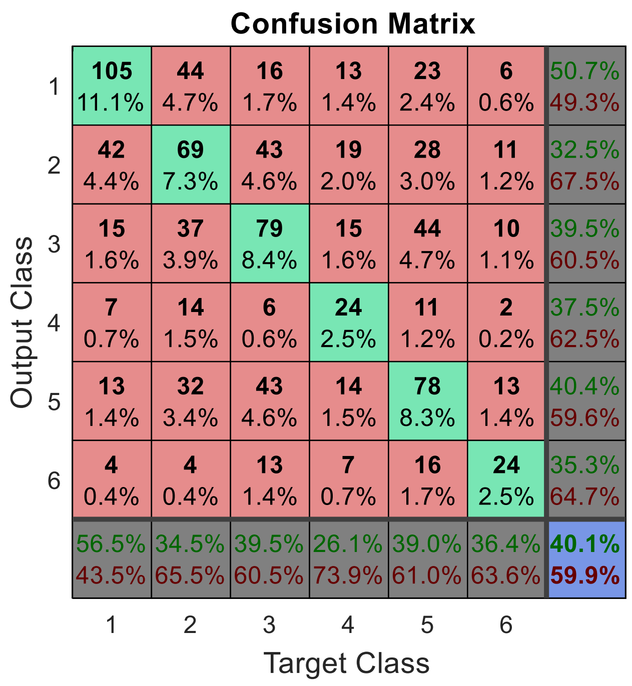
(a)



(b)

Figure 3.10 Error distribution from (a) design load prediction trained using DL6 and (b) load rating prediction using LR10

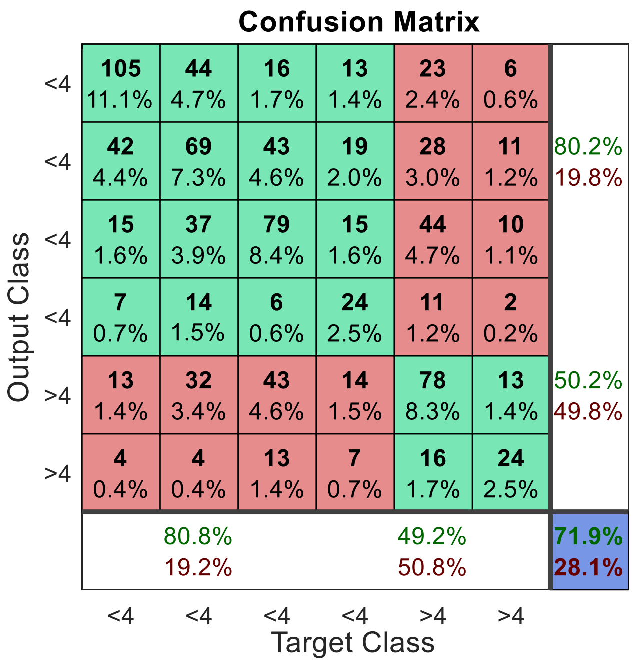
From this error distribution, it is possible to increase the performance by merging two classes adjacent to each other. The simplest way is by converting the multiclass classification into binary classification. In this case, the model is not predicting the exact value of either load rating or design of a bridge in the image. However, the model is used to predict whether a bridge load rating or design load is higher or lower than a certain value. This conversion for DL6 can be seen in Figure 3.11.



Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	6	
1	**105** 11.1%	**44** 4.7%	**16** 1.7%	**13** 1.4%	**23** 2.4%	**6** 0.6%	50.7% 49.3%
2	**42** 4.4%	**69** 7.3%	**43** 4.6%	**19** 2.0%	**28** 3.0%	**11** 1.2%	32.5% 67.5%
3	**15** 1.6%	**37** 3.9%	**79** 8.4%	**15** 1.6%	**44** 4.7%	**10** 1.1%	39.5% 60.5%
4	**7** 0.7%	**14** 1.5%	**6** 0.6%	**24** 2.5%	**11** 1.2%	**2** 0.2%	37.5% 62.5%
5	**13** 1.4%	**32** 3.4%	**43** 4.6%	**14** 1.5%	**78** 8.3%	**13** 1.4%	40.4% 59.6%
6	**4** 0.4%	**4** 0.4%	**13** 1.4%	**7** 0.7%	**16** 1.7%	**24** 2.5%	35.3% 64.7%
	56.5% 43.5%	34.5% 65.5%	39.5% 60.5%	26.1% 73.9%	39.0% 61.0%	36.4% 63.6%	**40.1%** **59.9%**

(a)



Confusion Matrix

Output Class \ Target Class	<4	<4	<4	<4	>4	>4	
<4	**105** 11.1%	**44** 4.7%	**16** 1.7%	**13** 1.4%	**23** 2.4%	**6** 0.6%	
<4	**42** 4.4%	**69** 7.3%	**43** 4.6%	**19** 2.0%	**28** 3.0%	**11** 1.2%	80.2% 19.8%
<4	**15** 1.6%	**37** 3.9%	**79** 8.4%	**15** 1.6%	**44** 4.7%	**10** 1.1%	
<4	**7** 0.7%	**14** 1.5%	**6** 0.6%	**24** 2.5%	**11** 1.2%	**2** 0.2%	
>4	**13** 1.4%	**32** 3.4%	**43** 4.6%	**14** 1.5%	**78** 8.3%	**13** 1.4%	50.2% 49.8%
>4	**4** 0.4%	**4** 0.4%	**13** 1.4%	**7** 0.7%	**16** 1.7%	**24** 2.5%	
		80.8% 19.2%			49.2% 50.8%		**71.9%** **28.1%**

(b)

Figure 3.11 Conversion from (a) multiclass classification into binary classification (b) to detect if a bridge load rating is lower than 27 tons.

Using binary classification conversion, the network is now used to predict:

- If a bridge has design load lower than 10 ton (LV1)
- If a bridge has design load lower than 15 ton (LV2)

- If a bridge has design load lower than 20 ton (LV3)
- If a bridge has design load lower than 27 ton (LV4)
- If a bridge has design load lower than 36 ton (LV5)

Notice from Figure 3.11 (a), before conversion, true prediction only occurs on the diagonal part of the confusion matrix that is identified in the green region. After the conversion, there are some regions that become either true positive or true negative that are identified with green regions in Figure 3.11 (b). By performing this conversion, improvement on prediction model can be obtained. The accuracy, precision, recall, and F1 score produced by the system on each low level after conversion can be seen in Figure 3.12. Following the conversion, it can be seen in Figure 3.12 that an increase in the performance is produced. In addition, it is shown that the maximum accuracy, precision, recall, and F1 score are produced when predicting in level 5. In this level the accuracy, precision, recall, and F1 score are 90.89, 95.21, 94.99, and 95.10 respectively.

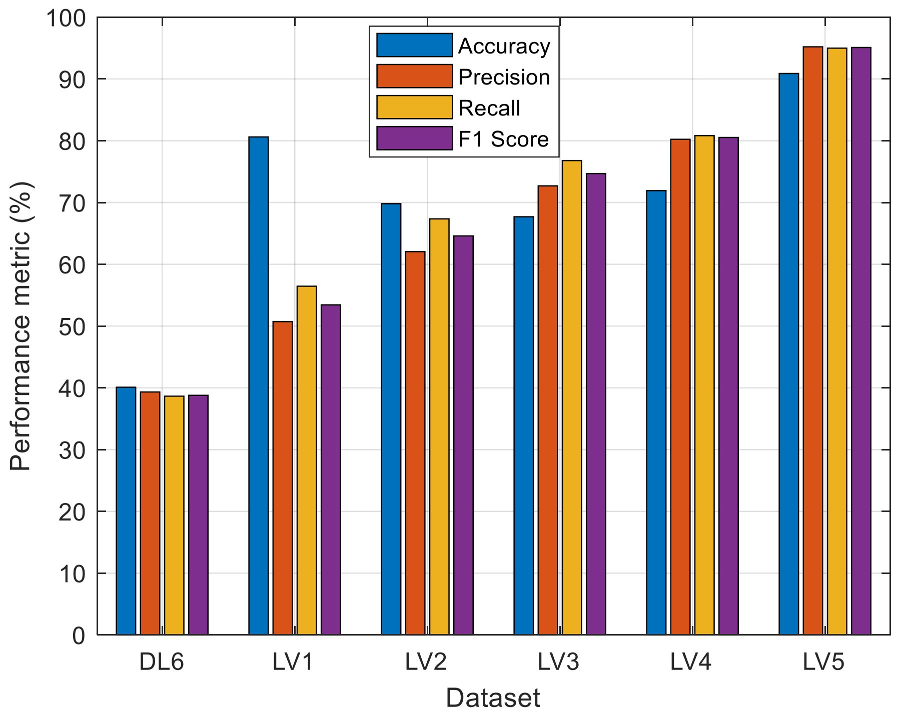


Figure 3.12 Performance of prediction models measured in accuracy, precision, recall, and F1 Score after conversion to binary classification

As shown in Figure 3.12, after conversion, accuracy has different trend compared to other metrics. From the figure, all metrics experience positive trend from LV1 to LV5.

On the other hand, the accuracy falls between LV1 and LV3 and it then rises until LV5. As explained in equation (3-1), accuracy is affected by both the true positive and true negative values. On the other hand, both precision and recall are only influenced by the true positive as given in (3-2) and (3-3) respectively. Therefore, high accuracy does not indicate high recall and precision as having large number of true negatives might increase the accuracy yet both recall and precision are unaffected.

As previously mentioned, after the conversion, the model is used to predict whether a bridge's design load is lower than a value. In this case, 'positive' class represents the bridge whose design load is lower than the value while 'negative' class are bridges with design load higher than the value. In LV1, the model produces prediction accuracy of 80.61% from 105 true positives and 656 true negatives out of 944 testing data. When LV2 conversion is performed, the number of true positive becomes higher and the increase in the true positive is higher than the increase in the false prediction. As a result, both the precision and recall of the model rise as illustrated in Figure 3.12. On the other hand, the number of true negative decreases thus the accuracy of the model drops between LV1 and LV2. Similar result also occurs between LV2 and LV3 where the accuracy of the model decreases while other metrics such as precision, recall, and F1score increases. However, between LV3 and LV4 as well as between LV4 and LV5, the number of false predictions decreases while the true positive rises, raising all prediction metrics including accuracy, precision, recall, and F1score as depicted in Figure 3.12.

3.3.5 Analysis on the Model's Uncertainties

Certain factors might limit the applicability of our proposed method. Although this method might predict the bridge load rating using the features that are extracted from images, it might be challenging to extract some important features from images. First of all, according to [203], one of the factors that influences load rating is the superstructure type. Therefore, successful identification of the superstructure might improve the prediction accuracy. One of the key points of CNNs is the capability in identifying various shapes from images. Hence, given good quality dataset for training and testing, CNNs will be able to obtain the superstructure information from the images.

The next factor that affects the load rating is the number of lanes. This information can be obtained visually from images depending on the angle where the images are taken. Therefore, it is possible that a CNN model fails to gather this information from bridge images that are not sufficiently taken, leading to uncertainty to the prediction. As it has been discussed in section 3.3.2, the perspective of a bridge in the image might influence the prediction result. This view might help in the extraction of features such as the bridge type, length, and span. Hence, the view of the image should be taken into consideration when implementing this method.

In addition, there are other features which can be difficult to extract using images. This can be seen from the result in section 3.3.1.3. In this section, it has been shown that predicting design load from bridge images is more achievable compared to predicting load rating from bridge images. Load rating of a bridge is affected by its condition thus the bridge's condition is one important feature for load rating estimation. However, it is a challenging task to quantify a bridge's condition from its image, especially from an image that is taken remotely from the bridge. This situation might introduce error since the prediction models are unable to obtain this information from images. On the other hand, bridge's condition provides no effect on the capacity which the bridge is designed for. This might lead to a better performance in design load prediction model.

The bridge material also influences the load rating. In section 3.3.3, it has been shown that eliminating colour results to decrease in prediction accuracy. This might be caused by the loss of information in obtaining material information of the bridge when implementing grayscale images. On the other hand, CNN model has limitation in gathering the material information especially the material inside the bridge. For example, it is possible for the CNN model to differentiate between steel and concrete bridges by using image processing. On the other hand, determining whether a structure is a reinforce concrete structure or a pre-stressed concrete structure is a challenging task. The limitation of image processing in extracting this feature might introduce error for the proposed method. Therefore, to tackle this limitation, in our future work we are planning to incorporate the features that are difficult to extract using image processing as additional inputs to the prediction models. Hence, along with the bridges' images, these features will be utilised as input data.

3.4 Conclusions and summary

In this research, novel CNN-based prediction models for estimating the load rating and design load of bridges have been trained based on crowdsourced image data. The conclusions are as follows:

- By using equal number of samples, the models trained for design load prediction produce higher performance than the models for load rating prediction. This can be seen from the higher accuracy, recall, precision and F1 score of the models trained for design load prediction compared to those of the load rating prediction models.
- The image quality affects the performance of prediction models. Therefore, in order to improve the performance, the quality of images used to train models should be taken into account.
- It was found that the implementation of colourful images for this application is more suitable than using grayscale images.
- Converting multiclass classification into binary classification can improve the prediction performance.

Chapter 4 Estimation of Structural Response using Convolutional Neural Network: Application to Suramadu Bridge

Summary

This chapter presents a deep learning-based data interpretation method that employs correlation between sensor measurements for estimating structural responses. The method employs CNN architecture to process raw measurement data. The method can be implemented for handling missing data problem or calibration purposes. Section 4.2 explains the methodology performed in this work including the data collection, data processing, model training and testing, and evaluation. Section 4.3 presents the case study employed to validate the proposed approach. In this study, the method is validated by using real monitoring data. Section 4.4 presents the result and discussion from the validation of the proposed method. Finally, section 4.5 provides the summary and concluding remarks of the chapter.

4.1 Introduction

In Chapter 3, a deep learning-based method which employs image data for SHM application has been presented. In addition to image data, SHMs also utilise time series data collected from sensor measurements in monitoring structures condition. In addition to measurement of structural responses, the time histories often consist of measurements of surrounding environment such as temperature or wind speed. The next crucial step in SHM is how to interpret the data to better understand the structure's condition.

The data interpretation methods for SHM have been presented in section 2.2. The data-based interpretation method offers promising potential for SHM application. This method requires no geometrical or material information of the monitored structure and data interpretation is performed based on the statistical trend extracted from the data. In general, this method employs machine learning techniques which might require high level of domain expertise especially in feature extraction step as it is mentioned in section 2.2.2. Deep learning techniques that have capability in learning feature automatically can be seen as a potential solution for data-based interpretation.

Maintaining continuous data collection is one of the main challenges in implementing SHM. Many factors might impact the monitoring activity such as sensor malfunction, the need of sensor calibration, hardware problems, network faults, and so on. These problems might lead to the loss of important monitoring data and while some problems might be solved directly, others can be time consuming. In the emergency scenario where continuous monitoring is necessary, this condition might potentially harm the structure. Therefore, implementing a method that can recover the missing data might help improving the monitoring system.

In this research, we propose a framework based on 1-D CNN to predict structural response of bridges. The proposed approach is validated through a full-scale case study of Suramadu Bridge in Indonesia. The CNN model estimates the cable force measured by a sensor using other measurement time series from other sensors. This approach might be beneficial for the monitoring activity by providing solution for missing data (e.g., due to individual sensor fault). In addition, the estimation can be further employed for determining threshold for anomaly detection.

4.1.1 Summary of Novelty and Contribution

The work in this chapter is based on a published research [204]. This work investigates, for the first time, how cable stress can be estimated using temperature variations. The study presents the first application of 1-D CNN regressor on data collected from a full-scale bridge. This work also evaluates the comparison between CNN regressor and other techniques such as ANN and linear regression in estimating bridge cable stress which has not been performed previously.

4.2 Methodology

Figure 4.1 shows the activities conducted in this project. There are four main activities performed in this research such as:

1. Data collection
2. Data processing
3. Prediction models training and testing
4. Evaluation

Data collection is performed by using a sensor system that is deployed on a bridge. The sensor system gathers both response and environmental data from the monitored bridge. Then, data pre-processing is carried out to generate dataset that can be used by the prediction model. There are two steps performed on the data processing: normalisation and data conditioning.

Normalisation is performed on all measurement signals. For this purpose, statistical normalisation that yields zero mean and unit variance is conducted as follow [205]:

$$x_t = \frac{x - \mu}{\sigma} \tag{4-1}$$

where x, μ, and σ are the original data, mean, and the standard deviation of the data respectively. Normalisation is important especially if the method is applied on dataset consisting of measurements from different types of sensors. In this case, each type of sensor has its own unit of measurement thus normalisation might avoid the dominance of extreme value.

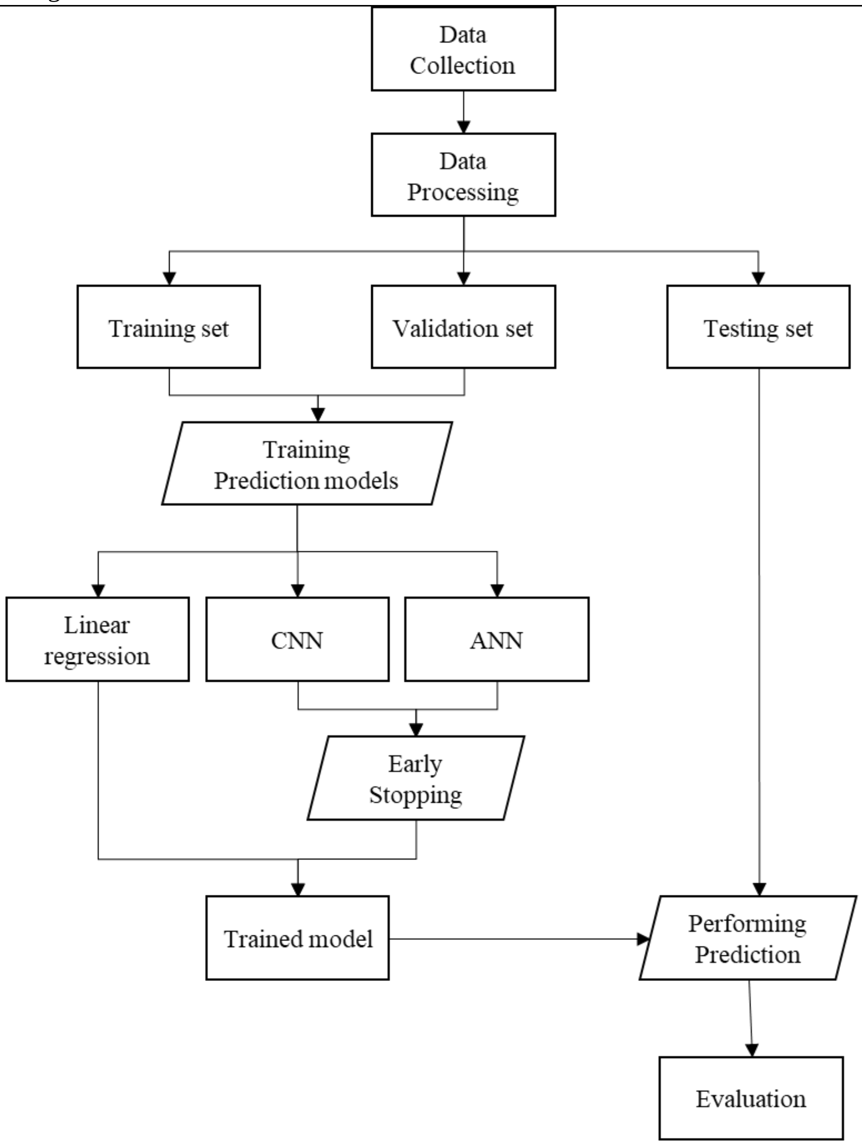


Figure 4.1 Schematic of the methodology performed in this research

The next step is to generate data frames from the dataset. This step is necessary when using CNN architectures since the models require data in the form of sequences. Consider a prediction model trained to perform a regression using the measurements from sensor S to estimate the measurements of sensor O. If the dataset consists of nd measurement points, the sensor data used for the input and the output can be represented as:

$$S_i = [S_1, S_2, S_3, \ldots., S_{nd}] \tag{4-2}$$

$$O = [O_1, O_2, O_3, \ldots., O_{nd}] \tag{4-3}$$

where S and O represent the sensor data utilised as the input and output in the prediction model respectively. In this step, these sensor signals are divided into a number of data frames each having a fixed length ns as it is described in Equation (4-4) and (4-5).

$$S_n = [S_n, S_{n+1}, S_{n+2}, \dots., S_{n-1+ns}] \tag{4-4}$$

$$O_n = [O_n, O_{n+1}, O_{n+2}, \dots., O_{n-1+ns}] \tag{4-5}$$

After the processing step, the dataset is divided into three sets: training, validation, and testing. The prediction models are trained by employing the training and validation sets. In the training process, early stopping is performed in order to avoid overfitting. In iterative algorithm, Early stopping offers regularisation to combat overfitting by determining whether to stop ongoing iteration. It is performed by monitoring training parameter, usually validation loss, that is generated when feeding validation set into the model on each epoch of the training process. When no improvement is produced by the model after a predefined number of epochs, known as early stopping patience, then the training is finished.

Finally, the testing set is utilised to evaluate the trained model. This set represents new data that has not been seen by the model. In this research, the models are evaluated by measuring the prediction error from applying the testing set on the trained models. The prediction is then compared with the actual value and the regression error metrics such as mean absolute error (MAE), mean absolute percentage error (MAPE), and RMSE are collected. The MAE, MAPE, and RMSE are calculated as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{ns} |\hat{y}_i - y_i| \tag{4-6}$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{ns} \left|\frac{\hat{y}_i - y_i}{y_i}\right| \tag{4-7}$$

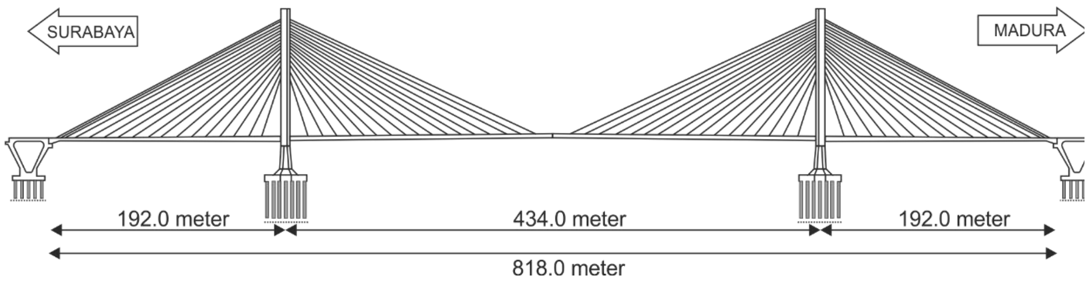
$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{ns}(\hat{y}_i - y_i)^2} \tag{4-8}$$

where $\hat{y}_i$ $(i = 1, \dots, ns)$ is the prediction at i-th position in the sequence, y_i is the actual value at the i-th position in the sequence, and ns is the total data points in a sequence.

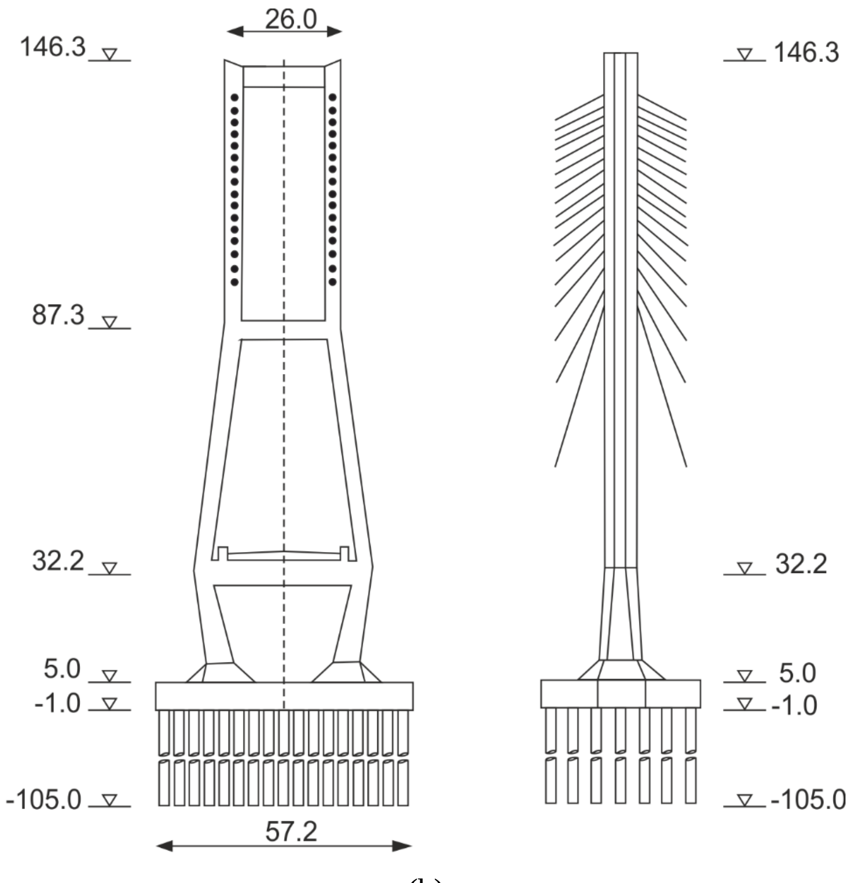
For comparison purpose, other data-based interpretation methods such as linear regression and ANN are employed in this research. As it is shown in Figure 4.1, these models are also trained using the training and validation sets and evaluated using the testing set. However, while early stopping is conducted in the ANN model, this method is not applied on the linear regression model. The trained models are then used on the testing set to obtain the regression error metrics such as MAE, MAPE, and RMSE from the discrepancies between the predictions and the actual values. Finally, comparison of these error metrics from the prediction models adopting CNN, ANN, and linear regression, are made in order to evaluate the effectiveness of the CNN framework.

4.3 Case Study: Suramadu Bridge SHM

The Suramadu Bridge is a cable-stayed bridge which is part of a 5.4 km long connection between Madura Island and Surabaya on the island of Java in Indonesia. The bridge consists of 3 parts. The first one is the fly-over on both sides, covering 1458 m from Surabaya and 1818 m from Madura. The second part is a pair of connectors (i.e. approach bridge) of 627 m each. The last part is a central bridge which consists of two 192 m side-extensions and a 434 m main span. Figure 4.2 shows the schematic representation of the central Suramadu Bridge alongside with the pylons. The bridge consists of two symmetrical cable-stayed cantilever sections. The deck is 30 m wide and comprises two longitudinal steel girders, approximately 2.8 m deep, stringers and floor beams and a reinforced concrete slab with 250 mm in depth. The cable plane is aligned with the central line of the main girders. There are 70 stay cables in each cable plane and 140 cables in total. The longitudinal spacing of cables is set to be 12 m for standard girders segments and for two segments under both towers is set to be 18.5 m. In the pylon tower, the distance between the intersecting points of the central lines of stay cables and the central line of the pylons tower is all 2.2 m.



(a)



(b)

Figure 4.2 Schematic representation of the central bridge (a) side view and (b) pylons (all dimensions are in meter)

To improve the earthquake-resistance performance of the main bridge, 4 sets of longitudinal viscous dampers were installed on lower pylon cross beams under the main deck. In the lateral direction (across the bridge deck), concrete stoppers connected to lower cross beam suppress the bridge deck movement in this direction.

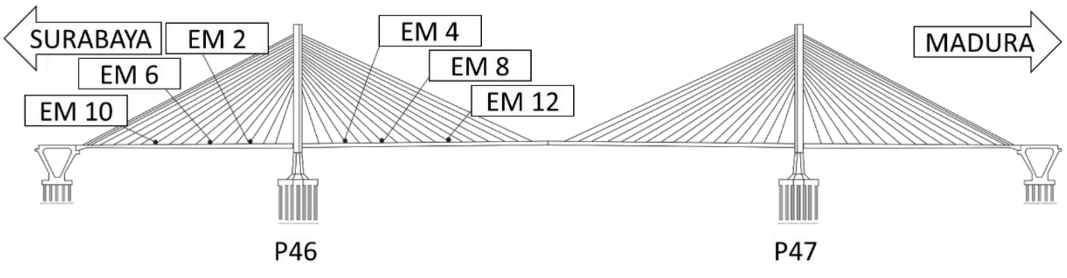
The longitudinal slope from one end to the mid-span is 1%, and the vertical circular curve with a radius of R=18000 m is set at the mid-span, whose tangent length is 180 m and vector height is 0.9 m. This vertical curve can enhance the comfort feeling when driving and also brings aesthetic appearance to the bridge. Expansions gaps and vertical bearings are located at both ends to allow the longitudinal elongation of the bridge due to change in temperature. The bridge foundation is made of a cast-in-situ group of piles with a diameter of 2.4 m and a depth of 100 m below the pile cap. The pile's foundation is set to be quincunx in plan, and the minimum distance between piles from centre to centre is 6 m. The external diameter of steel pile casing is 2.7 m and wall thickness is 20 mm. After finishing the construction, the steel pile casing becomes a part of the permanent structure, however, the contribution of the pile casings to the bearing capacity of piles was not taken into account in the design. The pile cap is an octagonal cube with a thickness of 6 m.

4.3.1 Suramadu Bridge Cable Force Monitoring

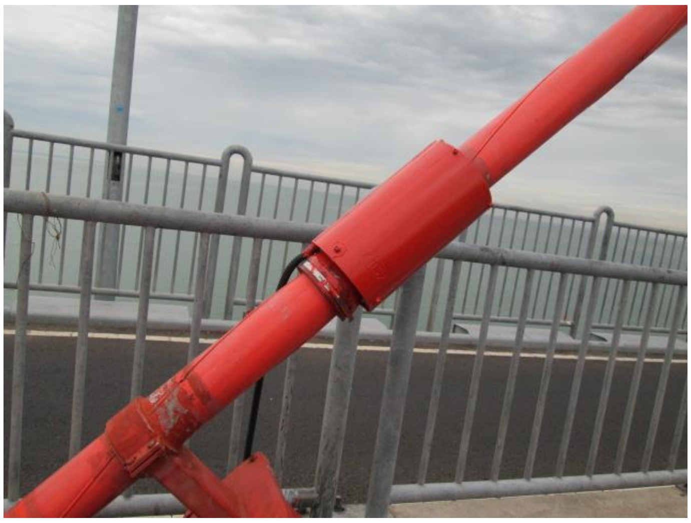
After construction of the Suramadu Bridge, the Indonesian Ministry of Public Works and Housing (IMPWH) invested in monitoring and maintenance of the bridge due to the importance of the bridge for the economy of the region. The bridge is the longest cable-stayed bridge in Indonesia and an essential part of transportation system for eastern Java. In order to continuously monitor the bridge, an extensive set of sensors were installed on some parts of the bridge. Among these sensors, elasto-magnetic (EM) sensor is employed to monitor the cable force. In particular, for cable-stayed bridges, monitoring the cable stress is important in assessing the health of the structures [21], [206], [207]. For example, in the collapse of Genoa bridge, the combination of fatigue and corrosion of the cable stays might be one of the possible reasons leading to the collapse of the bridge in 2018 [208].

In cable force measurement, EM sensors have gained remarkable attention since offer noncontact measurement, corrosion resistance, actual-stress measurement, and long service life [140] compared to conventional cable force measurement method. EM sensors exploit the change in magnetic permeability of a ferromagnetic material due to applied stress. By utilising this relation, it is possible to obtain cable stress applied to the material by measuring the change in the magnetic field [21]. EM sensors work by measuring the change in the magnetic field and convert it into voltage unit which later is further translated into cable stress. EM sensors have been widely employed in some bridges all over the world [209]–[212].

In Suramadu Bridge monitoring system, in total 32 cables have been equipped with elasto-magnetic (EM) sensors. The sensors measure force cable and temperature of the cable simultaneously. Figure 4.3 shows the EM sensor and location of EM sensors deployed in Suramadu Bridge. From these sensors, some cable force measurement data have been generated for Suramadu Bridge monitoring system.



(a)



(b)

Figure 4.3 Details of EM sensors in Suramadu Bridge monitoring system; (a) EM sensors location; (b) Photograph of EM sensor.

4.3.2 Data Processing and conditioning

To validate the CNN framework, this research employs six months monitoring data that have been collected from 24th April 2014 to 4th October 2014. In total, measurement data from six EM sensors (EM2, EM4, EM6, EM8, EM10, and EM12) from Suramadu Bridge monitoring system are utilised. As it has been mentioned in the section 4.3.1, Each EM sensor is equipped with its own temperature sensor and the sensor is sampled once in an hour. Figure 4.4 shows the temperature versus cable force graph from six EM sensors.

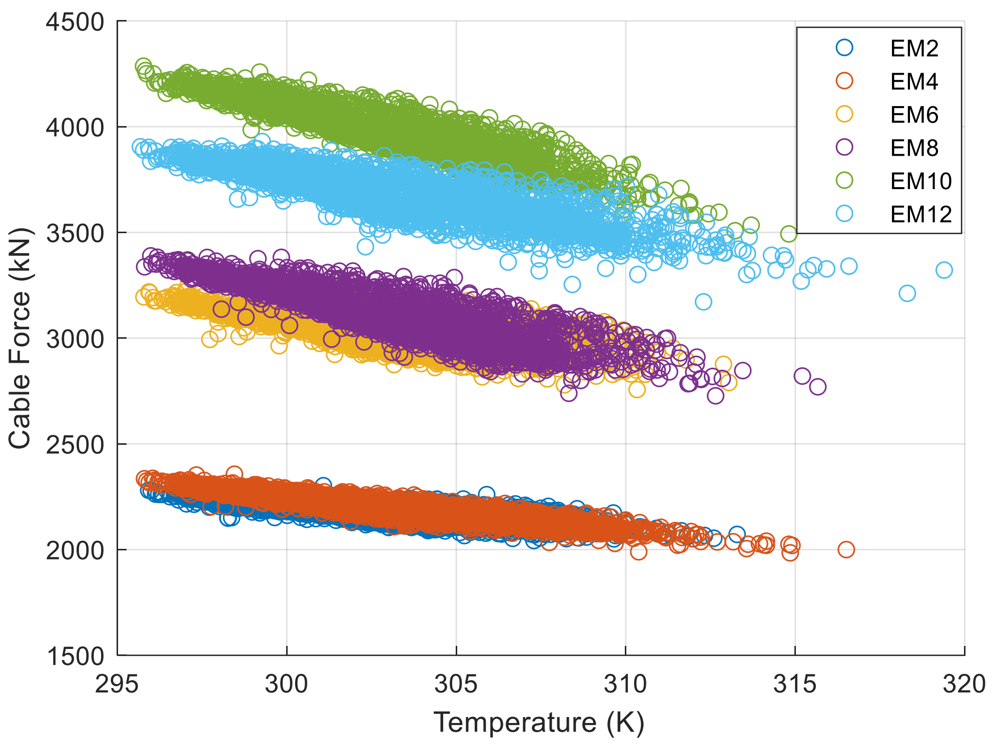


Figure 4.4 Temperature vs Cable Force from six EM sensors

In this work, temperature data are selected to learn cable force due to the bridge structural properties. Temperature variation causes bridge deformation, potentially affecting the cable stress measurement. In addition, previous research has shown that for cable-stayed bridges, it has been reported that temperature influences bridge response significantly [27]. In this research, estimation of cable force is performed by using the temperature data as it is illustrated in Figure 4.5.

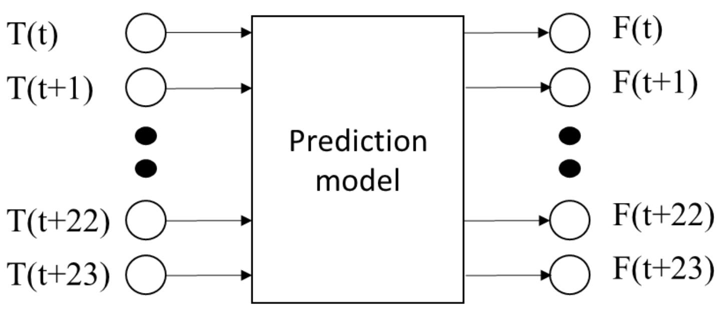


Figure 4.5 Schematic of the cable force estimation framework using temperature variance as the input

As illustrated in Figure 4.5, the prediction models utilise a data frame with a window size of 24 temperature data points T(t) that represent one daily measurement, to estimate a sequence of cable force measurements F(t). This is performed by splitting the measurement signals into a number of data frames each having a window size of 24 measurement points. For data augmentation purpose, data frames are generated by sliding the fixed window with overlap of 23 points between one frame to the next frame. The data augmentation step is illustrated in Figure 4.6. From this step, in total six sensor datasets have been generated.

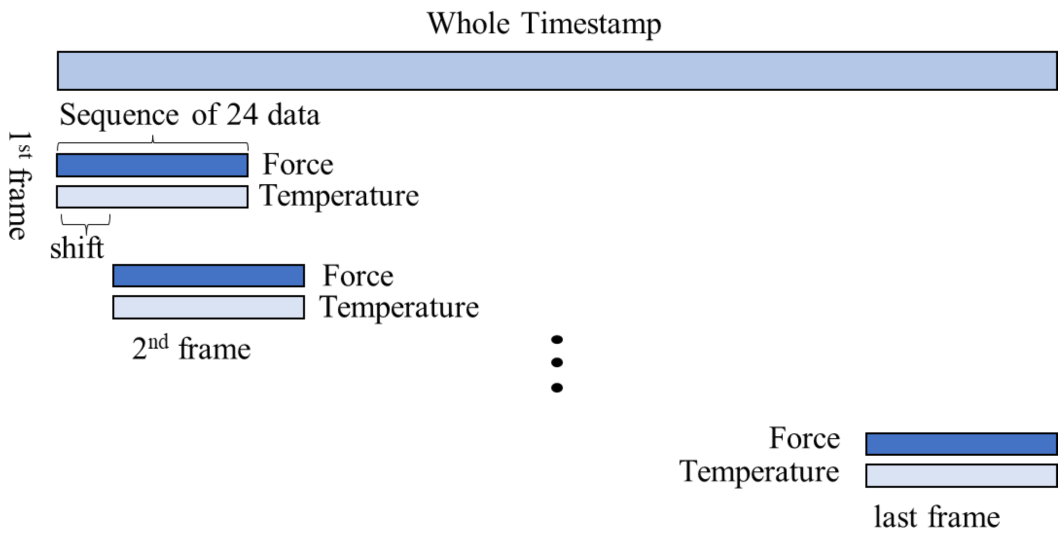


Figure 4.6 Production of data frames from Suramadu Bridge dataset

The next step is performed by splitting each dataset into training, validation, and testing sets. Both the training and validation datasets are employed in the training step. The former is used in the supervised training while the latter is employed for early stopping method. On the other hand, the testing set serves as new unseen data for the model. This way, the models are evaluated based on the accuracy of their prediction on completely new data. Therefore, it is important to avoid overlap between the training and testing set. In this research, the dataset is divided as follows: the first 60% of the time series for training set, the next 20% for validation set, and the last 20% of the time series for testing set. Table 4.1 shows the number of data frames for each set.

As it can be seen in Table 4.1, the data training, validation, and testing sets contain data collected from different months. Therefore, the operational and environmental conditions between these sets might be varying. However, the temperature in

Indonesia is stable around the year hence the variation between the sets will not affect the CNN model significantly.

Table 4.1 Detail of training, validation, and testing sets for the implementation of CNN framework on Suramadu Bridge monitoring data

EM	Sampling	Training		Validation		Testing	
number	rate	Number	Period	Number	Period	Number	Period
2	Hourly	2052	April-July	684	July-August	684	August-October
4	Hourly	2052	April-July	684	July-August	684	August-October
6	Hourly	2052	April-July	684	July-August	684	August-October
8	Hourly	2052	April-July	684	July-August	684	August-October
10	Hourly	2052	April-July	684	July-August	684	August-October
12	Hourly	2052	April-July	684	July-August	684	August-October

4.3.3 CNN Model Training and Testing

The CNN architecture adopted in this research is presented in Table 4.2. The architecture refers to CNN architecture proposed by Zhang et al. [185] where only a single convolution layer is adopted. However, some hyperparameters are configured to suit the investigated problem.

Table 4.2 CNN architecture for cable force estimation

Layer	**Input**	**Output**	**Parameter**	**Activation**
Convolutional	24×1	24×32	Filter number: 32, size: 16; stride: 1; padding: same	LReLU
Batch Normalisation	24×32	24×32	None	None
Max Pooling	24×32	12×32	Pooling size: 2; stride: 1	None
Dropout	384	384	Dropout rate: 0.4	None
Fully Connected	384	50	None	LReLU
Fully Connected	50	24	None	None

As described in Table 4.2, the architecture employs a single CNN layer that utilises 32 filters each having a size of 16. The activation function implemented in this research for the CNN layer is LReLu with α equals to 0.05. The convolution layer is followed by a batch normalisation layer. Generally, in the training process of deep learning, mini batches containing some training data are utilised. Batch normalization offers training acceleration by minimising internal covariate shift that occurs due to the change in the batch distribution [213]. It is achieved by calculating the mean and variance of the batch to normalise the batch distribution before feeding the batch to the next layer.

After the batch normalisation layer, a pooling layer is employed in the architecture. This layer halves the dimension of the data by utilising pooling size of 2 and stride of 1. Max pooling layer is utilised for the architecture. The output of the pooling layer is flattened to form an array containing 384 elements that are connected to a dropout layer. Deep learning models might suffer from overfitting where the models learn the training data too well yet providing poor generalisation when given completely new data. In this research, dropout layer is employed to combat overfitting. Dropout is one of regularisation techniques for deep neural networks that works by removing network nodes randomly during training process, thus combining some deep architectures [214]. No rule is specified on where to apply the dropout layer due to its stochastic nature. In this research, dropout layer with 0.4 dropout rate is adopted between the feature extraction and regressor.

The dropout layer is connected to a fully connected layer with 50 nodes. Similar to the convolution layer, this layer utilises LReLU as the activation function with α equals to 0.05. This layer is connected to another fully connected layer that serves as the output layer of the architecture. The last layer is a fully connected layer with 24 nodes representing the 24 measurement points of cable force. In this research, the early stopping patience is set to be 200.

4.3.4 Comparison with other Machine Learning Models

In this research, ANN models that utilise up to four hidden layers are trained to map the relation between the temperature and the cable force. The ANN models adopt 24 nodes at the input layer, up to four hidden layers, and 24 nodes at the output layer. Hyperparameter tuning is carried out by varying the number of nodes in the hidden layers. The hidden layer configurations can be seen in Table 4.3. By utilising these various hyperparameters, in total 96 models are generated and trained using training data. Similar to the training step that is conducted for CNN architecture, early stopping is employed by monitoring the validation loss using the validation dataset. All the trained models are tested using testing data and the model that produced the lowest testing error is selected as the optimised model.

Table 4.3 Details of hyperparameters for optimisation of ANN models

Hyperparameter	**Possible value**
Number of nodes in hidden layer 1	5, 10, 20
Number of nodes in hidden layer 2	0, 5, 10, 20
Number of nodes in hidden layer 3	0, 5, 10, 20
Number of nodes in hidden layer 4	5, 10

On the other hand, the training step performed for the linear regression model is straightforward. All testing data are fed to the model and evaluation is performed by employing the testing dataset to the model.

4.4 Results and Discussion

After the training process, in total 18 prediction models based on CNN, ANN, and linear regression for six EM sensors have been picked for evaluation. As mentioned in the previous section, these models are tested using the testing set to assess their performances on unseen data. Figure 4.7 illustrates the comparison between the actual measurements of EM12 from the testing set and the prediction from three models adopting various data-based interpretation methods.

In Figure 4.7, it is shown that all of the prediction models manage to capture the trend of the actual measurement In order to evaluate these prediction models, for every data point in each sequence, the residual from the discrepancy between the actual value and the prediction is calculated. For this purpose, the metrics we use are the MAE, MAPE, and RMSE. In each data sequence, the error metrics on all data points are calculated. From this process, the error metrics representing each data sequence are obtained. In order to further investigate the effectiveness of the CNN-based prediction model, the average values of the MAE, MAPE, and RMSE of six cable force measurements are also calculated. Table 4.4 shows the Average of MAE, MAPE, and RMSE generated from prediction models utilising CNN, linear regression, and ANN models.

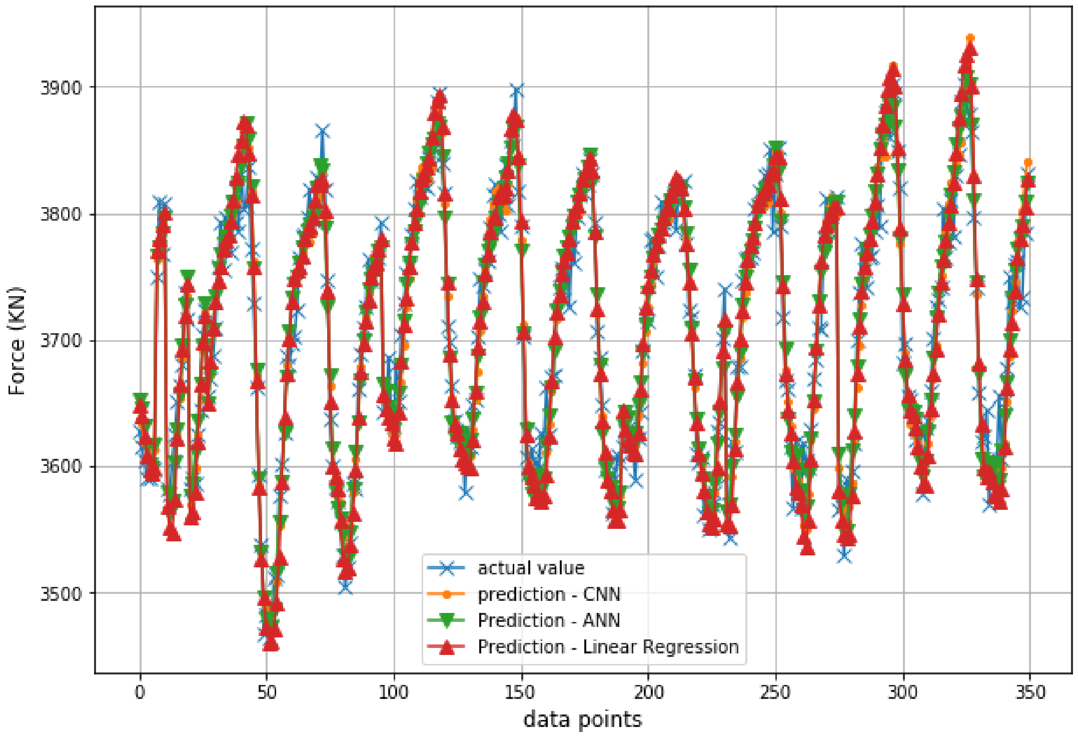


Figure 4.7 Comparison between the actual and the prediction values of EM12

In Table 4.4, it is shown that among the three models, the CNN-based models produce the lowest average value of MAE for all six sensors. Significant differences between the CNN-based model and the linear regression model are obtained in sensor EM6 and EM12. On these sensors, improvement higher than 1 KN of average MAE can be obtained by implementing CNN over linear regression. Furthermore, despite the small difference on other sensors, the CNN-based models still manage to yield lower average MAE compared to the linear regression model. In addition, the CNN-based model significantly outperforms ANN-based model on sensor EM6 and EM10. In these sensors, difference of higher than 1KN of average MAE values are obtained between the CNN-based and ANN-based models. In other sensors, lower average MAE values are still produced by the CNN-based model.

In term of MAPE and MSE, it can be seen from Table 4.4 that the CNN-based models also provide higher performance than the ANN-based and linear regression models. As shown in the table, the CNN models manage to yield lower MAPE and MSE compared to the other models for all sensors. From comparison of the average value of MAE, MAPE, and RMSE, it is obvious that the performance of CNN-based models is more superior than the other prediction models.

Table 4.4 Average MAE, MAPE, and RMSE from various prediction models

Sensor	MAE (kN)			MAPE (%)			RMSE (kN)		
	CNN	LR	ANN	CNN	LR	ANN	CNN	LR	ANN
EM2	10.23	10.28	10.49	0.469	0.470	0.480	13.38	13.47	13.62
EM4	10.47	11.12	10.65	0.472	0.502	0.480	13.64	14.48	13.80
EM6	13.17	17.03	15.22	0.434	0.559	0.500	16.94	21.62	19.55
EM8	16.19	16.81	16.41	0.516	0.535	0.523	21.14	21.68	21.21
EM10	18.39	18.56	19.48	0.460	0.464	0.488	23.27	23.60	24.73
EM12	19.82	21.24	19.84	0.536	0.574	0.537	25.32	27.51	25.62

In addition to the average value of the MAE, MAPE, and RMSE, the maximum values of those metrics are also calculated as presented in Table 4.5. From the table, it is shown that in term of maximum MAE, the CNN-based models also outperform both the linear regression and ANN-based models. This can be seen from the lowest maximum MAE in all six sensors generated by the CNN-based models. The low maximum MAE value can be beneficial when implementing threshold-based anomaly detection method since it can reduce the number of false positives that occurs when prediction is higher than the threshold defined by a certain confidence level.

On the other hand, there are cases when the CNN-based models are outperformed by the ANN-based models when observing the maximum MAPE and RMSE. As described in Table 4.5, in sensor EM2, CNN-based model produces maximum MAPE of 0.737%, higher than the 0.731% from the ANN-based model. In addition, in sensor EM8, the ANN-based model manages to produce lower RMSE than the CNN-based model. This might be caused by the lack of hyperparameter optimisation performed on the CNN-based models. As mentioned previously in section 4.3.4, the ANN model architectures for each sensor datasets are selected through a grid search process where a number of ANN architectures are constructed and the performance of these models on testing sets are collected and compared to find the best architecture for the given problem. In contrast, each CNN-based model only adopts a single architecture without involving hyperparameter optimisation. As a result, the CNN structure might not be the most suitable architecture in some problems hence higher error might be produced.

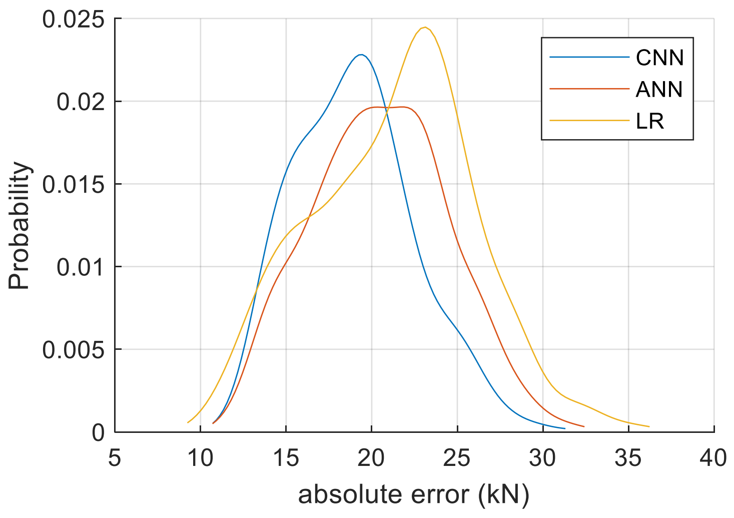
In order to further compare the performance of the prediction models, the probability distribution of the absolute error generated from these models are investigated. For this purpose, the results obtained using EM12 dataset are employed. The probability density functions (PDFs), and the cumulative distribution functions (CDFs) are illustrated in Figure 4.8.

Table 4.5 Maximum MAE, MAPE, and RMSE from various prediction models

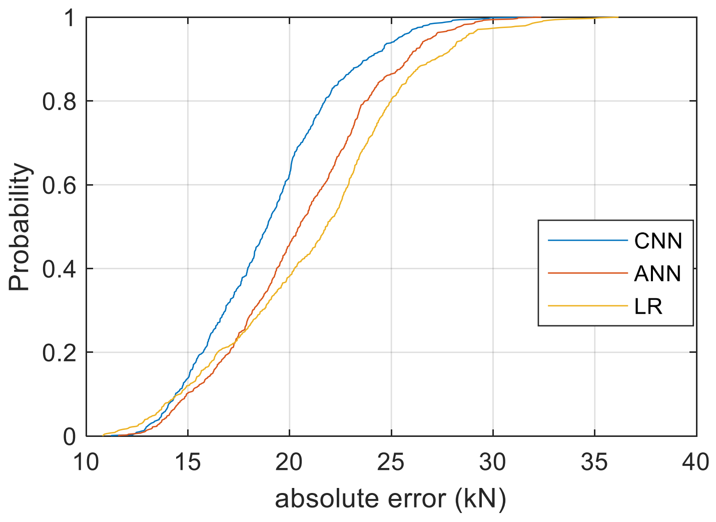
Sensor	**MAE (kN)**			**MAPE (%)**			**RMSE (kN)**		
	CNN	**LR**	**ANN**	**CNN**	**LR**	**ANN**	**CNN**	**LR**	**ANN**
EM2	15.74	16.60	15.95	0.737	0.777	0.731	25.27	27.54	25.67
EM4	16.22	17.85	17.79	0.738	0.805	0.789	21.33	24.02	22.80
EM6	23.55	32.36	27.91	0.789	1.062	0.895	33.86	45.92	44.63
EM8	29.19	30.18	34.29	0.947	0.988	1.095	46.64	46.79	45.08
EM10	27.31	30.46	32.04	0.700	0.779	0.819	35.82	42.87	43.08
EM12	32.43	36.17	32.84	0.870	0.970	0.892	41.16	46.82	40.41

As it is shown in Figure 4.8(a), the CNN model outperforms the other models in predicting the measurement values of EM12. As depicted in the figure, the mode from the absolute errors generated by the CNN model is 19.82 kN, lower than those produced by the ANN (21.17 kN) and LR models (22.77 kN). Furthermore, investigation on the effectiveness of the CNN model is also performed by comparing the maximum absolute error from the PDFs as shown in the figure. As it can be seen in the figure, maximum absolute error produced by the CNN model is lower than those generated by both the ANN and LR models. This result also confirms the result presented in Table 4.5.

From Figure 4.8(b), overall, it can be seen that the CNN model produces the lowest absolute error among all three prediction models. As illustrated in the figure, the median of the absolute error generated by the CNN model is lower than those produced by both the ANN and LR models. This result indicates that 50% of absolute errors generated by the CNN model is below 18.9 kN, lower than those of the ANN and LR models which are 20.47 kN and 21.8 kN respectively. In addition, among all models, the CNN model produces the lowest value for both the lower (16.2 kN) and upper quartiles (21.7 kN). In this case, most of the absolute errors generated by the CNN model is lower than those produced by both the ANN and LR models. Moreover, it is also shown that the CDF of the CNN model has steeper slope compared to both the ANN and LR model thus the CNN model produces higher prediction confidence and less variation compared to other models.



(a)



(b)

Figure 4.8 PDF (a) and CDF (b) of the absolute errors generated by the CNN, ANN, and LR models

The CNN models manage to outperform both the ANN and LR models due to the inherent capability of CNN in processing time history data. As it has been mentioned in section 4.3.2, all prediction models including the CNN, ANN, and LR models, accept an input consisting of 24 sequential data which correspond to a daily measurement. These sequential data form patterns and by using moving kernels on the input data, the CNN models extract these trends from the sensor data. On the other hand, both the LR and ANN models are unable to obtain sequential information from input data and compared to both models, CNNs perform better in finding local pattern from the input data. In this case, the CNN models are able to capture the daily pattern from the cable force data thus producing less error compared to the other prediction models.

4.5 Conclusions and Summary

In this study, a method implementing CNN for cable force estimation from Suramadu Bridge monitoring system using temperature data as the input has been evaluated. In this research, measurement data from six elasto-magnetic sensors and six temperature sensors installed in Suramadu Bridge have been used to train and test CNN-based prediction models. In addition, evaluation on these models have been performed by calculating the prediction errors in the form of MAE, MAPE, and RMSE. To validate the effectiveness of the proposed method, ANN and linear regression models for cable force estimation have been trained for comparison. The conclusions are as follows:

- It is found that the proposed CNN framework manages to capture the trend of cable force sensor measurements with the ranges of MAE between 10.23 kN and 19.82 kN, MAPE between 0.434 % and 0.536 %, and RMSE between 13.38 kN and 25.32 kN.
- From the comparison of error metrics produced by these models, it is found that the CNN-based models manage to outperform both the linear regression and the ANN-based models.
- Despite producing the lowest value in all error metrics, the CNN-based method does not produce significant improvement compared to the linear regression- and ANN-based methods. In this work, there is a strong linear relation between cable force and the temperature, hence the linear regression model can be implemented. However, in other cases where there is no linear relation between the predictor and the output, the linear regression might not be suitable thus significant improvement might be detected.
- The CNN-based method has the potential for both generating estimation of cable force measurement in case of missing data and performing anomaly detection which can contribute to the SHM application.

Chapter 5 Damage Detection and Identification Utilising Convolutional Neural Networks for Structural Health Monitoring

Summary

This chapter presents deep learning-based damage detection techniques for SHM. In this chapter, there are two damage detection approaches discussed: novelty detection and multiclass classification. Both methods employ CNN models to process raw measurement data in detecting the presence of damage in the structure. Section 5.2 explains the methodology performed in this research. Section 5.2.1 discusses the methodology for the novelty detection employing the correlation between sensor measurements whereas section 5.2.2 provides the methodology for damage detection using multiclass classification. Section 5.3 presents the case study utilised to validate the proposed methods. The results from the implementation of the proposed methods on the case study are presented in section 5.4. Finally, section 5.5 summarises and concludes the chapter.

5.1 Introduction

in Chapter 4, a deep learning-based method that is used for estimating structural responses for SHM application has been presented. As it has been mentioned in section 2.1, one of the main research areas of SHM is damage detection. In this chapter, methodologies for damage detection on structures utilising deep learning are discussed.

In this chapter, two damage detection approaches implementing deep learning are proposed: novelty detection and damage classification through supervised learning. The former aims to find abnormalities from the normal condition. In this case, a statistical model is trained only using data collected from undamaged state. Then, new data are tested on the trained model where deviation between the model prediction and the actual measurement is compared with a pre-defined threshold that defines data normality.

There are two monitoring strategies that are generally adopted in structure monitoring: dynamic and static monitoring. The dynamic monitoring approach has a main drawback whereby significant damage might only result in a small shift in the natural frequencies, especially for complex structures [113]. Furthermore, the technique also suffers from the presence of noise in the data [113]. On the other hand, static monitoring utilises static structural responses (strain or displacement) due to applied static loading. Despite the higher sensitivity to structural changes compared to the dynamic monitoring, static monitoring requires information about the applied load. Correlation-based methods that are not affected by load change are implemented to address this problem. The methods can be employed for damage detection based on novelty detection approach.

In addition to the novelty detection approach, damage detection in SHM can also be performed in multiclass classification. In multiclass classification, the structure’s conditions are categorised into a number of classes including undamaged and damaged states. All data are provided their corresponding labels based on the structure’s condition when the data are collected. While the undamaged state only has one class, the damaged state might consist of one or more classes which might be defined either from the types of damage or the locations of damage. Therefore, it might allow identification of damage types or damage localisation.

In multiclass classification problem, damage detection is conducted as pattern recognition problem where a prediction model is trained in supervised learning using both the features extracted from the training data and their corresponding labels. Then, by applying the features extracted from new data, the trained model predicts the condition of the structure. Conventionally, feature extraction is performed manually based on the professional experience of the researcher. This step requires high level of knowledge in the domain and might potentially reduce the performance of the prediction model when unsuitable features are used.

In this chapter, two damage detection techniques that employ deep learning are presented. First, we propose a method that utilises correlation between sensor measurements and employs deep learning architecture in detecting damage on the structure based on the novelty detection. Then, a deep learning framework that performs multiclass classification for damage detection and identification using SHM data is presented. Both methods employ raw measurement data from several types of sensors as the input without the need of feature extraction. By employing deep learning technique, features are learned automatically, effectively reducing the level of expertise required in feature extraction. The frameworks are validated using experimental data obtained from a laboratory-scale bridge.

5.1.1 Summary of Novelty and Contribution

There are two damage detection methods presented in this chapter. The first method is utilised on experimental data obtained from a laboratory-scale bridge. This research is aimed to develop a novel damage detection method employing 1-D CNN regressor. The proposed approach performs damage detection using raw data as input without the need of feature extraction, potentially reducing the required level of expertise in implementing the data-based interpretation for SHM. In addition, the proposed method also employs different types of sensors in the interpretation which shows the capability of the method for the implementation on real monitoring system that might consist of numerous types of sensors. This work also investigates the effectiveness of the method by comparing between the method and other approaches such as MLR, ANN, and Random Forest.

This chapter is also aimed to investigate the damage detection method that adopts CNN models that are trained through supervised learning. In this method, all data are labelled according to the bridge's condition when the data are collected. Similar research that employed CNN trained in supervised way mostly only utilised one type of sensor in the interpretation. On the other hand, this study employs more than one types of sensors for damage detection in order to demonstrate the applicability of the method in utilising multi-type sensors. This work also presents the investigation on the impact of CNN hyperparameters on the damage detection which has not been reported previously. For this purpose, two CNN hyperparameters including the network depth and the activation function are utilised in the observation. The results of the hyperparametric study might benefit other researchers that are employing damage detection using CNN model trained using supervised learning.

5.2 Methodology

5.2.1 Correlation- Convolutional Neural Network (CorCNN)

Figure 5.1 shows the schematic of the proposed CorCNN. The main idea of the proposed method is built on an assumption that when damage occurs, correlations between measurements are also affected. Calculating the correlation between sensor measurements enables us to estimate signal from one sensor by using measurements from other sensors. The intention is to detect damage by monitoring the residuals produced from the difference between the actual values of the bridge response and its predicted values. The latter is a result of a prediction model that has been trained using healthy datasets. In this case, we assume that higher residuals might be produced when using the prediction model on damaged state datasets due to the change in sensors correlation.

The next process involves normalisation process on both healthy and damaged datasets. In this step, all sensor signals are normalised using statistical normalisation (4-1). This process is applied on all data from both healthy and damaged datasets.

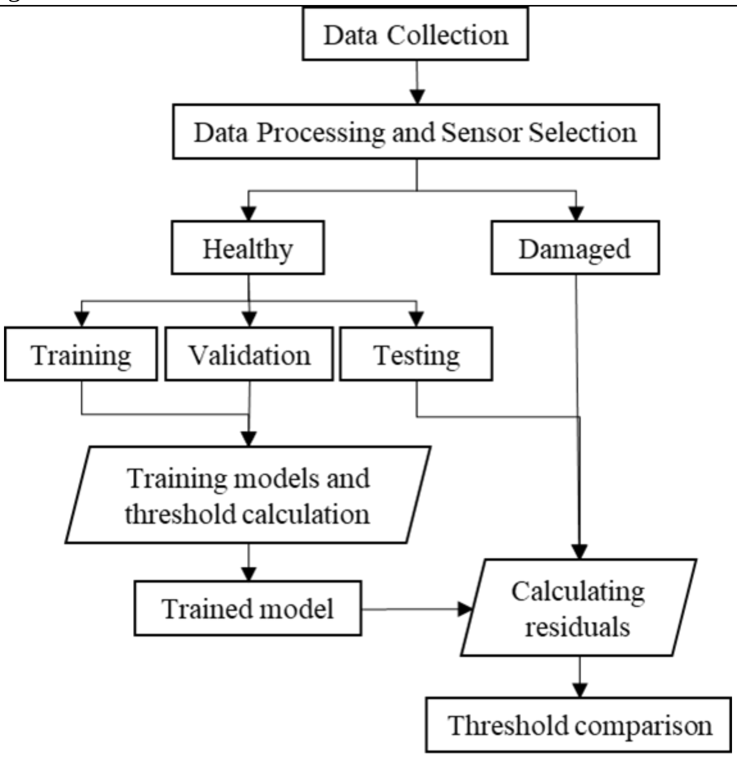


Figure 5.1 Schematic of the damage detection method using CorCNN

In the next step, all the measurement signals that are used as the inputs are divided into data frames. Consider a prediction model trained using j number of sensors to perform a regression to predict a value of a sensor O. If all sensors consist of n measurement points, the sensor data used for the input and the output can be represented as:

$$S_i = [S_{i.1}, S_{i.2}, S_{i.3}, \ldots., S_{i.n}] \tag{5-1}$$

$$O = [O_1, O_2, O_3, \ldots., O_n] \tag{5-2}$$

where S and O represent the sensor data utilised as the input and output in the prediction model, respectively. The S_i represents the sensor data for the i-th sensor ($i = 1, 2, \ldots, j$). The sensor signals are divided into a number of data frames each having a fixed length ns:

$$S_{i.n} = [S_{i.n}, S_{i.n+1}, S_{i.n+2}, \ldots., S_{i.n-1+ns}] \tag{5-3}$$

The data frames created from j sensors are then employed to predict a single value of sensor O as:

$$O_n = [O_{n-1+ns}] \tag{5-4}$$

For damage detection purpose, the strategy in sensor selection is as follows:

1. Select one sensor that will be used as the target output of the regression model.
2. Select other sensors for the predictors. These sensors should be located relatively far from the sensor selected in the first step. In addition, the sensors picked in this step should be close to each other.
3. Use sensor(s) selected in the second step as an input of CNN regressor to estimate the measurement of the sensor selected in the first step.

After the data frames have been generated, the dataset containing the frames from healthy state measurement is further split into testing, validation, and training sets. In this research, the supervised training process of the prediction model is only performed using data collected in healthy state. Hence, this separation process is only performed on the healthy dataset. The training set is utilised for the supervised training of CNN models hence it is important that the training data capture all the variance of the healthy state data. On the other hand, the validation set is employed for both hyperparameter optimisation and defining a threshold level. The hyperparameter optimisation is conducted by performing grid search [215] using various CNN architectures. These architectures are trained using the training set and tested using the validation set and the architecture that yields the lowest validation error is picked as the baseline prediction model.

As shown in Figure 5.1, the optimised prediction model is employed on the testing set and the damaged datasets. The testing data represents healthy state measurement that has not been seen by the prediction model in the training process. The residuals generated from the discrepancies between the actual and prediction values when applying the trained model on both healthy and damaged datasets are recorded for damage detection.

The presence of damage is detected by comparing the residuals with a threshold level. Assuming the distribution of the regression residuals from the healthy dataset used as the baseline follows normal distribution, then a threshold that defines the interval of the distribution can be calculated [84]:

$$threshold = \mu_b + 6\sigma_b \tag{5-5}$$

where μ_b and σ_b are the mean value and the standard deviation of the regression residuals produced from healthy dataset (in this case validation dataset), respectively. Anomalous data that might indicate the presence of damage is detected when a prediction residual is higher than the threshold value.

For damage detection application, the performance of the regression model is determined by calculating the total number of data detected as anomalies. The number of detections is then calculated by comparing the regression residuals with the threshold level. In this case, all residuals exceeding the threshold value are considered as anomalies. Then, the detection rate is calculated by:

$$Detection\ rate = \frac{total\ number\ of\ detection}{total\ number\ of\ samples\ in\ the\ set} \tag{5-6}$$

5.2.2 Damage Detection and Identification Using Supervised CNN

Figure 5.2 describes the activities conducted in this project. As it can be seen from the figure, there are four main activities performed in this research such as:

1. Data collection
2. Data processing and labelling
3. Prediction models training and testing
4. Evaluation

Data collection is carried out by measuring and storing the bridge’s responses using sensor system deployed on the bridge. In data processing step, normalisation is performed on all measurement signals. For this purpose, statistical normalisation as explained in (4-1) is executed. In this research, normalisation plays an important role since the proposed method might combine measurement signals from various sensors. Each type of sensor has unique measurement unit hence combining these signals requires normalisation to avoid the dominance of extreme values and enhance data quality. This step is applied on all sensor signals on both the healthy and damaged state datasets.

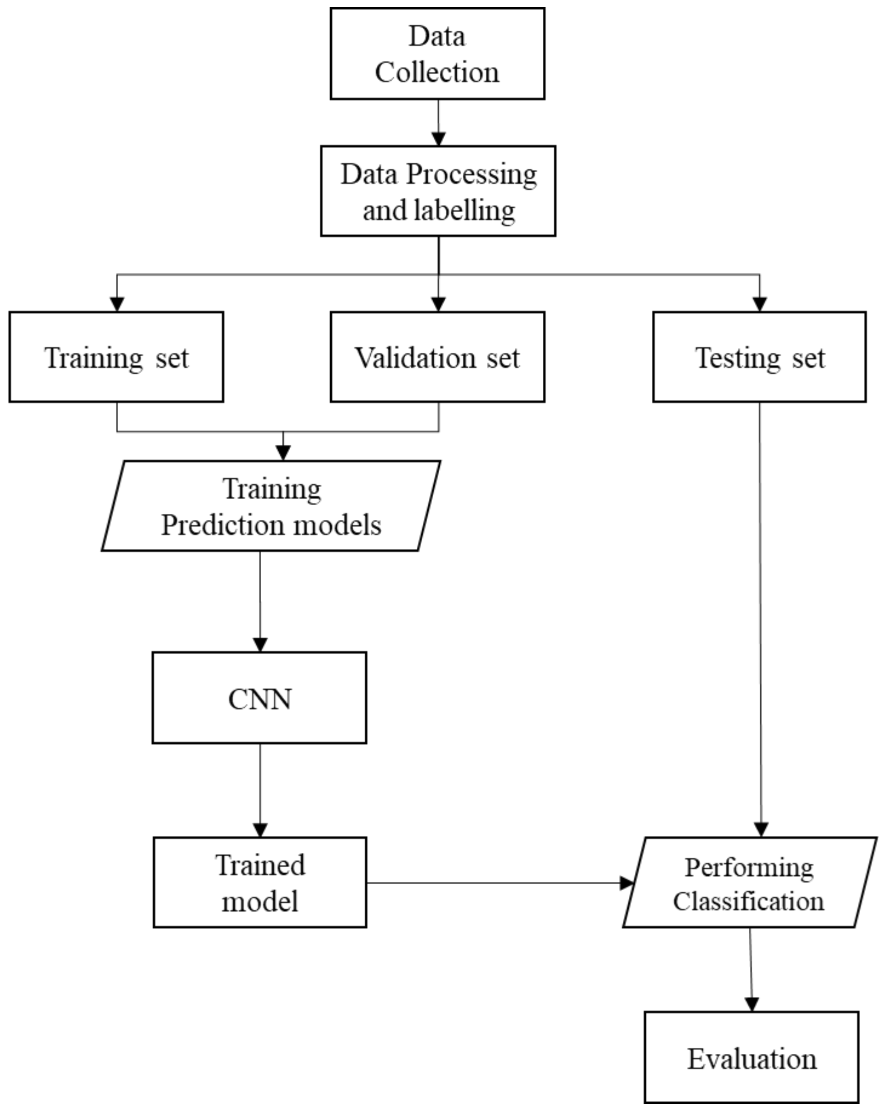

Data Collection
Data Processing and labelling
Training set
Validation set
Testing set
Training Prediction models
CNN
Trained model
Performing Classification
Evaluation

Figure 5.2 Framework for CNN-based damage detection method using supervised learning

Consider a sensor S_i consisting of n measurement points as written in (5-1). The sensor signals are then divided into data frames each having a fixed length of n_s points. When j sensors are employed as the input of the prediction model, then a number of data frames $D(t)$ each consists of $n_s \times j$ data as it is shown in (5-7):

$$D(t) = \begin{cases} [S_{1.t}, S_{1.t+1}, \dots, S_{1.t+n_s}] \\ [S_{2.t}, S_{2.t+1}, \dots, S_{2.t+n_s}] \\ \vdots \\ [S_{j.t}, S_{j.t+1}, \dots, S_{j.t+n_s}] \end{cases} \tag{5-7}$$

Each frame generated in this process is annotated with its corresponding label. The label corresponds to the bridge’s condition when the data are collected. For example, the frames generated from the data collected in the bridge initial state is labelled as ‘healthy’. On the other hand, the data frames produced using data obtained from damaged state are labelled as ‘damaged’. Furthermore, for the damaged state data, depending on the number of the damage types, multiple labels can be implemented. The prediction models are trained to predict these labels using raw measurement data of sensors as the input. Therefore, by predicting these labels, the presence and severity of damage in the bridge is detected. All frames from the healthy and damaged state are collected into one dataset.

The next step is dividing the dataset containing data frames into training, testing, and validation sets. In this step, any overlap between training, validation, and testing sets should be avoided so that the testing set consists of new data frames that are completely unseen from the training process. Training data are employed to train prediction models. On the other hand, validation data are also utilised in the training process. However, unlike the training data that are only use in the supervised learning process, the validation data are adopted for both hyperparameters optimisation and early stopping method. The explanation of early stopping method implemented in this study is provided in the next section. Finally, testing data serve as new data that have not been seen by the prediction models. Hence the prediction models are evaluated based on their performances on unseen data.

In this research, supervised learning is implemented hence the class label of each sample needs to be provided in the training process. The labels are converted into vectors using one-hot encoding method. The number of elements in the vector depends on the number of class labels adopted in the case study and each element represents the probability of a data frame belonging to a particular class. In this way, a data frame with healthy label should have 100% in the healthy class and 0% on the other classes.

Hyperparameter optimisation is performed by using the training and validation sets. In general, the simplest method for the optimisation is conducted by performing trial and error on architectures adopting a number of combinations of hyperparameters. The training set is employed to iteratively train prediction models while the validation set is utilised to obtain a prediction score in the form of validation accuracy. Finally, the combination of hyperparameter that yields prediction model with the highest validation accuracy is picked as the optimised hyperparameter.

In the training process, early stopping is employed in order to combat overfitting. For classification problem, the method is executed by monitoring either the validation loss or validation accuracy generated from applying the validation sets on the prediction model on every epoch of the training process. When no decrease in validation loss or no increase in accuracy is achieved after a predefined number of epochs, which is also known as early stopping patience, the training process is finished. Finally, the prediction model performance is evaluated by calculating the prediction accuracy obtained using (3-1).

5.3 Case Study: Warwick Bridge

The deep learning-based approaches proposed in this research are validated using experimental data that have been collected from Warwick Bridge experiment. This section provides information about Warwick Bridge, the sensor system deployed in the bridge, the description of the tests performed on the bridge, and the implementation of both the CorCNN and supervised CNN for damage detection.

Warwick Bridge (Figure 5.3) is a simply supported bridge (with overhangs) located in the Structures Laboratory at the University of Warwick [216]. The deck is 20 m long and 2 m wide and is constructed using class 40/50 concrete. In addition, two steel I-profiles that are 1.1 m apart from each other are placed longitudinally below the concrete deck. The steel-concrete composite structure has structural mass of 16500 kg. The bridge has two adjustable supports with 16 m distance between each other during the test. Detailed information about the bridge is available elsewhere [216].

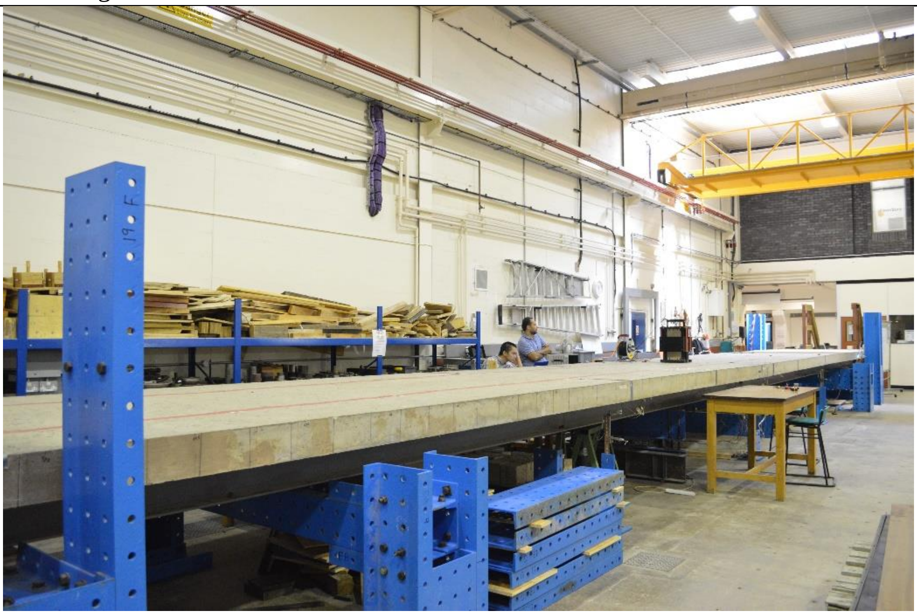


Figure 5.3 Photograph of Warwick Bridge in University of Warwick

In this study, measurements were performed for both healthy and damaged states of the bridge. The healthy state represents the state of the bridge at the start of the experimental work. In addition, three damage scenarios were implemented by introducing artificial damage to the bridge. The damage location can be seen in Figure 5.4. In the first damaged state, six circular holes with 2.5 cm diameter were drilled on the concrete deck (circled in Figure 5.5). In the second damaged state, six holes with the same diameter were added, as represented by remaining holes in Figure 5.5. Finally, in the third damaged state, 5 cm cut was added on the lower flange of the steel beam. The cut was only performed at one half of the flange on one beam located at the lab side as illustrated in Figure 5.4. Figure 5.4 also describes information on sensor installation on Warwick Bridge. The sensor system deployed in this study is explained in section 5.3.1.

The damage location was selected by taking into account both the expected displacement in the deck and the sensor location. The middle span of the bridge is expected to have the highest displacement thus damage might potentially occur in the middle span compared to other location. However, in the experiment, it was challenging to introduce damage in the middle span since some sensors were installed in the middle span area. As a result, the location illustrated in Figure 5.4 was selected. Most importantly, due to the safety issue, only one damage location was employed in the experiment.

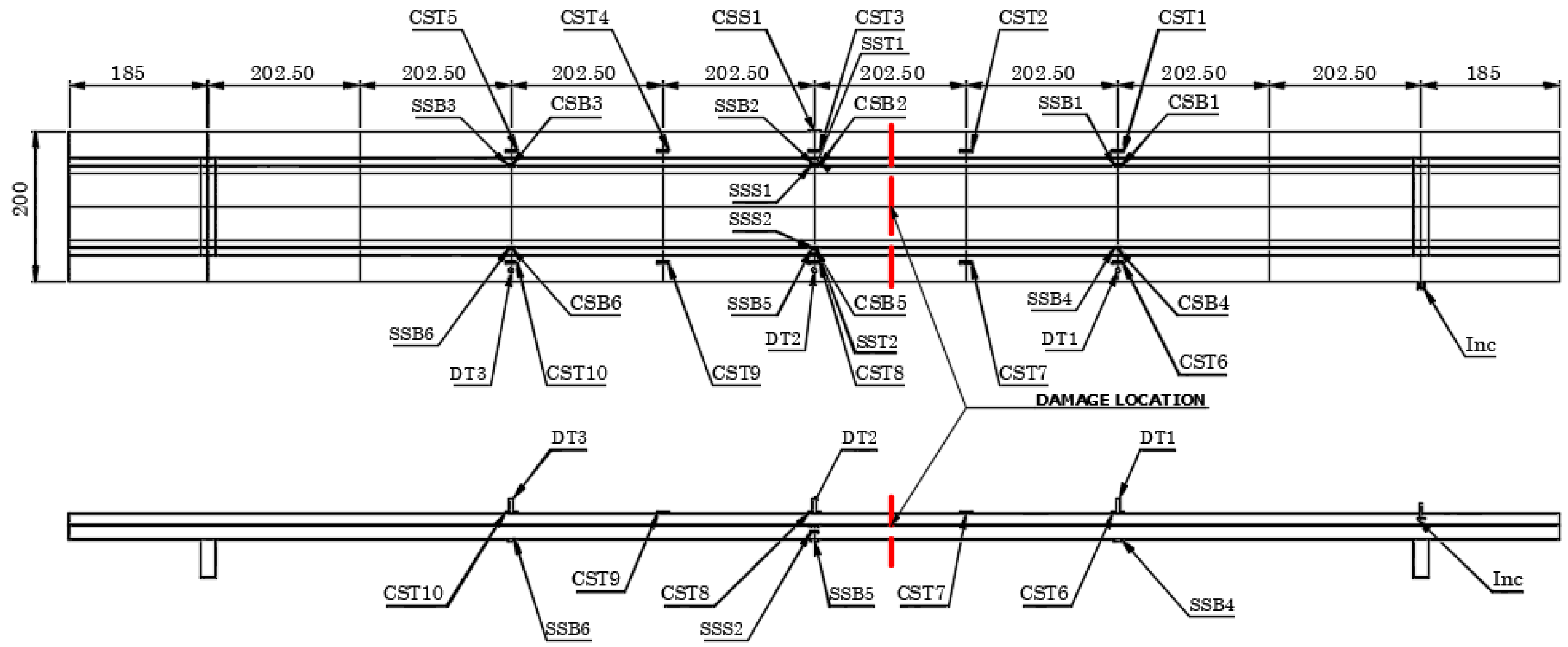
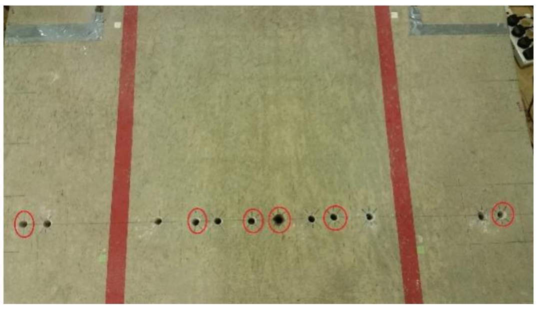


Figure 5.4 Detail of sensors and damage location at Warwick Bridge experiment. All dimensions are in m.


(a)

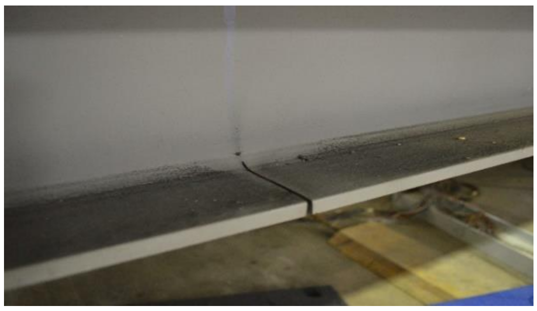

(b)

Figure 5.5 Artificial damage introduced to Warwick Bridge. (a) 12 holes in concrete deck. (b) 5 cm cut on the I-beam.

5.3.1 Structural Health Monitoring (SHM) Configuration

In this section, the sensor system employed in Warwick Bridge experiment is explained. This section presents all sensors and their corresponding DAQ utilised in the sensor system. In addition, the installation and calibration steps for each sensor are presented. Finally, the configuration used in the interface program is discussed.

To collect measurement data from the bridge, a wired-based SHM system consisting of 18 strain gauges (10 to measure strain in steel, 8 in concrete), three displacement transducers and eight accelerometers (to measure shaker force and bridge accelerations) was deployed. Details of the sensors and their exact roles are listed in Table 5.1. In addition, the sensor location is described in Figure 5.4. Table 5.2 describes the notation used in Figure 5.4.

Table 5.1 Details of instrumentation utilised in Warwick Bridge experiment

Sensor	**Quantity measured**	**Type**	**Quantity**
Strain gauge	Strain in steel	YEFLA-5-3LJC	10
Strain gauge	Strain in concrete	PL-60-11-1LJC	18
Displacement transducer	Displacement of the bridge	LSC HS50	2
		Celesco SP1-50	1
Accelerometer	Acceleration of the bridge and shaker's moving mass	QA750 (Honeywell)	8

Table 5.2 Description of notations used in Figure 5.4

Notation	Remark
CST	Strain gauges on the top side of the concrete deck
CSB	Strain gauges installed at the bottom side of the concrete deck
SSB	Strain gauges installed at the side of the concrete deck
SST	Strain gauges installed at the upper flange of the steel beam
SSB	Strain gauges installed at the bottom flange of the steel beam
SSS	Strain gauges installed at the web of the steel beam
DT	Displacement transducer

5.3.1.1 Strain Gauge

As shown in Figure 5.4, each of 18 concrete strain gauges was installed on the concrete deck in one of the three candidate positions: on the top of the concrete deck (denoted as CST in Figure 5.4), at the side of the deck (denoted as CSS), and below the deck (denoted as CSB). The steel strain gauges were also installed in one the three possible locations: at the upper flange of the I-beam (denoted as SST in Figure 5.4), at the lower flange (denoted as SSB), and at the web (denoted as SSS).

There are four steps required in strain gauge installation: surface preparation, sensor installation, connecting sensor to DAQ, and sensor calibration. Among these, only the first step is different depending whether the strain gauge is installed on the concrete or on the steel beam. The other three steps are similar for both types of strain gauges.

Surface preparation for concrete strain gauge installation is performed to provide the required base to install the gauge. This step is crucial since the sensor performance depends heavily on the base condition. The process is started by filing the concrete surface using sandpapers in order to provide a flat surface for the strain gauge. The process is then followed by dust removal using acetone. In this step, the dust scattered due to the filing process is wiped so the base can be attached on the surface. Lastly, pressure sensitive (PS) adhesive is applied on the prepared surface as the base for strain gauge installation. In this step, a ruler can be used to help developing a flat base. After the PS adhesive has been applied in the concrete, 5-6 hours are required to cure the adhesive. The surface preparation process for concrete strain gauge installation can be seen in Figure 5.6.

Unlike the concrete strain gauge installation, no additional base is required in steel strain gauge installation. The surface preparation for this type of sensor is performed by filing the steel beam using sandpapers. After the filing process, dust is removed by using acetone.

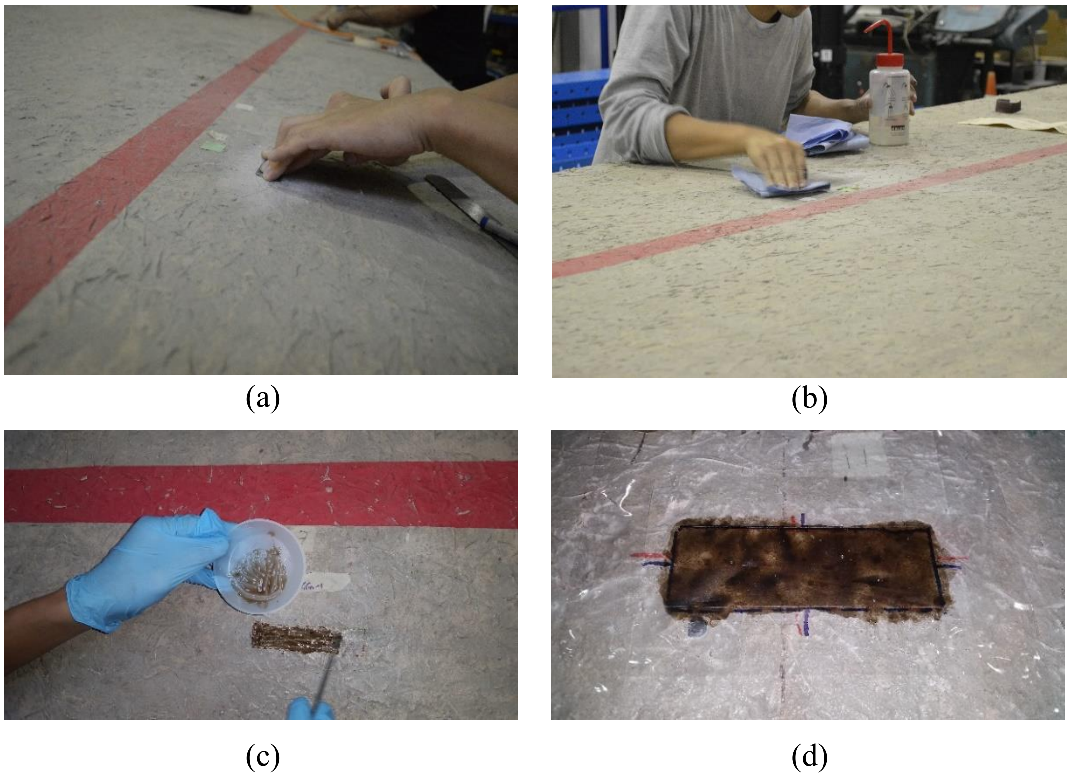


Figure 5.6 Surface preparation process for strain gauge installation on concrete deck; (a) Filing the surface with sandpaper; (b) Dust removal using acetone; (c) Pouring PS Adhesive on the concrete surface; (d) Installation base for strain gauge.

The installation methods for both concrete and steel strain gauges are similar. The only different part in this step is the types of glue for attaching the sensor. For this purpose, the YEFLA-5-3LJC [217] and PL-60-11-1LJC [218] use CN-Y adhesive and CN adhesive, respectively. Figure 5.7 illustrates the strain gauges that have been installed in the experiment.

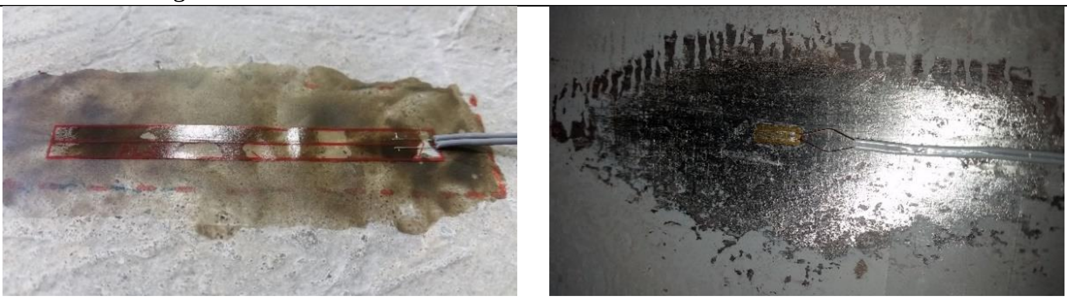

(a) (b)

Figure 5.7 Installed strain gauges on Warwick Bridge. (a) Concrete strain gauge. (b) Steel Strain Gauge.

After the sensors have been attached to the structure, connections between these sensors and the DAQ were made using wires. In the experiment, awg24 wires of four different colours (red, black, yellow, green) were used as a connection between sensor and DAQ. Each strain gauge has two terminals. In the experiment, one terminal of the sensor was connected to the red and green cables while the other was connected to the black and yellow cables.

In this research, four NI 9235 cards from National Instrument were utilised for collecting data from both the concrete and steel strain gauges [219]. These cards were selected due to the resistance of the strain gauges which is 120Ω. Each card has eight inputs for quarter-bridge measurement. To connect the strain gauges with the cards, three-cable configuration was implemented. Table 5.3 shows the wiring configuration for strain gauges implemented in this research.

Table 5.3 Wiring configuration for strain gauges in Warwick Bridge experiment

Cable Colour	Port in NI 9235
red	EXC
green	not connected
black	AI
yellow	RC

In the research, the calibration step for strain gauges was only performed once before any test was conducted. It was performed in order to remove the offset from the strain measurement. It was performed when no load was applied on the bridge using the internal calibration function provided by the NI 9235 cards. In this experiment, calibration is performed for every strain gauge by using the 50kΩ internal shunt resistor in the card. From the calibration step, the initial voltage parameters for each individual strain gauges were obtained. The parameters employed for strain gauges are illustrated in Table 5.4.

Table 5.4 LabView configuration for strain gauges data collection by using NI 9235 card

Parameter	**Strain Gauge**	
	Concrete	**Steel**
Max range	1m	
Min range	-1m	
Scaled Units	Strain	
Gage Factor	2.08	2.11
Gage Resistance	120.3	119.8
Initial Voltage	from calibration	
Vex Source	Internal	
Vex Value	2	
Strain Configuration	Quarter Bridge I	

5.3.1.2 Displacement Transducer

In addition to the strain gauges, three displacement transducers (denoted as DT) were deployed to monitor the bridge displacement as it is described in Figure 5.4. Two types of displacement sensors are utilised to measure the displacement produced in vertical direction in this research. This is due to the possibility of getting displacement higher than 10 cm in the mid span of the bridge. In order to ensure the safety of the sensor, a string gage displacement sensor (denoted as DT2 in Figure 5.4) with high operational range is installed for measuring the displacement in the mid span. On the other hand, two strain-based displacement transducers (denoted as DT1 and DT3) with lower operational range are employed as it is shown in Figure 5.4.

The string gauge sensor used in the measurement is SP1-50 [220]. The sensor has voltage divider output where voltage output will be determined by the length of string. The sensor can work up to 1270mm with 0.25% of accuracy. The string gauge sensor was mounted at a construction material that was built below the bridge as shown in Figure 5.8. In addition, to provide connection with the bridge, a bracket was glued at the bottom of the concrete deck and the string was connected to the bracket as illustrated in Figure 5.8.

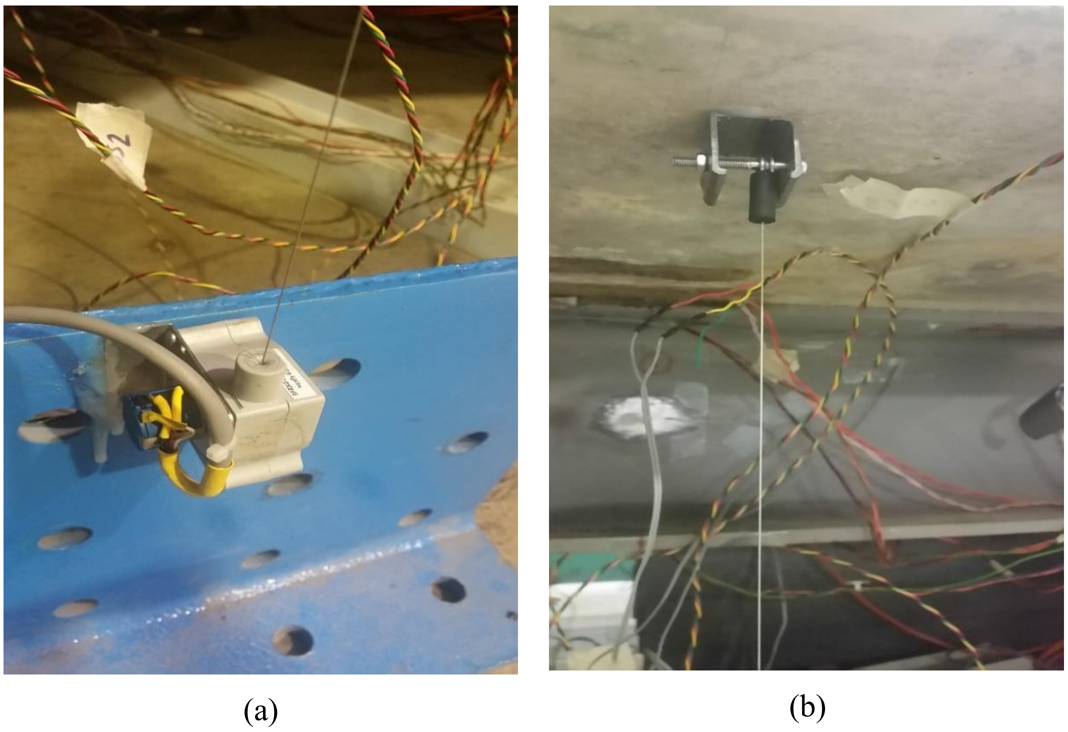


Figure 5.8 installation of string-based displacement transducer on Warwick Bridge; (a) Sensor body mounted on a supporting material; (b) Sensor eyelet attached on the bottom of the concrete deck.

To collect data from this sensor, an NI 9219 card [221] from National Instrument was employed. This card has four channels which can be used for voltage, current, 4-wire resistance, 2-wire resistance, thermocouple, 4-wire Resistance Temperature Detector (RTD), 3-Wire RTD, Quarter-Bridge, Half-Bridge, Digital input, and open contact measurement. The SP1-50 utilises voltage divider as the output. Therefore, the voltage measurement function provided by this card was implemented to obtain data from this sensor.

The sensor has four cables: signal, ground for signal, power, and excitation ground. In order to operate the sensor, an external power supply providing 30V excitation was utilised. The cable configuration for this sensor is provided in Table 5.5.

Table 5.5 Wiring configuration for string-based displacement transducer

Cable Colour	**NI 9219**		**Power Pack**
	Number	**Port**	
Red	-	-	+ terminal
Black (tangled with red cable)	-	-	- terminal
White	4	HI	-
Black (tangled with white cable)	5	LO	-

In addition to the string-based displacement transducer, two strain-based displacement transducers HS-50 from LSC Transducer [222] were utilised to measure displacement as shown in Figure 5.4. The sensor has full-bridge configuration with 350Ω bridge resistance. It can be used to measure up to 53.5mm of displacement with 0.35% accuracy. The sensor has a moveable spindle which serves as a sensing element. The sensor was mounted using a material construction with the sensing element touching the concrete deck as described in Figure 5.9. In order to measure displacement when the bridge is moving upside, the sensing element was initially positioned at 20mm stroke.

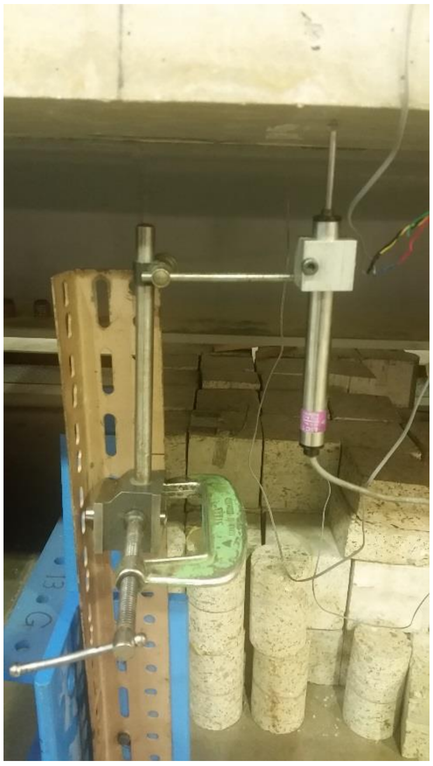


Figure 5.9 Installation of strain-based displacement transducer in Warwick Bridge experiment

The NI 9237 from National Instrument [223] was implemented to collect data from the strain-based displacement transducers. This card was selected due to its capability in performing a full bridge measurement. This card has four RJ50 inputs hence a connector is required to provide interface between the displacement transducer and the DAQ. The wiring configuration between the strain-based displacement transducer and DAQ is provided in Table 5.6.

Table 5.6 Wiring configuration for strain-based displacement transducers

Cable Colour	**NI9237**	
	Number	**Port**
Red	6	EX+
Blue	3	AI-
Green	2	AI+
Yellow	7	EX-

In this research the calibration process was conducted for all three displacement sensors. The DAQs for the displacement transducers still produce measurement outputs in voltage (for NI 9219) and strain (for NI 9237) hence they need to be converted into displacement. Equation describes the relation between the displacement and the measurement output from the DAQ.

$$d = m \times o + c \tag{5-8}$$

where d is the displacement (mm), o is the reading from the DAQ, m is the slope, and c is the offset. The calibration process aims to obtain the linear relation between the sensor output and the displacement. The slope and the offset obtained from the linear expression are then used to measure the displacement.

Firstly, the calibration was performed on the strain-based displacement transducer. To apply a certain level of displacement on the sensor, gauge blocks were used. In this step, an NI 9237 card was utilised as the DAQ. In the calibration, displacement was applied gradually in steps from 0 mm to 40mm with an increment of 10mm. During this process, data were collected using the NI 9237 card and from this process, the linear relation between the DAQ output and the displacement was obtained.

The string-based displacement sensor was then calibrated using the calibrated strain-based displacement sensor. This was completed by connecting the string of the string-based sensor with the sensing element of the calibrated displacement sensor. NI 9219 was employed to collect the voltage output from the string-based displacement transducer while NI 9237 was utilised for the calibrated sensor. To supply the string-based sensor, an external power supply was deployed. By moving the spindle of the calibrated strain-based sensor, measurement data for both the string-based and the calibrated displacement transducers were recorded. From the collected data, the linear expression between the output voltage of the NI 9219 and the actual displacement of the string-based sensor was derived. The coefficient for the linear expression obtained in the calibration process is provided in Table 5.7.

Table 5.7 Calibration coefficient for displacement transducers

Sensor	m	c	Remark
DT1	27975	2.564	Obtained using 30.1V Excitation
DT2	45.157	7.788	
DT3	27690	1.759	

In the last step of the displacement transducers installation, the calibrated sensors were connected to their corresponding DAQ. The configuration for the NI 9219 and NI 9237 cards can be seen in Table 5.8 and Table 5.9 respectively.

Table 5.8 LabView configuration for string-based displacement transducer by using NI 9219

Parameter	NI 9219
Max range	32
Min range	0
Terminal configuration	Differential
Scaled units	Volts

In the Labview programming, a visual instrument (VI) named ‘formula’ was utilised to convert the output of DAQ into displacement unit (mm) based on equation (5-8). For the strain-based displacement transducer, this process was performed by simply using the VI and inserting the calibration constants that have been obtained from the calibration step. On the other hand, different method was conducted for the string-based displacement sensor. Unlike, the strain-based displacement sensor that was powered using the DAQ, the string-based displacement sensor required an external power supply as the excitation. The string-based sensor consists of a voltage divider circuit as it can be seen in Figure 5.10.

Table 5.9 LabView configuration for strain-based displacement transducers by using NI 9237

Parameter	**NI 9237**
Max range	2me
Min range	-2me
Scaled units	Strain
Gage Factor	2
Gage resistance	350
Initial Voltage	0
Vex source	Internal
Vex Value	2.5
Strain configuration	Full Bridge I

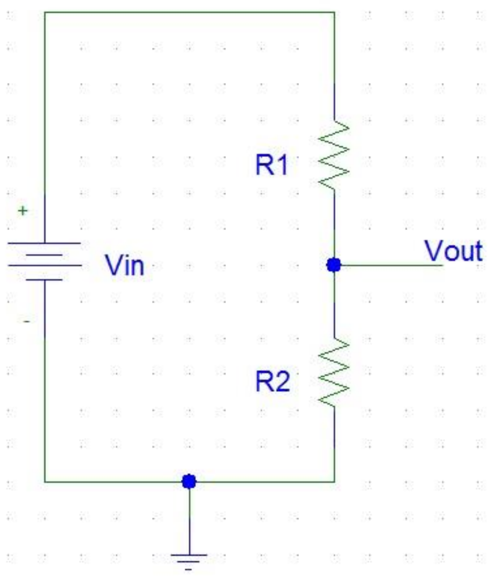


Figure 5.10 Schematic diagram of string-based displacement transducer used in Warwick Bridge experiment

From Figure 5.10, the V_{out} of the sensor is given by:

$$V_{out} = V_{in} \times R2 \tag{5-9}$$

where V_{out} is the measured output voltage of the sensor, V_{in} is the excitation voltage, and R2 is the variable resistance whose value changes with the change in displacement. From equation (5-9), it can be seen that the value of V_{in} should be constant thus the V_{out} can only be affected by the value of R2. However, due to the unstable power supply utilised in the experiment, the sensor output was also sensitive to the variation in the excitation voltage. Therefore, rather than using V_{out} for displacement calculation, $R2$ was utilised to tackle this problem. For this purpose, the value of V_{in} was also monitored. By using V_{out} and V_{in}, the value of R2 was obtained as follows.

$$R2 = \frac{V_{out}}{V_{in}} \tag{5-10}$$

The relation of displacement and DAQ reading is given in (5-8) where d is the displacement and o is the voltage reading for NI 9219. Substituting (5-8) with (5-10), the displacement is given by:

$$d = m \times o + c = m \times R2 + c = m \times \frac{V_{out}}{V_{in}} + c \tag{5-11}$$

where m and c are the slope and offset obtained from the calibration process.

5.3.1.3 Accelerometer

In this experiment, QA 750 accelerometers from Honeywell [224] were installed to measure the vibration of the bridge. Unlike the strain gauges and displacement transducers, the accelerometers were not installed permanently on the structure, and they were only utilised during the walking and the modal tests. During the walking test, the accelerometers were placed in fixed location as shown in Figure 5.11.

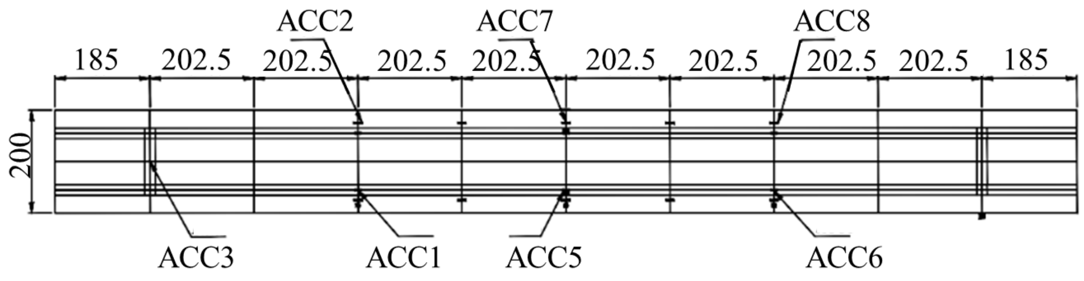


Figure 5.11 Location of accelerometers during walking test. All dimensions are in m.

On the other hand, during the modal test, the accelerometers were moved between two successive modal tests. The test was repeated until data have been collected from all 42 testing points as shown in Figure 5.12. In total, there were six sets of accelerometer installation.

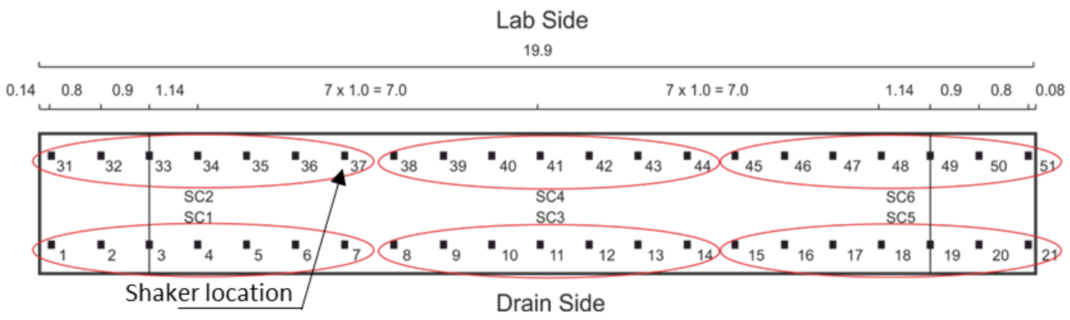


Figure 5.12 Testing points for accelerometer during modal test. All dimensions are in m.

To obtain the acceleration data NI9234 card from National Instruments [225] is employed. This card has four BNC input channels which is the output terminal of the accelerometer employed in this research. BNC cables were employed to provide the connection between the signal conditioner and NI9234 card. The measurement performed by NI9234 produces acceleration data on Voltage unit. In order to obtain the acceleration in g, the measured values have to be divided by the sensitivity of the corresponding sensors. The sensitivity of accelerometers which are implemented in Warwick Bridge measurement is provided in Table 5.10.

Table 5.10 Accelerometer sensitivity in Warwick Bridge experiment

Sensor No	Serial Number	Sensitivity (mv/g)
1	3988	1320.425
2	4005	1337.443
3	4006	1322.14
4	4007	1329.373
5	9224	1357.526
6	9447	1345.645
7	9448	1347.908
8	9351	1356.811

5.3.2 Test Description

Both static and dynamic tests were conducted in both healthy and damaged conditions. In the static test, a point load was applied in the mid-span of the bridge. This load was applied gradually in steps from 0 kg to 400 kg with an increment of 100 kg. Each increment lasted approximately 10 minutes during which the data were collected, and the data collected from the formed one dataset. In the static test, sample rate of 10 Hz was implemented for data collection. Figure 5.13 shows the displacement measured by DT2 from the static test conducted during the healthy and damaged states.

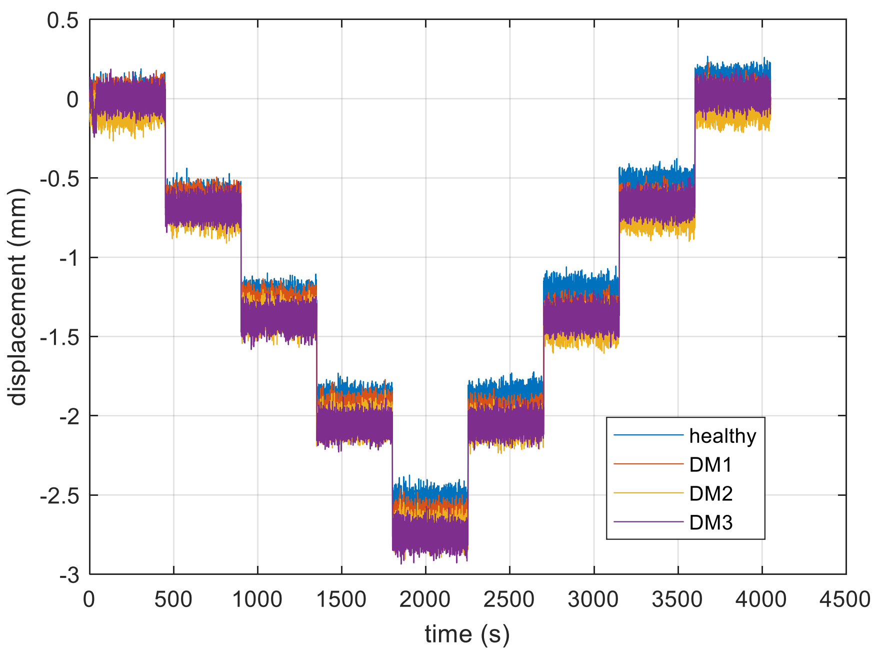


Figure 5.13 DT2 measurement during static test obtained four different bridge's states

Dynamic tests consisted of walking and modal tests. In the walking test, a test subject walked on the bridge back and forth with pacing rate of 2.5 Hz. In this experiment, the data were recorded for three minutes. The sampling rate employed in the dynamic tests was 100Hz. Hence, one dataset from the walking test consists of approximately 18000 data points. Figure 5.14 shows the data collected from three sensors including displacement transducer (DT2), strain gauge (CST8) and accelerometer (ACC7), during the walking test. Figure 5.15 further illustrates the power spectral density (PSD) from these measurements.

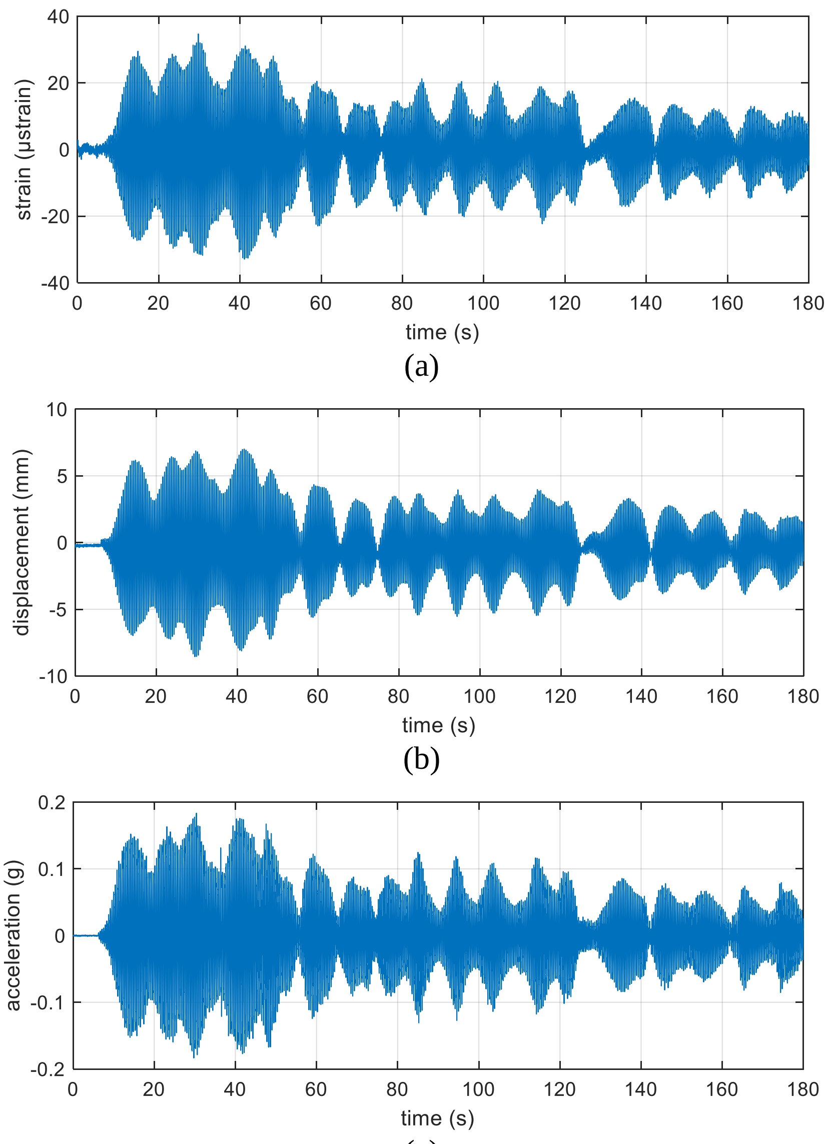
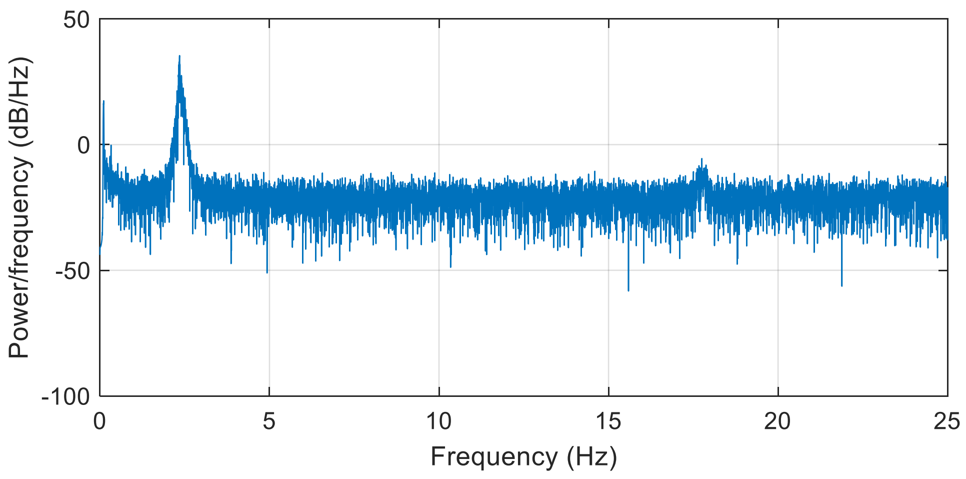
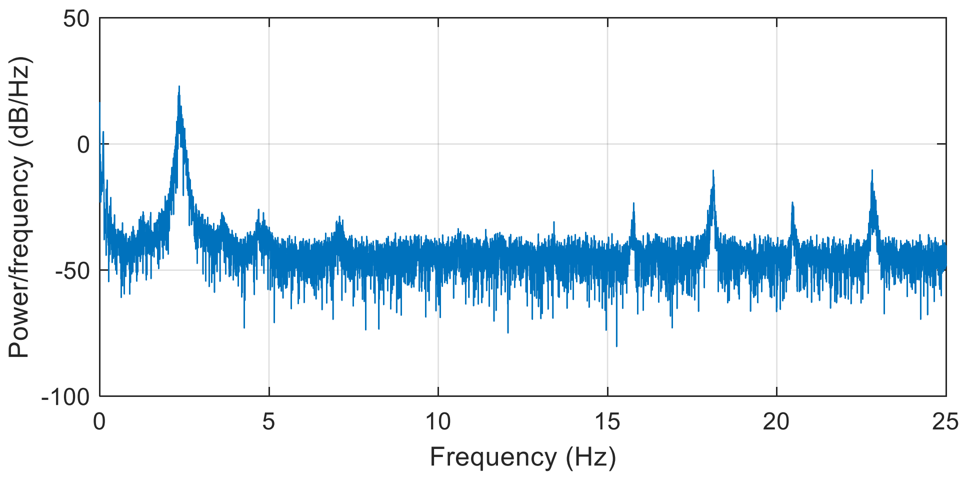


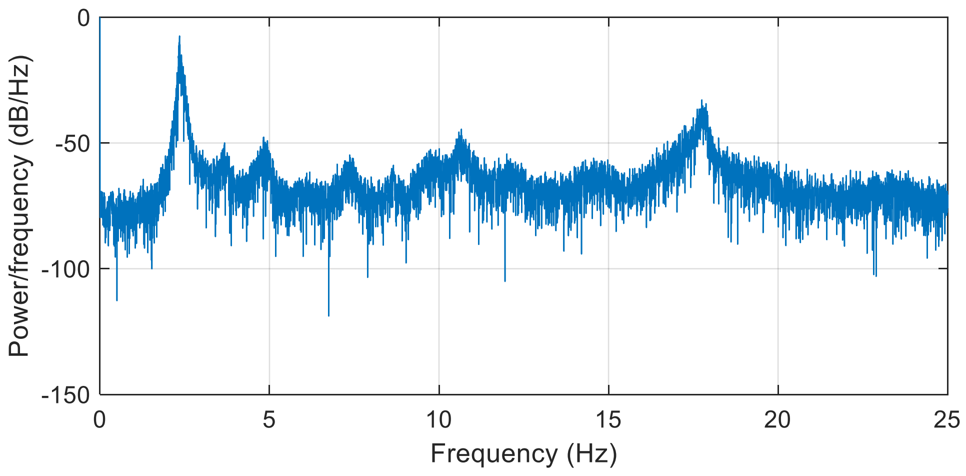
Figure 5.14 Data collected from walking test; (a) strain gauge measurement; (b) displacement transducer measurement; (c) accelerometer measurement.



(a)



(b)



(c)

Figure 5.15 Power Spectral Density from measurement data obtained during walking test; (a) PSD from strain sensor measurement; (b) PSD from displacement transducer measurement; (c) PSD from Accelerometer measurement.

The described static and dynamic tests were performed in all conditions of the bridge. Therefore, for each static and dynamic test, four datasets (healthy, DM1, DM2, and DM3) of four conditions were generated.

In addition to the walking test, modal test utilising an electrodynamic shaker (model APS400) was also performed. In this test, the shaker was employed at the quarter-span of the bridge (Figure 5.12). The shaker generated a chirp signal in a frequency band from 1-25 Hz for 64 s. The DAQ lasted 400 s, including tail period. One accelerometer was attached to the shakers' moving mass to indirectly measure the shaker force while the other seven accelerometers were deployed to measure the response of the bridge. Initially, the seven accelerometers were deployed at testing points 1-7 (Figure 5.12). After recording the data, the accelerometers were moved to the next seven points on the measurement grid (Figure 5.12). This process was repeated until the data from all the testing points has been obtained. Figure 5.16 shows the measurements from SSB, DT, and ACC collected from the modal test. The data from the modal test were analysed to enable a comparison between the proposed method and traditional frequency-based damage detection method. The PSDs from these measurements are depicted in Figure 5.17.

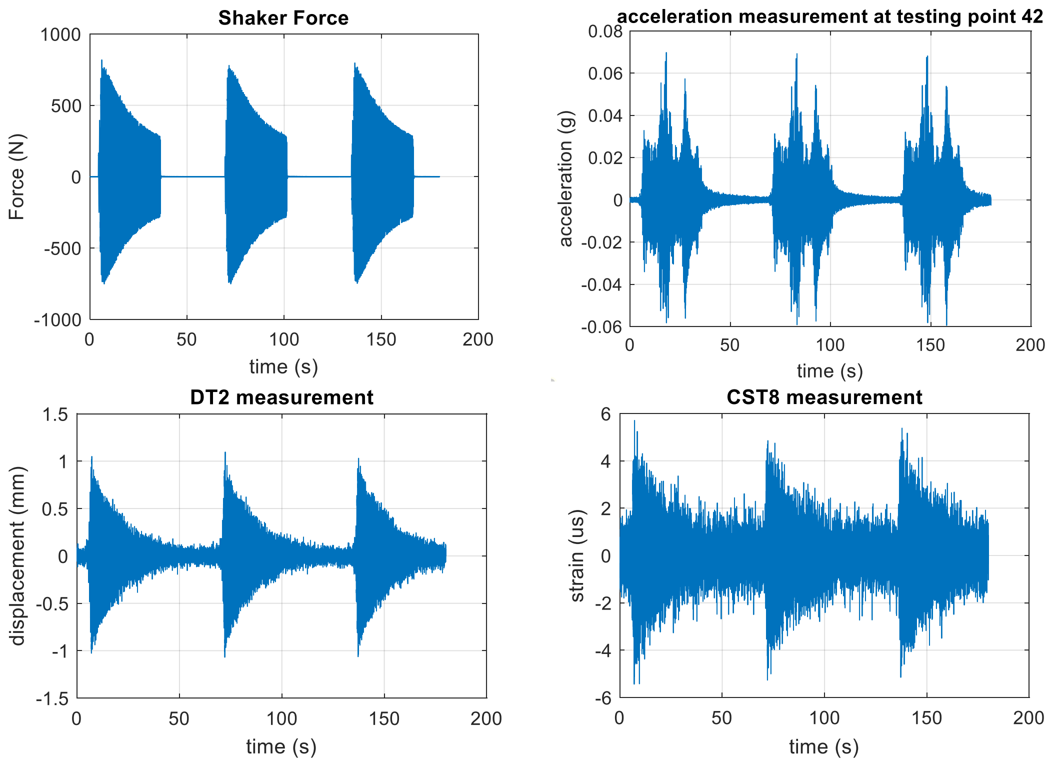


Figure 5.16 Sensor signals collected by the accelerometers, displacement transducer, and strain gauge during the modal test

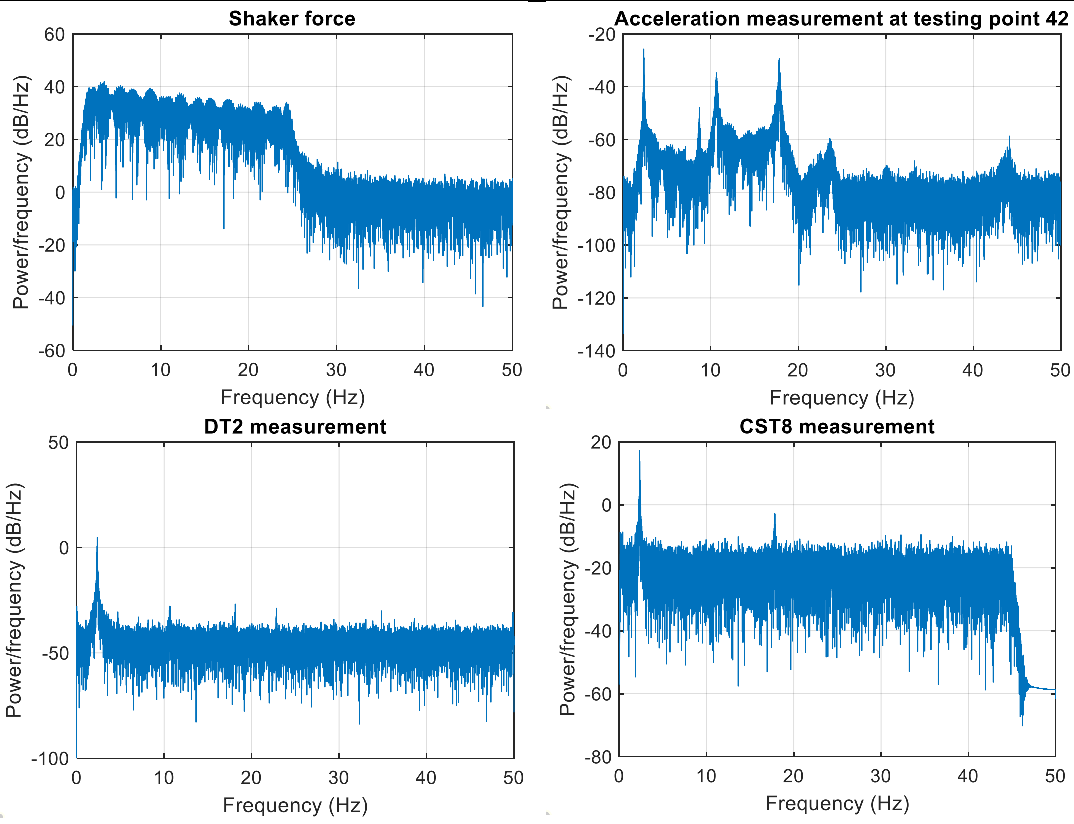


Figure 5.17 PSD from data collected in modal test

5.3.3 Damage Detection Using Frequency Response Function

In order to investigate the effectiveness of the proposed method, a damage detection method based on natural frequency is implemented as a comparison. For this purpose, damage detection method using FRF is employed. FRF describes the response of a structure to an applied force in frequency domain [226]. In the data collection stages, modal test was conducted and a dataset containing the measurement result has been generated. FRF function is then calculated using both the shaker-induced force as the input and the bridge acceleration. The first five modes in the FRF data are identified. In this method, the presence of damage might be detected by the shift in the frequency modes after damage has occurred.

Figure 5.18 shows the FRF measurements in the bridge's healthy and damaged condition from an accelerometer that is located at the mid span of the bridge. In the figure, it is shown that the damage produces insignificant shift on the FRF measurement. Figure 5.19 shows that there is no significant shift in the frequencies for the first five vibration models in the healthy and damaged states. Measurable shift in the frequency of mode 1 occurs for damaged state 3 only, when a decrease of 0.01 Hz was recorded. This decrease is by 0.43% of the initial frequency only. However, measurable change does not occur in damaged state 1 and 2. Figure 5.19 demonstrates that the maximum shift of 0.08 Hz occurs for the fifth mode at damage state 3. In practical application, this level of shift might be masked by the sensitivity of the bridge to other parameter such as temperature that makes it challenging for damage detection using frequency-based method. For example, in [227], it has been reported that between morning and night measurements, a shift in the measured natural frequency up to 0.57% was obtained. Furthermore, it was reported that the shift in the natural frequency increased to up to 2.18% between morning and noon time measurements. Thus, it might be challenging to detect the presence of damage using frequency-based method in the Warwick Bridge due the small frequency shift.

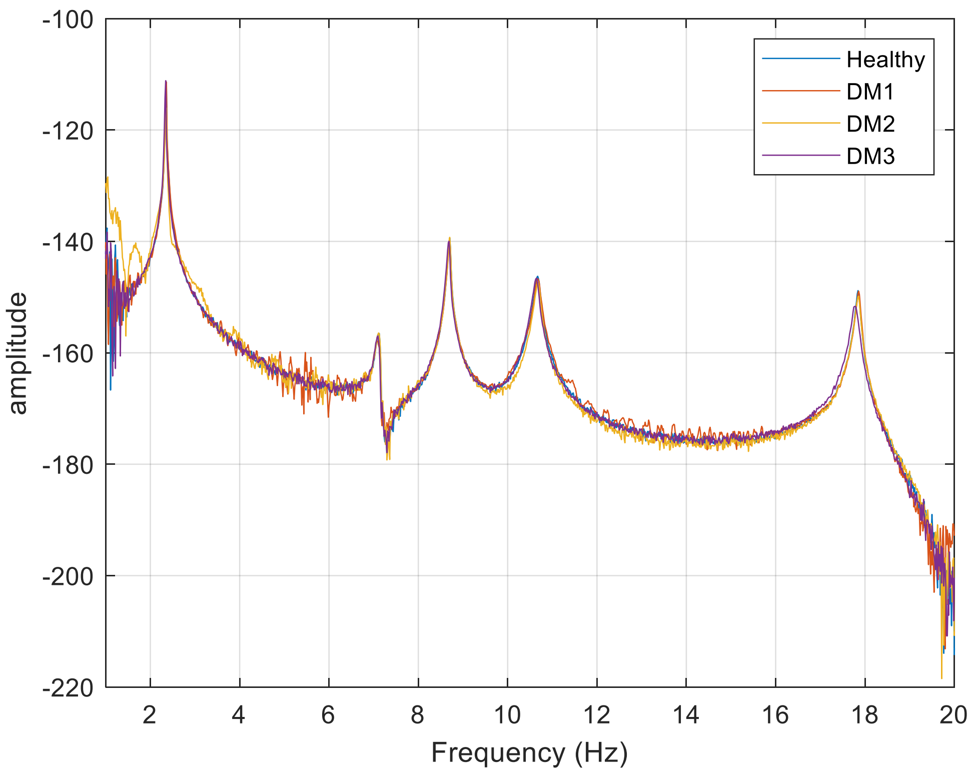


Figure 5.18 FRF measurement calculated on healthy and damaged states

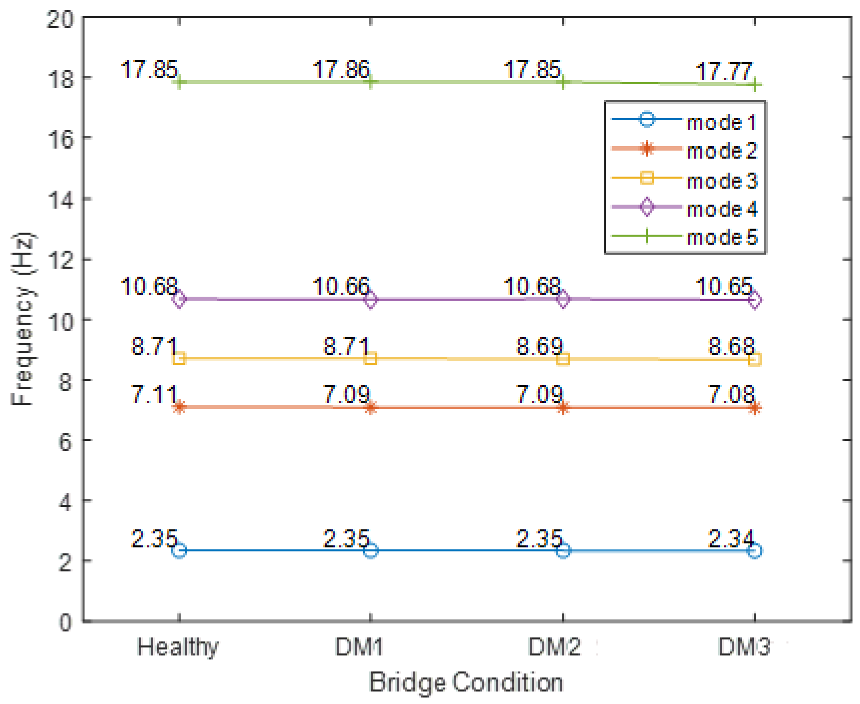


Figure 5.19 Comparison of natural frequencies of five vibration modes measured in healthy and damaged states

5.3.4 Damage Detection Using CorCNN

5.3.4.1 Data Processing for CorCNN

In this research, the data collected during walking test are employed. Initially, data processing using a low-pass filter, with 50 Hz cut-off frequency, is conducted to remove high-frequency noise. For this purpose, a Butterworth filter was employed. Linear trend from the data was removed using a predefined function "detrend" on MATLAB.

In the second step, a combination of inputs and output is selected. The inputs of the prediction model contain measurements obtained from the bridge. These inputs are employed to produce an output that is an estimation of sensor measurement at other location. In this research, this step is performed manually. As reported in [98], [99], [228], the sensors installed near the damage location are more likely to be more affected than the sensors further away from the damage. SSB5 that is located near the damage location is chosen as the output while DT3, CST10, and SSB6 are selected as the input of the prediction model. In this case, we train a prediction model that estimates the value of SSB5 by using DT3, CST10, and SSB6 data. After the sensor selection process, time histories of DT3, CST10, SSB6, and SSB5 from healthy and damaged state are stored.

Figure 5.20 illustrates the formatting of input and output employed. In this research, each data frame contains 500 measurement points. To obtain large number of samples, data augmentation is performed by sliding the fixed frame with a size of one measurement point.

The healthy state dataset is further split into training, testing, and validation sets. To avoid any overlap between training sets and testing sets, the testing data are selected from the last 20% of whole time series signals of the healthy dataset while the first 80% are employed as the training data. In addition, the last 20% of data in the training data is further divided as the validation data. The residuals obtained from the discrepancies between the actual and prediction value on the validation dataset are employed as the benchmark of the bridge's undamaged state. From this process, 8575 data points for training set, 3175 data points for validation set, and 4750 data points for testing set have been generated. In addition, each damaged state dataset consists of 17500 data frames.

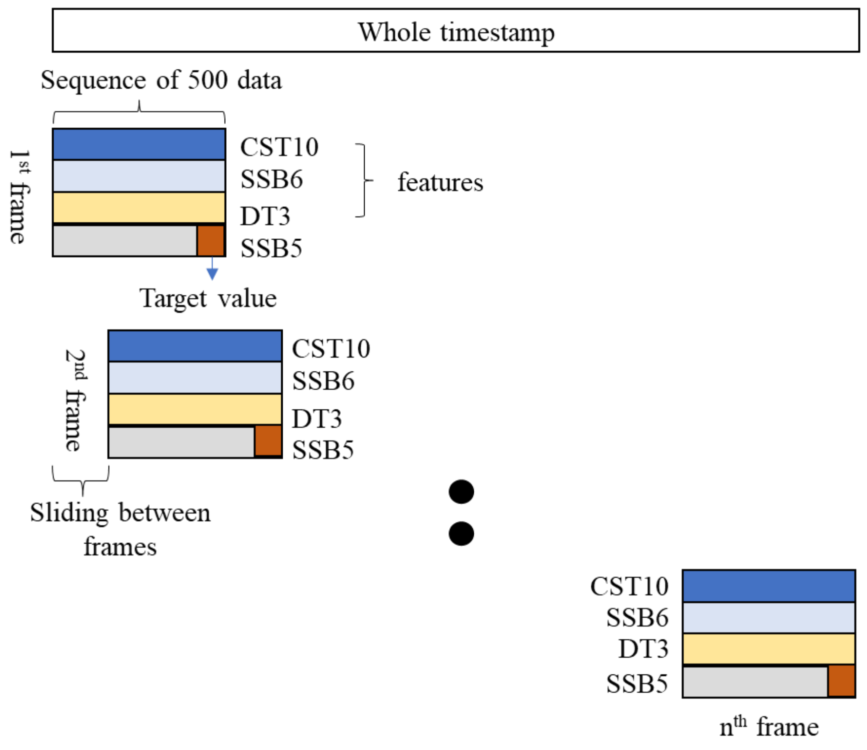


Figure 5.20 Production of data frames from time series measurement obtained from Warwick Bridge experiment for CorCNN

5.3.4.2 Optimisation for CNN Architecture

From the previous process, the healthy dataset has been divided into training, validation, and testing datasets. In this stage, supervised learning is conducted using training data to train a CNN-based prediction model. The basic architecture adopted in this research consists of four convolutional layers in the feature extraction stage and a fully connected layer in the classification stage. Each of the first three convolutional layers is followed by a pooling layer and the last convolutional layer is directly connected to the fully connected layer. The filter size for the first, second, third, and fourth convolutional layers is set to 32, 24, 12, and 4. This selection of filter size is influenced by popular CNN architectures such as AlexNet and GooogleNet where the filter size becomes smaller as the network goes deeper.

In addition, to obtain the hyperparameters for the prediction model, a grid search using varying combination of hyperparameters is performed. The hyperparameters that are optimised include the number of filters in each CNN layer, the number of fully connected layer, and the optimiser function used in the training process. Table 5.11 describes the hyperparameters employed in the grid search. In total, there are a total of 64 combinations of hyperparameters that are utilised in the grid search. In this process, prediction models with varying combination of hyperparameters are trained using the training data. Training is performed in 500 iterations using mini batch size of 32. Early stopping is employed by monitoring the validation loss in each iteration. The early setting patience is set to be 30. Then hyperparameter optimisation is conducted by selecting a prediction model that produces the lowest prediction error on validation data.

Table 5.11 Detail of hyperparameters for optimisation of CNN architecture

Hyperparameter	**Value**
Number of filters in 1st CNN layer	16 or 48
Number of filters in 2nd CNN layer	16 or 48
Number of filters in 3rd CNN layer	16 or 48
Number of filters in 4th CNN layer	16 or 48
Number of nodes in fully connected layer	50 or 100
Optimisation Function	Adam or SGD

After optimised model has been obtained, the model is employed on the validation set. Then the regression residuals from the validation set are utilised to calculate the threshold for anomaly detection by using equation (5-5).

5.3.5 Damage Detection Using Supervised CNN

5.3.5.1 Data labelling and processing

This research employs datasets collected from walking and modal tests. Initially, all sensor signals are filtered using a 50 Hz low pass filter to remove high-frequency noise. In addition, both measurement offset and Linear trend from the data were removed using a predefined function “detrend” on MATLAB. The next step is input selection. In this research, six strain gauges (SSB4, SSB5, SSB6, CST6, CST8, and CST10) and three displacement transducers (DT1, DT2, and DT3) were employed as the input of the prediction model. The accelerometers were excluded due to the configuration of the sensors on the modal test where these sensors were moved in six configurations during the test. By using these signals, the model aims to predict either the presence of damage or the severity of damage.

In order to increase the number of data, data augmentation is performed by having an overlap of s second of measurement between one frame and the next frame. This augmentation step keeps the nature of information from the data since it does not involve any modification of the original data. Figure 5.21 shows the data frames employed in the research.

In this research, sensor signals are divided into data frames each having windows length of 500 data points that corresponds to 5 s measurement. Data augmentation is performed by having an overlap of 0.1 s between one frame to the next frame. The augmentation is performed in order to generate more data since the large amount of data can be beneficial for deep learning architecture. All data that are collected during the bridge’s initial state are labelled as healthy while the data that are produced from the first, second, and third damage scenario of the experiment are labelled as DM1, DM2, and DM3 respectively. From this step, four datasets labelled as healthy, DM1, DM2, and DM3 each having 35400 data frames are generated.

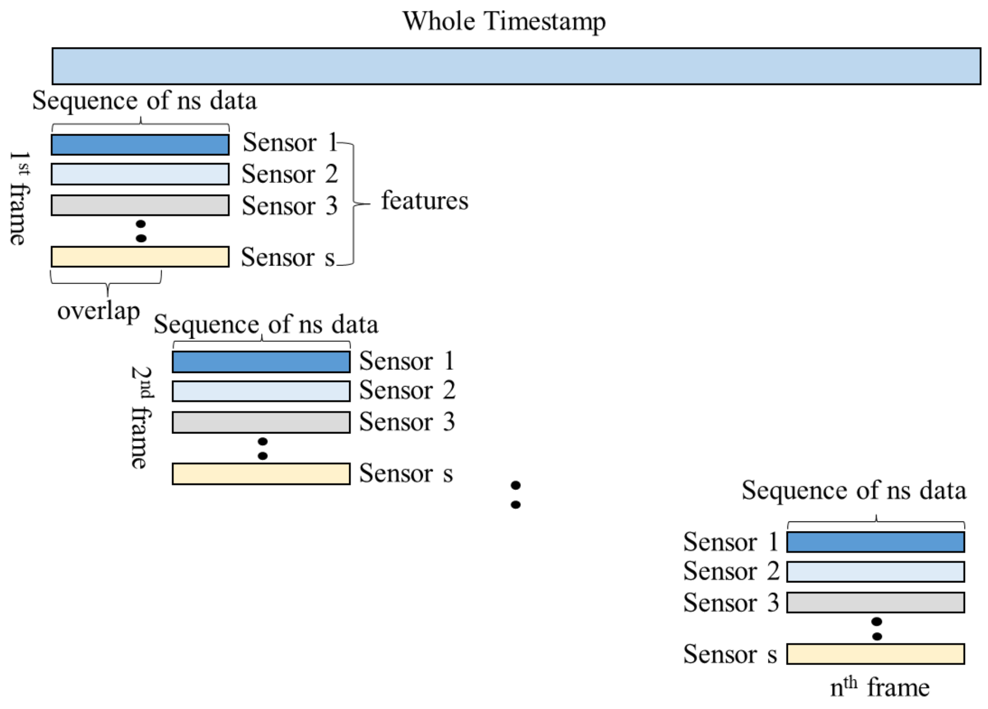


Figure 5.21 Data frames production from time series measurement

In order to observe the performance of the prediction models in detecting damage severity, several data labelling configurations such as two-label, three-label, and four label classifications, are adopted. These configurations are presented in Table 5.12. As it can be seen from the table, the two-label classification is only aimed to predict the presence of damage. For this purpose, only the healthy and DM3 datasets are employed thus the prediction models are trained to predict whether the bridge is in healthy or damaged condition.

Table 5.12 Description of data labelling performed on Warwick Bridge dataset collected from modal test

Classification type	Datasets employed	Number of samples		
		training	validation	testing
Two-label classification	Healthy, DM3	24780	3538	7082
Three-label classification	Healthy, DM2, DM3	37170	5307	10623
Four-label classification	Healthy, DM1, DM2, DM3	49560	7076	14164

On the other hand, in addition to detect the presence of damage, both the three-label and four-label classifications are also aimed to predict the damage severity. In the three-label classification, the DM2 dataset is also employed along with the healthy and DM3 datasets. In this case, not only are the prediction models trained to detect the presence of damage, but also to discriminate between the data obtained from the second and third damage scenarios. Finally, in the four-label classification, all datasets are employed.

The last step in the pre-processing step is dividing the datasets into training, testing, and validation sets. In this research, the data are split into 70% training, 10% validation, and 20% testing sets. The number of frames in each set for all classification scenarios can be seen in Table 5.12.

5.3.5.2 Parametric Study on Network Depth and Activation Function

In this research, parametric study is performed in order to investigate the impact of two hyperparameters, the activation function and the network depth, to the performance of prediction models for the investigated problem. This is performed by varying the activation function employed in each convolutional layer as well as utilising various CNN architectures as illustrated in in. In observing the impact of the activation function, four activation functions such as LReLU, ReLU, sigmoid, and tanh are implemented. These activation functions are used both in the convolution and fully connected layers as seen in Figure 5.22. To observe the impact of a certain activation function to the prediction model performance, the activation function is applied in these layers. In this case, when investigating the ReLU activation function, ReLU is employed in all layers in the architecture as the activation function.

On the other hand, parametric study on the network depth is performed by altering the architecture of the CNN models. For this purpose, five configurations are adopted: architectures with 1, 2, 3, 4, and 5 convolutional layers. The detail of architectures employed in this study can be seen in Figure 5.22. In the figure, ‘Conv’, ‘Pool’, ‘Drop’, and ‘FC’ represent a convolution layer, a pooling layer, a dropout layer and a fully connected layer respectively.

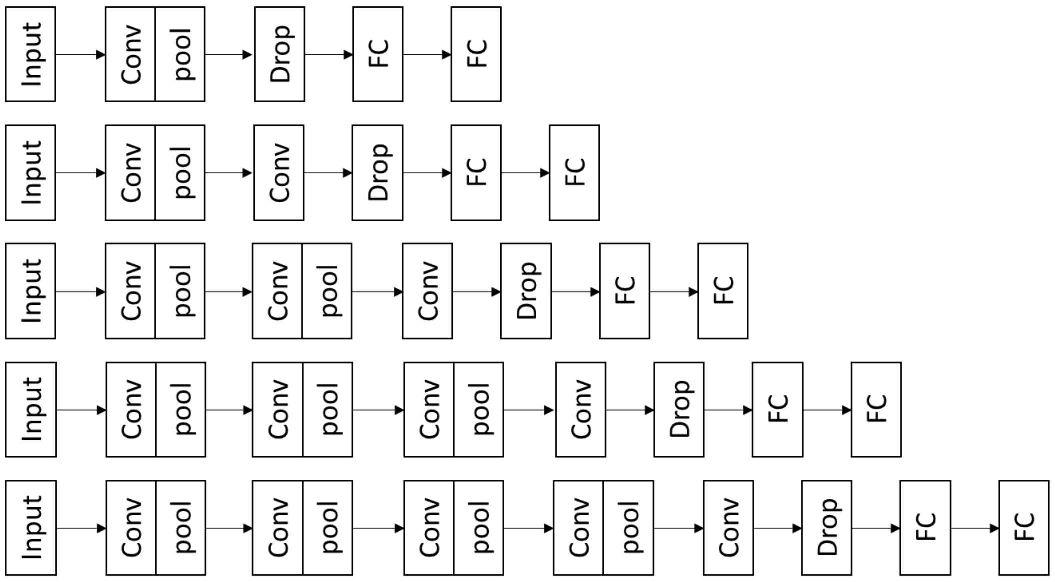


Figure 5.22 Architectures adopted for parametric study on network depth. ‘Conv’, ‘Pool’, ‘Drop’, and ‘FC’ represent convolution, pooling, dropout, and fully-connected layers, respectively.

As it is illustrated in Figure 5.22, the architecture consists of feature extractor and classifier. The feature extractor part of the prediction models consists of combination between convolution and pooling layers (denoted as Conv and Pool). The convolutional layers employed in this research consist of 32 filters each having filter size of 16. As it can be seen in the figure, each convolutional layer is followed by a pooling layer. All of the pooling layers perform max-pooling operation with a filter size of 2. Between the convolutional layers and poling layers, batch normalisation is performed.

To overcome overfitting, dropout is applied in this study. As mentioned in section 4.3.3, there is no specific rule in where to insert the dropout layer due to its nature as a stochastic method. Each of the neural network architecture employed in this study adopts a single dropout layer with dropout probability of 0.5 between the feature extraction and the classifier stage as shown in the figure.

As it is depicted in Figure 5.22, The classifier utilises two fully connected layers (denoted as FC). The first fully connected layer has 100 nodes and this layer processes all features that have been extracted from the feature extraction stage. The activation function applied in this layer is similar to the activation function in the previous convolutional layers. On the other hand, the second fully connected layer serves as the output layer of the architecture. The number of nodes in the output layer corresponds to the number of class labels. Unlike, the previous fully connected layer, this layer employs softmax activation.

5.3.5.3 Neural network training and testing

Due to the stochastic nature of deep learning methods, for every combination of activation function and network depth, five prediction models are trained using training and validation sets. Then, the testing data are employed to evaluate all trained models. This step generates prediction accuracy of the models given new data. Finally, the average accuracy from these models is recorded.

5.4 Results and Discussion

5.4.1 Evaluation on CorCNN method

The configuration of hyperparameters for the CNN architecture that produces the lowest prediction error on validation dataset has been obtained from the optimisation process. The hyperparameters (Table 5.13) have been utilised in the CNN architecture and the training process is conducted using training data. As it has been mentioned in the previous section, the model is trained to predict a single value of SSB5 by using DT3, CST10, and SSB6 data frames as the input. The trained model is then tested using validation data and the discrepancies between the actual and prediction values are used as a benchmark for the bridge's healthy state.

Table 5.13 Optimised hyperparameters combination for CNN architecture obtained by using the grid search method

Hyperparameter	Value
Number of filters in 1st CNN layer	16
Number of filters in 2nd CNN layer	48
Number of filters in 3rd CNN layer	48
Number of filters in 4th CNN layer	48
Number of nodes in fully connected layer	100
Optimisation Function	Adam

The trained prediction model is employed to create prediction for both healthy state and damaged state datasets and the residuals are recorded. In this case, testing data are used as new healthy data that has not been seen by the model. Figure 5.23 and Figure 5.24 present the comparison of the prediction and actual values when using both healthy and damaged state datasets on the trained prediction model.

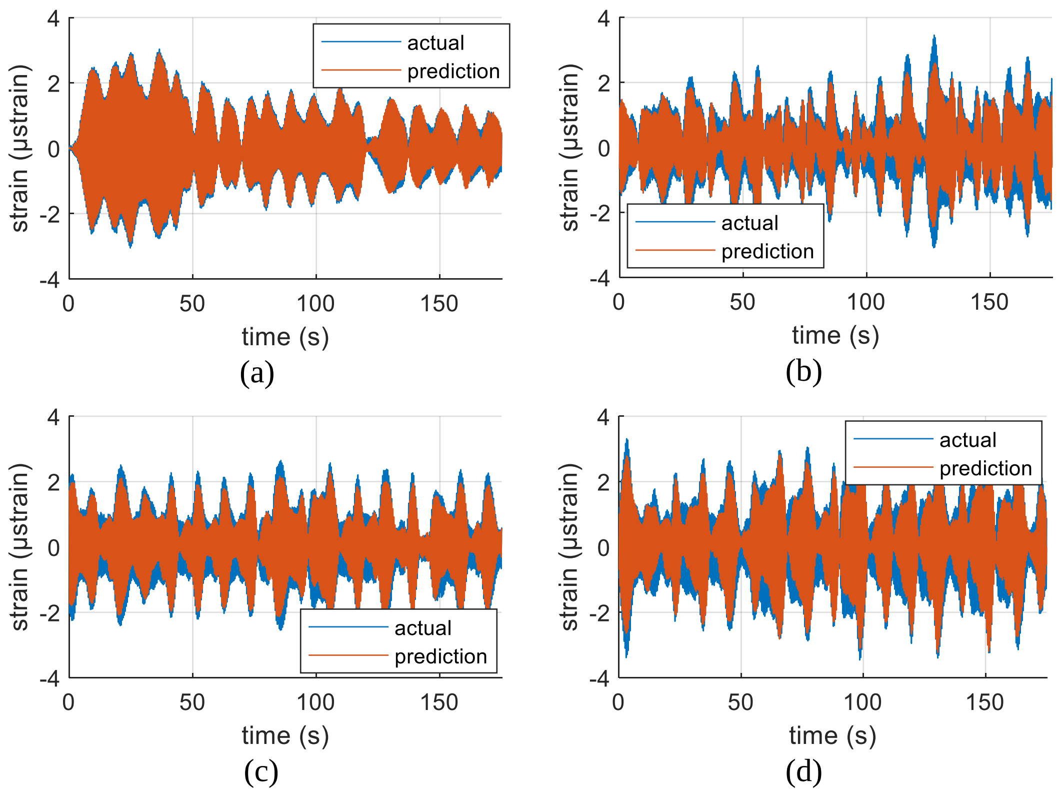


Figure 5.23 Comparison of predicted and actual values for (a) healthy, (b) DM1, (c) DM2, and (d) DM3 datasets

The model successfully reproduces the time-domain trend for SSB5 data in both healthy and damaged states as shown in Figure 5.23. In addition, as shown in Figure 5.24, the proposed CorCNN manages to produce lower error on the healthy dataset compared to the error in the damaged states. In the figure, when damage occurs, the differences between the actual and prediction values become compared to the differences produced in the healthy dataset. To further compare between the residuals produced in the healthy and damage states, the PDF and CDF of the residuals are generated as shown in Figure 5.25.

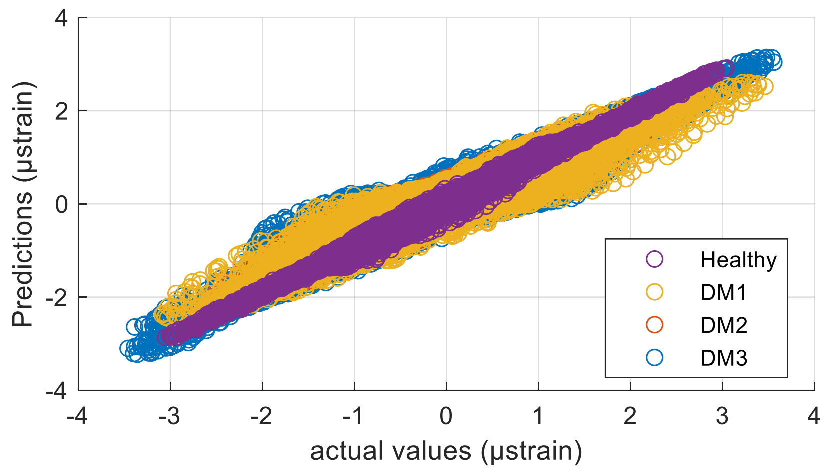
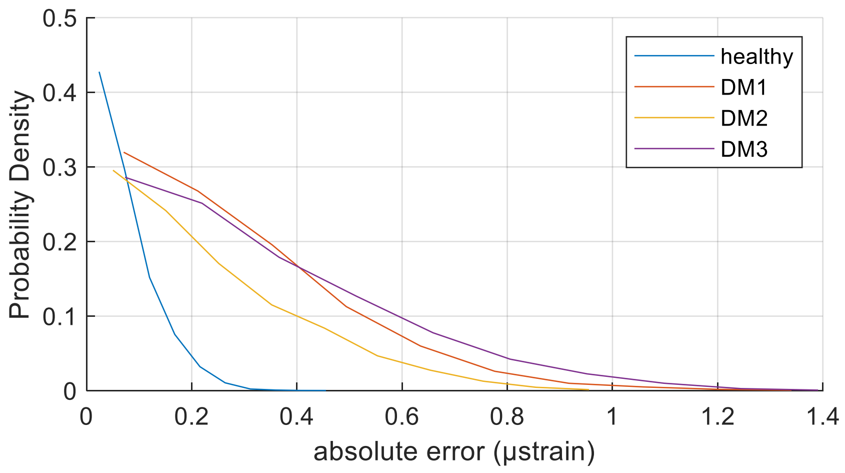
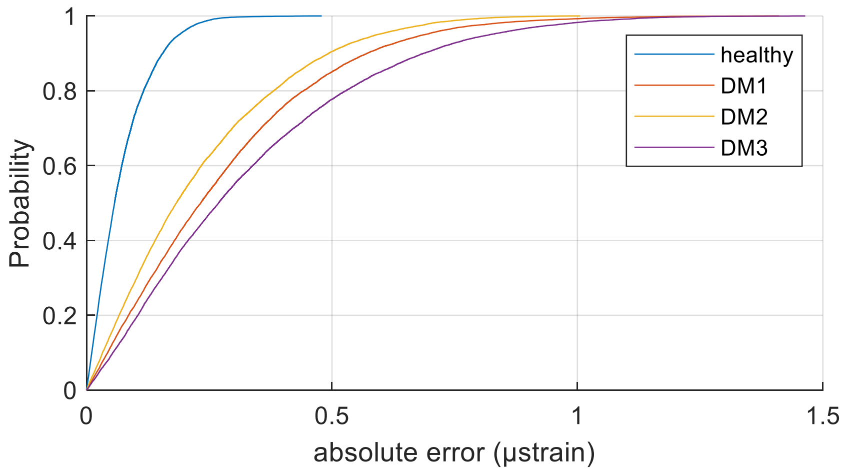


Figure 5.24 Scattered plot of the predicted and actual values from four bridge conditions



(a)



(b)

Figure 5.25 PDF (a) and CDF (b) from the residuals on healthy and damaged state datasets

As depicted in Figure 5.25(a), the distribution of residuals varies depending on the bridge condition. In the figure, it can be seen that the residuals from the healthy dataset are lower than the residuals from the damaged state datasets. In addition, the healthy dataset also produces lower maximum value of residual compared to those calculated from the damaged state datasets. Furthermore, by analysing the CDF of the residuals, it is found that the residuals increase when damage occurs. For example, as shown in Figure 5.25(b), 50% of residuals created in the healthy dataset are lower than 0.057 µstrain. On the other hand, only 14%, 17%, and 11% of the residuals from DM1, DM2, and DM3 datasets respectively, are lower than 0.057 µstrain. Moreover, as illustrated in the figure, only 5% of residuals from the healthy dataset are higher than 0.19 µstrain. However, more than 50% of the residuals produced from each damage state dataset are higher than 0.19 µstrain. Furthermore, the CDF of the residuals from the healthy dataset has the steepest slope among all thus it indicates higher prediction confidence and less variation compared to other distributions.

To further analyse the presence of damage in the bridge, the residuals from the model for the baseline, healthy state, and damaged states are calculated and shown in Figure 5.26. In addition, a threshold level for anomaly detection is calculated from the residuals of the validation set using equation (5-5).

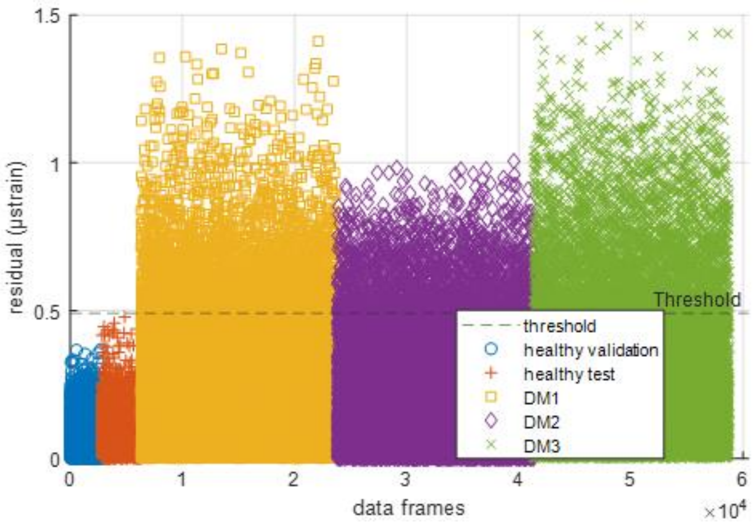


Figure 5.26 Discrepancies between the actual measurements and regression outputs generated on all sets using CorCNN

Figure 5.26 suggests that the residuals produced when the prediction model is employed on damaged state datasets is higher than the residuals produced for the baseline. In addition, some residuals produced from damaged state datasets are higher than the threshold value which indicate anomalies of the data that further translate to the presence of damage. In contrast, no residual on the testing set is exceeding the threshold level as it is shown in Figure 5.26. In this case, the model does not produce false positive where healthy data are falsely detected as anomalies.

Table 5.14 shows the number of data frames producing residuals higher than the threshold. From the table, it can be seen that the proposed method successfully detects the presence of damage in all damaged scenarios. The detection on the first, second, and third damage scenarios are 15.54%, 10.13%, and 23,01% respectively. As these numbers are higher than the confidence level employed when defining the threshold level, it can be concluded that damage was present in the structure in relation to the analysed datasets. On the other hand, no data frames produce anomaly in the healthy state datasets.

Table 5.14 Detection on all datasets using CorCNN

Condition	**Total Frames**	**Detection**	
		Total Number	**Rate (%)**
Healthy	3500	0	0
DM1	17500	2719	15.54
DM2	17500	1773	10.13
DM3	17500	4027	23.01

Figure 5.26 also illustrates that the proposed method is unable to detect the severity of the damage. From the figure, it can be seen that there is an increase in the calculated residuals as the bridge experiences an increase in the severity of damage from damage level 2 to damage level 3. However, the number of residuals exceeding the threshold in damage level 2 that represents more severed condition than the damage level 1 is lower than the number of exceeding residuals obtained in damage level 1. Table 5.14 confirms this as the number of detections drops from 15.54% for damage level 1 to 10.13% for damage level 2, even though the detections are higher than the confidence level in both cases. In this case, the severity of damage could not be detected from the

number of residuals higher exceeding the threshold value. Nevertheless, the proposed method is able to detect all three damage scenarios. as shown in Figure 5.26.

5.4.2 Comparison of CorCNN with other machine learning techniques

To further investigate the performance of the CorCNN method, the performance of other machine learning models utilising the correlation between sensor measurement is observed. For this purpose, three types of machine learning models such as linear regression, ANN, and Random Forest are employed.

The steps performed in this investigation is similar to the steps carried out for the proposed CorCNN as depicted in Figure 5.1. The dataset collected from the walking test is employed for the comparison. Initially, all sensors' signals are processed using a low pass filter with frequency cut-off of 50 Hz. Then, Matlab function 'detrend' is applied on the filtered data to remove offset and linear trend.

In the sensor selection stage, SST6, CST10, and DT3 are selected as the predictors while SSB5 is selected as the target value. This combination of sensors is equivalent to the one applied for the CorCNN. However, unlike in the implementation of CorCNN where tabular data are transformed into data frames, in this investigation this transformation is not conducted. Figure 5.27 illustrates the implementation of machine learning models utilised in this part of the research.

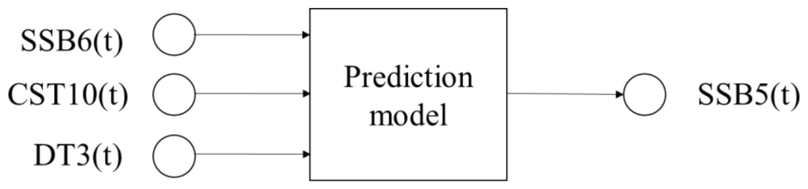


Figure 5.27 Schematic of prediction model for estimating the measurement from SSB5

Finally, the data from the sensors used for the prediction are split into training, validation, and testing sets. Similar to the splitting step performed for the CorCNN, 20% of data at the end of the time series are used as the testing data while the rest are used as the training data. Then the last 20% of data in the training set is employed as the validation set. From this process, the training, validation, and testing sets consist of 11520, 2880, and 3600 data points, respectively. In addition, each damaged set (DM1, DM2, and DM3) consists of 18000 data points.

5.4.2.1 Linear Regression Model

By using the training data, a linear regression model is trained. Then, the validation set is applied on the trained linear regression model. In this step, the prediction residuals from the validation set are generated and these residuals are employed to define the threshold for anomaly detection using equation (5-5). After the threshold has been determined, the testing and the damaged sets are applied on the linear regression model to calculate the regression error from each set. The regression residuals for the validation, testing, DM1, DM2, and DM3 sets produced from the linear regression model are illustrated in Figure 5.28.

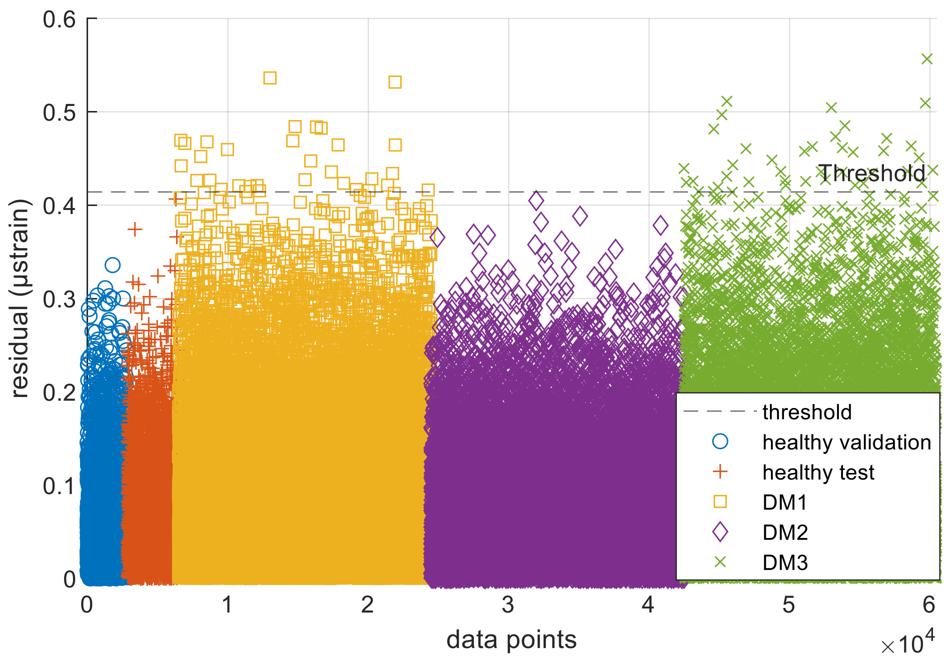


Figure 5.28 Discrepancies between the actual measurements and regression outputs generated on all sets using Linear Regression

As it can be seen in Figure 5.28, the linear regression model only detects abnormal data in both DM1 and DM3 sets. In addition, it is unable to detect the presence of damage in the second damage scenario. the model does not produce false positive since there is no residual from the testing set that exceeds the defined threshold. Table 5.15 shows the detection rates on the healthy and damaged sets that are produced by the linear regression model.

Table 5.15 Detection on all datasets obtained using linear regression model

Condition	Total data points	Detection	
		Total number	Rate (%)
Healthy	3600	0	0
DM1	18000	32	0.18
DM2	18000	0	0
DM3	18000	40	0.22

From Table 5.15, it is shown that the linear regression model is unable to detect the presence of damage in all damaged scenarios. The detection rates for DM1 and DM3 sets are only 0.18% and 0.22% respectively. This might suggest that the anomalies detected in these sets are outliers thus they do not indicate the presence of damage in the structure.

5.4.2.2 ANN

In the training process of the ANN models, grid search is carried out in order to find the best hyperparameter combination for the ANN architecture. Table 5.16 shows the hyperparameters employed for the grid search.

Table 5.16 Detail of hyperparameters for the optimisation of ANN architecture

Hyperparameters	Values
Number of nodes in the first hidden layer	10, 50, and 100
Number of nodes in the second hidden layer	0, 10, and 50
Number of nodes in the third hidden layer	0, 10, and 50
Dropout probability	0, 0.2, and 0.5

As it can be seen from Table 5.16, there are in total 81 combinations of hyperparameters utilised in the grid search. The first three hyperparameters define the network capacity. In addition, the number of nodes in the second and third hidden layer also determines the depth of the model. In this case, when 0 is selected, the corresponding hidden layer is not activated. In addition, a dropout layer is employed for each hidden layer. The optimisation is performed in order to obtain the optimised value for dropout probability.

From these combinations, 81 ANN-based prediction models are built. These prediction models are trained using the training data on 500 iterations. Early stopping technique with patience of 30 is employed in the training process thus in each iteration, validation loss is calculated by applying the validation set on the model. After training process has been finished on all models, the models are tested on the validation set to calculate the validation error. Then, out of 81 trained models, the model with the lowest validation error is then selected as the optimised model. Table 5.17 describes the optimised hyperparameters obtained from the grid search.

Table 5.17 Optimised hyperparameters for ANN architecture obtained by using the grid search method

Hyperparameters	**Optimised Values**
Number of nodes in the first hidden layer	50
Number of nodes in the second hidden layer	0
Number of nodes in the third hidden layer	0
Dropout probability	0.5

After the optimisation process, the threshold for anomaly detection is calculated. This is conducted by calculating the residuals produced from applying the validation set on the optimised ANN model. From these residuals, the threshold level is defined by using equation (5-5). Then, the model is employed to perform prediction on the testing and damaged sets. From this process, prediction residuals for each set are generated and compared with the defined threshold for anomaly detection. Figure 5.29 depicts the prediction residuals produced by ANN model.

As it is shown in Figure 5.29, the ANN model only manages to detect the anomalies in DM1 and DM3 sets. There is no false positive produced on the testing set as it can be seen in the figure. However, the ANN model detects no anomaly on the DM2 set as this set produce no residual that is higher than the threshold. In order to observe if the ANN model manages to predict the presence of damage in the DM1 or DM3 set, the detection rate is calculated. The detection rates produced by the ANN model on the validation, testing, DM1, DM2, and DM3 sets are provided in Table 5.18.

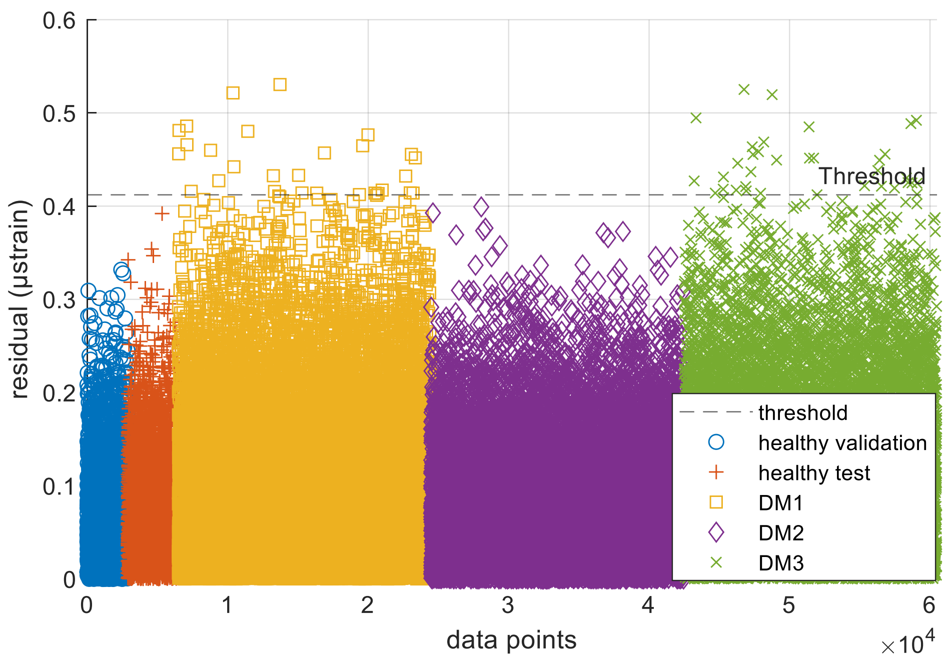


Figure 5.29 Discrepancies between the actual measurements and regression outputs generated on all sets using ANN model

Table 5.18 Detection on all datasets generated using ANN model

Condition	Total data points	Detection	
		Total number	Rate (%)
Healthy	3600	0	0
DM1	18000	28	0.16
DM2	18000	0	0
DM3	18000	35	0.19

In Table 5.18, it can be seen that the ANN model only produces detection rate of 0.16% and 0.19% on DM1 and DM3 sets respectively. These rates are lower than the detection rates generated by the CorCNN method which are 15.54% and 23.01% for DM1 and DM3 sets, respectively. Therefore, despite detecting anomalies from DM1 and DM3 sets, the ANN model might interpret them as outliers rather than the presence of damage on structure due to the low detection rates.

5.4.2.3 Random Forest

The step performed in observing the Random Forest model is similar to the step performed in the investigation on ANN model performance. Initially, hyperparameter optimisation is performed by performing grid search. The details of hyperparameters used in the grid search is described in Table 5.19.

Table 5.19 Detail of hyperparameters for the optimisation of Random Forest architecture

Hyperparameters	Values
Sample size	0.2, 0.5, and 0.7
Number of trees	200, 500 and 1000
Maximum depth	80, 90, 100, and 110

In the hyperparameter optimisation, 36 Random Forest-based prediction models are developed based on combination of hyperparameters used in the grid search. These models are trained using the training data and after the training process, all models are utilised to make predictions on the validation set. The optimised model is then picked by selecting model with the lowest validation error. Table 5.20 shows the optimised configuration of hyperparameters for Random Forest model obtained for this study.

Table 5.20 Optimised hyperparameters for Random Forest architecture obtained by using the grid search method

Hyperparameters	Values
Sample size	0.2
Number of trees	1000
Maximum depth	80

Threshold level is then calculated by applying validation set on the optimised Random Forest model. In addition, the model is employed on the testing and damaged sets to observe its capability in detecting damage. The residuals generated from applying these sets on the Random Forest model can be seen in Figure 5.30.

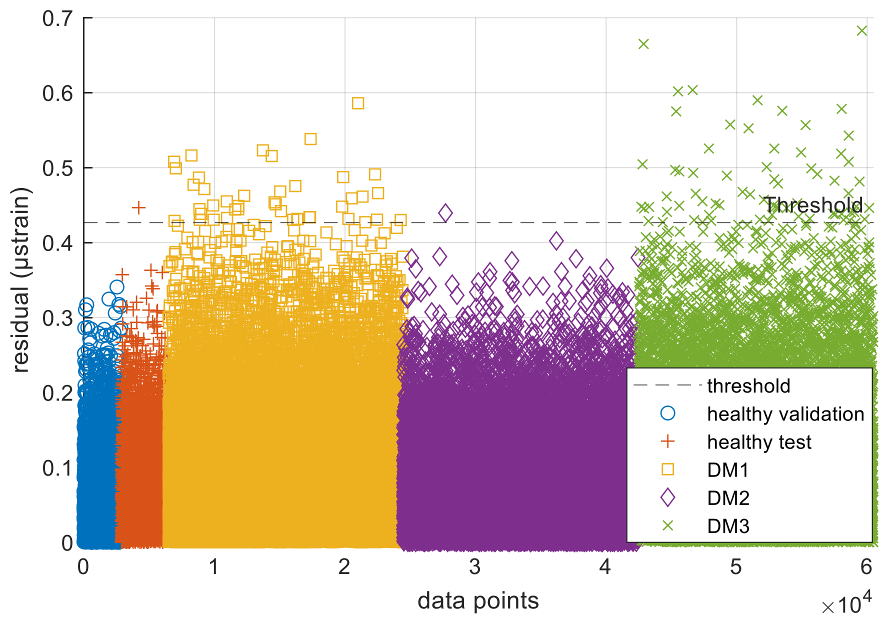


Figure 5.30 Discrepancies between the actual measurements and regression outputs generated on all sets using Random Forest model

As it is illustrated in Figure 5.30, the Random Forest model is able to detect anomalies in all damaged set. In both DM1 and DM3 sets, the model manages to detect some abnormal data as it can be seen in the figure. However, it only manages to detect one anomaly in DM2 set. Moreover, this method also mistakenly predicts a data point in the healthy state as an anomaly as depicted in the figure. In order to assess the model capability for damage detection, the detection rates of the models on the healthy and damaged sets are calculated and these rates are provided in Table 5.21.

Table 5.21 Detection on all datasets generated using Random Forest model

Condition	**Total data points**	**Detection**	
		Total number	**Rate (%)**
Healthy	3600	1	0.02
DM1	18000	42	0.23
DM2	18000	1	0.01
DM3	18000	69	0.38

From Table 5.21, it can be seen that the Random Forest model only successfully detects the presence of damage in the third damaged scenario. As shown in the table, the detection rates produced by this model on both DM1 and DM2 set are only 0.23% and 0.01% thus the anomalies detected from these sets might be confused as outliers rather than a presence of damage in the structure. In the third damaged scenario, the model produces 0.38% of detection rate. However, this rate is lower than the detection rate produced by the CorCNN method on DM3 set which is 23.01%.

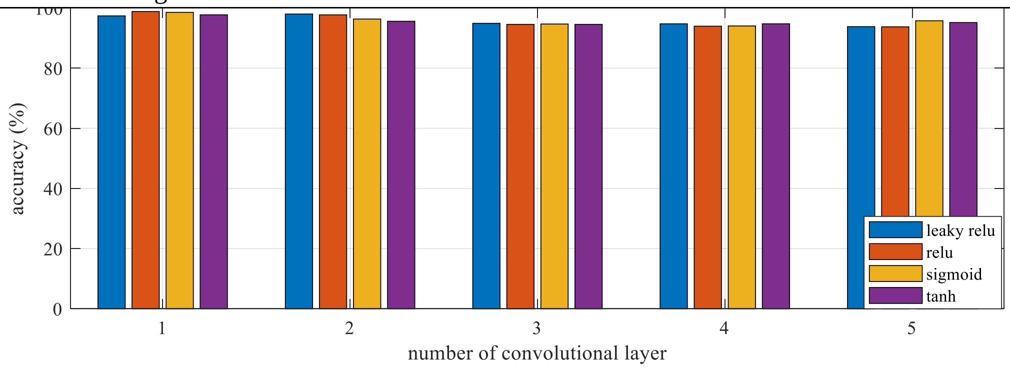
5.4.2.4 Summary of comparison between CorCNN and other methods

From the comparison, it is found that the CorCNN method outperforms the other machine learning methods in detecting the presence of damage. CorCNN employs CNN architecture which has the capability in extracting useful information from data automatically. In this case, the CNN architecture extracts and learn the patterns from data collected during the bridge healthy state in before making predictions. When damage occurs, it modifies the pattern in the data thus the CNN model produces large error when damaged state data are employed on the model. On the other hand, other machine learning methods are unable to perform automatic feature extraction. These models do not consider the change in data pattern due to the inability in obtaining sequential information from the input. As a result, these models produce lower detection rates compared to the detection of CorCNN.

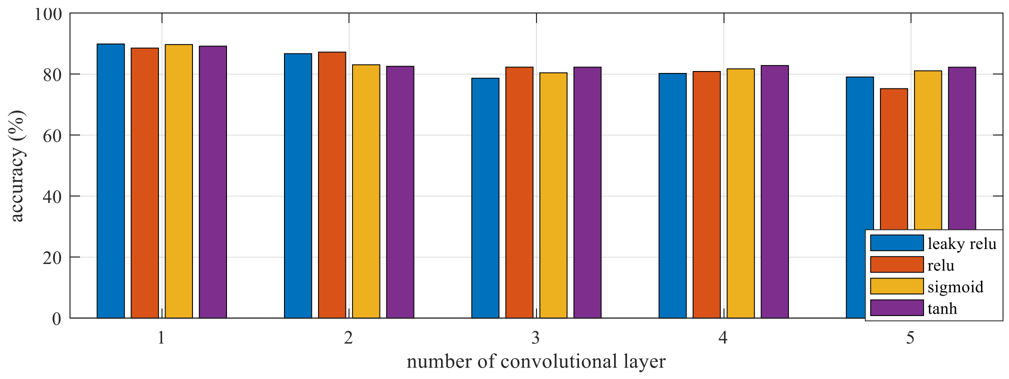
5.4.3 Evaluation on Damage Detection Using Supervised CNN

5.4.3.1 The impact of Network Depth to the Prediction Performance

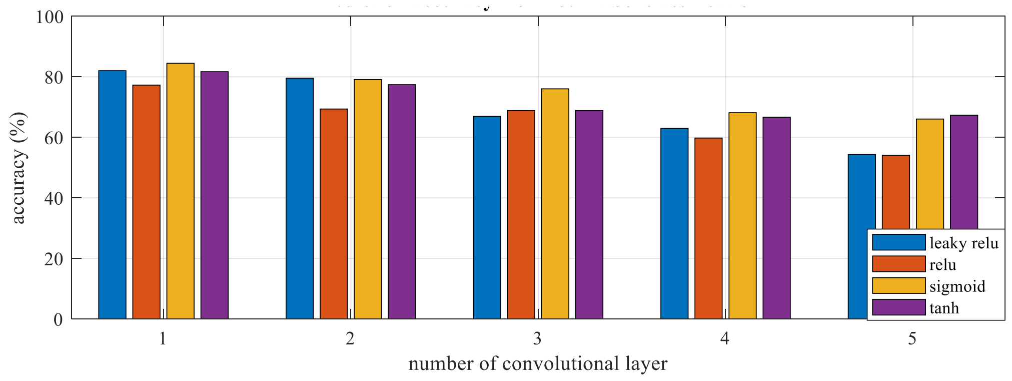
Figure 5.31 illustrates the prediction performance produced by several CNN architectures employing various activation function. From the figure, it can be seen that for each classification problem, the highest performance is achieved by the architecture that consists of one convolution layer. In two-label classification, the highest accuracy of 98.82% is achieved by the prediction models that employs one convolutional layer with ReLU activation function. Similar to this result, highest performance is also produced by architecture employing one convolution layer in both three-label and four-label classification. The highest prediction accuracy achieved in the three-label and four-label classification are 89.91% and 84.45% respectively. As it can be seen in the figure, as the architecture becomes deeper, the performance begins to drop. This trend occurs on all classification problem.



(a)



(b)



(c)

Figure 5.31 Prediction model performance for CNN-based damage detection method using supervised learning on data collected from modal test; (a) Two-label classification problem; (b) Three-label classification problem; (c) Four label classification problem.

To investigate the decrease in the performance due to the increase in the network depth, the training accuracy is investigated. For this purpose, the result obtained from four-label classification is taken as a case study. Figure 5.32 shows the training accuracy produced from all architectures in four-label classification. As it can be seen from the figure, the training accuracy increases as the architecture goes deeper. This trend is different than the trend in the testing accuracy where the increase in the neural network depth reduces the testing accuracy. This might be caused by overfitting where a prediction model learns the training data too well that it produces poor generalisation on unseen data. Overfitting tends to occur when employing complex CNN architectures, potentially decreasing the testing accuracy. The highest accuracy obtained in architecture with one convolution layer implies that for the problem under investigation, implementing a single convolution layer is sufficient.

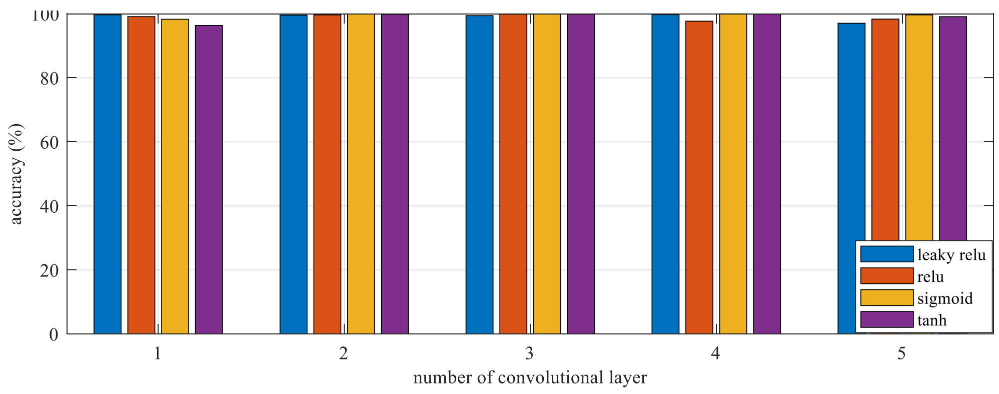


Figure 5.32 Training accuracy obtained from several CNN models adopting different number of convolution layer for four-label classification problem

To further investigate the applicability of CNN on detecting damage severity, a confusion matrix for four-label classification is derived. For this purpose, the performance of a model utilising single convolution layer that adopts LReLU activation function is observed. Figure 5.33 describes the confusion matrix generated from the model. In the figure, the columns represent the prediction made by the prediction model while the rows represent the actual class. Hence, true prediction occurs in the diagonal of the matrix and each value in the diagonal elements illustrates the total percentage of correct prediction made into the corresponding class. From the figure, it can be seen that 95% of data with healthy labels is predicted correctly with the remaining 5% of the data is wrongly predicted as DM1, DM2, and DM3. However, it can be seen that the model is unable to discriminate between data from DM2 and DM1 sets. It can be seen from the figure that 8% of data with actual label of DM1 is wrongly classified as DM2. Moreover, 20% of data labelled as DM2 is predicted as DM1. This error might be caused by the small difference between damage level 1 and damage level 2 which make it challenging to accurately differentiate between data from those conditions. Finally, the model also manages to predict correctly on 89%

data in DM3 set where 6% and 5% of the data are classified as DM2 and DM1 respectively. In addition, no data with actual label DM3 is predicted as healthy data.

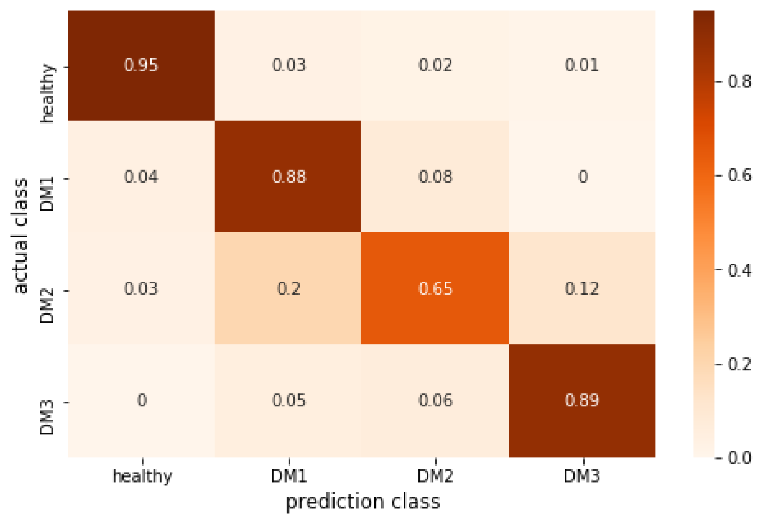


Figure 5.33 Normalised confusion matrix produced by the prediction model utilising a single convolution layer. True prediction occurs on the diagonal of the matrix.

In addition, the error distribution generated by comparing the actual class and the predicted value is also investigated. This error distribution can be seen in Figure 5.34. In the figure, the x axis represents the level of prediction error that illustrates how far the prediction deviates from the actual class, the y axis represents the actual class, and the z axis shows the number of predictions. From the figure, it can be seen that the error distribution for each class label follows normal distribution. In this case, most predictions are made to the true class while for each class, most prediction errors occur on the class adjacent to the actual class.

To further investigate the overfitting, the result from four-label classification is taken as a case study. Figure 5.35 shows the training and validation process of the prediction models adopting a single convolution layer and five convolution layers. From the figure, it can be seen that the training and validation accuracy for architecture with one convolution layer increase with similar trend in the first eight epochs. At 16th epoch, the overfitting starts to occur thus increasing the gap between the training and validation accuracy. Finally, since the model produces no increase in validation accuracy after the 43rd epoch, the early stopping mechanism is activated, and the training process is stopped after 63 epochs. On the other hand, the architecture with five convolution layers suffers from overfitting since the 5th epoch. In the figure, it can be seen that from the first four epochs, the trained model manages to achieve training accuracy higher than 90%. The training accuracy for this model keeps increasing with more iteration as it is shown in Figure 5.35. However, this trend does not apply on the validation accuracy where fluctuation occurs after four epochs.

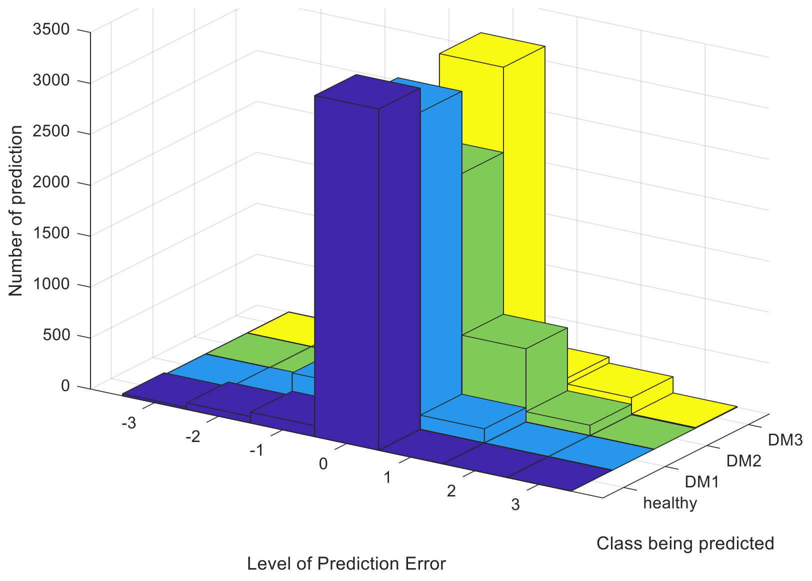


Figure 5.34 Error distribution from the comparison between the actual class and the prediction

As it is shown in Figure 5.35, the validation accuracy of 91% achieved by the model that employs one convolution layer does not reflect the testing accuracy. As it can be seen in Figure 5.31, the model only reaches 84.25% of testing accuracy. This might be caused by the lack of data used in the validation dataset since the validation set is only 10% of the whole data frames. In addition, the prediction model might also overfit the validation data where it is unable to produce good generalisation on new data. Therefore, it is a good practice to allocate both the validation and testing datasets in order to assess the real performance of prediction model on completely unseen data.

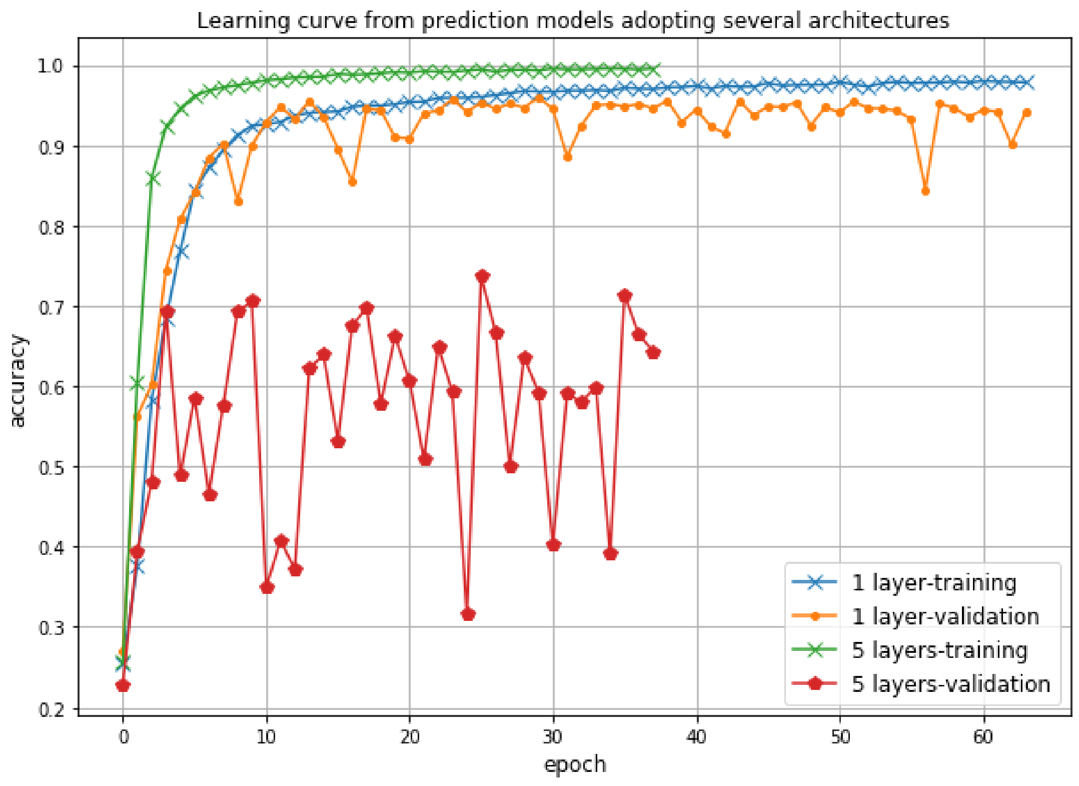


Figure 5.35 Training and validation accuracy for models adopting one and five convolution layers

5.4.3.2 The impact of Activation Function to the Prediction Performance

The other parameter that is investigated is the effectiveness of a certain activation function for the problem under study. As it is shown in Figure 5.31, there is no activation function that works best for all cases. In two-label classification, the architecture with ReLU activation function yields the highest testing accuracy. However, for three-label and four-label classification, the activation functions that produce the best performance are LReLU and sigmoid respectively. Similarly, for each classification problem (two-label, three-label, and four-label classification), there is no activation function that produces the highest prediction accuracy on all model architectures. As an example, in three-label classification, the model with one convolution layer achieves the highest performance using LReLU as the activation function. On the other hand, in three-label classification, the model adopting three convolution layers generates the highest prediction accuracy when ReLU activation function is utilised. Therefore, there is no activation function that provides a general solution for the problem in the case study.

Figure 5.36 describes the training and validation accuracy for prediction models adopting different activation functions for four-label classification. As it can be seen from the figure, the model adopting LReLU activation function produces the highest validation and testing accuracy. It achieves a training accuracy higher than 90% in the 9th epoch, faster than the other models. In this model, the training process is stopped in the 57th epoch due to the early stopping.

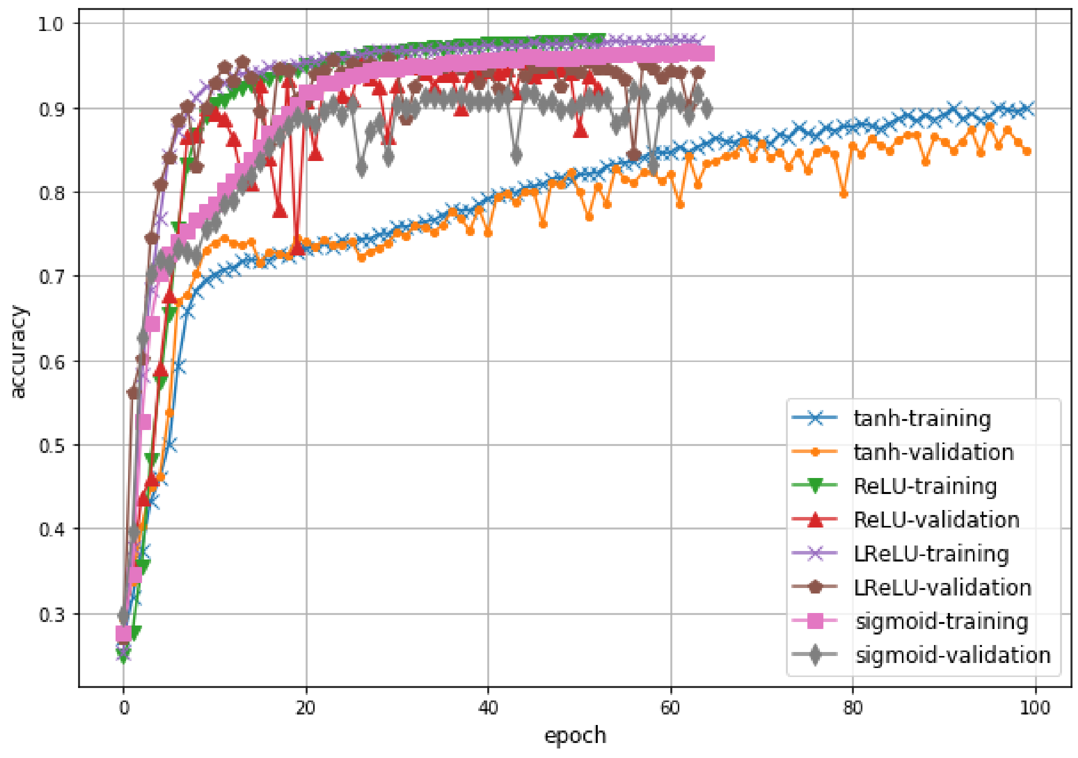


Figure 5.36 Learning curves of prediction models utilising various activation functions

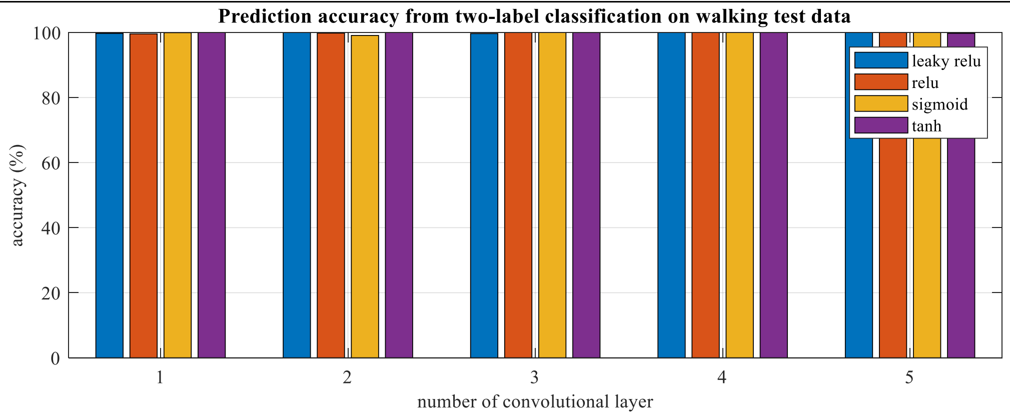
On the other hand, as shown in Figure 5.36, the prediction model that employs sigmoid activation function surprisingly produces faster training process than the model utilising tanh activation function. In the training process of the model, early stopping occurs at 60th epoch. Although it has lower validation accuracy compared to both models with ReLU and LReLU activation function, from Figure 5.31, it can be seen that the model with sigmoid activation function yields the highest accuracy on the testing set. Therefore, it provides better generalisation compared to the models that use LReLU or ReLU. Considering both the level of accuracy and the comparable training time with LReLU activation function, sigmoid activation function should also be taken into account when building CNN architecture on the investigated problem or other similar problems.

Finally, in Figure 5.36, it can be seen that model with ReLU activation function has the fastest training process. As it can be seen from the figure, the model triggers early stopping at 37th epoch, faster than the other models. On the other hand, the model utilising tanh activation function has the slowest training process. As it can be seen in the figure, the training process of model utilising tanh activation function only stops after it reaches 100th epoch, thus early stopping mechanism is not activated on the model.

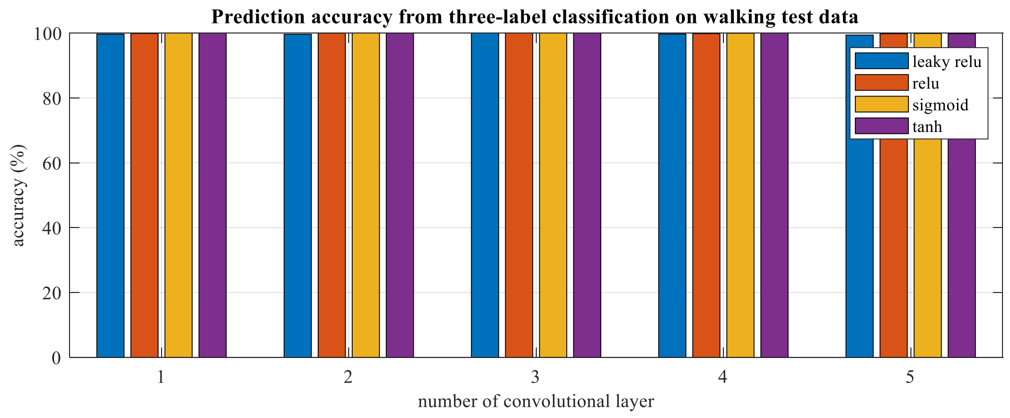
5.4.3.3 Implementation on walking test data

The method was also performed on datasets containing measurement from walking test. The performance of prediction model produced for all cases can be seen in Figure 5.37. In the figure, it can be seen that for the walking test, most of the prediction models manage to achieve accuracy higher than 99%. In addition, it is also shown that there is no relation between an increase in network depth with the prediction accuracy unlike the results obtained when using the datasets from the shaker test. However, similar to the previous result, there is no activation function that can yield the best performance in all cases.

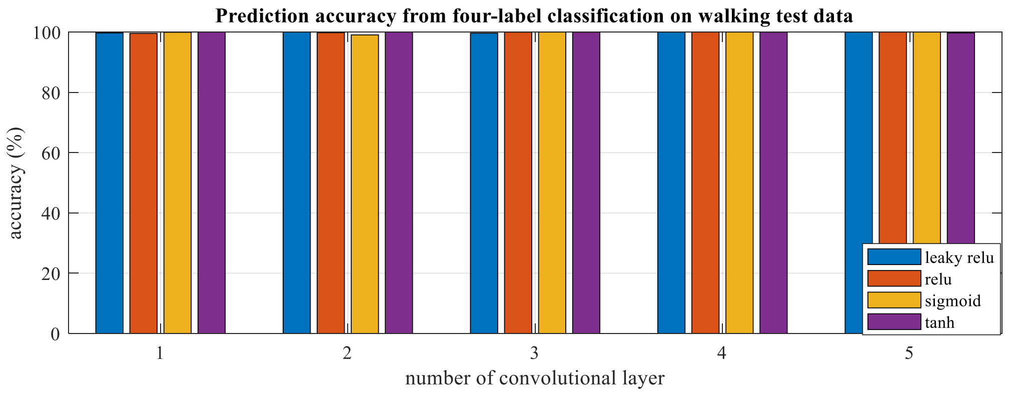
Despite the promising results from the multiclass classification approach, the approach has a significant drawback which might limit the application in real monitoring scenarios. The method requires information regarding the condition of the monitored structures. This information is needed in the labelling process so CNN models can be trained in supervised manner. In this research, this information is available thus performing supervised learning on CNN models can be carried out. However, in real monitoring scenarios, labelled data are mostly unavailable therefore damage detection is conducted to obtain the structure's condition. Obtaining the labelled data might be impractical and time consuming. Furthermore, even when we have labelled data from one bridge, it is challenging to apply a supervised model trained using the labelled data from the bridge on other bridges due to the different characteristics of bridges.



(a)



(b)



(c)

Figure 5.37 Prediction model performance for CNN-based damage detection method using supervised learning on data collected from walking test; (a) Two-label classification problem; (b) Three-label classification problem; (c) Four label classification problem.

5.5 Conclusions and summary

In this study, two damage detection approaches implementing deep learning have been presented. Firstly, CorCNN, a novel damage detection method based on the change in correlation between measurements has been validated using data obtained from a laboratory-scale bridge. The method employs 1-D Convolutional Neural Networks as a regressor. It has been shown that by exploiting the correlation between measurements on the bridge, a measurement at one point of the bridge can be estimated using sensor signals measured at other points in the bridge. In addition, damage detection approach utilising CNN models trained using labelled data has been discussed. Parametric study has been performed in order to observe the impact of hyperparameters to the damage detection performance. The conclusions from this study are as follows:

- The proposed approach, CorCNN, successfully detects damage that cannot be detected using standard vibration-based method (i.e., relying on the shift in natural frequencies).
- The method can be utilised to detect damage in all damage scenarios as it can be seen from the prediction residuals that exceeds the threshold on damaged state datasets.
- It has been shown that although the proposed method is unable to provide useful information on the severity of damage, it manages to detect the presence of damage in all scenarios.
- Based on the comparison with other machine learning techniques, it has been shown that the proposed approach manages to detect the presence of damage that can hardly be detected by the other machine learning methods.
- The parametric study performed in damage detection using multiclass CNN has shown that for the implementation of the framework on modal test dataset, models with shallow architecture outperform those with deeper architectures. This might be caused by overfitting to the training set that occurs on models adopting more than two convolution layers.
- It is important to split the dataset into training, testing, and validation datasets when implementing supervised learning. This way, the prediction model is assessed based on its performance on unseen data.

- It is found that by using the amount of data used in this study, the CNN models might achieve prediction accuracy of 98.82%, 89.91% and 84.45% for two-, three-, and four-label classification problem.
- There is no activation function that manages to produce highest performance in all cases. However, from the research it is found that tanh activation function can be removed in the hyperparameter optimisation due to the slow training process obtained from the implementation of the activation function for the investigated problem.

Chapter 6 Combined MPCA-CNN for Damage Detection on Structures

Summary

This chapter presents a damage detection method for SHM that combines MPCA and CNN. Section 6.2 describes the methodology conducted in the research. Section 6.3 explains the validation of the method using experimental data from a laboratory-scale bridge. In this section, parametric study for MPCA is firstly presented. Following the result of this investigation, explanation on parametric study on the CNN model is explained in section 6.3.4. Section 6.4 provides the result and discussion on the hyperparameter study on the CNN architecture as well as comparison between the proposed method and other machine learning techniques. Finally, Section 6.5 summarises the chapter.

6.1 Introduction

In Chapter 5, data interpretation methods implementing deep learning technique have been performed using both unsupervised and supervised learning. Despite having the capability in discriminating between data obtained in healthy and damaged state, the supervised learning method requires information on the structures' condition for data labelling which might limit the application of the method in real-life monitoring system. On the other hand, unsupervised learning is more applicable in real application since it does not require any labelling in the implementation. It is desirable to combine deep learning with other existing unsupervised method to further improve the damage detection capability.

MPCA is one of the existing methods that has been implemented for damage detection without the needs of data labelling in the application. The method has been applied successfully to enhance the distinction between features of undamaged and damaged condition. MPCA has been implemented individually and in combination with regression analysis such as RRA, MLR, SVR, and Random Forest [84]. It has been shown that combining MPCA with regression analysis might improve the performance both in detection and time delay.

In this chapter, a damage detection method that combines MPCA and CNN is proposed. This study is aimed in order to improve the existing MPCA-based damage detection methods by incorporating deep learning architectures. In this chapter, the steps executed in the study is presented. To verify the proposed method, the datasets collected from Warwick Bridge experiment are employed in the case study. Parametric study is performed to observe the impacts of some parameters for both MPCA and CNN methods. Finally, in order to assess the effectiveness of the combined MPCA-CNN method, comparison between the combined MPCA-CNN method and other combined methods is presented.

6.1.1 Summary of Novelty and Contribution

This study proposes a novel method combining MPCA and CNN to improve existing damage detection method. In addition, this work also observes for the first time the impact of parameters employed in both MPCA and CNN to the damage detection performance. Moreover, this research for the first time also evaluates the comparison between the combined MPCA-CNN and other previously reported combined methods such as combined MPCA-linear regression, MPCA-ANN, and MPCA-Random Forest.

6.2 Methodology

Figure 6.1 describes the steps conducted in the research. Timeseries $U(t)$ containing measurement of bridge responses are collected using sensor system installed on the structure. The next step is performing MPCA analysis by employing a moving window with a size of N_s and a time step of t. As the window moves through the timeseries, an individual PCA operation is performed on the data inside the window to extract the principal components. In this stage, the following procedures are conducted:

- normalisation - data inside the window is normalised using statistical normalisation.
- Covariance - covariance matrix C is derived from the normalised data.
- Eigen - eigenvalues λ_i and eigenvectors ψ_i are extracted from the covariance matrix.
- sorting - the eigenvectors are sorted based on the significance of the eigenvalues.

Among all eigenvectors, only the first eigenvector or the first principal component ψ_1 is recorded due to its largest variance. As the moving window moves, a new first principal component is generated. This operation is repeated until the moving window reaches the end of the time series. From this process, a new dataset containing timeseries of the first principal component of the measurement is generated. The whole MPCA procedures are executed on both healthy and damaged state dataset. The size of the new dataset can be calculated as follow:

$$N_w = \frac{N_o - N_s}{i} + 1 \tag{6-1}$$

where N_s is the window size, N_o is the total number of data points in the original dataset $U(t)$, and i is the time step. Although the number of data point is reduced, the number of variables is constant since the number of elements in the eigenvector is equal to the number of sensors.

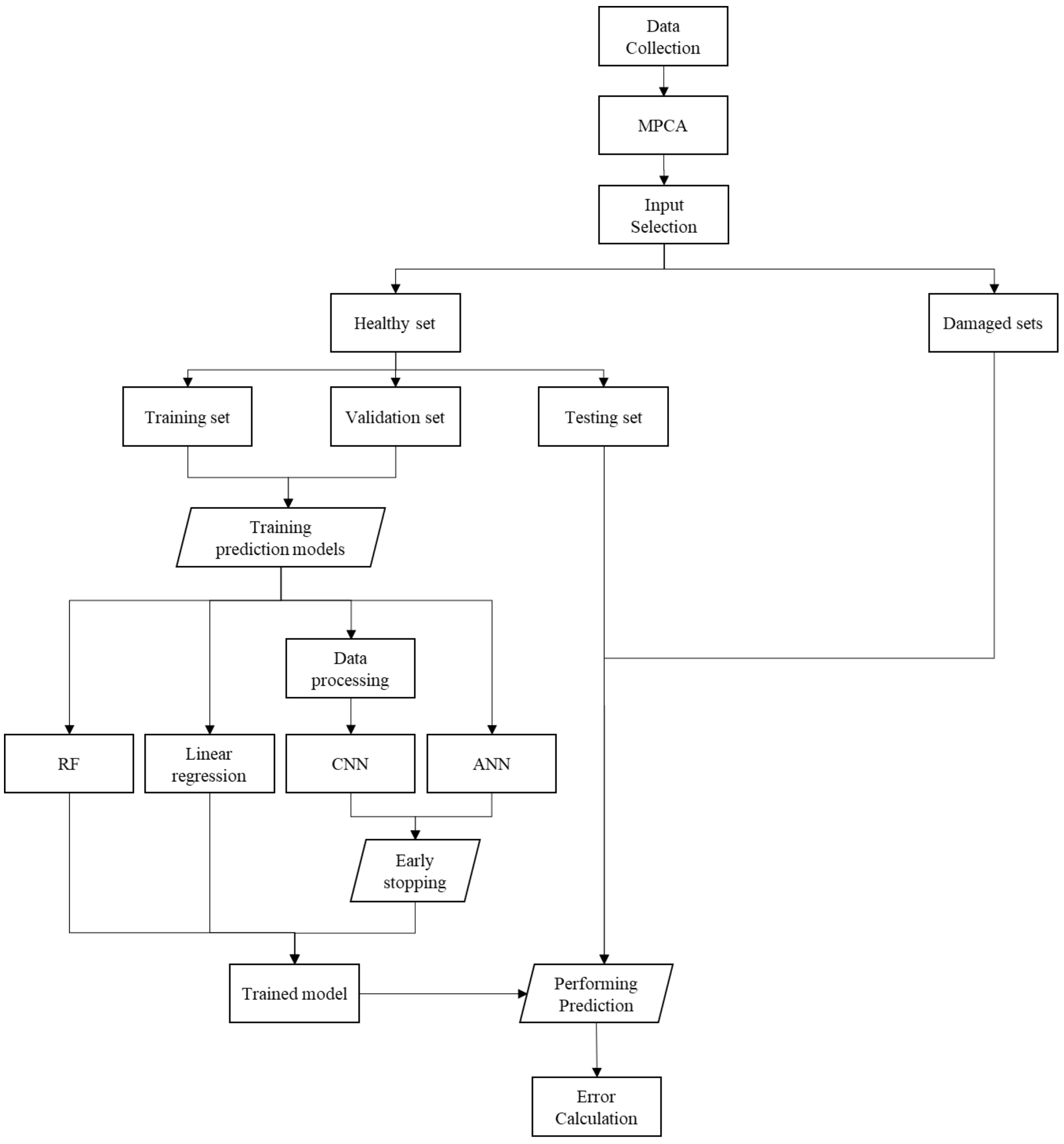


Figure 6.1 Flowchart of the combined MPCA-CNN damage detection method

It has been reported that the presence of damage might influence the structure's responses, thus changing the covariance matrix extracted from the measurement [98], [228]. The change in the covariance matrix will modify the eigenvectors and consequently, the correlation between the elements in the eigenvectors might be affected. Therefore, the sensor selection stage involves identifying pairs of sensors that have high correlation. This is executed by calculating the Pearson correlation coefficient [229] of the newly created dataset that contains the time history of the first principal component from the healthy state measurement. In this study, the correlation coefficient is calculated using:

$$r_{xy} = \frac{(N\sum x_i y_i - \sum x_i \sum y_i)}{\sqrt{Nx_i^2 - (\sum x_i)^2}\sqrt{Ny_i^2 - (\sum y_i)^2}} \tag{6-2}$$

By using the correlation matrix, all sensor pairs that have correlation coefficient higher than 0.85 are stored for anomaly detection. By using these pairs, regression functions that estimate the value of one variable using the value of its pair as the predictor can be generated. For this purpose, new datasets that only contain measurements from these pairs are obtained. The correlation coefficient might be affected by the choice of window size hence in this research, parametric study in the impact of window size is also observed. The pairs that have been identified from the healthy state dataset are also utilised on the damaged state dataset.

As it is illustrated in Figure 6.1, The healthy dataset is then further divided into training, validation, and testing sets. Both the training and validation sets are employed in the supervised training of regression models. In the training process, the training set is utilised in the iterative backpropagation process to update the weights inside the prediction model while the validation set is applied for both hyperparameter optimisation and early stopping. Early stopping is employed for regularisation purpose. In hyperparameter optimisation, a set of models with various architectures are trained using the training data. After the training is finished, the validation loss from these models is computed. The architecture that yields the lowest validation loss is then selected as the optimised model. On the other hand, the testing set is not employed in the supervised training. Instead, it serves as new healthy state data that have not been seen by the prediction model. This way, the performance of the model when given new healthy state data can be observed.

The next step performed in the research is training the prediction models using various techniques such as CNN, linear regression, Random Forest, and ANN. CNN has been known for its capability in finding spatial pattern from data. Unlike, some machine learning techniques, CNN architectures accept input in form of sequences instead of tabular data. Therefore, data processing is required in order to form data frames that can be supplied into CNN architectures. For this purpose, another sliding window with a fixed size N_d is employed on the datasets that are generated in the sensor selection stage. Consider a dataset $D(t)$ consisting of time series of two pairing elements from the first principal component ψ_1:

$$D(t) = \begin{bmatrix} \psi_{1,1}(t_1) & \psi_{1,2}(t_1) \\ \psi_{1,1}(t_2) & \psi_{1,2}(t_2) \\ \vdots & \vdots \\ \psi_{1,1}(t_N) & \psi_{1,2}(t_N) \end{bmatrix} \tag{6-3}$$

where $\psi_{1,1}$ are $\psi_{1,2}$ the first and second element, respectively. The window moves through $\psi_{1,1}$, and the data inside the window forms a data frame. On the other hand, for each data frame created by the moving window, a single data point from $\psi_{1,2}$ is stored. In this case, the CNN model employs a sequence of data point from the first element to estimate a single value of the second element. Figure 6.2 shows the construction of data frames that is implemented in this research.

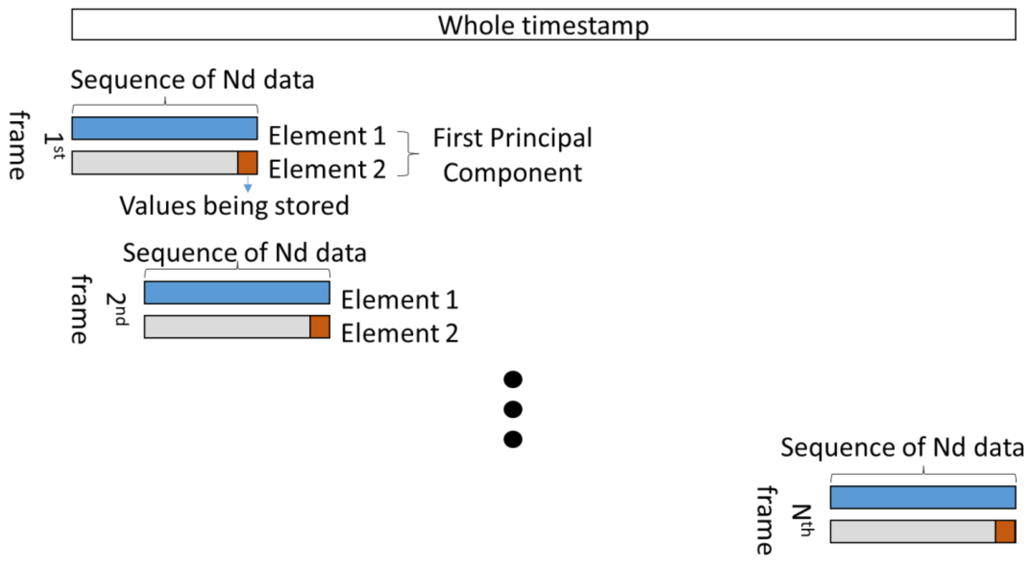


Figure 6.2 Construction of data frames using a pair of sensors for regression analysis in combined MPCA-CNN. Each data frames (presented in blue) are used as the input of CNN models to predict its corresponding target output (presented in brown).

From this process, a new dataset that contain data frames and their corresponding target output is created. This step is performed on both the healthy and damaged state datasets and these newly created sets are only employed on CNN architectures.

After the hyperparameter optimisation process, the optimised architecture for each prediction model is obtained. The validation set is then implemented in defining a threshold level for anomaly detection for each prediction model. Firstly, the validation set is employed on the optimised model. Then, the residuals that are produced from the discrepancies between the values from the actual measurement and the model prediction are computed and stored. From these residuals both the mean value and the standard deviation σ_r are calculated. By using these variables, the threshold level is defined by $\pm 6\sigma_r$ [84]. This process is performed on all prediction models.

As it can be seen in Figure 6.1, the optimised model from the supervised training performs regression on the testing and damage state sets. The prediction generated by the model is then compared with the actual value to calculate the residual error. The previously defined threshold level is then employed for anomaly detection by using the residual error. All residual values that exceed the threshold will be detected as abnormal data. Following this process, detection rate is calculated using

$$Detection\, rate\ (\%) = \frac{n_d}{N} \times 100 \tag{6-4}$$

where n_d is the number of residuals exceeding the threshold and N is the total number of data points. False positive might occur when healthy state data are detected as abnormal data. However, as long as the total number of detections is lower than the confidence level that is utilised in defining the threshold, then this condition might be considered as outlier.

6.3 Case Study: Warwick Bridge Experiment

6.3.1 Data Processing

Initially, high frequency noise from the data was removed by using a low pass filter with 50 Hz cut-off frequency. For this purpose, a Butterworth filter was employed. In addition, to remove Linear trend from the data, a predefined function “detrend” on MATLAB was employed.

6.3.2 Parametric Study on MPCA

As mentioned in section 5.3.2, the modal test in Warwick Bridge experiment was performed in six configurations in order to collect vibration data in all testing points. From this process, six datasets containing observation from strain gauges and displacement transducer have been collected for every condition (healthy, DM1, DM2, and DM3). Each dataset contains 30000 datapoints. The healthy state datasets are divided into training, validation, and testing sets. These datasets are assigned to be training, testing, and validation sets with following configuration:

- Datasets 1-4 are utilised as training set.
- Dataset 5 is employed as the validation set.
- Dataset 6 is used as the testing set.

Since the proposed method only employs healthy state data in the training process, the damaged state datasets are not split. All damaged state datasets are used along the testing set to evaluate the effectiveness of the proposed damage detection method.

In the investigation on the MPCA parameters, regression analysis using linear regression model is performed. This is due to the cheap computation resources required in applying linear regression model. The results from the combined MPCA-linear regression approach also serve as benchmarks for the combined approaches. There are three investigations onducted in this research: investigation on the impact of MPCA window size, investigation on the sensor type, and investigation on the correlation coefficient.

6.3.2.1 Investigation on the Impact of MPCA Window Size

The first parametric study performed in this research is investigating the impact of MPCA window size. For this purpose, in total 11 window sizes N_s are used such as 10, 50, 100, 200, 500, 1000, 5000, 6000, 10000, 15000, and 20000. By performing MPCA using various MPCA window sizes, several time series of the first eigenvector from all datasets (training, validation, testing, DM1, DM2, and DM3) are generated. Due to the variation in the window size, the number of observations in the eigenvector time histories after MPCA is also varied according to (6-1).

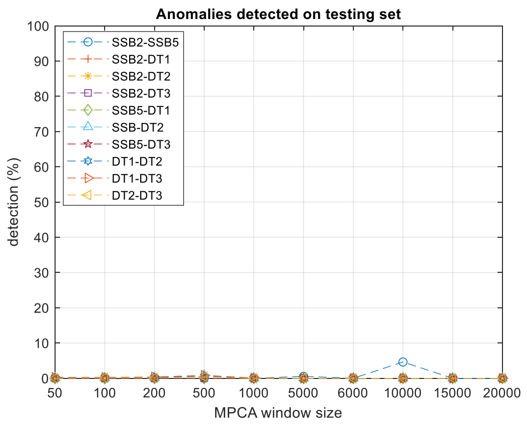
The next step is calculating the correlation coefficient on the eigenvector time histories in the training set. From this process, pairs of sensors with correlation coefficient higher than 0.85 are picked and stored. The total number of pairs stored for various MPCA window size can be seen in Table 6.1.

As it can be seen in Table 6.1, the correlation coefficients between sensor pairs are affected by the MPCA window size. When the window size is too small, no sensor pair creates correlation coefficient higher than 0.85. On the other hand, it is also found that increasing the window size above a certain value will not increase the correlation coefficient between sensors. As it can be seen in Table 6.1, only 51 pairs are having correlation higher than 0.85 when window size of 15000 is implemented. This number is also decreased to 17 as the window size is increased to 20000. In this parametric study, regression analysis using linear regression models is then performed on the pairs of sensors having correlation coefficient higher than 0.85 for each MPCA window size.

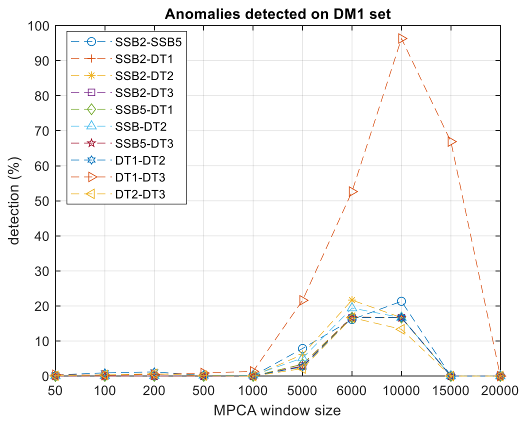
Table 6.1 The number of pairs having correlation coefficient higher than 0.85 from various MPCA window sizes

MPCA Window Size	Number of Pairs Identified
10	0
50	41
100	64
200	89
500	148
1000	246
5000	445
6000	465
10000	464
15000	51
20000	17

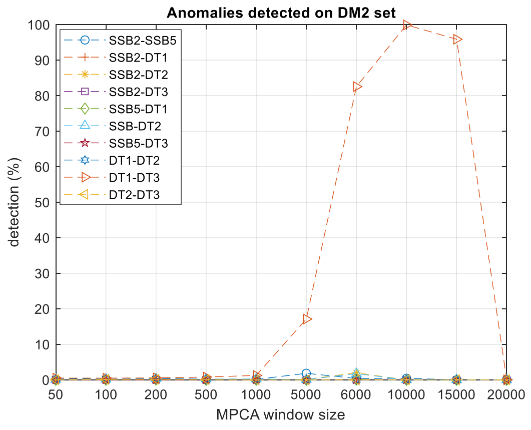
As it is shown in Table 6.1, the number of sensor pairs having correlation coefficient higher than 0.85 is affected by the choice of MPCA window size. Some pairs might produce correlation coefficient higher than 0.85 on several MPCA window sizes. However, these pairs might also produce correlation coefficient lower than 0.85 on other MPCA window sizes. In order to observe the impact of window size to the performance, the results from sensor pairs that have correlation coefficient higher than 0.85 on every MPCA window size are observed. For this purpose, in total, eight pairs have been identified and collected for observation. By using these pairs, linear regression models have been trained and detection rates from these models have been collected. Figure 6.3 shows the detection rate obtained from the regression model utilising data generated by using various MPCA window sizes.



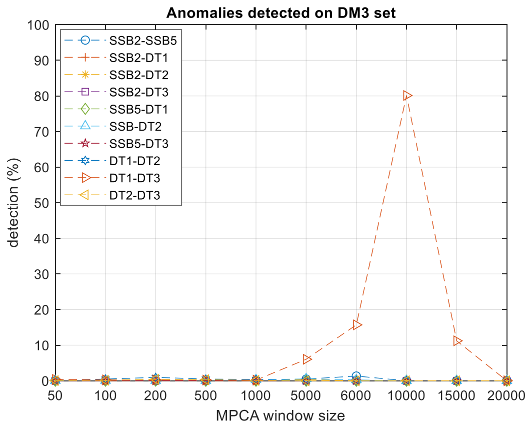
(a)



(b)



(c)



(d)

Figure 6.3 Detection of Anomaly from various MPCA window size; (a) at testing set; (b) at DM1 set; (c) at DM2 set; (d) at DM3 set.

From Figure 6.3, it can be seen that be seen that the damage can be successfully detected when using MPCA window size of 5000, 6000, and 10000. When smaller window sizes are implemented (50, 100, 200, 500, and 1000), most sensor pairs such as the SSB2-SSB5, SSB2-DT1, SSB2, DT2, SSB2-DT3, SSB5-DT1, SSB5-DT2, SSB5-DT3 pairs are unable to detect the damage. On the other hand, other sensor pairs such as DT1-DT2, DT1-DT3, and DT2-DT3 manage to produce detection on these window sizes. However, these detections are achieved at the cost of high level of false positives. As an example, at MPCA window size of 500, the DT1-DT3 pair manages to produce detection rate of 0.87%, 0.79%, and 0.2% on DM1, DM2, and DM3 sets respectively. However, the pair also generates 0.83% of false detection on the testing set. The false positive occurs when the healthy state data are falsely detected as abnormal data. The number of false positives is calculated from the number of residuals from the testing set that exceed the threshold level. Ideally, the model should produce low level of residuals on the testing set because the model is trained using healthy state data and the testing set contains data from the healthy state. Therefore, the anomaly that is generated on the testing set is considered as false detection.

When the MPCA window size is configured at 5000, 6000, and 10000, some pairs manage to perform damage detection from the dataset. As depicted in Figure 6.3, The DT1-DT3 pair is able to yield detection on all damage scenarios at these MPCA window sizes. Other potential pairs also manage to detect the presence of damage such as SSB2-SSB5 and SSB5-DT2. At MPCA window size of 6000, the former generates detection rate of 16.14%, 0.38%, and 1.33% on DM1, DM2, and DM3 sets respectively While the latter achieves detection rate of 19.41%, 1.75%, and 0.3%. In addition, from the figure, it can be seen that some pairs are only able to detect damage in one or two scenarios out of three damage scenarios. For example, at window size of 10000, the SS2-DT3 pair generates 16.7% detection rate on DM1 set. However, the pair does not yield high detection rate on the other damaged set. This result might be caused by the selection of sensor pair that is not suitable for detecting the introduced damage.

At MPCA window size of 15000, only the DT1-DT3 pair manages to detect the damage with detection rate of 66.9%, 95,88%, and 11.23% for DM1, DM2, and DM3 sets respectively. In addition, no false positive is produced by the pair when MPCA window size of 15000 is implemented. On the other hand, no anomaly is detected by other pairs at this MPCA window size. Finally, at window size of 20000, all sensor pairs are unable to detect any anomaly from the data.

In addition, it is also observed that correlation coefficient is affected by the MPCA window size. In Figure 6.4, it can be seen that the correlation coefficients of all sensor pairs rise as the window size is increased from 50 to 6000. When the window size is further increased to 10000, the coefficients in all sensor pairs start to drop and the values continue to decrease when the window size is set to 15000 and 20000. This result is in good agreement with the information in Table 6.1.

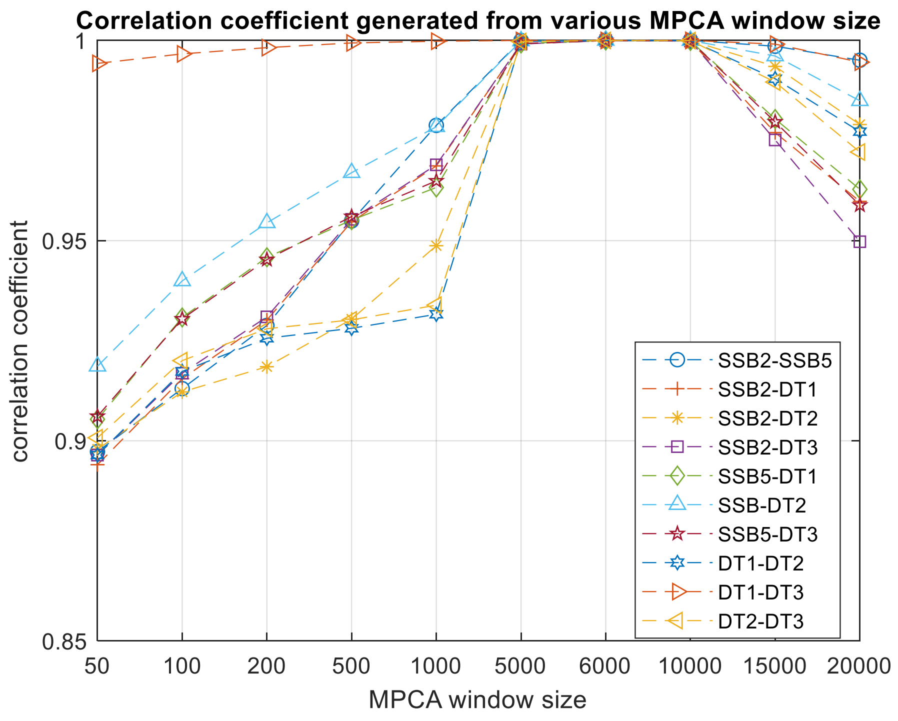


Figure 6.4 Correlation coefficient between sensors from various MPCA window sizes

From the observation, it is found that the ideal window size for the problem under study is between 5000 to 10000. One factor that might contribute to the result is the period of the chirping signal implemented on the shaker. As mentioned in section 5.3.2, including the free decay period, each excitation from the shaker takes about 64s. With a sampling rate of 100Hz, the excitation period corresponds to 6400 data points. As it is reported in [84], [98], when periodic variability presents in the dataset, the MPCA window size has to be at least the longest periodic behaviour.

When small window size (50, 100, 200, 500, and 1000) is implemented, the window is unable to capture the trend from the dataset. When the optimal window size (5000, 6000, or 10000) is applied, the window manages to collect the data pattern thus generating high correlation coefficient between sensors as shown in Figure 6.4. In these window sizes, the correlation coefficients almost reach 1 which is the maximum value of the coefficient therefore producing nearly flat area as it can be seen in the figure. On the other hand, configuring the MPCA window size above certain level also does not yield better results. This can be seen from the lower detection on DT1-DT3 pair when the MPCA window size is 15000 and 20000 compared to its detection at window size of 5000, 6000, and 10000. Therefore, further observation is performed by applying the MPCA window size of 5000, 6000, and 10000.

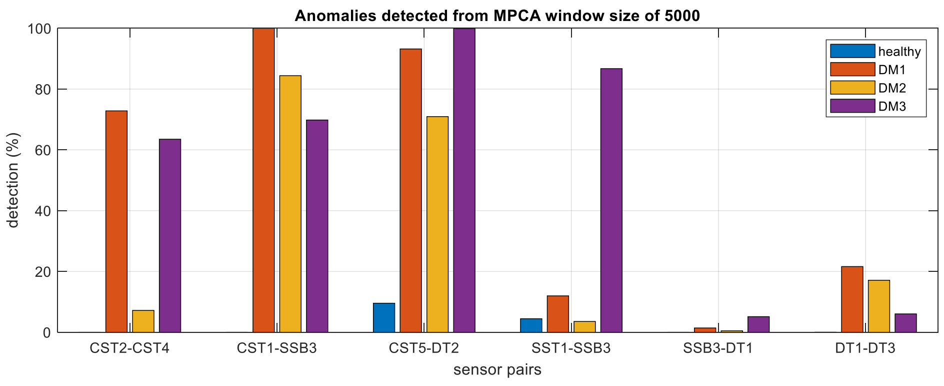
6.3.2.2 Investigation on Sensor Type

The next parametric study is observing the detection rate based on the sensor type selected for the regression model. Data from displacement transducers and strain gauges are employed in this study. Furthermore, two types of strain gauges are utilised: strain gauges that are installed on concrete deck and strain gauges which are deployed at the steel beam. In this part of the research, the sensors are grouped into three categories: displacement transducer (DT), concrete strain gauge (CS), and steel strain gauge (SS). Then, from these categories, in total six combinations that form sub-categories are generated: CS-CS, CS-SS, CS-DT, SS-SS, SS-DT, and DT-DT. All sensor pairs are further split into these sub-categories. In order to investigate the impact of sensor type to the detection, the pair that yields the highest detection rate for each sub-category is picked. The sensors pairs selected in this parametric study are as follows:

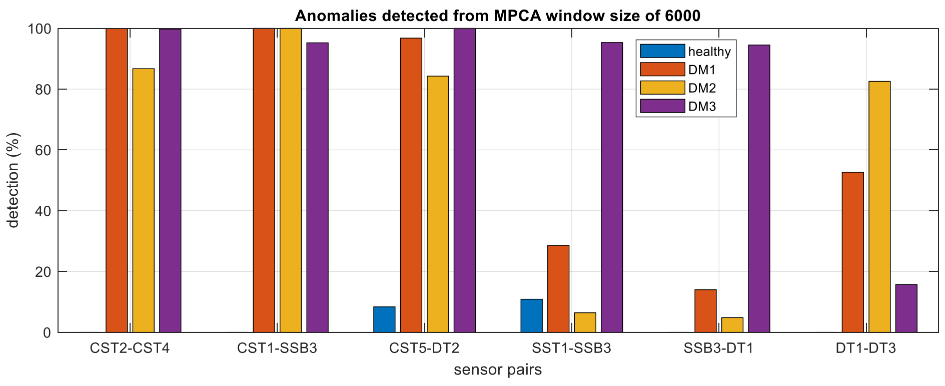
- CST2-CST4 for CS-CS
- CST1-SSB3 for CS-SS
- CST5-DT2 for CS-DT
- SST1-SSB3 for SS-SS
- SSB3-DT1 for SS-DT
- DT1-DT3 for DT-DT

As mentioned previously, the MPCA window size utilised in this observation is 5000, 6000, and 10000. Figure 6.5 shows the detection rate generated by the selected sensor pairs at MPCA window size of 5000, 6000, and 10000.

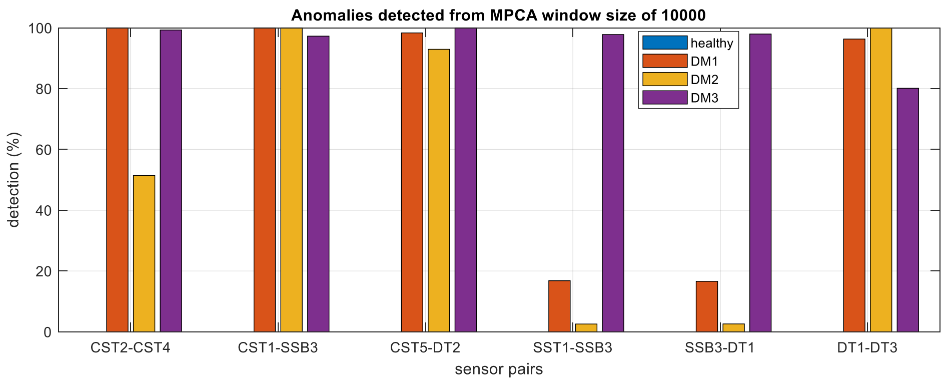
From Figure 6.5, it is shown that most sensor pairs manage to predict between undamaged and damaged condition correctly. As shown in Figure 6.5, there are two sub-categories that are unable to correctly identify the damage: CS-DT (CST5-DT2) and SS-SS (SST1-SSB3). Despite producing high detection on the damaged state sets, both pairs produce high false positive rate on MPCA window size of 5000 and 6000. However, when the MPCA window size is set to be 10000, both pair manages to correctly differentiate between healthy and damaged data as the false positive rate produced by these pairs are reduced to 0.



(a)



(b)



(c)

Figure 6.5 Maximum detection produced by sensor pairs from six sub-categories; (a) at MPCA window size of 5000; (b) at MPCA window size of 6000; (c) at MPCA window size of 10000.

In addition, it is also observed that the best MPCA window size differs among the sub-categories. As it can be seen in Figure 6.5, the CS-CS sub-category perform best when the window size is 6000. For this pair, increasing the MPCA window size to 10000 results to a significant drop in detection on DM2 set from 86.7% to 51.4%. In addition, For CS-SS sub-category, no significant difference is observed between the detection produced at MPCA window size of 6000 and 10000. On the other hand, the rest of the sub-categories produce highest detection at MPCA window size of 10000. Hence, from this observation, it can be concluded that different sensor types might require different value of MPCA window size to successfully detect the damage.

From this parametric study, it is found that the sensor types are sensitive to the type of damage. As it can be seen in Figure 6.5, the SS-SS pair that is formed from two steel strain gauges is more sensitive in detecting damage at the DM3 set. In DM3 scenario, damage at the steel beam is introduced, while on the previous DM1 and DM2 sets, damage only occurs at the concrete deck. Therefore, this pair does not yield high detection rate on DM1 and DM2 sets. In addition, it is also found that the pairs from CS-CS and CS-SS sub-category manage to achieve high detection in all damaged sets. The latter combines sensors that are deployed on both the concrete deck and the steel beam thus it is likely to be sensitive on the damage applied on both the concrete deck and the steel beam. On the other hand, the pair from CS-CS category is able to produce high detection at DM3 set because the scenario also includes damage at the concrete deck that has been introduced on the previous damage scenario in addition to the cut at the steel beam. Finally, it is observed that the DT-DT pair has shown high sensitivity in detecting damage on all damage scenarios.

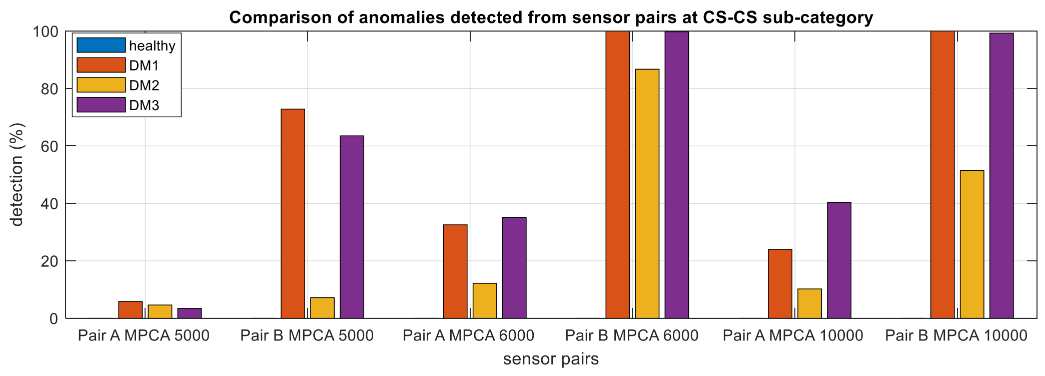
6.3.2.3 Investigation on the Correlation Coefficient

Parametric study on the impact of correlation coefficient to the detection rate is also conducted in this study. In the previous part of the research, all sensor pairs have been categorised into six sub-categories. In this parametric study, the sensor pair that produces the highest correlation coefficient for each sub-category is selected for analysis. The MPCA window size is set to be 5000 since most sensor pairs manage to perform high detection rate at this window size. The pairs selected in this part of the study are as follows:

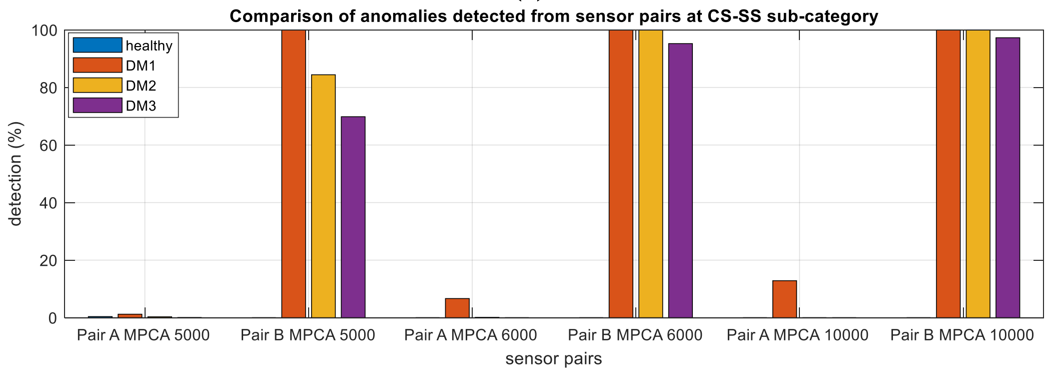
- CST4-CST9 for CS-CS
- CST3-SSS1 for CS-SS
- CST3-DT2 for CS-DT
- SSB2-SSB5 for SS-SS

- SSB5-DT2 for SS-DT
- DT1-DT3 for DT-DT

Aside from the DT1-DT3 pair for DT-DT sub-category, the other pairs selected in this parametric study differ from the pairs picked in section 6.3.2.2. Therefore, analysis is only performed by comparing the pairs in five sub-categories. Figure 6.6 shows the detection on sensor pairs with highest correlation coefficient and sensor pairs with highest detection at five sub-categories using MPCA window size of 5000, 6000, and 10000.

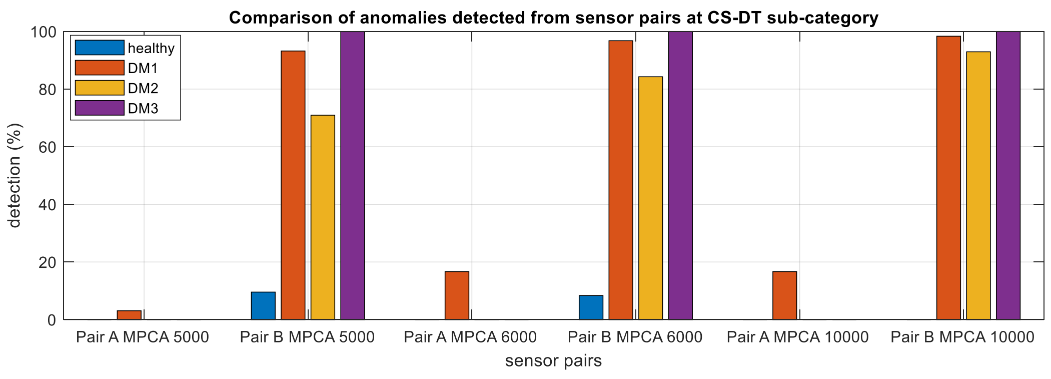


(a)

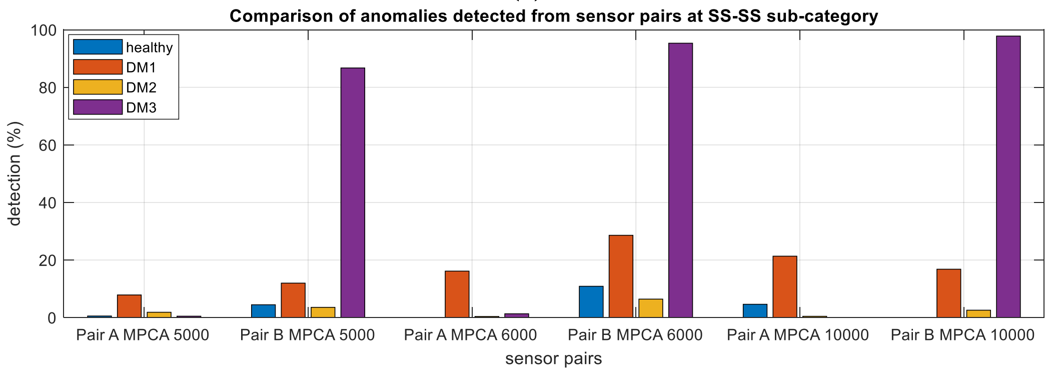


(b)

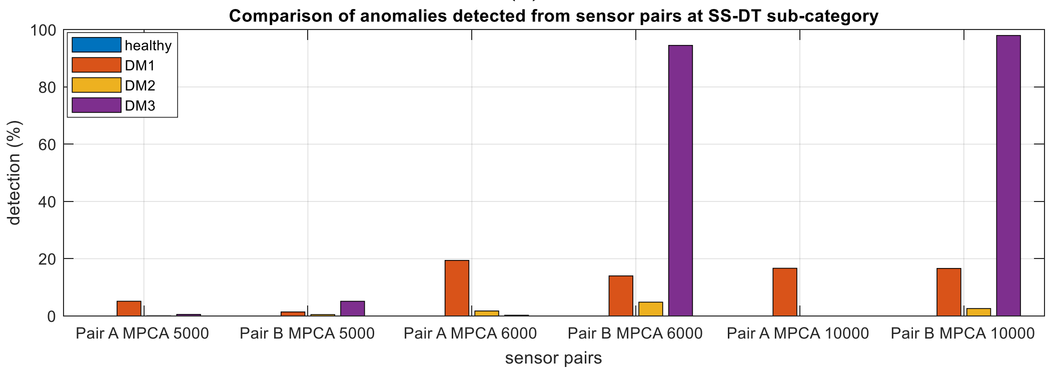
Figure 6.6 Comparison of detection produced from sensor pairs from six sub-categories. (a) detection from CST4-CST9 (Pair A) and CST2-CST4 (Pair B); (b) detection from CST3-SSS1 (Pair A) and CST1-SSB3 (Pair B); (c) detection from CST3-DT2 (Pair A) and CST5-DT2 (Pair B); (d) detection from SSB2-SSB5 (Pair A) and SST1-SSB3 (Pair B); (e) detection from SSB5-DT2 (Pair A) and SSB3-DT1 (Pair B).



(c)



(d)



(e)

Figure 6.6 Continued.

As it is illustrated in Figure 6.6, for each sub-category, the pair with highest correlation coefficient in the sub-category is not producing the highest detection rate. Furthermore, significant differences in the detection rates are also detected as it can be seen from the figure. For example, at MPCA window size of 5000 in CS-CS sub-category, the CST4-CST9 pair achieves 5.84%, 4.6%, and 3.4% detection at DM1, DM2, and DM3 sets respectively. On the other hand, the CST2-CST4 pair manages to produce 72.8%, 7.2%, and 63.51% detections at those sets. In this sub-category, significant differences are also produced at MPCA window size of 6000 and 10000. Similar results are also observed at other sub-categories. Therefore, high correlation coefficient does not guarantee high number of detection rate although it is important to select the highly correlated sensor pairs for regression analysis purpose.

Referring to Figure 5.4, the pairs with highest correlation coefficient in each sub-category are formed by sensors whose location are close to each other. According to [98], [99], [228], a sensor that is deployed near the damage location is more affected than the sensor that is far from the damage. Since the pairs with the highest correlation tend to be made by sensors that are close to each other, these pairs do not experience large change in the correlation coefficient due to the damage. On the other hand, by referring to Figure 5.4, most sensor pairs that produce high detection rate consist of a sensor that is located near the damage and a sensor far from the damage. For these pairs, one sensor is greatly influenced by the damage while the other is affected insignificantly. Therefore, when damage occurs, the change of correlation in these pairs are larger compared the change of correlation of sensor pairs with the highest correlation coefficient. Hence, this factor might further result to higher anomaly detection. Since the method exploits the change in correlation between sensor due to damage, it is more beneficial to select the pair whose correlation coefficient is greatly affected by the damage rather than the pair with the highest correlation coefficient. In addition, by exploiting the difference in the detection from various sensor pairs, it is possible to perform damage localisation.

6.3.3 Combined MPCA-CNN for Damage Detection

In the previous parametric study, some sensor pairs that are able to detect the presence of damage despite having low detection rate have been identified. As it has been mentioned previously, when the detection is higher than 0.3%, than the data can be considered to be anomaly. In this research, a novel method that function to elevate the number of detections is proposed.

The proposed method combines MPCA and CNN to detect the presence of damage. In the previous investigation, the regressor utilises a linear regression model. The proposed method employs CNN architecture as the regression model. In order to evaluate the effectiveness of the proposed method, comparison between the result from the proposed method and the benchmark is presented. In addition, the implementation of other machine learning techniques such as the ANN and Random Forest are also investigated for comparison. As a case study, the CST4-CST9 pair with

MPCA window size of 5000 is selected. From the previous investigation, the combined MPCA-linear regression model produces 5.84%, 4.6%, and 3.4% detections rate at DM1, DM2, and DM3 sets, respectively.

6.3.4 Parametric Study on Combined MPCA-CNN for Damage Detection

Firstly, MPCA using window size of 5000 is performed on all sets (healthy and damaged sets) to extract the first principal component of those datasets. To employ CNN architecture, data needs to be transformed into sequences. For this purpose, a moving window is used on the data. The size of the CNN window determines the amount of data in each sequence. For this study, the CNN window size is considered as one of the CNN hyperparameters. The CNN window sizes implemented in this study are: 50, 100, 500, 1000, 2000, and 3000. Using these window sizes, data frames whose construction is described in Figure 6.2 are generated. This step is conducted on all sets: training, validation, testing, DM1, DM2, and DM3 sets.

The new transformed sets are then employed on CNN architectures. As it has been mentioned previously, the prediction model is trained using the training set. The total number of epochs is 1000 and the early stopping patience applied in the research is 200. In each epoch, the model is tested using the validation set and the training process is stopped when no decrease in validation loss is achieved after 200 epochs. The batch size applied in this research is 512 as high batch size might lead to better accuracy and convergence [230].

In addition to the CNN window size, the CNN architectures are investigated in the parametric study. Figure 6.7 describes the CNN architecture employed in this study.

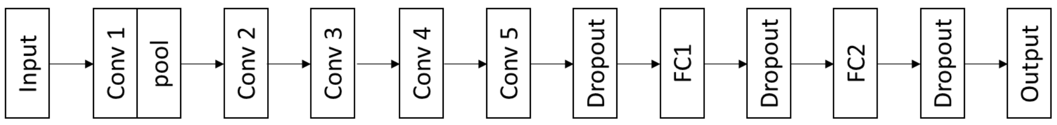


Figure 6.7 Basic CNN architecture employed in the research

As it is illustrated in Figure 6.7, the basic architecture used in this research consists of five convolution layers at the feature extraction stage and two fully connected layers at the regression stage. The first convolution layer is followed by a pooling layer whose size is 2. The size and the number of filters in each convolution layer differ as it can be seen in Table 6.2.

Table 6.2 Detail of filter number and size in each convolution layer implemented in the research

Layer number	Number of Filter	Filter Size
1st	16	16
2nd	16	8
3rd	32	5
4th	32	3
5th	32	2

As it can be seen in Table 6.2, the higher the number of convolution layer the larger the number of filters and the lower the filter size. This design is influenced by the Alexnet architecture [161]. The first and the second fully connected layer consist of 50 and 25 nodes, respectively. The activation function applied in all layers is ReLU. Since the output of the model is a single value, then the output layer only consists of one node. For hyperparameter optimisation purpose, variation in the architecture is introduced as it is described in Table 6.3.

Table 6.3 Detail of hyperparameters employed in the hyperparameter optimisation of CNN architecture

Parameter	Possible Values
Number of convolution layers	1, 2, 3, 4, and 5
Dropout probability	0, 0.2, and 0.5
Number of fully connected layers	1 and 2

It has been reported that the accuracy of CNN models could be enhanced by increasing the depth of the models [162], [163]. For example, the implementation of CNN model with deep architectures for ImageNet dataset have been producing high prediction accuracy [162], [163], [231]. However, the increase in the network depth leads to the higher number of parameters, which can potentially make the computation slower and decrease the testing accuracy because of overfitting [232]. Moreover, the models might be more susceptible to vanishing gradient issue as the depth increases [233], [234]. Hence the model depth can be seen as an important factor that impact the performance. In this research, Variation in the model depth is performed by modifying the number of convolution layers. As shown in Figure 6.7, the basic architecture is made of five convolution layers. The model depth in Table 6.3 then determines the number of active convolution layers. When the model depth is 5, then all convolution layers are utilised on the architecture. On the other hand, when the model depth is three, than only the first three convolution layers (denoted as Conv1, Conv2, and Conv3 in Figure 6.7) are employed on the architecture.

The next hyperparameter that is observed in this research is the dropout layer. As illustrated in Figure 6.7, there are in total three dropout layers. As it can be seen in Table 6.3, there are three possible values for dropout probability: 0, 0.2, and 0.4. When the value is 0, the architecture does not implement dropout layer. On the other hand, if other value is selected, then that value will be assigned to the three dropout layers applied in the architecture as the dropout probability.

Finally, the impact of the number of fully connected layers utilised in the CNN architecture is also investigated. As shown in Table 6.3, there are two options for the number of fully connected layers: one and two layers. When the architecture only consists of one layer then the FC1 layer on Figure 6.7 is deactivated. Otherwise, both FC1 and FC2 are active. In the hyperparameters optimisation, in total 180 combinations of hyperparameters are observed.

6.3.5 Comparison with Other Machine Learning Techniques

In addition to the comparison with the benchmark that employs linear regression as the regression model, the combined MPCA-CNN method is also compared with other combined methods utilising ANN and Random Forest as the regression model. Unlike the CNN modesl, the ANN and Random Forest do not require the input in the form of data sequences. In addition, the models are able to process multiple features as the input. For comparison purpose, both the ANN and Random Forest will perform regression to estimate the measurement value at CST9 by using all other measurements. Similar to the case study in combined MPCA-CNN method, the MPCA window size implemented for comparison is set to be 5000. For comparison purpose, a number of ANN and Random Forest architectures are generated for hyperparameter optimisation. The ANN and Random Forest architectures that yield the highest detection rate are then picked for comparison with the combined MPCA-CNN model.

6.3.5.1 Optimisation of ANN Architectures

Hyperparameters optimisation for ANN is also carried out by performing grid search on several ANN architectures. Table 6.4 shows the detail of ANN architectures investigated in this study.

Table 6.4 Detail of hyperparameters utilised for optimisation of ANN architecture

Hyperparameters	Possible Values
Number of nodes in the first hidden layer	10, 50, and 100
Number of nodes in the second hidden layer	0, 10, and 50
Number of nodes in the third hidden layer	0, 10, and 50
Dropout probability	0, 0.2, and 0.5
Activation function	ReLU and Sigmoid

The basic architecture of the ANN employed in this study consists of three hidden layers and a single node at the output layer. As shown in Table 6.4, the first three parameters determine the number of nodes in the hidden layers. For the second and third hidden layers, if 0 is picked, then the corresponding layer is deactivated, reducing the number of hidden layers. A dropout layer is implemented on each hidden layer, thus in total maximum of three dropout layers are employed. The dropout probability is optimised in the hyperparameter optimisation. Three possible values are utilised as it can be seen in Table 6.4. These values determine the dropout probability in all dropout layers. Finally, the activation function used in all hidden layer is also optimised. There are two activation function investigated: ReLU and sigmoid. In this part of the study, grid search is performed on combinations. In total there are 162 combinations of hyperparameters investigated.

Training is performed for 1000 epochs with mini batch size of 512. This is similar configuration that is applied on the training process of the CNN models. In addition, early stopping is also conducted using 200 early stopping patience.

6.3.5.2 Optimisation of Random Forest Architectures

In this work, there are four hyperparameters which are tuned in the optimisation of Random Forest architectures: the number of trees, the maximum number of variables at each split, the number of data point for bootstrapping, and the maximum depth of each regression tree. Table 6.5 describes the values implemented for each Random Forest hyperparameter in this study.

Table 6.5 List of hyperparameters for Random Forest optimisation

Hyperparameters	Values
Sample size	0.1, 0.3, 0.5, 0.6 and 1
Number of trees	500 and 1000
Maximum number of features at a split	5, 10, and 15
Maximum depth	10, 20, 50, and 'None'

The configuration for the random forest hyperparameters tuning is based on the suggestion from [118]. According to [118], the number of trees is either 500 or 1000. In addition, the maximum number of features is suggested to be p/3 for regression where p is the number of variables in the dataset. In this study, the number of input features that are used in the regression is 30. Therefore, the value of this parameter is chosen to be 5, 10, or 15. The sample size can be set between 0 to 1 and it determines the percentage of training dataset collected from the original dataset for training each tree. Smaller sample size might result to more diverse trees, decreasing the correlation between the trees thus it can potentially increase the accuracy after the ensemble process. On the other hand, using high sample size will make the trees more similar. Despite the positive impact, using smaller sample size means less data used in the training process which might reduce the accuracy of each regression tree [118]. Finally, the maximum depth of each regression tree is investigated with possible values provided in Table 6.5. The value in the table represents the maximum number of levels for each regression tree. When 'none' is selected, all regression trees in the corresponding model will have no depth limit. In total, 120 combination of hyperparameters are observed for Random Forest model.

6.4 Results and Discussions

In this section, parametric study on CNN model is presented. First, the result obtained from the proposed combined MPCA-CNN damage detection method is explained. Then, the result from the parametric study including the investigations on the number of data points in the data frame, the number of convolution layer in the architecture, the dropout layer probability, and the number of fully connected layer are presented. Furthermore, the comparison between the result from combined MPCA-CNN model and other combined MPCA with other regression analysis such as Random Forest, ANN, and linear regression is provided.

In the hyperparameter optimisation, in total 180 CNN models have been trained using both training and validation sets. Then, evaluation on these models is performed by applying the testing set on the models. Residuals calculated between the prediction and actual values are stored and compared with the threshold value calculated from the validation set. The detection rate is then measured from the ratio between the number of residuals above threshold and the total number of data points. In finding the most optimised model, the detection rates in both healthy and damaged state datasets are taken into account. While high detection rate in damage state dataset shows the sensitivity of the model in detecting the damage, high detection rate in the healthy state dataset indicates the model to be susceptible to false detection.

6.4.1 Hyperparametric Study on CNN Architecture

This section presents the results on parametric study performed for combined MPCA-CNN architecture.

6.4.1.1 Number of data in a sequence

The first parameter observed in this study is the number of data points in a data frame. This number is specified by the size of CNN moving window that is implemented in transforming the tabular data into sequences. There are in total six CNN window sizes employed in this study: 50, 100, 500, 1000, 2000, and 3000. The Investigation is performed in two stages. In the first stage, some combinations of hyperparameters are picked for the investigation. The CNN window size is varied while the other parameters are kept constant in order to observe the impact of the CNN window size to the model performance. In this purpose four combinations of hyperparameters are selected as shown in Table 6.6.

The detection rate produced from these combinations of hyperparameters with varying CNN window sizes can be seen in Figure 6.8. From the figure it can be seen that different trends are produced from the variation of the CNN window size on four hyperparameter combinations. In the first combination, no detection is produced when the window size is set from 50 to 500. The combination begins to perform detection when the window size is 1000 and reaches the peak when the window size is 2000.

The detection of the first combination decreases as the window size is increased to 3000. On the other hand, in combination 2, high detection only occurs when the CNN window size is 2000. When other window sizes are implemented, the combination produces detection rate lower than 1% in all damage scenarios. In addition, as it can be seen in Figure 6.8, on combination 3, high detection rate is only achieved on window size of 3000 while low detection rates are generated on other window sizes. Finally, on the last combination, the highest detection rate is generated at window size of 1000. In this combination, fluctuation in the detection rate is observed when the window size is increased to 2000 and 3000.

Table 6.6 Details of hyperparameter combinations for parametric Study on CNN window size

Combination	**Network Depth**	**Dropout Probability**	**Number of FC**
1	4	N/A	1
2	4	0.2	1
3	4	0.4	1
4	3	N/A	2

Due to the different in the trend obtained from the three combinations, the investigation on the effect of CNN window size to the detection rate is performed by picking the combination that yields the highest detection rate for each CNN window size. Figure 6.9 shows the highest detection rate generated for each window sizes. From the figure, it can be seen that high detection rates are produced only when the window size is 1000, 2000, and 3000 and increase in detection rate is achieved with the increase in window size. In addition, as it can be seen in Figure 6.9, the highest detection rate is generated in window size of 3000. However, despite the high detection rate on the damaged state dataset, this window size produces detection rate higher than 0.3% at the healthy state dataset. In this case, the model falsely predicts healthy state data as damaged state data. In this research, the CNN window size also affects the number of data frames as it is explained in (6-1). In this case, the higher the window size, the lower the number of data frames generated which might leads to the low amount of training data, potentially decreasing the performance of the CNN models. Furthermore, the implementation of high window size might be resource and time consuming. Because of these factors, the CNN window size of 2000 is picked as the optimised solution.

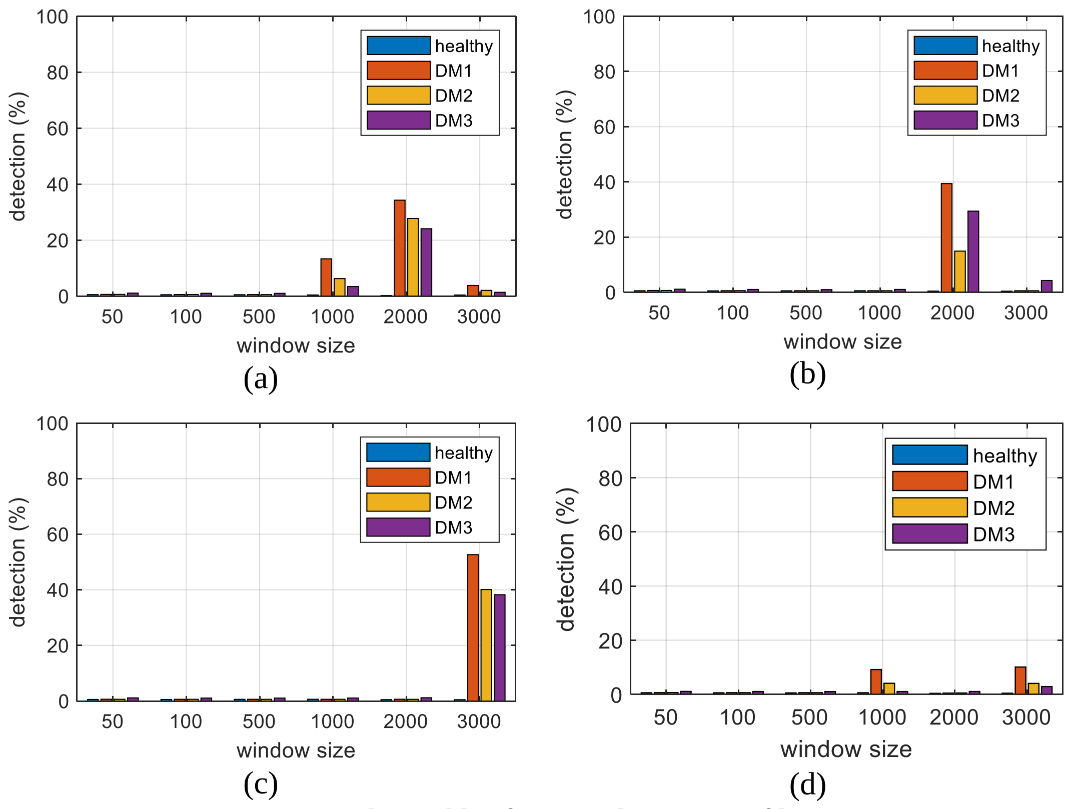


Figure 6.8 Detection rates achieved by four combinations of hyperparameters using various CNN window sizes

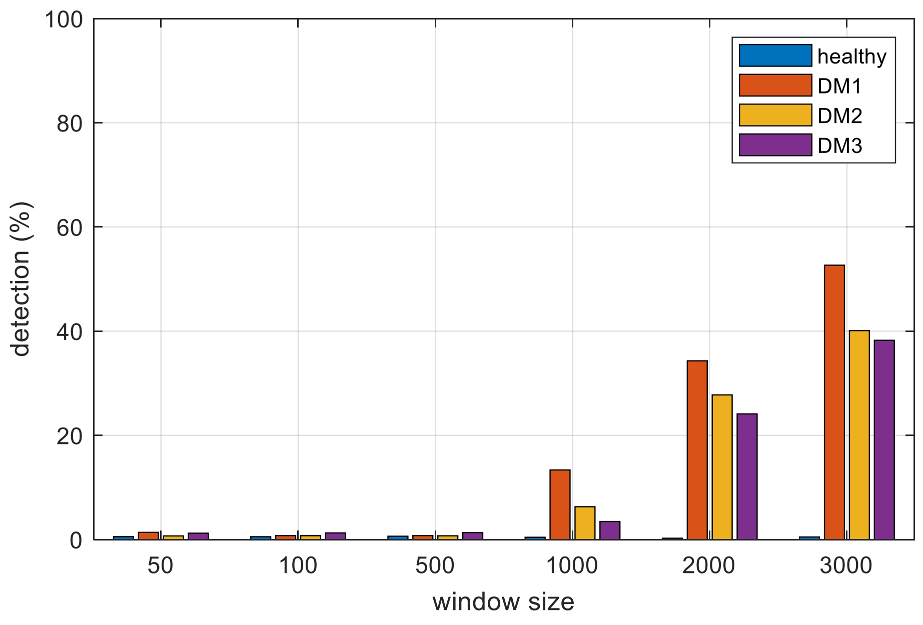


Figure 6.9 Highest detection rate generated for each CNN window size

From the investigation, it is found that the CNN window size has a complex relation with the detection rate. In some combinations of hyperparameters, the highest detection rate is achieved on CNN window size of 2000, while on others, applying the window size of 2000 could lead to low detection rate. However, it is found that the CNN models perform well in detecting the damage on window size of 1000, 2000, and 3000. Therefore, it is possible to narrow down the value of CNN window size in performing hyperparameter optimisation for the future work.

The size of CNN window corresponds to the number of data points in one data frame. When small window sizes are implemented, the CNN models are unable to retrieve important information from the data which leads to low detection rate on damaged state datasets. In this case, similar to the MPCA window size, the CNN window size needs to be sufficiently large in order to capture the important pattern from the data. Thus, the selection of CNN window size holds an important role in obtaining high detection rate and optimisation should be performed in order to obtain the optimal size.

6.4.1.2 Architecture Depth

In order to observe the impact of model depth, several models with various number of convolution layers have been trained and tested. In the observation, four combinations of hyperparameters are selected where all hyperparameters excluding the number of convolution layer are kept constant. These combinations are given in Table 6.7.

Table 6.7 Details of hyperparameter combinations for parametric Study on the model depth

Combination	**CNN Window Size**	**Dropout Probability**	**Number of FC**
1	2000	N/A	1
2	2000	0.2	1
3	3000	0.4	1
4	1000	0.2	2

Figure 6.10 presents the detection rates produced by the combinations described in Table 6.7, using different number of convolution layers. As it can be seen from the figure, it is shown that almost all combinations produce the highest detection rate when using four convolution layers. The first three combinations generate highest detection

rates with four convolution layers while obtaining low detection rate when other numbers of convolution layer are utilised. On the other hand, interesting result is obtained from the fourth combination. In this combination, low detection is produced when the architecture adopts four convolution layers. However, it should be noted that for this combination of hyperparameters, there is no optimal value for the model depth as it is shown in Figure 6.10.

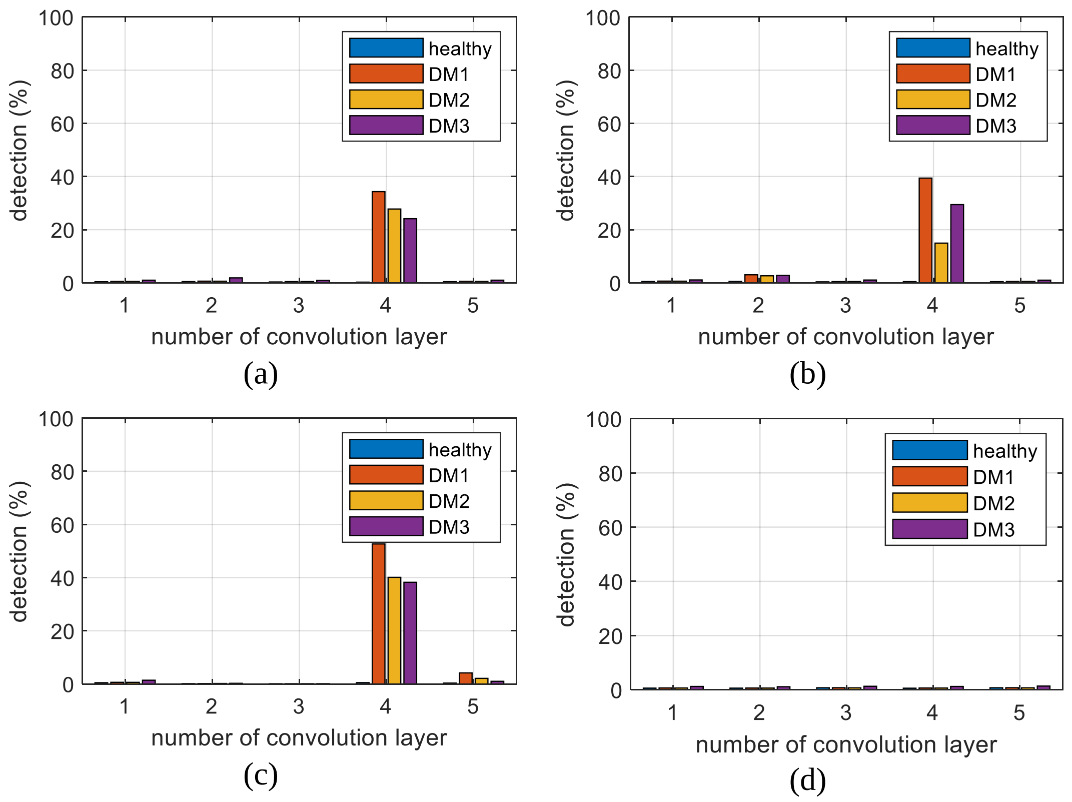


Figure 6.10 Detection rates achieved by four combinations of hyperparameters from several model depths

To further investigate the influence of network depth to the detection rate, for each number of convolution layer employed in this work, the combination that yields the highest detection rate is selected. The detection rates of these combinations can be seen in Figure 6.11. As it is illustrated in the figure, CNN model that implements four convolution layers achieves the highest detection rate for the problem under study. When lower number of convolution layers is applied on the CNN models, significant decrease in the performance is achieved. However, as the number if convolution layers is increased to five, the detection rate decreases slightly. In general, low performance on the implementation of low network depth occurs due to the incapability of the model to extract features from the data. On the other hand, overfitting tends to occur when employing CNN model using high number of convolution layers, potentially decreasing the detection rate. In addition, the implementation of deep architecture requires high number of parameters to train thus increasing the computational resources. Therefore, it is essential to find the suitable value of network depth due to its impact to the performance of CNN models.

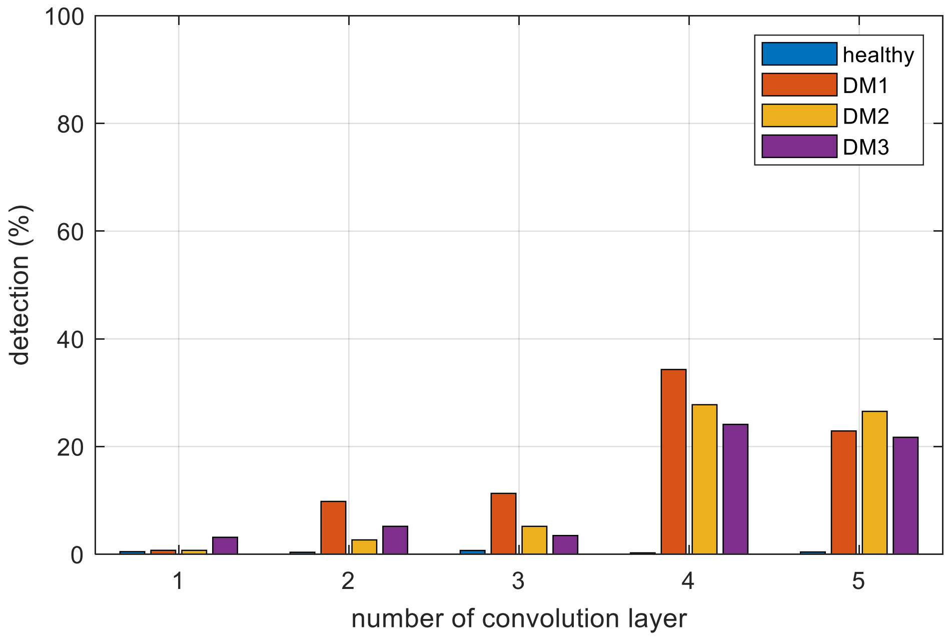


Figure 6.11 Highest detection rates generated for each model depth

The result on this investigation is different than the result obtained in the Chapter 5 where the CNN models achieve the highest classification performance when implementing a single convolution layer and increasing the number of convolutional layers will cause overfitting. This might be caused by the different in the approaches since this study executes the damage detection by using unsupervised learning through regression analysis while in the previous chapter, supervised learning for classification problem is implemented for damage detection. This result shows that there is no general solution when applying deep learning. Despite using the same dataset, the optimised hyperparameter on one approach is different from the other. Thus, it is essential to perform hyperparameter optimisation when employing deep learning on every problem.

6.4.1.3 Dropout Layer

In this study, a number of CNN architectures employing various dropout layers have been trained. Dropout layer has been implemented widely in deep architectures to combat overfitting. To observe the effect of dropout layer on the problem under investigation, four hyperparameters combination are selected. In the study, all hyperparameters excluding the dropout probability are unchanged while the dropout probabilities are varied. Table 6.8 describes the combinations investigated in this study.

Table 6.8 Details of hyperparameter combinations for parametric study on dropout probability

Combination	**CNN Window Size**	**Network Depth**	**Number of FC**
1	2000	4	1
2	1000	4	1
3	3000	4	1
4	1000	3	1

The detection rates produced by CNN architectures constructed using the combinations of hyperparameters presented in Table 6.8 can be seen in Figure 6.12. As it is presented in the figure, there is no clear trend produced between the dropout probability and the detection rate from all models. The model built using the fist combination generates the highest detection rate when no dropout layer is implemented. This model also yields detection on dropout probability of 0.2 and it is unable to detect anomaly when the dropout probability is set to 0.5. On the other hand, the model constructed using the second combination only manages to produce high detection without dropout layer. This model generates low detection rate with the addition of dropout layers on the architectures. As it is also described in Figure 6.12, the model that employs the third combination only detects anomaly when the dropout layer is set to be 0.5. However, no detection is made when no dropout layer is applied to the model thus the model generates different trend as opposed to the trends obtained from the first two combinations. Finally, the model constructed using the fourth combination of hyperparameters only performs detection when the dropout probability is 0.2. The model is unable to detect anomaly both when no dropout layer is implemented and at dropout probability of 0.5.

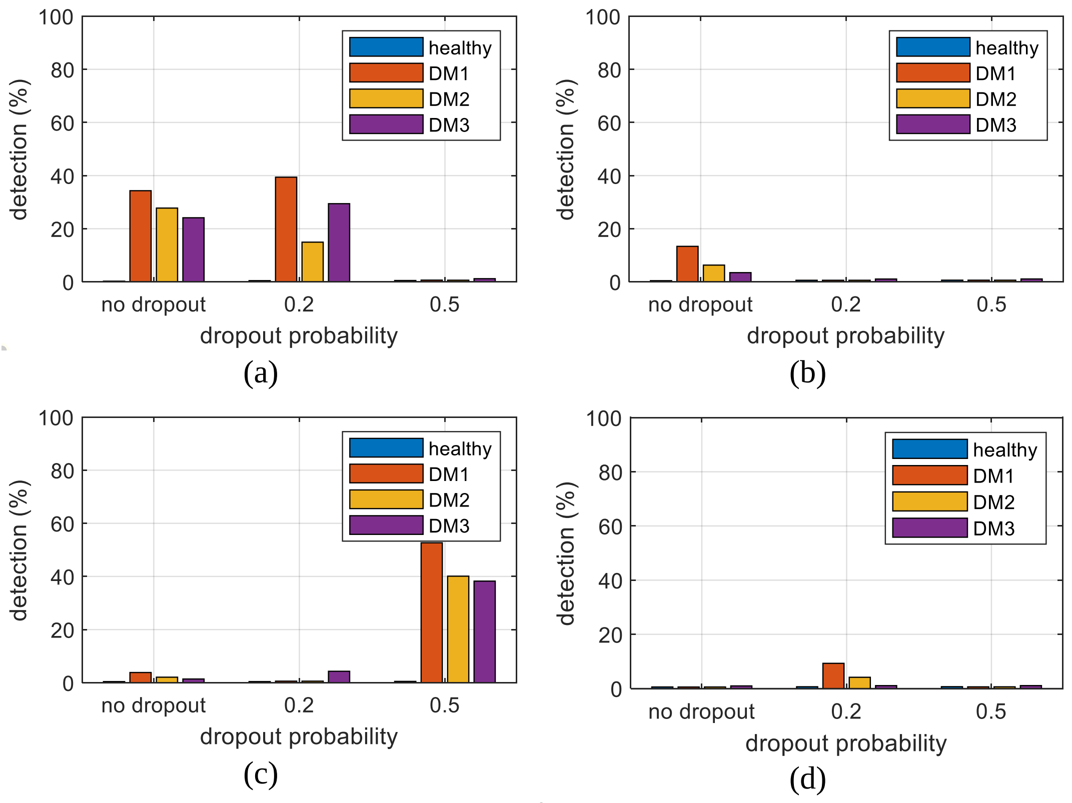


Figure 6.12 Detection rates achieved by four combinations implementing various dropout probabilities

Similarly, to the previous investigation on network depth and CNN window size, the highest detection rates obtained on various dropout probabilities are collected as it can be seen in Figure 6.13. From the figure, it is shown that there is no significant difference on the detection rates produced by models implementing various dropout probabilities. As it can be seen from the figure, highest detection rate is produced by model implementing dropout probability of 0.5. However, despite the highest detection on damaged state dataset, this model suffers from high false positive rate, due to the detection rate higher than 0.3% on the testing data. In addition, from the previous discussion, the optimal value of dropout probability tends to be specific for each combination. Therefore, when implementing dropout layer to solve a problem, optimisation algorithm should be applied to find the best value of dropout probability that works on the problem.

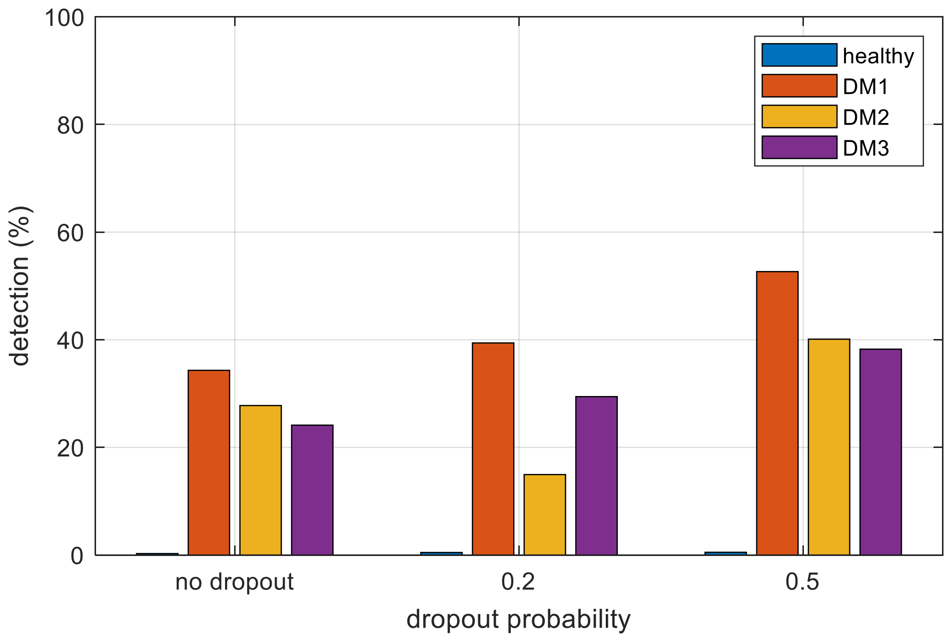


Figure 6.13 Highest detection rates generated for each dropout probability

6.4.1.4 Fully Connected layer

From the previous feature extraction stage, useful information from the data has been extracted and flattened. These features are fed to a regressor that is formed by fully connected layers. The architecture of the fully connected layer consists of input layer, hidden layer, and output layer. The input layer accepts features extracted from the feature extraction stage while the output layer generates regression results from the model. As it has been mentioned previously, the impact of fully connected layers is investigated by altering the number of hidden layers in the regressor. In this work, the investigation is carried out on models constructed using four combinations of hyperparameters by changing the number of hidden layers in the regressor while keeping the other hyperparameters unchanged. The combination of hyperparameters observed in this study is given in Table 6.9.

As it is illustrated in Figure 6.14, the number of fully connected layers affects the detection rates differently. On the first three combinations, the implementation of one hidden layer in the model architecture for the problem under study yields higher detection rate compared to the detection rate obtained from architectures with two hidden layers. In addition, significant difference in the detection rate is generated in the first three models. On the other hand, the detection rate for the fourth combination is lower when employing one hidden layer compared to using two hidden layers. Therefore, the utilisation of a single fully connected layer does not provide a general solution on the problem.

Table 6.9 Details of hyperparameter combinations for parametric study on number of hidden layers implemented in the regressor

Combination	CNN Window Size	Network Depth	Dropout Probability
1	2000	4	N/A
2	1000	4	0.2
3	3000	4	0.4
4	1000	3	N/A

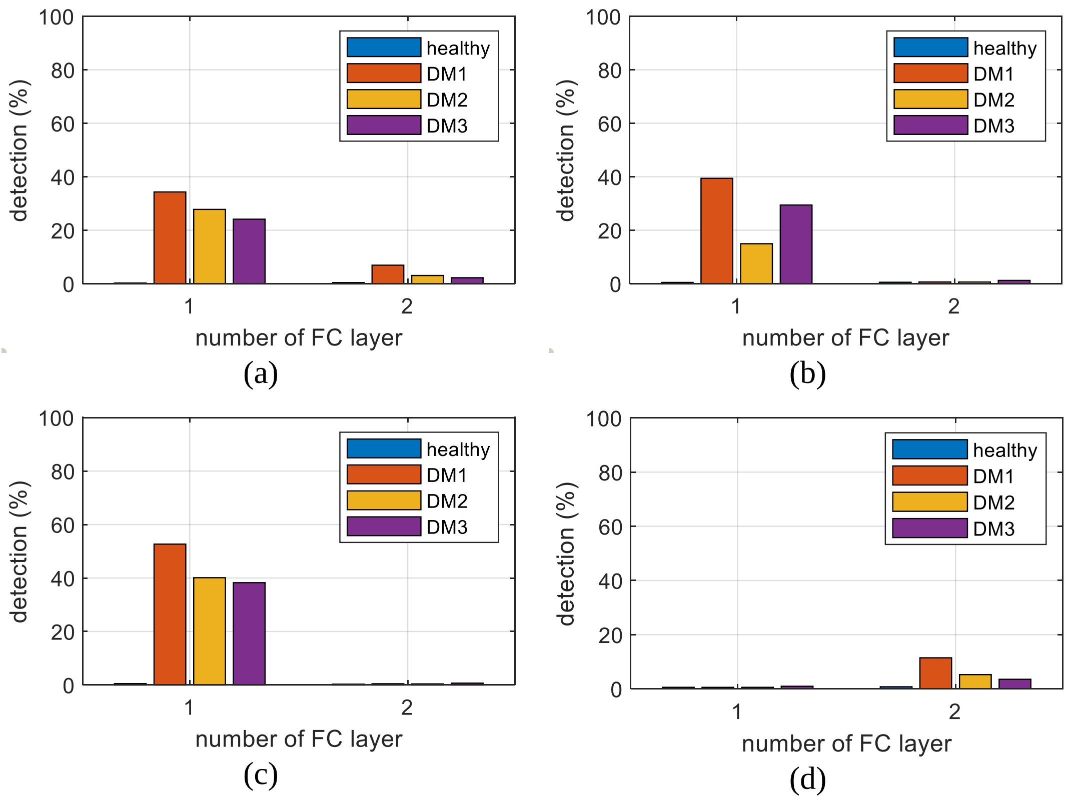


Figure 6.14 Detection rates achieved by four combinations of hyperparameters implementing various number of hidden layers on the regressor

To further observe the hidden layer’s impact to the performance of CNN models, for each number of hidden layers, the model that produces the highest detection rate is selected. The detection rates of these models are presented in Figure 6.15. In the figure, it can be seen that no significant difference on the detection rate is produced from implementing one or two hidden layers. In addition, from the previous investigation, it is found that finding the optimised value for hidden layer number can be tricky since this parameter impacts differently on various CNN architectures. Therefore, optimisation is essential in finding the optimal value of hidden layer number.

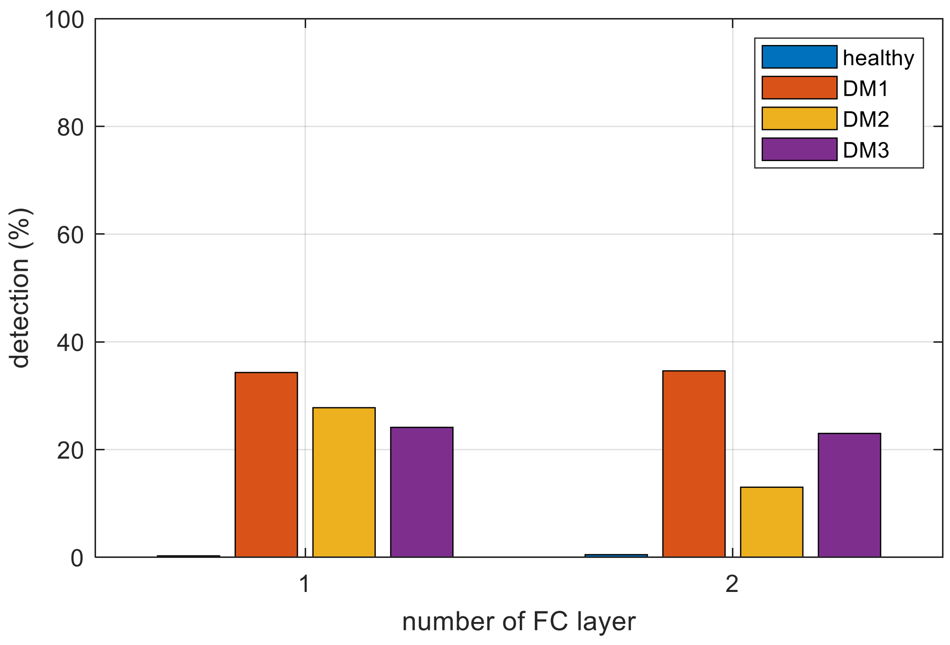


Figure 6.15 Highest detection rates generated for different regressor architectures

6.4.1.5 Summary of parametric study on combined MPCA-CNN

From the investigation on the impact of CNN hyperparameters on the detection rate, it is found that some hyperparameters such as the dropout probability and the number of hidden layers on the regressors have a complex relation with the detection rate. To obtain the optimised combination of hyperparameters, two parameters are taken into account: detection rate on the damaged state dataset, and detection rate on the healthy state dataset. Detection on the healthy state dataset represents false positive where healthy state data are falsely detected as damaged state data. Therefore, the optimised model should generate high detection rate on damaged state dataset while producing detection rate below 0.3% on the healthy state dataset. Table 6.10 shows, the optimised hyperparameter configuration.

Table 6.10 Optimised hyperparameters combination for CNN Architectures obtained using grid search

Parameter	Possible Values
Number of data in a sequence	2000
Number of convolution layers	4
Dropout probability	0
Number of fully connected layers	1

6.4.2 Comparison with other Machine Learning Techniques

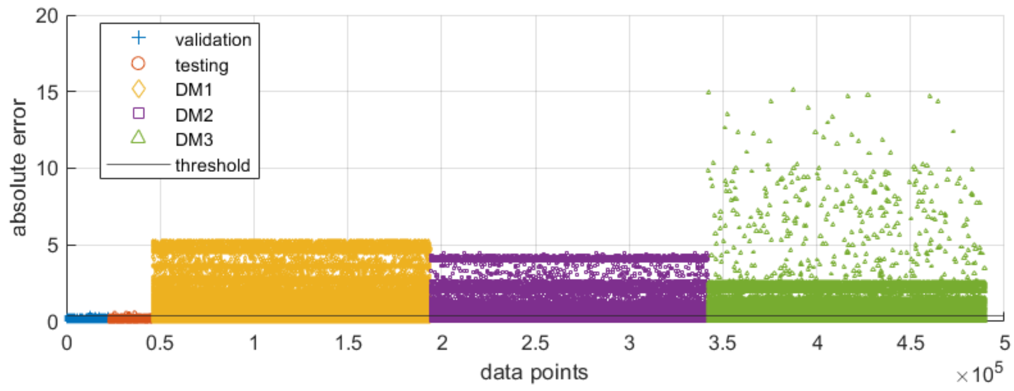
Hyperparameter optimisation has also been performed on the Random Forest and ANN architecture. In this research, two parameters including the detection rate and the magnitude of error are implemented to compare the combined MPCA-regression analysis methods. The first parameter used for the comparison is the detection rate. Table 6.11 shows the detection rate produced from linear regression, CNN, ANN, and Random Forest models.

Table 6.11 Comparison of detection rates generated using several regressors

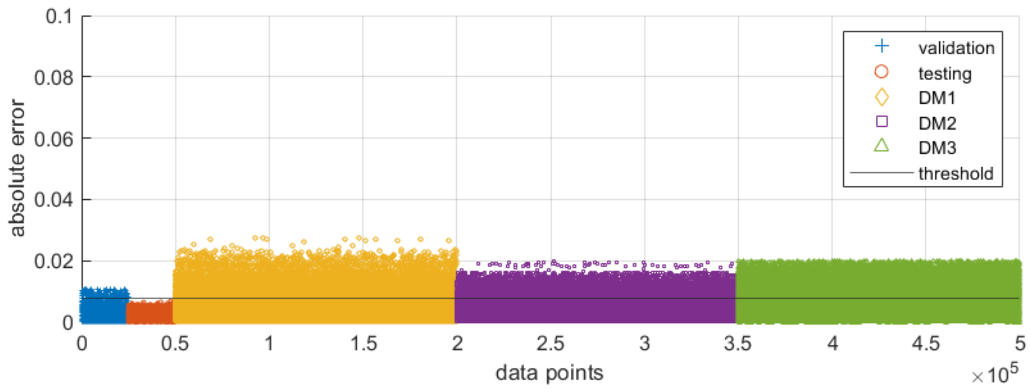
Model	Detection Rate (%)			
	Healthy	DM1	DM2	DM3
Linear Regression	0.2	5.84	4.61	3.4
CNN	0.27	36.8	29.79	25.88
ANN	0.32	15	17.44	33.90
Random Forest	0.16	11.30	1.22	2.17

As it is shown in Table 6.11, the CNN model outperforms other machine learning models in the detection on DM1, and DM2 datasets. This can be seen from the highest detection produced by the CNN model in these damaged state datasets. For DM3 dataset, CNN model is outperformed by the ANN model despite having higher detection rate than both the linear regression and Random Forest models.

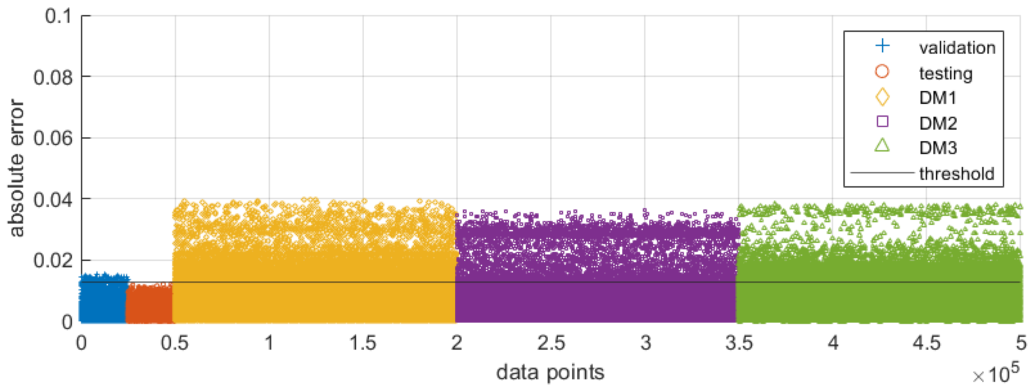
The next parameter for comparison is the magnitude of the absolute error produced by the prediction models. In this research, residuals calculated from the difference between the prediction and the actual values are calculated in the form of absolute error. Figure 6.16 illustrates the absolute error generated by four prediction models on the training, validation, testing, DM1, DM2, and DM3 sets.



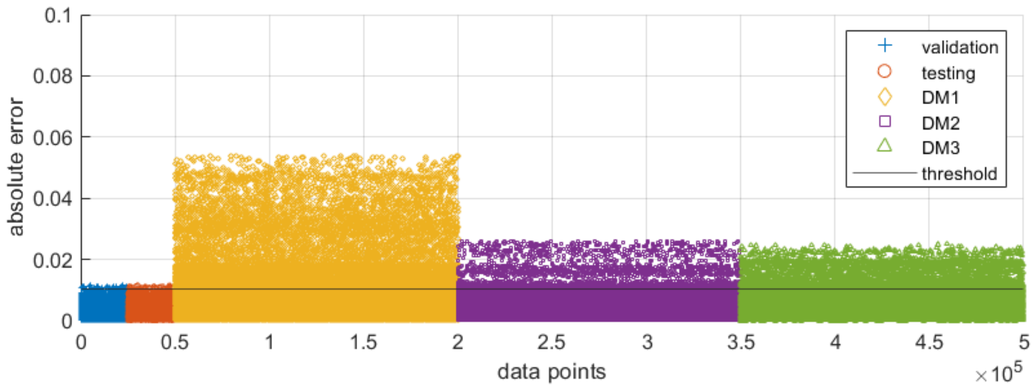
(a)



(b)



(c)



(d)

Figure 6.16 Residuals generated from Various combined methods; (a) MPCA-CNN; (b) MPCA-linear regression; (c) MPCA-ANN; (d) MPCA-Random Forest.

As it can be seen in Figure 6.16, the CNN model produces the highest absolute error compared to the other machine learning models on damaged state dataset. As it is shown in the figure, the CNN model manages to produce residual as high as 15 on the DM3 set. On the other hand, the magnitude of all residuals generated from the other models are less than 0.06. In this case, the CNN model shows high sensitivity to the damage thus when damage occurs, the model manages to produce high residuals between the prediction and the actual values.

On the other hand, as described in Figure 6.16, the CNN model produces false positive detection where heathy state data are detected as damaged state data. The false detection also occurs on other models as it is illustrated in the figure. However, as summarised in Table 6.11, the number of false detection rate generated by the CNN model is only 0.27% thus they can be considered as outliers. In the application of structure monitoring, the false positive scenario is more preferable than the false negative scenario in which the prediction model is unable to detect the presence of damage.

6.5 Conclusions and Summary

In this study, a novel damage detection approach that combines MPCA and CNN has been reported. The method has been validated using measurement on a laboratory-scale bridge. In this work, parametric study on MPCA method has been performed. The parameters investigated are the MPCA window size, the sensor type, and the correlation coefficient between sensors for regression analysis. In addition to the investigation on the parameters used in MPCA, parametric study on the CNN window size, model depth, dropout probability, and number of hidden layers, has also been performed on the combined MPCA-CNN method. Finally, the combined MPCA-CNN method has been compared with other methods that combine MPCA with other machine learning techniques such as linear regression, ANN, and Random Forest. Comparison using two parameters such as detection rate and magnitude of error has been presented. The conclusions in this chapter are as follows:

- For MPCA window size, it is found that the method can successfully detect anomaly using MPCA window size ranging from 5000 to 10000 and the Implementation of MPCA window size outside of the optimal range leads to low detection. This could be due to the period of excitation used in the experiment.

- In addition, parametric study on the sensor type has shown that the optimum MPCA window size depends on the sensor types used in the regression analysis. In addition, it has been shown that the damage sensitivity is also influenced by the type of sensors used in the regression.
- It is also found that using sensor pairs with high correlation coefficient does not guarantee high detection rate. Due to the different impact from the presence of damage, it is more beneficial in using a sensor pair formed by a sensor that is close to the damage location and a sensor far from the location rather than using pair with the highest correlation coefficient.
- In the observation on the CNN window size, it is found that when small window size is implemented, the method is unable to capture the trend from the data. On the other hand, the utilisation of large CNN window size reduces the number of training data, potentially minimising the performance.
- In addition, for the problem under study it is found that the models adopting four convolution layers generate the highest detection rate. The optimal number of CNN layers obtained for this study is different than the optimal number of CNN layers obtained in Chapter 5 although both studies employ similar dataset. This shows that there is no general solution on the implementation of deep learning.
- It is found that there is no optimal value for both the dropout probability and the number of hidden layers. These two hyperparameters affect the detection rate differently depending on the other hyperparameters. Therefore, employing efficient algorithms for hyperparameter optimisation is necessary to find the optimal value for these hyperparameters.
- In term of detection rate, the MPCA-CNN method outperforms the other methods in most of damaged state datasets. It only produces lower detection than the MPCA-ANN at DM3 set yet it manages to outperform the other methods in DM1 and DM2 sets.
- In term of magnitude of error, it is found that the MPCA-CNN method is able to generate larger amplification on the residuals produced on damaged state sets compared to other combined methods.

Chapter 7 Conclusions and Future Works

7.1 Conclusions

This thesis is aimed to discover the solution for problems in data-based interpretation method for SHM where high level of expertise is required in order to retrieve important information from raw data, both in the form of visual and time series data. In addition, the objective of the thesis is to improve the existing data-based interpretation method and perform validation using laboratory- and full-scale case studies. For this purpose, some projects implementing deep learning for data-based interpretation have been conducted through the research.

From the case study presented in Chapter 3, it has been shown that the deep learning-based model manages to provide estimation on load capacity, either in the form of load rating or design load, of bridges using their images. It is conducted by processing raw visual data without performing feature extraction, thus minimising the required level of expertise. From the parametric study conducted in the research, it has been found that the models for design load estimation outperform the models for load rating estimation. In addition, it has been shown that the angle which the images are taken might influence the performance of the prediction models. Moreover, it is observed that the use of colourful images is more suitable for the problem under study compared to the use of grayscale images. Finally, it has been shown that converting the multiclass classification into binary classification can enhance the performance of the prediction models.

In Chapter 4, a deep learning-based method that utilises correlation between sensor for SHM data interpretation has been presented. The method employs raw measurement data hence it requires no feature extraction. The method has been validated using a case study on a full-scale bridge. In this work, the method is implemented to estimate cable forces by using raw temperature measurement as the input. From this case study, it has been shown that the method is able to capture the trend of the cable force time histories with MAE ranging between 10.23 kN and 19.82 kN, MAPE ranging between 0.434 % and 0.536 %, and RMSE ranging between 13.38

kN and 25.32 kN. In addition, it has been shown that the method outperforms other regressors that utilise linear regression and ANN techniques.

In Chapter 5, damage detection approaches for SHM that implement CNN architecture have been presented. Two methods utilising CNN are presented: CorCNN that is based on anomaly detection and damage identification through supervised CNN. Both the CNN-based damage detection frameworks utilise raw measurement data from various sensors in detecting the presence of. These approaches have been validated using a case study on a laboratory-scale bridge. In this study, it is found that CorCNN manages to detect the presence of damage that is challenging to detect using conventional vibration-based method. In addition, it has been shown that the method is able to detect the presence of damage in all damage scenarios. However, it is found that the method is unable to provide information about the damage severity.

On the other hand, it has been shown that the damage detection framework utilising supervised CNN is able to detect both the presence and the severity of damage that might be challenging to be detected by using conventional frequency-based approach. From the parametric study performed in this research, it has been shown that for the case study, the models with shallow architecture produce higher classification accuracy compared to those with deeper architectures. In addition, it has been shown that there is no activation function that works best for all cases. However, from the research it is found that models utilising tanh activation function produces the lowest accuracy and require longest training time.

In Chapter 6, a novel damage detection method combining MPCA and CNN has been presented. The validation using measurement on a laboratory-scale bridge has been discussed and investigation on both the MPCA and CNN parameters has been reported. From the case study, it has been shown that the optimal MPCA window size depends on the periodic variability on the data. In addition, it has been shown that sensor types influence the sensitivity to certain damage type. On the other hand, investigation on CNN parameters has shown that it is necessary to perform optimisation in order to find the best configuration of CNN parameters. Finally, comparison between the combined MPCA-CNN method and other combined method has been presented. The combined MPCA-CNN method produces highest detection rate in most of the damage scenarios while producing the highest magnitude of residuals.

7.2 Future Works

Several limitations have been established through the course of this thesis. This section discusses the limitations of the contributions of this thesis and presents the promising areas for future work.

7.2.1 Deep Learning for Bridge Load Capacity Estimation in Post-Disaster and -Conflict Zones

In this thesis, it has been found that the quality of the image affects the performance of prediction models in estimating the load capacity (refer to section 3.3.2). In addition, the number of available data might also influence the prediction model performance (refer to section 3.3.1). However, this research employs images that are obtained through web scraping thus it is challenging control both the data quality and quantity. These limitations might be remedied by artificially generating bridge data and images.

In addition, it is possible to further the performance by training models from scratch. This research employs transfer learning to train prediction models due to the limited number of data available. By generating data artificially, more data will be available therefore prediction models can be trained from scratch.

In addition, improvement in the computational efficiency can be achieved by improving the hardware/software. As it has been mentioned in section 3.2.2, decrease in the computational time was achieved when GPU was implemented. Therefore, the computational time can be further reduced by utilising more advance GPU. On the other hand, it is possible to minimise the computing time through programming especially when the model is trained from scratch. When a model is trained from scratch, hyperparameter optimisation is performed to obtain the configuration of hyperparameters that work best for a given problem. The optimisation can be improved by using a large hyperparameter space. On the other hand, performing grid search in a large hyperparameter space can significantly increase the computing time. Therefore, efficient algorithm such as random search or genetic algorithm can be implemented in the process to minimise the time.

Finally, as it has been mentioned in section 3.3.5, limitation of image processing in extracting useful features such as bridge's condition and material is another factor that limit the applicability of the method. To address the current limitation of image processing in detecting structure's condition and gaining information about structural material, it might be useful to add the information regarding bridge's condition and material along with bridge images as the input of prediction model.

7.2.2 Estimation of Structural Response using Convolutional Neural Network: Application to Suramadu Bridge

In this thesis, a CNN-based method has also been implemented for generating data that can potentially improve SHM by tackling missing data problem as well as assisting the calibration process. The proposed method only employs temperature measurement as the input of the model (refer to section 4.3.2). However, the bridge responses can also be influenced by other factors such as traffic and wind loading. Future works are aimed to improve the performance of the CNN-based cable force estimation framework by taking into consideration other parameters that might impact the cable force measurement such as the wind and traffic loading. In current research, these parameters are excluded thus might become the source of uncertainties.

In addition, the expensive computation resources required in training CNN models is one of the limitations of the research. As a result, only a few hyperparameters are utilised in the optimisation process. In the optimisation, the impact of parameters such as the length of the input frame, the number of convolutional layers, the filter attributes including the size and stride, the type of pooling layer, the depth of fully connected layer, the type of activation layer are not observed. This is due to the large number of possible combinations of hyperparameters that might not be suitable for grid search. The future work will focus on employing genetic algorithm or random search for hyperparameter optimisation, which will enable the use of more hyperparameters without significant increase in the processing time.

7.2.3 Damage Detection and Identification Utilising Convolutional Neural Networks for Structural Health Monitoring

CorCNN, the first damage detection method presented in this chapter, performs the novelty detection based on the change in sensor correlation before and after the damage (refer to section 5.4.1). although the method directly utilises raw data for damage detection, the selection of combination for the inputs and output is still performed manually. In this case, the combination used in the research is selected based on the on the information that damage affects sensor measurements differently depending on the location of sensors relative to the damage. Hence, it is picked due to the prior understanding on the location of damage. On the other hand, the combination selected in the research might not be the best for the problem study. In addition, the combination used in this research might not produce high detection when damage occurs at a different location, and different combination of sensors might be more suitable. The aim of the future work will be to perform measurement system design to automatically select the sensors that ensure best performance for any given problem especially for real monitoring application that generally employs a large number of sensors.

Another potential approach for sensor selection can be performed by monitoring the correlation coefficient between sensors. During the initial step, the correlation coefficients from all sensor pairs are recorded. Then, on the monitoring phase, these coefficients are compared with the new coefficients. When significant deviation from the initial coefficient is detected, then, CorCNN can use the sensor pair for its input and output.

The parametric study for the damage detection method employing supervised CNN approach only investigates the impact of model depth and activation function (refer to section 5.4.3). To further increase the prediction accuracy, other hyperparameters of the CNN architecture can be tweaked. These parameters include the number of filters, the size of filter, the dropout probability, the batch size, the learning rate, and the window size of the data frame. However, adding more hyperparameters might significantly increase the required computing time. The future work will be focused on employing efficient algorithm for hyperparameter optimisation such as genetic algorithm or random search. By employing the algorithm, more hyperparameters can be investigated, thus optimum model architecture can be obtained without significant rise in the processing time. Based on the result obtained from this research, the implementation of shallow networks is more suitable for the problem. Hence, the hyperparameter optimisation will require less parameter to tweak.

In addition, due to the nature of the developed method, information detailing the condition of a structure is required in the training process. However, in the real-life situation, this information will not be available. Therefore, the future work is also aimed to generate data from a mathematical model (FEM model) representing the monitored structure. This way, damage can be simulated by using the model and damaged dataset can be produced for the training purpose CNN-based damage prediction model. The CNN-based model will then be validated by using experimental data. Finally, damage localisation is also aimed for the future work. By employing the FEM, damage can be simulated in various location in the bridge, and by labelling the data with information of the damage location, the detection of damage location might be achieved.

7.2.4 Combined MPCA-CNN for Damage Detection on Structures

The current research is only aimed to detect the presence of damage in the structure. Therefore, the investigation is mainly performed to assess if the combined MPCA-CNN method can detect the presence of damage on the structure. However, it has been shown that the combined MPCA-CNN method has the potential in predicting the damage location (refer to section 6.3.2.2). Therefore, this potential in locating damage can be further studied for future work.

In addition, the case study employed for validation of the proposed combined MPCA-CNN method only utilises response data without considering the impact of temperature. However, it has been well known that temperature might significantly influence the measurement. Thus, the robustness of the method from the impact of temperature can be further investigated for the future work.

Finally, it is possible to enhance the performance of the MPCA-CNN method by utilising larger hyperparameters space in the optimisation process. In this research, only the impact of model depth, number of data points in a sequence, dropout probability, and number of fully connection layers in the regressor are investigated. Other hyperparameters such as filter number, filter size, activation function applied in each layer, batch size, learning rate, node size, are not included in the optimisation. This is caused by significant increase in the processing time when adding these hyperparameters for the optimisation. By applying efficient algorithm for hyperparameter optimisation, potential problem related to time and computing resources can be avoided.

Bibliography

[1] J. M. . Brownjohn, “Structural health monitoring of civil infrastructure,” *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 365, no. 1851, pp. 589–622, Feb. 2007.

[2] Y. L. Xu and Y. Xia, *Structural health monitoring of long-span suspension bridges*. CRC Press, 2011.

[3] D. J. Thomson, “The economic case for service life extension of structures using structural health monitoring based on the delayed cost of borrowing,” *J. Civ. Struct. Heal. Monit.*, vol. 3, no. 4, pp. 335–340, 2013.

[4] F. J. Carrión, J. A. Quintana, and S. E. Crespo, “Techno-economical and practical considerations for SHM systems,” *J. Civ. Struct. Heal. Monit.*, vol. 7, no. 2, pp. 207–215, 2017.

[5] C. C. Comisu, N. Taranu, G. Boaca, and M. C. Scutaru, “Structural health monitoring system of bridges,” *Procedia Eng.*, vol. 199, pp. 2054–2059, 2017.

[6] G. W. Housner *et al.*, “Structural Control: Past, Present, and Future,” *J. Eng. Mech.*, vol. 123, no. 9, pp. 897–971, 1997.

[7] B. A. E. Aktan, N. F. David, L. B. Vikram, A. J. Helmicki, V. J. Hunt, and S. J. Shelley, “J. j. j.,” vol. 2, no. 3, pp. 108–117, 1996.

[8] C. R. Farrar and K. Worden, “An introduction to structural health monitoring,” *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 365, no. 1851, pp. 303–315, Feb. 2007.

[9] M. Radzieński, Ł. Doliński, M. Krawczuk, and M. Palacz, “Damage localisation in a stiffened plate structure using a propagating wave,” *Mech. Syst. Signal Process.*, vol. 39, no. 1–2, pp. 388–395, 2013.

[10] A. Rytter, “Vibrational based inspection of civil engineering structures.” Dept. of Building Technology and Structural Engineering, Aalborg University, 1993.

[11] K. F. Alvin, A. N. Robertson, G. W. Reich, and K. C. Park, "Structural system identification: From reality to models," *Comput. Struct.*, vol. 81, no. 12, pp. 1149–1176, 2003.

[12] T.-H. Yi, H.-N. Li, and M. Gu, "Optimal sensor placement for structural health monitoring based on multiple optimization strategies," *Struct. Des. Tall Spec. Build.*, vol. 20, no. 7, pp. 881–900, Nov. 2011.

[13] H. Sohn *et al.*, "A review of structural health monitoring literature: 1996–2001," *Los Alamos Natl. Lab. USA*, vol. 1, 2003.

[14] H. Li *et al.*, "Structural health monitoring system for the Shandong Binzhou Yellow River highway bridge," *Comput. Civ. Infrastruct. Eng.*, vol. 21, no. 4, pp. 306–317, 2006.

[15] D. M. Siringoringo and Y. Fujino, "Observed dynamic performance of the Yokohama-Bay Bridge from system identification using seismic records," *Struct. Control Heal. Monit.*, vol. 13, no. 1, pp. 226–244, 2006.

[16] S. Kim *et al.*, "Health Monitoring of Civil Infrastructures Using Wireless Sensor Networks," pp. 254–263, 2007.

[17] K. Y. Koo, J. M. W. Brownjohn, D. I. List, and R. Cole, "Structural health monitoring of the Tamar suspension bridge," *Struct. Control Heal. Monit.*, vol. 20, no. 4, pp. 609–625, 2013.

[18] Y. L. Xu, B. Chen, C. L. Ng, K. Y. Wong, and W. Y. Chan, "Monitoring temperature effect on a long suspension bridge," *Struct. Control Heal. Monit.*, no. May 2011, p. n/a-n/a, 2009.

[19] Z. Sun, Z. Zou, and Y. Zhang, "Utilization of structural health monitoring in long-span bridges: Case studies," *Struct. Control Heal. Monit.*, vol. 24, no. 10, 2017.

[20] R. Zaurin, T. Khuc, and F. N. Catbas, "Hybrid Sensor-Camera Monitoring for Damage Detection: Case Study of a Real Bridge," *J. Bridg. Eng.*, vol. 21, no. 6, p. 05016002, 2016.

[21] C. Cappello, D. Zonta, H. A. Laasri, B. Glisic, and M. Wang, "Calibration of

Elasto-Magnetic Sensors on In-Service Cable-Stayed Bridges for Stress Monitoring," *Sensors*, vol. 18, no. 2, p. 466, Feb. 2018.

[22] M. Takeshi and N. Masatsugu, "Vibration-based Structural Health Monitoring for Bridges using Laser Doppler Vibrometers and MEMS-based Technologies," *Steel Struct.*, vol. 8, no. 2008, pp. 325–331, 2008.

[23] J. M. W. Brownjohn, K.-Y. Koo, A. Scullion, and D. List, "Operational deformations in long-span bridges," *Struct. Infrastruct. Eng.*, vol. 11, no. 4, pp. 556–574, Apr. 2015.

[24] Q. Zhu, Y. L. Xu, and X. Xiao, "Multiscale Modeling and Model Updating of a Cable-Stayed Bridge. I: Modeling and Influence Line Analysis," *J. Bridg. Eng.*, vol. 20, no. 10, p. 04014112, 2015.

[25] D. M. Siringoringo and Y. Fujino, "System identification applied to long-span cable-supported bridges using seismic records," *Earthq. Eng. Struct. Dyn.*, vol. 37, no. 3, pp. 361–386, Mar. 2008.

[26] T. Ganev, F. Yamazaki, H. Ishizaki, and M. Kitazawa, "Response analysis of the Higashi-Kobe bridge and surrounding soil in the 1995 Hyogoken-Nanbu earthquake," *Earthq. Eng. Struct. Dyn.*, vol. 27, no. 6, pp. 557–576, 1998.

[27] J. X. Mao, H. Wang, D. M. Feng, T. Y. Tao, and W. Z. Zheng, "Investigation of dynamic properties of long-span cable-stayed bridges based on one-year monitoring data under normal operating condition," *Struct. Control Heal. Monit.*, vol. 25, no. 5, pp. 1–19, 2018.

[28] S. Cho *et al.*, "Structural health monitoring of a cable-stayed bridge using wireless smart sensor technology: data analyses," *Smart Struct. Syst.*, vol. 6, no. 5_6, pp. 461–480, 2010.

[29] M. J. Chae, H. S. Yoo, J. Y. Kim, and M. Y. Cho, "Development of a wireless sensor network system for suspension bridge health monitoring," *Autom. Constr.*, vol. 21, no. 1, pp. 237–252, 2012.

[30] Y. Yu, J. Wang, X. Mao, H. Liu, and L. Zhou, "Design of a wireless multi-radio-frequency channels inspection system for bridges," *Int. J. Distrib. Sens. Networks*, vol. 2012, no. Figure 2, 2012.

[31] D. González-Aguilera, J. Gómez-Lahoz, and J. Sánchez, "A new approach for structural monitoring of large dams with a three-dimensional laser scanner," *Sensors*, vol. 8, no. 9, pp. 5866–5883, 2008.

[32] P. Milillo *et al.*, "Monitoring dam structural health from space: Insights from novel InSAR techniques and multi-parametric modeling applied to the Pertusillo dam Basilicata, Italy," *Int. J. Appl. Earth Obs. Geoinf.*, vol. 52, pp. 221–229, 2016.

[33] S. Beskhyroun, L. D. Wegner, and B. F. Sparling, "Integral resonant control scheme for cancelling human-induced vibrations in light-weight pedestrian structures," *Struct. Control Heal. Monit.*, no. May 2011, p. n/a-n/a, 2011.

[34] G. Sevieri, M. Andreini, A. De Falco, and H. G. Matthies, "Concrete gravity dams model parameters updating using static measurements," *Eng. Struct.*, vol. 196, no. May, p. 109231, 2019.

[35] F. Kang, J. Liu, J. Li, and S. Li, "Concrete dam deformation prediction model for health monitoring based on extreme learning machine," *Struct. Control Heal. Monit.*, vol. 24, no. 10, pp. 1–11, 2017.

[36] G. Sevieri and A. De Falco, "Dynamic structural health monitoring for concrete gravity dams based on the Bayesian inference," *J. Civ. Struct. Heal. Monit.*, vol. 10, no. 2, pp. 235–250, 2020.

[37] G. Bossi, L. Schenato, and G. Marcato, "Structural health monitoring of a road tunnel intersecting a large and active landslide," *Appl. Sci.*, vol. 7, no. 12, 2017.

[38] X. Ye, Y. Ni, and J. Yin, "Safety monitoring of railway tunnel construction using FBG sensing technology," *Adv. Struct. Eng.*, vol. 16, no. 8, pp. 1401–1409, 2013.

[39] P. J. Bennett, Y. Kobayashi, K. Soga, and P. Wright, "Wireless sensor network for monitoring transport tunnels," *Proc. Inst. Civ. Eng. Geotech. Eng.*, vol. 163, no. 3, pp. 147–156, 2010.

[40] H. Mohamad, P. J. Bennett, K. Soga, R. J. Mair, and K. Bowers, "Behaviour of an old masonry tunnel due to tunnelling-induced ground settlement,"

Geotechnique, vol. 60, no. 12, pp. 927–938, 2010.

[41] L. L. K. Cheung, K. Soga, P. J. Bennett, Y. Kobayashi, B. Amatya, and P. Wright, "Optical fibre strain measurement for tunnel lining monitoring," *Proc. Inst. Civ. Eng. Geotech. Eng.*, vol. 163, no. 3, pp. 119–130, 2010.

[42] P. Reynolds, A. Pavic, and Z. Ibrahim, "A remote monitoring system for stadia dynamics," *Proc. Inst. Civ. Eng. Struct. Build.*, vol. 157, no. 6, pp. 385–393, 2004.

[43] A. Cigada, G. Moschioni, M. Vanali, and A. Caprioli, "The measurement network of the san siro meazza stadium in milan: Origin and implementation of a new data acquisition strategy for structural health monitoring: Dynamic testing of civil engineering structures series," *Exp. Tech.*, vol. 34, no. 1, pp. 70–81, 2010.

[44] A. Datteo, F. Lucà, and G. Busca, "Statistical pattern recognition approach for long-time monitoring of the G.Meazza stadium by means of AR models and PCA," *Eng. Struct.*, vol. 153, no. October, pp. 317–333, 2017.

[45] H. N. Li, L. Ren, Z. G. Jia, T. H. Yi, and D. S. Li, "State-of-the-art in structural health monitoring of large and complex civil infrastructures," *J. Civ. Struct. Heal. Monit.*, vol. 6, no. 1, pp. 3–16, 2016.

[46] D. Phanish *et al.*, "A wireless sensor network for monitoring the structural health of a football stadium," *IEEE World Forum Internet Things, WF-IoT 2015 - Proc.*, no. 3, pp. 471–477, 2015.

[47] S. DelloRusso, G. Juneja, B. Gabby, and D. Dusenberry, "Monitoring and Repair of the Milwaukee City Hall Masonry Tower," *J. Perform. Constr. Facil.*, vol. 22, no. 4, pp. 197–206, 2008.

[48] D. Bindi, I. Iervolino, and S. Parolai, "On-site structure-specific real-time risk assessment: perspectives from the REAKT project," *Bull. Earthq. Eng.*, vol. 14, no. 9, pp. 2471–2493, 2016.

[49] F. Ubertini, N. Cavalagli, A. Kita, and G. Comanducci, "Assessment of a monumental masonry bell-tower after 2016 central Italy seismic sequence by long-term SHM," *Bull. Earthq. Eng.*, vol. 16, no. 2, pp. 775–801, 2018.

[50] A. Saisi, C. Gentile, and A. Ruccolo, “Continuous monitoring of a challenging heritage tower in Monza, Italy,” *J. Civ. Struct. Heal. Monit.*, vol. 8, no. 1, pp. 77–90, 2018.

[51] C. Gentile, M. Guidobaldi, and A. Saisi, “One-year dynamic monitoring of a historic tower: damage detection under changing environment,” *Meccanica*, vol. 51, no. 11, pp. 2873–2889, 2016.

[52] Y. Q. Ni, Y. Xia, W. Lin, W. H. Chen, and J. M. Ko, “SHM benchmark for high-rise structures: A reduced-order finite element model and field measurement data,” *Smart Struct. Syst.*, vol. 10, no. 4, pp. 411–426, 2012.

[53] A. Cabboi, C. Gentile, and A. Saisi, “From continuous vibration monitoring to FEM-based damage assessment: Application on a stone-masonry tower,” *Constr. Build. Mater.*, vol. 156, pp. 252–265, 2017.

[54] D. Goyal and B. S. Pabla, “Development of non-contact structural health monitoring system for machine tools,” *J. Appl. Res. Technol.*, vol. 14, no. 4, pp. 245–258, 2016.

[55] M. Soualhi, K. T. P. Nguyen, A. Soualhi, K. Medjaher, and K. E. Hemsas, “Health monitoring of bearing and gear faults by using a new health indicator extracted from current signals,” *Meas. J. Int. Meas. Confed.*, vol. 141, pp. 37–51, 2019.

[56] X. Zhang, R. Xu, C. Kwan, S. Y. Liang, Q. Xie, and L. Haynes, “An integrated approach to bearing fault diagnostics and prognostics,” *Proc. Am. Control Conf.*, vol. 4, pp. 2750–2755, 2005.

[57] A. A. Cavalini, L. Sanches, N. Bachschmid, and V. Steffen, “Crack identification for rotating machines based on a nonlinear approach,” *Mech. Syst. Signal Process.*, vol. 79, pp. 72–85, 2016.

[58] A. A. Cavalini, R. M. Finzi Neto, and V. Steffen, “Impedance-based fault detection methodology for rotating machines,” *Struct. Heal. Monit.*, vol. 14, no. 3, pp. 228–240, 2015.

[59] N. Baydar and A. Ball, “Detection of gear failures via vibration and acoustic signals using wavelet transform,” *Mech. Syst. Signal Process.*, vol. 17, no. 4,

pp. 787–804, 2003.

[60] D. S. Rabelo, K. M. Tsuruta, D. D. de Oliveira, A. A. Cavalini, R. M. F. Neto, and V. Steffen, "Fault Detection of a Rotating Shaft by Using the Electromechanical Impedance Method and a Temperature Compensation Approach," *J. Nondestruct. Eval.*, vol. 36, no. 2, pp. 1–13, 2017.

[61] J. B. Ihn and F. K. Chang, "Pitch-catch active sensing methods in structural health monitoring for aircraft structures," *Struct. Heal. Monit.*, vol. 7, no. 1, pp. 5–19, 2008.

[62] X. Zhao *et al.*, "Active health monitoring of an aircraft wing with an embedded piezoelectric sensor/actuator network: II. Wireless approaches," *Smart Mater. Struct.*, vol. 16, no. 4, pp. 1218–1225, 2007.

[63] A. Baker, N. Rajic, and C. Davis, "Towards a practical structural health monitoring technology for patched cracks in aircraft structure," *Compos. Part A Appl. Sci. Manuf.*, vol. 40, no. 9, pp. 1340–1352, 2009.

[64] X. Qing, W. Li, Y. Wang, and H. Sun, "Piezoelectric transducer-based structural health monitoring for aircraft applications," *Sensors (Switzerland)*, vol. 19, no. 3, pp. 1–27, 2019.

[65] H. Chan, B. Masserey, and P. Fromme, "High frequency guided ultrasonic waves for hidden fatigue crack growth monitoring in multi-layer model aerospace structures," *Smart Mater. Struct.*, vol. 24, no. 2, 2015.

[66] N. Takeda, "Fiber optic sensor-based SHM technologies for aerospace applications in Japan," *Smart Sens. Phenomena, Technol. Networks, Syst. 2008*, vol. 6933, no. March 2008, p. 693302, 2008.

[67] W. C. Wilson *et al.*, "Orthogonal Frequency Coded SAW Sensors for Aerospace SHM Applications," *IEEE Sens. J.*, vol. 9, no. 11, pp. 1546–1556, 2009.

[68] G. C. Kahandawa, J. Epaarachchi, H. Wang, and K. T. Lau, "Use of FBG sensors for SHM in aerospace structures," *Photonic Sensors*, vol. 2, no. 3, pp. 203–214, 2012.

[69] A. Panopoulou, T. Loutas, D. Roulias, S. Fransen, and V. Kostopoulos, “Dynamic fiber Bragg gratings based health monitoring system of composite aerospace structures,” *Acta Astronaut.*, vol. 69, no. 7–8, pp. 445–457, 2011.

[70] I. Abdel-Qader, S. Pashaie-Rad, O. Abudayyeh, and S. Yehia, “PCA-Based algorithm for unsupervised bridge crack detection,” *Adv. Eng. Softw.*, vol. 37, no. 12, pp. 771–778, 2006.

[71] Z. Liu, S. A. Suandi, T. Ohashi, and T. Ejima, “Tunnel crack detection and classification system based on image processing,” in *Machine Vision Applications in Industrial Inspection X*, 2002, vol. 4664, pp. 145–152.

[72] I. Abdel-Qader, O. Abudayyeh, and M. E. Kelly, “Analysis of edge-detection techniques for crack identification in bridges,” *J. Comput. Civ. Eng.*, vol. 17, no. 4, pp. 255–263, 2003.

[73] Y. Hu, C. X. Zhao, and H. N. Wang, “Automatic pavement crack detection using texture and shape descriptors,” *IETE Tech. Rev. (Institution Electron. Telecommun. Eng. India)*, vol. 27, no. 5, pp. 398–405, 2010.

[74] M. Q. Feng, Y. Fukuda, D. Feng, and M. Mizuta, “Nontarget vision sensor for remote measurement of bridge dynamic response,” *J. Bridg. Eng.*, vol. 20, no. 12, p. 4015023, 2015.

[75] D. Lydon *et al.*, “Development and field testing of a time-synchronized system for multi-point displacement calculation using low-cost wireless vision-based sensors,” *IEEE Sens. J.*, vol. 18, no. 23, pp. 9744–9754, 2018.

[76] D. Feng, T. Scarangello, M. Q. Feng, and Q. Ye, “Cable tension force estimate using novel noncontact vision-based sensor,” *Meas. J. Int. Meas. Confed.*, vol. 99, pp. 44–52, 2017.

[77] D. Feng and M. Q. Feng, “Model updating of railway bridge using in situ dynamic displacement measurement under trainloads,” *J. Bridg. Eng.*, vol. 20, no. 12, p. 4015019, 2015.

[78] Y. Xu, J. Brownjohn, and D. Kong, “A non-contact vision-based system for multipoint displacement monitoring in a cable-stayed footbridge,” *Struct. Control Heal. Monit.*, vol. 25, no. 5, pp. 1–23, 2018.

[79] W. Wang, J. E. Mottershead, and C. Mares, “Vibration mode shape recognition using image processing,” *J. Sound Vib.*, vol. 326, no. 3–5, pp. 909–938, 2009.

[80] D. Feng and M. Q. Feng, “Vision-based multipoint displacement measurement for structural health monitoring,” *Struct. Control Heal. Monit.*, vol. 23, no. 5, pp. 876–890, 2016.

[81] J. G. Chen, N. Wadhwa, Y.-J. Cha, F. Durand, W. T. Freeman, and O. Buyukozturk, “Modal identification of simple structures with high-speed video using motion magnification,” *J. Sound Vib.*, vol. 345, pp. 58–71, 2015.

[82] Y.-Z. Song, C. R. Bowen, A. H. Kim, A. Nassehi, J. Padget, and N. Gathercole, “Virtual visual sensors and their application in structural health monitoring,” *Struct. Heal. Monit.*, vol. 13, no. 3, pp. 251–264, 2014.

[83] Z. Dworakowski, P. Kohut, A. Gallina, K. Holak, and T. Uhl, “Vision-based algorithms for damage detection and localization in structural health monitoring,” *Struct. Control Heal. Monit.*, vol. 23, no. 1, pp. 35–50, 2016.

[84] I. Laory, T. N. Trinh, D. Posenato, and I. F. C. Smith, “Combined Model-Free Data-Interpretation Methodologies for Damage Detection during Continuous Monitoring of Structures,” *J. Comput. Civ. Eng.*, vol. 27, no. 6, pp. 657–666, Nov. 2013.

[85] K. Worden and G. Manson, “The application of machine learning to structural health monitoring,” *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 365, no. 1851, pp. 515–537, 2007.

[86] F. N. Çatbaş, T. Kijewski-Correa, and A. E. Aktan, “Structural identification of constructed systems: approaches, methods, and technologies for effective practice of St-Id,” 2013.

[87] M. I. Friswell, “Damage identification using inverse methods,” *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 365, no. 1851, pp. 393–410, 2007.

[88] R. Kromanis and P. Kripakaran, “SHM of bridges: characterising thermal response and detecting anomaly events using a temperature-based measurement interpretation approach,” *J. Civ. Struct. Heal. Monit.*, vol. 6, no.

2, pp. 237–254, Apr. 2016.

[89] M. Friswell and J. E. Mottershead, *Finite element model updating in structural dynamics*, vol. 38. Springer Science & Business Media, 2013.

[90] J. M. W. Brownjohn, A. De Stefano, Y.-L. Xu, H. Wenzel, and A. E. Aktan, "Vibration-based monitoring of civil infrastructure: challenges and successes," *J. Civ. Struct. Heal. Monit.*, vol. 1, no. 3–4, pp. 79–95, Dec. 2011.

[91] D. Giagopoulos, A. Arailopoulos, V. Dertimanis, C. Papadimitriou, E. Chatzi, and K. Grompanopoulos, "Structural health monitoring and fatigue damage estimation using vibration measurements and finite element model updating," *Struct. Heal. Monit.*, vol. 18, no. 4, pp. 1189–1206, 2019.

[92] F. P. Kopsaftopoulos and S. D. Fassois, "A functional model based statistical time series method for vibration based damage detection, localization, and magnitude estimation," *Mech. Syst. Signal Process.*, vol. 39, no. 1–2, pp. 143–161, 2013.

[93] E. Z. Moore, J. M. Nichols, and K. D. Murphy, "Model-based SHM: Demonstration of identification of a crack in a thin plate using free vibration data," *Mech. Syst. Signal Process.*, vol. 29, pp. 284–295, 2012.

[94] T. Khuc and F. N. Catbas, "Structural Identification Using Computer Vision–Based Bridge Health Monitoring," *J. Struct. Eng.*, vol. 144, no. 2, p. 04017202, 2018.

[95] F. N. Catbas, R. Zaurin, M. Gul, and H. B. Gokce, "Sensor Networks, Computer Imaging, and Unit Influence Lines for Structural Health Monitoring: Case Study for Bridge Load Rating," *J. Bridg. Eng.*, vol. 17, no. 4, pp. 662–670, 2012.

[96] M. I. Friswell, J. E. T. Penny, and S. D. Garvey, "Parameter subset selection in damage location," *Inverse Probl. Eng.*, vol. 5, no. 3, pp. 189–215, 1997.

[97] Y. Zhu, Y. Q. Ni, H. Jin, D. Inaudi, and I. Laory, "A temperature-driven MPCA method for structural anomaly detection," *Eng. Struct.*, vol. 190, no. April, pp. 447–458, 2019.

[98] I. Laory, T. N. Trinh, and I. F. C. Smith, “Evaluating two model-free data interpretation methods for measurements that are influenced by temperature,” *Adv. Eng. Informatics*, vol. 25, no. 3, pp. 495–506, 2011.

[99] G. Zhang, L. Tang, L. Zhou, Z. Liu, Y. Liu, and Z. Jiang, “Principal component analysis method with space and time windows for damage detection,” *Sensors (Switzerland)*, vol. 19, no. 11, 2019.

[100] J. Shlens, “A Tutorial on Principal Component Analysis,” 2014.

[101] M. M. Suarez-Alvarez, D. T. Pham, M. Y. Prostov, and Y. I. Prostov, “Statistical approach to normalization of feature vectors and clustering of mixed datasets,” *Proc. R. Soc. A Math. Phys. Eng. Sci.*, vol. 468, no. 2145, pp. 2630–2651, 2012.

[102] Y. Q. Ni, X. T. Zhou, and J. M. Ko, “Experimental investigation of seismic damage identification using PCA-compressed frequency response functions and neural networks,” *J. Sound Vib.*, vol. 290, no. 1–2, pp. 242–263, 2006.

[103] R. Kromanis and P. Kripakaran, “Predicting thermal response of bridges using regression models derived from measurement histories,” *Comput. Struct.*, vol. 136, pp. 64–77, 2014.

[104] J. X. Mao, H. Wang, Y. G. Fu, and B. F. Spencer, “Automated modal identification using principal component and cluster analysis: Application to a long-span cable-stayed bridge,” *Struct. Control Heal. Monit.*, vol. 26, no. 10, pp. 1–20, 2019.

[105] S. Park, J. J. Lee, C. B. Yun, and D. J. Inman, “Electro-mechanical impedance-based wireless structural health monitoring using PCA-data compression and k-means clustering algorithms,” *J. Intell. Mater. Syst. Struct.*, vol. 19, no. 4, pp. 509–520, 2008.

[106] F. Magalhães, A. Cunha, and E. Caetano, “Vibration based structural health monitoring of an arch bridge: From automated OMA to damage detection,” *Mech. Syst. Signal Process.*, vol. 28, pp. 212–228, 2012.

[107] W. H. Hu, C. Moutinho, E. Caetano, F. Magalhes, and L. Cunha, “Continuous dynamic monitoring of a lively footbridge for serviceability assessment and

damage detection," *Mech. Syst. Signal Process.*, vol. 33, pp. 38–55, 2012.

[108] R. S. Concepcion and L. C. Ilagan, "Application of Hybrid Soft Computing for Classification of Reinforced Concrete Bridge Structural Health Based on Thermal-Vibration Intelligent System Parameters," *Proc. - 2019 IEEE 15th Int. Colloq. Signal Process. its Appl. CSPA 2019*, no. March, pp. 207–212, 2019.

[109] G. Comanducci, F. Ubertini, and A. L. Materazzi, "Structural health monitoring of suspension bridges with features affected by changing wind speed," *J. Wind Eng. Ind. Aerodyn.*, vol. 141, pp. 12–26, 2015.

[110] P. Van Overschee and B. L. De Moor, *Subspace identification for linear systems: Theory—Implementation—Applications*. Springer Science & Business Media, 2012.

[111] H. N. Ho, K. D. Kim, Y. S. Park, and J. J. Lee, "An efficient image-based damage detection for cable surface in cable-stayed bridges," *NDT E Int.*, vol. 58, pp. 18–23, 2013.

[112] E. J. Cross, K. Y. Koo, J. M. W. Brownjohn, and K. Worden, "Long-term monitoring and data analysis of the Tamar Bridge," *Mech. Syst. Signal Process.*, vol. 35, no. 1–2, pp. 16–34, 2013.

[113] D. Posenato, F. Lanata, D. Inaudi, and I. F. C. Smith, "Model-free data interpretation for continuous monitoring of complex structures," *Adv. Eng. Informatics*, vol. 22, no. 1, pp. 135–144, 2008.

[114] F. Cavadas, I. F. C. Smith, and J. Figueiras, "Damage detection using data-driven methods applied to moving-load responses," *Mech. Syst. Signal Process.*, vol. 39, no. 1–2, pp. 409–425, 2013.

[115] M. Malekzadeh, M. Gul, I. B. Kwon, and N. Catbas, "An integrated approach for structural health monitoring using an in-house built fiber optic system and non-parametric data analysis," *Smart Struct. Syst.*, vol. 14, no. 5, pp. 917–942, 2014.

[116] I. Laory, N. B. Hadj Ali, T. N. Trinh, and I. F. C. Smith, "Measurement System Configuration for Damage Identification of Continuously Monitored

Structures," *J. Bridg. Eng.*, vol. 17, no. 6, pp. 857–866, 2012.

[117] Y. L. Pavlov, "Random forests," *Random For.*, pp. 1–122, 2019.

[118] P. Probst, M. N. Wright, and A. L. Boulesteix, "Hyperparameters and tuning strategies for random forest," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 9, no. 3, pp. 1–19, 2019.

[119] I. Laory, T. N. Trinh, I. F. C. Smith, and J. M. W. Brownjohn, "Methodologies for predicting natural frequency variation of a suspension bridge," *Eng. Struct.*, vol. 80, pp. 211–221, 2014.

[120] D. Xu *et al.*, "Using statistical models and machine learning techniques to process big data from the forth road bridge," *Int. Conf. Smart Infrastruct. Constr. 2019, ICSIC 2019 Driv. Data-Informed Decis.*, vol. 2019, pp. 411–419, 2019.

[121] D. Huang, D. Hu, J. He, and Y. Xiong, "Structure Damage Detection Based on Ensemble Learning," *Proc. 2018 9th Int. Conf. Mech. Aerosp. Eng. ICMAE 2018*, pp. 219–224, 2018.

[122] X. Lei, L. Sun, Y. Xia, and T. He, "Vibration-based seismic damage states evaluation for regional concrete beam bridges using random forest method," *Sustain.*, vol. 12, no. 12, 2020.

[123] F. Wang and G. Song, "Monitoring of multi-bolt connection looseness using a novel vibro-acoustic method," *Nonlinear Dyn.*, vol. 100, no. 1, pp. 243–254, 2020.

[124] B. B. Van Der Horst, R. C. Lindenbergh, and S. W. J. Puister, "Mobile laser scan data for road surface damage detection," *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. - ISPRS Arch.*, vol. 42, no. 2/W13, pp. 1141–1148, 2019.

[125] Q. Zhou, H. Zhou, Q. Zhou, F. Yang, and L. Luo, "Structure damage detection based on random forest recursive feature elimination," *Mech. Syst. Signal Process.*, vol. 46, no. 1, pp. 82–90, 2014.

[126] Z. Noshad *et al.*, "Fault detection in wireless sensor networks through the random forest classifier," *Sensors (Switzerland)*, vol. 19, no. 7, pp. 1–21,

2019.

[127] I. . Basheer and M. Hajmeer, “Artificial neural networks: fundamentals, computing, design, and application,” *J. Microbiol. Methods*, vol. 43, no. 1, pp. 3–31, Dec. 2000.

[128] N. Aloysius and M. Geetha, “A review on deep convolutional neural networks,” *Proc. 2017 IEEE Int. Conf. Commun. Signal Process. ICCSP 2017*, vol. 2018-Janua, no. November 2020, pp. 588–592, 2018.

[129] A. H. Namin, K. Leboeuf, R. Muscedere, H. Wu, and M. Ahmadi, “Efficient hardware implementation of the hyperbolic tangent sigmoid function,” *Proc. - IEEE Int. Symp. Circuits Syst.*, no. June, pp. 2117–2120, 2009.

[130] H. F. Lam, K. V. Yuen, and J. L. Beck, “Structural health monitoring via measured ritz vectors utilizing artificial neural networks,” *Comput. Civ. Infrastruct. Eng.*, vol. 21, no. 4, pp. 232–241, 2006.

[131] Z. X. Li and X. M. Yang, “Damage identification for beams using ANN based on statistical property of structural responses,” *Comput. Struct.*, vol. 86, no. 1–2, pp. 64–71, 2008.

[132] R. P. Finotti, A. A. Cury, and F. de S. Barbosa, “An SHM approach using machine learning and statistical indicators extracted from raw dynamic measurements,” *Lat. Am. J. Solids Struct.*, vol. 16, no. 2, pp. 1–17, 2019.

[133] X. Xie, J. Guo, H. Zhang, T. Jiang, R. Bie, and Y. Sun, “Neural-network based structural health monitoring with wireless sensor networks,” *Proc. - Int. Conf. Nat. Comput.*, no. 61171014, pp. 163–167, 2013.

[134] C. A. Jeyasehar and K. Sumangala, “Damage assessment of prestressed concrete beams using artificial neural network (ANN) approach,” *Comput. Struct.*, vol. 84, no. 26–27, pp. 1709–1718, 2006.

[135] A. C. Neves, I. González, J. Leander, and R. Karoumi, “Structural health monitoring of bridges: a model-free ANN-based approach to damage detection,” *J. Civ. Struct. Heal. Monit.*, vol. 7, no. 5, pp. 689–702, 2017.

[136] A. A. Mousavi, C. Zhang, S. F. Masri, and G. Gholipour, “Structural damage

localization and quantification based on a CEEMDAN hilbert transform neural network approach: A model steel truss bridge case study," *Sensors (Switzerland)*, vol. 20, no. 5, 2020.

[137] M. E. Torres, M. A. Colominas, G. Schlotthauer, and P. Flandrin, "A complete ensemble empirical mode decomposition with adaptive noise," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 4144–4147.

[138] N. E. Huang *et al.*, "The empirical mode decomposition and the Hubert spectrum for nonlinear and non-stationary time series analysis," *Proc. R. Soc. A Math. Phys. Eng. Sci.*, vol. 454, no. 1971, pp. 903–995, 1998.

[139] H. F. Zhou, Y. Q. Ni, and J. M. Ko, "Constructing input to neural networks for modeling temperature-caused modal variability: Mean temperatures, effective temperatures, and principal components of temperatures," *Eng. Struct.*, vol. 32, no. 6, pp. 1747–1759, 2010.

[140] R. Zhang, Y. Duan, Y. Zhao, and X. He, "Temperature Compensation of Elasto-Magneto-Electric (EME) Sensors in Cable Force Monitoring Using BP Neural Network," *Sensors*, vol. 18, no. 7, p. 2176, Jul. 2018.

[141] J. C. M. Pires, F. G. Martins, S. I. V Sousa, M. C. M. Alvim-Ferraz, and M. C. Pereira, "Selection and validation of parameters in multiple linear and principal component regressions," *Environ. Model. Softw.*, vol. 23, no. 1, pp. 50–55, 2008.

[142] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[143] H. Sohn, M. Dzwonczyk, E. G. Straser, A. S. Kiremidjian, K. H. Law, and T. Meng, "An experimental study of temperature effect on modal parameters of the Alamosa Canyon Bridge," *Earthq. Eng. Struct. Dyn.*, vol. 28, no. 7–8, pp. 879–897, 1999.

[144] B. Peeters, J. Maeck, and G. De Roeck, "Vibration-based damage detection in civil engineering: Excitation sources and temperature effects," *Noise Vib. Worldw.*, vol. 35, no. 6, p. 33, 2004.

[145] Y. Deng, Y. L. Ding, and A. Q. Li, "Structural condition assessment of long-

span suspension bridges using long-term monitoring data," *Bridg. Maintenance, Safety, Manag. Life-Cycle Optim. - Proc. 5th Int. Conf. Bridg. Maintenance, Saf. Manag.*, vol. 9, no. 1, pp. 1993–2000, 2010.

[146] C. Liu and J. T. DeWolf, "Effect of temperature on modal variability for a curved concrete bridge," in *Smart Structures and Materials 2006: Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems*, 2006, vol. 6174, no. December, p. 61743B.

[147] Y. Ding and A. Li, "Assessment of bridge expansion joints using long-term displacement measurement under changing environmental conditions," *Front. Archit. Civ. Eng. China*, vol. 5, no. 3, pp. 374–380, 2011.

[148] Z. X. Li, T. H. T. Chan, and R. Zheng, "Statistical analysis of online strain response and its application in fatigue assessment of a long-span steel bridge," *Eng. Struct.*, vol. 25, no. 14, pp. 1731–1741, 2003.

[149] J. Mata, A. Tavares de Castro, and J. Sá da Costa, "Time-frequency analysis for concrete dam safety control: Correlation between the daily variation of structural response and air temperature," *Eng. Struct.*, vol. 48, pp. 658–665, 2013.

[150] R. A. Saeed, A. N. Galybin, and V. Popov, "Crack identification in curvilinear beams by using ANN and ANFIS based on natural frequencies and frequency response functions," *Neural Comput. Appl.*, vol. 21, no. 7, pp. 1629–1645, 2012.

[151] W. T. Yeung and J. W. Smith, "Damage detection in bridges using neural networks for pattern recognition of vibration signatures," *Eng. Struct.*, vol. 27, no. 5, pp. 685–698, 2005.

[152] S. J. S. Hakim and H. Abdul Razak, "Structural damage detection of steel bridge girder using artificial neural networks and finite element models," *Steel Compos. Struct.*, vol. 14, no. 4, pp. 367–377, 2013.

[153] P. Cornwell, C. R. Farrar, S. W. Doebling, and H. Sohn, "Environmental variability of modal properties," *Exp. Tech.*, vol. 23, no. 6, pp. 45–48, 1999.

[154] B. Peeters and G. De Roeck, "One-year monitoring of the Z24-Bridge:

environmental effects versus damage events," *Earthq. Eng. Struct. Dyn.*, vol. 30, no. 2, pp. 149–171, 2001.

[155] D. F. Giraldo, S. J. Dyke, and J. M. Caicedo, "Damage detection accommodating varying environmental conditions," *Struct. Heal. Monit.*, vol. 5, no. 2, pp. 155–172, 2006.

[156] Y. Xia, H. Hao, G. Zanardo, and A. Deeks, "Long term vibration monitoring of an RC slab: temperature and humidity effect," *Eng. Struct.*, vol. 28, no. 3, pp. 441–452, 2006.

[157] S. Soyoz and M. Q. Feng, "Long-term monitoring and identification of bridge structural parameters," *Comput. Civ. Infrastruct. Eng.*, vol. 24, no. 2, pp. 82–92, 2009.

[158] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[159] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[160] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, no. November 2016, pp. 11–26, 2017.

[161] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[162] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv Prepr. arXiv1409.1556*, 2014.

[163] C. Szegedy *et al.*, "Going deeper with convolutions," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[164] D. CireAan, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," *Neural networks*, vol. 32, pp. 333–338, 2012.

[165] N. Tajbakhsh *et al.*, “Convolutional neural networks for medical image analysis: Full training or fine tuning?,” *IEEE Trans. Med. Imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[166] H.-C. Shin *et al.*, “Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning,” *IEEE Trans. Med. Imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.

[167] Q. Li, W. Cai, X. Wang, Y. Zhou, D. D. Feng, and M. Chen, “Medical image classification with convolutional neural network,” in *2014 13th international conference on control automation robotics & vision (ICARCV)*, 2014, pp. 844–848.

[168] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708.

[169] O. M. Parkhi, A. Vedaldi, and A. Zisserman, “Deep face recognition,” 2015.

[170] K. Gopalakrishnan, S. K. Khaitan, A. Choudhary, and A. Agrawal, “Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection,” *Constr. Build. Mater.*, vol. 157, pp. 322–330, 2017.

[171] R. Fan *et al.*, “Road Crack Detection Using Deep Convolutional Neural Network and Adaptive Thresholding,” in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 474–479.

[172] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, “Road crack detection using deep convolutional neural network,” in *2016 IEEE international conference on image processing (ICIP)*, 2016, pp. 3708–3712.

[173] E. Protopapadakis *et al.*, “AUTONOMOUS ROBOTIC INSPECTION IN TUNNELS.,” *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. 3, no. 5, 2016.

[174] Y. Cha, W. Choi, and O. Büyüköztürk, “Deep learning-based crack damage detection using convolutional neural networks,” *Comput. Civ. Infrastruct. Eng.*, vol. 32, no. 5, pp. 361–378, 2017.

[175] B. Xu, N. Wang, T. Chen, and M. Li, “Empirical evaluation of rectified activations in convolutional network,” *arXiv Prepr. arXiv1505.00853*, 2015.

[176] D. Yu, H. Wang, P. Chen, and Z. Wei, “Mixed pooling for convolutional neural networks,” in *International conference on rough sets and knowledge technology*, 2014, pp. 364–375.

[177] O. Abdeljaber, O. Avci, S. Kiranyaz, M. Gabbouj, and D. J. Inman, “Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks,” *J. Sound Vib.*, vol. 388, pp. 154–170, 2017.

[178] Y. Lin, Z. Nie, and H. Ma, “Structural damage detection with automatic feature-extraction through deep learning,” *Comput. Civ. Infrastruct. Eng.*, vol. 32, no. 12, pp. 1025–1046, 2017.

[179] L. Ali, F. Alnajjar, H. Al Jassmi, M. Gochoo, W. Khan, and M. A. Serhani, “Performance evaluation of deep CNN-based crack detection and localization techniques for concrete structures,” *Sensors*, vol. 21, no. 5, pp. 1–22, 2021.

[180] B. Kim, N. Yuvaraj, K. R. Sri Preethaa, and R. Arun Pandian, “Surface crack detection using deep learning with shallow CNN architecture for enhanced computation,” *Neural Comput. Appl.*, vol. 8, 2021.

[181] L. Jing, M. Zhao, P. Li, and X. Xu, “A convolutional neural network based feature learning and fault diagnosis method for the condition monitoring of gearbox,” *Measurement*, vol. 111, pp. 1–10, 2017.

[182] W. Zhang, C. Li, G. Peng, Y. Chen, and Z. Zhang, “A deep convolutional neural network with new training methods for bearing fault diagnosis under noisy environment and different working load,” *Mech. Syst. Signal Process.*, vol. 100, pp. 439–453, 2018.

[183] O. Abdeljaber, O. Avci, M. S. Kiranyaz, B. Boashash, H. Sodano, and D. J. Inman, “1-D CNNs for structural damage detection: Verification on a structural health monitoring benchmark data,” *Neurocomputing*, vol. 275, pp. 1308–1317, 2018.

[184] A. Pamuncak, W. Guo, A. Soliman Khaled, and I. Laory, “Deep learning for bridge load capacity estimation in post-disaster and -conflict zones,” *R. Soc.*

Open Sci., vol. 6, no. 12, p. 190227, Dec. 2019.

[185] Y. Zhang, Y. Miyamori, S. Mikami, and T. Saito, "Vibration-based structural state identification by a 1-dimensional convolutional neural network," *Comput. Civ. Infrastruct. Eng.*, vol. 34, no. 9, pp. 822–839, Sep. 2019.

[186] D. H. Nguyen, Q. B. Nguyen, T. Bui-Tien, G. De Roeck, and M. Abdel Wahab, "Damage detection in girder bridges using modal curvatures gapped smoothing method and Convolutional Neural Network: Application to Bo Nghi bridge," *Theor. Appl. Fract. Mech.*, vol. 109, no. June, p. 102728, 2020.

[187] F. T. Ni, J. Zhang, and M. N. Noori, "Deep learning for data anomaly detection and data compression of a long-span suspension bridge," *Comput. Civ. Infrastruct. Eng.*, vol. 35, no. 7, pp. 685–700, 2020.

[188] A. A. of S. H. and T. Officials, *Manual for condition evaluation and load and resistance factor rating (LRFR) of highway bridges*. American Association of State Highway and Transportation Officials, 2003.

[189] G. Fu, *Effect of truck weight on bridge network costs*, vol. 37. Transportation Research Board, 2003.

[190] "National Bridge Inventory." Federal Highway Administration, 2016.

[191] J. Baughn, "Bridgehunter.com: Historic Bridges of the United States." [Online]. Available: https://bridgehunter.com/. [Accessed: 01-Dec-2017].

[192] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Adv. Eng. Informatics*, vol. 29, no. 2, pp. 196–210, 2015.

[193] "Recording and Coding Guide for the Structure Inventory and Appraisal of the Nation's Bridges," 1995.

[194] C. Kanan and G. W. Cottrell, "Color-to-grayscale: does the method matter in image recognition?," *PLoS One*, vol. 7, no. 1, p. e29740, 2012.

[195] S. Nazlibilek, D. Karacor, T. Ercan, M. H. Sazli, O. Kalender, and Y. Ege, "Automatic segmentation, counting, size determination and classification of

white blood cells," *Measurement*, vol. 55, pp. 58–65, 2014.

[196] M. Z. Rashad, B. S. El-Desouky, and M. S. Khawasik, "Plants images classification based on textural features using combined classifier," *Int. J. Comput. Sci. Inf. Technol.*, vol. 3, no. 4, pp. 93–100, 2011.

[197] N. Wang and D. Y. Yeung, "Learning a deep compact image representation for visual tracking," *Adv. Neural Inf. Process. Syst.*, 2013.

[198] R. I.-R. BT, "Studio encoding parameters of digital television for standard 4: 3 and wide-screen 16: 9 aspect ratios," 2011.

[199] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv Prepr. arXiv1405.3531*, 2014.

[200] B. Funt and L. Zhu, "Does colour really matter? Evaluation via object classification," in *Color and Imaging Conference*, 2018, vol. 2018, no. 1, pp. 268–271.

[201] J. McKee and E. Weber, "Effect of Image Classification Accuracy on Dasymetric Population Estimation," *Urban Remote Sens. Monit. Synth. Model. Urban Environ.*, pp. 283–304, 2021.

[202] G. Raskutti, M. J. Wainwright, and B. Yu, "Early stopping and non-parametric regression: an optimal data-dependent stopping rule," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 335–366, 2014.

[203] Texas Department of Transportation Bridge Division, "LRFD Bridge Design Manual," *Manual 5-392*, no. October. WSDOT Washington, DC, 2020.

[204] A. P. Pamuncak, M. R. Salami, A. Adha, B. Budiono, and I. Laory, "Estimation of structural response using convolutional neural network: application to the Suramadu bridge," *Eng. Comput.*, 2021.

[205] T. Jayalakshmi and A. Santhakumaran, "Statistical Normalization and Back Propagation for Classification," *Int. J. Comput. Theory Eng.*, vol. 3, no. 1, pp. 89–93, 2011.

[206] Y. Deng, Y. Liu, and S. Chen, "Long-term in-service monitoring and

performance assessment of the main cables of long-span suspension bridges," *Sensors (Switzerland)*, vol. 17, no. 6, 2017.

[207] S. Sumitro, A. Jarosevic, and M. L. Wang, "Elasto-magnetic sensor utilization on steel cable stress measurement," in *The First fib Congress, Concrete Structures in the 21th Century, Osaka*, 2002, pp. 13–19.

[208] M. Morgese, F. Ansari, M. Domaneschi, and G. P. Cimellaro, "Post-collapse analysis of Morandi's Polcevera viaduct in Genoa Italy," *J. Civ. Struct. Heal. Monit.*, vol. 10, no. 1, pp. 69–85, 2020.

[209] D. Zonta *et al.*, "Calibration of Elasto-Magnetic Sensors for Bridge-Stay Cable Monitoring," pp. 1–8.

[210] Y.-F. Duan *et al.*, "Development of elasto-magneto-electric (EME) sensor for in-service cable force monitoring," *Int. J. Struct. Stab. Dyn.*, vol. 16, no. 04, p. 1640016, 2016.

[211] Y. Cao, Y. Zhang, Y. Zhao, and M. L. Wang, "Distributed Health Monitoring System for Zhanjiang Bay Bridge," *Adv. Struct. Eng.*, vol. 14, no. 1, pp. 1–12, Feb. 2011.

[212] J. Yim *et al.*, "Field application of elasto-magnetic stress sensors for monitoring of cable tension force in cable-stayed bridges," *Smart Struct. Syst.*, vol. 12, no. 3_4, pp. 465–482, 2013.

[213] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," Feb. 2015.

[214] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[215] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: A big comparison for nas," *arXiv Prepr. arXiv1912.06059*, 2019.

[216] S. Živanović, R. P. Johnson, H. V Dang, and J. Dobrić, "Design and Construction of a Very Lively Bridge," in *Topics in Dynamics of Civil Structures, Volume 4*, Springer, 2013, pp. 371–380.

[217] "Post-yield Strain Gauge | Tokyo Measuring Instruments Laboratory Co., Ltd." [Online]. Available: https://tml.jp/e/product/strain_gauge/post-yield.html#yef_list.html. [Accessed: 30-Mar-2018].

[218] "PL-60-11-1LJC - Techni Measure Online." [Online]. Available: https://store.technimeasure.co.uk/product/pl-60-11-1l/. [Accessed: 30-Mar-2018].

[219] "DATASHEET NI 9235." National Instruments, p. 14, 2015.

[220] "SP1 Compact String Pot • Voltage Divider." Mouser Electronics, p. 5, 2015.

[221] "DATASHEET NI 9219." National Instruments, p. 17, 2016.

[222] "Linear Displacement Sensor." Vishay Precision Group, p. 2, 2005.

[223] "DATASHEET NI 9237." National Instruments, p. 15, 2015.

[224] "Q-FLEX QA-750 ACCELEROMETER." Honeywell, Redmond, Washington, p. 2, 2020.

[225] "Datasheet NI 9234." National Instruments, p. 13, 2015.

[226] R. P. C. Sampaio, N. M. M. Maia, and J. M. M. Silva, "Damage detection using the frequency-response-function curvature method," *J. Sound Vib.*, vol. 226, no. 5, pp. 1029–1042, 1999.

[227] A. A. Mosavi, R. Seracino, and S. Rizkalla, "Effect of Temperature on Daily Modal Variability of a Steel-Concrete Composite Bridge," *J. Bridg. Eng.*, vol. 17, no. 6, pp. 979–983, 2012.

[228] D. Posenato, P. Kripakaran, D. Inaudi, and I. F. C. Smith, "Methodologies for model-free data interpretation of civil engineering structures," *Comput. Struct.*, vol. 88, no. 7–8, pp. 467–482, 2010.

[229] H. Xu and Y. Deng, "Dependent evidence combination based on shearman coefficient and pearson coefficient," *IEEE Access*, vol. 6, pp. 11634–11640, 2017.

[230] G. E. Sakr, M. Mokbel, A. Darwich, M. N. Khneisser, and A. Hadi, "Comparing deep learning and support vector machines for autonomous waste

sorting," *2016 IEEE Int. Multidiscip. Conf. Eng. Technol. IMCET 2016*, pp. 207–212, 2016.

[231] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1026–1034.

[232] A. OrShea, G. Lightbody, G. Boylan, and A. Temko, "Investigating the Impact of CNN Depth on Neonatal Seizure Detection Performance," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2018, pp. 5862–5865.

[233] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *J. Mach. Learn. Res.*, vol. 9, pp. 249–256, 2010.

[234] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Networks*, vol. 5, no. 2, pp. 157–166, Mar. 1994.