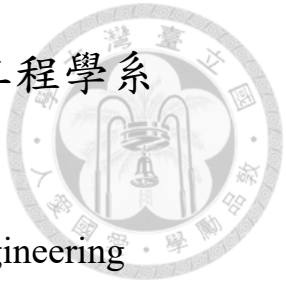


國立臺灣大學工學院工程科學及海洋工程學系

碩士論文

Department of Engineering Science and Ocean Engineering
College of Engineering
National Taiwan University

Master's Thesis

開發新型領域適應方法於深度學習與混合資料庫應用於複合式損傷結構健康監測系統

A Novel Technique for Domain Adaption in Hybrid Databases applied in Multi-Damage Structure Health Monitoring with DNNs

江昱翰

Yu-Han Chiang

指導教授：黃心豪 博士

Advisor: Hsin-Haou Huang, Ph.D.

中華民國 113 年 5 月

May, 2024

doi:10.6342/NTU202401548

Acknowledgements

Conducting research is not an easy task, often times it is a lonely and gloomy path with nothing that promises success. I would like to thank those who brought beams of sunlight into my life so that they made my path as a researcher so much more pleasant. Firstly, I want to thank my professor who's always there to give me new ideas and feedback when I feel stuck. Changing a research path is one of the most pivotal moves that helped me in my academic career and this thesis would never be here without you. Secondly, want to thank my family that always gives me the emotional stability that I needed and the motivation to finish the research. It is always a plus when you know there's good food that is waiting every night. Thirdly, I want to thank my girlfriend Chloe who is always there to cheer me up bring me joy outside of academics, every time I see you smile it makes me want to be a better and more accomplished person. I would also want to thank my friends in the lab who always brought great friendly competition to the environment and made me step up my research quality. Last but not least, I want to thank the chads in old gym, especially my friend Yasser Salum, Rolando Ruiz, Old Chen, Mr. Tungwei and Hank. Whenever I feel down and out, you guys can always provide guidance, wisdom and "chadness" into my life. I hope the fellowship of iron would never break and each of us can get all the "gainz" we want in our lives.

doi:10.6342/NTU202401548

摘要

為了降低例如離岸風電等在高度複雜操作環境下的的運營和維護成本並提高在能源穩定性，本研究提出了一種利用深度學習和大數據的力量來自動化和簡化監測過程的方法。本研究嘗試將傳統的結構健康監測技術、信號處理知識和深度學習技術相結合以克服這一挑戰。本研究試圖使用由真實世界數據和電腦模擬數據組成的混合資料庫訓練深度神經網絡（DNN），以解決在取得有限的真實數據下從而導致數據匹配度問題，並且取得良好的監測效果。本研究開發了一種基於卷積的創新技術以實現領域適應，增加模擬數據和真實世界數據之間的相似性，使 DNN 在有限數據領域中也能有良好的表現。卷積領域展開（CDE）的方法將不同來源的數據映射到新的領域供 DNN 分類，然後將分類後的數據重新映射回其原始的健康/損壞狀態。本研究應用了分散式的系統設計和多種深度學習技術來識別在無先驗知識下的未知損壞組合，並且與我們的 CDE 技術結合後有很好的領域適應效果。一個實驗室尺度的離岸風電塔架與水下機基礎被使用於採收數據且分析，最後應用開發的算法進行監測任務。本論文希望對大數據數據的土木結構監測研究提供貢獻，並且期待於未來可以用用於於風力發電機、電塔和電網等，提高能源的安全性和穩定性。

關鍵字： 離岸風電、深度學習、領域適應、結構健康監測、混合資料庫

doi:10.6342/NTU202401548

Abstract

In order to lower the Operation and Maintenance cost and increase energy stability on structures in complex conditions such as offshore wind turbines, this research presents a method which leverages the power of deep learning and big data to automize and simplify the process of monitoring. In this research, an attempt to merge traditional Structural Health Monitoring techniques, Signal Processing knowledge, and Deep Learning technology is used to overcome this challenge. This research attempts to train a DNN with the use of a hybrid database comprised of real-life and computer simulated data to achieve good monitoring results with limited usage of real-life data, which would lead us to a data mismatch problem. A novel convolution-based technique is developed to increase the similarity between the simulated data and real-life results to achieve domain adaption in order for the DNN to perform well on a limited data domain. The Convolution Domain Expansion (CDE) method maps different sources of data into new domains for the DNN to classify, and then re-maps the data back to its original healthy /damaged state. A decentralized design and multiple deep learning techniques are applied to identify unknown damage combinations without prior knowledge and is proven to have great results combined with our CDE technique. A lab scale structure of a foundation of an Offshore Wind Turbines is used for modelling and a monitoring task is performed with the developed algorithm. This paper hopes to contribute to data-centric civil structure monitoring research that could potentially be used on OWTs, electric towers and grids, and many more and help increasing the safety and stability of energy.

Keywords: Structural Health Monitoring (SHM), Deep Learning (DL), Domain Adaption, Offshore Wind Turbines (OWT), Hybrid Database

doi:10.6342/NTU202401548

Contents

doi:10.6342/NTU202401548

doi:10.6342/NTU202401548

doi:10.6342/NTU202401548

Content of Tables

doi:10.6342/NTU202401548

Content of Figures

doi:10.6342/NTU202401548

doi:10.6342/NTU202401548

doi:10.6342/NTU202401548

doi:10.6342/NTU202401548

doi:10.6342/NTU202401548

List of Abbreviations

A

AI / Artificial Intelligence

B

BP / Back Propagation

C

CV / Computer Vision

CNN / Convolutional Neural Network

CMS / Condition based Monitoring System

CRF / Conditional Random Field

D

DL / Deep Learning

DIC / Digital Image Correlation

DNN / Deep Neural Network

F

FEM / Finite Element Modeling

FFT / Fast Fourier Transform

FP / Front Propagation

FT / Fourier Transform

FIR / Finite Impulse Response

G

GPR / Gaussian Process Regression

GRU / Gated Recurrent Unit

H

doi:10.6342/NTU202401548

HL / Hidden Layer

L

LSTM / Long Short-Term Memory

M

ML / Machine Learning

N

NST / Neural Style Transfer

NN / Neural Network

NLP / Natural Language Processing

O

OD / Object Detection

OF / Optical Flow

OWT / Offshore wind turbine

R

ReLU / Rectified Linear Unit

S

SHM Structure Health Monitoring (Systems)

W

WT / Wind Turbine

doi:10.6342/NTU202401548

Nomenclature

α	Learning rate
λ	Penalty Term for regularization
$A^{[l]}$	Vectorized notation of unit in layer l
$a^{[l]}{}_n$	Unit in the lth layer
$b^{[l]}$	Vectorized notation of bias in layer l
$b^{[l]}{}_n$	Bias in the lth layer
C	Calculation Cost of CNNs
$dA^{[l]}$	Gradient of output of layer l
$db^{[l]}$	Gradient of bias of layer l
$dW^{[l]}$	Gradient of weights of layer l
dZ	Sum of error of all examples
$g^{[l]}(x)$	Activation function of layer l
l	Layer l
n	Kernel size
f	Filter size
f_{sample}	Sample Frequency
$f_{Nyquist}$	Nyquist Frequency
p	Padding size
$w^{[l]}{}_n$	Weight in the lth layer
$W^{[l]}$	Vectorized notation weights in layer l
$w^{[l]}{}_n$	Weight in the lth layer
$\hat{Y}$	Ground truth, correct label of Y
$z^{[l]}{}_n$	Inner factor in the lth layer
×	Scalar product or image size
·	Dot product of vectors
*	Convolution operator
θ	Cosine Similarity

doi:10.6342/NTU202401548

Chapter 1 Introduction

1.1 Research Motivation

At the 2023 United Nations Climate Change Conference (COP28), more than 150 heads of states and governments have agreed on the new year will be the ‘‘beginning of the end of fossil fuels.’’ There is a huge void that need to be filled because fossil fuels consist of 82% of the total primary energy consumption according to the 2023 Statistical Review of World Energy from Forbes. In the fragile state of the environment and energy sector, solar and wind energy has proven to be a reliable, resourceful, and safe replacement to fossil fuels, now reaching a 7.5% share of the primary energy consumption and is set to grow twice as much until 2050. Although solar energy coupled with wind energy are promising solutions, the intermittent nature of these sources is a major problem to be solved. Both of these infrastructures are set outdoors therefore they are exposed to high stress and continuous environmental loads and this increases the risk of unreliability, especially for wind turbines which produces more energy per unit. There is a lack of sophisticated technique in the commercial sector that can not only monitor the energy generated from the turbine, but also detect and predict the damage that years of environmental loads inflicted on the turbine over its lifetime. A better and more comprehensive and automatic monitoring technique can greatly benefit the O&M (Operation and Maintenance) process by potentially reducing manpower, ship rental cost and increase efficiency and safety of operators. This research aims to utilize the power of artificial intelligence to streamline the process of O&M by creating a fully automatic and deployable SHMS (Structure Health Monitoring System) that could potentially help the energy companies monitor their wind farm with greater efficiency.

doi:10.6342/NTU202401548

1.2 Research Background

With a lifespan of 25 years of an offshore wind turbine, O&M cost varies from 25% to 35 % of the total cost, which includes difficult and costly tasks such as regular maintenance and repairment[1]. The cost will potentially rise as the wind turbine accumulate longer service years and increase in complexity and generating power. It is now well established in the industry that increasing the size of the wind turbine and venturing to wind farms that are in deep waters and farther from shore have an economical advantage to lower the cost and increase the generated energy [2]. But this strategy also comes with risks such as increased environmental loads and harsher conditions [3], and this comes with generally two to three times more maintenance costs compared to onshore wind farms [4]. The increased difficulty stems from the increased accessibility of offshore wind farms which would result in an increase in downtime which results in lower profitability and money to hire maintenance fleets and skilled technicians. With all the increased challenges and opportunities, it becomes evident that a successful wind farm management would require proper O&M strategies that can maximize the profit of the companies, provide abundant energy output to the public, and minimize the risks of irregularity and environmental loads.

1.3 Research Objective and Contributions

Neural Networks are an extremely data hungry algorithm, but it is also why deep learning works extremely well on large complex problems that increases its difficulty with the size of the data. In the real-world structure health monitoring tasks, damage detection is not as easy as in laboratory conditions when the damaged conditions are well thought out by humans to perform the damage detection optimally. In the real life there

doi:10.6342/NTU202401548

are multiple magnitudes of damages and also various states of healthy conditions, which makes the fitting extremely complex and hard to conduct by human inspection. What makes this issue worse is that in real life scenarios, the damaged state of a structure is hard to come by, thus it makes it even harder to model the damaged scenarios with great complexity. With limited sources of data and for it to be split into the training, testing and development set, it would be hard for us to a database big enough so that we can leverage the power of deep learning without encountering common issues such as overfitting for our implementation.

A novel method to solve this issue is to use a hybrid database that have the potential to supplement the lack of real-world data so that we can have a bigger pool of data to train our neural network well. In the real-world scenario, we have unlimited access to real-life healthy data and it is the cheapest to acquire. The second source of data that we can use is from computer simulation, with which we could perform different types of analysis and output the same type of data to accompany our real-life healthy data. This source is also unlimited if we have enough computational power and resources. We could use this source to simulate damage that might happen to our structure but we don't have any resources at the moment, such as the effects of scouring that reduces boundary stiffness or erosion that can cause beam stiffness reduction with time. We could also use visual inspection to acquire data from consumer cameras of the real-life structure and mix it in with the hybrid database.

Just like any digital simulation or digital twin platform, the simulated data would need to be based on the real-life data and the differences needs to be as small as possible. We may model the damage that have a probability to happen, but because of the ideal circumstances of the computer simulation, the results might differ a lot from the real-life data. And because the type of damage that we simulate might be hard to acquire in real-

doi:10.6342/NTU202401548

life circumstances, it would be risky to extrapolate a fitting towards unknown circumstances for the current computational methods. Therefore, another method that could bridge the gap between the virtual simulations to the real-world conditions needs to be made so that the database could be more coherent and we would require a method to calculate the similarities after the transformation. Another problem that we may face in the hybrid database is the difference of sample rates. Because of the different data acquisition devices from different sources, we have to regulate the samples so that the inputs are coherent with each other.

The highlight of this research covers both of the problems stated above and delivers promising solutions. The theory and practice of sample rate conversion will be showcased in section 3.3.2, and the methods of bridging simulated data and real-world data would be presented in section 3.3.3. and 3.3.4. Novel methods and techniques that uses the power of deep learning accompanied with the hybrid database would be covered in section 3.3.5.

The main purpose of this research is to create an automatic and data-driven Structure Health Monitoring (SHM) method that could be used on structures that are situated in harsh environmental conditions such as a wind turbine. The research aims to utilize the strength of Artificial Intelligence (AI) and Deep Learning (DL) to break through the limits of traditional SHM techniques, which requires extensive human compilation and lacks flexibility in real-life implementations. This research aims to:

1. Create a data-driven AI based SHM technique and explore the opportunities of applied AI on structural analysis.
2. The AI method aims to be data-driven so that it that requires minimal human intervention so that it minimizes human errors.

doi:10.6342/NTU202401548

3. The AI needs to be able to automatically craft and detect features from the signals so that the automized potential of AI could be highlighted.
4. The AI needs to be highly generalizable and adaptable so that it could be transferred to different implementations and different projects.
5. Create a standardized and general AI-applied SHM framework for future researchers to apply and optimize.

1.4 Research Process and Thesis Structure

The research can be simplified into two simple stages. The first stage is to create a database of a lab-scale 2D wind turbine foundation model. This stage includes using accelerometers to acquire data of the structure and also use Finite Element Model (FEM) to create a digital replica of the structure, which could be used to acquire data to supplement the data that are hard to acquire or model from experimental setups. The data from these two sources will be joined as a hybrid database for the neural network so that it can learn from more damage scenarios and experiment with the idea of hybrid measuring technique, which of merges different sources of data [5]. In this stage signal processing techniques would be heavily used to create the database so that the sample rate from different sources could be synced.

The second stage is to build an appropriate AI suitable for the damage detection task. The AI would be trained on different databases and the hybrid database so that it can meet good accuracy along with the 5 goals on the previous sector. The AI would then be compared to the other powerful contemporaries to compare the accuracy and efficiency of different designs. Finally, the proposed AI along with the database would be saved and a more flexible version that users could tuned for different applications will be set public for public use.

doi:10.6342/NTU202401548

Chapter 2 Literature Review

2.1 Overview of O&M Procedures of Wind Turbines

Operation and Maintenance (O&M) consists a huge portion of the cost in the cost of energy of any offshore wind farm, according to studies conducted, O&M cost takes up to 23% of the total invested capital and it is the main expenditure in most of the an OWT's lifetime [6] [7]. Renting maintenance vessels are very costly and professional maintenance teams are a limited resource that should be saved as much as possible but too infrequent visits to windfarms might result in complete failure of the OWT thus increase downtime. O&M is a task that that faces challenges in risk management, human resources, capital expenditure and much more and an optimized maintenance strategy thus is coveted by energy companies. Main failure modes of the wind turbines and its substructures will be discussed in the section 2.1.1 and three kinds of mainstream O&M strategies and the details will be covered in the section 2.1.2.

2.1.1 Failure modes and effects of an OWT

OWTs is subjected to many kinds of harsh environmental loads such as wind and wave loading chemical damage from pollution during operation. It is also subject to interior damage the turbine such as shaft imbalance, gear tooth damage and high temperature. These problems create two main categories of failure modes that will occur in OWTs, the first one is caused by long term loads and aging such as material fatigue and corrosion and the second one is caused by short term and high stress overload that would cause abrupt breakdown without any predictability. These two failure modes have the potential to create major failures, while only 25% of all failures that contributes to 95% of the downtime [8].

There are various components in a OWT system and each of them have different

doi:10.6342/NTU202401548

major failure modes [9]. The rotors, gearboxes, and shafts are constantly loaded by rotational stress thus imbalances and misalignment play a huge role in failures. The blade and rotors are also subject to wear and fatigue by wind and rain which creates crack and corrosion. Bearings take a lot of pressure from other mechanical components thus they have a large modality of damages such as overheating, shell damage such as spalling and cracks, and also fatigue. The generator is one of the most complex components and it faces challenges like overheating and electrical problems and due to its function, there are also risks of overheating and excessive vibration. The tower and foundation are what holds the whole turbine together and it is the component that requires the most steel and it also faces the most environmental loads from wind, waves, mooring system stress. Thus fatigue, foundation deterioration, soil scoping and welding problems are all potential failure modes.

2.1.2 Mainstream O&M Strategies

Because of complex damage modes and expensive cost of O&M tasks on offshore wind farms, optimizing the O&M frequencies becomes an integral part of a sustainable wind farm. A high visiting frequency would result in a huge rise in costs because renting specialized fleets and trained technicians are very expensive, but a low visiting frequency would expose the wind farm to high risks of major failures and increase downtime. There are two mainstream O&M strategies and the pros and cons would be introduced in this section.

Corrective maintenance is the first O&M strategy and it is employed only when the failure has already occurred in a turbine. Corrective maintenance can be efficient because it cuts out the need for unnecessary visits, this strategy is suitable if the risk of downtime is lower and there is an easier accessibly towards the subject. In larger scale offshore wind farms, this strategy has proven to be impractical because compared to onshore wind farms

doi:10.6342/NTU202401548

offshore wind farms have greater complexity [10], unreliability, and bad accessibility. This would add on to the risk of increasing unwanted downtime and less efficiency.

Proactive maintenance is a much more sophisticated strategy because it can patch up minor flaws before it develops into a major failure. Preventive maintenance is a time-based proactive maintenance strategy that applies scheduled visits based on a predetermined period or a give level of power generation [9]. The premise of this method is to balance the economic benefits with risk management by optimizing both power generation and routine maintenance. Compared to corrective maintenance, this method can have a better use of vessels and manpower with reduced effect on exterior factors such as weather. In order for this method to be effective, an optimized selection of intervals should be carefully considered to get the full benefit of this method and there are many efforts that have been made.

Another proactive maintenance method mainly utilizes sensors to achieve a Condition-based Monitoring System (CMS). Because of the decrease in prices and it provides increased knowledge of real-time OWT information in operation conditions, this technique have been widely researched and developed in the past decade. Frequency analyzation is a classic method for damage detection and fault isolation. Supervisory Control Data Acquisition (SCADA) have gradually played a bigger part in supervising operation conditions and providing reliable information for optimizing maintenance scheduling. The valuable knowledge that CMS provides can aid engineers to have a better understanding of operation conditions of OWTs and create a better plan and strategy for O&M, thus avoiding over or under maintenance. Predictive methods have taken this a step further by calculating parameters before failure occurs, and maintenance plans can be planned to prevent major failures from happening. The latest popular research in this topic is Digital-Twin Platforms [11], this method combines structural mechanics and

doi:10.6342/NTU202401548

virtual modes to predict the remaining life of the structure or when maintenance should be performed. These sensor-based technologies also have a couple downsides, firstly it greatly increases the complexity of the structure and mounting and constantly examining the wellness of the sensors could be a problem. Also, false alarms and missed reports also are present in this technique therefore a more wholistic and robust methods such as AI and advanced algorithms should be considered in future implementation of CMS.

2.2 Structure Health Monitoring Methods (SHM)

Because of material defects that can occur throughout the lifetime of a structure, structural damage from deterioration is inevitable unless there are methods to quantify and detect the damage before large scale damage occurs. Structure health monitoring is a subject focused on studying parameters that are related to the well-being of the structure and its main goal is to achieve an early detection on these damages so that it can prevent catastrophes from happening, and even extend the lifetime of a structure. SHM could potentially aid the O&M process by using advanced methods to acquire detailed information about the well-being of a structure so that researchers and on-site operators can optimize the O&M tasks with a better understanding of a structure.

In a review paper by Keith Worden [12], damage is defined as material and/or geometric property changes of a structure which includes deviations in the boundary conditions or intra-system connectivity, which might affect the performance of the structure in the future adversely. Structure health monitoring is a subject focused on studying parameters that are related to the well-being of the structure. There are four main stages in SHM, suggest by Farrar and Sohn [13]. The first stage is Operational Evaluation (OE), which mainly concerns about the conditions and economic benefits of deploying the SHMS. The second stage is data acquisition, fusion, and cleansing. There are two

doi:10.6342/NTU202401548

main methods, contact measurements and non-contact measurements. Contact measurements utilizes strain gauges or accelerometers to acquire data from structures and mainstream non-contact methods include visual measuring methods or ultrasonic techniques. Data normalization is the process of organizing the data acquired from different environmental and operation conditions. The third stage is Feature extraction and information condensation. The damage condition of the structure can be analyzed by selecting and examining different features. Numerous techniques are utilized in this process, such as time and frequency methods and more recently data driven methods. Data condensation is crucial for researchers to analyze and access the conditions of a structure throughout its lifetime, thus developing a robust data reduction technique is needed to preserve data variety and sensitivity. The last process is statistical model development. This is a more refined damage severity assessment and damage location using more sophisticated algorithms and the topic that attracts the most researchers.

Data acquisition is one of the main focuses of SHM because recent technology advancements have given researchers the opportunity to acquire data with different mediums. Traditionally, there are two data acquisition mediums: strain based and vibration based. Strain based methods [14] measures the strain of a structure then detects the modal strain energy at different modes and elements. Vibration based methods usually measures the acceleration of certain points on the structure then conducted Fourier-based methods on it. Yet these methods traditionally require equipment to be mounted on the structure, and for long-term monitoring the well-being of the equipment could be a potential issue. Using non-contact visual detection methods techniques coupled with Unmanned Arial Vehicles (UAVs) have been a major field of research because it has the potential to conduct visual inspection in hard-to reach places and the signals are acquired using object tracking algorithms.

doi:10.6342/NTU202401548

Optical Flow (OF) [15], also known as KLT tracking algorithm, have proven to be a useful object tracking method that could detect the movement of a certain object over time and it was developed by developed by Lukas and Kanade. The methods select a region of interest and then tracks the pixel movement frame by frame and finally it will output the movement of the object in the region of interest over time. Yet there is a challenge that needs to be overcome for UAVs to use this technique, and that is to separate the structure movement from the camera movement since the camera captures the sum of the movement together. This subfield of research is called camera image stabilization and one example of this example to real life structures is from Satoru Yoneyama [16], focusing on measuring the deformation of a bridge under dynamic loading. To capture accurate vibration signals in real life that are subject to the vibrations from vehicles and constant shaking from wind loads, an algorithm capable of image stabilization is necessary. The measurements are divided into fixed measurements without image stabilization, fixed measurements with image stabilization, and fixed measurements without shaking. Therefore, initial experiments were conducted in the laboratory for algorithm verification and by knowing the camera displacement, the displacement of reference points (assumed to be stationary) was measured using digital image correlation to verify the effectiveness of the image stabilization. Subsequently, measurements were performed on a large-scale structure in the laboratory to examine the accuracy of the overall algorithm without knowing the camera displacement. After achieving satisfactory results in this phase, field measurements were conducted.

Digital Image Correlation (DIC) is another method that could be used with OF methods because it can directly track the object of interest by using correlation methods in pixels. A more recent study is done by Ashim Khadka [17] on a real scale wind turbine. In the paper, the team first used a drone to capture wind turbines on a smaller scale indoors,

doi:10.6342/NTU202401548

and the conditions were controlled ideally. By comparing the shape of the first mode using fixed and contact accelerometers (standard values), they were able to optimize the DIC algorithm they used. In this stage, the experimental team changed from using six symmetric blades to three blades for comparison in data processing. Then, the team moved the instruments used in the first stage to the outside and applied them to measurements on a flying drone. In this stage, it was necessary to eliminate the effects of image stabilization and analyze the modal differences between rotating and non-rotating blades. The last stage involved using the drone to perform measurements at an actual location. However, since the wind turbine blades were 150 meters long, the team used stitching to divide the blades into two sections and ultimately plotted the modal shapes and overcame the difficulty of the whole structure being out of frame. Jongbin Wang [18] further expanded on this method by combining deep learning with OF methods to achieve a multi-point tracking methods that also have better robustness. Through detailed analysis of the frequency and modal shape of natural modes, it was found that the tracking errors in KLT originated from the selection of feature points. Some of the selected feature points were located on the background, leading to an underestimation of the magnitude of vibrations after averaging all the points. This significantly increased the temporal errors in KLT. The proposed tracking method in the paper can reduce this effect by ensuring that the tracked points are evenly distributed on the structure, resulting in more accurate calculations. Additionally, the feature point tracking method proposed by the team can mitigate external disturbances, such as sudden occlusion of the structure by an object. Therefore, compared to KLT, it exhibits greater robustness in real-life conditions. Yoon [19] utilizes the KLT method to track the displacement of a multi-story building and investigates whether commercial cameras (GoPro, LG) can match the accuracy of complex cameras. The team employs the Eigenvalue Realization Algorithm (ERA) for

doi:10.6342/NTU202401548

detection. One of the major highlights of the paper is the exploration of errors. The team finds that the frame rate is a crucial factor in detecting natural frequencies. As the sampling frequency increases, the aliasing error related to the observed natural frequencies decreases. Moreover, higher resolution leads to better results in terms of temporal displacement and modal shape analysis. Lastly, the paper emphasizes the importance of selecting a region of interest (ROI). When manually setting the ROI, it is crucial to ensure that the size of the ROI guarantees that the feature points on the structure are more prominent than those in the background environment. This ensures favorable tracking results on the structure. It is also essential to ensure that the observed structure behaves as close to an ideal rigid body as possible to avoid misjudging the overall structural displacement.

Damage detection make use of the basic principles of SHM and applied advanced monitoring or detection algorithms for real-life scenarios. In order to perform SHM to for damage identification, one must compare the same system in different states, in other words there needs to be damaged and undamaged datasets for the algorithm and this would usually require feature extraction beforehand. There are numerous methods to achieve damage detection for SHM process, frequency-based methods are one of the most prominent methods for detecting defects in a structure. In a detailed review from M. M. Reda Taha [20], it detailed some recent findings about the usage of wavelet transform, which could bring potentially more to signal analyzing such as dealing with noises in the signal and being more flexible to different analyzing tasks. It is also stated in the review that the process of decomposition can be useful for denoising or for damage detection in SHM systems and an entropy-based criterion has been nominated by researchers to be the most successful method for selecting the optimal wavelet packet tree for efficient damage diagnosis. It is worth noting that the choice of wavelet function is simply application

doi:10.6342/NTU202401548

dependent and requires careful scrutiny in its usage and results. Kurtis Gurley [21] conducted research on wavelet transform on the application of earthquake, wind and ocean engineering and have found that wavelet transform could have the potential to be more favorable compared to the traditional spectral analysis (Fourier) of nonstationary signals because spectral methods cannot describe the local transient features due to averaging over the duration of the signal. Yet the limitations of wavelet transform is that high resolution cannot be obtained in both time and frequency domains simultaneously because of its inherent calculation methodology. Yu Xin [22] have focused on overcoming this issue by mixed spectral based and wavelet-based methods in an Operational Modal Analysis (OMA), which conducts modal analysis of structures in an operational state, and found success in reducing noise. The team firstly used Standardized Auto-Regression Spectrum (SAR) to accurately find out which range of the natural frequency is present in the structure and then used Empirical Wavelet Transform (EWT) to conduct detailed analysis on each frequency range to find out the exact frequencies of the natural modes. Hilbert Transform (HT) is then used to extract the modal parameters, such as mode shape, frequencies, and damping ratios. The whole process has been examined with a lab scale structure, a numerical structure, and an on-site experiment. SAR spectrums can greatly reduce the effects of the ambient noise and nearby and neighboring complex peaks of natural frequencies of the structure could be found with good accuracy because of the use of EWT.

Another issue in the system identification phase would be the presence of noise and harmonics, which would greatly interfere with the modal parameters that we would extract from the raw signal. Thus, a great area of opportunity is to modify the traditional SHM techniques so that it can apply to real-life structures by increasing the noise resiliency or be more generalizable to different applications. Carlos Andres [23] have

doi:10.6342/NTU202401548

conducted a detailed review on all of the system identification techniques since 2012, and have placed significant focus on frequency and time-frequency based methods because of the effects of lowering noises and increasing the measured accuracy in system identification phase. Xiaofeng Dong [24] conducted a long-term monitoring task on a OWT in China compared to his contemporaries that only conducted studies on numerical models. The team mounted their equipment in the turbine and then studied the vibration conditions in different wind conditions and have found that modal parameters would slightly shift in different environmental loadings. For system identification, a modified stochastic subspace identification technique is used to separate the harmonics and noise of the environment. The effects of different environmental conditions on the modal parameters are then compiled to serve as a support for O&M safety evaluations and possibly future design considerations. Premjeet Singh [25] proposed a method of using a building information modeling method to create a modal database that could can let the engineers monitor the state of the structure remotely by transmitting the accelerometer data. The easy visualization of modal parameters enables engineers to get a quick analysis of the structure.

In the field of FEM model analysis or model updating, the local damage is usually modelled as a loss in stiffness in certain substructures. The stiffness reduction and damage localization can then be extracted using various methods like Frequency Response Functions or modal strain energy. Qianhui Pu [26] detected and quantified the damage on a concrete beam of several sections by using experimental FRFs and model updating methods. The concrete beam is divided into 14 sections and are applied different stiffness percentage reductions at different places of the beam. V Srinivas [27] used a genetic algorithm-based searching method and used strain energy change ratio in different stiffness reduction segments to identify damage on large scale structures. Qiang Mao [28]

doi:10.6342/NTU202401548

used a Multireference impact test (MRIT) and experimental modal analysis to conduct a research on a three-span simply supported bridge which is located in Mossy, West Virginia, USA. The researchers analyzed the boundary conditions using sensor data to extract modal parameters then applied FEM model updating to get a clearer vision on how the uncertainties of scouring effects and intricacies of the soil-pier interactions at the foundations of a bridge. But to conduct a more realistic analysis, the parameters of the concrete material is unchanged but other parameters such as depth, distributed spring for soil pier interaction and spring on bearings are updated.

2.3 Deep Learning and Structure Health Monitoring

2.3.1 A brief evolution of Deep Learning

In the scope of machine learning, there are many kinds of machine learning algorithms that can help us on performing different tasks. For regression problems, there are basic algorithms such as linear regression that is capable of dealing with simple problems. There are also highly advanced regression techniques such as Gaussian Process Regression (GPRs) [29] that allow the user to actively add data points to explore and exploit unknown functions with great flexibility. There are also Natural Language Processing (NLP) models that can trains a network with words and embeddings that creates great technologies such as machine translation and generative AI. More sophisticated models will utilize Gated Recurrent Units (GRUs) or Long Short Term Memory (LSTMs) to prevent errors on long sentences. There are also advanced implementations for better translation such as Bleu Score and the Transformer Network that can help NLP models achieve human level performance. There are also Convolutional Neural Networks, CNNs that are specialized on tasks such as image

doi:10.6342/NTU202401548

recognition and classification. CNNs are a highly specialized DNN that could greatly reduce computational time by sharing computational units on local features while retaining hierarchical features that could be used for further classification and analysis. U-Net is a network that can classify images pixel by pixel and then recreate the image with the classified labels, which is now gaining huge interest in the field of self-driving cars. Because of the complexity of the network, they generally require a great amount of data to train well. Some researchers have presented strategies such as One-Shot Learning and You Only Look Once (YOLO) algorithm that requires little amount of data for image recognition and the technology is now widely used on facial recognition tasks. As the dataset gets bigger and more diverse, networks have to also grow larger and deeper, but problems such as vanishing and exploding gradient will arise and this will result in poor training results. Networks such as Residual Networks (ResNets) counter this problem by introducing residual blocks that can relay the information from previous layers by a skip connection and therefore calculate valid numerical values while ensuring better training result.

2.3.2 Key Advantages of Deep Learning

The main advantage of deep learning is its highly automized and adaptable nature. A major limitation of NNs is that they only as good as their dataset and generally NNs perform poorly on extrapolation. Therefore, if we increase diversity of the dataset, the NN would have better generalizability and performance across different datasets. If this condition is met, the network can perform well on a variety of different environmental conditions. Data Augmentation is a great way to increase data variability and supply more data for the NN and increase accuracy and performance. In recent times, sensor technology has made a huge leap and this have led to huge drop in sensor prices but rises

doi:10.6342/NTU202401548

in quality and accessibility and it made costly long term data acquisition tasks easier to perform than ever. Therefore, data driven monitoring methods are now gaining more attraction toward researchers because it could fit better in realistic conditions with harsh environmental loads and time-varying changes.

Wei Zhan [30] developed a simple DL framework that can manage raw vibration signal well by using a wide first deep learning framework. This algorithm can have good local feature (wide kernels) to detect low frequency signals and periodic changes while maintaining non-linear information (deep layers). This method works also adapt well to raw noisy signals. Yequein Bao [31] developed a DNN to detect anomalies in an Bridge SHM system. Through the usage of a unique network structure called fusion network that can combine networks from small substructures and a visual database, the team yielded a training set accuracy of 90.7% and validation set accuracy of 85.8% in 7 different cases of structural damage. Through various comparisons, this architecture increased the accuracy compared to different networks.

Because the NN blow up in units and memory usage when we stack more and more layers after one another, another method to increase the efficiency and memory usage of DNNs is to use a convolutional based method which convolves the neural units in hidden layers so that the hidden units can share common parameters. As an extended method of the popular method 2D CNNs, 1D CNNs can help with this problem effectively. Onur Avci [32] used 1D-CNN on wireless sensor networks for structural damage detection. The damage is from inflicted by loosening the bolts on a large-scale steel frame. The data of each sensor is then fed to a unique CNN to identify local damage. Of each three directions of freedom a CNN was assigned, then the team selected the best direction of damage indication. The CNN is able to achieve good accuracy in a wireless, ambient setting while using less computational resources compared to 2D CNNs. A major advantage of this

doi:10.6342/NTU202401548

CNN is that it can be automatically down-sampled, doesn't require signal processing techniques and filtering.

Another major advantage of deep learning is its generalizability. Transfer learning is a common tactic for deep NNs to apply on different but similar tasks. The concept of transfer learning is to utilized a pre-trained network along with weights and biases to perform similar but slightly training tasks. To perform transfer learning the user simply just has to load the pretrain network, then un-freeze the last couple of layers of the neural network, then apply fine-tuning techniques to train the un-freezed layer for the specialized task. The parameters in the pre-loaded layers will not be changeable but the un-freezed layer is fully customizable so that it can perform specialized tasks with the power of the pre-trained layers. The benefit of transfer learning is that it doesn't require a lot of training data to perform a specialized training task because the transferred DNN was pretrained with big amounts of data in original layers, which performs great in basic tasks such like edge detection. This can work greatly if we only have a small amount of a highly niche data that we need to apply analysis on.

Mona Chamangard [33] countered the problem of having limited data by creating a 1D CNN and applied transfer learning on three different structures. Firstly, the team pretrained their proposed network with FEM simulations from Quatar University (QU). Then the pretrained network is then applied to a benchmark structure developed by Burkett. The network is then trained on only limited data on the Tianjin Yonghe Bridge. The results demonstrated that the proposed method can successfully be utilized for damage detection in civil structures, even with a low amount of acceleration data. Eleonora M. Tronci [34] developed a Time-Delay Neural Network (TDNN) using a collection of human voice recordings. Cepstral and pitch features derived from the speech data are used as input features for the TDNN. Then applied transfer learning from the

doi:10.6342/NTU202401548

intermediate parameters to the structural datasets. It also introduced a noise augmentation in the audio domain improves the model performance, proving to be a key optional step for enhancing the model's classification capability.

Because CNNs have great image identification efficiency and precision, a lot of researchers used it on visual inspection on various SHM tasks. Y. Narazaki [35] used a CNN to automize object classification pixel by pixel from bridge images by creating a 22 layer multi-scale CNN with a similar architecture as ResNet to apply pixel to pixel mapping of objects. The team used a Softmax layer is used for classification of 10 classes and adam optimization is used to optimize parameters while a ReLU activation function was used as activation function. Batch normalization, weight decay of 0.0001, and median frequency balancing are implemented. For pixel classification, techniques like Superpixel averaging and conditional random field (CRF), which is effective in removing isolated or unsmooth pixels, have been used so that the model is used to improve performance. Kasthurirangan [36] trained numerous final classifier layer using ImageNet pre-trained VGG-16 CNN features for the crack detection tasks. The images are taken from FHWA/LTPP database without data augmentation, with in total 1056 images. The pavement images datasets used in this study are significantly more complex than those used in most previous studies, in order to solve the false positive problems. Among these 5 classifiers, the single-layer NN classifier trained on ImageNet pre-trained VGG-16 DCNN features seem to yield the best performance, followed by SVM and LR classifiers.

Heng Li [37] created a CNN and a unique end-to-end architecture to learn crack measurements. The network is composed of two components: the down sampling layer, which is a traditional convolutional layer setup and a up sampling layer, which can process cracks back to the original data size. Through various comparisons, this architecture greatly reduced the training time while only trading off a little accuracy

doi:10.6342/NTU202401548

compared to previous methods like CrackNet. This method also exposed some weakness such as fake cracks and missing cracks. N. S. Gulgec [38] used a relatively shallow CNN to detect visual damage on a welded steel plate generated by ABAQUS simulation. Variability in training, validation, and test data sets are formed by exploiting different loading cases, damage scenarios and noise levels. Hyper-parameters are selected by dropping poor performing network designs. This study identifies the unseen damages with 2.06% error in noiseless signals, most of the results came from False Negatives. Even with 2% increased noise, the system is robust enough to result in 3.51 % error. Chuncheng Feng [39] used transfer learning from Inception-v3 Net to apply transfer learning on 18605 images on 5 different damages of a hydro-junction infrastructure. The images were collected by UAVs with applied data augmentation for further training. They froze the final layer of the Inception network and trained only the fully connected layer which outputs the damage classification via a SoftMax Layer. The proposed method achieved 96.8% accuracy, which was much higher than manually selected features from a SVM (61.2%) The deep features that a CNN could capture is the main reason for this achievement. Also due to GPUs and a small dataset, the images take little time to train. Chen Lu [40] proposed a 2D CNN structure to classify the health state of a rolling bearing. Firstly, the team used a time series to image permutation to change the 1D data into 2D image. The images are then trained using supervised greedy training, combined with the hierarchical structure of CNN to ensure robust results. With a relatively shallow model, only using convolution computation, rectified linear units, sub-sampling, and a classification layer, the model is able to extract salient features and acquire good accuracy for different bearing conditions. The team has also reported that the training heavily relies on the data quality, thus generalization could be one of the future research topics.

doi:10.6342/NTU202401548

With some detailed crafting of the original 1D raw data into an 2D image, researchers can also use a CNN to train on such datasets. Nur Sila [41] used normalized strain data to detect and then identify the localized damage on a CNN. The author used two optimization techniques (binary & regression) in the same pipeline in the deep learning process, which provided a more efficient learning. Crack detection in pavements is a huge issue of infrastructure health and it has attracted a lot of researchers to automize the process. One method is to transform the signal into a power spectrum and use the neural network to train on a dataset of spectrums. Hyunjun Kim [42] create a region-based CNN that can detect peak regions on a power spectrum. Three experiments including indoor structures and out-door structures were conducted to verify the performance of the algorithm. The detector can detect most peaks with high accuracy. The network is only trained by the data of the experiments. Another method is by automatically transforming the raw data into an image by directly plotting the time-series data. Zhang Wei [43] transformed 1-D vibration data (40000) into 2-D time (200x200) image. Use CNN to learn different fault patterns in images. The transformation technique from raw data to image can condense the data for faster computation for CNN, while retaining quality results.

doi:10.6342/NTU202401548

Chapter 3 Frequency Identification with CNNs and DNNs

3.1 Research Methods

3.1.1 Basic Calculations of DNNs

A complex DNN is built from fundamental units, and by training every single unit in the full network, the network can make accurate predictions, a representation of a unit is drawn in Figure 1. Each unit is associated with a weight *(w)* and bias *(b)* that will determine the influence of the units of the previous layer. The unit will accept inputs from the *l-1*th layer and each unit will be calculated by multiplying the associated weights and biases, but for simplicity we assume that there is only one unit in the previous layer $a^{[l-1]}{}_n$. In 3-1, $w^{[l]}{}_n$ and $b^{[l]}{}_n$ represents the weights and biases of this single unit. $z^{[l]}{}_n$ is the result of this calculation, and this variable will be fed to $g^{[l]}(x)$, also known as an activation function. There are many different activation functions that could be used in different projects and implementation, but the common practice is to use a rectified linear unit (ReLU) or tanh activation function as the initial implementation. The ReLU activation function and tanh activation functions are listed below in 3-3 and 3-4.

$$z^{[l]}{}_n = w^{[l]}{}_n a^{[l-1]} + b^{[l]}{}_n \quad (3\text{-}1)$$

$$a^{[l]}{}_n = g^{[l]}\left(z^{[l]}{}_n\right) \quad (3\text{-}2)$$

$$f(x) = \max(0, x) \quad (3\text{-}3)$$

$$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (3\text{-}4)$$

doi:10.6342/NTU202401548

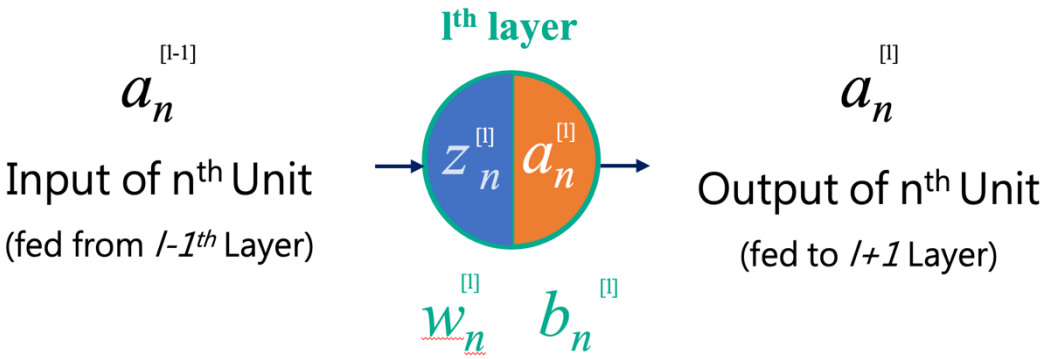


Figure 1. A basic unit of a DNN

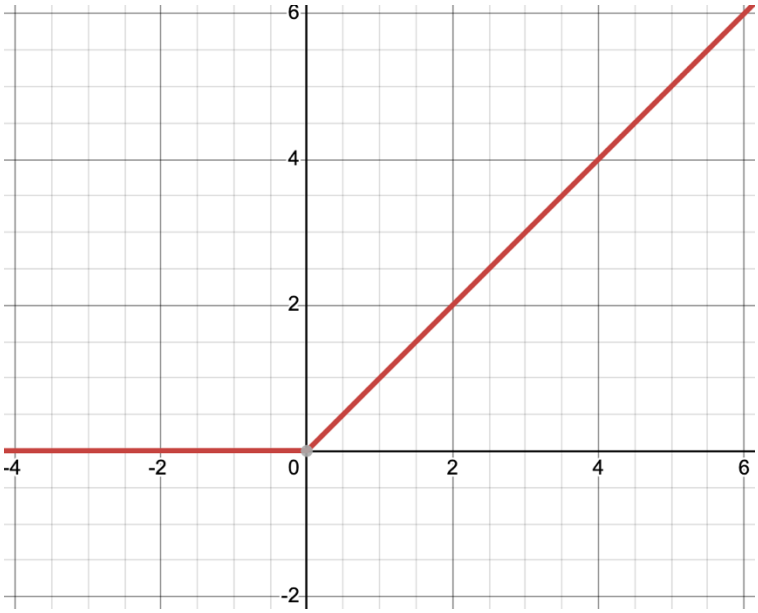


Figure 2. ReLU Function

The first layer of a Neural Network is the input layer, which we will input the data that would be used to train the neural network. The last layer of the neural network is the output layer, which will classify every data in that is used to be trained or predict results from a different database. The layers between the first layer and last layer are all called hidden layers, which are all made from a neural network unit. Each weight and bias in a single layer are now vectorized as $W^{[l]}$ and $b^{[l]}$ and the nth notation is now denoted and the importance of a unit is controlled by these trainable parameters. With

doi:10.6342/NTU202401548

each training epoch, the NN will learn from the errors by comparing the predictions from the known results and update the weight and biases until it reaches the best result.

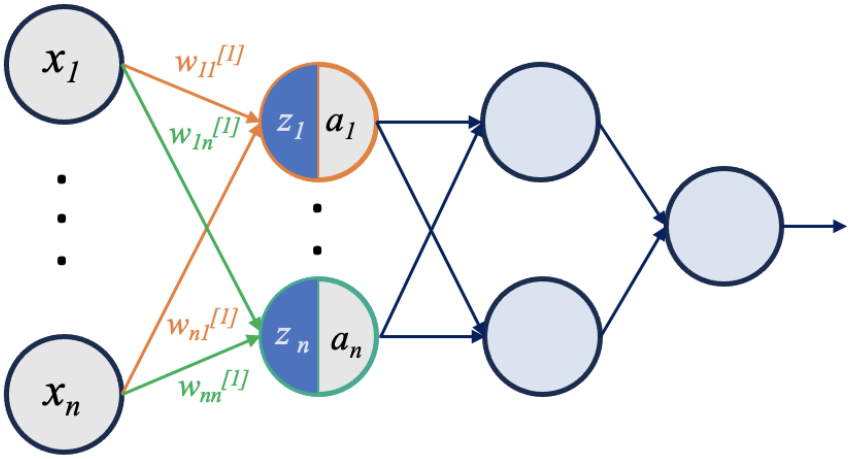


Figure 3. A simplified example of a neural network

$$Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]} \quad 3\text{-}5$$

$$A^{[l]} = g^{[l]}(Z^{[l]}) \quad 3\text{-}6$$

$$dZ = A - \hat{Y} \quad 3\text{-}7$$

The learning process of a neural network consists of two basic procedures, Front Propagation (FP) and Back Propagation (BP). The goal of FP is to estimate the result and calculate the error of the training dataset each epoch. Firstly, the inputs from the previous layer $A^{[l-1]}$ will be propagated to the current layer l, and the results will be calculated in the units of the current layer following the procedure and this process will be followed until the last layer. In the last layer, the NN will make a prediction based on the probability calculated by using a Soft-max function, then it will calculate the error between the predictions A to the correct labels $\hat{Y}$ and this will give us a cost function dZ, which is the sum of the error of all examples.

doi:10.6342/NTU202401548

After the FP process then NN will then continue the BP procedure to update the weights and biases in the neural network. In order to update the results of FP, we must obtain the gradient of the output of each layer $dA^{[l]}$. It is worth noting that in the final layer $dA^{[l]}$ is the same as cost function dZ. We can use 3-8 to calculate the error of the inner variable $dZ^{[l]}$ and then use 3-8 and 3-9 to calculate the gradient of the weight and bias in every layer. In this notation $dZ^{[l]}{}_i$ represents the elements of the i[th] row and this process only sums up the elements in row-wise.

$$dZ^{[l]} = dA^{[l+1]} * g^{[l-1]\prime}(Z^{[l-1]}) \quad 3\text{-}9$$

$$dW^{[l]} = \frac{1}{m} dZ^{[l]} \cdot A^{[l]\cdot T} \quad 3\text{-}10$$

$$db^{[l]} = \sum_{i=1}^{n} dZ^{[l]}{}_i \quad 3\text{-}11$$

$$dA^{[l]} = W^{[l]^T} \cdot dZ^{[l]} \quad 3\text{-}12$$

In the final step we can use 3-13, 3-14 to update the weights and biases. The goal of the neural network is to find the minima of the cost function, which would also represent a good fit of the dataset. After this step the NN will repeat the FP and BP until it reaches convergence or attains enough accuracy for the user. For each learning epoch the weights and biases will be updated until the training is terminated.

$$W^{[l]} = W^{[l]} + \alpha * dW^{[l]} \quad 3\text{-}13$$

$$b^{[l]} = b^{[l]} + \alpha * db^{[l]} \quad 3\text{-}14$$

3.1.2 Basic Calculations of CNNs

In order for researchers in Computer Vision (CV) research field to deal with high resolution images, the Convolutional Neural Network (CNN) was created. The CNN uses a simple yet effective structure to make analyzing huge amounts of pixels effective and

doi:10.6342/NTU202401548

thus is widely used in subfields like Image Recognition (IM), Object Detection (OD) and Neural Style Transfer (NST).

The CNN uses filters to detect edges on images. By convolving a matrix pixel by pixel, different direction of edges can be detected. In Fig. 1, the red edge is detected by convolving the left image matrix by the filter matrix (middle). The result if the right-hand matrix, which the edge is signified by the middle two columns. Although it seems like the width of the edge is amplified a lot, the effect is minimized in real practices because the image pixel is relatively large compared to the column. There are more filters developed for different edge directions and qualities, such as the Sobel Filters and Schorr Filters. The dimensions of the output matrix are determined by this formula (without padding on the edges, and the layers are denoted):

$$(image_{nxn}) * (image_{fxf}) = (image_{(n-f+1)\times(n-f+1)}) \qquad 3\text{-}15$$

If we were to add padding to make sure the output matrix has the same dimensions as the input, the padding pixels on one side will be:

$$p = \frac{f-1}{2} \qquad 3\text{-}16$$

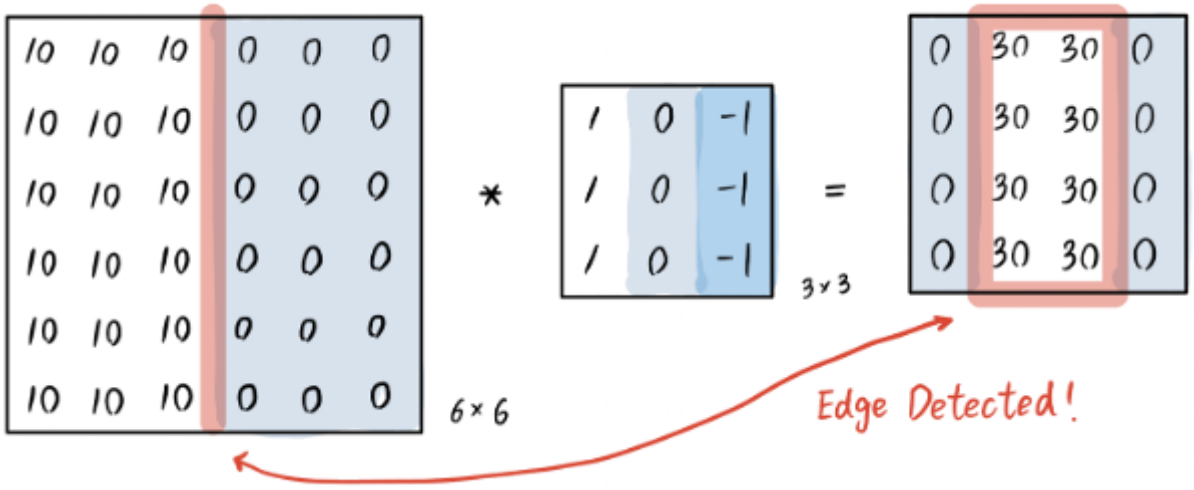


Figure 4. A simple CNN calculation process

doi:10.6342/NTU202401548

A common practice of using CNN to the highest quality is by using RGB channels. It has been cited in multiple articles that training RGB images yield better results than grayscale images. A single 2D convolution process is demonstrated in Figure 2. The first step is separating the image into three channels: Red, Green and Blue then convolving all the images with the filter. Note that the channels of the images and the filters should be the same. The second step is to add the different channels and then combining the filters into a single output matrix. Figure 3 is a representation of a single block in a full convolutional neural network. By passing in the input image as a volume of $(n \times n \times 3)$ and passing it by m different filters with the same channel size, the output will be a volume of $((n-f+1) \times (n-f+1) \times m)$. By combining multiple blocks of the same concept with increasing depth and reducing width, we can build a fully operating CNN. The benefits of the CNN come from the shared parameters during the convolutional process. The calculation cost C of a block is as follows:

$$C = \left(f^{[i]} \times f^{[i]} \times m^{[i-1]} + 1\right) * m \qquad 3\text{-}17$$

The calculation cost represents the memory used in the process. For the same output dimensions, CNN could save up to a thousand times than a regular DNN. This results in more efficient computation and potential to build a deeper and more complex network.

doi:10.6342/NTU202401548

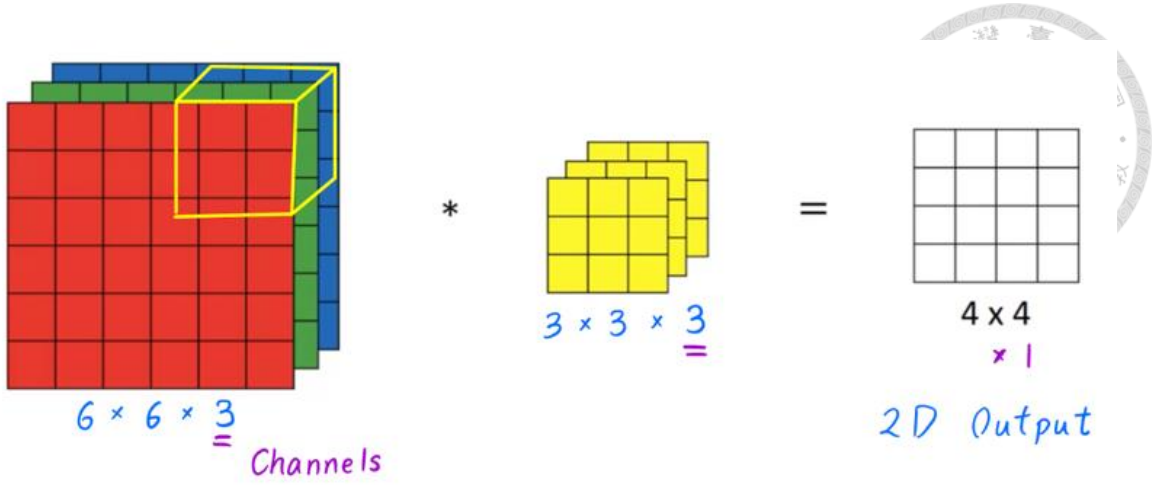


Figure 5. The calculation process of a CNN of multiple channels

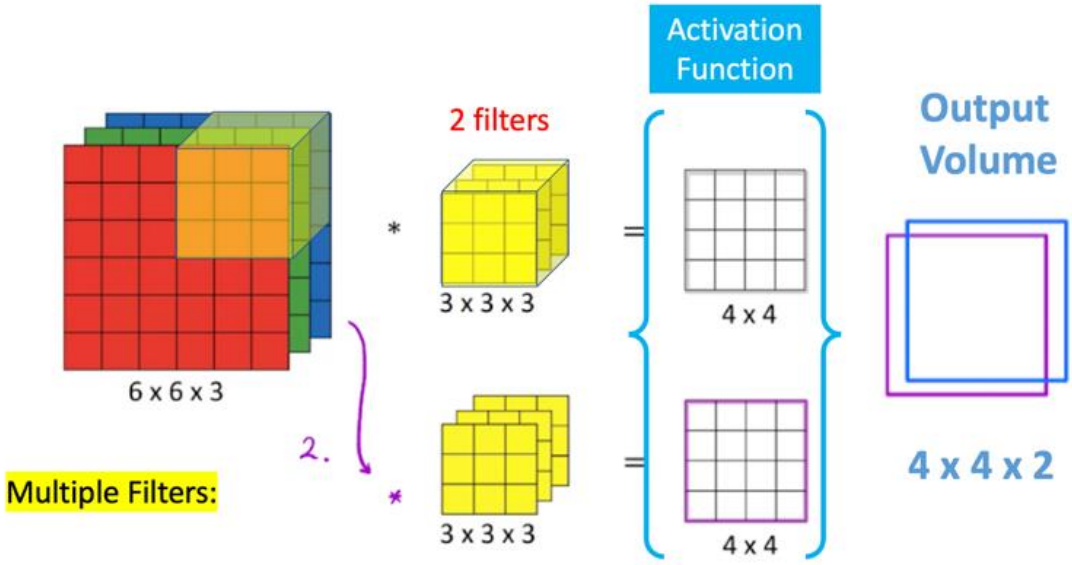


Figure 6. The calculation process of a CNN of multiple filters

The first CNN is built by LuCun et al at 1998, now called LeNet [44]. It is a relatively simple network with only 60,000 parameters, yet it inspired countless neural network researchers and people are still using more advanced iterations of LeNet to conduct complex analysis with more complex activation functions. In 2012, a much wider and deeper network is introduced called AlexNet [45], inspired by LeNet – 5. By expanding the size of the network with almost 60,000,000 parameters and utilizing multiple GPUs

doi:10.6342/NTU202401548

for training, AlexNet has shown revolutionary results in image recognition, and has been the performance benchmark since. It is also the first network that uses the concept of Local Response Activation, but this algorithm has been mostly replaced by Batch Normalization to obtain better results. In 2014, the concept of Residual Network has been developed by google. The concept of this algorithm is to shortcut the results of a previous layer to two layers after it. This method effectively reduces the problem of exploding and vanishing gradients, which are almost always encountered when building a deep network. By stabilizing the numerical solutions in deeper layers, we can create a deeper network that obtains better results.

3.2 Accelerometer Database Construction

3.2.1 Data Acquisition of Accelerometer Database Experiment

Figure 7 showcases the damage conditions in the accelerometer experiment. Ch1 to Ch4 are the locations that the accelerometer is placed. It is set up so that the full mode shape of the structure can be probed by using modal analysis methods. P1 to P4 are the points where the damage is inflicted, the damage is inflicted by unscrewing the bolts to inflict local stiffness reduction on the structure. P1, P2 are I joints and P3, P4 are L joints and the how the bolts were unscrewed are stated in Figure 8. The local stiffness reduction effect should be altering the spectrum peaks to different shapes and thus our algorithm can identify different cases by differentiating the different characteristics of the spectrum. The damage combinations of different joints are shown in Figure 8 where each damage combination constitutes 25% of the total amount of the whole damage data.

doi:10.6342/NTU202401548

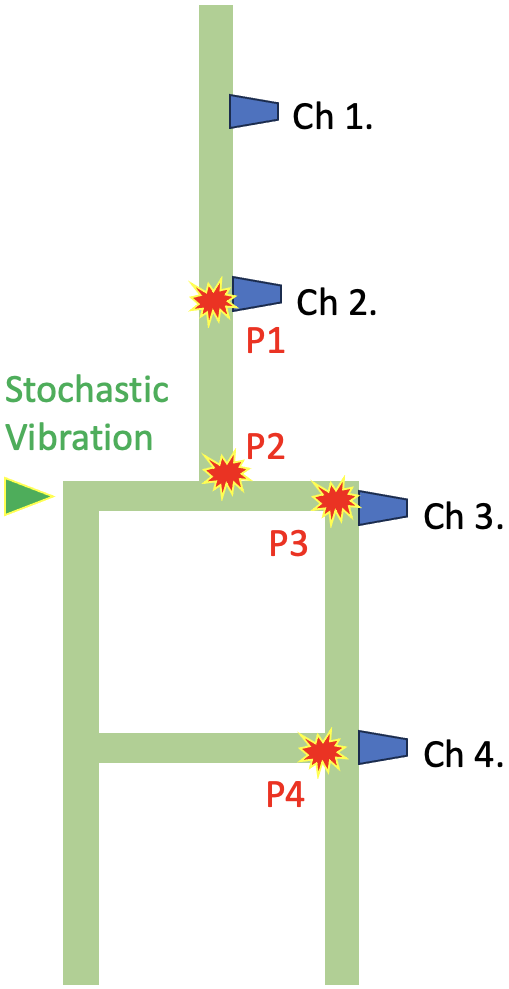


Figure 7. Damage Locations of Accelerometer Database

(a) Damage for I joint (b) Damages for L joint

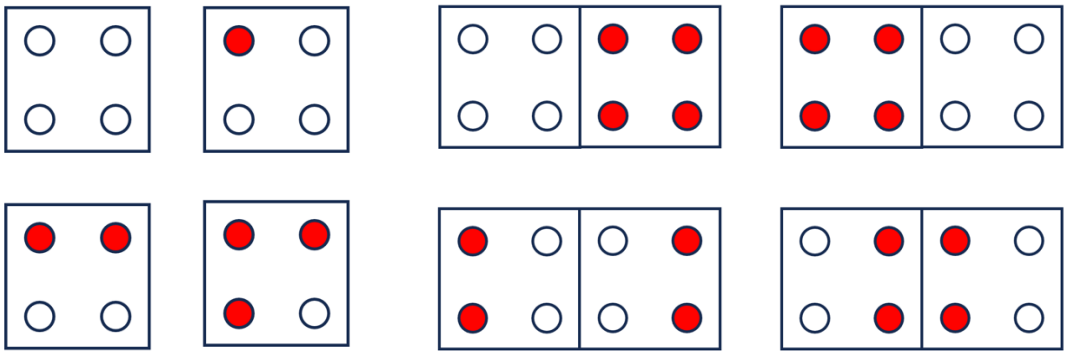


Figure 8. Damage for I joint and L joint

doi:10.6342/NTU202401548

3.2.2 Experiment Equipment and Material Qualities

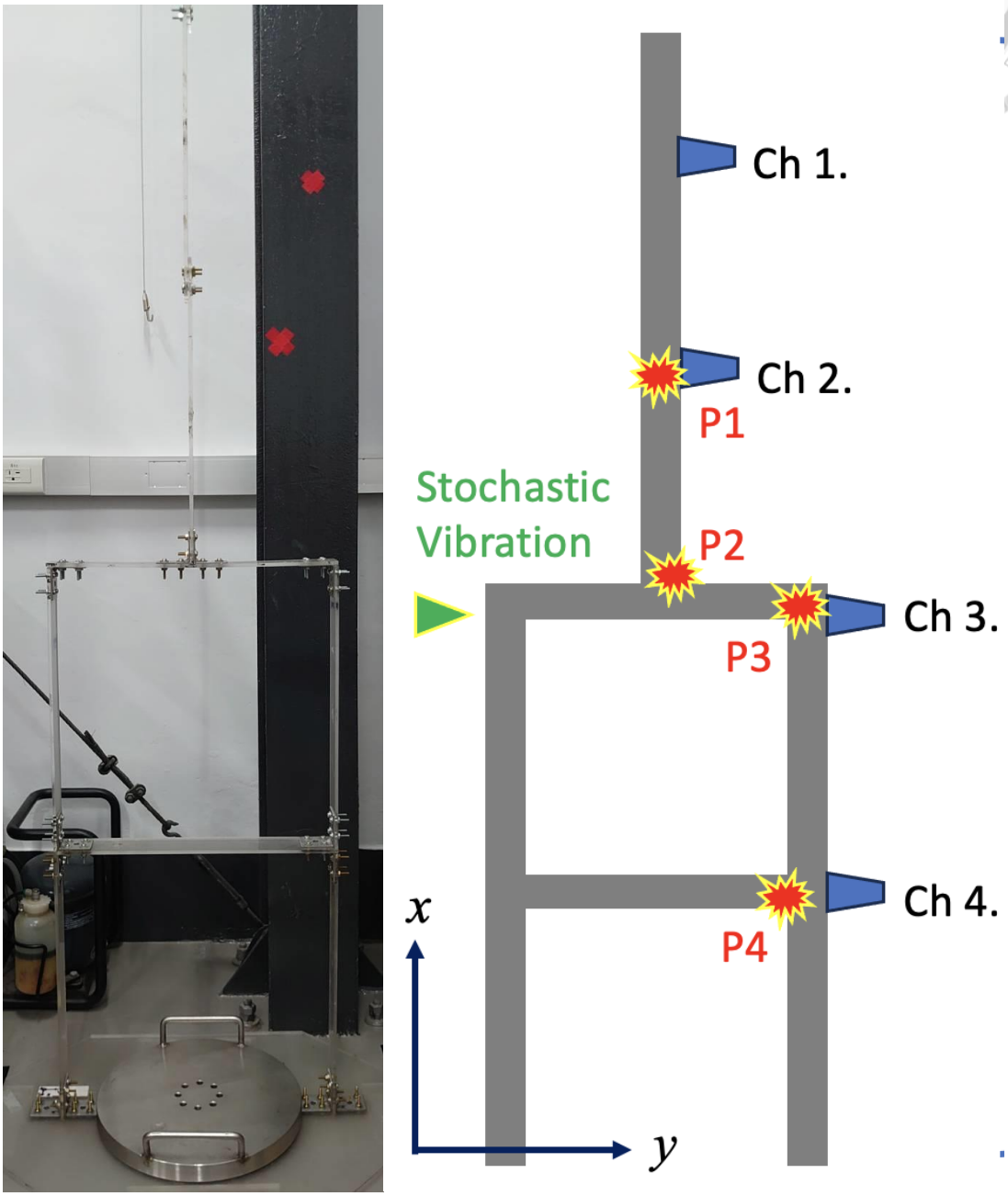


Figure 9. The acrylic structure used for experiment

As shown in Figure 9, this case depicts a scaled model of an acrylic structure. The upper part of the structure simulates the air zone of a wind turbine structure, while the lower part simulates the underwater portion of the structure. The material of the scaled model structure is acrylic. Table 1 provides detailed parameters of the material for the

doi:10.6342/NTU202401548

scaled model structure, as shown below.

A stochastic input is provided in the point of T3, where we hang a double pendulum at the joint of T3 to provide an input that can impact the vibration of the full structure. The double pendulum swings in a way such that every time step, the magnitude of the vibration is different and then the energy is transferred to the structure to provide an output for the whole structure. The picture of the double pendulum is provided in the figure below.

The accelerometers are set up in the places highlighted in Figure 9, from ch1 to ch4. The accelerometers are hooked to a computer that utilized a data acquisition system to harvest the data in real time into excel files. The four channels are saved in the same file and then are processed together as a unit in the following section. The accelerometers

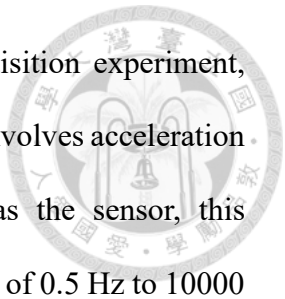
Table 1. Parameters of Acrylic Structure

Parameters	**Values**
Density	$1190\ {}^{kg}/_{m^3}$
Young's Modulus	$2.94\ GPa$
Poisson's Ratio	0.4
Total Height	170 cm
Component Size	Top Half：$0.5\ cm\ \times 15\ cm\ \times 5\ cm$ Bottom Half：$0.5\ cm\ \times 40\ cm\ \times 40\ cm$

Table 2. Measurement Equipment

Accelerometer	PCB-356C65
Data Acquisition System	NI cDAQ-9179

doi:10.6342/NTU202401548



Introduction to the relevant instruments used in the data acquisition experiment, including the brand name and purpose explanation. This experiment involves acceleration measurement, using the PCB-356C65 single-axis accelerometer as the sensor, this accelerometer is a piezoelectric accelerometer with a frequency range of 0.5 Hz to 10000 Hz, which better suits the needs of this experiment. This accelerometer comes with a transmission cable between the accelerometer and the signal acquisition card. The accelerometer is connected to the NI-9234 vibration signal acquisition card, and then the acquisition card is connected to the NI cDAQ-9179 chassis. LabView graphical control software is used for collecting the time-series data output for further implementation.

3.2.3 Signal Processing and Database Construction

The raw data is processed with a few procedures. First, the time series signal from the accelerometer is down-sampled from 1600 Hz to 30 Hz then Fast Fourier Transform (FFT) is applied on the data. Only the frequency until the Nyquist frequency is selected then the data from the four channels are concatenated in order from channel 1 to channel 4 and this returns a 1-dimensional array of size 1 x 600. This process is repeated for the whole folder containing the N cases in the same damage class, and the array are then concatenated in rows and returns a 2-dimensional array of size N x 600. After the transformation of all 6 damage classes is finished, the six cases are then combined into a big array of 6N x 601 with one more unit to save the label of the class. The whole array is then randomly shuffled and split into training and validation set with a 6:4 ratio. Then the big array is split into raw features (X_train, X_validation) and a 1D array for the corresponding labels (Y_train, Y_validation).

During the FT process, we shrunk the database based on some engineering considerations so that it would require less memory and potentially train the NN faster. We down-sampled the frequency from 1600 Hz to 30 Hz due to three reasons. The reason

doi:10.6342/NTU202401548

is the major modes of the structure only go up until 12 Hz, therefore we shouldn't need that much frequency content to have a good result, furthermore with so much frequency content it is less likely that we can find the modes that actually matter. Secondly, we also considered the opportunity of future works that could combine hybrid measurement methods such as visual inspection with UAVs and or mobile phones that could be deployed by on-site engineers. Given that most UAVs and consumer grade cameras have a sample frequency that only goes as high as 30 to 60 Hz, it is convenient to just down-sample the data to 30 Hz as it only takes half of down-sampling in the frequency domain and it would automatically match the frequency of our database. This additional step would be much helpful for easier data preparation and future implementations without sacrificing useful information.

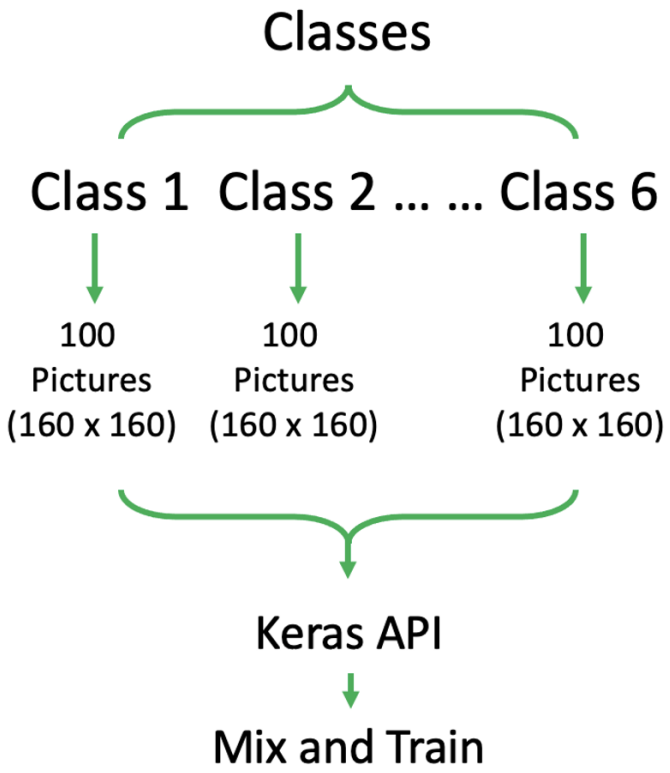


Figure 10. Data Preparation for Visual Accelerometer Data

doi:10.6342/NTU202401548

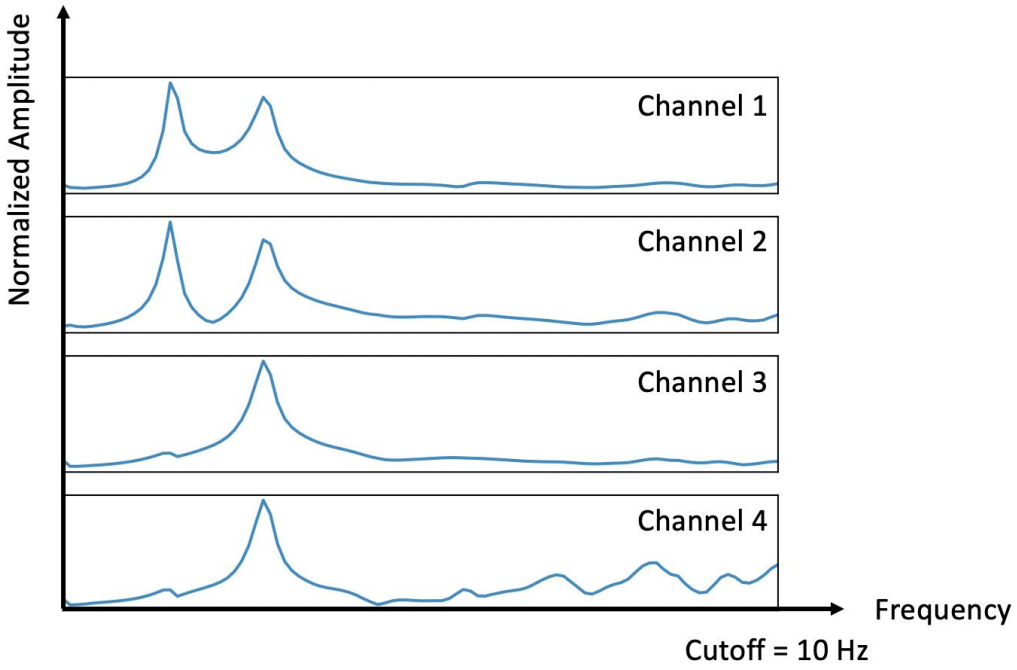


Figure 11. Visual Accelerometer Data for CNN Preparation

Although showcasing the difference of the damaged data and the undamaged data isn't the focus of this paper, the comparison is shown in the appendix. In order to make a basic comparison, the sum of the FFT coefficients of all the data of all of the same damage class are compared to the undamaged class.

3.3 Results and Discussions

3.3.1 CNN Results

In this section we have created a simple CNN for the task of image recognition for the accelerometer database. we are using the flexible functional API of Keras to build a ConvNet that can distinguish between 6 different categories. The functional API can handle models with non-linear topology, shared layers, and models with multiple inputs or outputs. Imagine that while the sequential API requires the model to move linearly between its layers, the functional API allows for more flexible operations. If the

doi:10.6342/NTU202401548

sequential approach is like a straight line, then the functional model is like a graph, where the nodes of the layers can connect in multiple ways. Figures 48 and 49 illustrate the architecture of this neural network.

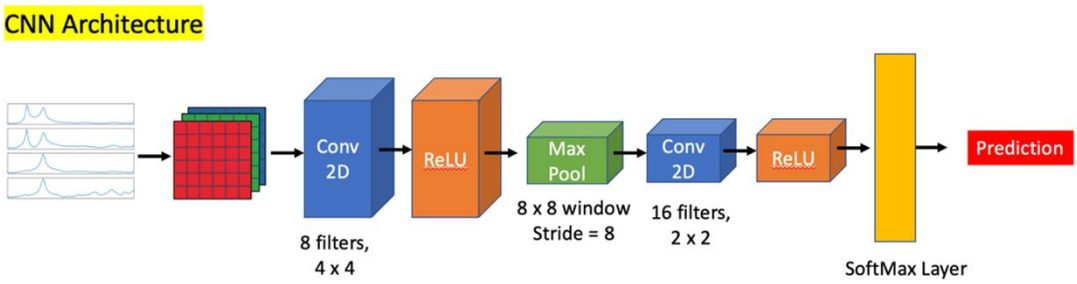


Figure 12. Picture of the structure of CNN

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 160, 160, 3)]     0

 conv2d_2 (Conv2D)           (None, 160, 160, 8)       392

 re_lu_2 (ReLU)              (None, 160, 160, 8)       0

 max_pooling2d_2 (MaxPooling  (None, 20, 20, 8)        0
 2D)

 conv2d_3 (Conv2D)           (None, 20, 20, 16)        528

 re_lu_3 (ReLU)              (None, 20, 20, 16)        0

 max_pooling2d_3 (MaxPooling  (None, 5, 5, 16)         0
 2D)

 flatten_1 (Flatten)         (None, 400)               0

 dense_1 (Dense)             (None, 6)                 2406

=================================================================
Total params: 3,326
Trainable params: 3,326
Non-trainable params: 0
_________________________________________________________________
```

Figure 13. Structure of CNN

The first layer is the input layer, which reduces the dimensionality of the original image (640×480) to (64 x 64) patterns (RGB). This neural network has two convolution blocks. The first block uses a convolutional layer, a learning layer, and a max-pooling layer. The second block also uses a convolutional layer, a learning layer, and a max-

doi:10.6342/NTU202401548

pooling layer. Afterward, it is flattened into a one-dimensional vector and used for predictions. The structure of the CNN is shown in Figure 13 and Figure 12. The structure is similar to LeNet 5 but simplified for this implementation.

```
Epoch 200/200
68/68 [==============================] - 7s 105ms/step - loss: 0.7622 - accuracy: 0.7046 - val_loss: 1.1007 - val_acc
uracy: 0.6300
CPU times: user 1h 58min 37s, sys: 20min 30s, total: 2h 19min 8s
Wall time: 25min 40s
```

Figure 14. Training time and result of CNN on Accelerometer Database

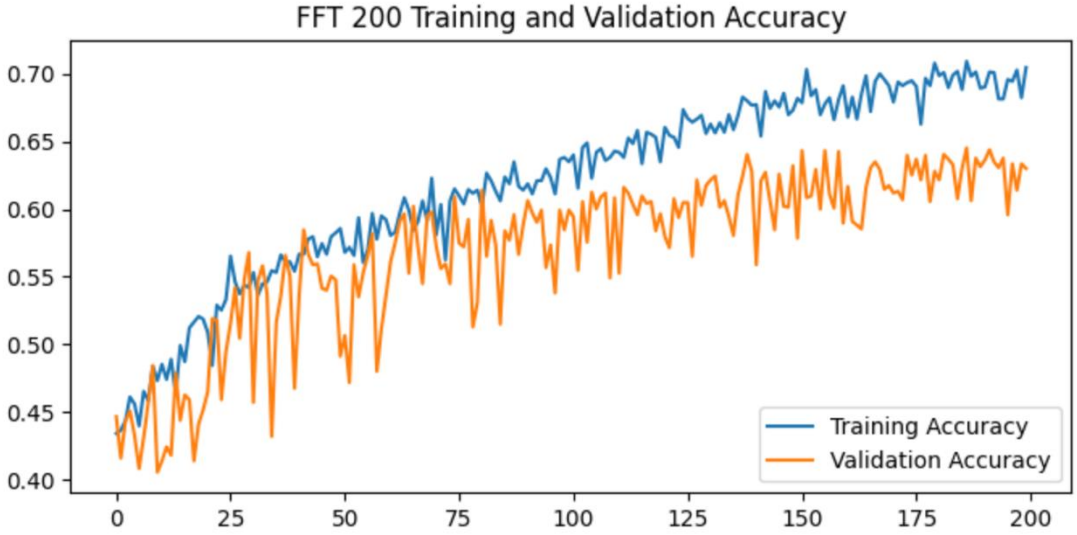


Figure 15. Training accuracy plot of CNN on accelerometer database

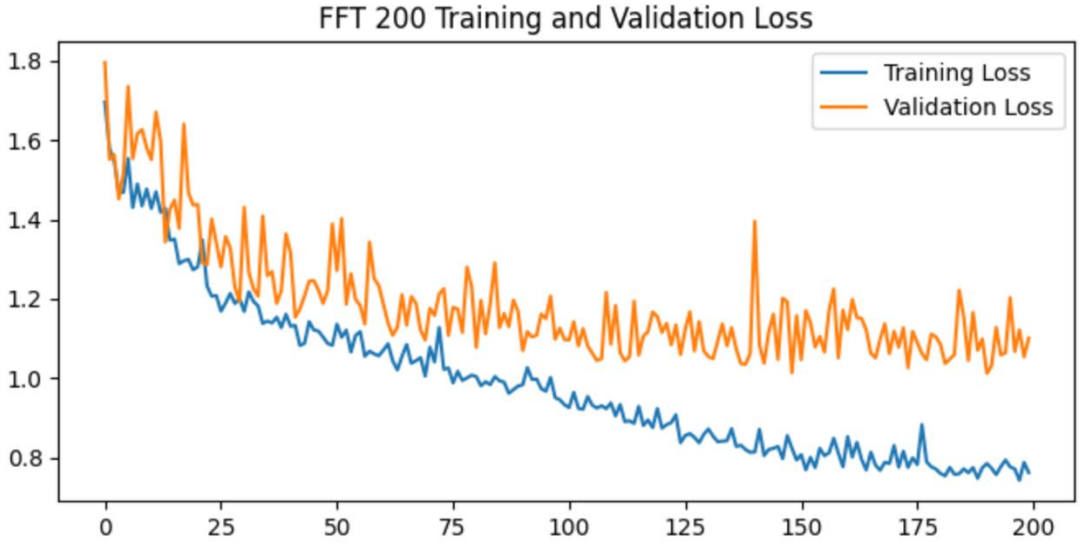


Figure 16. Training loss plot of CNN on accelerometer database

doi:10.6342/NTU202401548

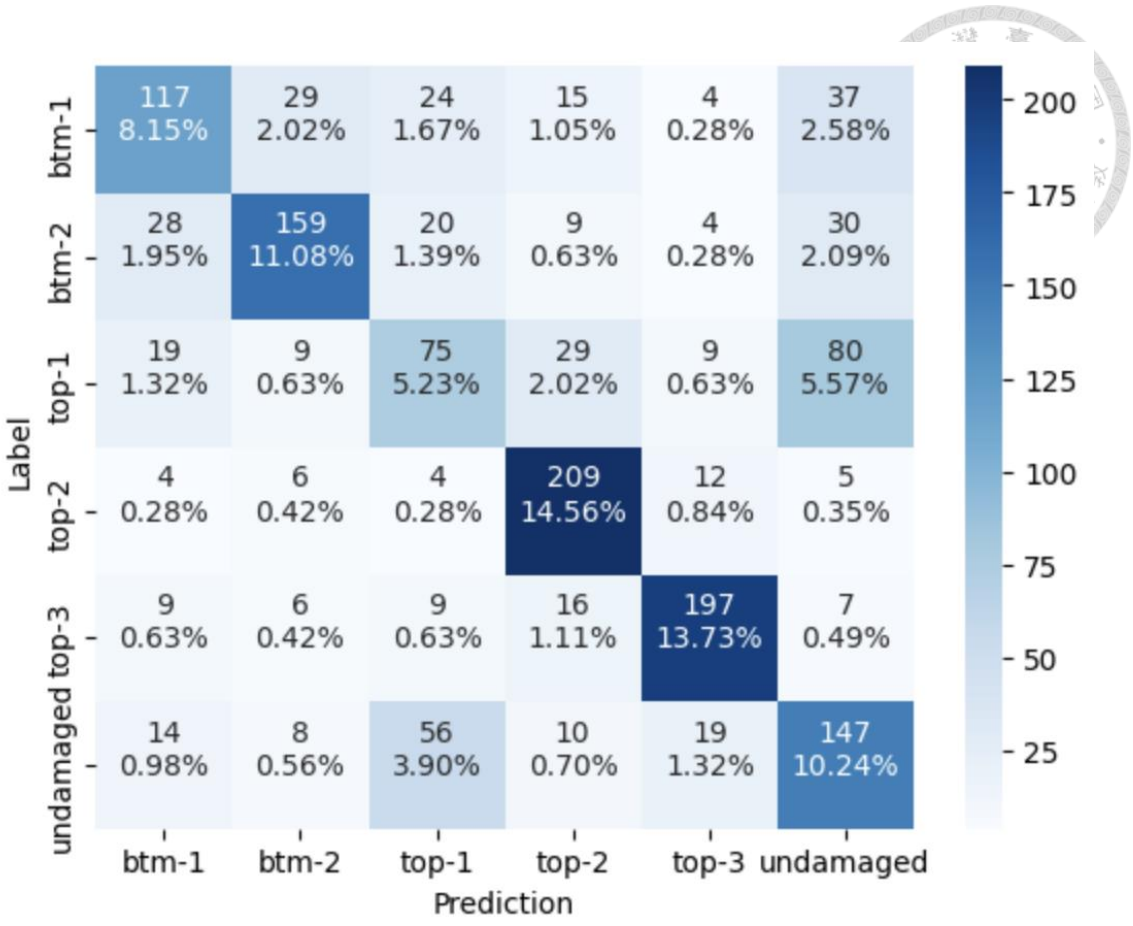


Figure 17. Confusion matrix of CNN before fine tuning

After achieving results that still have room for improvement, this section intends to return to the most primitive Fourier transform, hoping for better outcomes. Since directly normalizing the Fourier transform coefficients without taking the logarithm can enhance the significance of the main frequencies, we hope for better training results. In the previous stage, we observed overfitting starting to occur around 200 cycles, so in this training, we attempt to fine-tune the learning rate to improve accuracy. During the training process, we achieved approximately 0.67 training accuracy and approximately 0.58 validation accuracy. Additionally, the loss during training was 0.87, while the loss during validation was 1.11. The entire training process took 25 minutes and 40 seconds. These metrics reflect the performance of our model during training and validation. It is worth noting that the training accuracy is slightly higher than the validation accuracy, and the

doi:10.6342/NTU202401548

validation loss is slightly higher than the training loss, which may indicate some degree of overfitting. Further optimization of the model is necessary to improve performance.

After realizing the limitations of neural networks in fitting datasets, a finer training fine-tuning technique was implemented. I applied fine-tuning after 20 epochs on the previous results to achieve higher accuracy. Figure 18 shows the training accuracy and loss function. The sudden improvement in accuracy and loss when using a slower training rate (after the vertical green line) is normal because it represents a more refined and conservative approach towards reaching the minimum.

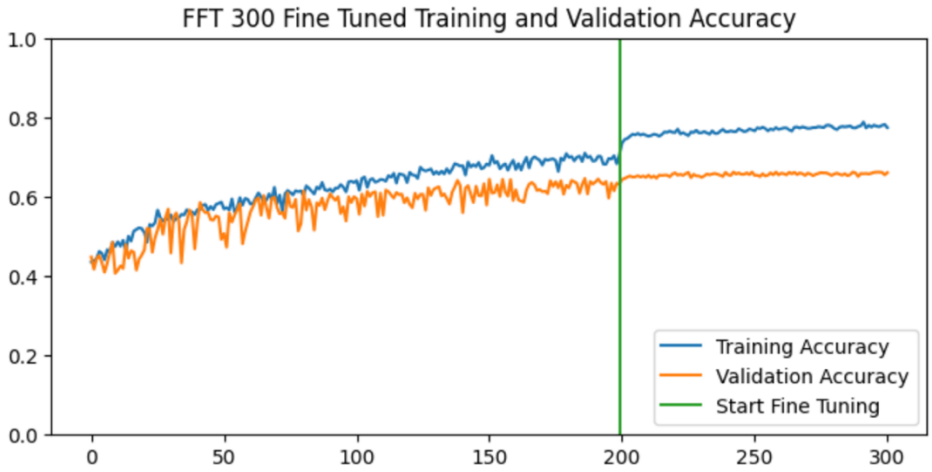


Figure 18. Training accuracy plot of CNN on accelerometer database after fine tuning

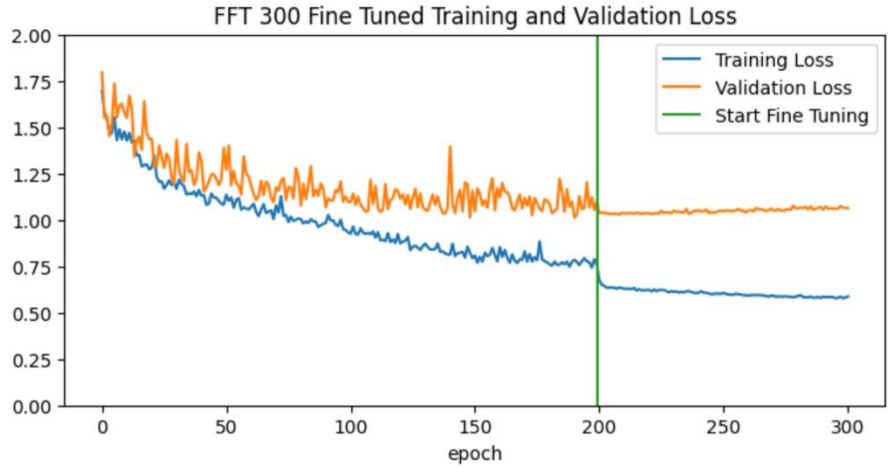


Figure 19. Training Loss plot of CNN on accelerometer database after fine tuning

doi:10.6342/NTU202401548

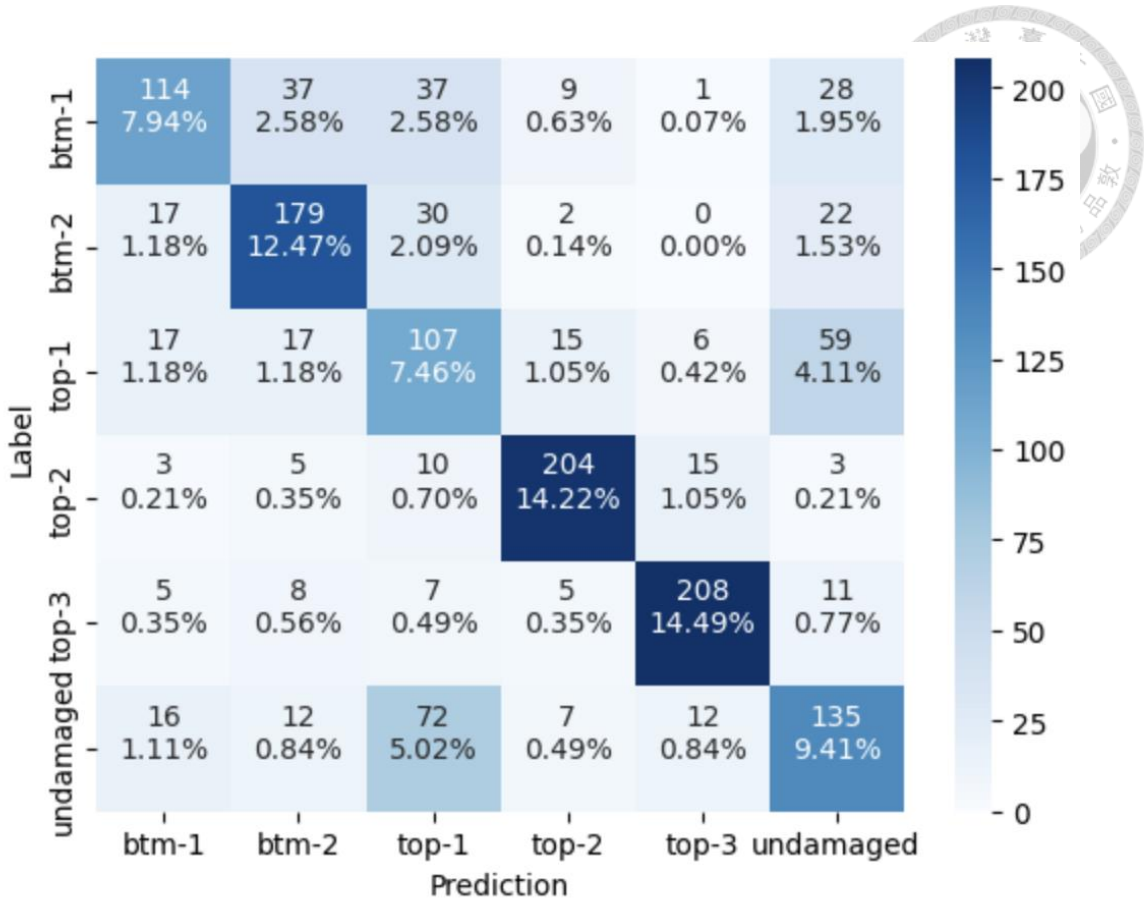


Figure 20. Confusion matrix of CNN after fine tuning

We can see that the performance on the btm-1, btm-2, and top-2 datasets is more consistent, so adopting finer training has led to limited improvements. Although the accuracy here is not significantly different from the data without fine-tuning, having more consistent results is a positive outcome.

3.3.2 DNN Training Results on Accelerometer Data

The neural network designed for this task is a one hidden layer neural network. The input layer has 600 units, which is the same data length as the reformatted data. There are 30 units for the hidden layer and 6 units for the output layer, corresponding to the 6 classes for classification. There are 18,126 total units in this DNN. ReLU function is used for the activation function and SoftMax function for the output layer. In the compiling process, Adam optimization is used. The full layout is shown in Figure 21.

doi:10.6342/NTU202401548

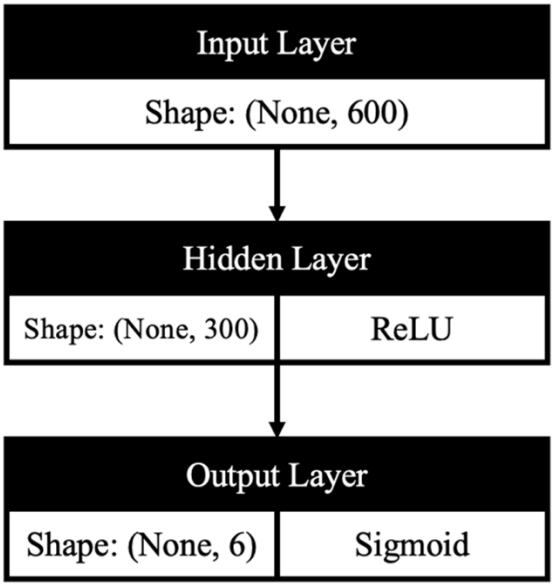


Figure 21. 1 Hidden Layer DNN Layout

The neural network is trained for 200 epochs with a constant learning rate of 10^{-3} with a total training time of 26.3 seconds. In the training set achieved 98% on the training set and 66% on the validation set. This indicates that the network could be overfitting the training set and results in a high variance problem. This implies that 18,000 units are enough for the network to fit the dataset, thus some regularization techniques could be used to simplify the network so that it could have better generalization towards the validation set. has shown a high identification rate between T2, T3 towards all damage classes, especially the F1 score of T2 dataset as high as 77.8% The performance between undamaged (UD) and top-1 (T1), bottom 1 (B1) and bottom 2 (B2) is not as ideal. The reason might be that T1 is only a small part of the whole structure and has little to no impact towards the result. B1 and B2 are located in a stiffer region, thus because of a smaller amplitude of vibration, the features are harder to be distinguished. This network has shown good performance towards data that are have a distinct difference and mediocre performance towards data that are hard to identify.

doi:10.6342/NTU202401548

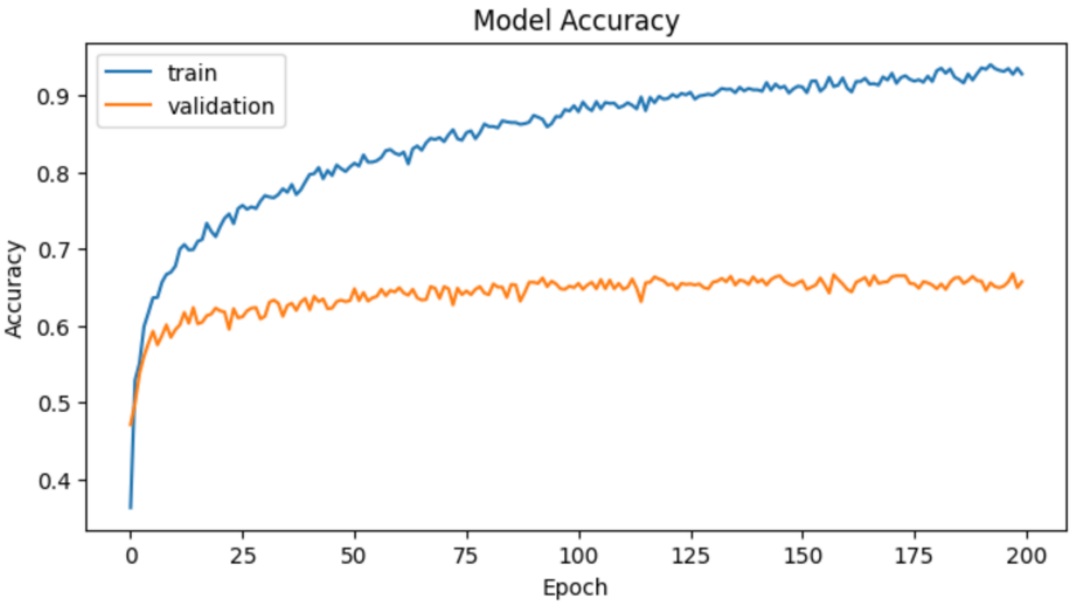


Figure 22. Model Accuracy of 1HL Model on Accelerometer Database

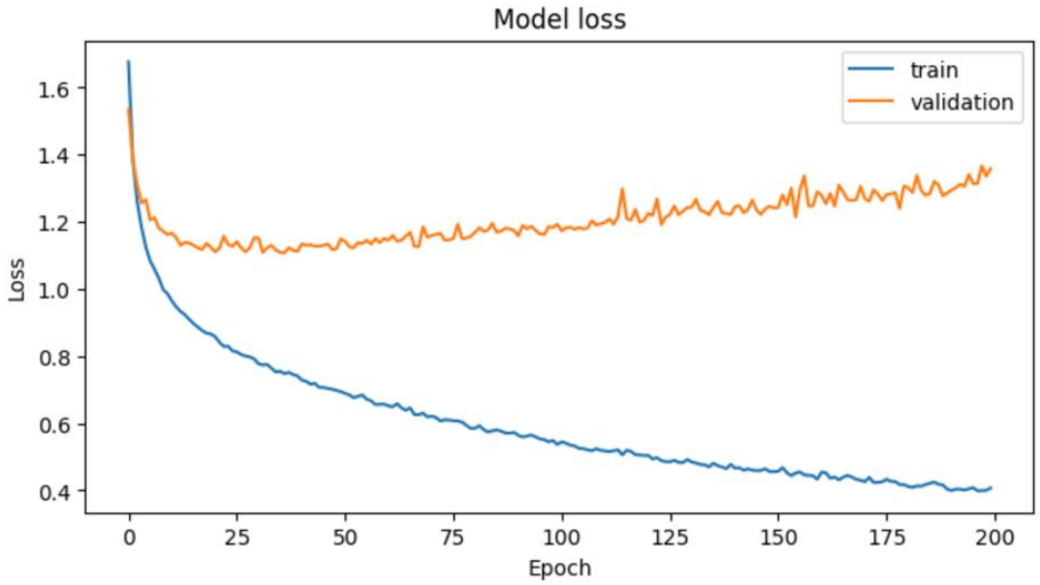


Figure 23. Model Loss of 1HL Model on Accelerometer Database

doi:10.6342/NTU202401548

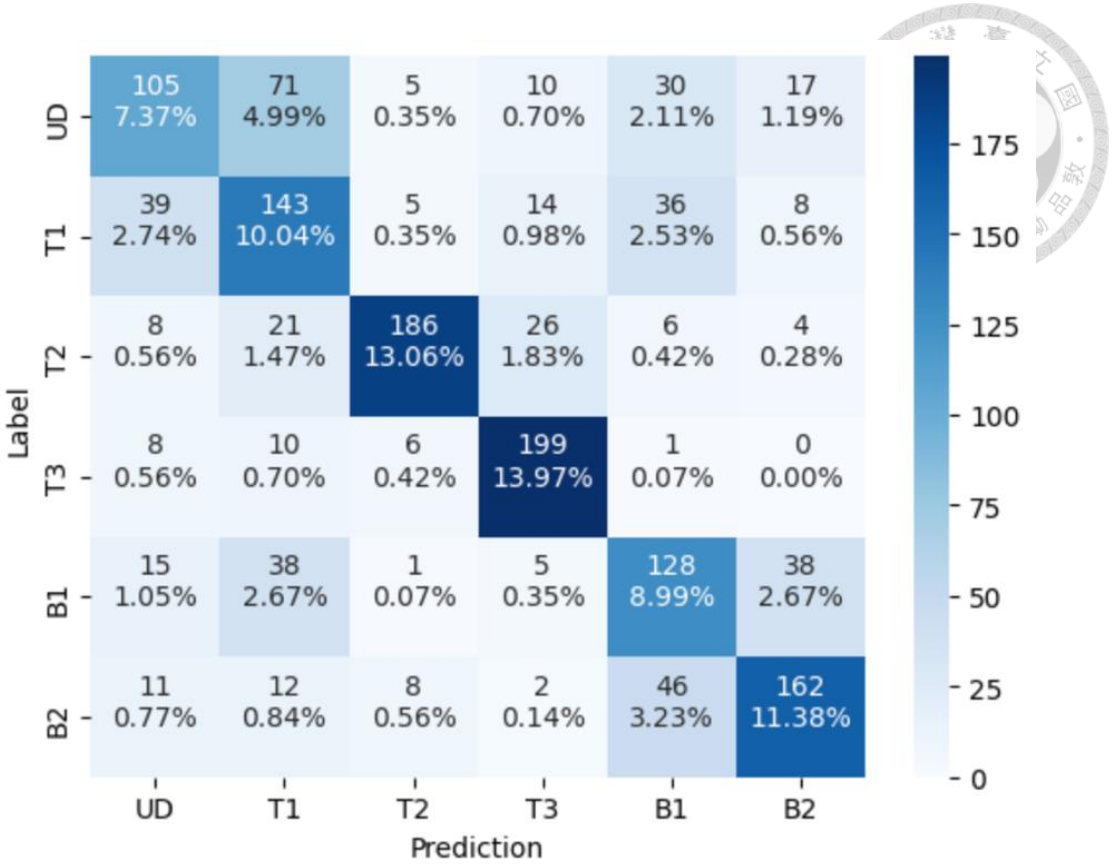


Figure 24. Confusion Matrix of 1HL DNN with original data set

Upon further examination, after implementing neural network optimization by using early stopping and regularization, it became evident that the first training with no optimization is flawed due to training for too long so that it overfitted the training set. Therefore, in the following training the base model should also have early stopping applied in the training procedure. Figure 25 shows an important finding in regards of the current neural network optimization, in this figure we are discovering the relation of hidden layer units and the effects of regularization on the accuracy. The deep and light blue dots represent the accuracy of the training set and validation set result of a model with a regularization value set extremely low that could be compared to the base model, therefore it is referenced as the base model in the following analysis. Because the models are created in a loop, therefore we have to pass on a non-zero value and this is why the regularization value R is not zero. The red dots are the model that have R = 0.001 and the

doi:10.6342/NTU202401548

green dots are the model that have R = 0.0033 which is almost three times as much as the previous model. The R and hidden values are fed as variables and are created as separate models using a nested loop. The learning rate is set to the standard 0.001 and the tolerance of early stopping is set to 5 epochs.

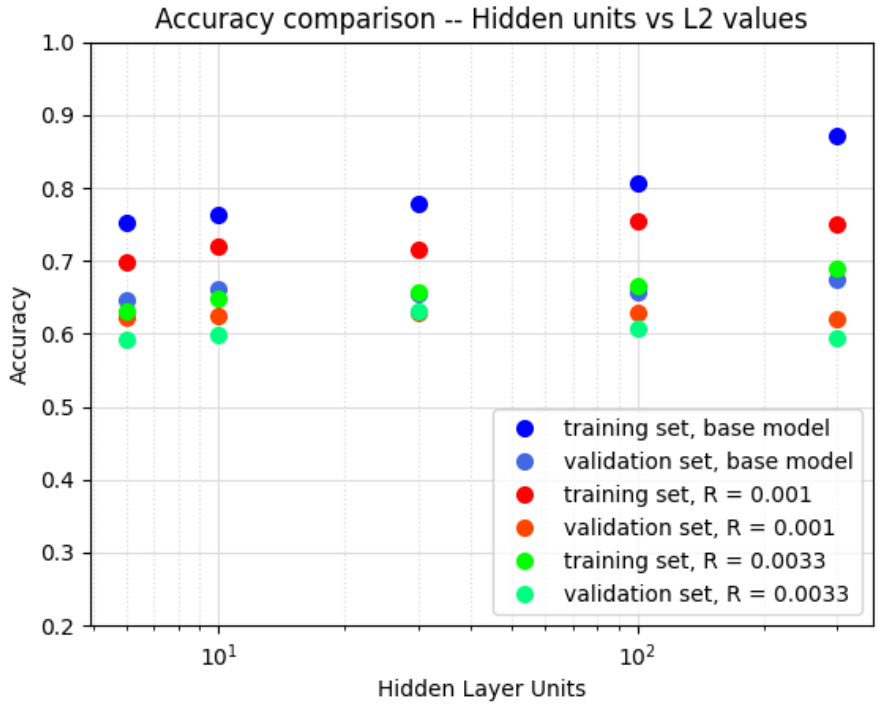


Figure 25. Accuracy comparison of Hidden Units and L2 Values

It is evident in the graph that with a higher R value, the regularization effect is more prominent thus reducing the variance but increasing the bias, this is also known as bias-variance tradeoff. Also, there is generally a trend of higher hidden layer units having a better accuracy in the training set across all R values in the training set. But when there are R values, the best validation results seem to peak at 30 units. The best training set and validation set accuracy of this whole graph is the base model which has little to none regularization and more units.

doi:10.6342/NTU202401548

Chapter 4 DNNs and Hybrid Database

4.1 Research Methods

4.1.1 Advanced Neural Network Optimization Methods

Regularization techniques are useful for high variance problems. In order to reduce overfitting, a penalty term λ can be added to the cost function to reduce the weight matrix in the whole network. In best practice, L2 regularization is used and it is shown as the latter term in the right-hand equation. It is also called a Frobenius Normalization (4-2), which is the squared sum of all the entry in the matrix that holds the weight of the l[th] layer. In the formula below (4-1), $w^{[l]}, b^{[l]}$ means all the weights and biases in the l[th] layer. By increasing the value of λ, most of the weights are forced near zero and the output of the activation function would land in a small range. This will minimize effects of some units and result in a simpler network that would be less prone to overfitting.

$$J(w^{[1]}, b^{[1]}, \dots, w^{[L]}, b^{[L]}) = \frac{1}{m} \sum_{i=1}^{n} L(\hat{y}, y) + \frac{\lambda}{2m} \sum_{l=1}^{L} \left\| w^{[l]} \right\|_F^2 \quad \text{4-1}$$

$$\left\| w^{[l]} \right\|_F^2 = \sum_{i=1}^{n^{[l-1]}} \sum_{j=1}^{n^{[l]}} \left(w_{ij} \right)^{[l]} \quad \text{4-2}$$

Another powerful regularization technique is Dropout, which randomly knocks out units of a neural network to reducing overfitting. Dropout and regular regularization both have the same philosophy of reducing the NN complexity but dropout directly ‘drops out’ units of a certain layer, in contrast to the regularization which only minimizes the effects of unimportant units. As shown in Figure 26, the NN with dropout has less units, almost similar to a regular linear regression model and this would reduce the overfitting of the

doi:10.6342/NTU202401548

NN. To implement in a NN a user usually specifies the ratio of the units that would be randomly knocked out during training.

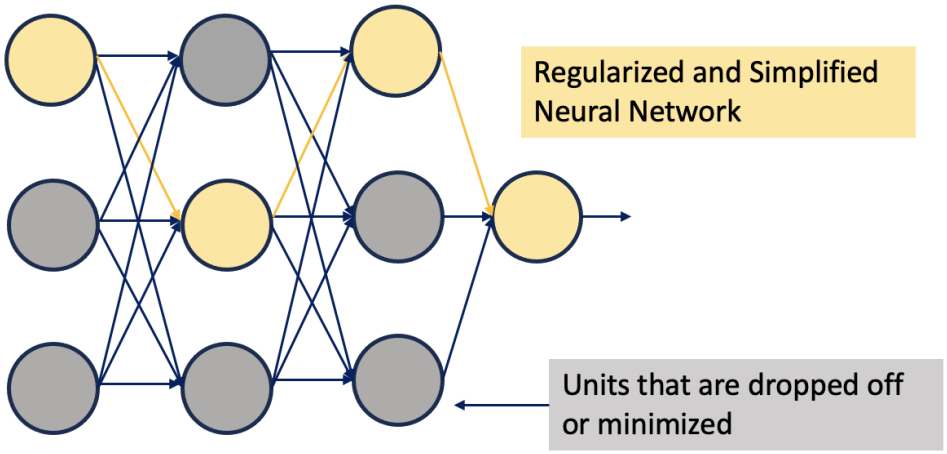


Figure 26. Concept of Dropout

To prevent a neural network from overfitting the training set, we would usually use early stopping to terminate the training to prevent the loss function of the validation set from exploding. This method can be easily used in Keras and it can be tuned by setting the tolerance of training epochs when we compile the training procedure. Early stopping is a call-back object that the training API examines after each training epoch, when we set the tolerance to a certain number, the API will check if the previous loss functions keep increasing after a certain number of epochs. We can also examine different training parameters such as loss function of the training set or the validation set, the latter one is the best practice when we want to prevent overfitting.

Based on the personal experience, when we use a smaller learning rate (0.0001), we can couple it with a smaller tolerance because there is only a small change in bias and variance, which leads to a smoother training curve. But when we use a larger learning rate (0.001), we need to set a higher tolerance because the turbulence of the loss function would be bigger, thus in some cases the Network would stop the training too early, thus

doi:10.6342/NTU202401548

yielding us unsatisfactory results due to underfitting. The following figure shows a similar graph as Figure 4 but applied appropriate early stopping. In order to examine the effect, Figure 7 shows the effect of different magnitude of regularization parameters on neural networks that have different hidden units (from 6 to 300). Figure 28 (a) and (b) is shown to demonstrate the relation between training rate and early stopping. The same model and same dataset is applied in both training, but in Figure 28.(a) a 100 times faster learning rate (0.001) is used and in Figure 28.(b) a much smaller learning rate is used (0.0001). In the faster learning rate case, the model can learn to the good-enough values faster, but if we restrict the early stopping to a value that is small, it will stop the model before it achieves the optimal value, therefore the tolerance of early stopping should be set to more epochs. In contrast since a model with a smaller learning rate has a smoother learning curve due to smaller gradient changes, if we set the tolerance too high, we would train the model too long and thus overfit out training model.

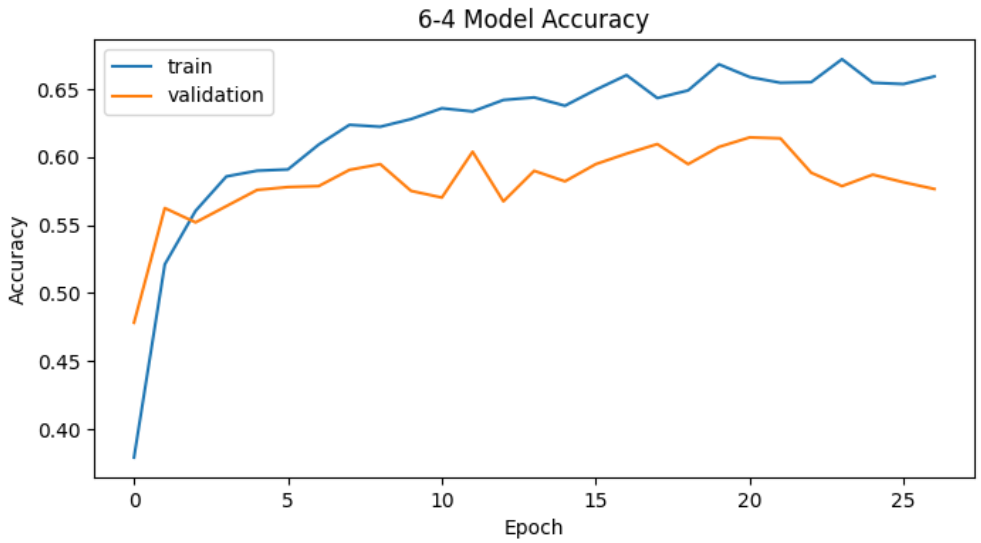


Figure 27. Demonstration of a learning rate of 0.001

doi:10.6342/NTU202401548

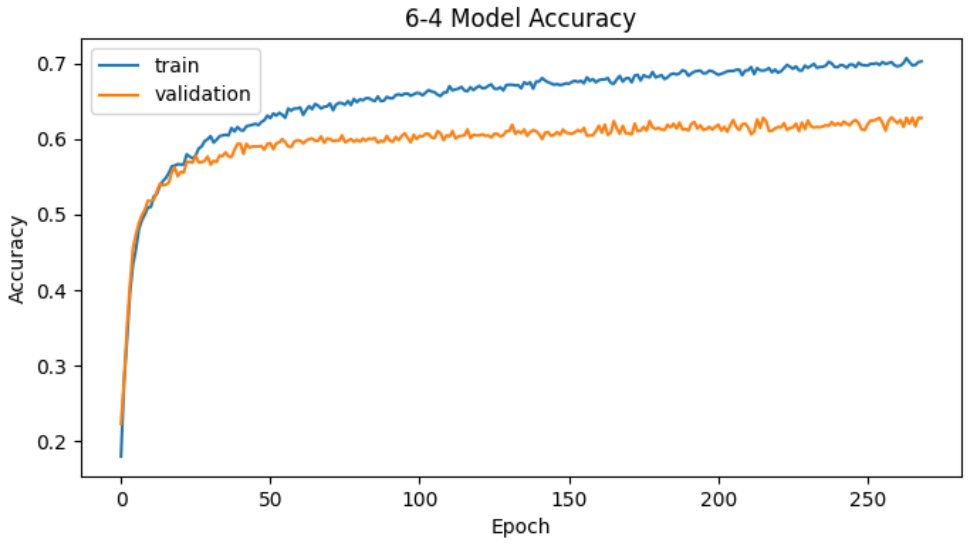


Figure 28. Demonstration of a learning rate of 0.0001

4.1.2 Data Augmentation

Data Augmentation is a common practice to increase neural network performance by creating artificial data for the neural network for training. Deep Learning is an algorithm that highly leverages its accuracy and generalizability on the quantity and quality of the data. Data Augmentation can be achieved by introducing some permutations of the original data such that the neural network can learn the features of the same object in different scenarios.

In the CV field, the commonly used CNNs for visual identification require a significant amount of data so that it can learn features from thousands of different classes. Acquiring such a significant amount of data for learning could be a costly but inevitable procedure for the performance of the neural network. The standard procedure of image data augmentation includes flipping, zooming, cropping, and changing color scales (Figure 29). Other advanced methods include mixing images, random erasing much more researches are documented in a detailed survey is done by Connor Shorten [46].

doi:10.6342/NTU202401548

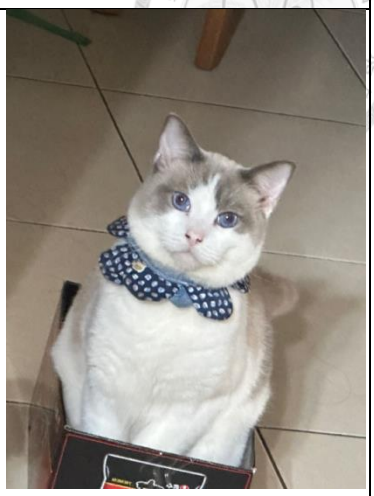
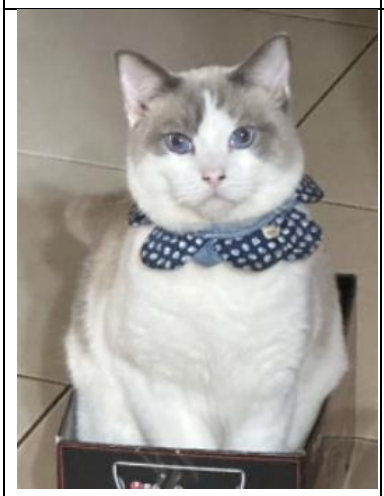
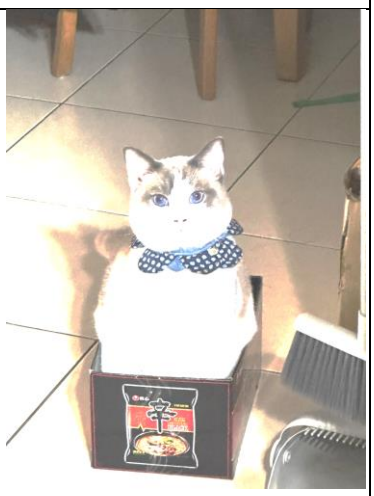
(a) Original	(b) Flipping	(c) Rotating
		
(d) Zooming in	(e) Warping	(f) Changing exposure
		

Figure 29. Different examples of image augmentation (a) Original , (b) Flipping, (c) Rotating, (d) Zooming in, (e) Warping, (f) Changing exposure

In the audio recognition field, researchers often leverage time and frequency domain permutations of the data to create more variations of the same data. A couple of commonly used tactic in the time domain are slicing (window cropping) and noise injection. Window cropping [47] samples a slice of the data with the length of the sub-sample as a tune-able parameter along with the original labels while noise injection [48] mixes a small number of different noises such as spikes, white noises, and slope-like trends to the original data.

doi:10.6342/NTU202401548

There are also attempts of window-warping (up-sampling and down-sampling) to create a brand new training example, but it is also stated in that this procedure will change the length of the window so the deep learning models need to also be modified [49].

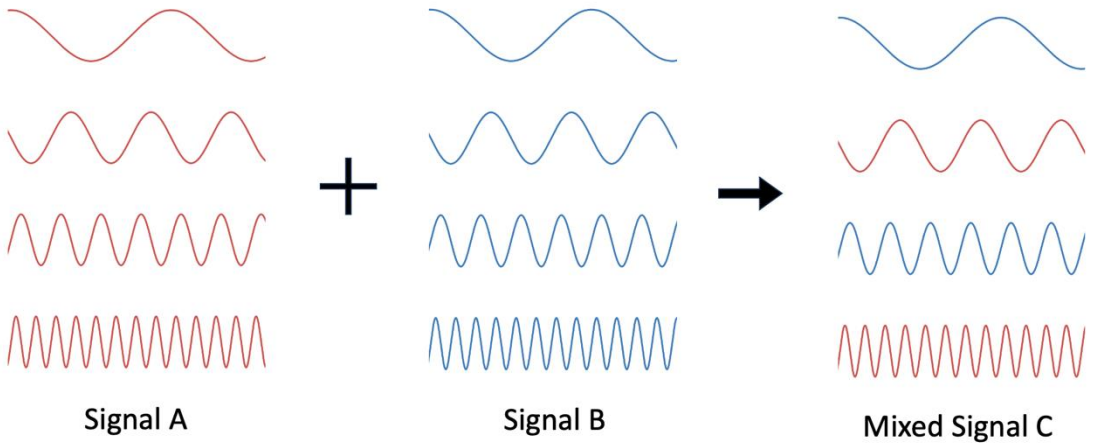


Figure 30. Frequency Mixing

Frequency domain augmentation primarily performs augmentation steps on the spectra that the NN trains from and according to experiments done by A. Le Guennec [47] it can perform better on time domain augmentations. Frequency masking is a commonly used procedure that masks a part of the spectra with zero values or Gaussian distribution to create new data, in Figure 31 the original data is plotted blue and the orange line is the masked data. The spectrum is divided into different frequency segments the frequencies are masked within the segment to artificially create similar data of the same kind. The reason that makes this method work is that the NN has more opportunities to learn the same features of the same set of data for longer, thus guaranteeing us a more accurate result. Another method is to mix different frequency components of different examples to generate an artificially mixed signal of different components (Figure 30). The theory of how this works is the same as the frequency masking, yet the theoretically the amount of data that could be generated in the mixing procedure because the combinations generated

doi:10.6342/NTU202401548

from thousands of data and segmented ways are almost endless.

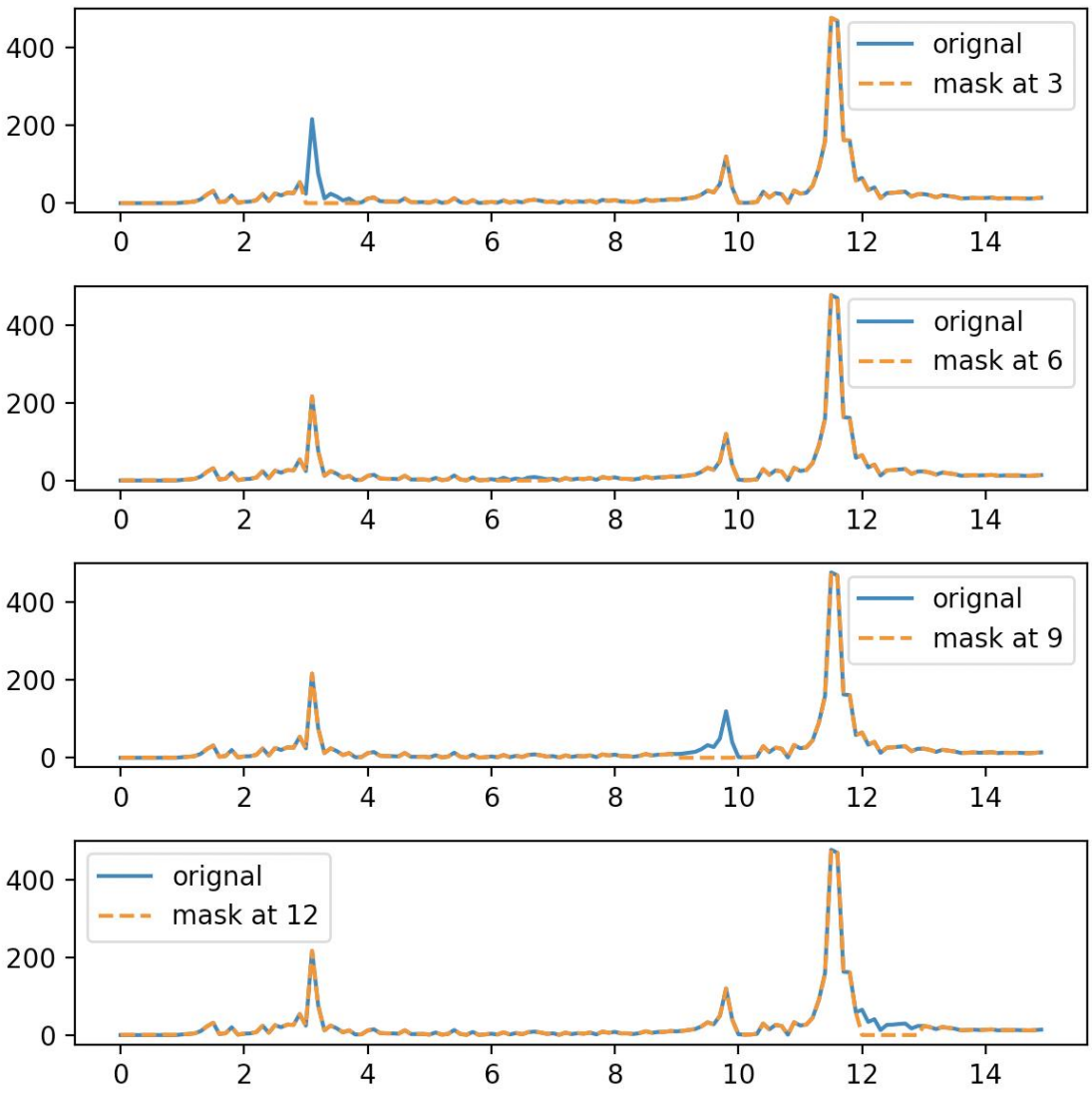


Figure 31. Frequency Masking Demonstration

4.1.3 Sample Rate Conversion

In order to match the different sampling rate of different equipment, sample rate conversion needs to be applied so that the feature extractor can match inputs from different sources. Sample rate conversion refers to the process of changing the sample rate of a discrete digital signal to obtain a new discrete representation of the same continuous signal with a different sample rate. Down-sampling is the process of

doi:10.6342/NTU202401548

converting a lower sample rate, and Up-sampling is the process of converting to a higher sample rate. The scope of this research will only cover the details of down sampling. which is half of the sample rate, in our reconstructed frequency without aliasing.

$$f_{sample} \geq 2 * f_{Nyquist} \quad \text{4-3}$$

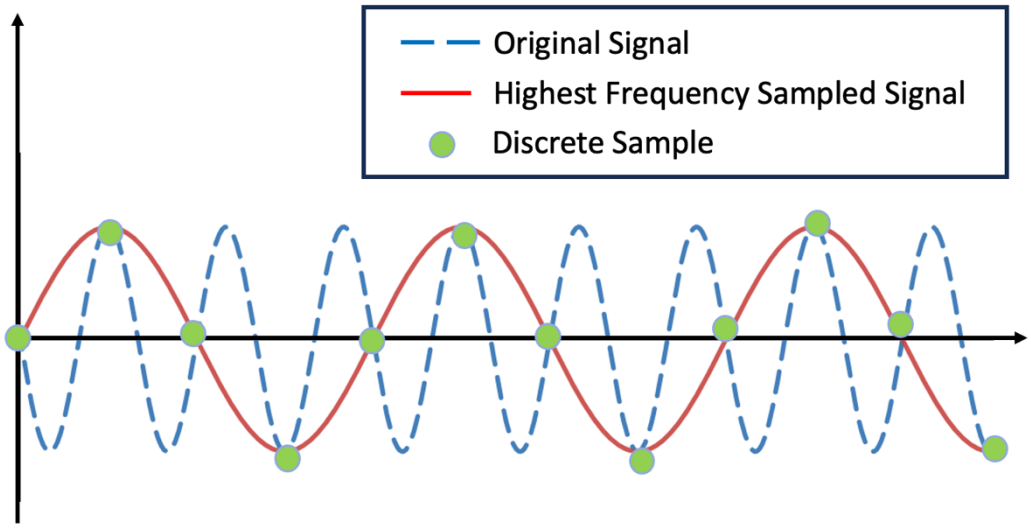


Figure 32. Nyquist Frequecny Sampling Example

Down-sampling can be useful in compressing data by reducing the density of it, it will help us obtain a close approximation of the same signal at a lower rate. Traditional down-sampling is usually done by scaling down in an integer factor, this process is also known as decimation (keeping only the 10^{th} signal). There are two steps to reduce a signal by the factor of M, the first step is to reduce the high-frequency component by passing the signal by a filter and the second step is to decimate the filtered signal by a factor of M by keeping the n x M^{th} example. The second step produces aliasing, a phenomenon in digital processing that overlaps the frequency components below the Nyquist rate, and distorts the original continuous signal. According to the Nyquist sampling theorem, the sampling rate should be double of the max frequency that we are interested to sample, the exact sample rate is called the Nyquist rate. This means that we can only get to keep frequencies

doi:10.6342/NTU202401548

less than the Nyquist frequency,

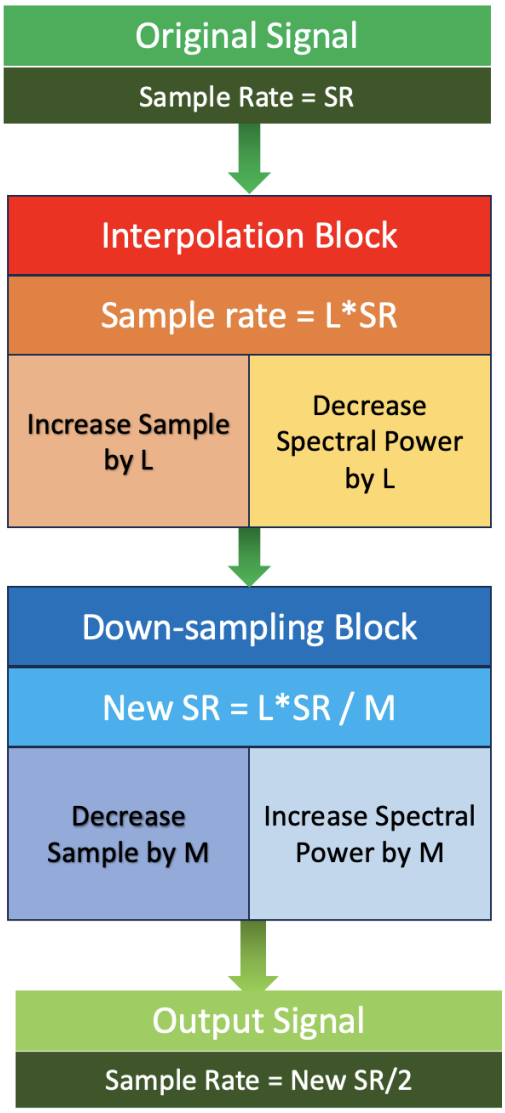


Figure 33. Sample Rate Conversion calculation process

In order to convert to an arbitrary sample rate, we would need to use an up-sampling block that performs interpolation of the signal and down-sampling that uses decimation or Finite Impulse Response (FIR). In the MATLAB documents, to perform a re-sampling procedure that converts with an rational number L/M of a continuous signal, the up-sampler increases the sample rate of the signal by a factor L and the down-sampler reduces the sample rate of the signal by a factor M. Use up-sampling and down-sampling factors that are relatively prime or coprime. The resulting discrete-time signal has a sample rate

doi:10.6342/NTU202401548

that is L/M times the original sample rate. Sandwiched between the up-sampling block and down-sampling block is an anti-imaging and anti-aliasing FIR filter which aims to reduce aliasing by passing the signal through a low pass filter. Figure 34 is an example of re-sampling a 100 Hz sine wave signal by a factor of 3/2, every 3 output samples are matched because of the refactor rate.

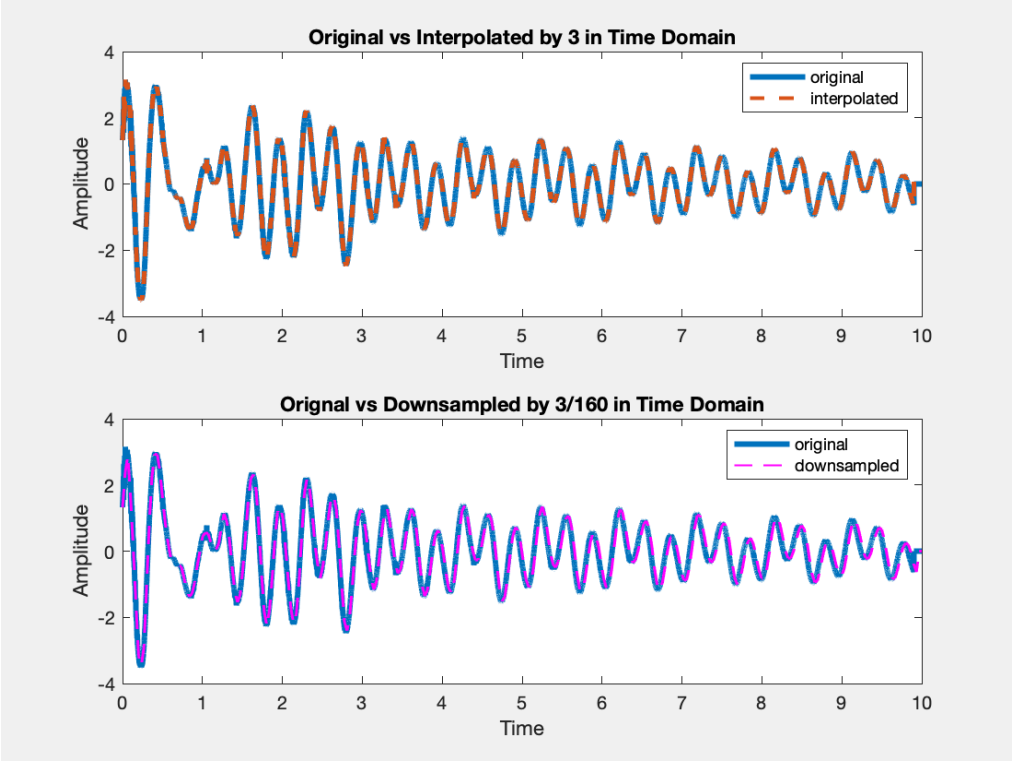


Figure 34. (a) Interpolated signal plotted against the original signal (b) Resampled signal plotted against the original signal

The following example showcases the reliability of the re-sampled data with a sample data from the acceleration database. A single channel of sample frequency of 1600 Hz is first interpolated by 3 times and it is plotted against the original discrete signal (Figure 6 (a)), it is worth noting that the current highest frequency that we can accurately sample is still 800 Hz which is the same as the Nyquist frequency of the original signal. The up-sampled data is then down-sampled by 160 times so that it reaches the re-sample rate of 3/160 and it is plotted against the original case (Figure 6 (b)). It is pretty evident

doi:10.6342/NTU202401548

that the resampled data clearly fits perfectly to the original data.

In the frequency domain Figure 35 it also fits perfectly, Figure 35 (a) showcases the interpolated signal against the original signal and Figure 35 (b) plots the re-sampled signal against the original signal. It is worth noting that by increasing the sample size by 3 times in the interpolation procedure, the estimated spectral also has 3 times energy, thus the spectrum energy is down-scaled 3 times. The case is also true for the down-sampled data, the spectrum energy is lowered for 160 times compared to the interpolated signal, thus it is re-scaled up to 160 times so that the energy under the spectrum can match.

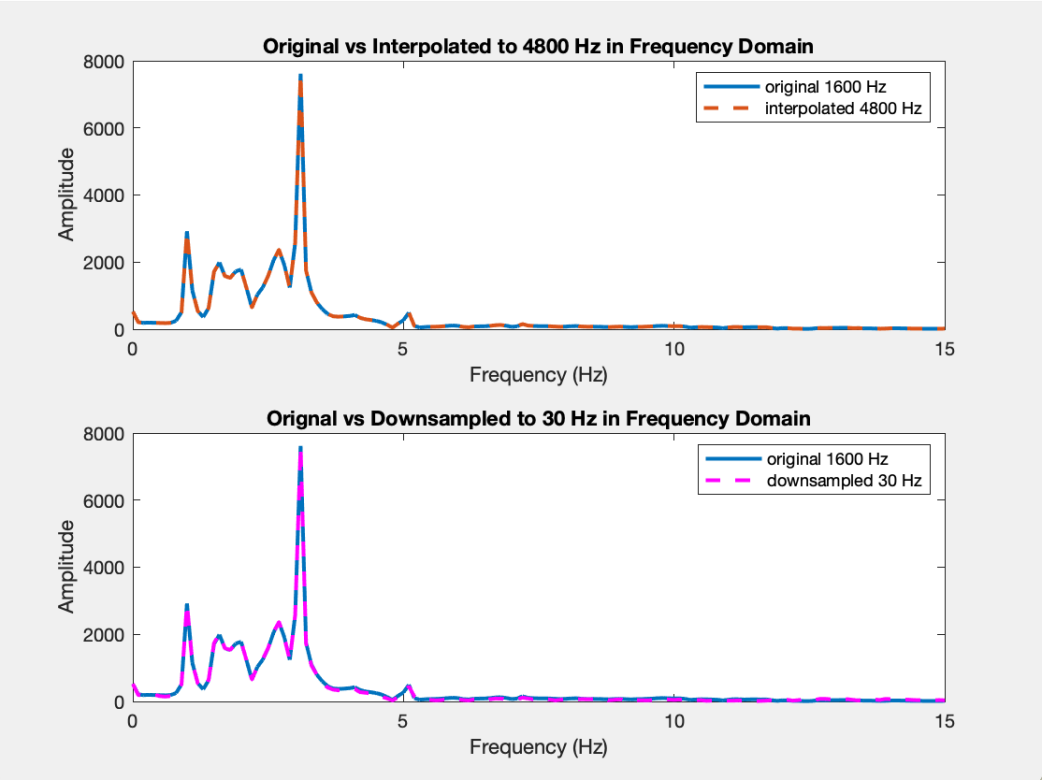


Figure 35. Frequency component comparison of the interpolated and resampled signal. (a) Interpolated signal plotted against the original signal (b) Resampled signal plotted against the original signal

doi:10.6342/NTU202401548

4.2 Construction of Hybrid Database

4.2.1 Finite Element Model

The finite element model is constructed using COMSOL Multiphysics and the properties of the materials and boundary conditions are listed in the form below. In order to fully mimic the conditions of the experiment setup, a time-dependent analysis method is used in the structural module. The module is capable to conduct eigenvalue analysis to extract natural frequencies and mode shapes, and it is also capable to extract time series data. The digital model is constructed in a 1:1 ratio as the physical structure and the boundary conditions are set to be completely rigid.

To verify the structural frequency natural frequency of the structure, we analyzed the first three eigen-frequencies of the structure. To compensate the metal bolts in the structure, density and stiffness are slightly changes are made so that the natural frequencies of the FEM structure can be more similar to the physical structure and the changes are listed in Table 3. The first three natural frequencies are 1.45, 2.65, 3.11 Hz and the percentage difference of between the physical model structure is 3.3%, 1.9% and 0%, with 3.11 Hz being the greatest natural frequency component. The first three dominant mode shapes are shown in the following Figure 36 the 1st and 3rd modes vibrate in the Y-Z Plane and the and the 2nd mode vibrates in the X-Z plane, but since the structure is only stimulated in the Y direction, the 2nd mode should be insufficiently excited. The full layout it shown in Figure 36 (a), (b), (c).

doi:10.6342/NTU202401548

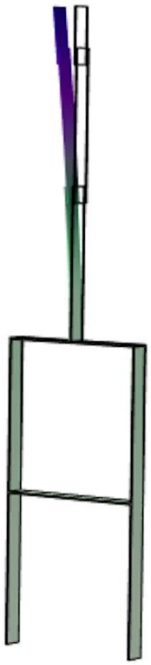
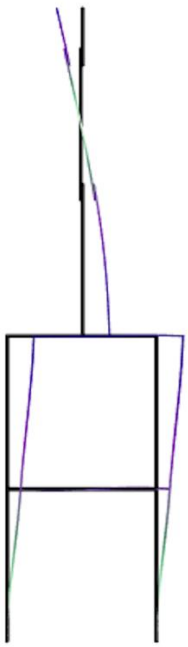
1st Mode 1.5 Hz	2nd Mode 2.3 Hz	3rd Mode 3.0 Hz
		

Figure 36. (a) 1st dominant mode shape, (b) 2nd dominant mode shape,

(c) 3rd dominant mode shape

A random vibration is set on the middle part of the structure to induce a random vibration in every time step analysis. The software will then output the corresponding acceleration data in 4 probe points, which are the same points that the accelerometers are placed. The FEM model will calculate the acceleration output of each time step and output a series of time-series acceleration data of the four channels. A sample of the random input load is shown in Figure 37 (a) below, the load is then applied to the red arrow in the structure below in Figure 38 and the output of the four channels are showcased in Figure 37(b)

doi:10.6342/NTU202401548

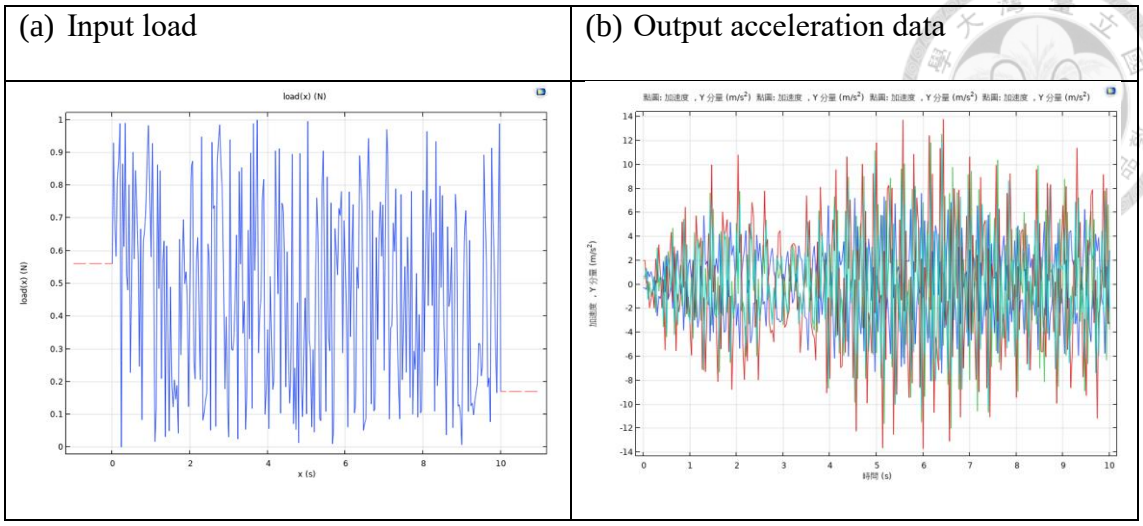


Figure 37. (a) Input load (b) Output acceleration data

The damage cases are inflicted by replacing the beams in model with one that has less stiffness, it is set to be full acrylic and also fully rigidly connected to the rest of the structure, this technique is inspired from researches done on numerical models [26] [27]. The purpose of this set-up is to change the vibrational outputs and mode shapes in a different way by using a softer material which stiffness is chosen as 50% of the original stiffness to simulate a stiffness reduction regionally.

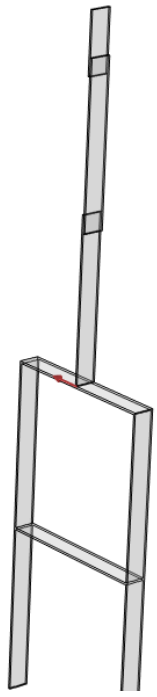


Figure 38. Point load on the structure

doi:10.6342/NTU202401548

Table 3. Material properties of the Acrylic model

Upper Structure	
Density	1300 kg/m^3
Young's Modulus	$3.2*e^9$ (Pa)
Poisson's Ratio	0.35
Heat Expansion Coefficient	$7*e^{-5}$ (1/K)
Lower Structure	
Density	1450 kg/m^3
Young's Modulus	$3.1*e^9$ (Pa)
Poisson's Ratio	0.35
Heat Expansion Coefficient	$7*e^{-5}$ (1/K)

To continuously generate huge amounts of data, we used MATLAB to control the COMSOL FEM model using the built-in launching code and ran our results in a loop. The full procedure is shown in the figure below: The first step is to convert the FEM model into MATLAB code by saving it as a .mat file, then transform the code into a function that could accept a variable for file saving. The random excitation input is also written in a code so that it will generate a new file and overwrite the previous file in the path of the model such that every time a new time-series excitation is generated. The function will then read the excitation array in the pathway then generate the time-series acceleration of the four channels into a separate excel file.

doi:10.6342/NTU202401548

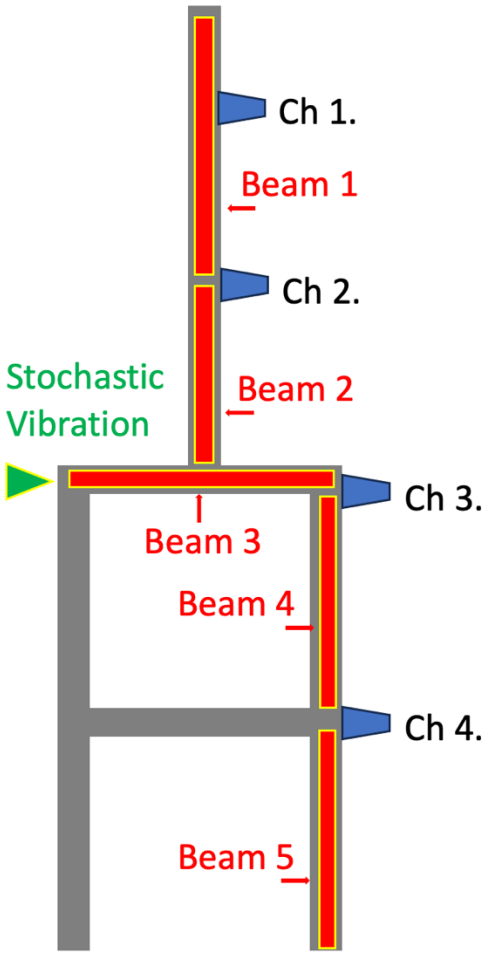


Figure 39.Beam damage cases

In order to complement the damage conditions from the accelerometer data, the FEM database is designed with a different strategy with different damage conditions. … shows the damage location of the FEM model. B1 to B5 represent a stiffness reduction throughout the whole beam. A stiffness reduction locally would be able to cause a frequency shift in local peaks at some natural frequencies. Ch1 to Ch4 are placed in the same location as the previous experiment and they are set as nodes that would output acceleration signals that could be saved as csv files.

In order to demonstrate the effects of stiffness reduction of the model, Figure 40, Figure 41 showcased the beam damage compared to undamaged scenario. The magnitudes are the sum of all 100 cases of the simulation and we can see that there are

doi:10.6342/NTU202401548

some peak shifts and damping phenomenon that occurs when we reduce the beam stiffness at different locations.

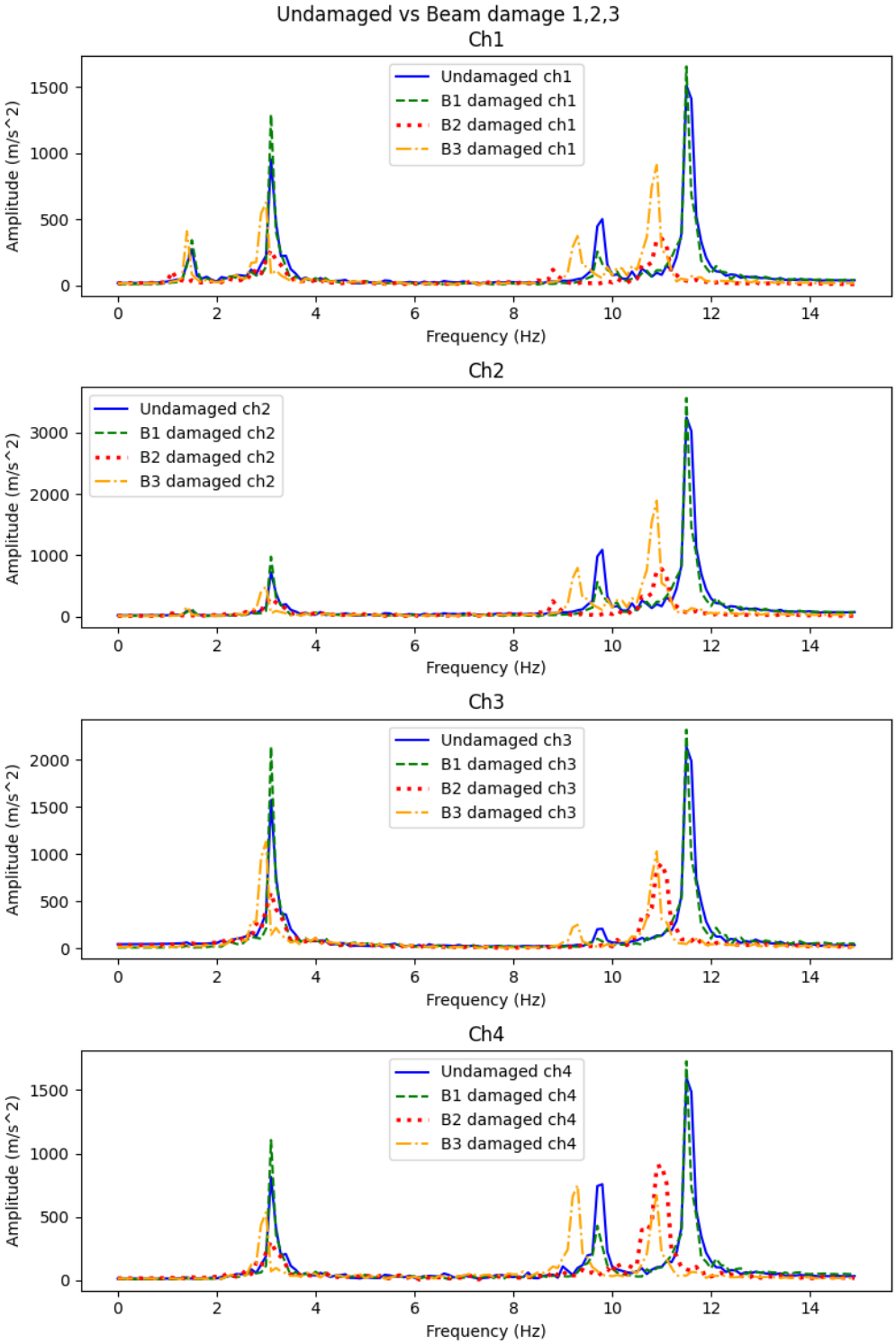


Figure 40. Beam damage 1,2,3

doi:10.6342/NTU202401548

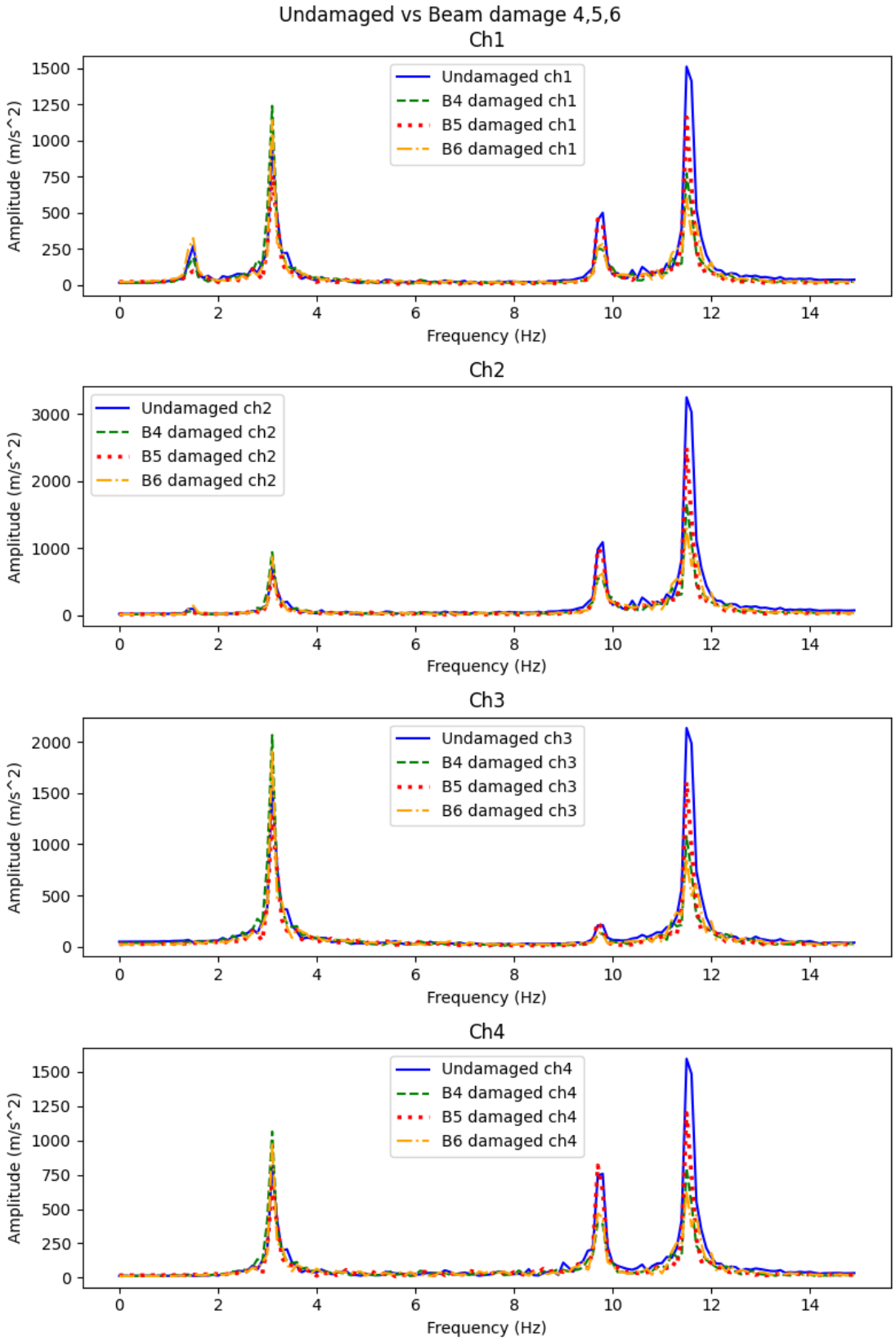


Figure 41. Beam damage 4,5,6

doi:10.6342/NTU202401548

4.2.2 Hybrid Database Preparation

In order to feed the NN the data from the accelerometer and the FEM results, we attempt to use a hybrid database to join the results of the two databases. Firstly, the FEM data would need to be transformed into a Numpy object that contains all of the Fourier coefficients in the same damage case. Each example contains four channels but joined into an 1D array column-wise and they are joined from channel 1 to channel 4 corresponding from top to bottom of where the accelerometers are placed on the structure.

During the FT process, considerate changes have been made to the accelerometer database to shrink the database so that it would require less memory and potentially train the NN faster. In this implementation we didn't need to down-sample instead we just set the FEM software to calculate only 300 datapoints per example. The benefit is now shown because if we require less datapoints per example, the time-based analysis on the FEM model can generate huge amounts of data faster because it the time for calculation is directly correlated to each time step.

The FT data of the FEM are collected in the same procedure as the Accelerometer data, each example of the 5 damage cases is combined into a .np file and there are 5 files corresponding to the 6 classes after this step. Then the FEM data are joined directly behind with the Accelerometer data to create a hybrid database, in this step the examples are not mixed yet. Then after this additional step, it follows the same procedure as the previous database and it would give us a fully-mixed hybrid database with mixed cases. The neural network will then train on this database.

4.3 1 Hidden Layer Fully Connected DNN

The neural network designed for this task is a one hidden layer neural network. The input layer has 600 units, which is the same data length as the reformatted data. There are

doi:10.6342/NTU202401548

30 units for the hidden layer and 6 units for the output layer, corresponding to the 6 classes for classification. There are 18,126 total units in this DNN. ReLU function is used for the activation function and SoftMax function for the output layer. In the compiling process, Adam optimization is used. The full layout is shown in Figure 42.

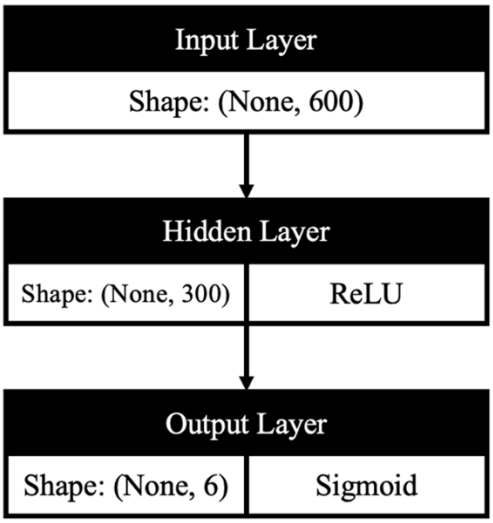


Figure 42. 1 Hidden Layer DNN Layout

4.3.1 Training on augmented Accelerometer Database

The neural network is trained for 50 epochs with epochs with a constant learning rate of 10^{-3} with an early stopping of 5 epochs monitoring the loss difference between the train-validation difference. In the accuracy plot in Figure 44 (b), the training set achieved 97.4% on the training set and 96.1% on the validation set. This indicates that the network fits both the training and validation set really well. This implies that with only one hidden layer, it is enough for the network to fit the dataset, thus some regularization techniques could be used to simplify the network so that it could have better generalization towards the validation set. It is evident that the increase of data size contributed the most in boosting the training results by a large margin because the NN can have more time to learn the intricacies of the damage cases. It not only boosted the accuracy of the neural

doi:10.6342/NTU202401548

network, it also contributed to less variance, which means that the network has more generalizability over different datasets. By this finding we can conclude that the poor performance of the previous network stems from the lack of data, instead of the complexity of the network itself, and performing optimization on a network that has limited data only further compromises the efficacy of the network since the accuracy should be high enough for regularization to be conducted to lower the variance.

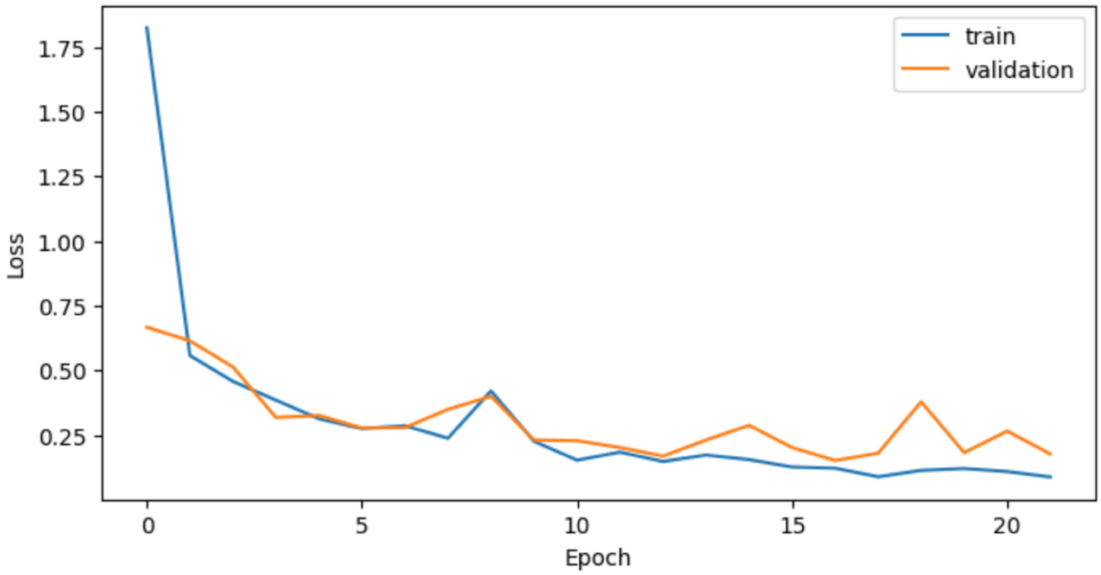


Figure 43. Loss function of the 1HL NN with data augmentation

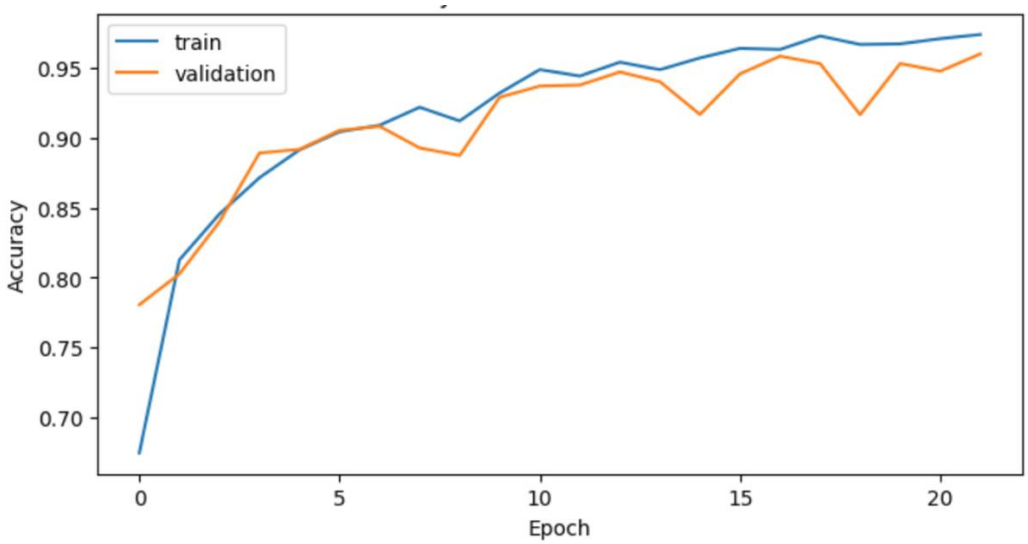


Figure 44. Loss and Accuracy plot of 1HL NN

doi:10.6342/NTU202401548

Upon further analysis on the confusion matrix of the validation set, we can see that in Figure 45 has shown a high identification rate towards all damage classes, although the performance between undamaged and Point 4 is not as ideal as the other cases, with over 100 misidentified cases.

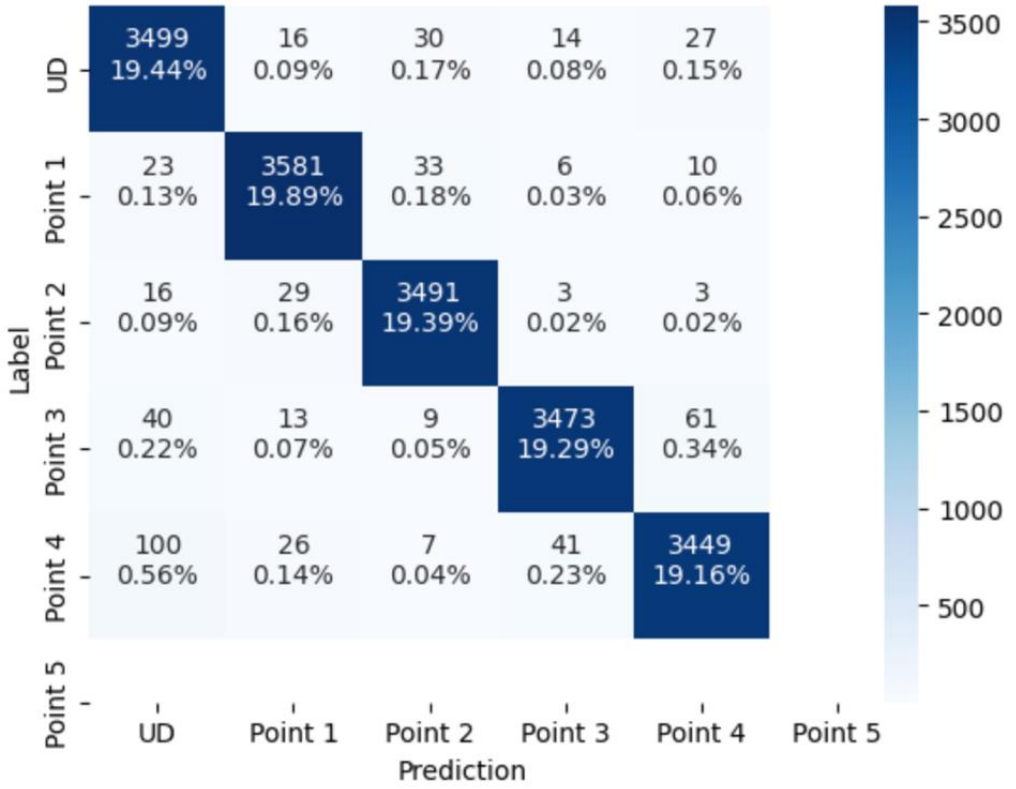


Figure 45. Confusion Matrix of 1HL DNN

4.3.2 Training on FEM data with data augmentation

The neural network designed for this task is a one hidden layer neural network. The input layer has 600 units, which is the same data length as the reformatted data. There are 300 units for the hidden layer and 5 units for the output layer, corresponding to the 6 classes for classification. There are 181,805 total units in this DNN. For the activation functions, a ReLU function is used for the in the hidden layer and a SoftMax function is used for the output layer. In the compiling process, Adam optimization is used to

doi:10.6342/NTU202401548

minimize the loss function. The full layout is shown in Figure 42. The data set consists of 54000 examples after data augmentation of frequency masking with a train-validation split of 6:4. Because data augmentation has so much power in increasing both the accuracy and variance of the model, it is now used as the default method to counter the problem of having limited data.

The neural network is trained for 28 epochs when applied early stopping with a tolerance of 5 epochs monitoring the loss of the validation set. In the first training stage, a constant learning rate of 10^{-3} is applied and the training set achieved only 97% on the training set and 95% on the validation set with a loss of only 0.09 on the training set and 0.18 on the validation set. This indicates that the network can accurately identify the data in both the training and validation and there is little to no overfitting of the dataset which also means that the generalizability towards different datasets.

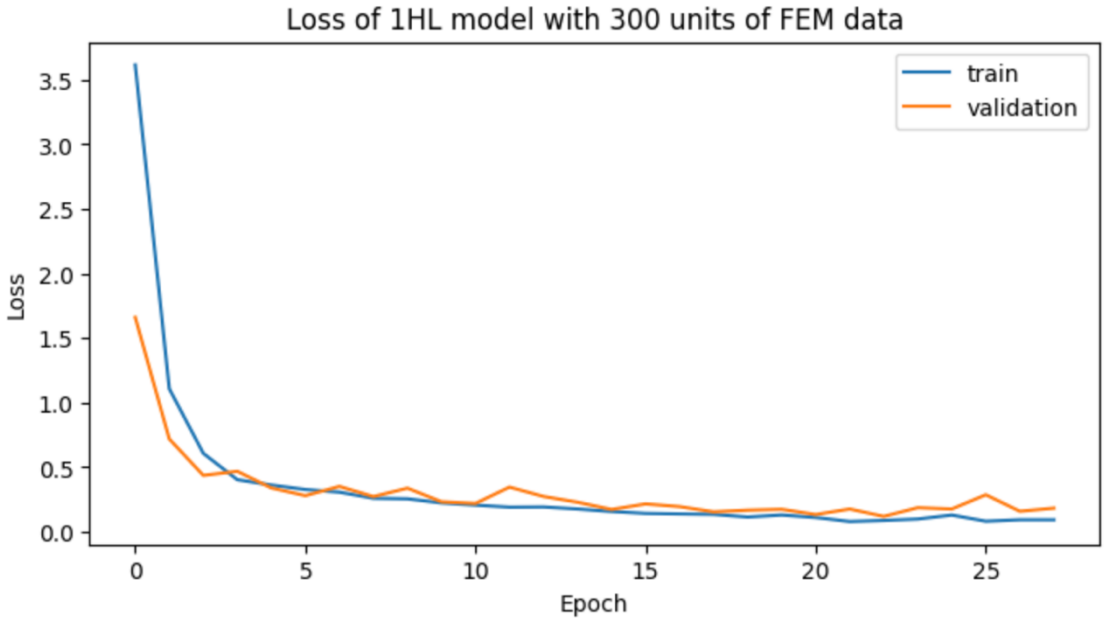


Figure 46. Loss function plot of FEM data

doi:10.6342/NTU202401548

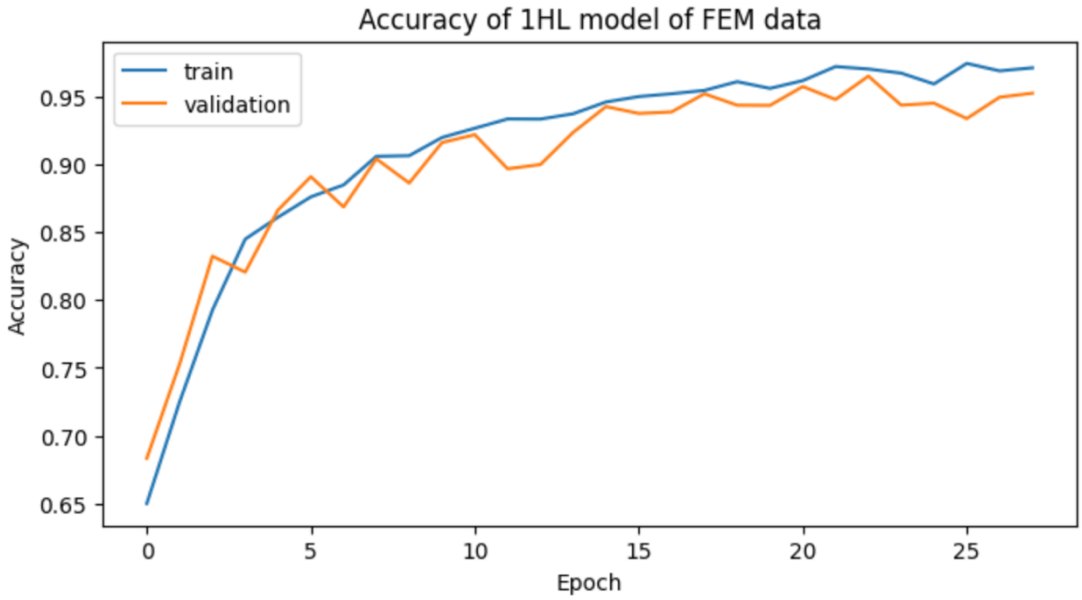


Figure 47. Accuracy of FEM dataset

In order to boost the accuracy of the model, a fine-tuning scheme is used after training the base mode. The model is then trained on a smaller learning rate of 0.0001, which is only 1/10 of the original state, could give us a boost in accuracy thus optimizing the model. In order to conduct this procedure, the previous network is then compiled with the updated learning rate to and fine-tuned over 5 epochs.

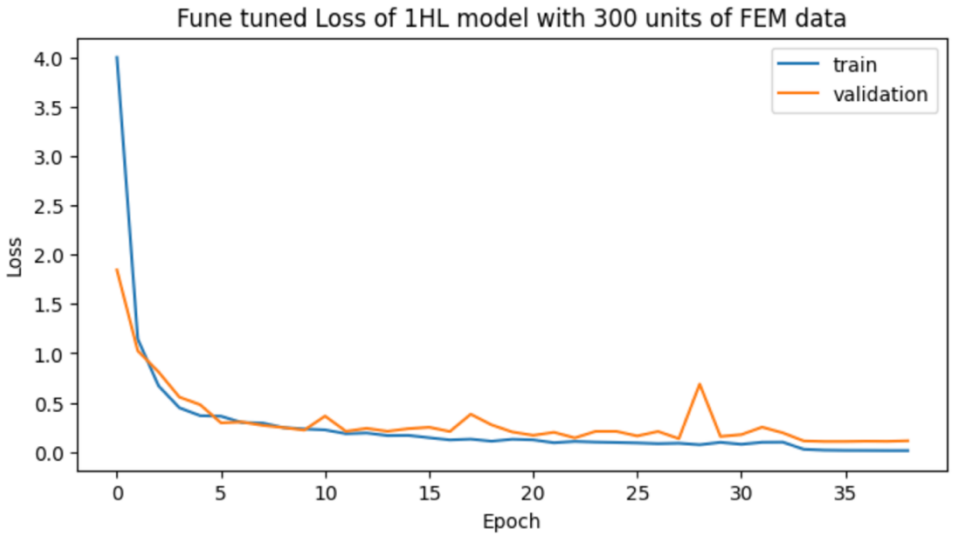


Figure 48. Fine-tuned FEM loss function

doi:10.6342/NTU202401548

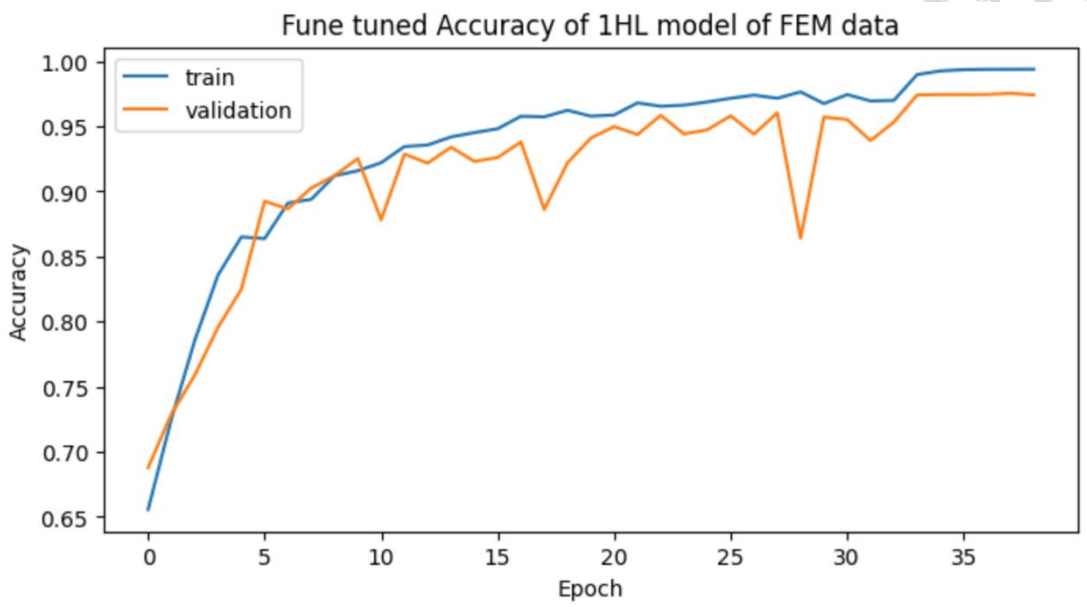


Figure 49. Fine-tuned FEM model accuracy of data augmentation

After the neural network is trained for 38 epochs when applied fine tuning of a learning rate of 10^{-4} with a limit of 5 epochs, the 1HL-NN achieved an exceptional 99.3% on the training set and 97.4% on the validation set with a loss of only 0.013 on the training set and 0.011 on the validation set. This means that the NN after fine tuning has a greater accuracy and confidence of predicting the right damage identification of multiple damage cases. By applying fine tuning just for a small number of epochs, the network can identify the data even more accurately in both the training and validation without compromising the generalizability of the network from risking overfitting.

doi:10.6342/NTU202401548

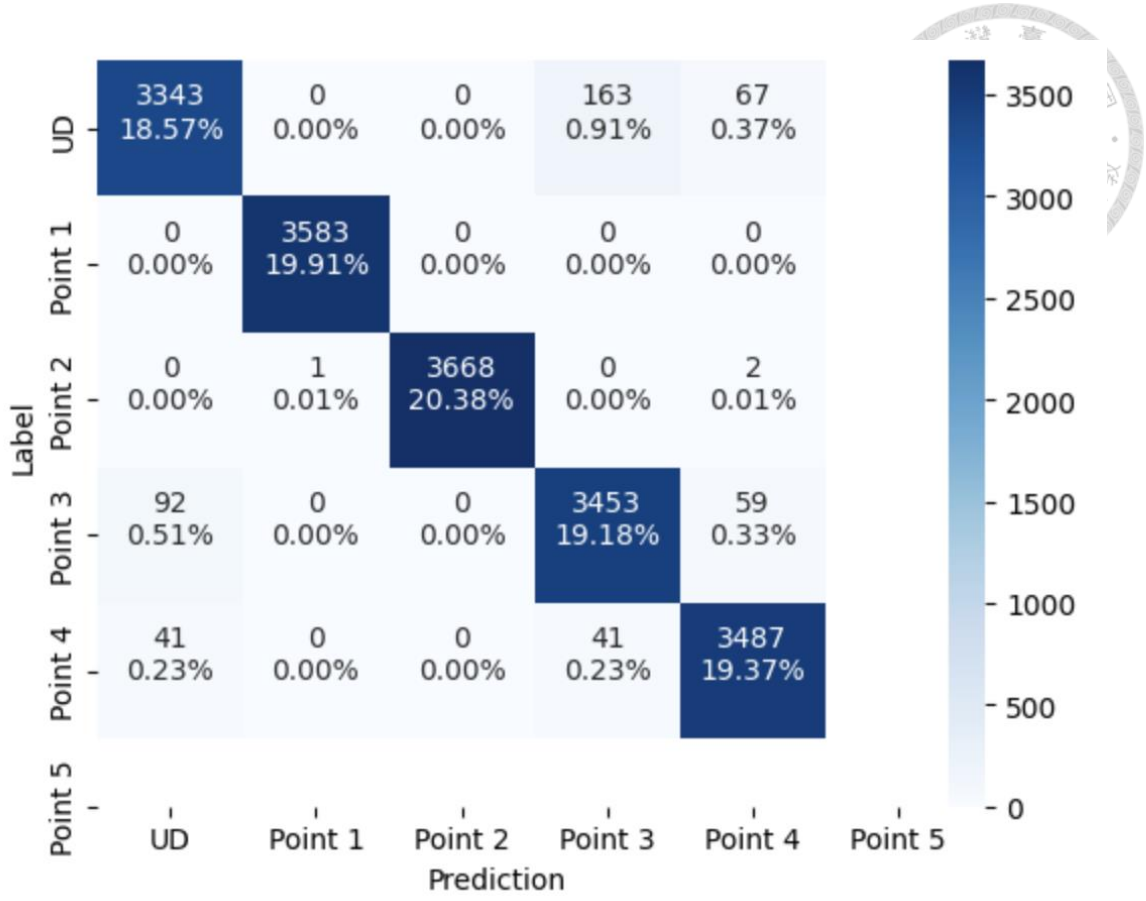


Figure 50. Confusion matrix of 1HL-NN on augmented FEM data

4.3.3 Hybrid database

In the hybrid database training, database is created by simply combining the two databases of the accelerometer and FEM results. The hybrid database is aimed to predict multiple damages from both sources and see if the two databases of the same structure could be merged and identified with precision. The accelerometer database consists of point damage of the original undamaged case UD along with the four damage points, P1 to P4 and the accelerometer database consists of beam stiffness reduction of 5 beams, B1 to B6. The databases are first gone through data augmentation process of a data size increase of 15 times and then all the different damage classes are joined together into a single damage file with 10 classes with 9000 examples in each class. The total file has 90000 examples and the train-validation split is 6:4, with 54000 examples in the training

doi:10.6342/NTU202401548

set and 36000 examples on the validation set.

The first layer is unchanged because the input method is all the same, with four channels with a frequency bandwidth of 0 to 15 Hz with a sample rate of 15 Hz. The model is updated in the final layer to 10 units to create a layer that can classify 10 different classes with a Sigmoid activation function that can predict the chances of each class. The hidden layer is consistent with the previous NNs, with 300 units in the hidden layer and the activation function is a traditional ReLU function. The total layout is shown in Figure 51.

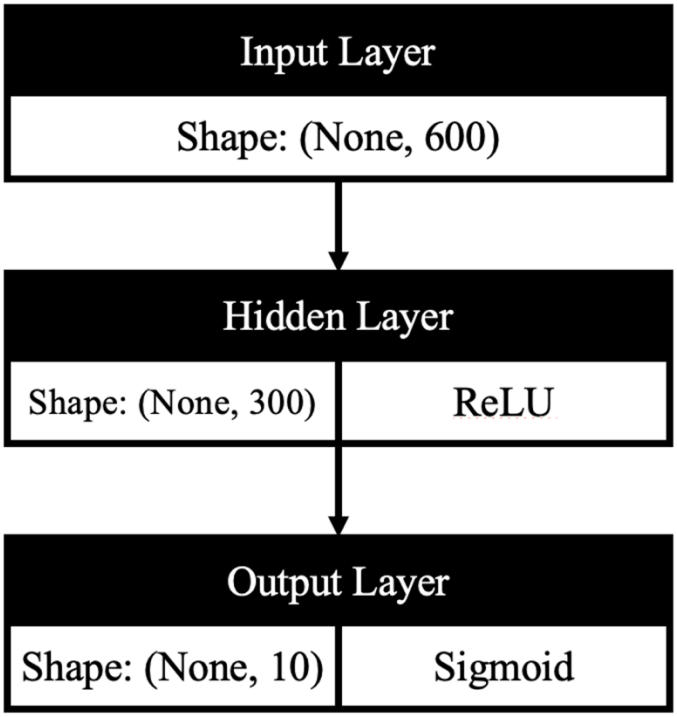


Figure 51. Hybrid database 1HL-NN Layout

doi:10.6342/NTU202401548

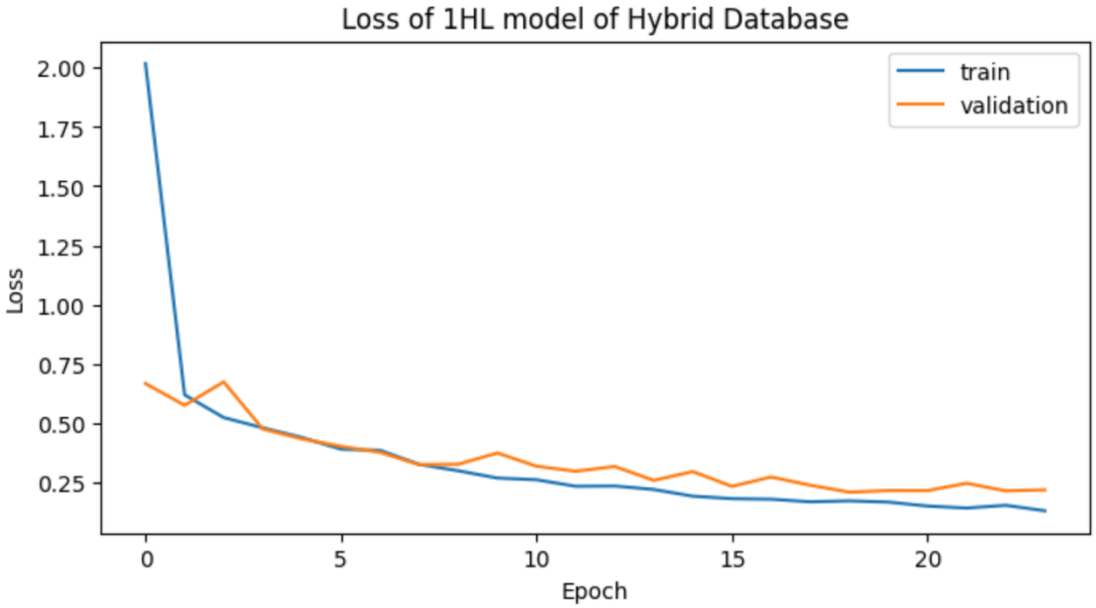


Figure 52. Loss of 1HL model of Hybrid Database

Figure 52 shows the loss of the initial training and Figure 53 shows the corresponding accuracy. The neural network finished its training at the 24th epoch when applied early stopping with a tolerance of 5 epochs, monitoring the loss of the validation set. In the first training stage, a constant learning rate of 10^{-3} is applied and the training set achieved 95.8% on the training set and 94.5% on the validation set with a loss of only 0.12 on the training set and 0.21 on the validation set. This indicates that the network can accurately identify the data in both the training and validation and there is little to no overfitting. The accuracy of the training and validation set is quite similar, which also means that the neural network has good generalizability towards two absolutely different datasets and the results are ideal.

doi:10.6342/NTU202401548

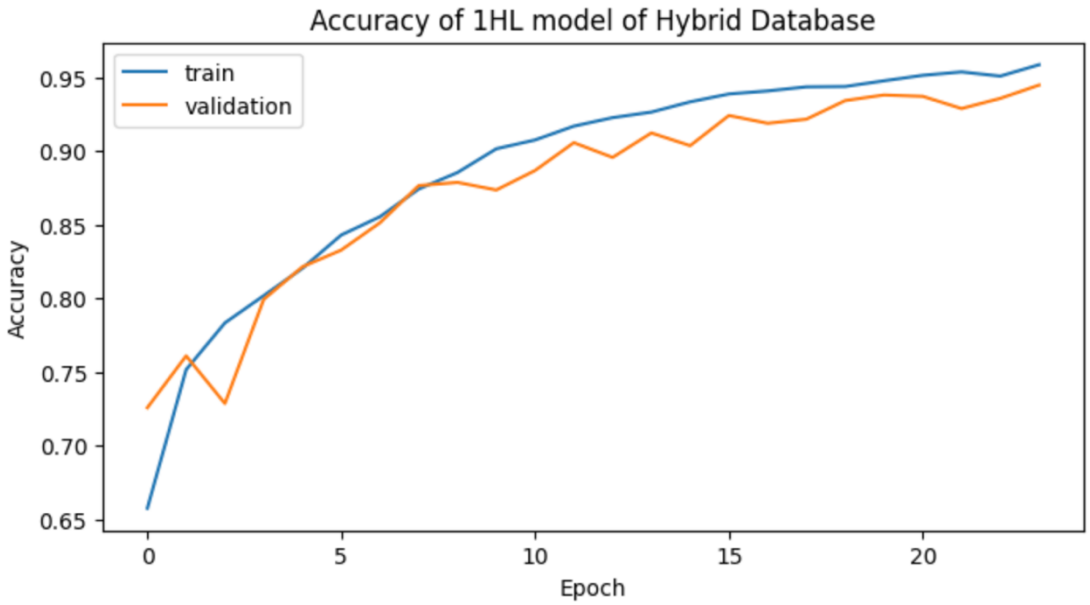


Figure 53. Accuracy of 1HL model of Hybrid Database

In order to boost the accuracy of the model, a fine-tuning scheme is used after training the base mode. The model is then trained on a smaller learning rate of 0.0001, which is only 1/10 of the original state, could give us a boost in accuracy thus optimizing the model. In order to conduct this procedure, the previous network is then compiled with the updated learning rate to and fine-tuned over 5 epochs. After the neural network stopped the training at 29th epoch when applied fine tuning of a learning rate of 10^{-4} with a limit of 5 epochs, the 1HL-NN achieved an exceptional 98.8% on the training set and 96.4% on the validation set with a loss of only 0.035 on the training set and 0.14 on the validation set (Figure 54, Figure 55). This means that the NN after fine tuning has a greater accuracy and confidence of predicting the right damage identification of multiple damage cases. By applying fine tuning just for a small number of epochs, the network can identify the data even more accurately in both the training and validation without compromising the generalizability of the network from risking overfitting.

doi:10.6342/NTU202401548

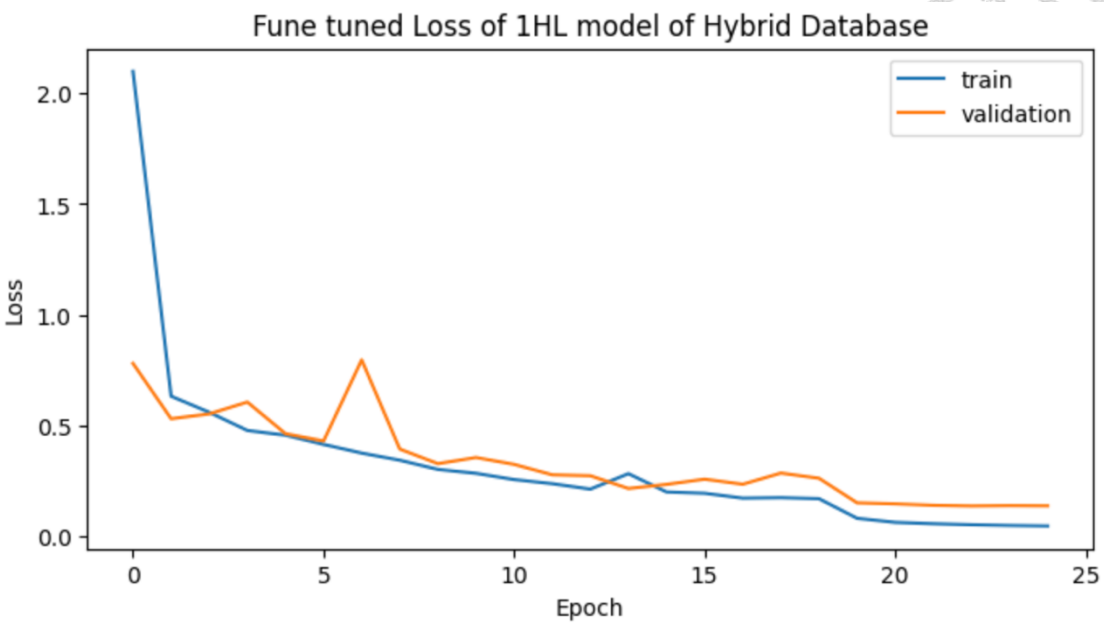


Figure 54. Loss of 1HL model of Hybrid Database after fine tuning

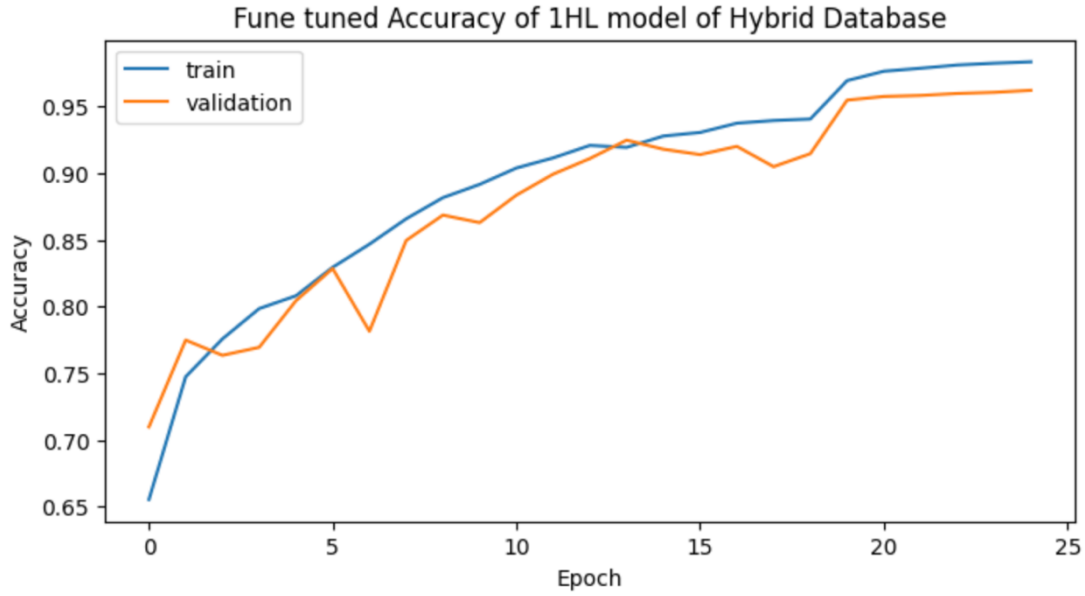


Figure 55. Accuracy of 1HL model of Hybrid Database after fine tuning

Upon further examination of the fine-tuned confusion matrix in Figure 56, we can see that there are only a few cases mis-identified across databases and the reason could be that the difference between databases identifiable enough so that different classes could be distinguished. Most classes perform well in identification between each case, with the

doi:10.6342/NTU202401548

less performant being Beam 1 to Beam 4, with over 438 misidentified cases totally and it contributed to the most confusion of the neural network. It is also worth noting that it performed slightly worse compared to a single database. The reason might come from the slight increase of network size has slightly increased the database size demand, which we technically didn't expand since there are no new data added to the database so the NN kept is trained using the same data.

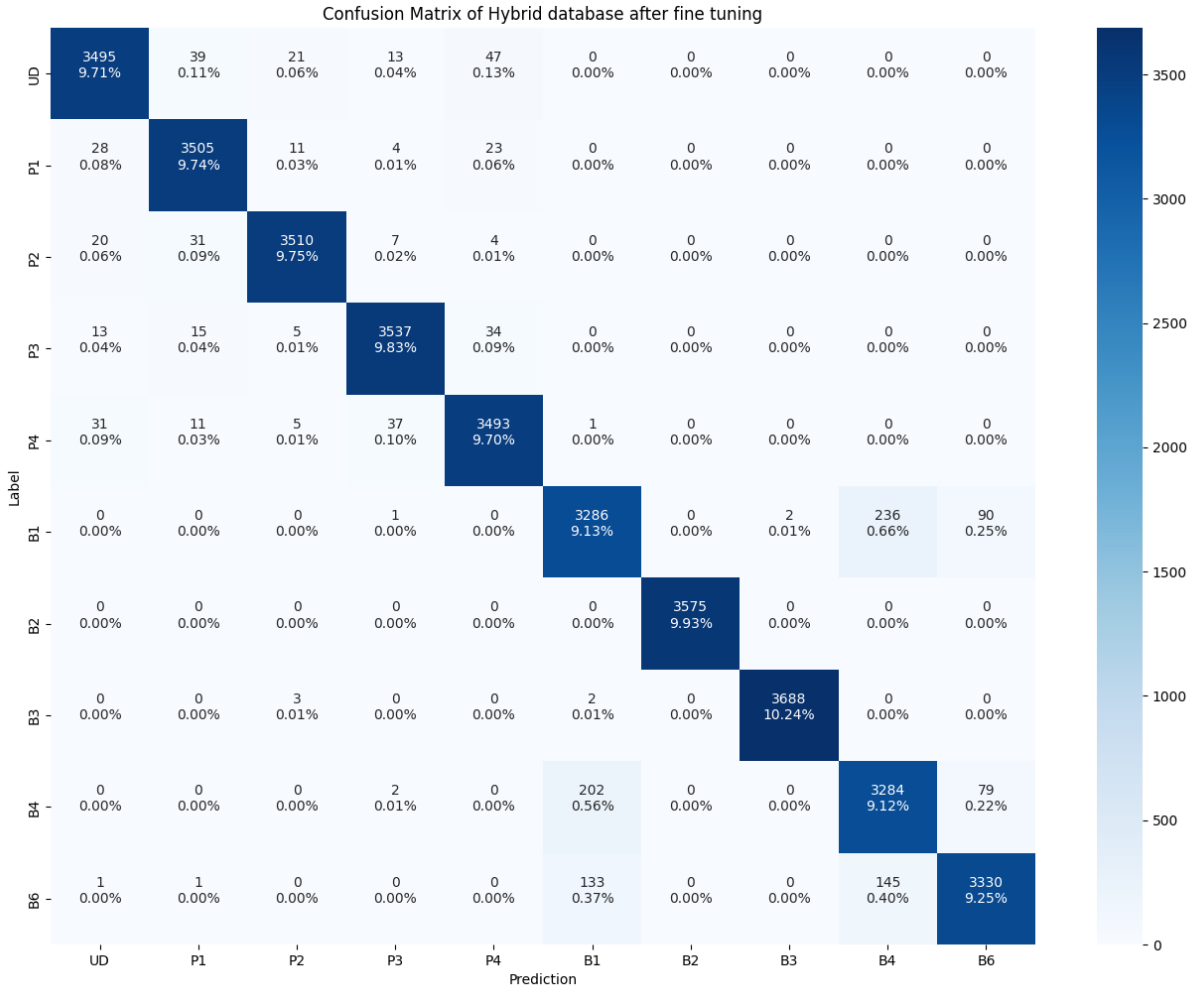


Figure 56. Confusion Matrix of 1HL model on hybrid database

doi:10.6342/NTU202401548

4.4 Zero Layer Neural Network Results

4.4.1 Neural Network Design and Training Methods

In order to verify that the proposed NN has the best results for frequency domain identification, a zero-layer NN (later referred as ZL-NN) has been designed and trained on the same parameters as the 1HL design. The ZL-NN is comprised of a layer of input units that corresponds to the input shape of the four frequency channels, which is 600 units. The output units are the same as the classes that will be classified, a total of 10 classes in the hybrid database which corresponds to the same databases as the previous training. For the training process, a base learning rate of 0.001 per step is used to achieve a good estimation of the fitting algorithm. The layout is shown in Figure 57, and the NN propagation theory will suggest that the 10 units of the output layers would decide the probability of the classes by examining all the frequency components of the four channels.

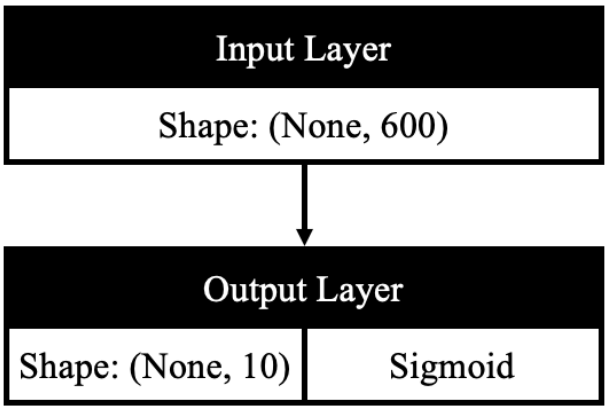


Figure 57. Layout of zero layer model

4.4.2 Zero-layer Hybrid Dataset Training Results

In the initial training phase, a total epoch of 200 is used to train the network and an early-stopping procedure of monitoring the validation loss is set to 5 epochs to prevent overfitting from occurring. In the first training phase, only 11 epoch was trained when the

doi:10.6342/NTU202401548

validation loss started to rise above 5 epochs. The loss of the training set and validation set is 1.32 and 2.65 subjectively, and the accuracy of the training set and validation set is 71% and 0.69% subjectively and the total loss and accuracy plot is shown in Figure 58 and Figure 59. It is obvious that by reducing the hidden layer, the ZL-NN is too simple to identify tasks so that it under-fits the dataset, suggesting that a network with more complexity might be required to obtain good results.

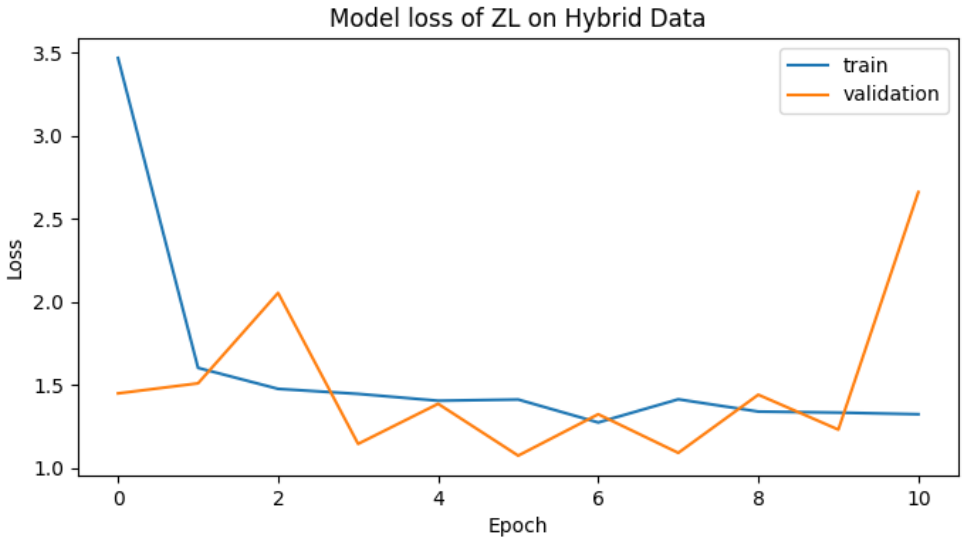


Figure 58. Zero Layer Model Loss on Hybrid Database

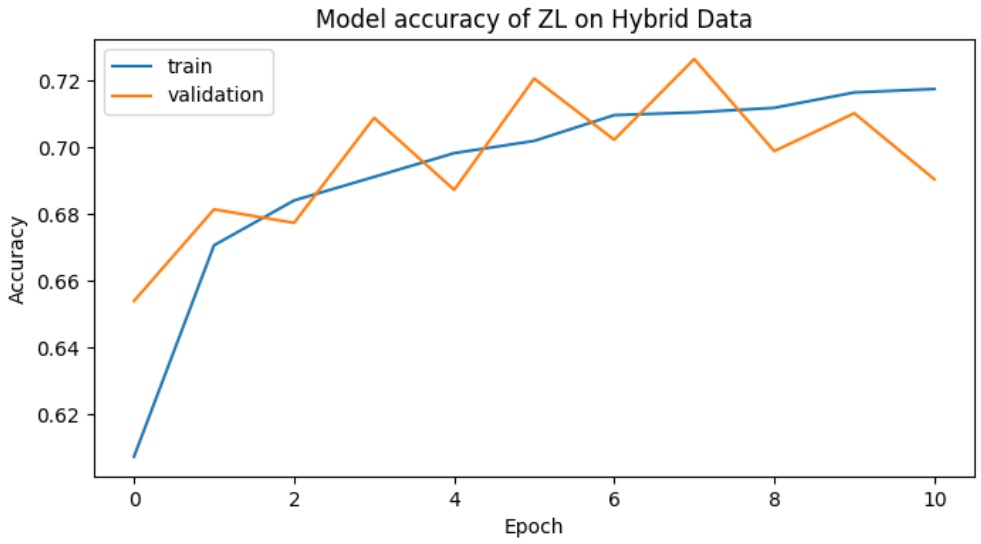


Figure 59. Zero Layer Model Accuracy on Hybrid Database

doi:10.6342/NTU202401548

We then also used the same fine-tuning procedure on the ZL-NN to train the hybrid database, hoping to obtain a better result for the network. The fine-tuning process consists of a training of 5 epochs with a smaller learning rate of 10^{-4} per step that would be able to obtain a more refined result than the larger learning rate, albeit the larger learning rate could obtain a better result faster initially. After fine tuning, the loss of the training set and validation set is 0.62 and 0.77 subjectively, the low value of loss suggests that the ZL-NN has a high confidence of identifying the results as opposed to the initial training. The accuracy of the training set and validation set is 77.4% and 74.8% subjectively, which is quite an improvement compared to the initial stage and the boost of accuracy and smoother training curve is as expected with the smaller training rate. The total loss and accuracy plot is shown in Figure 60 and Figure 61. Although good progress has been made with fine tuning technique, the network is just simply too simple to identify all the features of the 600 units of the 4 channels so that even with a fine-tuning procedure, the accuracy and loss has plateaued quite quickly. This implies that a more complex network such as the 1HL-NN model performs better for a variety of different classes and is more suitable for a multi-damage detection technique.

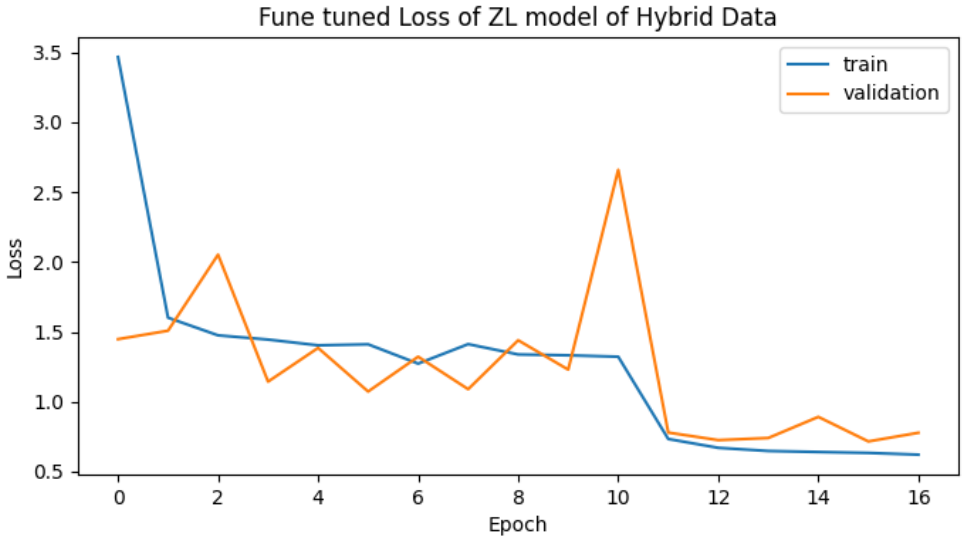


Figure 60. Fine Tuned Zero Layer Model Loss on Hybrid Database

doi:10.6342/NTU202401548

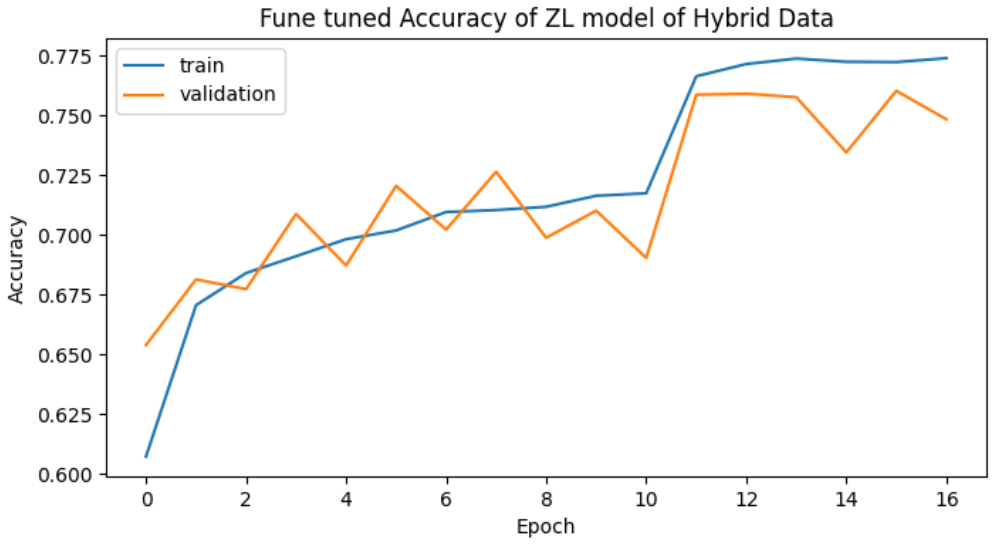


Figure 61. Fine Tuned Zero Layer Model Accuracy on Hybrid Database

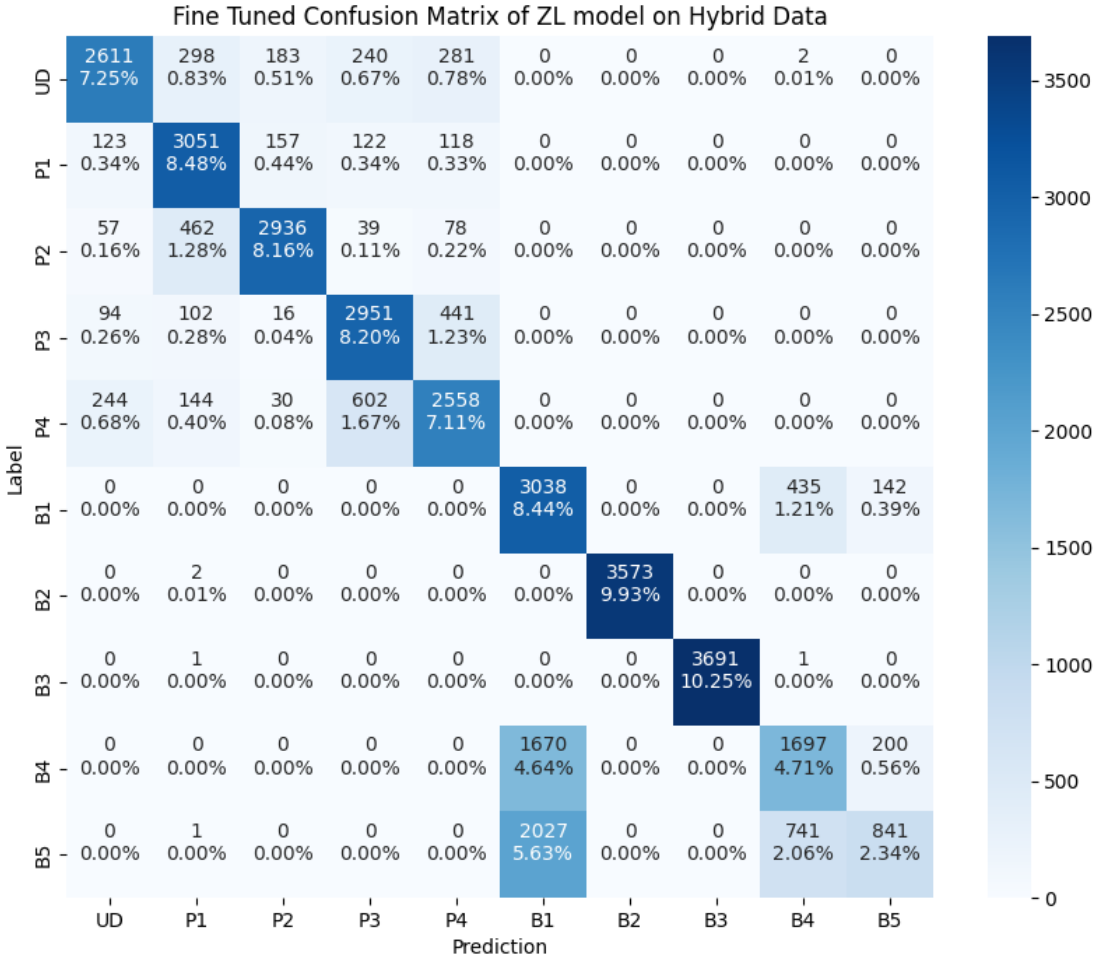


Figure 62. Fine Tuned Zero Layer Model Confusion Matrix on Validation Set in

doi:10.6342/NTU202401548

Hybrid Database

Figure 62 is the confusion matrix of the ZL-NN trained on the hybrid database after fine tuning. It is plotted by examining the validation set, which is only not used to update the parameters of the NN, so that it can obtain a neutral result. By observing the results of the training, the ZL-NN can successfully perform identification tasks on most cases, except for the Beam 1 -Beam 5. And Beam 4 – Beam 5, and Point 3 – Point 4. There are only little cases of the FEM dataset data that is confused with the Accelerometer dataset because of the Fourier coefficient distribution of the FEM data is generally more peaked than the Accelerometer data. The results suggest that the network is over simplified so that the intricacies of the changes in similar damage cases cannot be distinguished as well as we expected. Even with a fine-tuning procedure it's still not able to produce satisfactory results.

4.5 2 Hidden Layers

4.5.1 2HL-NN Design and Layout

In the previous section we have found out that a zero-layer NN is too simple to identify accurately the changes in frequency of the SHM system and just by adding 1 layer could have significantly better results. Therefore, in order to investigate if a deeper NN could potentially have a better result for frequency domain identification, a 2 hidden-layer NN (later referred as 2HL-NN) has been designed and trained on the same parameters as the 1HL NN.

The 2L-NN is comprised of a layer of input units that corresponds to the input shape of the four frequency channels, which is 600 units. The output units are the same as the classes that will be classified, a total of 10 classes in the hybrid database which

doi:10.6342/NTU202401548

corresponds to the same databases as the previous training. The two hidden layers both have 300 units, which is the same as hidden layer units as the 1HL model. It is worth noting that this is a initial guess for the NN and is not optimized for the best results. For the training process, a base learning rate of 0.001 per step is used to achieve a good estimation of the fitting algorithm. The layout is shown in Figure 63, and the NN propagation theory will suggest that the hidden units of the output layers would decide the probability of the classes by examining all the frequency components of the four channels.

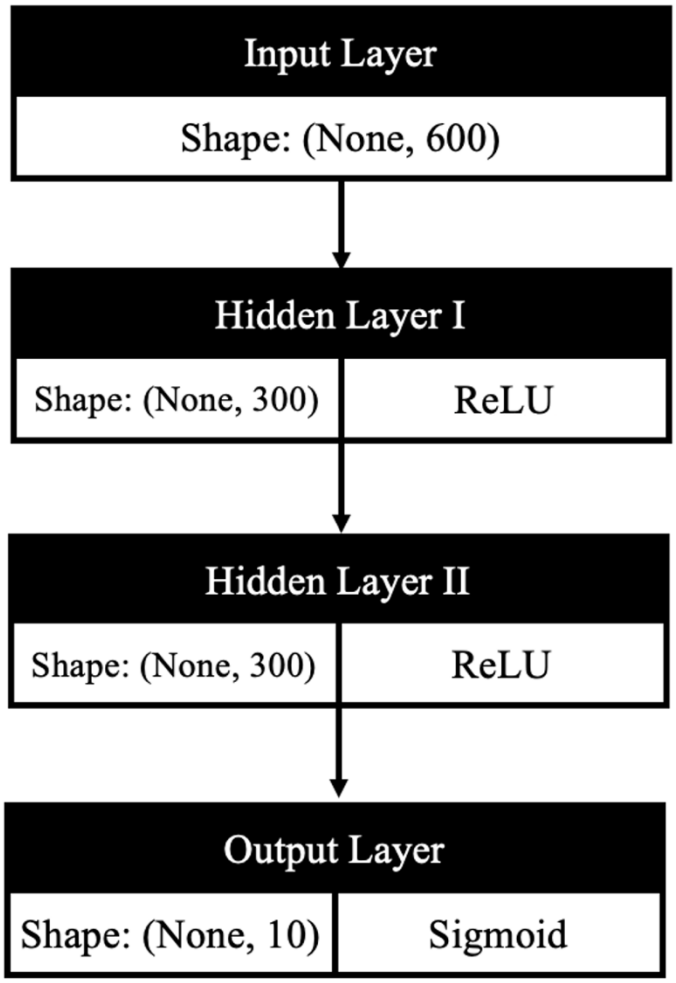


Figure 63. 2HL NN design layout

doi:10.6342/NTU202401548

4.5.2 2HL-NN Training Results on Hybrid Database

In the initial training phase, a total epoch of 200 is used to train the network and an early-stopping procedure of monitoring the validation loss is set to 5 epochs to prevent overfitting from occurring. In the first training phase, a total of 23 epochs was trained when the validation loss started to rise above 5 epochs and thus ends the training procedure. The loss of the training set and validation set is 0.1375 and 0.2462 subjectively, and the accuracy of the training set and validation set is 94% and 92% subjectively and the total loss and accuracy plot is shown in Figure 64 and Figure 65. The results did not show a significant increase in accuracy compared to the 1HL model, yet it has shown a slight decrease compared to the 1HL model. This suggests that one hidden layer might be enough complexity to achieve frequency identification.

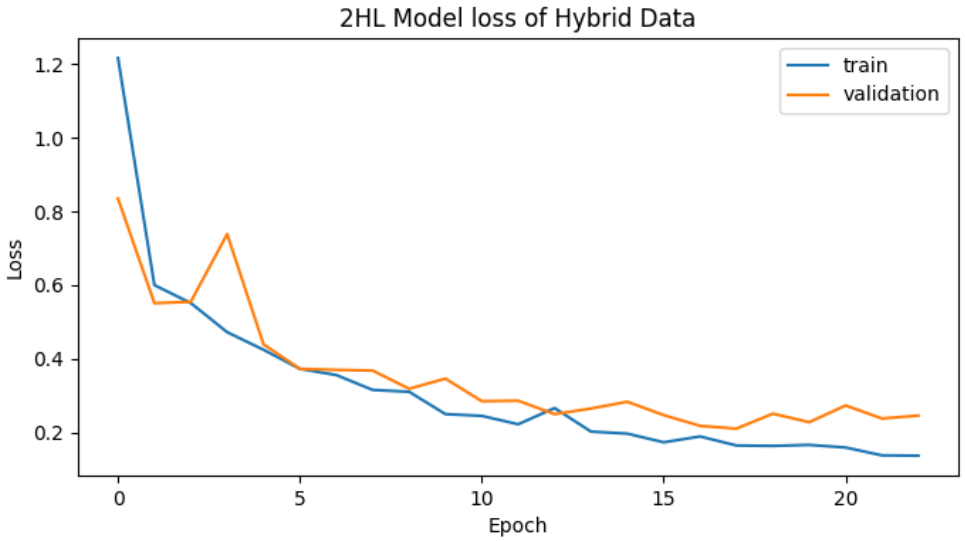


Figure 64. 2HL Model loss of Hybrid Database

doi:10.6342/NTU202401548

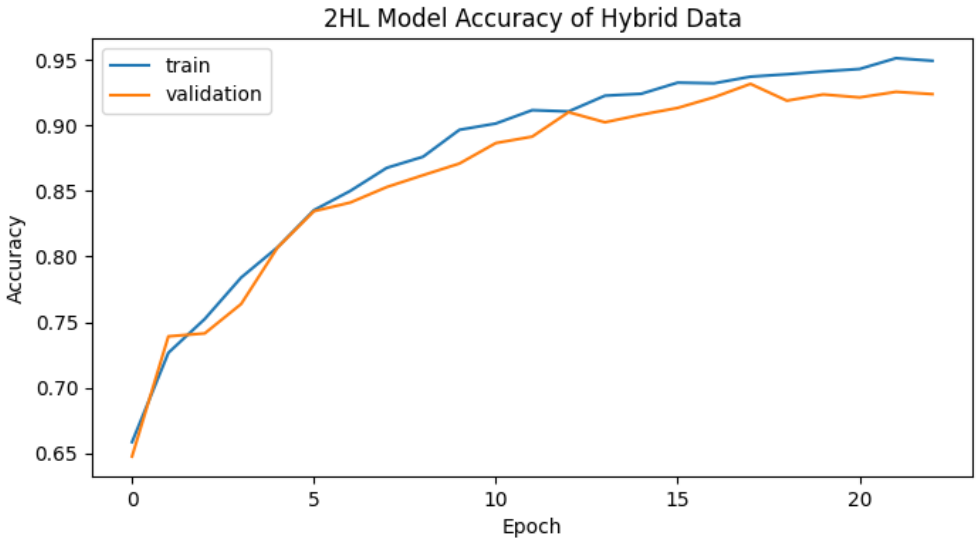


Figure 65. 2HL Model Accuracy of Hybrid Database

We then also used the same fine-tuning procedure on the 2HL-NN to train the hybrid database, hoping to obtain a better result for the network. The fine-tuning process consists of a training of 5 epochs with a smaller learning rate of 10^{-4} per step that would be able to obtain a more refined result than the larger learning rate, albeit the larger learning rate could obtain a better result faster initially.

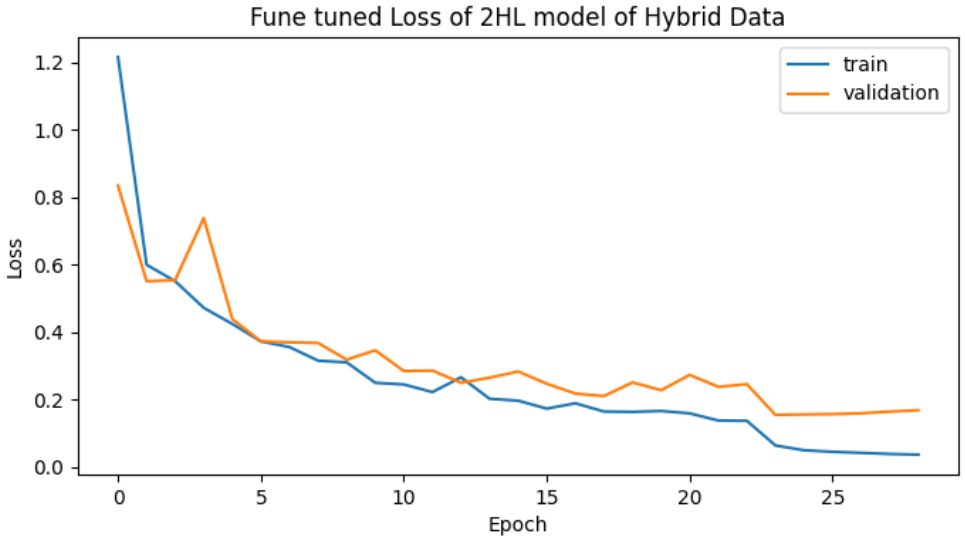


Figure 66. Fine Tuned 2HL Model loss of Hybrid Database

doi:10.6342/NTU202401548

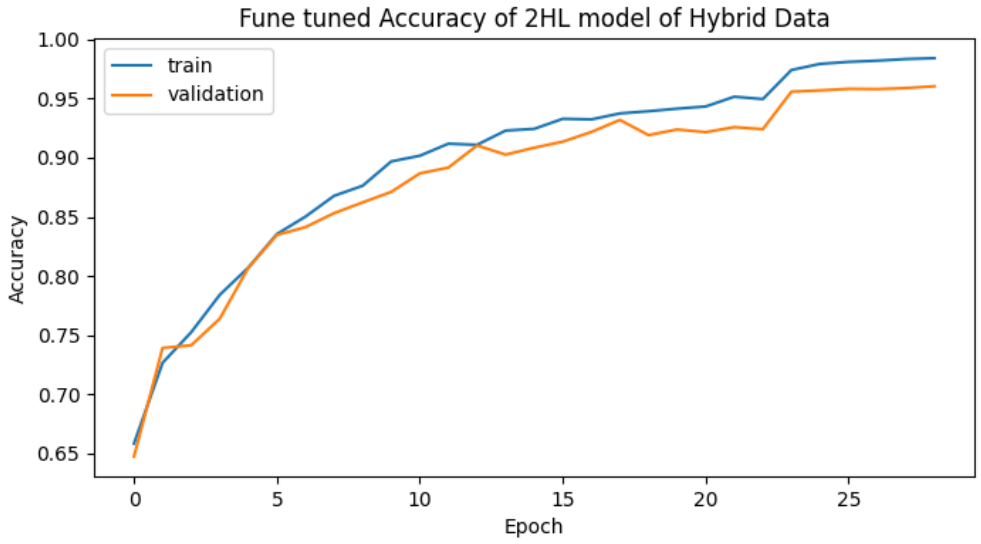


Figure 67. Fine Tuned 2HL Model loss of Hybrid Database

After fine tuning, the loss of the training set and validation set is 0.037 and 0.168 subjectively, the low value of loss suggests that the 2HL-NN has a high confidence of identifying the results as opposed to the initial training. The accuracy of the training set and validation set is 98.4% and 96.0% subjectively, which is a slight improvement compared to the initial stage and the boost of accuracy and smoother training curve is as expected with the smaller training rate. The total loss and accuracy plot is shown in Figure 66 and Figure 67. The network has shown a healthy training result since the accuracy is high and the variance is only 2%, which implies good fitting of the dataset. Although good progress has been made with fine tuning technique, the network still did not beat the accuracy of the 1HL-NN. This implies that even with a more complex network with more trainable parameters of the 2HL-NN model could not boost the accuracy of the network since the dataset does not require a complex model for identification. A simple 1HL-Model with more optimization what we need for this implementation, a more complex layers that requires more data to train would guarantee good results and might risk overfitting the dataset, which leads to bad generalizability.

doi:10.6342/NTU202401548

Upon further examination of the fine-tuned confusion matrix in Figure 68, we can see that there are only a few cases mis-identified across databases and the reason could be that the difference between databases identifiable enough so that different classes could be distinguished. Most classes perform well in identification between each case, with the less performant being Beam 1 - Beam 4 and Beam 1 – Beam 5, with over 400 misidentified cases totally and it contributed to the most confusion of the neural network. This might imply that when the data is not as distinguishable, even with increased complexity the network could not fit the data as satisfactory and there will be mislabeled cases.

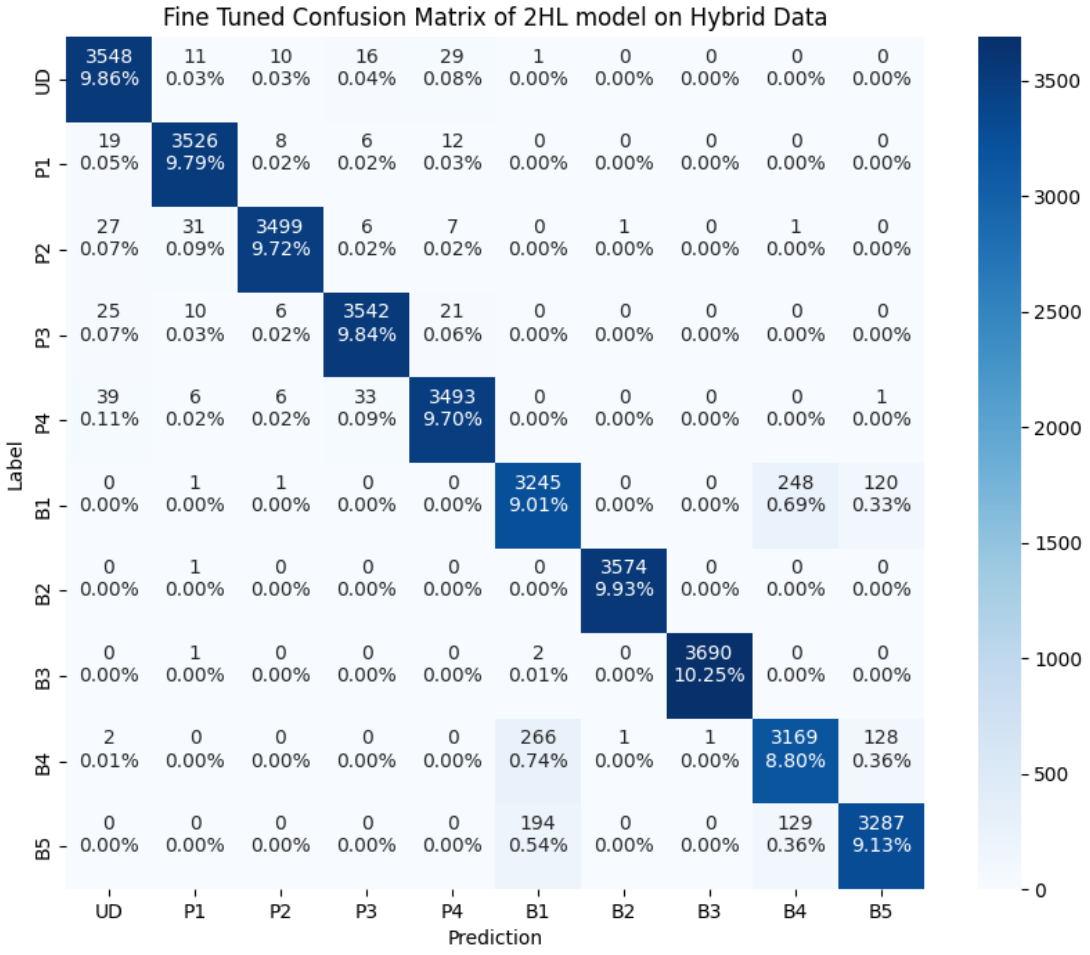


Figure 68. Fine Tuned Confusion Matrix of 2HL model on Hybrid Database

doi:10.6342/NTU202401548

4.6 Motivation for New Design

In order to improve the generalizability of the DNN for a good transferability towards real life structures and more complex damage scenarios, this section now leverages the immense identifiable power of DNNs to create a better network so that it could be used more generally and more efficiently.

In the previous four channel setup, it is mandatory that the user specify the damage locations or scenarios in the output (SoftMax) layer so that the DNN could determine the probability of each damage scenario and classify the training data into the classes. This was the default setup for supervised learning, which works well when the damage cases are known. The downside of this approach is that NNs are generally bad at extrapolation and it is only as good as the dataset that it is trained on, therefore if an unknown case is shown to the NN, such as a combination of multiple damage locations, the NN only have the ability to classify as one of the previous classes that it was trained on. This would mean that the NN would classify the multi-damage scenario by most prominent features, which in the previous case would only classify as one certain location. Researchers could certainly generate multi-damage data by (1.) directly conducting multi-damage experiments and generate data or (2.) synthesize multi-damage combinations. The first approach could potentially better than the second approach in a data purity and variability regard but creating new samples of different damage combinations are very cost-inefficient especially when the structure so complex that the combinations of damage cases will multiply exponentially. Thus, the first method would render impractical in real-life cases, where the structures are much larger and much more complex. The second method would be more economically efficient, yet the damage of a certain location might also affect the neighboring data-acquisition device. A good example is that the mode shape of a long beam, such as those of a wind turbine tower, is continuous and curvy.

doi:10.6342/NTU202401548

Thus, what we originally thought to be local damage could turn out to affect a much wider range, and when we mix the damage cases it may not be as ideal. What makes both methods perform are both undesirable is that by creating such a large amount of classes of damage combinations, it is inevitable that we have to increase the complexity of our NN, at least in the final layer. A larger NN would require much more data to train, thus the cost of acquiring data will rise for both approaches. It is even harder in real-life scenarios since practitioners most likely don't have the luxury to conduct such a thorough analysis on so many different components and structures, especially damaged ones.

Another more interesting problem to tackle is the transference between the simulated data and the data in real life. In real life scenarios, the data would often have outer noise and due to the imperfections of the material the spectrum will differ from the simulated ones quite a bit, even if the modal parameters are kept very similarly. It is also impossible to keep all of the modal parameters the same between the simulated data and the real-life data because of many inherent differences of the materials, junctions, and manufacturing errors, not to mention the boundary condition also plays a large role of the difference between the simulated data with the real-life data. The phenomenon, named data-mismatch is one of the major challenges to overcome for researchers trying to bridge the gap between simulation and real-world.

Finite Element Method-based methods can utilize looped simulations to constantly update of the digital model to make it fit the real-world model more, but most cases are dealt by using more ideal and less-complex materials and designs so there's still an opening for more advancements. In deep learning circles, there are also many implementations that would face data mismatch and it is a common problem for researchers to overcome. Deep neural networks are a notoriously data-hungry method that require a huge pool of data to train on, sometimes involving millions of data just for

doi:10.6342/NTU202401548

training. To find the resources of this huge pool of data, researchers often utilize web-crawlers to gather huge amounts of data on the internet automatically. Usually, the gathered data are often high-resolution pictures from professional websites and they will appear quite differently from the pictures that are uploaded from common users with consumer grade cameras. In applications of deep learning, it would often require the user to input his/her pictures to the pre-trained neural network so that the desired identification process can occur automatically, but the difference in resolution and even the sheer quality of the different data-pools will result in poor recognition. Based on the previous section (4.1 ~ 4.3 Hybrid database training), we can see from the confusion matrix in Figure 56. Confusion Matrix of 1HL model on hybrid database it is clear that the FEM generated database very distinguishable from the accelerometer database, with only 1~2 misidentified cases among thousands of validation examples. This would indicate that the data mismatch between the FEM generated database and the accelerometer database is very severe, signifying that data mismatch would be a great hurdle to overcome. In order for our model to achieve the desired result of transferring knowledge from the digital or simulated database to the real-life scenarios, we need to overcome the giant problem: data mismatch.

The research presented in this section aims to bring two major breakthroughs in the field of structural health monitoring and applied deep learning. The first breakthrough is to arrange the neural network differently so that it could be capable to identify different combinations of damage without knowing the damage places a-priori. The second breakthrough is to successfully transfer the knowledge from FEM based data towards real-life scenarios, with the limitations being the limited amount of real-life data. These two breakthroughs will greatly enhance the generalizability of the neural network for real-world practitioners. A new neural network structure is proposed in this section along with

doi:10.6342/NTU202401548

three different training methods to overcome this issue. The first is to leverage the transferability of DNNs by using a technique that is called transfer learning. The second and third methods both utilize the power of the hybrid database to create a training environment that could achieve our goals.

4.7 Research Methods

4.7.1 Domain Adaption and Convolutional Expansion

In the practice of deep learning, a common problem is to work with a new novel batch of data and there will be some challenges that practitioners need to face when dealing with this situation. For example, in the case of picture identification, the team used to train the NN would get most of its data source from searching the internet, which would likely have the network train on images that are high quality and resolution. But for the users, there might be a lot of low-quality images that the users would upload to the platform. The increased blur and low resolution would be hard to identify for the neural network, thus causing misidentification.

In practical application of deep learning, we have to assume that the test set and development set have almost the same distribution as what we would expect to get in real life. Our goal would be to optimize the training on the development set and use the test set as an unbiased source to see how the neural network performs in real life. Ideally, we can make the training set have the same distribution the same as the test and development set, but because of limited resources of the new variation of the data, we will have problems of distributing the new variation of data. For example, assume we only have 1,800 cases of data the new variation, and we use a traditional split of 70%-15%-15% on the train-dev-test split and we plan to use at least 10000 cases to train our neural network, this would mean that we would need 7000 cases for training alone along with 1500 sets

doi:10.6342/NTU202401548

for development and testing. If we assume that the new variation of data will make up 30% of the total training data in our real-life scenario, to make sure that they all come from the same distribution, we would need 2,100 examples of the new variation for training, and 900 examples for the test and development set in total. This means that we (a.) need to acquire more data from the new variation, or (b.) would need to change the distribution of the training set.

In order to make sure that our development and test set have the same distribution as the real-world, we would need to pull 900 examples out from the training and then we are left with 900 examples for training. Now the training set would have 12% of the new variation in our training data, which is a much different distribution compared to 30%, which is the test and development sets are comprised of. This will result in a data mismatch problem, where the neural network will potentially overfit the training set and generalize bad to the development set. There are various research papers that deal with this particular problem and such related methods has been named “Domain Adaptation Techniques, yet there still isn’t a systematic methodology of dealing with this problem as it is a new challenge that rises with the presence of big data. This circumstance is less than ideal because we have bad performance on the task that we actually want to perform well on. There are methods to work around this problem and the best one is to get more data from the new variation and feed it into our training set, while keeping the development set the same size. The development set just needs to be big enough so that we can realize the small changes that our implementation has improved on. A second method is to use some mixing methods to somehow mix the features of the new variation to old data of the training set. However, there are potentially some risks such as by creating a brand-new set of data that which the neural network will adapt to, will result in poor generalization towards the real-life data. The problem is shown in Figure 69, when

doi:10.6342/NTU202401548

we look at the features of a certain data in the 2D plane by extracting feature 1 and feature 2. If we want to fit the solution plane well, especially with limited Real-Damaged data for training, we could possibly mix the Real-Damaged data with the FEM-Damaged data to let the neural network learn the features within. Yet, because there is no distribution of the mixed features data in real life, there is no guarantee that this method can fit well towards the Real-Damaged data.

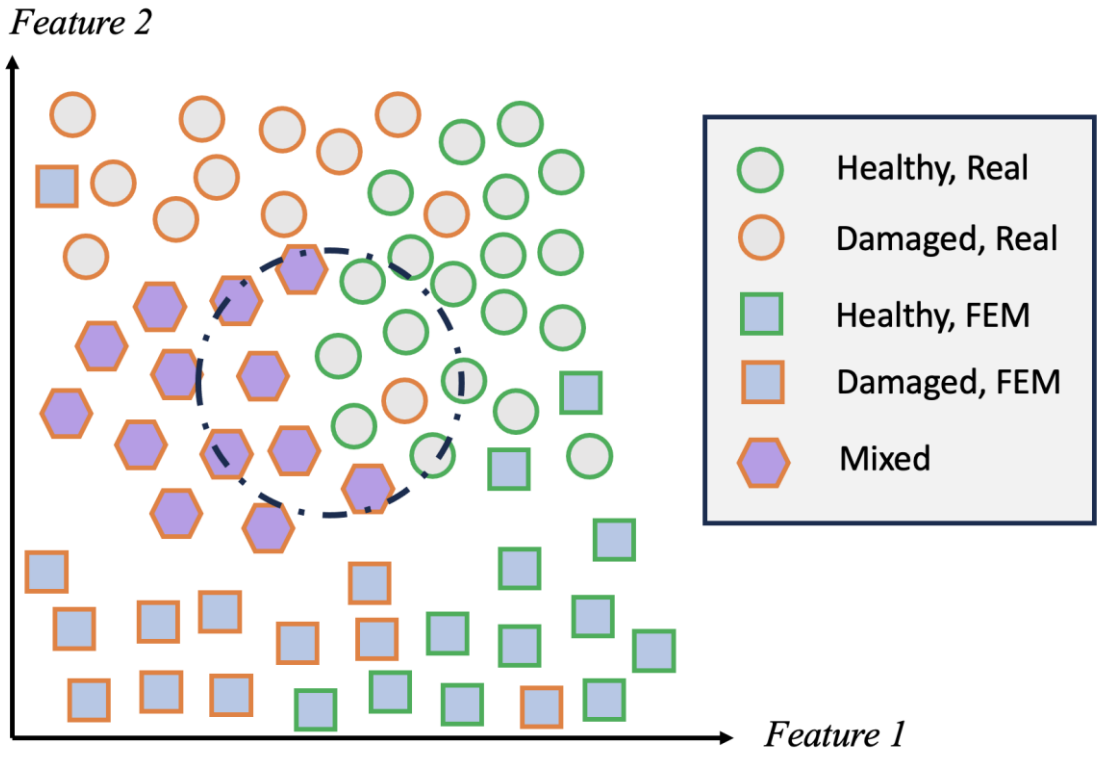


Figure 69. Problem of creating a mixed feature database

Our method to work around this is to convolve everything around the same object. The convolutional filter is like a filter and the object after convolution will exhibit the properties of both of the filter and the object before convolution. In fields of computer science, the convolutional technique is used to extract features such as vertical, horizontal, and diagonal lines for the computer to later process on. In this implementation, we have used the healthy spectrum from real-life data as a filter to convolve with healthy spectrum of simulated data, healthy spectrum from real-life data and damaged spectrum from real-

doi:10.6342/NTU202401548

life data and it is equivalent to operate on a brand-new domain, which is expanded from the real-life data. And by analyzing the data through the convolved or mixed plane (with the same type of data), we do not need to second guess if there is a new type data that the neural network will optimize. We can just conduct neural network fitting on the convolutional plane on all of the data that we get, as shown in Figure 34.

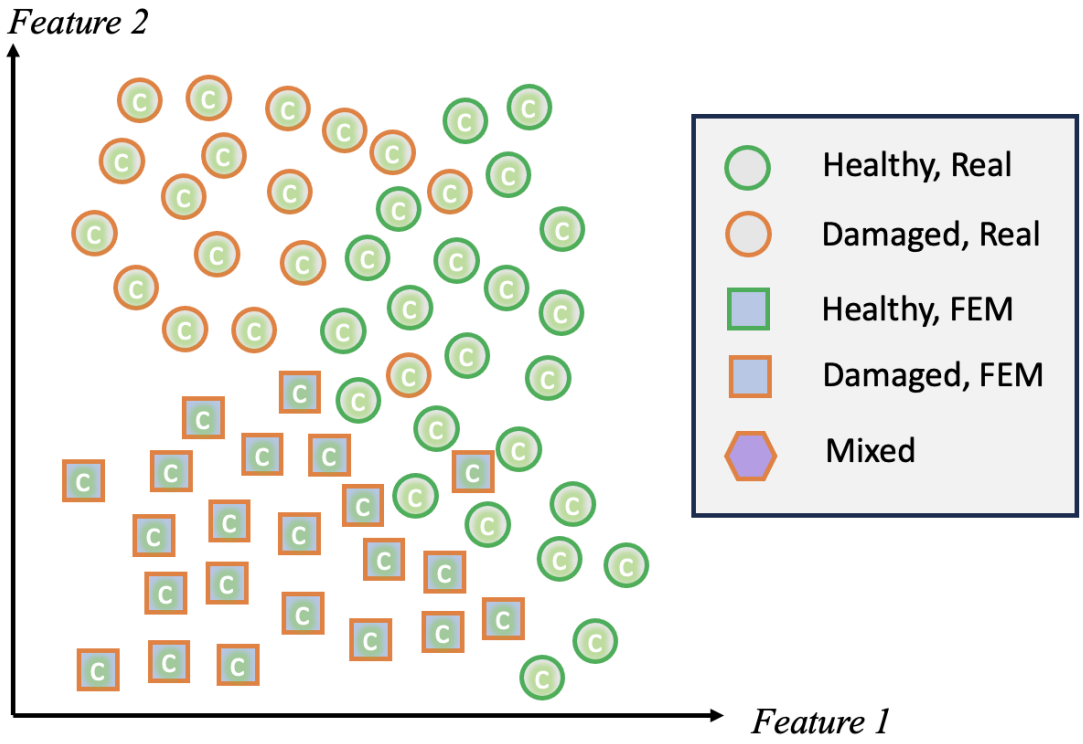


Figure 70. The effects of convolutional effect with the healthy, real-life signal

In order to verify the efficacy of this technique, cosine similarity is used to measure the likeliness of the FEM generated signal and real-life signal. The specifics of cosine similarity is in section 5.2.4. The FEM generated signal of a damaged structure is convolved with a real-life signal from a healthy structure, and a real-life signal of a damaged structure is also convolved with a real-life signal from a healthy structure. The result is shown in Figure 74, by doing the convolutional operation, the similarity of the FEM generated data and the real-life accelerometer data have increased to 0.7, which is far more alike than pre-convolution (0.4). This means that by adding a convolutional step,

doi:10.6342/NTU202401548

the simulated data is now much more alike than the real-life data, and this makes our analysis more complete. Because of the convolutional operation, the data size is now increased from 150 x 1 to 299 x 1 with no truncation being used, thus a wider network in the input layer needs to be used to perform this operation. And because of the increased complexity of the new solution plane, the network would likely need to be expanded and more data needs to be used in order to prevent the neural network from overfitting.

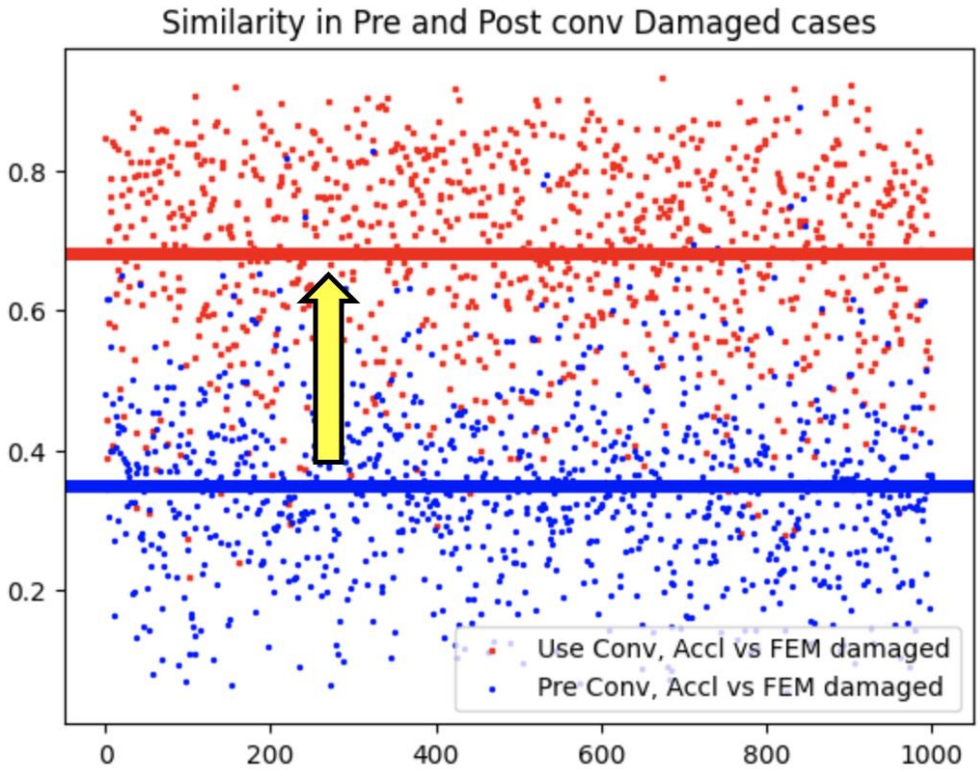


Figure 71. Cosine similarity comparison, pre and post convolution

The problem of just mixing the features and making it into a training set is that it creates a brand-new set of data, on which the neural network will train on. In order for the neural network to fit the database of the mixed features, it classifies the data that looks alike to the mixed features into their class and excludes the other sets of data into the other classes (Figure 72). Therefore, the generalization of the real-life data suffers because we have created a new form of data mismatch, the mixed features and the real-life

doi:10.6342/NTU202401548

database.

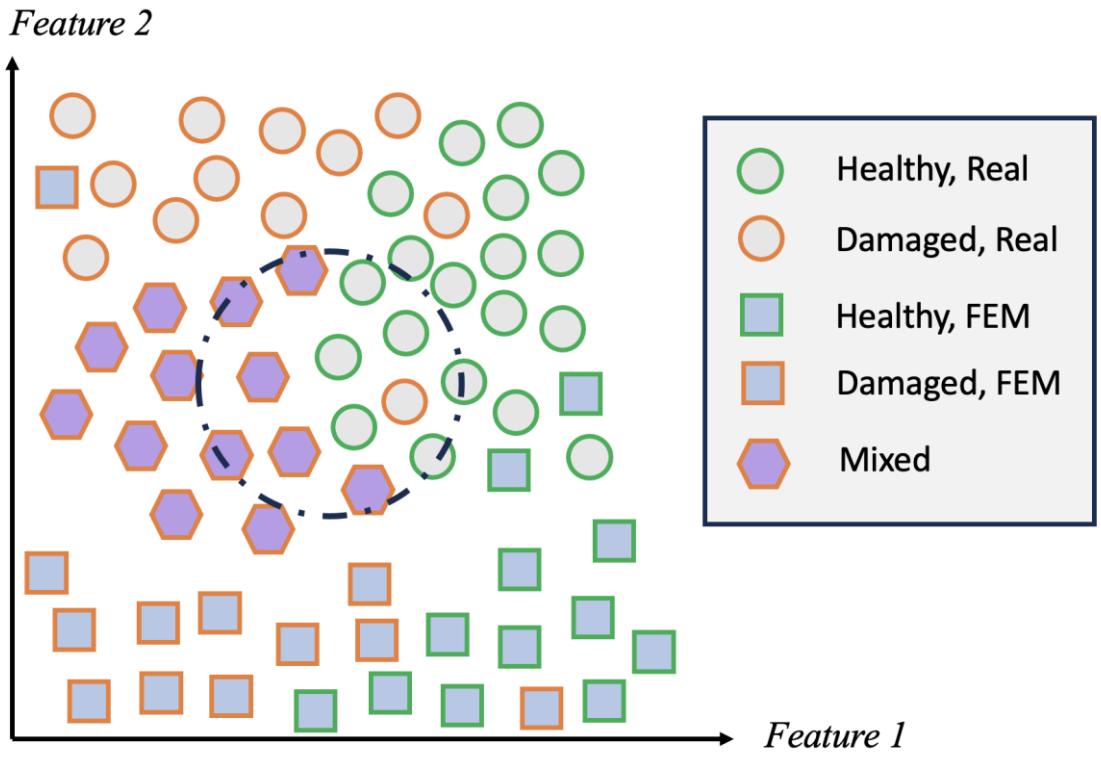


Figure 72. Problem of creating a mixed feature database

One method to work around this is to convolve everything around the same object (Figure 72). The convolutional filter is like a filter and the object after convolution will exhibit the properties of both of the filter and the object before convolution. In fields of computer science, the convolutional technique is used to extract features such as vertical, horizontal, and diagonal lines for the computer to later process on. In this implementation, we have used the healthy spectrum from real-life data as a filter to convolve with healthy spectrum of simulated data, healthy spectrum from real-life data and damaged spectrum from real-life data and it is equivalent to operate on a brand-new domain, which is expanded from the real-life data.

doi:10.6342/NTU202401548

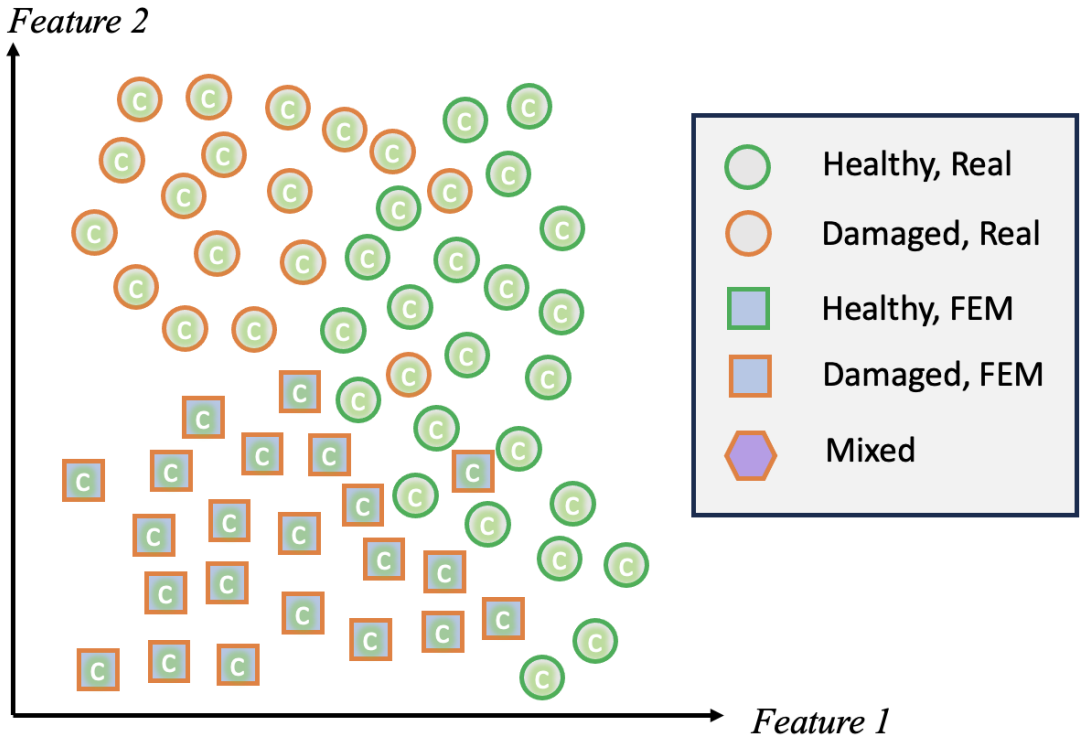


Figure 73. The effects of convolutional effect with the healthy, real-life sign

In order to verify the efficacy of this technique, cosine similarity is used to measure the likeliness of the FEM generated signal and real-life signal. The FEM generated signal of a damaged structure is convolved with a real-life signal from a healthy structure, and a real-life signal of a damaged structure is also convolved with a real-life signal from a healthy structure. The result is shown in Figure 74, by doing the convolutional operation, the similarity of the FEM generated data and the real-life accelerometer data have increased to 0.7, which is far more alike than pre-convolution (0.4). This means that by adding an convolutional step, the simulated data is now much more alike than the real-life data, and this makes our analysis more complete. Because of the convolutional operation, the data size is now increased from 150 x 1 to 299 x 1 with no truncation being used, thus a wider network in the input layer needs to be used to perform this operation. And because of the increased complexity of the new solution plane, the network would likely need to be expanded and more data needs to be used in order to prevent the neural

doi:10.6342/NTU202401548

network from overfitting.

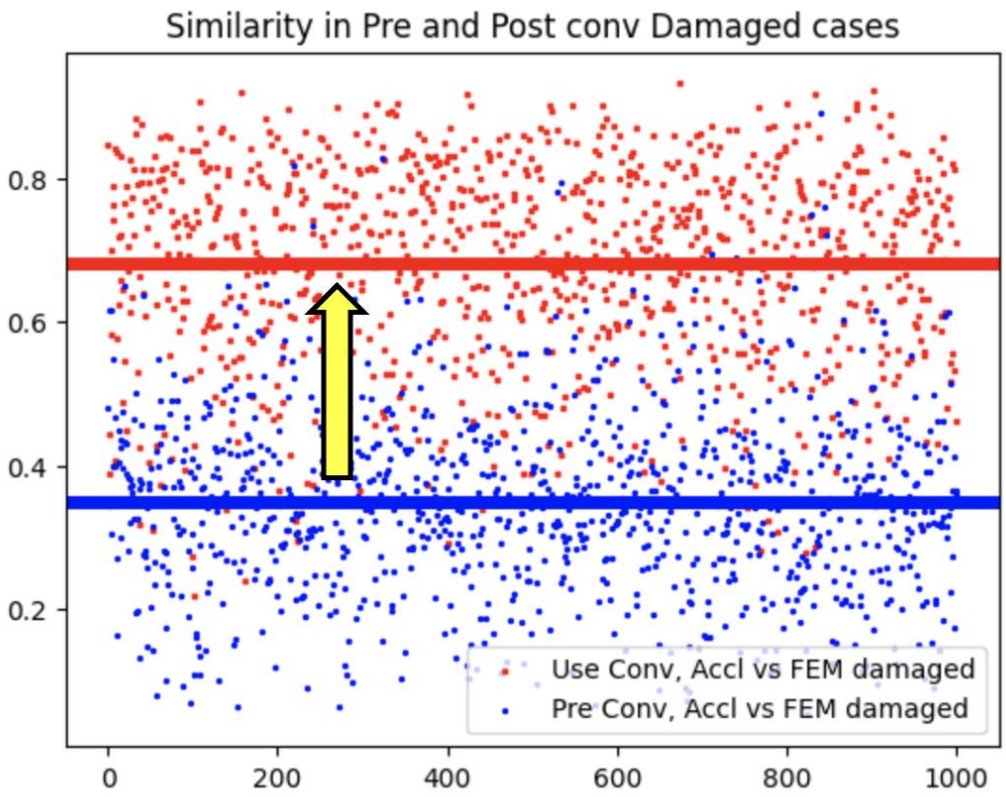


Figure 74. Cosine similarity comparison, pre and post convolution

Another bonus of this method is that it could act as a data augmentation technique. This method acts similarly as what we've shown in section 4.1.2, which could create a new data from two absolutely different examples, which makes the new data completely independent and the more amount of data that we have, our new type of data could grow exponentially. Let's assume that we have some data according to Table 4, after the convolutional process by randomly choosing two examples, we could have roughly three thousand times of unique data for us to work on because of unique combinations of two data (Table 5). This hugely increased pool of data allows us to create a much bigger database and network to fit the more complex data plane better. This concept also makes sense when we take the principle of superposition of structures, where a state of a linear

doi:10.6342/NTU202401548

invariant system could be seen as a liner combination of two different states. Here we are using the same concept, where we convolve two linear invariant system states to generate a unique set of data that might occur in the solution plane. By increasing the variability of the data, we are able to make our damage detection algorithm much more robust to different scenarios.

Table 4. Some cases of the original data

FEM – Healthy (FH)	3500 examples
FEM – Damaged (FD)	3500 examples
Real – Healthy (RH)	2000 examples
Real – Damaged (RD)	2000 examples

Table 5. Convolved unique cases of the original data

FH – RH	7,000,000 examples
FD – RH	7,000,000 examples
RH – RH	4,000,000 examples
Real – Damaged (RD)	4,000,000 examples

4.7.2 Single Channel Design and Healthy Optimized Training

In the traditional way of training neural networks, we would see each type of damage as a single class, and then concatenate the features from each channel to create a network that could recognize every damage class. This would create a neural network that is

doi:10.6342/NTU202401548

slightly bigger because of the larger input size of multiple channels that are concatenated together, and potentially solve a more complex problem. A bigger network is always better when we have more data to train on in the realm of deep learning, given that we have enough computational resources.

4.7.3 New Network Structure

Yet there is one problem that this type of network will face, which is when a different combination of damages is fed in the neural network, the neural network would guess it as a most probable case, meaning that this would lead to misdiagnosis of this situation. Let us assume that our network is only trained on classes where there is only a single channel that is damaged, if multiple damages simultaneously occur then the neural network will only give an predict the highest probability of damage, which would only result in good diagnosis in a single channel. This means that the other damaged channel would be neglected and thus result in major consequences. If we were to successfully diagnose both of the damaged channels, we would rely a lot on luck to give us an almost equal distribution of both of the damaged places or get data at the condition when both of the regions are damaged. For a simple four channel implementation, there will be in total 24 (4!) combinations of damage and we have to get an equal and ample amount of them to train our neural network. Because in complex structures, there might be more channels that we have to monitor, the combination and the amount of data that we need to get will grow exponentially, which is not realistic.

Inspired by the work of Onur Avci[31], a de-centralized single channel implement would solve this problem elegantly. In this type of design, we can reduce the problem of a single channel into a classic binary classification problem. This means that the combinations of damage at different channels could be simplified into just binary classification of the certain region. This means that we can accept different combination

doi:10.6342/NTU202401548

of damage inputs and identify the damage at each channel so that we don't need to know the damaged cases beforehand. This method is easy to conduct and would not require any previous knowledge of damage locations as the location of damage is now independent to the class. This also opens a new realm of training method that would greatly reduce the complexity of the problem.

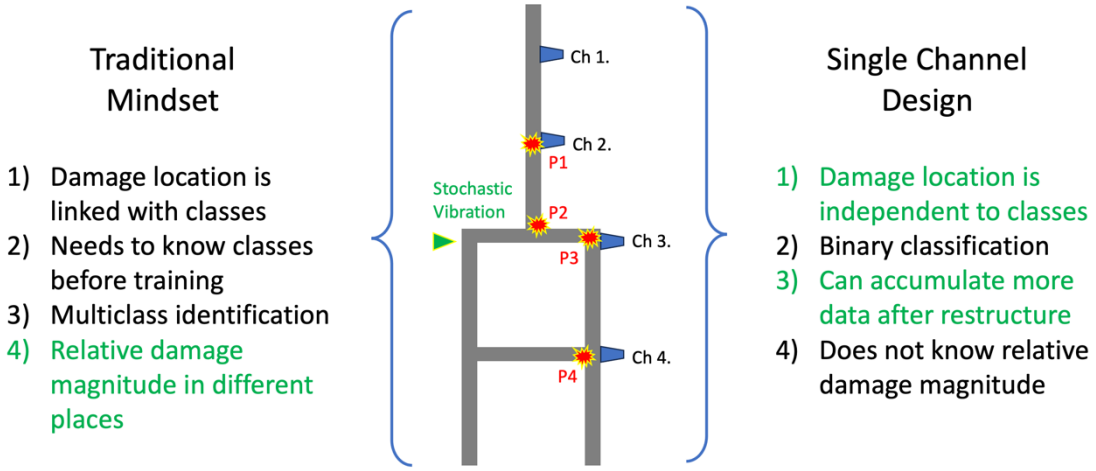


Figure 75. Traditional design vs Single Chanel Design

In order to over-come the problem un-identifiability of different combinations of damages, a single-channel DNN is proposed as a solution to this problem. In the previous section, all of the channels are concatenated into a single vector that is used as training examples and the output are locked as single damage, which would limit the neural network on learning different combinations. A single channel DNN only deals with the healthy state of the joint and it performs a binary classification task to identify the structure on whether if it is healthy or damaged. The single DNN only identifies the health state of the data of the single joint, and because of each DNN is independent any combination of damage on the joint can be identified without knowing the combination a-priori. This greatly increases the flexibility of the task without any need of additional data and the lightweight structure of the neural network means that it could be possibly

doi:10.6342/NTU202401548

be embedded in AI microchips onto the structure and achieve real-time identification without any added equipment. Practitioners can even increase or decrease the number of channels required for system identification without major modifications being made to the neural network, which would mean increased units and complexity and the need for more data to avoid overfitting issues.

This compact single channel DNN (called SC-DNN for short) is comprised of an input layer of 150 units and a hidden layer of 150 units. This set-up is chosen because in the previous task a single hidden layer seems to achieve the best performance along with the less computational cost. In the hidden layer a dropout ratio of 0.2 is used to prevent over fitting. There are only two output units that determine if the health state of the network, which is a traditional binary classification task.

Because we have reduced the problem into a simple binary classification task, we can focus the mapping of a complex solution plane of healthy and damaged data into a simple binary classification task of healthy data and unhealthy data. Because we don't have a lot of knowledge on unhealthy data in real life training scenarios, we can just simply optimize the training of the healthy dataset and then exclude all the others as non-healthy. Even when we have a relatively small unhealthy data to work on, we can use the hybrid database to supplement the damaged data amount so that we can use more data to train our neural network, and this will give us a better neural network that is able to fit multiple conditions better. This method will work even better when the real-life domain does not fit what we have suspected. The damaged data that we would get from real life is very limited at the initial phase of implementing this system, therefore we can use the hybrid database to humanly define the healthy and unhealthy state, such as when the stiffness has lost a certain percent that it would be unhealthy. Then the neural network will identify the case if it truly happened. If the solution plane is not what we have

doi:10.6342/NTU202401548

suspected, because of the network have great identification in the healthy data domain, any data that is not remotely alike the healthy data would be classified as unhealthy, therefore increasing the sensitivity of our classifier.

4.7.4 Cosine Similarity

In the audio recognition domain, there is a method called cosine similarity which measures the domain of two n^{th} degree vector. By creating word embeddings, which are the features of the words in a sentence, the computer algorithm is able to compute tasks such as measuring similarities of different translations with the same meaning, or even selecting a better translation depending on the context. The cosine similarity θ is simply the distance of angles between two n^{th} degree vector measured in radians in by the Euclidian cosine distance. Given two vectors A_i and B_i, we can calculate the cosine value of the two vectors by 5-2.

$$\theta = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2}\sqrt{\sum_{i=1}^{n} {B_i}^2}} \qquad 5\text{-}4$$

doi:10.6342/NTU202401548

We can draw some inspirations from the audio recognition domain by using this technique to measure the similarity of FEM data to our real-life data. In our current research, the spectrum is the identity of the case, which resembles the word embedding of the audio recognition researches. The two vectors A_i and B_i could both be the spectrum of case A and case B, and by calculating the spectrum of different cases, we can obtain and quantify the similarity or difference between spectrums in their own domain and even between domains. Most importantly, this tool could be used to measure the effectiveness of the method that we are presenting and further implementation details would be listed in the next section.

4.7.5 Visual Data Acquisition

To acquire the visual data, the single-lens camera used in this case is the Nikon D600. Its front and bottom panels are made of engineering plastic, while the top and back panels are made of magnesium-aluminum alloy, ensuring dust and splash resistance. Additionally, the camera body weighs only 760 grams, increasing to 850 grams when including the lithium battery and memory card. The unit pixel size can be used to calculate the scale factor, calculated by dividing the photosensitive element size by the resolution.

The process is to use the camera to acquire the video of the structure while it is excited with a stochastic double pendulum. The code provided by Tsai [50] is able to calculate the vibration of the structure frame by frame and output the time series displacement as a excel file. The file is then processed through FFT to create the spectra to feed in the neural network for identification. The four points of interest are also the points that the accelerometer traced in section 3. The four channels are calculated independently but simultaneously in the codes. The checkerboard is the reference point that does not move in relation to the structure.

doi:10.6342/NTU202401548

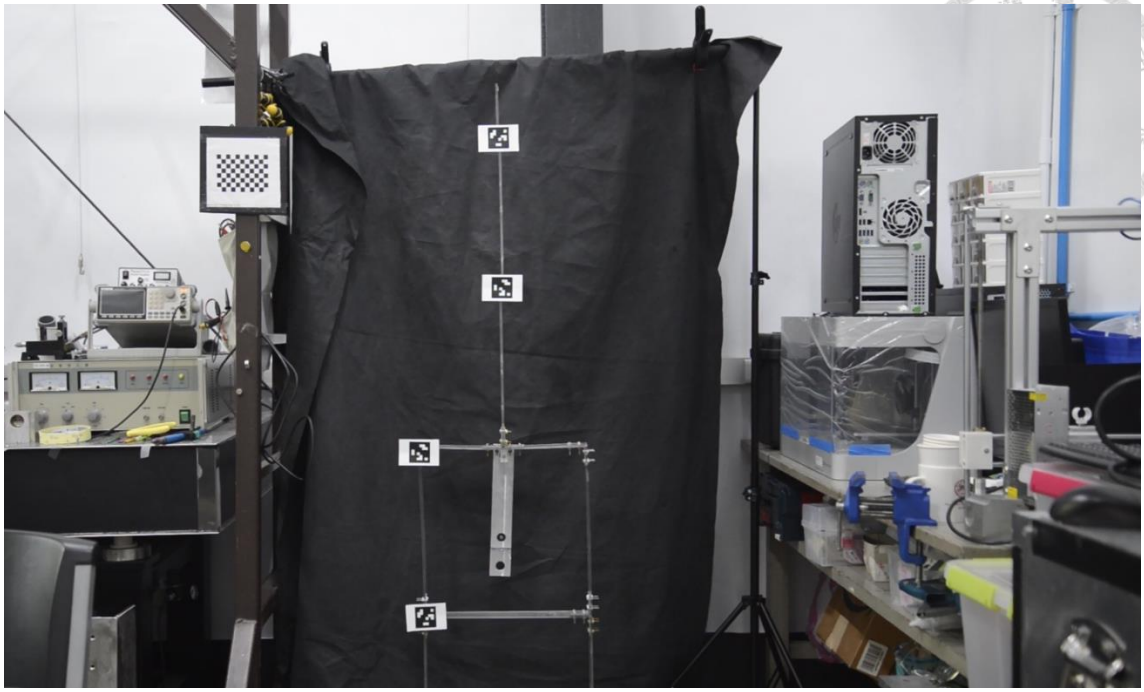


Figure 76. Experimental Setup of Visual Inspection Task

4.8 Strategies to address data mismatch

4.8.1 Data mismatch between FEM and Accelerometer Dataset

In the FEM domain, the simulated data are calculated in an ideal environment with homogeneous materials and perfectly connected boundary conditions. But in the real world, there are many inherent defects in the materials along with imperfect boundaries which are caused by natural causes. Along with the advances with AI and computing power, a new concept of digital twins has been introduced to supplement, simulate real world data. A problem then arises, which is the alikeness of real-world data with the simulated digital data. If the data are not alike, then the relevancy of the digital model would be challenged as it is not representative enough to the real-world data.

In order to address this problem, a simple experiment is used in order to quantify the difference of the real-life data compared to the FEM data. In this experiment, we used a measurement tool called cosine similarity (Section 3.3.3) to measure the similarity

doi:10.6342/NTU202401548

between data of two different domains.

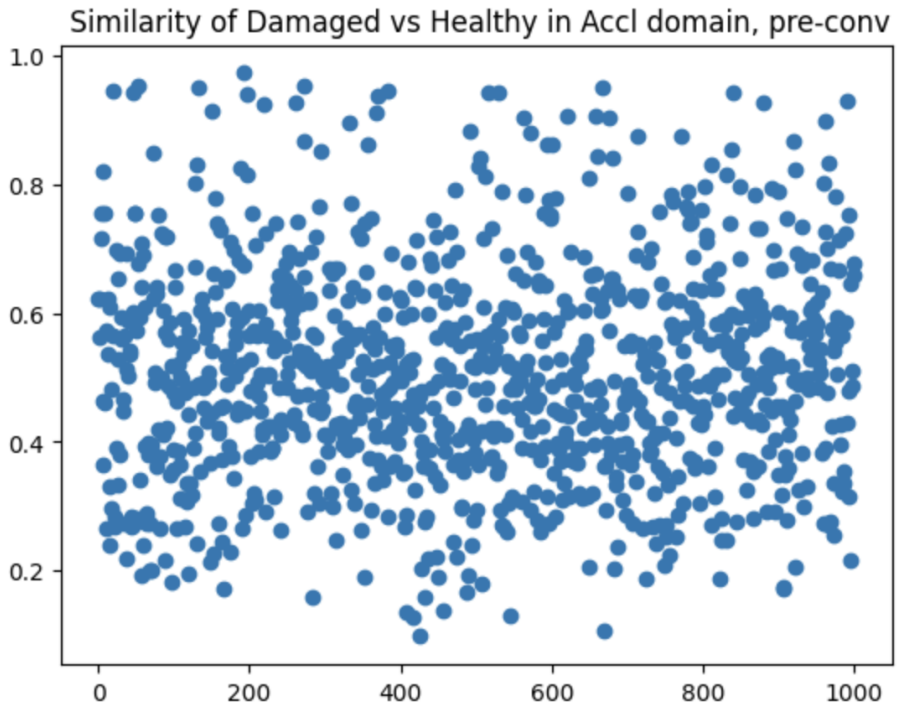


Figure 77. Similarity of Damaged vs Healthy of real life domain, pre convolution

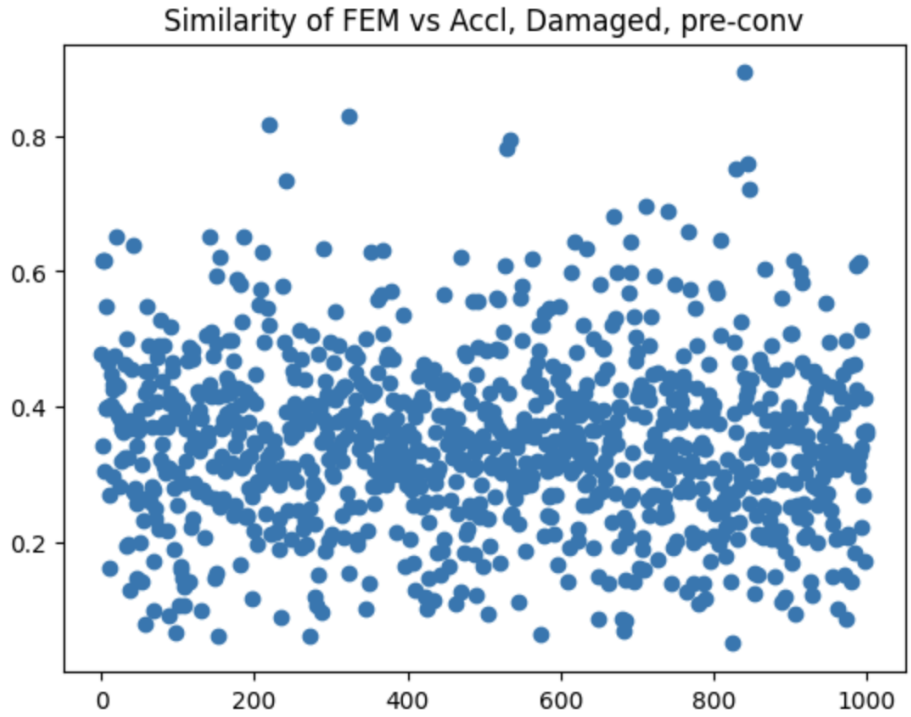


Figure 78. Similarity of FEM vs Real-life of domain of damaged data, pre convolution

doi:10.6342/NTU202401548

Cosine similarity measures the similarity of two vectors in nth space by doing a dot operation and measure the cosine values. The closer the value is to one, the more similar the two vectors are. In each operation, we have taken 1000 random examples from each pool (FEM-Healthy / FEM-Damaged / Accl-Healthy / Accl-Damaged) and applied cosine similarity operation to measure the similarity of the two sets of data. Firstly, we examined the similarity of damaged and healthy examples from the real-life domain before any operation (Figure 77). We have found out that most data have a cosine similarity of 0.5, which means that they are actually quite alike. In comparison, Figure 78 shows the similarity of the FEM generated data compared to the data from real-life of damaged cases before any operation. The cosine similarity dropped a large bit, with only 0.3 and most of the data clustered in this region. This means on average the similarity of the FEM generated data and the real-world data are quite apart, even compared to the healthy-and damaged data of the real-life domain. We can observe the difference in Figure 80, where the red data presents the real-world healthy – real-world damaged, and the blue data presents the FEM-damaged to real-world damaged data. We can see that there is a very noticeable gap in between the two sets of data.

Figure 79 shows the similarity of FEM generated data and real-life of healthy data, before any operation. It has the same attributes like the damaged case, with an average similarity score as low as 0.3.

doi:10.6342/NTU202401548

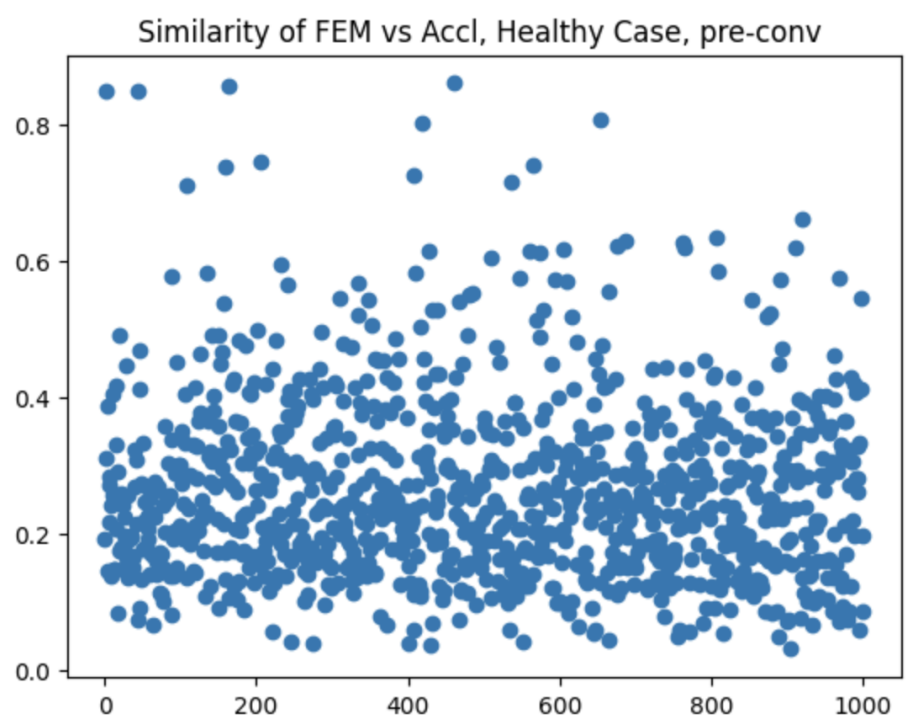


Figure 79. Similarity of FEM vs Real-life of domain of healthy data, pre convolution

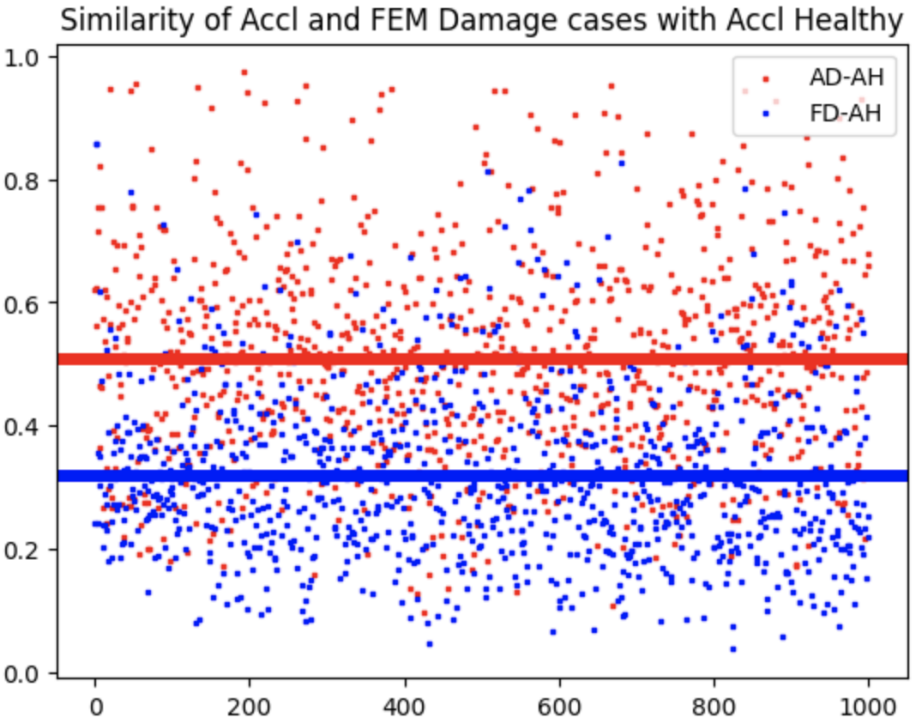


Figure 80. Similarity of real-world damaged cases and FEM damage cases with real-world healthy cases

doi:10.6342/NTU202401548

4.8.2 Transfer Learning

Transfer learning is a technique used to train across domains which uses a pre-trained network with a big set of data to perform identification tasks on a small set of data. Deep neural networks have hierarchical features which are coarse to fine from shallow to deep layers, practitioners usually leverage a network that is usually trained well on a lot of data to perform coarse analysis and then fine tune (or unfreeze) the deeper layers for a specific implementation.

To verify the transferrable nature of DNNs on the FEM data and the accelerometer database, we first let the DNN train on full FEM data. The FEM data are split into 57600 examples for training and 14400 examples for validation, the test set is omitted in this phase. After only 8 epochs of training on a batch size of 32, the SC-DNN has a loss of 0.0022 and accuracy of 99.97% on the training set and for the validation set it has a loss of 0.0046 and accuracy of 99.95%. It should be noted that the training and experiments covered in this section are all clipped with an early stopping of a tolerance of 5 epochs to prevent overfitting.

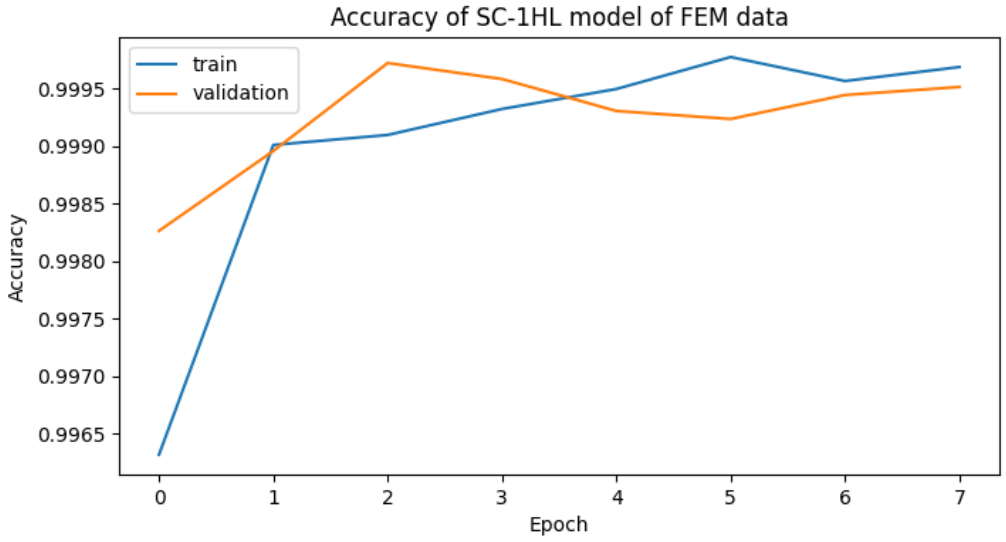


Figure 81. Accuracy of 1 hidden layer single channel model on FEM data

doi:10.6342/NTU202401548

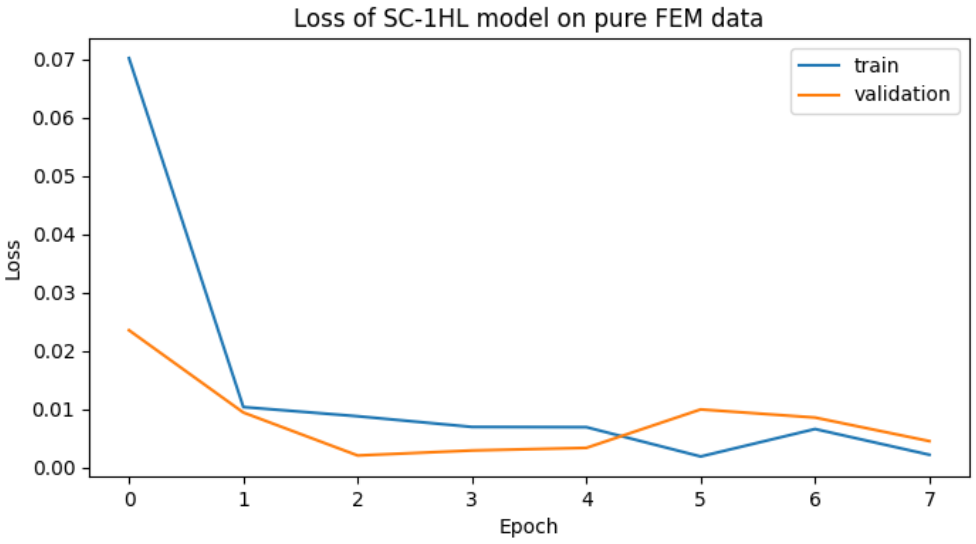


Figure 82. Loss of 1 hidden layer single channel model on FEM data

Table 6. Dataset Setup of pre-trained FEM Neural Network

Dataset	Source of Data	Total Percentage
Train	57600 Examples FEM Data (100%)	60%
Validation (Dev + Test)	14400 FEM Data (100%)	40%

Table 7. Dataset Setup for transfer learning tasks

Task	Dataset	Source of Data	Total Percentage
Transfer Learning 1	Train	5760 Accelerometer Data	60%
	Validation (Dev + Test)	1440 Accelerometer Data	40%
Transfer	Train	11520	60%

doi:10.6342/NTU202401548

Learning 2		Accelerometer Data	
	Validation (Dev + Test)	2880 Accelerometer Data	40%

Then we utilized the pre-trained model on a full accelerometer data with only 1/10 of the examples to demonstrate the power of transfer learning of DNNs. There are in total 5760 examples for training and 1440 examples for validation. The transfer learning in conducted in manner that only the last layer is able to train, which results in only 302 units for training. After 21 epochs of training, the loss in the training set is 0.1605 and the accuracy of the training set is 93.58%, the validation loss is 0.3731 and the validation set accuracy is 86.53%. The results are decent but there are a lot of space for improvement.

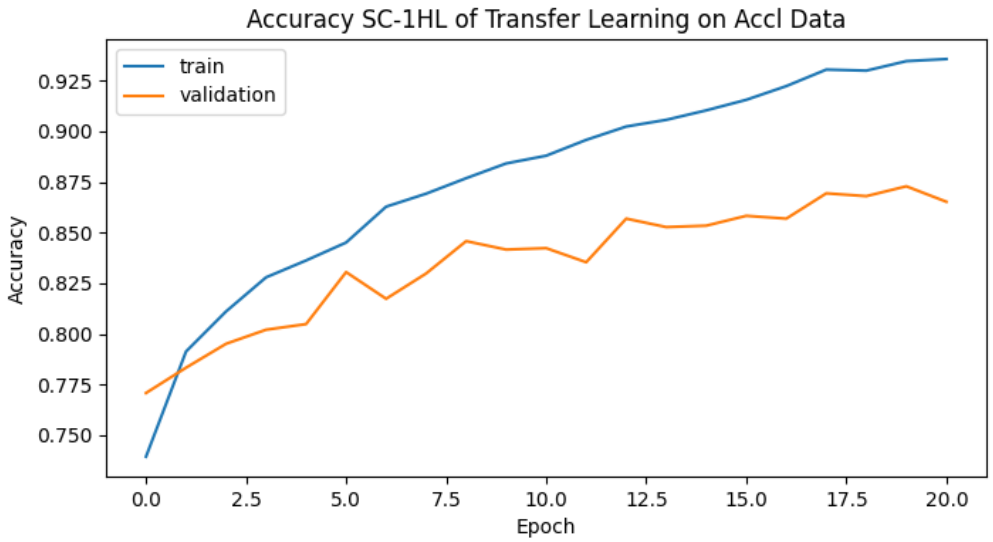


Figure 83. Transfer Learning Accuracy of 1 hidden layer single channel model on 10% Accelerometer data

doi:10.6342/NTU202401548

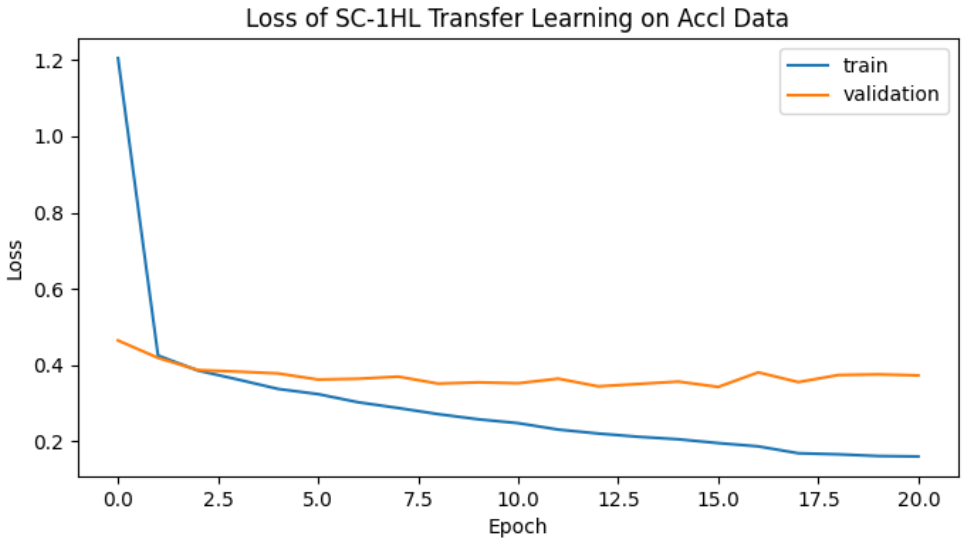


Figure 84. Transfer Learning Loss of 1 hidden layer single channel model on 10% Accelerometer data

Now we transfer the emphasis of transfer learning on getting it to learn with more accelerometer data. In this experiment we use 20% of the original accelerometer data after data augmentation, which results in total 11520 examples for training and 2880 examples for validation. As expected, the accuracy has risen after the boosted training set after transfer learning. The loss of the training set is 0.1347 and the accuracy of the training set is 94.70%. The validation loss is 0.2948 and the validation accuracy is 89.62%, which is slightly better than only using 10% of the original data. The boost in the accuracy comes from the high cost of using twice the amount of data to train, therefore it slightly defeats the purpose of using less real-life data for training to achieve a reasonably good result.

In order to further validate the effects of transfer learning, the following experiment used the DNN to directly train on 10% of the data without any pre-training. The boosted accuracy will come from the result of learning the base features from the FEM database and thus verifying the impact of transfer learning strategy to connect artificial data to real-life data. The 1 hidden layer SC-DNN has been used for this experiment and all of the

doi:10.6342/NTU202401548

training parameters are the same, including a learning rate of 0.001. After 21 epochs of training, the loss of the training set is 0.1896 and the accuracy is 92.5%, the validation dataset loss is 0.6732 and the accuracy of the validation set is 85.07%.

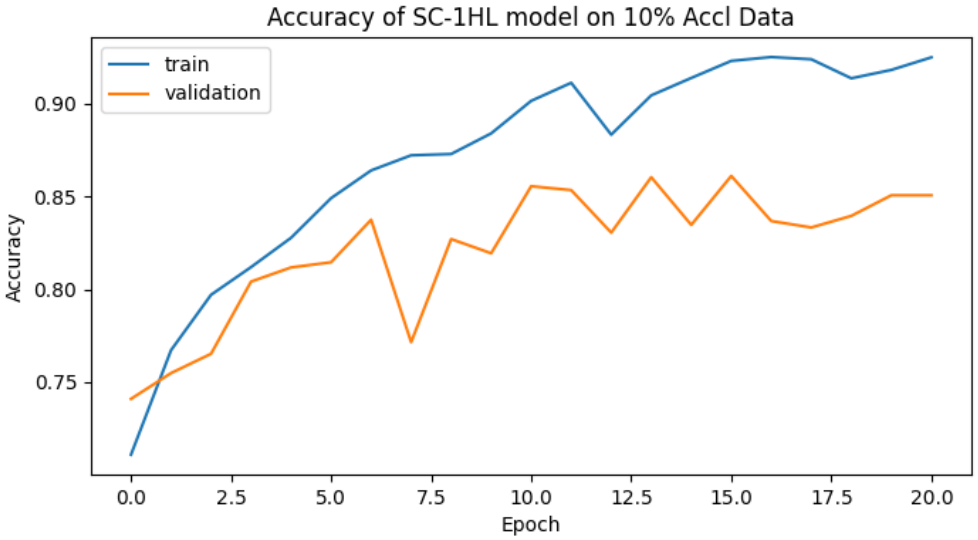


Figure 85. Accuracy of direct training on accelerometer data of 1 hidden layer data

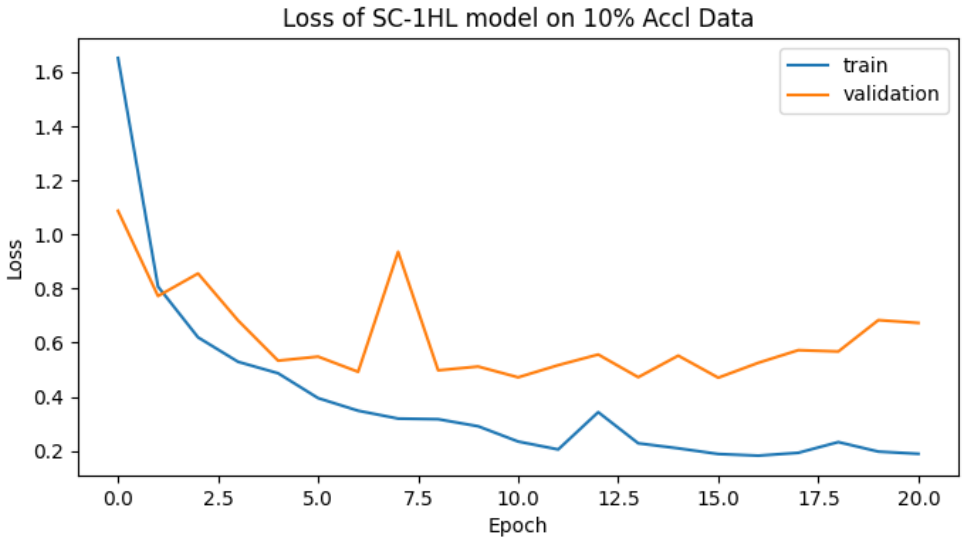


Figure 86. Accuracy of direct training on accelerometer data of 1 hidden layer data

Compared to the results of the transfer learning SC-DNN, we can see that with just one layer, transfer learning didn't work as well since direct training yielded almost the same results as transfer learning with the same amount of data. The reason why the

doi:10.6342/NTU202401548

transfer learning result is not as prominent is that a single layer and 302 units may be too little for transfer training. Therefore, a two-layer design in this sub-section has been attempted so that the base features could be pretrained and a higher-level feature could be updated by a different set of data. The two-layer DNN architecture is as follows: The first layer has 150 units to match the units from the arranged single channel frequency spectrum. The second layer is half of the units (75 units) so that the prominent units can be selected and a decision could be made in the final layer, which only has two units to perform a binary classification task. The same training set is used for the 2HL-SC DNN as the previous 1HL-SC DNN transfer learning task and the dataset that it is trained on is a full FEM generated dataset.

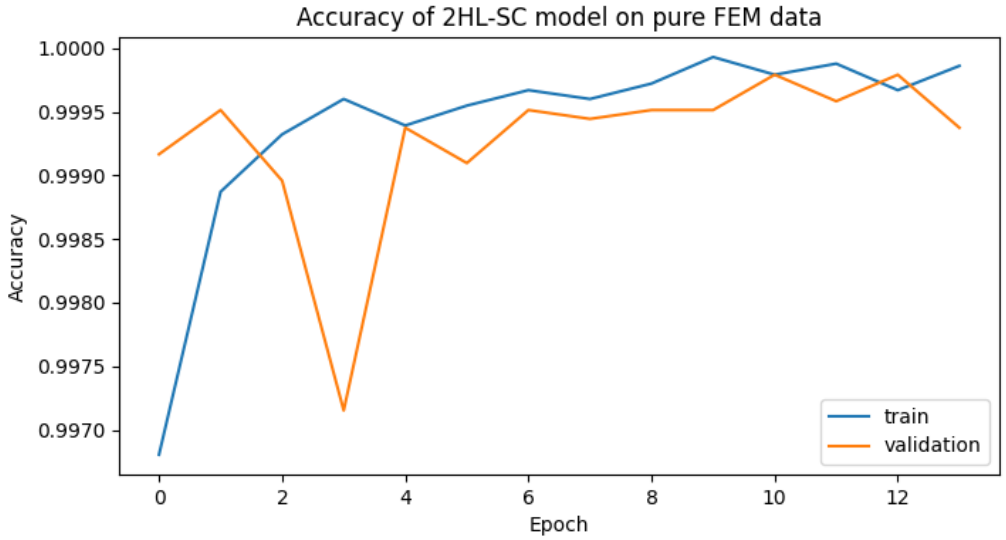


Figure 87. Accuracy of 2 hidden layer single channel model on FEM data

doi:10.6342/NTU202401548

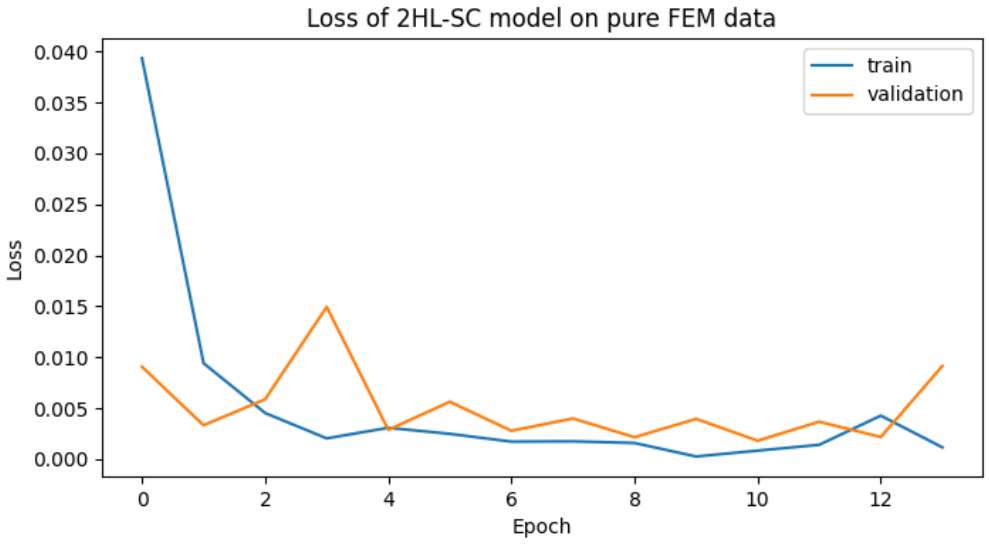


Figure 88. Loss of 2 hidden layer single channel model on FEM data

After 12 epochs of training on pure FEM generated data, the loss of the training set is 0.0011 and the accuracy of the training set is 99.99%. The validation set has a loss of 0.0091 along with an accuracy of 99.94% Which indicates that the 2HL-SC neural network can fit the data really well for frequency identification task.

For the transfer learning task, the same procedure and training mechanisms are used for cross-validation between models. Firstly, the network is trained on a 10% of the accelerometer data on a transfer learning task, which unfreezes the last two layers of the neural network. This opens up the trainable parameters to 11325, which is almost 30 times the amount of the 1HL-SC model. After applying transfer learning, the loss of the training set is 0.2045 and the accuracy is 90.36%, the validation set loss is 0.3747 and the validation set accuracy is 85.07%, which are both slightly lower than the previous model.

doi:10.6342/NTU202401548

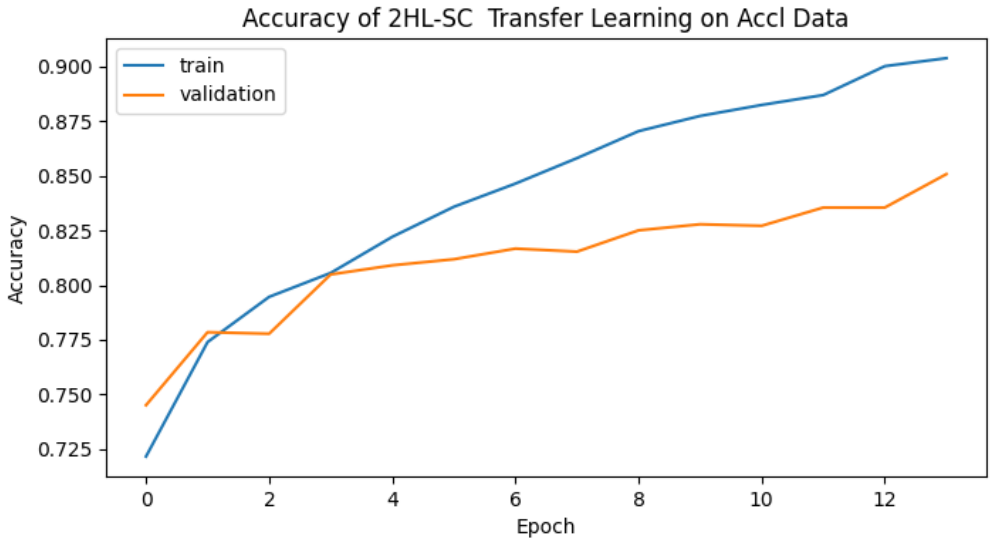


Figure 89. Transfer Learning Accuracy of 2 hidden layer single channel model on 10% Accelerometer data

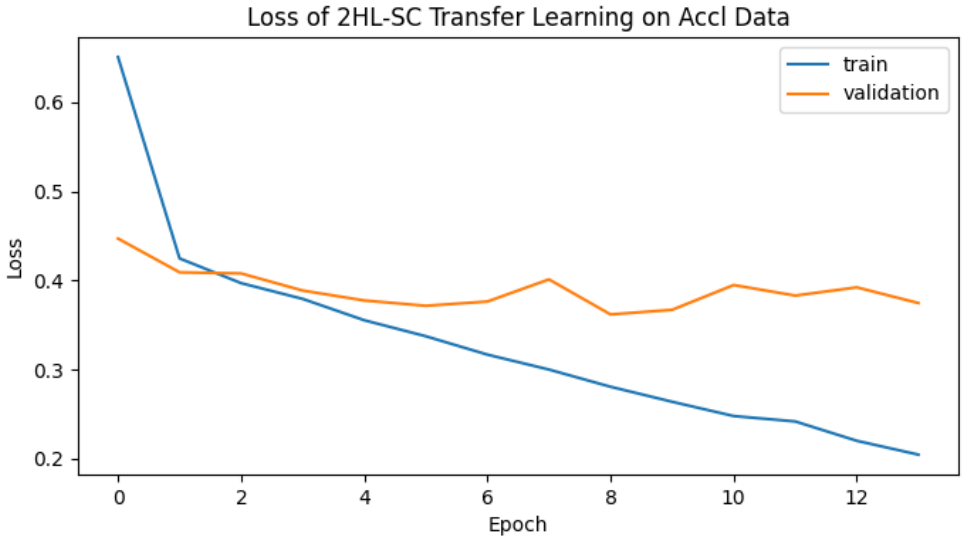


Figure 90. Transfer Learning Loss of 2 hidden layer single channel model on 10% Accelerometer data

Now 20% of the acceleration data are used to demonstrate and verify the boost of accuracy of the increased layer. The loss of the training set is 0.0824 and the accuracy of the training set is 96.88%. The loss of the validation set is 0.3416 and the accuracy of the validation set is 90.66%, which performed better than the 1HL model.

doi:10.6342/NTU202401548

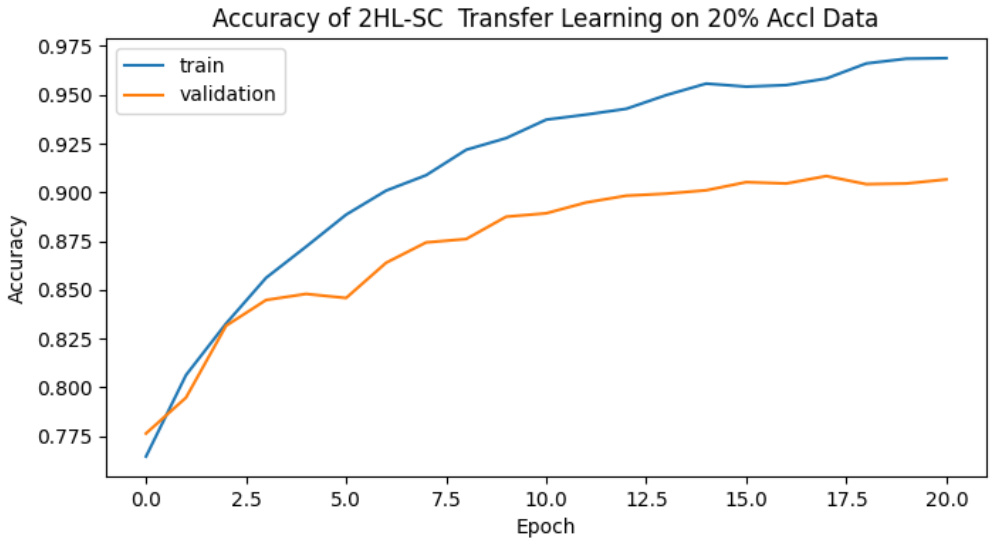


Figure 91. Transfer Learning Accuracy of 2 hidden layer single channel model on 20% Accelerometer data

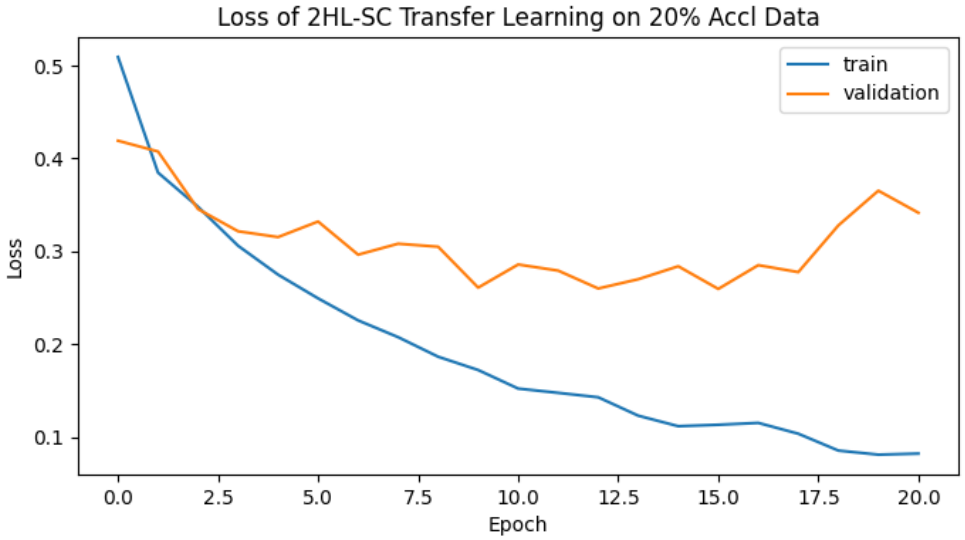


Figure 92. Transfer Learning Loss of 2 hidden layer single channel model on 20% Accelerometer data

doi:10.6342/NTU202401548

Table 8. Comparison of Transfer Learning Models

Model	Training set accuracy	Validation set accuracy
Direct Training 10%	92.5%	85.07%
Transfer Learning (1HL) 10% Data	93.58%	86.53%
Transfer Learning (1HL) 20% Data	94.70%	89.62
Transfer Learning (2HL) 10% Data	90.36%	85.07%
Transfer Learning (1HL) 20% Data	96.88%.	90.66%

The added data for transfer learning have accounted for 5% of more accuracy for the whole neural network, but too much leveraging on the added data will defeat the purpose of using transfer learning from a big dataset to a smaller dataset. Although the results of transfer learning are promising and sufficiently accurate compared to peer studies the results still have a lot more to be desired.

4.8.3 Mixed Features Hybrid Database

Another method is to use the hybrid database to train both of the accelerometer dataset and the FEM generated dataset together to create a bigger pool of training set so that the neural network could have the possibility of fitting the accelerometer dataset better. Theoretically, by increasing the data-pool, the DNN would have the opportunity to train the data longer, thus would result in a better accuracy and a better fitting algorithm. In traditional deep learning training strategies, to deal with a completely new subset of data and especially with a much smaller data-size, it is common for practitioners to mix

doi:10.6342/NTU202401548

the new subset of data with the original dataset and use the NN to let it fit directly.

The first strategy to use for this this common problem of fitting a NN from different subsets of data is to mix the new and smaller subset into the validation set, which would then be split into a development set and test set. The purpose of the development set is the same as the validation set in the previous section, which mainly deals with cross validation of the neural network. During the training phase the development set such that it is unable to change the parameters of the neural network but the performance would be evaluated on each epoch to adjust the training procedure of the neural network. And the test set is a completely new set of data, which would only be "seen" by the neural network after the training is completed and its only purpose is to be used to evaluate the performance of the NN.

In the first setup, the training set is comprised of 90% of FEM generated data and 10% of accelerometer data, which would be in line of our purpose of successfully fitting the dataset on just little percentage of accelerometer data. The full composition of the training set is listed in the table below. The 1HL-SC neural network is first trained on the hybrid training data so that it could learn how to recognize the patterns of different damaged scenarios, and then each training epoch the neural network is then verified from the results of the development set, which is comprised of 100% of accelerometer data which is not used to optimize the training process. At last, it is evaluated by a test set that is also comprised by pure accelerometer data which is not seen by the NN beforehand.

doi:10.6342/NTU202401548

Table 9. Dataset Setup of Data Augmented Hybrid Database V1

Dataset	**Source of Data**	**Total Percentage**
Train	57600 Examples FEM Data (90.9%) + Accelerometer Data (9.1%)	76.9.%
Bridge	7200 Examples FEM Data (90.9%) + Accelerometer Data (9.1%)	7.7%
Development	7200 Examples Accelerometer Data (100%)	7.7%
Test	7200 Examples Accelerometer Data (100%)	7.7%

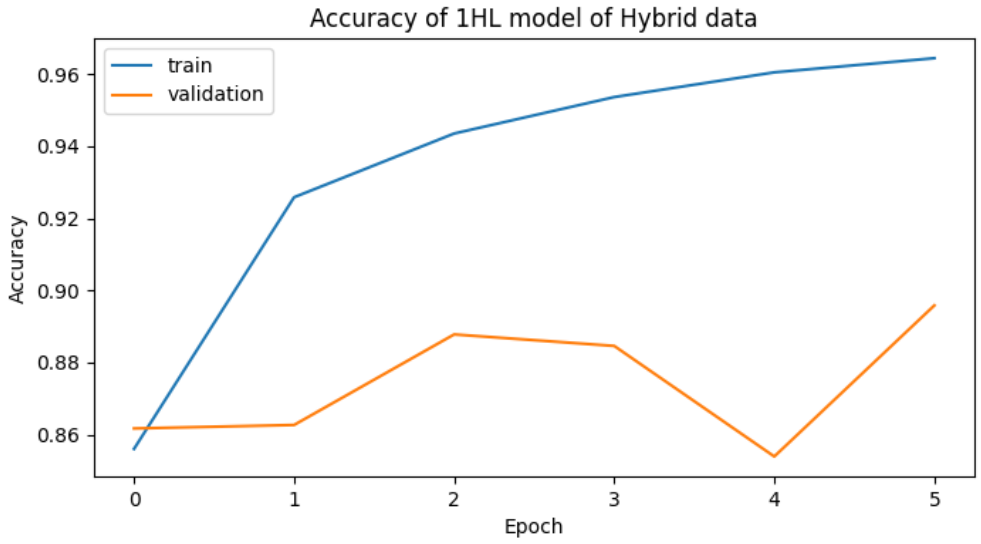


Figure 93. Accuracy of Hybrid data with 1HL model

doi:10.6342/NTU202401548

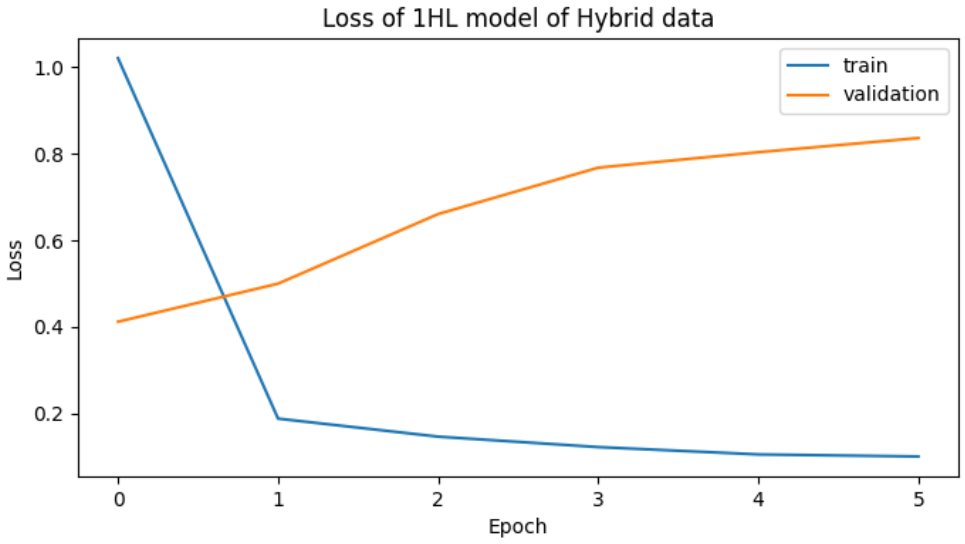


Figure 94. Loss of Hybrid data with 1HL model

After 8 epochs of training, the loss of the training set is 0.1003 and the accuracy of the training set is 96.45. The loss of the development set is 0.8365 and the accuracy of the development set is 89.58%. At this point, when we used only 10% of the data for training the neural network, the results are just slightly better than the cases used for transfer learning. After a fine-tuned process with setting the base learning rate to one-tenth of the original learning rate, the results are slightly better, with the training set accuracy boosted to 98.73% and the validation set accuracy boosted to 90.10. The fine tune process didn't do a lot for the validation set. Then the network is validated through the test data and the bridge data. The full result is shown in Figure 95, where the accuracy of the test data is 91.92% (the second index in the first array), which means that after slight fine tuning, the neural network has good generalizability toward the a completely different test data. And the bridge data resulted in a greater accuracy at 98.61%(the second index in the second array), which indicates that the data mismatch problem is still apparent in the hybrid database by just mixing the accelerometer data and it resulted in the 7% difference between the bridge data and test data.

doi:10.6342/NTU202401548

```
Evaluate on test data
225/225 [==============================] - 0s 892us/step - loss: 0.1964 - accuracy: 0.9192
test loss, test acc: [0.19644127786159515, 0.9191666841506958]
Evaluate on bridge data
225/225 [==============================] - 0s 883us/step - loss: 0.0517 - accuracy: 0.9861
test loss, test acc: [0.05169632285833359, 0.9861111044883728]
```

Figure 95. Test and bridge data analysis

Another attempt has been made to offset this problem by directly mixing the signals of the accelerometer data and the FEM generated data. This trial uses a frequency mixing approach which has been documented in the literature review above as a feasible tool to artificially generate more data, but in this case to mix data features so that the accelerometer data could be trained for more times with more accuracy. In the following figure shows the first 5 examples of mixed data.

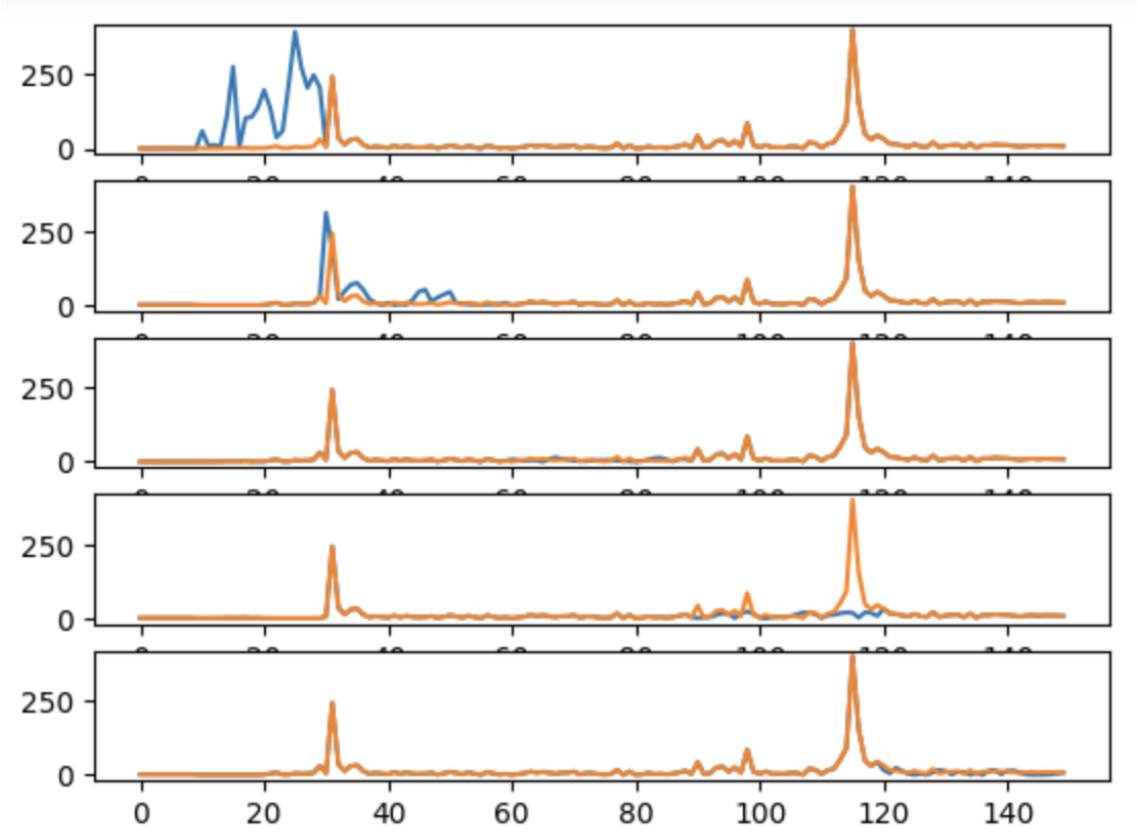


Figure 96. Mixed Feature Examples

doi:10.6342/NTU202401548

For the training procedure, in order to obtain a deeper knowledge of the database and the neural network, we have dropped the early stopping procedure and trained for more epochs to know how would the neural network fit the database and when will the loss and accuracy plateau. Further changes would then be used after this first step to optimize the neural network by observing the performance with the development set. Currently the neural network is trained on a default of 100 epochs without any early stopping or regularization techniques.

Table 10. Dataset Setup of Data Augmented Mixed features

Dataset	**Source of Data**	**Total Percentage**
Train	57600 Examples FEM Data (90.9%) + Accelerometer Data (9.1%)	76.9.%
Bridge	7200 Examples FEM Data (90.9%) + Accelerometer Data (9.1%)	7.7%
Development	7200 Examples Accelerometer Data (100%)	7.7%
Test	7200 Examples Accelerometer Data (100%)	7.7%

doi:10.6342/NTU202401548

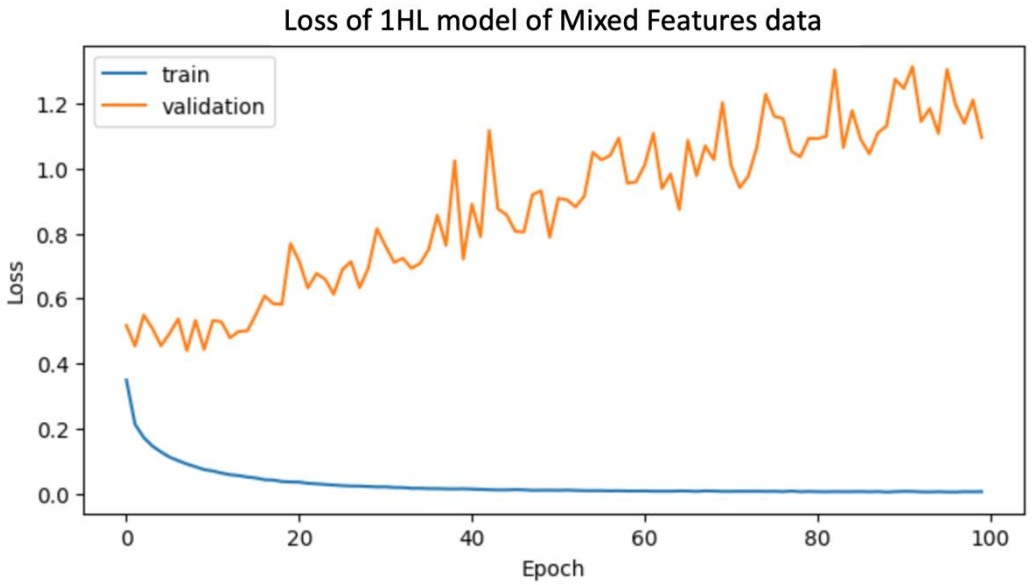


Figure 97. Loss of 1HL model of Mixed Features data

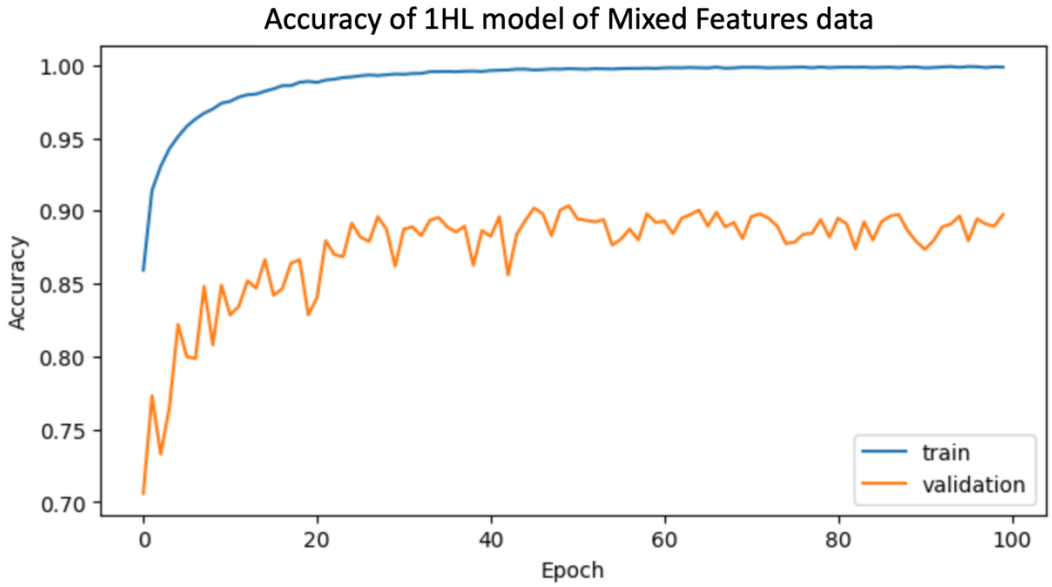


Figure 98. Accuracy of 1HL model of Mixed Features data

The loss of the mixed features hybrid dataset is 0.0151 and the training set accuracy is 99.55%. The development set loss is 0.2957 and the validation accuracy is 89.75%. If we clipped the training at the 25th epoch when the neural network starts to severely overfit, the training accuracy is 97.35% and the validation set accuracy is 89.00%. This value is

doi:10.6342/NTU202401548

not as high as we would suspect, which means that there are some underlying issues that we need to address.

```
Evaluate on test data
63/63 [==============================] - 0s 1ms/step - loss: 0.3637 - accuracy: 0.9180
test loss, test acc: [0.3636779487133026, 0.9179999828338623]
Evaluate on bridge data
63/63 [==============================] - 0s 1ms/step - loss: 0.0592 - accuracy: 0.9825
test loss, test acc: [0.05917336419224739, 0.9825000166893005]
```

Figure 99. Analysis on the bridge data of the Mixed 1HL model

By using a bridge dataset, which is a subset of data that is randomly selected from the training set and evaluating it, we can see the discrepancy between the data mismatch and see if there's any method that we can address it. A test set is used as an unbiased set to examine the performance of the neural network after it is finished training. In Figure 99, we can see the full evaluation of the test and bridge data in comparison of the test data. We can see that the bridge data has very high accuracy (98.25%, the second index of the second matrix, bridge data) whereas the test data only has slightly better accuracy than the test set (91.8%, the second index of the first matrix, test data). Note that the test set has the same distribution of data as the validation set, which is the distribution that we want to optimize. And the bridge set has the same distribution as the training set, which is what we used to train the network. We can see that there's a high discrepancy between the accuracy of the test and bridge set, this means that there is a high data mismatch problem, even when we mixed the features of real-life data into our bridge set.

We can clearly observe the problem that we discussed in section 3.3.4, where when we mix the features of a two vastly different sources would create a new source of data, and if we let the neural network train on the new source of data, it will adapt to this domain and then optimize on the data that we don't want it to optimize on. The neural network would not learn the single features that we mixed, instead it learns the entire spectrum and fits the solution plane as a new variation of data.

doi:10.6342/NTU202401548

4.8.4 Healthy Optimized Hybrid Dataset

Because of the re-structuring of the problem, we can try to look at the question at a different angle, which might broaden the view and usage of this implementation. The usage of a Hybrid database is to use FEM to simulate a different type of damage that would be able to supplement the lack of damage in the Accelerometer database. In a simplified binary classification task, the NN would either classify the data into two categories, which are healthy data and unhealthy data. If we were to optimize the recognition of the healthy state structure, then by logic the other types of data as damaged, therefore increasing the accuracy of the neural network.

In order to verify this hypothesis, the author utilized a training method that leverage on the hybrid database and the deep neural network. In order to make sure the neural network can "recognize" the healthy data of the real-life structure well; the training set is comprised of 50% of real-life structure data in a healthy state. Because of the accessibility of healthy state real life structures, the healthy database could be considered limitless and very humanly definable, therefore it makes it ideal to learn the complex features of a healthy structure in different states of input. The first component of the damaged data is comprised of 90% of FEM generated data, which make up 45% of the total composition, is randomly selected from the database and is made to simulate novel damaged data. The second component is real-life damaged data, which are randomly selected from the Accelerometer database, and it makes up 10% of the damaged data and 5% of the total training data. This approach makes sure that we don't overfit the healthy data, but instead learn the important features of the healthy data in accordance to the other unhealthy data, which makes the following classification task possible while dealing with a severe data mismatch problem.

The development and test set are made up in a different composition, which would

doi:10.6342/NTU202401548

ensure that real-life data could play a bigger role in training. The development and test set are firstly made up with 50% of healthy real-life structure data and 50% of damaged data. The damaged data is comprised of 50% of FEM generated data which are also randomly selected from the database and also 50% of Accelerometer data to make sure that the it resembles the real-world scenario more and ensuring enough damage variety and examples. The full hybrid dataset is shown below.

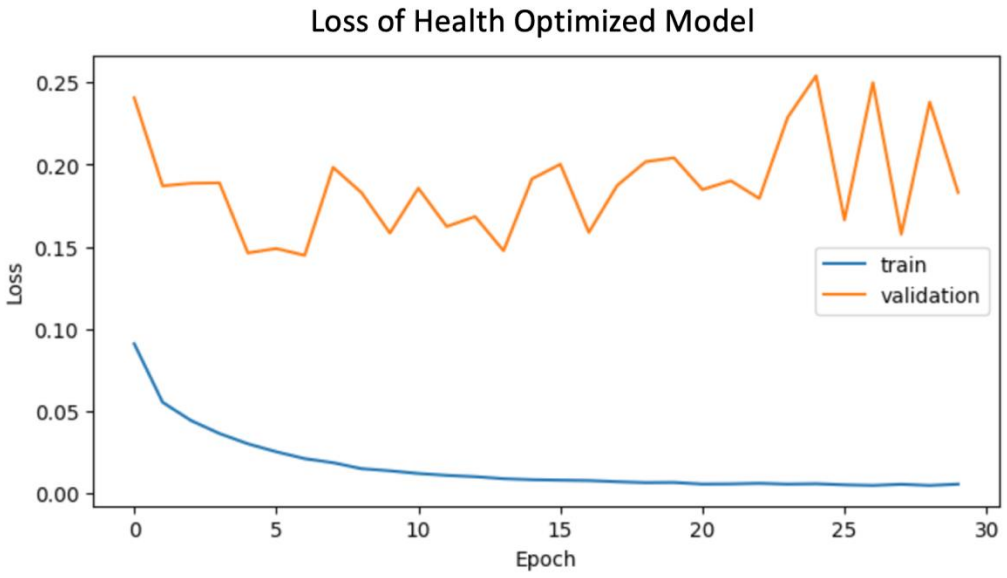


Figure 100. Loss of Health Optimized Model

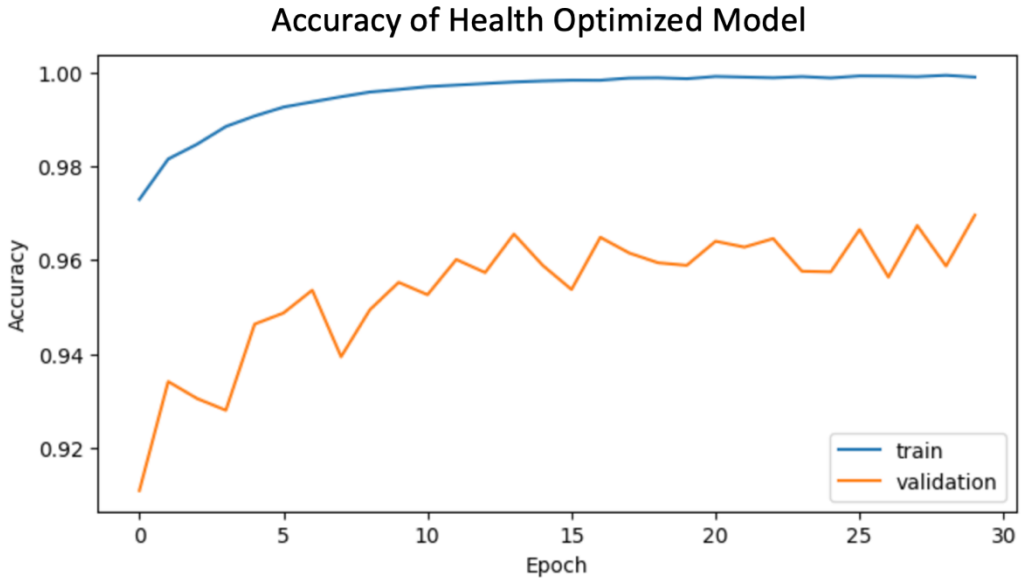


Figure 101. Accuracy of Health Optimized Model

doi:10.6342/NTU202401548

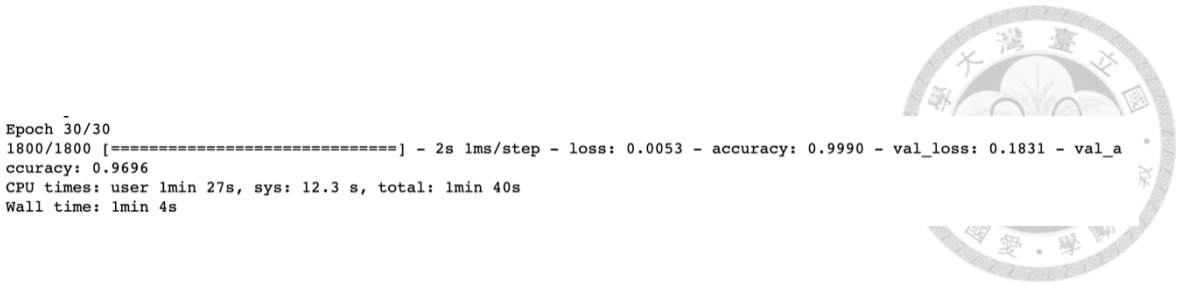


Figure 102. Training Results of Health Optimized Model

To prevent overfitting, the epochs are limited at 30 according to experience of the training, which only has slight turbulence in the loss function before it increases too much. In Figure 102, we can observe that the training set has a great accuracy of 99.9%, and the accuracy of the validation set is at 96.9%. This means that the classifier can generalize to a different composition of data quite naturally while obtaining great accuracy inside the datasets. The training time takes only above a minute to complete with the full set of data.

```
Evaluate on test data
225/225 [==========] - 0s 823us/step - loss: 0.1596 - accuracy: 0.9690
test loss, test acc: [0.15956205129623413, 0.9690277576446533]
```

Figure 103. Test set evaluation of Health Optimized Model

Upon further inspection on the test, we can see that in Figure 103, the loss of the test set is around 0.15 and the accuracy of the test set is 96.9% as seen in the matrix of the bottom row. With more accurate analyzation in Figure 104, the test set only identifies 6 cases of “healthy” data to unhealthy data, which means that the features of the healthy data are learned quite well during training. There are 217 cases that are false negatives, which are unhealthy data that are classified as healthy. Upon closer inspection, most (about 75~80%) of the data that are false negatives come from the introduced accelerometer damaged database, and FEM data have about (20~ 25 %) that contribute to this error. The reason of this comes from the limited amount of accelerometer damaged data, but never the less it still generalizes really well for novel damaged accelerometer data. And it is because of the limited amount of damaged data of the from the

doi:10.6342/NTU202401548

accelerometer pile, the classifier still has some ambiguity between the healthy and damaged cases of the accelerometer data. This is result is currently the best when we have limited damaged data to make use of, and it could also generalize really well towards clearly unseen cases. The possibility of good diagnosis of healthy and unhealthy data is good, and thus when we string along minutes of data, the possibility of misdiagnosis is very small.

```
225/225 [==============================] - 0s 834us/step
false positives: 6.0
false negatives: 217.0
```

Figure 104. Mis-labelled cases of Health Optimized Model

When trained a clean network on a full accelerometer database, which does not have the benefits the hybrid database of learning a different variety of data, we have yielded slightly better results. With the same amount of damaged data, we are bound to cut down the amount of data of the healthy database, which would result in risk of lower accuracy or if we change the damaged-undamaged distribution so that it biases the healthy distribution much more, the neural network would have a great bias toward the majority of the database. After 100 epochs of training, the result is documented in Figure 105. The accuracy comes at 93% of the validation set, which is about 4% lower than our hybrid database model. The variance is lower as expected because the train, development set and come from the same distribution. But overall, the hybrid model can accept more variety of data and has a higher accuracy and lower error. The training time is 42.6 seconds and it is to be expected because it trains on less data.

```
Epoch 100/100
324/324 [==============================] - 0s 1ms/step - loss: 0.1143 - accuracy: 0.9549 - val_loss: 0.2089 - val_acc
uracy: 0.9323
CPU times: user 54.2 s, sys: 7.34 s, total: 1min 1s
Wall time: 42.6 s
```

Figure 105. Direct Training Results on accelerometer database of 1HL Model

doi:10.6342/NTU202401548

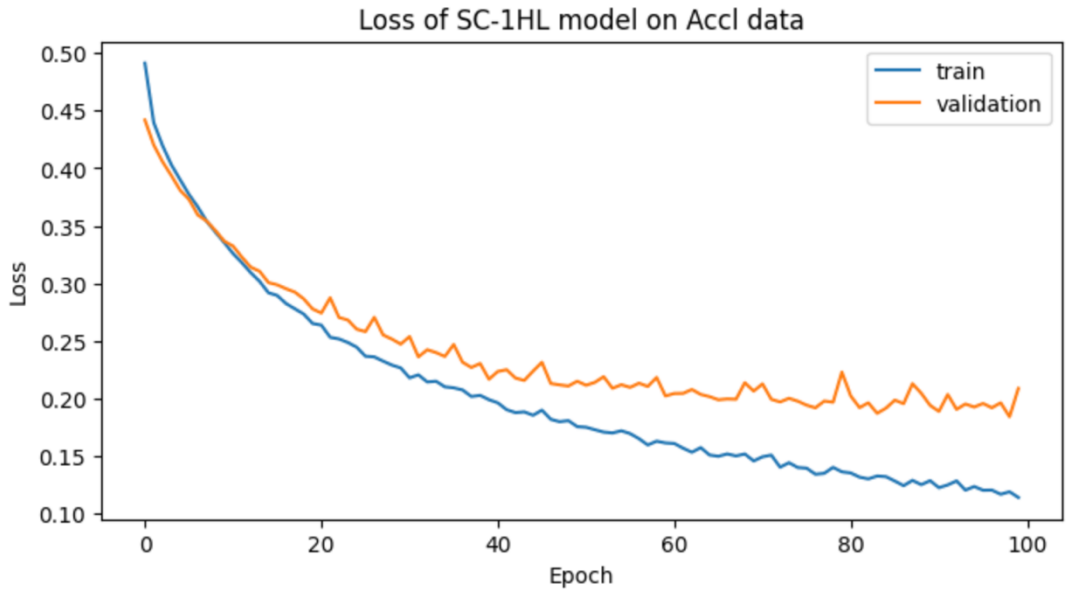


Figure 106. Loss of 1HL Model with direct training on Accelerometer Data

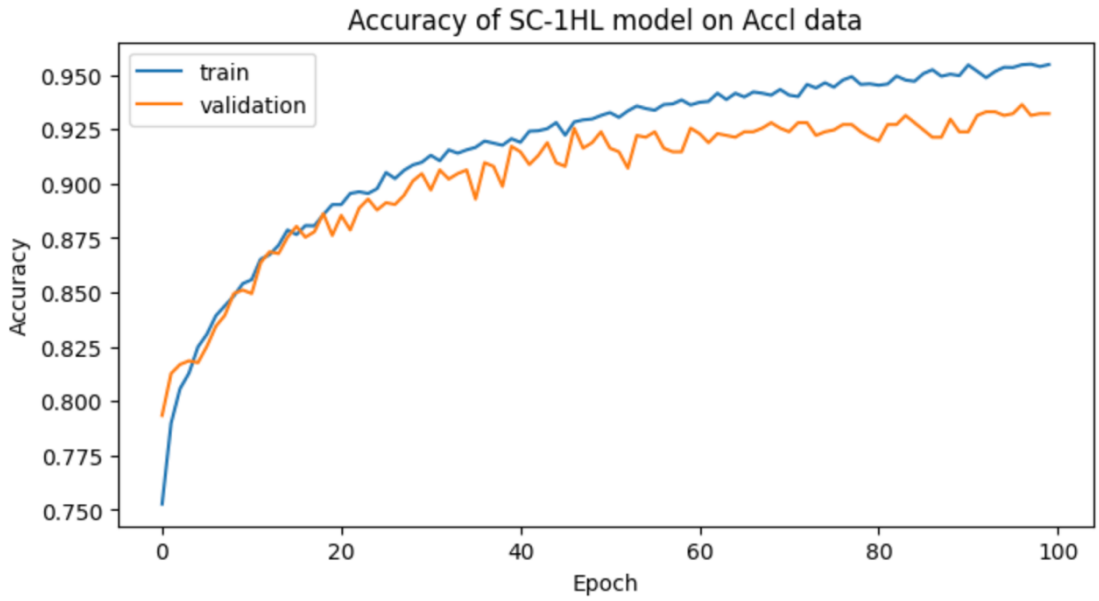


Figure 107. Accuracy of 1HL Model with direct training on Accelerometer Data

doi:10.6342/NTU202401548

4.8.5 Convolution Mixed Features

Table 11. Original data used for analysis

Original Class	**Examples**
FEM – Healthy (FH)	3500 examples
FEM – Damaged (FD)	3500 examples
Real – Healthy (RH)	2000 examples
Real – Damaged (RD)	2000 examples

The convolutional operation could act as a way to artificially generate more data. In Table 11, we can see the original data used for this task. Also, since we are using a single channel design, we can fit more examples in healthy and damage cases so that the neural network can have more data and less classes to train on. In this operation, we choose a sample of the data and convolve it with the real-life, healthy data thus giving us much more unique combinations of different data, expanding the database size in 10^3 to 10^4, as shown in Table 12. Convolution can be used as a means of data-augmentation technique which gives us unique combinations instead of just masking from the same spectrum, increasing the variability of the database with just a small amount of data.

Table 12. Total combinations of data used for convolutional analysis

Convolutional Plane Class	**Examples**
FH – RH	7,000,000 examples
FD – RH	7,000,000 examples
RD – RH	4,000,000 examples

doi:10.6342/NTU202401548

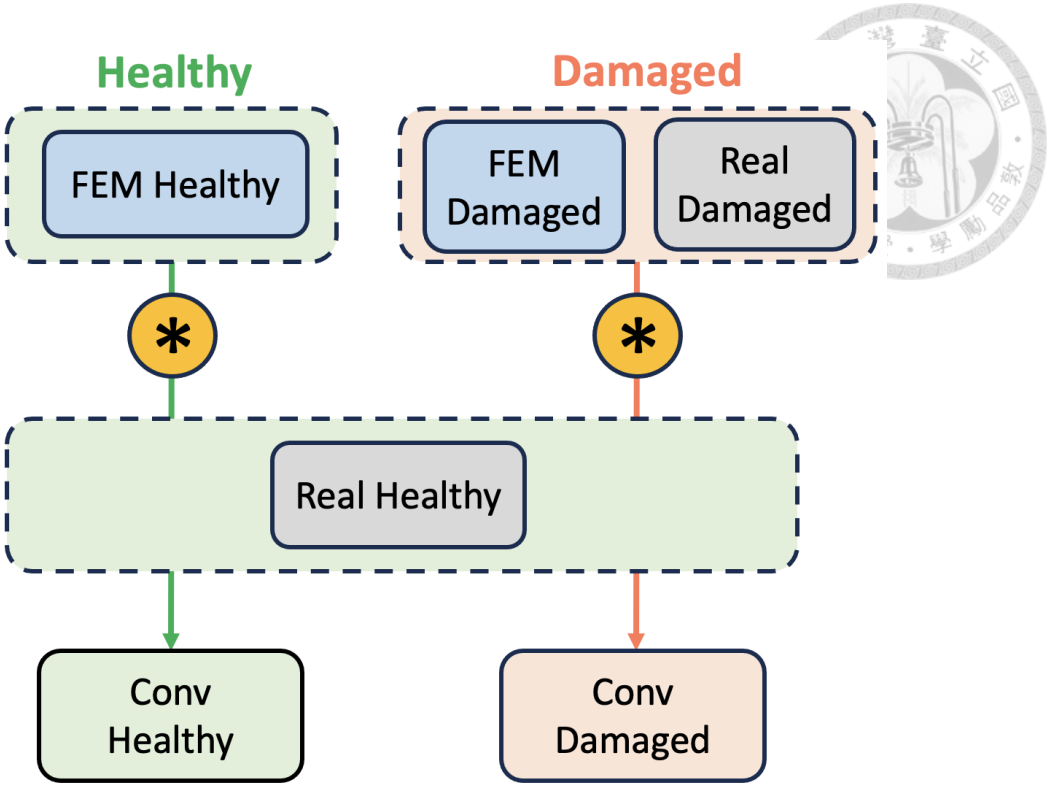


Figure 108. Convolutional Process

In order to prepare for the new set of data and the newly designed neural network, in order to prevent overfitting, the dataset needs to be expanded. But because of randomly convolutional process is inherently a data augmentation process, we can expand the dataset from the original thousands of data to a lot more combinations. The following table has listed out the dataset that are used to optimize the training of this implementation, and the training with less dataset along with different hidden layers are in the appendix section.

The training technique uses the same concept as the Health Optimized model, with 5% of the real-world damage injected in the training set. The goal is to let the model learn the features of the healthy model well thus generalize better towards different distributions of the data pool to increase domain adaption. The development and test set are both comprised of 50% healthy data and 50% of damaged data, and the damaged data is comprised of 25% from the FEM pool and 25% from the real-life pool.

doi:10.6342/NTU202401548

Table 13. Dataset for Convolutional Process

Train-Dev-Test split	**Healthy (50%)**	**Damaged (50%)**
Train (273600)	FHRH (50%)	FDRH (5%) RDRH (45%)
Dev (7200)	FHRH (50%)	FDRH (25%) RDRH (25%)
Test (7200)	FHRH (50%)	Real (25%) FEM (25%)

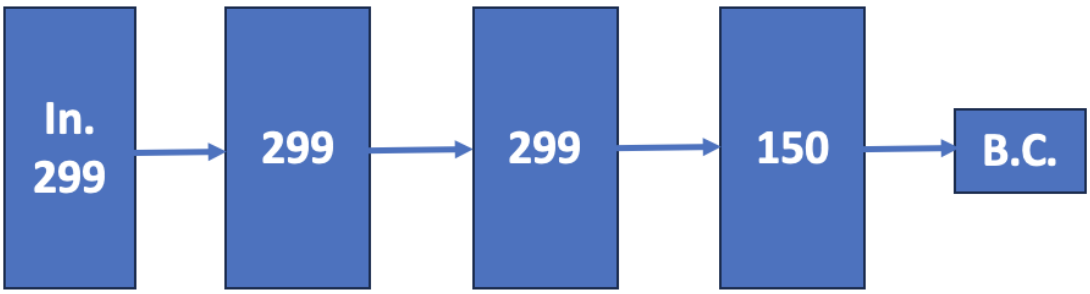


Figure 109. Proposed structure of Convolutional Health Optimized Model

Figure 109 showcases the proposed structure of the Convolutional Health Optimized Model that fits well of the new dataset domain. We have increased the layer counts and increased the input size to 299 units, as shown in the figure above. The first and second hidden also set to 299 units and the activation functions are set with ReLU functions. The last layer has 150 units and a dropout percentage of 90% is set in order to reduce overfitting.

doi:10.6342/NTU202401548

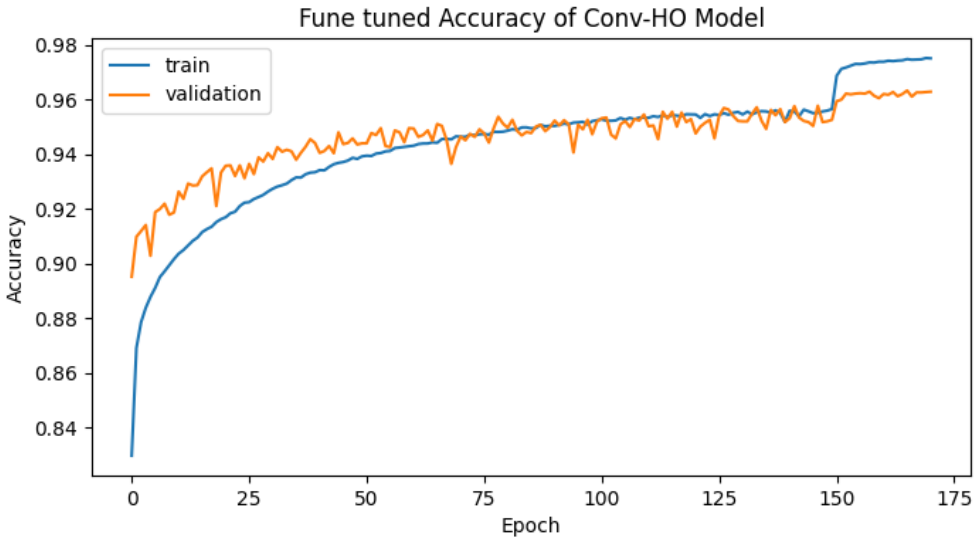


Figure 110. Fine Tuned Accuracy of Convolutional + Health Optimized Model

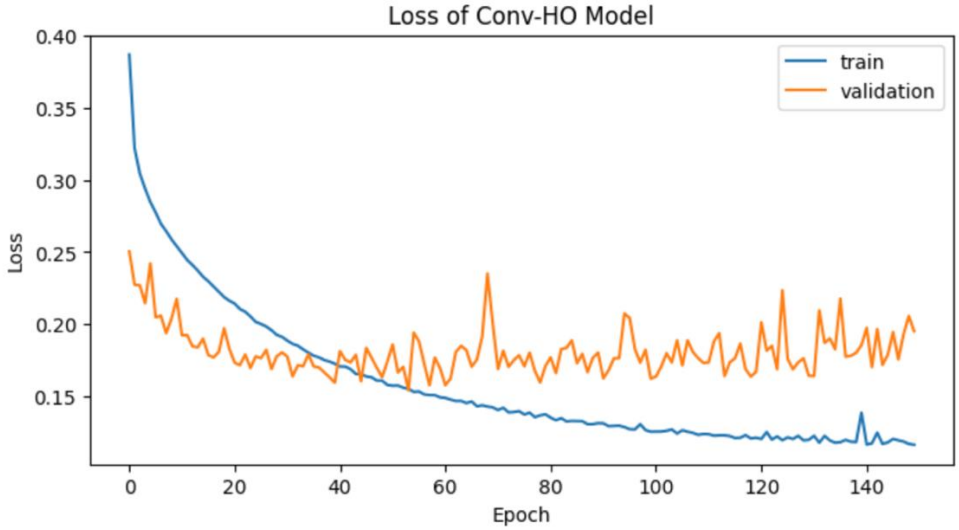


Figure 111. Loss of Convolutional + Health Optimized Model

The following is the accuracy plot of the Convolutional + Health Optimized Model. The training procedure uses traditional neural network optimization techniques. which is designed so that the model can perform good on both of the training set and the validation set. We stopped the training procedure when the development (validation) set loss starts to rise (Figure 111), which is at approximately 150 epochs. In Figure 110, we can see that after fine tuning, the training set has an accuracy of 97.5% and the development set has

doi:10.6342/NTU202401548

an accuracy of 96.28. We can see that with this procedure, we need a bigger network to fit which would require a larger dataset, and in turn requires more epochs to train but yields a more stable result and less overfitting.

```
Evaluate on test data
225/225 [==============================] - 0s 2ms/step - loss: 0.1878 - accuracy: 0.9614
test loss, test acc: [0.18775102496147156, 0.961388885974884]
```

Figure 112. Accuracy of Test Data of convolutional features

Inside the test data that has 3600 examples, the model fits the test set with an accuracy of 96.14%, with 188 misidentified cases in the false positives, which are the healthy data that are misidentified as damaged cases. There are 90 false negatives, which are damaged cases that are diagnosed as healthy cases. Inside 1800 damaged data, there are only 90 that are mis-diagnosed, which indicates that the model can fit the healthy and damaged data quite well with a high diagnosis rate.

In order to conduct error analysis of this model, we have prepared an eyeball test set. Because eye inspection is hard to conduct, we have arranged the eyeball test set with "known sequences", the first 250 are healthy data in the convolutional domain. From 251 to 375 are the data of FEM damaged convolved with real-life healthy. From 375 to 500 are the data of real-life damaged convolved with real-life healthy. Because we know the sequence of the data, we can now eyeball the distribution of the mis-labelled data. In order to further showcase the hybrid database technique, we have used consumer grade cameras to track certain damaged points in order to acquire the real-world damaged signal. The sample frequency of the cameras is 30 Hz, which is compatible with the database that we created.

doi:10.6342/NTU202401548

4.9 Real-Life Monitoring Task

4.9.1 Creating a hybrid database with more sources

We can expand the hybrid database so that it would be able to accept data types from multiple sources. In the previous section (2.1.2), we have spent some time listing visual methods that are able to trace the movements of a structure and extract vibrational data from the videos. This means that it is possible for visual data to integrated into our hybrid database.

The database is designed to contain spectral information of the structure at the sample frequency of 30 Hz. Most consumer cameras and mobile phones also have a 30 frame per second setting. This means that we can integrate visual data as real-life data just by transforming it into vibrational data. In this research we used the framework of optical flow and the codes created by Tsai [50] to obtain the visual vibration data from the structure.

There are more opportunities to conduct hybrid monitoring with this hybrid database and deep learning method. The deep neural network essentially is a classifier so that when new data is fed into the system, it would output the classified results with minimal computational time. We can obtain data from any source as long as it produces from acceleration, speed and displacement data where we could later transform them into spectrums to feed into the neural network.

4.9.2 Independent Monitoring with hybrid database

In order to further showcase the independence of damage from the Single Channel design, we split the eyeball-test into two equal channels, with 250 examples each and a total recording time of 2500 seconds. The first channel has 125 damaged data stringed together followed by 125 healthy data. The second channel has 125 healthy data

doi:10.6342/NTU202401548

sequenced together and followed by 125 damaged data. Therefore, the damage occurrence of the channels is completely independent and the network should be able to detect this implementation. The data are passed through the trained and the mis-identified cases are listed. For the damaged data, the visual inspection of the camera of the damaged data are sequenced as the last 15 examples. The full database is shown in Table 14 and Table 15.

Table 14. Test set Ch1

Class	**Contents**	**Index**
Healthy	FH-RH	1~125
Damaged	FD-RH	126~180
	RD-RH	181~235
	Visual RD-RH	235~250

Table 15. Test set Ch2

Class	**Contents**	**Index**
Damaged	FD-RH	1~55
	RD-RH	56~110
	Visual RD-RH	110~125
Healthy	FH-RH	125 ~ 250

The result of channel 1 is shown in Figure 114 and the misidentified indexes are shown in Figure 113. The result of channel 2 is shown in Figure 116 and the misidentified indexes are showed in Figure 115. We can see that we have a perfect score at detecting damage and that the independence of channels could be shown because of the damage detection results are pretty evident that the distribution of healthy and damaged are distinct. Although the dataset performs well on the damaged dataset and the visual data

doi:10.6342/NTU202401548

can also be integrated into the database well, the “healthy” database has most of the misidentifications. The reason might stem from by convolving all of the database with the real-life data, it makes the results lean much closer to the healthy state, which would increase the false positives.

```
8/8 [==============================] - 0s 2ms/step
[ 21  31  35  43  44  52  65  71  83  85  86  95  96 104]
```

Figure 113. Misidentified cases in channel 1

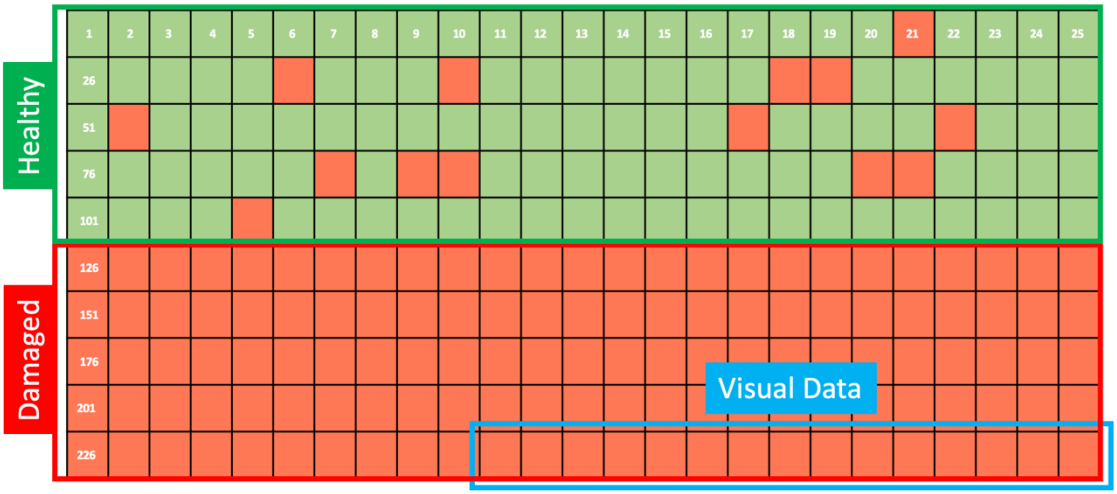


Figure 114. Channel 1 monitoring results

```
8/8 [==============================] - 0s 2ms/step
[126 133 145 149 150 152 160 175 176 177 184 192 205 208 215 218 220 222
 239]
```

Figure 115. Misidentified cases of channel 2

doi:10.6342/NTU202401548

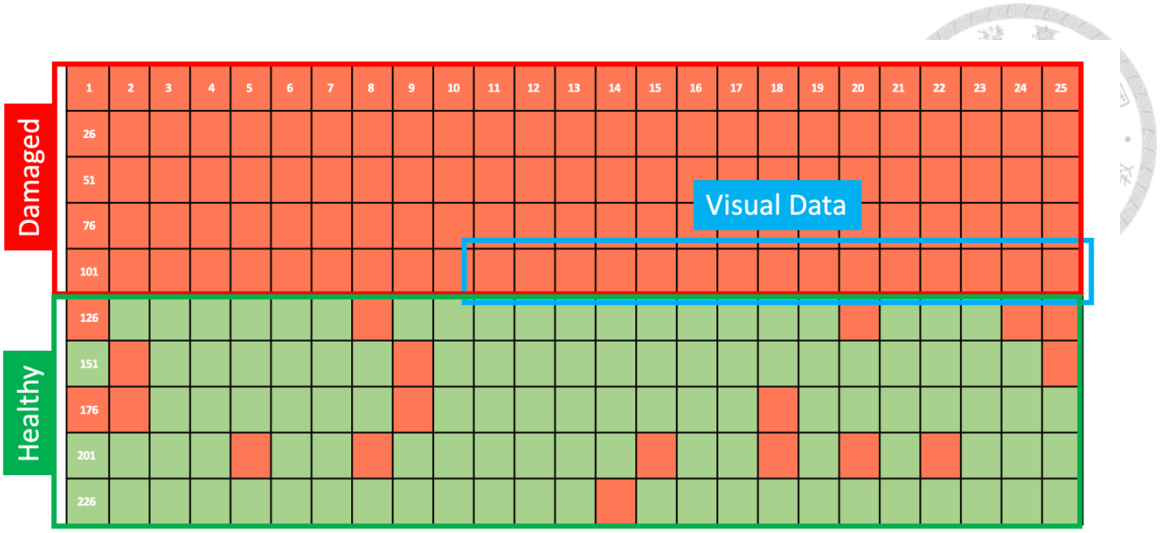


Figure 116. Channel 2 monitoring results

We can see that the damage distribution of the two channels is identified clearly. In channel 1 where the damage distribution is in the last 1/2 of the data are identified in red clearly. In channel 2 where the damage distribution is in the first 1/2 of the whole series are identified accurately in red. There are more confusions in the healthy set of data the damaged data are all identified clearly but all of the damaged data are identified. With this method, although there are more confusions in the healthy state, consecutive damaged data can be used as a clear identification toward the damaged data and this will increase a confidence of us identifying the damaged state of a structure.

doi:10.6342/NTU202401548

Chapter 5 Conclusions and Future Work

5.1 Conclusion

During this research we have first found out that the DNN approach can fit and map a wide range of data effectively with a high detection rate of 96% in the frequency domain, which makes it ideal if we have a complex set of states that we need to distinguish. Also, this research has made an effort to increase the similarity of the simulated FEM database and the real-life data, so that we can supplement damage cases accurately with the network. We have found out that using a convolution approach can increase similarity between FEM and real-life data by measuring the cosine similarity, which bumped up the correlation factor from 0.4 to 0.7.

In order to deal with limited damaged data, we have used two approaches, firstly we tackled the problem by optimizing on the healthy database and supplement the training with FEM data and have found out that the Health Optimized model generalizes well to different distributions of data. In the final experiment we used the convolutional method, which could act as a data augmentation technique to acquire more data in the magnitude of thousands, the convolutional model with slight modifications to the neural network could work well with the new domain of the database, and we can mix data from different sources well, including visual, accelerometer, and computer simulated results.

5.2 Future Work

In this research we have covered the techniques and results of Deep Learning Applied SHM, and the results have been promising for further investigation for this method because of the potential convenience and knowledge that it can aid engineers in the future.

doi:10.6342/NTU202401548

Firstly, there are hundreds of wind turbines in an offshore wind farm and most of them don't have the same boundary conditions. We could potentially study some turbines with representative characteristics in a vast offshore wind farm, such as acquiring data from OWTs that are at the forefront of the wind and current, the middle of the farm, and the farthest from the impact of the wind and current. We could also study wind turbines that have scouring effects in the bottom, which could potentially be very beneficial at determining healthy and unhealthy states because we can integrate the knowledge into the FEM models and increase the variability of the database and accuracy of the FEM model. The blades and rotors are also important components of the OWT because they bear the blunt force of wind and rain and generate the kinetic energy into electricity. They are very suspectable for failure and currently there isn't a lot of research done in this section. We could possibility integrate our system into the blades as they are similar to the tower structure that we have inspected and possibly easier to perform. Because of the nature of the simple single channel design, we can place the data acquisition modules in modal points that are very suspectable for damage. Vibration is one of the most prominent signals generated by the gearboxes. We have currently found out that by doing spectral analysis on a neural network we can fit the complex solution plane with multiple types of data. This means that if there are noise present, as it should be in gear boxes, the neural network might be a good solution as it might be able to would bypass the noise frequencies and obtain the natural frequencies of the gear boxes. By acquiring more knowledge in the healthy states from more component of OWTs, we can have a better understanding and estimate of damaged states.

Secondly, the transferability of the knowledge of wind turbines could be a great work for the future. Across wind farms, there would be multiple different types of wind turbines

doi:10.6342/NTU202401548

that uses different types of components. Even the scale of the wind turbines could be different. In order for our proposed method to be more valuable for future implements, it would be exciting for us to explore the option of transferring data from one type of wind turbine to another. Transfer learning could be used to accomplish this task but there are some problems that could arise, which is discussed in the research. We could also use the convolutional mindset to link the two databases from different types of wind turbines together, and maybe using another neural network to optimize the process. There are researches that uses CNNs to detect the "style" of an image and transfer the style into another image by a technique called Neural Style Transfer. We could also use this method to extract prominent features of a spectrum of a certain type of structure and transfer it to another type of structure. The transferability could be an exciting method to implement because we can now apply this technique into house structures, which are all built slightly different from one another.

Finally, this method is not limited to be used on wind turbines. It is a tool that could be used in all types of civil structures that mainly use vibration as a tool for damage identification. We could potentially first implement this tool on bridges and other public resources and apply it to houses when we acquire more data and characteristics of damaged states as we grow our database and knowledge. This technology could potentially be widely used for safety inspection after earthquakes, typhoons and other natural disasters and engineers would be able to come up with repair and maintenance plans swiftly to prevent major disasters from happening.

doi:10.6342/NTU202401548

Appendix

Appendix A

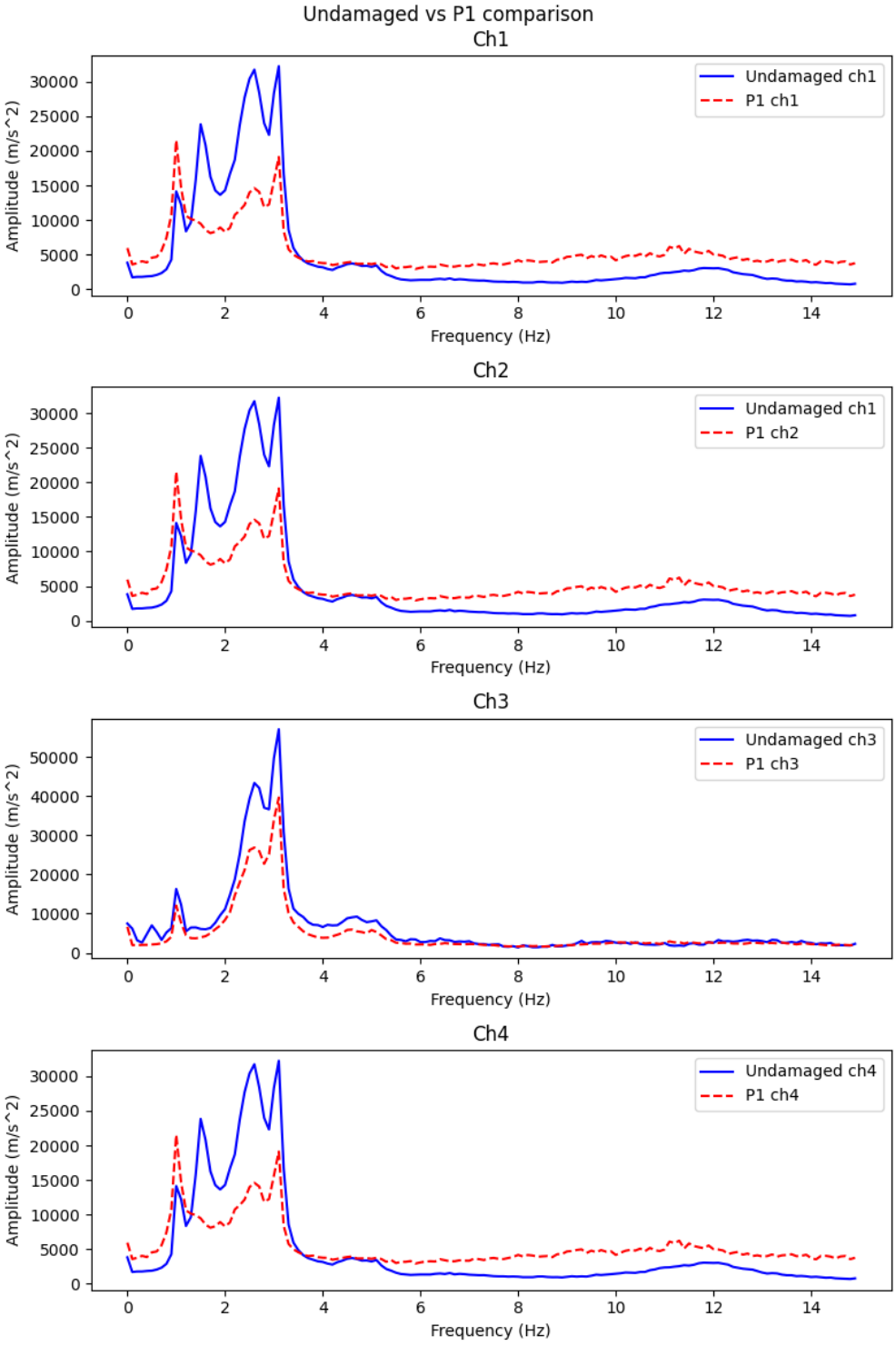


Figure 117. FEM Damaged vs Undamaged, P1

doi:10.6342/NTU202401548

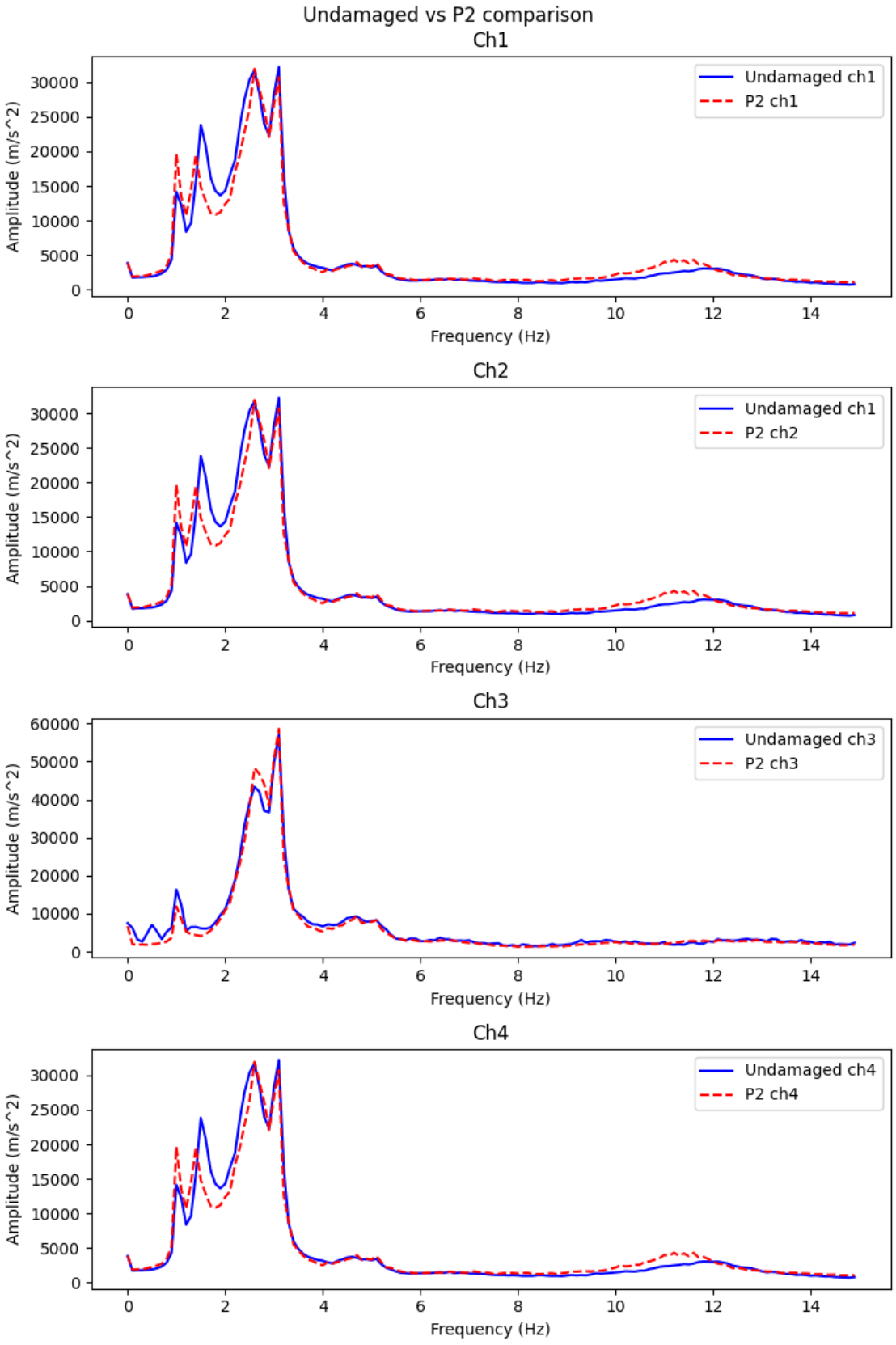


Figure 118. FEM Damaged vs Undamaged, P2

doi:10.6342/NTU202401548

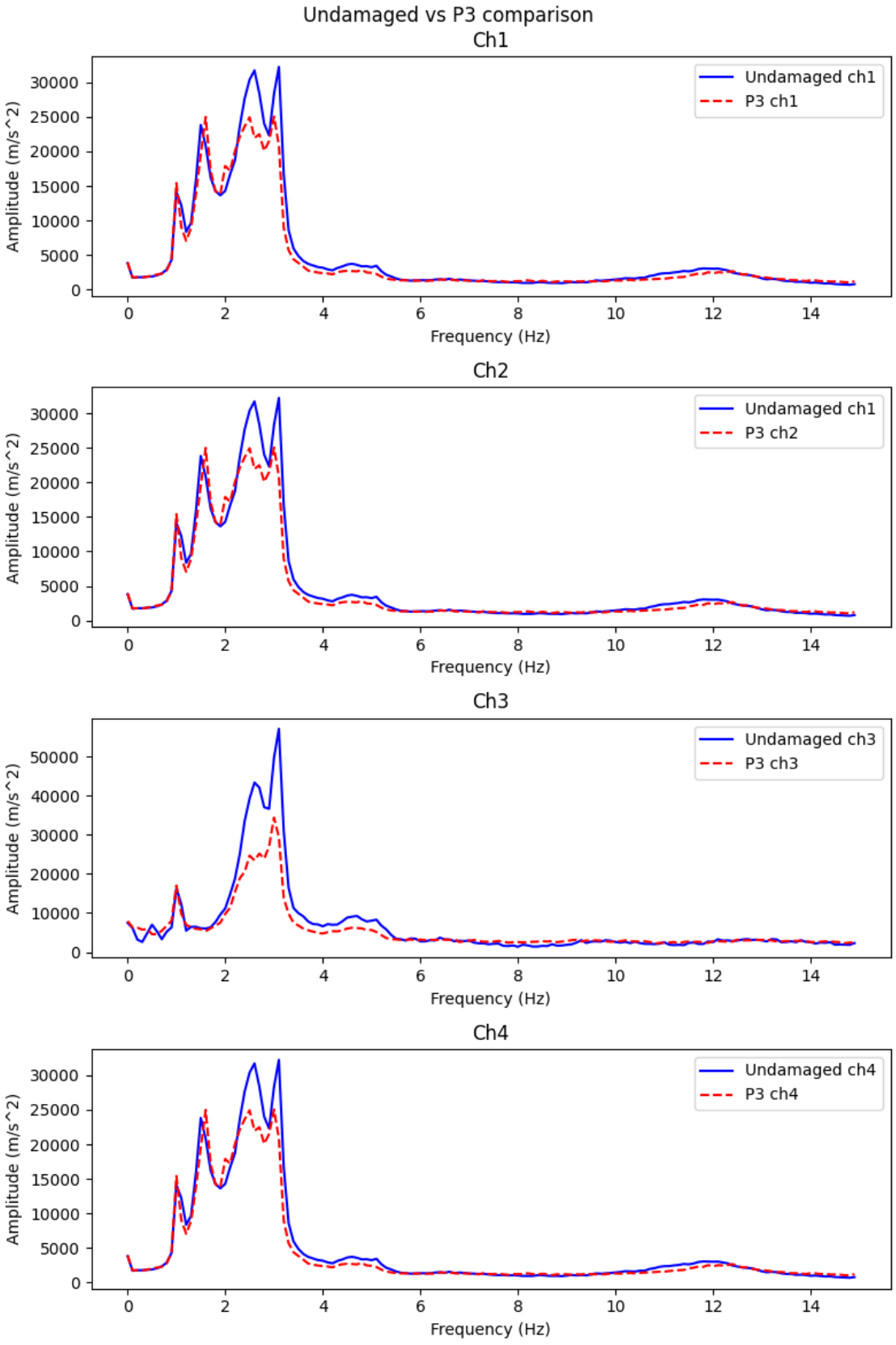


Figure 119. FEM Damaged vs Undamaged, P3

doi:10.6342/NTU202401548

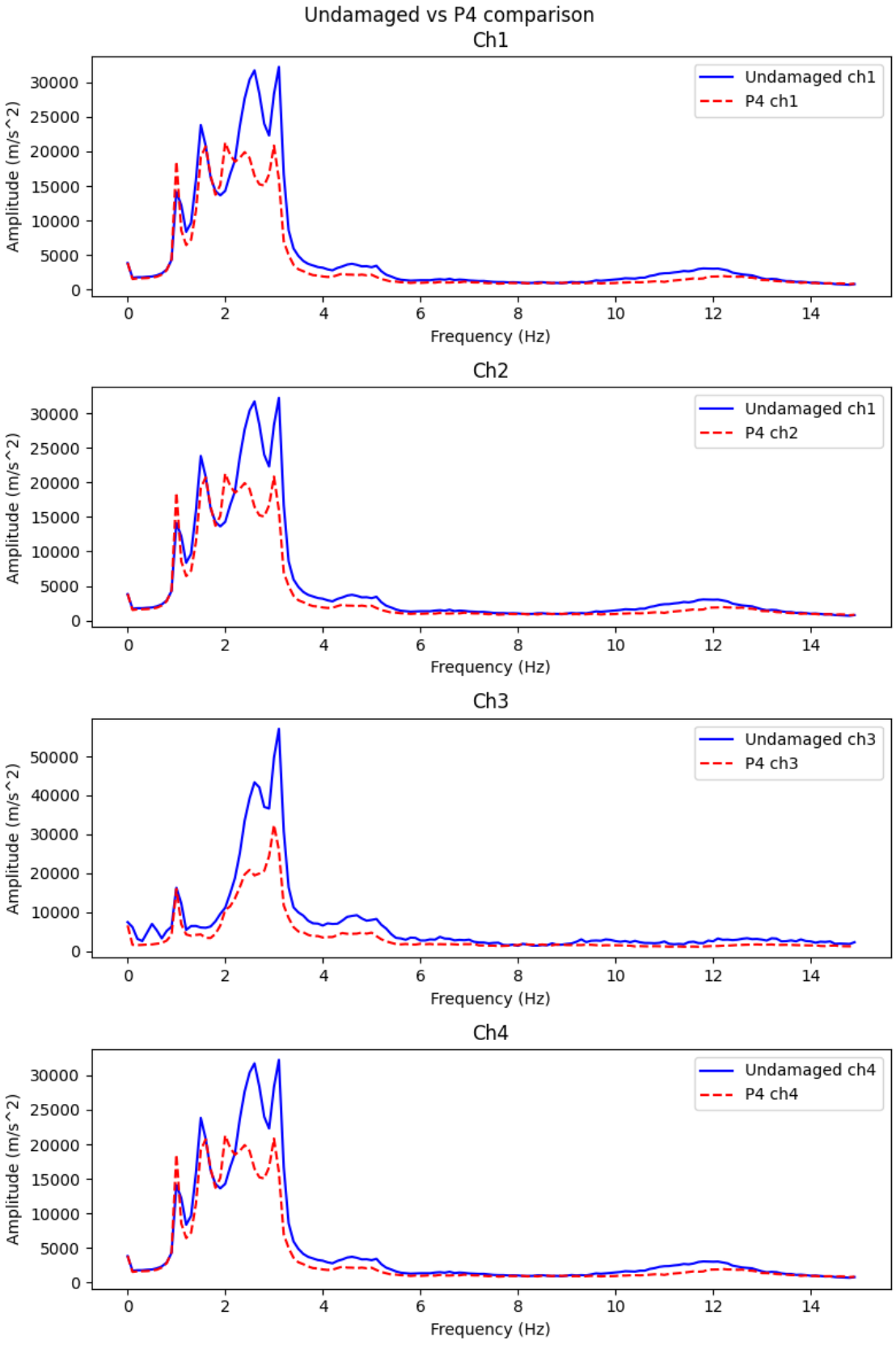


Figure 120. FEM Damaged vs Undamaged, P4

doi:10.6342/NTU202401548

References

[1] A. Zaoui, R. Meziane, E. Chatelet, and F. Lakdja, "Economic evaluation of the life cycle of a wind farm and improving the levelized cost of energy in region Champagne-Ardenne, France," *International Journal of Energy and Environmental Engineering,* pp. 1-11, 2022.

[2] M. Hofmann and I. B. Sperstad, "Will 10 MW wind turbines bring down the operation and maintenance cost of offshore wind farms?," *Energy Procedia,* vol. 53, pp. 231-238, 2014.

[3] Z. Jiang, "Installation of offshore wind turbines: A technical review," *Renewable and Sustainable Energy Reviews,* vol. 139, p. 110576, 2021.

[4] T. Stehly, P. Beiter, and P. Duffy, "2019 cost of wind energy review," National Renewable Energy Lab.(NREL), Golden, CO (United States), 2020.

[5] E. Ozer, D. Feng, and M. Q. Feng, "Hybrid motion sensing and experimental modal analysis using collocated smartphone camera and accelerometers," *Measurement Science and Technology,* vol. 28, no. 10, p. 105903, 2017.

[6] I. Dinwoodie, O.-E. V. Endrerud, M. Hofmann, R. Martin, and I. B. Sperstad, "Reference cases for verification of operation and maintenance simulation models for offshore wind farms," *Wind Engineering,* vol. 39, no. 1, pp. 1-14, 2015.

[7] X.-g. Zhao and L.-z. Ren, "Focus on the development of offshore wind power in China: Has the golden period come?," *Renewable Energy,* vol. 81, pp. 644-657, 2015.

[8] S. Faulstich, B. Hahn, and P. J. Tavner, "Wind turbine downtime and its importance for offshore deployment," *Wind Energy,* vol. 14, no. 3, pp. 327-337, 2011.

[9] Z. Ren, A. S. Verma, Y. Li, J. J. Teuwen, and Z. Jiang, "Offshore wind turbine operations and maintenance: A state-of-the-art review," *Renewable and Sustainable Energy Reviews,* vol. 144, p. 110886, 2021.

doi:10.6342/NTU202401548

[10] A. Karyotakis and R. Bucknall, "Planned intervention as a maintenance and repair strategy for offshore wind turbines," *Journal of marine engineering & technology*, vol. 9, no. 1, pp. 27-35, 2010.

[11] K. Sivalingam, M. Sepulveda, M. Spring, and P. Davies, "A review and methodology development for remaining useful life prediction of offshore fixed and floating wind turbine power converter with digital twin technology perspective," in *2018 2nd international conference on green energy and applications (ICGEA)*, 2018: IEEE, pp. 197-204.

[12] K. Worden, C. R. Farrar, G. Manson, and G. Park, "The fundamental axioms of structural health monitoring," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 463, no. 2082, pp. 1639-1664, 2007.

[13] H. Sohn and C. R. Farrar, "Damage diagnosis using time series analysis of vibration signals," *Smart materials and structures*, vol. 10, no. 3, p. 446, 2001.

[14] P. Moradipour, T. H. Chan, and C. Gallage, "Benchmark studies for bridge health monitoring using an improved modal strain energy method," *Procedia Engineering*, vol. 188, pp. 194-200, 2017.

[15] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods," *International journal of computer vision*, vol. 61, pp. 211-231, 2005.

[16] S. Yoneyama and H. Ueda, "Bridge deflection measurement using digital image correlation with camera movement correction," *Materials transactions*, vol. 53, no. 2, pp. 285-290, 2012.

[17] A. Khadka, B. Fick, A. Afshar, M. Tavakoli, and J. Baqersad, "Non-contact vibration monitoring of rotating wind turbines using a semi-autonomous UAV," *Mechanical Systems and Signal Processing*, vol. 138, p. 106446, 2020.

doi:10.6342/NTU202401548

[18] J. Won, J.-W. Park, K. Park, H. Yoon, and D.-S. Moon, "Non-target structural displacement measurement using reference frame-based deepflow," *Sensors*, vol. 19, no. 13, p. 2992, 2019.

[19] H. Yoon, H. Elanwar, H. Choi, M. Golparvar-Fard, and B. F. Spencer Jr, "Target-free approach for vision-based structural system identification using consumer-grade cameras," *Structural Control and Health Monitoring*, vol. 23, no. 12, pp. 1405-1416, 2016.

[20] M. R. Taha, A. Noureldin, J. L. Lucero, and T. J. Baca, "Wavelet transform for structural health monitoring: a compendium of uses and features," *Structural health monitoring*, vol. 5, no. 3, pp. 267-295, 2006.

[21] K. Gurley and A. Kareem, "Applications of wavelet transforms in earthquake, wind and ocean engineering," *Engineering structures*, vol. 21, no. 2, pp. 149-167, 1999.

[22] Y. Xin, H. Hao, and J. Li, "Operational modal identification of structures based on improved empirical wavelet transform," *Structural Control and Health Monitoring*, vol. 26, no. 3, p. e2323, 2019.

[23] C. A. Perez-Ramirez, J. P. Amezquita-Sanchez, H. Adeli, M. Valtierra-Rodriguez, R. d. J. Romero-Troncoso, A. Dominguez-Gonzalez, and R. A. Osornio-Rios, "Time-frequency techniques for modal parameters identification of civil structures from acquired dynamic signals," *Journal of Vibroengineering*, vol. 18, no. 5, pp. 3164-3185, 2016.

[24] X. Dong, J. Lian, H. Wang, T. Yu, and Y. Zhao, "Structural vibration monitoring and operational modal analysis of offshore wind turbine structure," *Ocean Engineering*, vol. 150, pp. 280-297, 2018.

[25] P. Singh and A. Sadhu, "System identification-enhanced visualization tool for infrastructure monitoring and maintenance," *Frontiers in Built Environment*, vol. 6, p. 76, 2020.

doi:10.6342/NTU202401548

[26] Q. Pu, Y. Hong, L. Chen, S. Yang, and X. Xu, "Model updating–based damage detection of a concrete beam utilizing experimental damped frequency response functions," *Advances in Structural Engineering*, vol. 22, no. 4, pp. 935-947, 2019.

[27] V. Srinivas, K. Ramanjaneyulu, and C. A. Jeyasehar, "Multi-stage approach for structural damage identification using modal strain energy and evolutionary optimization techniques," *Structural Health Monitoring*, vol. 10, no. 2, pp. 219-230, 2011.

[28] Q. Mao, M. Mazzotti, J. DeVitis, J. Braley, C. Young, K. Sjoblom, E. Aktan, F. Moon, and I. Bartoli, "Structural condition assessment of a bridge pier: A case study using experimental modal analysis and finite element model updating," *Structural Control and Health Monitoring*, vol. 26, no. 1, p. e2273, 2019.

[29] E. Schulz, M. Speekenbrink, and A. Krause, "A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions," *Journal of Mathematical Psychology*, vol. 85, pp. 1-16, 2018.

[30] W. Zhang, G. Peng, C. Li, Y. Chen, and Z. Zhang, "A new deep learning model for fault diagnosis with good anti-noise and domain adaptation ability on raw vibration signals," *Sensors*, vol. 17, no. 2, p. 425, 2017.

[31] Y. Bao, Z. Tang, H. Li, and Y. Zhang, "Computer vision and deep learning–based data anomaly detection method for structural health monitoring," *Structural Health Monitoring*, vol. 18, no. 2, pp. 401-421, 2019.

[32] O. Avci, O. Abdeljaber, S. Kiranyaz, M. Hussein, and D. J. Inman, "Wireless and real-time structural damage detection: A novel decentralized method for wireless sensor networks," *Journal of Sound and Vibration*, vol. 424, pp. 158-172, 2018.

[33] M. Chamangard, G. Ghodrati Amiri, E. Darvishan, and Z. Rastin, "Transfer learning for CNN-based damage detection in civil structures with insufficient data," *Shock and Vibration*, vol. 2022.

doi:10.6342/NTU202401548

[34] E. M. Tronci, H. Beigi, R. Betti, and M. Q. Feng, "A damage assessment methodology for structural systems using transfer learning from the audio domain," *Mechanical Systems and Signal Processing*, vol. 195, p. 110286, 2023.

[35] Y. Narazaki, V. Hoskere, T. A. Hoang, and B. F. Spencer Jr, "Automated vision-based bridge component extraction using multiscale convolutional neural networks," *arXiv preprint arXiv:1805.06042*, 2018.

[36] 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," *Construction and building materials*, vol. 157, pp. 322-330, 2017.

[37] X. Yang, H. Li, Y. Yu, X. Luo, T. Huang, and X. Yang, "Automatic pixel-level crack detection and measurement using fully convolutional network," *Computer-Aided Civil and Infrastructure Engineering*, vol. 33, no. 12, pp. 1090-1109, 2018.

[38] N. S. Gulgec, M. Takáč, and S. N. Pakzad, "Structural damage detection using convolutional neural networks," in *Model Validation and Uncertainty Quantification, Volume 3: Proceedings of the 35th IMAC, A Conference and Exposition on Structural Dynamics 2017*, 2017: Springer, pp. 331-337.

[39] C. Feng, H. Zhang, S. Wang, Y. Li, H. Wang, and F. Yan, "Structural damage detection using deep convolutional neural network and transfer learning," *KSCE Journal of Civil Engineering*, vol. 23, pp. 4493-4502, 2019.

[40] C. Lu, Z. Wang, and B. Zhou, "Intelligent fault diagnosis of rolling bearing using hierarchical convolutional network based health state classification," *Advanced Engineering Informatics*, vol. 32, pp. 139-151, 2017.

[41] N. S. Gulgec, M. Takáč, and S. N. Pakzad, "Convolutional neural network approach for robust structural damage detection and localization," *Journal of computing in civil engineering*, vol. 33, no. 3, p. 04019005, 2019.

doi:10.6342/NTU202401548

[42] H. Kim and S. H. Sim, "Automated peak picking using region-based convolutional neural network for operational modal analysis," *Structural Control and Health Monitoring*, vol. 26, no. 11, p. e2436, 2019.

[43] W. Zhang, G. Peng, and C. Li, "Bearings fault diagnosis based on convolutional neural networks with 2-D representation of vibration signals as input," in *MATEC web of conferences*, 2017, vol. 95: EDP Sciences, p. 13001.

[44] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998.

[45] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[46] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1-48, 2019.

[47] A. Le Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time series classification using convolutional neural networks," in *ECML/PKDD workshop on advanced analytics and learning on temporal data*, 2016.

[48] T. Wen and R. Keyes, "Time series anomaly detection using convolutional neural networks and transfer learning," *arXiv preprint arXiv:1905.13628*, 2019.

[49] Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu, "Time series data augmentation for deep learning: A survey," *arXiv preprint arXiv:2002.12478*, 2020.

[50] Wei-Han Cheng, Cheng-En Tsai, Hsin-Haou Huang, "Vision-based algorithm for structural response measurement using movable camera and damage localization" Measurement, Volume 232, 2024.

doi:10.6342/NTU202401548