2024

Deep learning approach for robust structural health monitoring using guided Lamb wave responses

Sameer Gopali

DEEP LEARNING APPROACH FOR ROBUST STRUCTURAL HEALTH MONITORING USING GUIDED LAMB WAVE RESPONSES

Sameer Gopali

A THESIS

**Submitted in partial fulfillment of the requirements
for the degree of Master of Science
in
The Department of Computer Science
to
The Graduate School
of
The University of Alabama in Huntsville**

May 2024

Approved by:

Dr. Huaming Zhang, Research Advisor/Committee Chair
Dr. Gang Wang, Committee Member
Dr. Deepak Acharya, Committee Member
Dr. Letha Etzkorn, Department Chair
Dr. Rainer Steinwandt, College Dean
Dr. Jon Hakkila, Graduate Dean

Abstract

DEEP LEARNING APPROACH FOR ROBUST STRUCTURAL HEALTH MONITORING USING GUIDED LAMB WAVE RESPONSES

Sameer Gopali

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science

Computer Science

The University of Alabama in Huntsville

May 2024

Guided Lamb waves offer a promising solution for the early detection of internal damages in structures due to their high sensitivity to small damages. However, noise can adversely impact the development of data-driven models for damage detection, leading to inaccurate monitoring systems. This thesis explores deep learning techniques to robustly predict the location and severity of damage in cantilevered beams using noisy guided wave responses. Initially, Multi-Layer Perceptron (MLP) is trained with frequency domain features to achieve robust performance against noisy data. Further performance improvement is achieved using end-to-end learning models, which include autoencoder and one-dimensional Convolutional Neural Network (CNN). The autoencoder demonstrates better dimensionality reduction compared to frequency-based feature extraction while also exhibiting better performance. The one-dimensional CNN model outperforms other techniques, achieving an R^2 score of 0.9908 in the highest noise level settings. These results facilitate the development of robust structural health monitoring using deep learning techniques.

Acknowledgements

I want to express my deepest appreciation to my advisor, Dr. Huaming Zhang, for his unwavering guidance, expertise, and support throughout this research journey. His insightful feedback, encouragement, and patience helped me navigate challenges and refine this work.

I am also grateful to Dr. Gang Wang, who served as a member of my thesis committee. His generous support and assistance, including access to research materials, data, and domain expertise, were invaluable to the completion of this research.

I would also like to thank Dr. Deepak Acharya, another member of my thesis committee, for his insightful feedback and constructive critiques. His valuable input greatly enriched the overall quality of my research.

Finally, I am deeply indebted to my incredible friends and family for their unwavering support and understanding throughout this process. Their constant encouragement, patience, and belief in me fueled my motivation and helped me persevere through challenging moments.

Table of Contents

Abstract . . . . **ii**

Acknowledgements . . . . **iv**

Table of Contents . . . . **vii**

List of Figures . . . . **viii**

List of Tables . . . . **ix**

List of Abbreviations . . . . **x**

Chapter 1. Introduction **1**

1.1 Structural Health Monitoring 1

1.2 Guided Lamb Wave 3

1.3 Research Problem 4

1.3.1 Feature Selection 5

1.3.2 End-to-End Deep Learning Approach 5

1.3.3 Comparative Analysis 6

1.4 Approach 6

1.5 Thesis Organization 7

Chapter 2. Background and Related Works 9

2.1 Feature Selection Techniques 9

2.2 Deep Learning Methods 11

2.2.1 Multi-Layer Perceptron 11

2.2.2 Long-Short Term Memory 14

2.2.3 Convolutional Neural Network 15

2.2.4 Autoencoder 21

Chapter 3. Data Preparation 24

Chapter 4. Methodologies 29

4.1 Multi-layer Perceptron 29

4.2 Autoencoder 31

4.3 1D CNN 34

4.4 Model Selection 35

4.4.1 Training Hyperparameters 35

4.4.2 Regularization 36

4.4.3 R^2 score 36

Chapter 5. Evaluations and Results 38

5.1 Numerical Test 1: Evaluating Performance of Feature Selection Method 38

5.2 Numerical Test 2: Evaluating End-to-End Deep Learning Models 39

5.3 Discussions 41

Chapter 6. Conclusion and Future Work 47

6.1 Conclusion 47

6.2 Future Work 48

References 50

Appendix A. Training LSTM 54

List of Figures

1.1 A typical components of SHM 2

1.2 Simulation of Lamb waves 4

2.1 Typical architecture of MLP 13

2.2 Architecture of CNN 17

2.3 1D Convolutional Neural Network 20

2.4 Architecture of Autoencoder 22

3.1 A cantilevered beam system 24

3.2 Waveform of tip force 25

3.3 Time-domain response: undamaged case vs damaged case 26

3.4 Noise free wave response 27

3.5 Wave response with 5% added noise level 27

3.6 Wave response with 10% added noise level 28

3.7 Wave response with 15% added noise level 28

4.1 Frequency Spectrum of Wave Response 31

4.2 Flowchart of proposed FFT based MLP model 32

4.3 Proposed Architecture of Autoencoder Network 33

5.1 Noisy and reconstructed wave response using FFT features . . . 43

5.2 Noisy and reconstructed wave response using autoencoder 44

5.3 Predicted vs Actual values for MLP at 15% noise level 45

5.4 Predicted vs Actual values for Autoencoder at 15% noise level . . 45

5.5 Predicted vs Actual values for 1D CNN at 15% noise level 46

List of Tables

4.1 Convolutional and Max Pool layer configuration of 1D CNN . . . 34
4.2 Training Hyperparameters 36
5.1 Performance Comparision for FFT-MLP model and MLP model . 39
5.2 Performance of 1D CNN and autoencoder under various noise level 40
5.3 Examples of actual and predicted values at 15% noise level 42

List of Abbreviations

Abbreviation	Description
1D	1 dimensional
2D	2 dimensional
CNN	Convolutional Neural Network
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
KHz	Kilo Hertz
LDA	Linear Discriminant Analysis
LSTM	Long Short Term Memory
MLP	Multi Layer Perceptron
PCA	Principal Component Analysis
POD	Proper Orthogonal Decomposition
RNN	Recurrent Neural Network

Abbreviation	Description
SFEM	Spectral Finite Element Method
SHAP	SHapely Additive exPlanations
SHM	Structural Health Monitoring

Chapter 1. Introduction

1.1 Structural Health Monitoring

The performance of different civil and mechanical infrastructures can degrade due to several factors like usage, aging, and environmental conditions which can lead to catastrophic failures causing great economic costs and loss of human life. Therefore, it is highly important to monitor the performance to ensure the reliability and safety of infrastructure. SHM is the process of monitoring and assessing the conditions of the structures. The main objective is to identify any potential damage to the structure. The process starts by placing sensors on infrastructures. The data are then transmitted and collected into storage databases. Various techniques and algorithms are then applied to assess the current structural health. After that, experts make decisions related to inspection and maintenance based on the severity, location, and predicted future propagation of identified damage [22].

There are two popular approaches in SHM: the model-driven approach and the data-driven approach [10]. Model-based SHM uses mathematical models to predict and assess the structural health of the infrastructure. This method creates a mathematical model of the structure's dynamics and response which is compared with the collected data to detect any damages or anomalies. However,

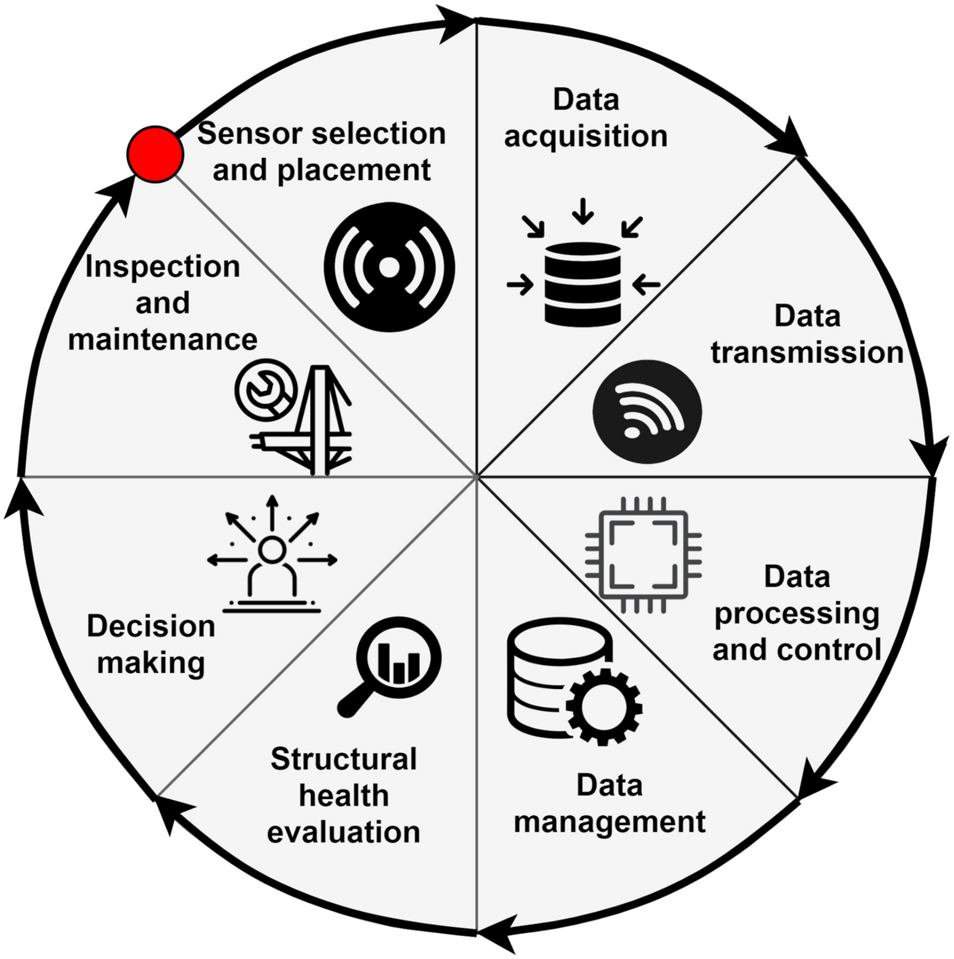
Sensor selection and placement
Data acquisition
Data transmission
Data processing and control
Data management
Structural health evaluation
Decision making
Inspection and maintenance

Figure 1.1: A typical components of SHM [22].

constructing such models can be difficult due to the complexity of their geometry, material properties, and environmental conditions. Moreover, uncertainties in these parameters could affect the accuracy of such models.

In contrast, data-driven approaches do not depend on such mathematical models and only use the collected data to identify any damages. It uses machine learning algorithms, statistical analysis, and data mining techniques to discover patterns from the data directly. Due to the advancements in big data technologies and machine learning, the capability of data-driven methods to uncover patterns from the vast amount of data has significantly increased. This has led to an increase in the popularity of data-driven SHM. Data-driven techniques may involve analyzing either time series-based vibration data or image-based data [35]. While image data are limited to revealing external damage and can be affected by environmental conditions, vibration-based signal data can also identify problems within the internal state of the structure.

Vibration-based SHM utilizes modal information, electromechanical impedance, acoustic emissions, and ultrasonic wave-based methods such as Lamb waves [6]. Among these, Lamb waves are particularly sensitive to small damages and are capable of traveling long distances with only little energy loss [11]. As a result, they can they can cover large areas. Thus, they are the primary focus of our study.

1.2 Guided Lamb Wave

Guided Lamb waves, named after Horace Lamb, are a type of ultrasonic waves that are guided between two parallel free surfaces, such as the upper and lower surfaces of a plate [11]. Lamb waves can exist in two fundamental modes: symmetric S_0 and antisymmetric A_0 modes. In the symmetric modes, the oscillations on the structure surface are in phase while in the antisymmetric modes, the motion is out of phase [28]. Lamb waves can travel through structures in different patterns by utilizing these different mode configurations.

In guided Lamb wave-based SHM, waves are propagated through the structure. The structure's response to these waves when it interacts with any irregularities or inconsistencies is then used for the detection of any structural anomalies or defects.

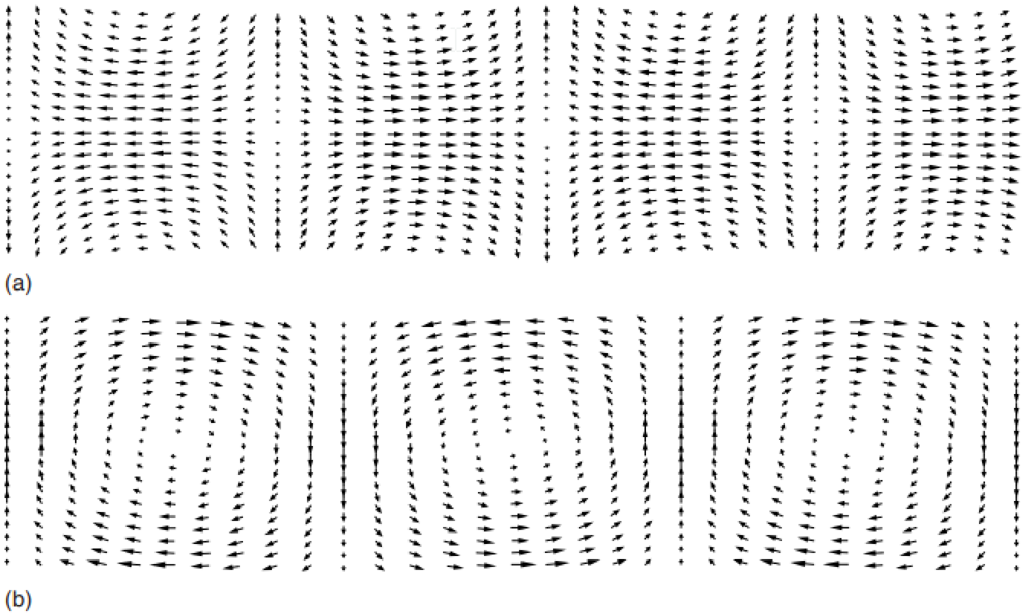

Figure 1.2: Simulation of Lamb waves (a) symmetric S_0 mode; (b) antisymmetric A_0 mode [11].

1.3 Research Problem

The thesis deals with the application of guided Lamb waves to the SHM of cantilevered beams subjected to a tip force. The aim is to localize and predict the severity of damage. Lamb waves are very sensitive to small damages therefore they are suitable for early detection and characterization. However, the data are complex as they capture interactions between multiple propagation modes, reflections from boundaries, and potential defects within the structure. Thus it is difficult to identify clear signatures that accurately localize damage and its extent. Furthermore, a large number of sample points is needed to capture wave behavior over time. This makes the data high dimensional for analysis using traditional machine-learning methods. So, feature engineering is needed, but this requires domain knowledge. Moreover, the chosen feature could be affected by noise and it might not generalize well to different environmental conditions. It can result in unreliable predictions and make SHM systems inefficient.

So, to build a predictive model that can overcome these challenges and identify damage location as well as predict its extent, this thesis explores two strategies:

1.3.1 Feature Selection

Initially, we intend to explore feature selection methods that can provide meaningful representation from Lamb wave responses to train deep learning models. Our focus will be on techniques that can extract useful information from the data even in the presence of noise, ensuring reliable features for real-world data.

1.3.2 End-to-End Deep Learning Approach

Secondly, we aim to utilize deep learning models that will directly process raw data without using handcrafted features. As deep learning models can directly learn complex patterns from the data, they offer the potential for more accurate and efficient models. We will explore various deep learning architectures for creating accurate end-to-end systems even with noisy data.

1.3.3 Comparative Analysis

To gain a comprehensive understanding, we will compare the performance of both approaches: deep learning models trained directly on raw data and models utilizing extracted features. This comparison will be conducted under varying noise levels to assess the impact of noise on each approach's effectiveness. We will employ various performance metrics to evaluate the models' ability to accurately localize and predict damage severity.

By investigating these two methodologies and conducting a thorough comparison, this thesis aims to develop robust and accurate solutions for detecting damages using Lamb wave responses. This will ultimately contribute to more reliable and effective SHM systems, ensuring the safety of infrastructure.

1.4 Approach

To address the research challenges, we begin by utilizing frequency domain transformations for feature engineering and dimensionality reduction. We aim to extract informative features that will filter the noise in the signal by selecting features of specific frequency bands from the frequency domain. These extracted features will then serve as input for training MLP.

Additionally, we will employ the following models to explore the different end-to-end deep learning models:

- **1D Convolutional Neural Network:** This architecture specializes in learning spatial features directly from sequential data which makes it suitable for analyzing Lamb wave responses.

- **Autoencoder:** This neural network is designed to learn to compress representation and reconstruct input data. This allows it to learn important features from the data, which in turn automates the feature extraction process. Additionally, this capability helps to reduce noise and data dimensionality.

These models will be trained on datasets with different noise levels to simulate real-world sensor data. The models' performances will be evaluated using metrics such as R^2 and percentage error.

1.5 Thesis Organization

The thesis is divided into six chapters, each exploring different parts of the research problem. In Chapter 1, we discussed the SHM background and introduced a guided Lamb wave-based damaged detection method. The objectives and the proposed approach are discussed as well.

Chapter 2 provides an overview of various deep learning algorithms that potentially benefit SHM applications.

Chapter 3 discusses the dataset preparation for these guided Lamb wave responses. Additionally, physical insights are collected from the wave response data to demonstrate the damage location/extent effects on the wave responses.

Chapter 4 describes the methodologies used for training different neural network models. It provides information about the architectures and the training hyperparameters. Additionally, it presents a discussion on model selection and evaluation techniques.

Chapter 5 presents the results of various tests conducted. This chapter includes a comparison of the performance of the different models under various noise levels.

Chapter 6 concludes the thesis by summarizing the key findings and also provides potential areas for future research work.

Chapter 2. Background and Related Works

This chapter delves into the various feature selection techniques and deep learning methods for SHM. First, it provides an overview of feature selection methods commonly used in this domain. After this, the chapter explores the various deep-learning models employed for SHM. It focuses on different unique architectures and their suitability for analyzing Lamb wave response data.

2.1 Feature Selection Techniques

Analyzing vibration-based data such as Lamb wave response data, presents a significant challenge due to its high dimensionality. This happens as numerous data points are sampled to capture the behavior of the wave responses. Traditional data mining methods, which are designed for simpler data structures, often struggle with such complexity. For high dimensional data, dimensional reduction techniques like PCA, and LDA offer some solutions. In a study by Prabhav *et al.* [6], POD was applied to extract features and reduce the dimensionality of data, and then the features were used to train a shallow neural network for damage location and thickness reduction. Similarly, Farrar *et al.* [9] applied to LDA for vibration-based health monitoring of a concrete bridge column to detect damaged and undamaged states. However, such dimensionality reduction methods can potentially discard essential information embedded within the signal. This can compromise the accuracy and effectiveness of SHM systems.

So, for processing of vibration-based data which is essentially time series data, there are four common categories of representations – piecewise linear, transform-based, symbolic, and model-based [5]. Among the four common categories, transform-based representations are common in SHM. In this method, the original time series data is converted into a different domain (*e.g.,* frequency domain) using Fourier, cosine or wavelet transforms.

The Fourier transform is a mathematical technique for analyzing functions and signals in terms of their frequency components. For a time-continuous signal $x(t)$ with frequency f, it is given by equation 2.1:

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt, \tag{2.1}$$

where $\omega = 2\pi f$ is the angular frequency.

These methods also often provide efficient data compression as the first few coefficients contain most of the sequence energy. Moreover, high-frequency components often correspond to noise. Thus, selecting low-frequency components naturally filters them out, providing noise reduction for damage detection.

Several studies have demonstrated the effectiveness of transform-based representations in SHM. Hoshyar *et al.* [15] successfully localized concrete cracks using frequency information obtained from vibration signals. Furthermore, Tehrani *et al.* [1] employed the generalized S-transform combining features of both Fourier and wavelet transforms, to assess damage severity in structures. Similarly, Nguyen *et al.* [25] achieved better performance when combining frequency transform-based representations with deep learning for damage evaluation. This suggests the possibility that this integrated approach can achieve robust damage detection using lamb wave responses.

2.2 Deep Learning Methods

While feature selection has traditionally played a crucial role in preparing data for machine learning models, recent advancements in deep learning suggest possibilities for bypassing this step in Lamb wave-based SHM. Deep learning, is a subset of machine learning, which has achieved success in numerous tasks such as image and speech recognition, natural language processing, and generative artificial intelligence [37]. The success of deep learning can be attributed to its capacity to handle vast amounts of data, scalability, and adaptability to diverse domains. There also has been a lot of research involving deep learning in the field of vibration-based SHM [35].

2.2.1 Multi-Layer Perceptron

The artificial neuron model was first proposed by Warren McCulloch and Walter Pitts in 1943 [36]. Later, it was implemented by Rosenblatt in 1958 as a perceptron, which demonstrated the ability to learn through a combination of linear transformations and nonlinear thresholding functions [30]. Minsky and

Papert [23] presented mathematical proofs that showed that while perceptrons can effectively learn any linearly separable function, they face challenges in handling functions that require nonlinear decision boundaries, such as the XOR function.

In the artificial neuron model, inputs are multiplied by learned weights, and the resulting product is then subjected to a nonlinear activation function. Finally, the output is passed on to the subsequent layer. If we denote the input feature vector as x, the weight matrix as $\mathbf{W}$, the bias as b, and the non-linear activation function as σ, we can express the output y as follows:

$$y = \sigma(\mathbf{W}x + b). \tag{2.2}$$

Despite these limitations, the introduction of networked layers in an MLP has shown its capabilities in learning complex patterns and functions. In terms of SHM applications, Hekmati Athar *et al.* [13] demonstrated the effectiveness of MLP by training it on a collection of direct and indirect sensor data to detect damage to bridge structures.

A multi-layer perceptron, also known as a feed-forward network, consists of an input layer, one or more hidden layers, and an output layer. The input layer of an MLP receives the data or features that are input into the neural network. In this layer, each neuron represents a specific feature, with the values at these nodes corresponding to the input features of the dataset.

There are one or more hidden layers in an MLP between the input and output layers. The connection between neurons in different layers is associated

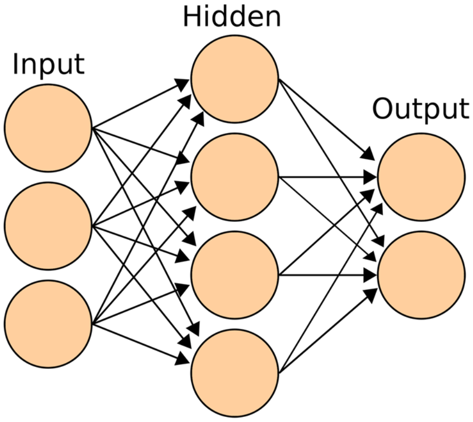

Figure 2.1: Typical architecture of MLP.

with weight. The weights are adjusted during the training process. To introduce non-linearity, nonlinear activation functions are applied to the weighted sum of inputs. This non-linearity enables the network to capture and learn complex patterns in the data.

The output layer is responsible for producing the final predictions of the neural network. The number of neurons in the output layer depends on the specific task. For instance, a classification task might have one neuron per class.

When dealing with high-dimensional data, the computational cost of MLP models can grow exponentially as there is significant growth in the number of parameters with the increase in features. This can result in challenges like overfitting [31]. To address these issues, a combination of dimensionality reduction and feature selection techniques are often employed along with MLP models.

2.2.2 Long-Short Term Memory

RNN is a specialized type of neural network that is often applied to sequence datasets. The architecture of an RNN includes recurrent connections that allow the network to maintain a memory of previous inputs. Thus, making it suitable for tasks involving sequential dependencies [16]. However, traditional RNNs face the challenge of vanishing gradients. The vanishing gradient problem occurs during training when the gradients diminish as they are backpropagated. To address this issue, LSTM networks were introduced, which incorporate gating mechanisms [14]. LSTMs control information flow with three special gates:

- Forget gate: Decides which information from the previous cell state to discard
- Input gate: Selects the relevant information from the current input and combines it with a processed cell state.
- Output gate: Determines which part of the processed cell state to expose as the output.

These gates empower LSTMs to learn long-term dependencies within sequences.

In SHM, Sony *et al.* [33] used an LSTM network with a sequence of windowed acceleration signals for damage detection and localization. Lin *et al.* [21] trained LSTM with raw multipoint measurements getting superior results over other techniques. Beyond standalone applications, LSTMs are often combined with other networks. For eg. a hybrid approach has also been tried by combining RNNs with features extracted by CNNs for SHM [7]. While these studies highlight the potential of LSTMs in extracting relevant information directly from wave response, the training of the network can still be unstable. While gating mechanisms mitigate the vanishing gradient problem, exploding gradients can also arise, leading to unstable training. Moreover, the addition of gating mechanisms in LSTMs increases the computational cost and this can be a challenge for resource-constrained environments.

2.2.3 Convolutional Neural Network

The CNN is a deep learning model that has gained immense popularity for its unique architecture featuring convolutional layers. This model was first developed by Yann LeCun, who pioneered the development of ”LeNet” and achieved great success with the MNIST dataset [19]. CNN gained even more widespread recognition following AlexNet’s victory in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 [18].

The advantage of CNNs is that they integrate feature extraction within a single framework. This is achieved by the use of convolutional layers, which extract relevant features from input data. Additionally, CNNs offer sparse connectivity, which means that each neuron is only connected to a small subset of the input data. This helps to reduce the computational complexity of the model and make it more efficient.

Another important feature of CNNs is parameter sharing. This means that the same set of weights is used for multiple inputs. This helps to reduce the number of parameters that need to be learned. Finally, CNNs exhibit shift invariance, which means that they can recognize patterns in an image regardless of their position. This has enabled CNN to be well-suited for image-related tasks such as image classification, and object detection.

CNN architectures typically incorporate convolutional layers, subsampling layers, and fully connected layers. Convolutional layers extract features from the input data. This layer performs convolution with several filters or kernels. These filters move over the input data, learning various spatial features such as edges, textures, and shapes. The output feature map from the convolutional layer can be as given by equation 2.3:

$$o_j = b_j + \sum_{i=1}^{N} \mathrm{conv2D}(w_{ij}, x_i). \tag{2.3}$$

Here, conv2D is a 2D convolutional operation, o_j is the j^{th} output channel, x is the input with N channels, b_j is the bias term associated with j^{th} output channel, w_{ij} is the kernel weight associated between the i^{th} input channel and j^{th} output channel.

The subsampling layers lower the spatial dimensions of the data while keeping significant properties. These layers often apply pooling, which involves calculating the maximum or average value in a particular window of input data. Subsampling layers reduce the amount of parameters in the network, which helps to prevent overfitting and improve network performance.

Finally, the fully connected feed-forward layer integrates all of the high-level features learned in the convolutional layers. This layer is responsible for the network's output.

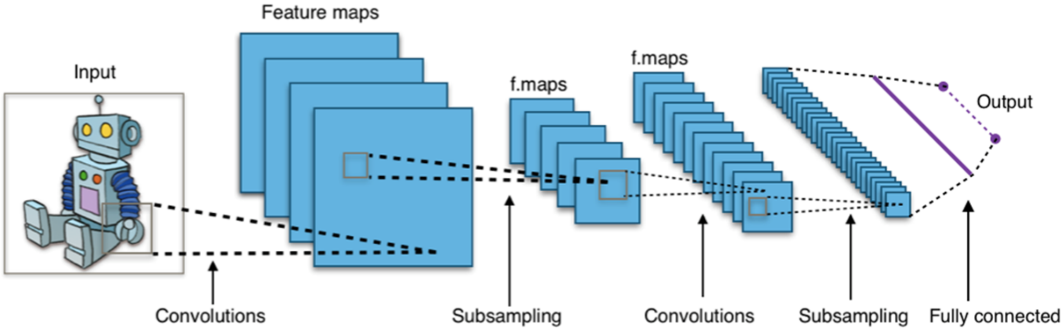

Figure 2.2: Architecture of CNN [8].

While CNNs were initially developed for 2-dimensional image-related tasks such as image classification, segmentation, and object detection, they have found diverse applications in SHM. Various studies have been conducted for damage detection and localization. For example, Rosafalco *et al.* [29] approached damage detection as a classification problem, utilizing a fully convolutional network by stacking multivariate time series data. Other implementations involve the transformation of 1D signal data into 2D formats to be processed by CNNs. In a study focused on a steel frame structure with bolted connections, a time-frequency scalogram was employed to convert 1D signals into 2D data [26]. Another study conducted by He *et al.* [12] utilized the fast Fourier transform and dimensional transformation to convert 1D signals into 2D data.

2.2.3.1 1-dimensional Convolutional Neural Network

The 1D CNN is a type of CNN that uses a one-dimensional convolution operator in its convolutional layers. The output is given by the equation 2.4:

$$o_j = b_j + \sum_{i=1}^{N} \text{conv1D}(w_{ij}, x_i). \tag{2.4}$$

The difference between equation 2.3 and 2.4, is that the 1D CNN uses one-dimensional convolutional operation (conv1D). Given an input sequence $x = [x_1, x_2, ..., x_n]$ and one dimensional kernel $w = [w_1, w_2, ...w_k]$, the output of 1D convolutional operator can be given as follows:

$$y_i = \sum_{j=1}^{j=k} w_j \cdot x_{i-j}. \tag{2.5}$$

So, the kernels in 1D CNNs slide across the temporal axis of the data, which means they could be directly applied to time series. And unlike applying 2D CNNs, it does not force unnecessary spatial structures that might not be meaningful.

Furthermore, if the convolution of continuous time domain signal $f(x)$ and $g(x)$ is given as:

$$(f * g)(x) = \int_{-\infty}^{\infty} f(t)g(x-t)dt. \tag{2.6}$$

And the Fourier transform of the function $f(x)$ and $g(x)$ is $F(\omega)$ and $G(\omega)$ respectively, then from equations 2.1 and 2.6 convolution operation in the frequency domain can be given as follows:

$$
\begin{aligned}
\mathcal{F}\{f(x) * g(x)\} &= \int_{-\infty}^{\infty} (f * g) e^{-j\omega x} dx \\
&= \int_{-\infty}^{\infty} \left(\int_{-\infty}^{\infty} f(t) g(x-t) dt \right) e^{-j\omega x} dx \\
&= \int_{-\infty}^{\infty} f(t) \left(\int_{-\infty}^{\infty} g(x-t) e^{-j\omega x} dt \right) dx \\
&= \int_{-\infty}^{\infty} f(t) e^{-j\omega t} \left(\int_{-\infty}^{\infty} g(u) e^{-j\omega u} du \right) dt \qquad (2.7) \\
&= \int_{-\infty}^{\infty} f(t) e^{-j\omega t} G(\omega) dt \\
&= G(\omega) \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt \\
&= F(\omega) G(\omega),
\end{aligned}
$$

where $u = x - t$ and $du = dx$ and $\mathcal{F}$ denotes Fourier Transform. This means when we convolve two functions in the time domain, it's equivalent to multiplying their Fourier transforms in the frequency domain. So the features extracted by the convolutional operator could correspond to various filtering operations that can filter out noise or extract specific frequency components from the input signal. Thus, the learned features by 1D CNN can not only capture relevant spatial patterns but also help in denoising the signal.

In 2015, Kiranyaz *et al.* [17] introduced the compact 1D CNNs designed specifically for patient-specific electrocardiogram (ECG) data. These networks are smaller and more efficient than traditional MLP and 2D CNNs. In this architecture, the kernel slides over the time dimension of the data and captures local patterns within sequential data.

1D CNNs have also proven useful in SHM, where they have been used for damage detection and real-time monitoring. For example, Avci *et al.* [2] used 1D CNNs to detect damage on benchmark data. They later used these networks for realtime monitoring using a wireless sensor network [3]. Similarly, Zhang *et al.* [38] introduced a 1D CNN architecture that can process raw sensor data directly. They created three separate acceleration databases using a T-shaped steel beam, a short steel girder bridge, and a long steel girder bridge. The network they developed was able to detect small local changes in both structural mass and stiffness, making it a useful tool for SHM applications.

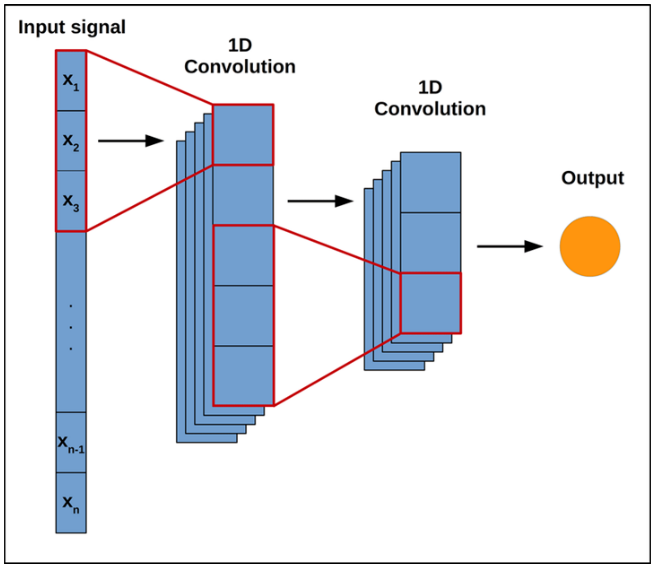

Figure 2.3: 1D Convolutional Neural Network [32].

2.2.4 Autoencoder

Autoencoders are another type of neural network architecture that first transform input data into a smaller representation that can later be reconstructed back into its original form [4]. This two-part system consists of an encoder network and a decoder network.

The encoder can be viewed as a transform $f_\phi(x)$ which maps a d dimensional input vector $x \in \mathcal{R}^d$ into q dimensional hidden representation $z \in \mathcal{R}^q$. Encoder network usually applies affine transformation followed by nonlinear transformation ψ, which can be given as follows:

$$z = \psi(\mathbf{W}x + b), \tag{2.8}$$

where $\mathbf{W} \in \mathcal{R}^{q\text{x}d}$ is the weight matrix, $b \in \mathcal{R}^r$ is the bias and ψ is the activation function.

The decoder $g_\theta(x)$ then maps the hidden representation z back into reconstructed vector $x' \in \mathcal{R}^d$ in the input space. The operation performed can be given as below:

$$x' = \psi(\hat{\mathbf{W}}z + \hat{b}), \tag{2.9}$$

where $\hat{\mathbf{W}} \in \mathcal{R}^{q\text{x}d}$ is the weight matrix of decoder, $\hat{b} \in \mathcal{R}^d$ is the bias, and ψ is the activation function.

The parameters $\mathbf{W}$, b, $\hat{\mathbf{W}}$, $\hat{b}$, are optimized with respect to reconstruction loss which can be given as equation 2.10. The reconstruction loss $L(x', x)$ measures the difference between the original and reconstructed signal:

$$[\mathbf{W}^*, b^*, \hat{\mathbf{W}}^*, \hat{b}^*] = \mathbf{argmin}_{\mathbf{W},b,\hat{\mathbf{W}},\hat{b}} L(g(f(x)), x). \tag{2.10}$$

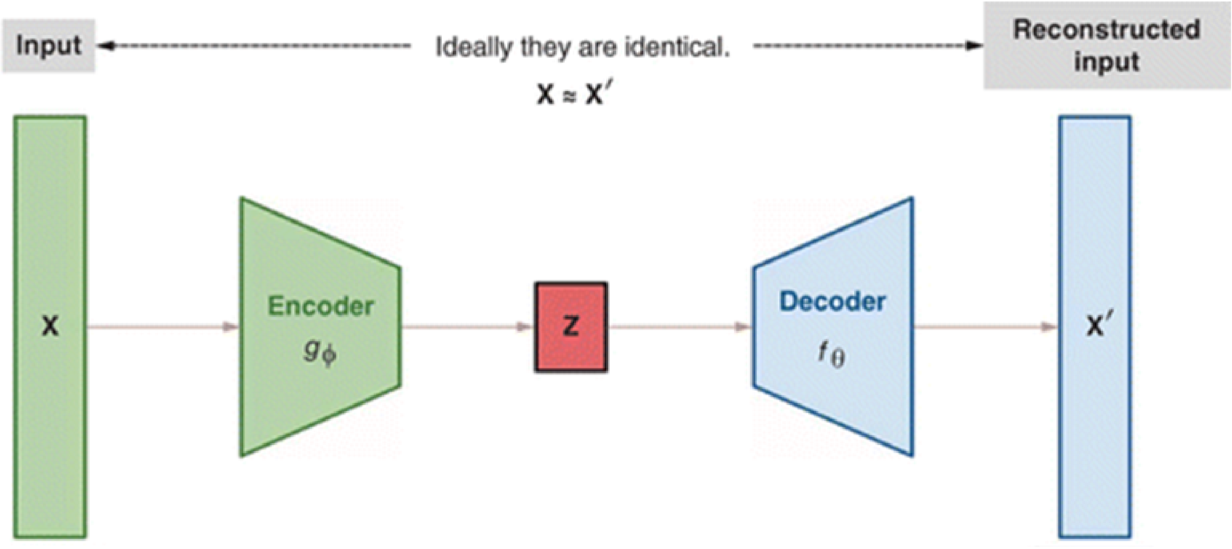

Figure 2.4: Architecture of Autoencoders [24].

Denoising autoencoders are a type of autoencoders that are fed noisy input data and are forced to reconstruct the original clean input data. By doing so, the network learns a more robust representation in the process. This approach is particularly useful for filtering out unwanted noise from input data.

In a study, Pathirage *et al.* [27] used autoencoder to first obtain a compact representation of frequency and modal information. Then, they utilized non-linear regression to train and learn structural stiffness parameters. This highlights the promise of using this approach for guided lamb wave response-based damage detection.

Chapter 3. Data Preparation

This study used a dataset created by simulating Lamb wave responses in a cantilevered beam. To capture the wave responses of the beam, we employed a SFEM based on the Timoshenko beam theory [20] [34]. The SFEM accounted for shear deformation and rotary inertia. The beam was subjected to a tip force, and we computed the resulting tip displacement.

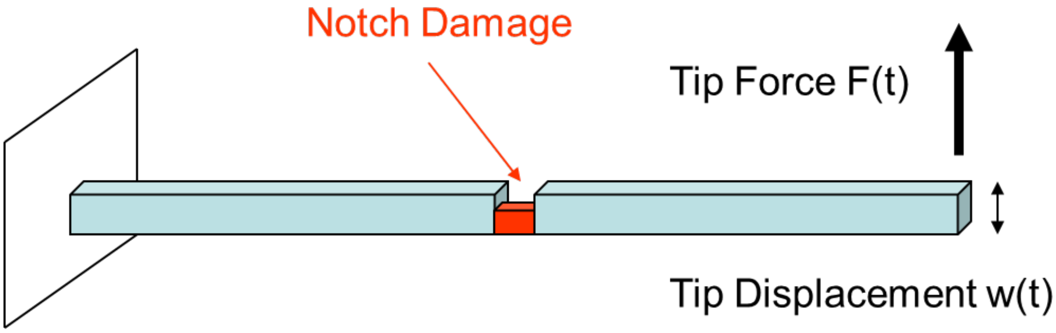

Figure 3.1: A cantilevered beam system.

The beam model was 0.5 m long and had a rectangular cross-section of 10x2mm (WxH). It had a density of 2700 kg/m^3, a Young's modulus of 70 GPa, and a Poisson ratio of 0.25. The input force waveform was a 50 KHz sine burst Hanning window at the tip with a duration of 0.1 msec [6], as shown in Figure 3.2

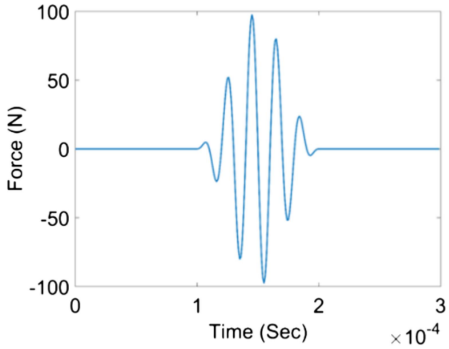

Figure 3.2: Waveform of tip force.

To replicate the damaged condition, a single notch was introduced into the beam. Damage locations varied from 25% to 85% of the beam's length, while thickness reduction ranged from 2% to 15%. A total of 853 samples were generated. The time-series signal had a duration of 2.5ms and consisted of 2500 data points.

When a force is applied to a beam, time-domain signal response consists of instantaneous response and reflection from the boundary. However, if there is any damage to the beam, like a notch, an additional reflection from the damaged area can also be observed. Figure 3.3 shows the signal response for undamaged and damaged cases which has been truncated at 1ms to show entire wave propagation characteristics. Thus, the presence of damage can be reflected in wave response and the time of flight and the amplitude of the reflective response indicate the location and severity of the damage [6].

In real-world situations, data collected from sensors are often noisy due to various factors. To better model actual sensor data, we add different levels

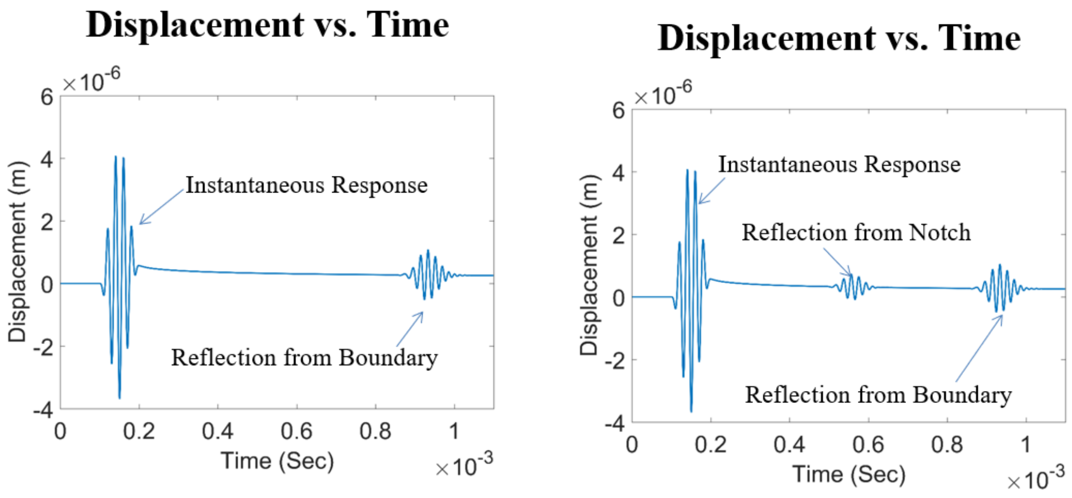

Figure 3.3: Time-domain response: undamaged case(left) vs damaged case(right) [6].

of noise to the simulated dataset as given by equation 3.1. Datasets with three different noise levels 5%, 10%, and 15% were generated:

$$\hat{x}_i = x_i(1 + \alpha * n_i). \tag{3.1}$$

Here, $\hat{x}_i$ represents the signal with added noise, n_i denotes the noise sample drawn from a uniform distribution (-1, 1), and α signifies the intensity level of the noise in percentage.

The figures 3.4 - 3.7 show the samples of data with damage location at 25% of beam length and thickness reduction level at 12% for varying intensity levels of noise.

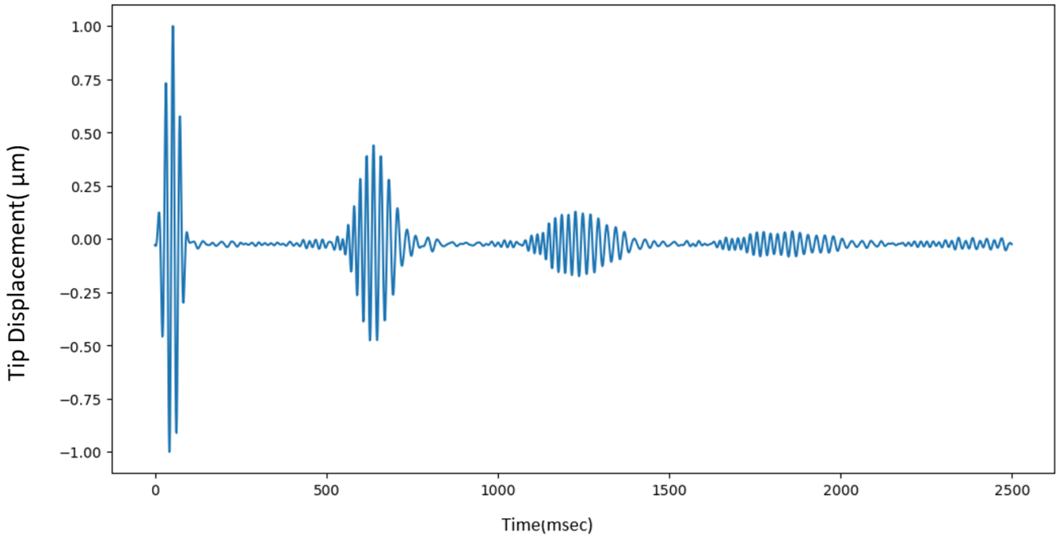

Figure 3.4: Noise free wave response.

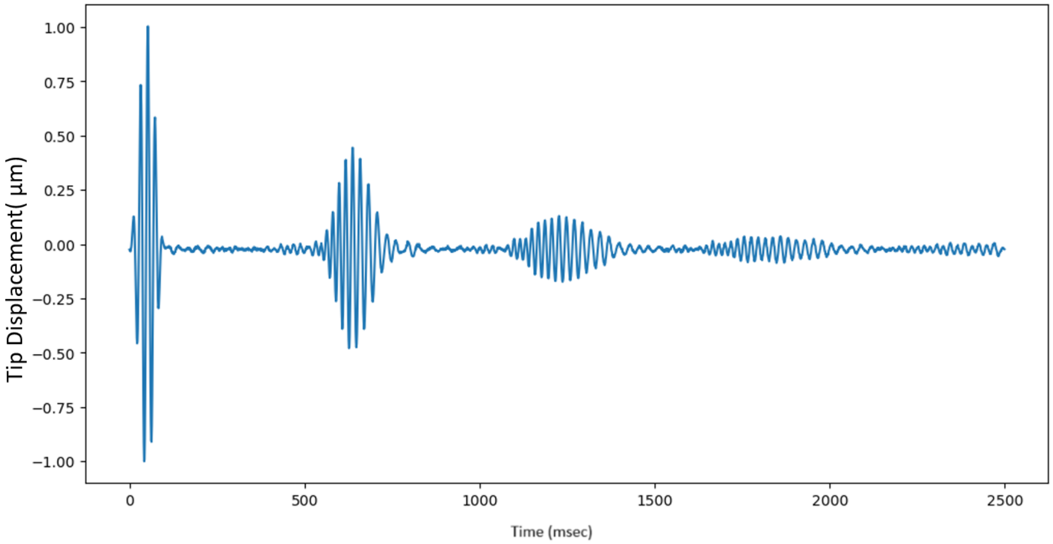

Figure 3.5: Wave response with 5% added noise level.

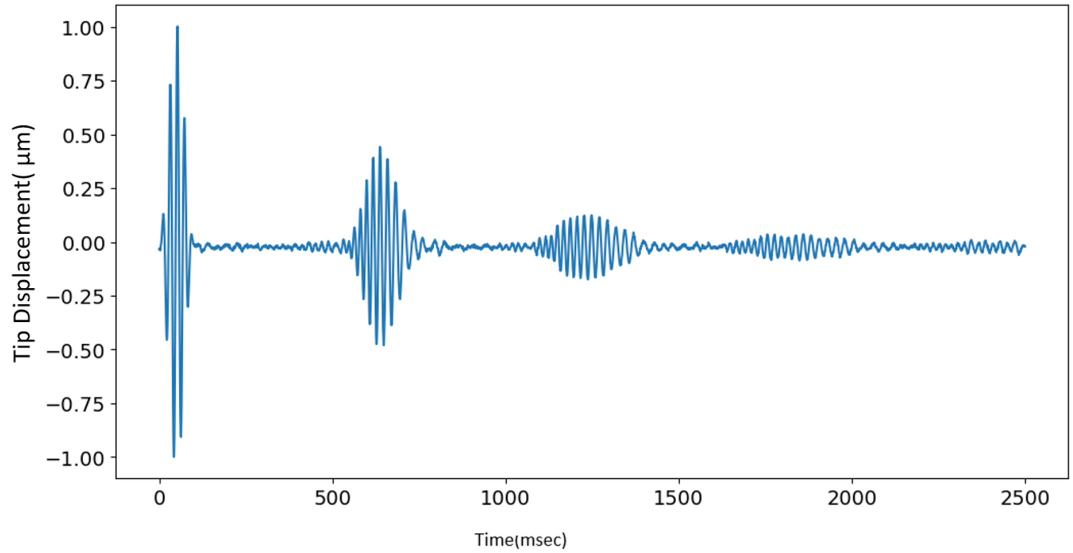

Figure 3.6: Wave response with 10% added noise level.

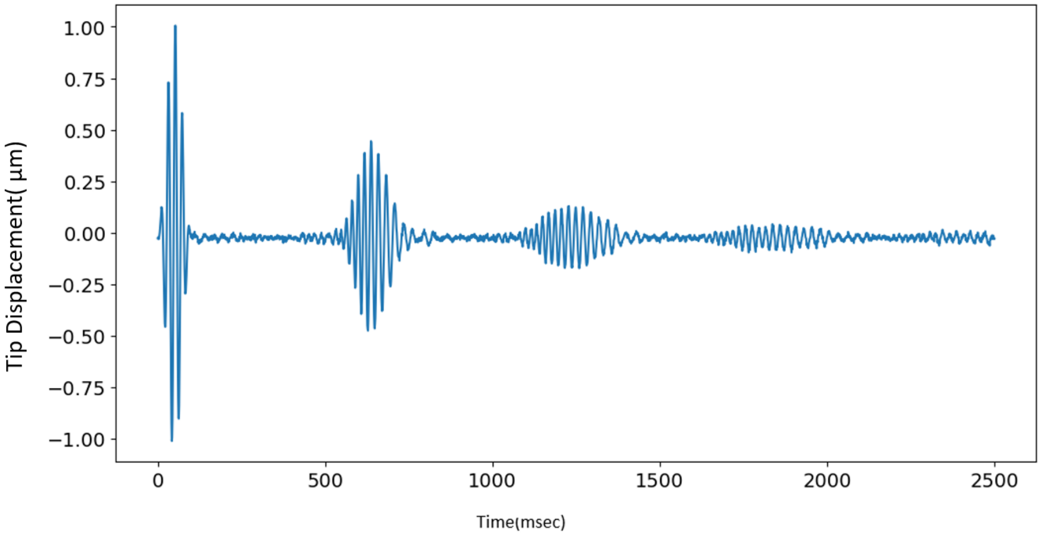

Figure 3.7: Wave response with 15% added noise level.

Chapter 4. Methodologies

In this chapter, the methodologies employed in the training of distinct deep neural network architectures are outlined. As the investigation focuses on three specific models: MLP, 1D CNN, and autoencoder, we provide details of their architecture and the specific methodologies applied during the training process. The discussion also delves into model selection and model evaluation strategies for trained models.

4.1 Multi-layer Perceptron

For training MLP networks with frequency domain features, the signal data is first passed onto the FFT layer which transforms the time domain signal to the frequency domain. This layer helps to extract a more meaningful feature representation of the data and subsequently reduces the data dimension for effectively training the network.

The DFT is a specific form of the Fourier transform that is designed for digital signals. It is given by equation 4.1:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N}, \tag{4.1}$$

where X_k is time-domain data points, N is the total number of sampling points and x_n is the complex numbers.

The FFT is an efficient algorithmic implementation of the DFT. The FFT algorithm reduces the time complexity needed for N points from $O(N^2)$ to $O(NlogN)$. During the data transformation process, both the very low and very high frequency components contribute negligibly to the overall signal, as illustrated in figure 4.3. It is seen that the most of signal component is between 22KHz to 76 KHz. Consequently, the frequency components are truncated to a selected band within this range, resulting in a 135-dimensional vector representation. After this transformation step, the input data is fed to the MLP network, consisting of three hidden layers sizes of 768, 512, and 256 neurons respectively. The network is trained to predict the damage location and the thickness reduction in the beam.

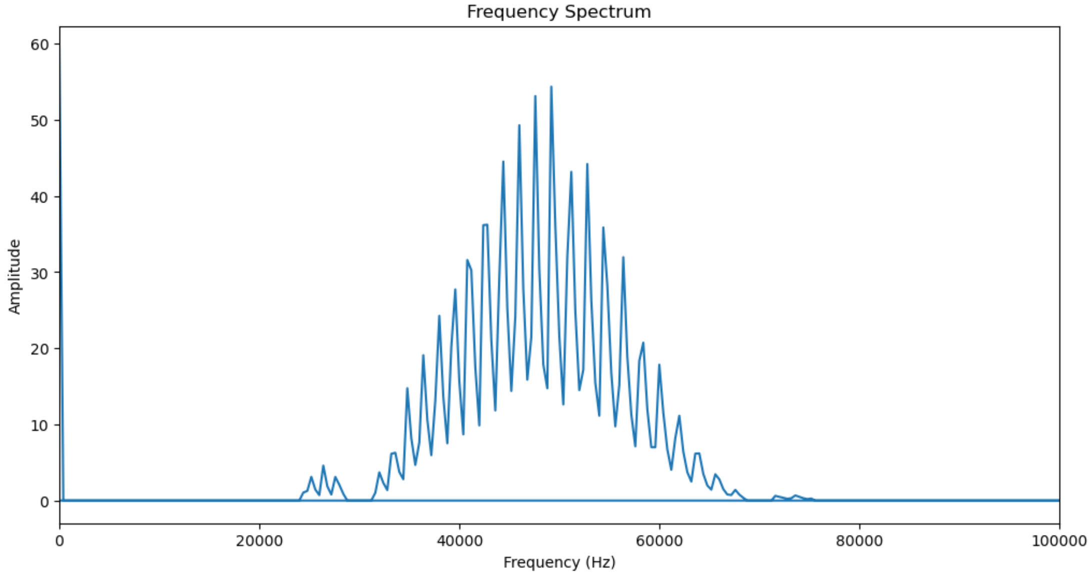

Figure 4.1: Frequency Spectrum of Wave Response.

4.2 Autoencoder

The proposed autoencoder architecture comprised an encoder, a decoder, and a regression head implemented using fully connected layers. The encoder network had hidden layer sizes of 1024, 512, and 256 neurons, while the decoder network had hidden layer sizes of 100 and 256 neurons. The encoder network compressed the input signal into a latent vector space of 50 dimensions. The regression network, with hidden layer sizes of 1024, 768, 512, and 256 neurons, was responsible for predicting the location and intensity of damage based on the encoded features from the latent space. The input data fed into the autoencoder was the noisy wave responses. The decoder network aimed to reconstruct the orig-

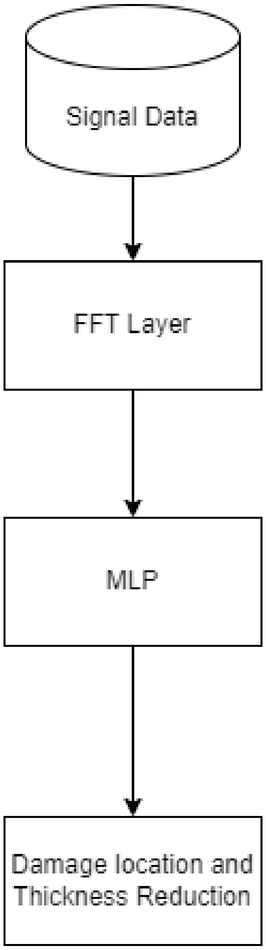

Figure 4.2: Flowchart of proposed FFT based MLP model.

inal clean signal from the compressed latent representation. During the training phase, the autoencoder had two primary objectives:

1. Minimizing the reconstruction error to learn a meaningful representation of input data, and

2. Optimizing the regression loss to accurately predict damage location and the extent of thickness reduction.

During the training process, the mean squared error loss function was employed to optimize two objectives simultaneously.

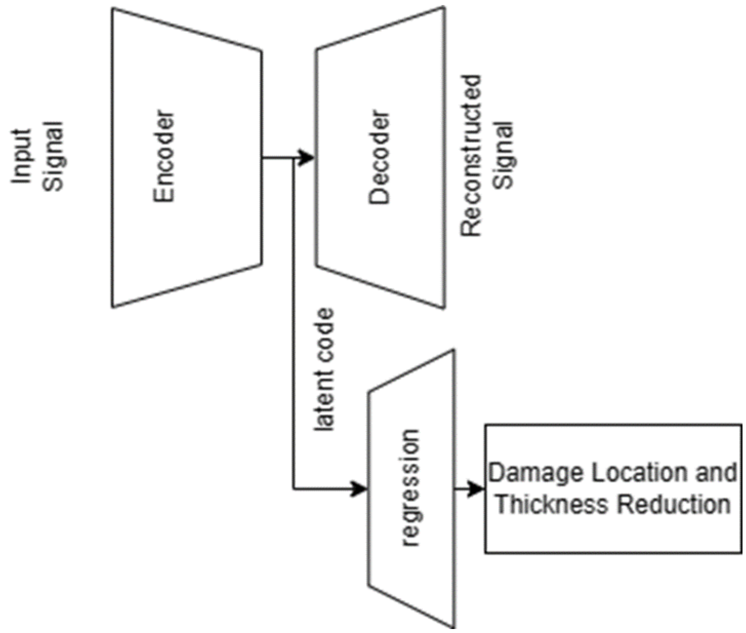

Figure 4.3: Proposed Architecture of Autoencoder Network.

4.3 1D CNN

The CNN model architecture consisted of three 1D convolution layers, each followed by a pooling layer. Max pooling was employed for the pooling operation. The configuration parameters of the convolutional and max pooling layers are shown in Table 4.1. The convolutional layers utilized kernel sizes of 3 and had 32, 64, and 128 filters, respectively. After the convolutional layers, a flattening layer was applied, followed by fully connected layers sized 768,512 and 256 neurons. Given the ability of 1D CNN to handle time series data directly, no preprocessing of the data was performed, and the model was trained directly on the raw time domain signals.

Table 4.1: Convolutional and Max Pool layer configuration of 1D CNN.

Layer Name	Filter Size	Kernel Size	Padding	Stride
Convolution1D 1	32	3	0	1
Max Pool1D 1	32	2	0	2
Convolution1D 2	64	3	0	1
Max Pool1D 2	64	2	0	2
Convolution1D 3	128	3	0	1
Max Pool1D 3	128	2	0	2

4.4 Model Selection

In our study, we divided the dataset into 80% for the training set, 10% for the validation set, and the remaining 10% for the test set. The training set is the largest subset of the data, which is used to train the machine learning model. During the training process, the neural network learns patterns and relationships from the training data by adjusting its parameters to minimize the loss function. The test set is a subset of data that the model has not seen during training. It serves as an independent dataset to assess how well the trained model generalizes to new, unseen data. The validation set is another subset of unseen data that is used to fine-tune the hyperparameters of the model. This helps to approximate the generalization error and prevent data leakage during the training phase. The validation set is also used to ensure that the model is not overfitting the training data and to check that it can generalize well to unseen data.

4.4.1 Training Hyperparameters

The table 4.2 shows the hyperparameters used for the training neural networks. The training algorithm used Adam optimizer, the batch size was 64 and the learning rate was 0.0001 for MLP. The learning rate for the autoencoder model and 1D CNN model was 0.001. To train the regression model, we minimize mean square error (MSE) loss, which is given as follows:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (\hat{y_i} - y_i)^2, \tag{4.2}$$

where y_i is the actual value, $\hat{y}$ is the predicted value, and n is the total number of data points.

Table 4.2: Training Hyperparameters.

Model Name	Optimizer	Batch Size	Epochs	Learning Rate
MLP	Adam	128	6000	0.0001
1D CNN	Adam	128	3000	0.001
Autoencoder	Adam	128	3000	0.001

4.4.2 Regularization

Regularization is a technique used to prevent overfitting in the model and enhance the generalization capability of the machine learning model. Dropout and weight decay were used in our training for regularization. Dropout involves randomly deactivating (dropping out) a fraction of neurons during training, forcing the network to learn redundant representations and enhancing its ability to generalize to unseen data. Weight decay penalizes large weights during training, pushing them closer to zero and preventing overfitting. A dropout value of 0.1 and a weight decay value of 0.001 were used during training.

4.4.3 R^2 score

To evaluate the performance of regression models, R^2 score metric is usually employed. R^2 score, also known as the coefficient of determination, is a statistical measure that assesses the proportion of the variance in the dependent variable (target) that is predictable from the independent variables (features). It indicates the goodness of fit of the model. It is given as:

$$\begin{aligned} R^2 &= 1 - \frac{SS_{res}}{SS_{tot}} \\ SS_{res} &= \sum_{i}^{n}(y_i - \hat{y})^2 = \sum_{i} e_i^2 \\ SS_{tot} &= \sum_{i}^{n}(y_i - \bar{y})^2. \end{aligned} \tag{4.3}$$

SS_{res} is the sum of squared residuals (the sum of the squared differences between the predicted and actual values of the dependent variable). SS_{tot} is the total sum of squares (the sum of the squared differences between each data point and the mean of the dependent variable).

A R^2 score of 0 indicates that the model does not explain any variability in the target variable and the model's performance is no better than predicting the mean, whereas 1 indicates that the model perfectly predicts the target variable.

Chapter 5. Evaluations and Results

5.1 Numerical Test 1: Evaluating Performance of Feature Selection Method

To evaluate the effectiveness of frequency domain feature selection, we employed two separate MLP models. One model used the raw wave response data while the other model utilized the selected frequency components from the FFT as input features. Both the MLP models were configured with the same network architecture and were trained using the same hyperparameters.

The table 5.1 shows the performance metrics of both models in various noise conditions. The FFT-MLP model outperformed the time-domain data model, especially in noisy environments. In the higher noise signal at the level of 15%, the FFT-MLP model had R^2 score of 0.9586 compared to the R^2 score of 0.9152 for the raw data model. This demonstrates the effectiveness of FFT in filtering noise and extracting relevant features for training the neural network.

Figure 5.1 shows this filtering process. By only using the low-frequency range components, the clean signal can be reconstructed. Thus, these lower frequencies retain the necessary information to represent the original signal data.

Table 5.1: Performance Comparision for FFT-MLP model and MLP models.

Model	Noise level(%)	R^2 score	Mean Percentage error	
			Location	Thickness
MLP	0	0.9893	2.3	6.9
	5	0.9834	3.1	6.3
	10	0.9569	6.1	6.0
	15	0.9152	9.5	5.4
FFT-MLP	0	0.9977	1.4	2.1
	5	0.9887	2.8	5.1
	10	0.9681	4.7	8.6
	15	0.9586	5.2	7.0

5.2 Numerical Test 2: Evaluating End-to-End Deep Learning Models

In the second experiment, we trained and evaluated the autoencoder and 1D CNN model to investigate the effectiveness of end-to-end deep learning models in noisy environments.

The performance of both models demonstrated robustness to noise. They maintained consistently high performance across all noise levels. The 1D CNN achieved the overall best performance, with high R^2 scores ranging from 0.9964 at no noise condition to 0.9908 at the highest noise level of 15%. Furthermore, the mean percentage error remained within 1.2% for damage localization and 5.7% for thickness reduction prediction. Similarly, the autoencoder also exhibited better performance than the MLP model, with R^2 scores ranging from 0.9943 to 0.9869 across the noise levels. The mean percentage error for the autoencoder remained within 2.7% for damage localization and 5.5% for thickness reduction at the maximum noise level of 15%. These results are summarized in table 5.2.

Table 5.2: Performance of 1D CNN and autoencoder under various noise conditions.

Model	Noise level(%)	R^2 score	Mean Percentage error(%)	
			Location	Thickness
1D CNN	0	0.9964	1.0	3.9
	5	0.9956	1.2	3.9
	10	0.9920	1.5	5.1
	15	0.9908	1.2	5.7
Autoencoder	0	0.9943	0.97	5.1
	5	0.9911	2.1	5.5
	10	0.9918	1.8	6.1
	15	0.9869	2.7	5.5

5.3 Discussions

From the result of these two experiments, we find that the end-to-end deep learning approach has a more robust performance than the FFT-MLP model. While the FFT-MLP demonstrated a good level of performance in low noise level data, the model experienced performance drops with increasing noise levels. This demonstrates that 1D CNN and autoencoder can capture relevant representation directly from the Lamb wave response data.

Compared to the FFT-MLP model, the convolutional layers in 1D CNN can extract spatial information from sequential input, and it can also learn various filters due to the use of convolutional operations contributing to its higher damage prediction accuracy. Additionally, the autoencoder also demonstrates its representation learning by encoding the data with only 50 latent dimensions compared to the 135-dimensional features extracted using FFT-based methods. From the figure 5.2, we can see that noise is reduced in the reconstructed signal from the 50 latent dimensions. This compact representation potentially contributes to the autoencoder's noise resilience.

Figure 5.3-5.5 illustrates the actual values and predicted values for damage localization and thickness reduction generated by each model under the noise level of 15%. From these figures, it is seen that there is more variability in predicting the damage location than in predicting the thickness reduction. The models suffer more when predicting damage location when the severity of damage is very low(2%). In such scenarios, the high noise level present in the guided

Table 5.3: Examples of actual and predicted values at 15% noise level.

Actual value (Location%, Reduction%)	1D CNN Predicted value	FFT-MLP Predicted value	Autoencoder Predicted Value
(25, 12)	(24.5, 11.5)	(23.5, 11.0)	(25.5, 12.1)
(33, 2)	(33.5, 2.3)	(41.4, 2.0)	(36.7 2.0)
(33, 5)	(33.8, 5.0)	(31.7, 3.7)	(31.7, 4.8)
(49, 11)	(48.9, 10.8)	(49.5, 10.2)	(49.9, 11.3)
(49, 14)	(48.9, 14.2)	(48.8, 14.2)	(48.9, 14.1)
(50, 6)	(49.1, 5.3)	(51.2, 6.1)	(49.5, 6.0)
(53, 2)	(50.1, 1.9)	(67.1, 2.58)	(59.1, 2.4)
(54, 4)	(52.4, 3.6)	(53.1, 4.3)	(53.6, 4.0)
(57, 11)	(56.9, 11.2)	(57.2, 10.5)	(57.1, 11.8)
(68, 4)	(68.8, 3.8)	(68.8, 3.7)	(67.8, 4.0)
(71, 12)	(71.0, 12.1)	(77.8, 10.77)	(70.6, 12.6)

wave responses appears to mask the subtle signatures associated with the low severity damage. For FFT-MLP, this is more pronounced as the damage location predictions deviate significantly from the true values. Table 5.3, which provides sample predicted and actual values for the models at the 15% noise level, clearly highlights this limitation. For example, at a thickness reduction of 2%, the FFT-MLP predicts a value of 41.4%, and 67.1% when the damage location is at 33% and 53% of beam length respectively. In contrast, the 1D CNN, and autoencoder, exhibit better localization accuracy. Their predictions often fall closer to the true values. This can be attributed to their ability to automatically learn robust representations directly from the raw guided wave response data through end-to-end training.

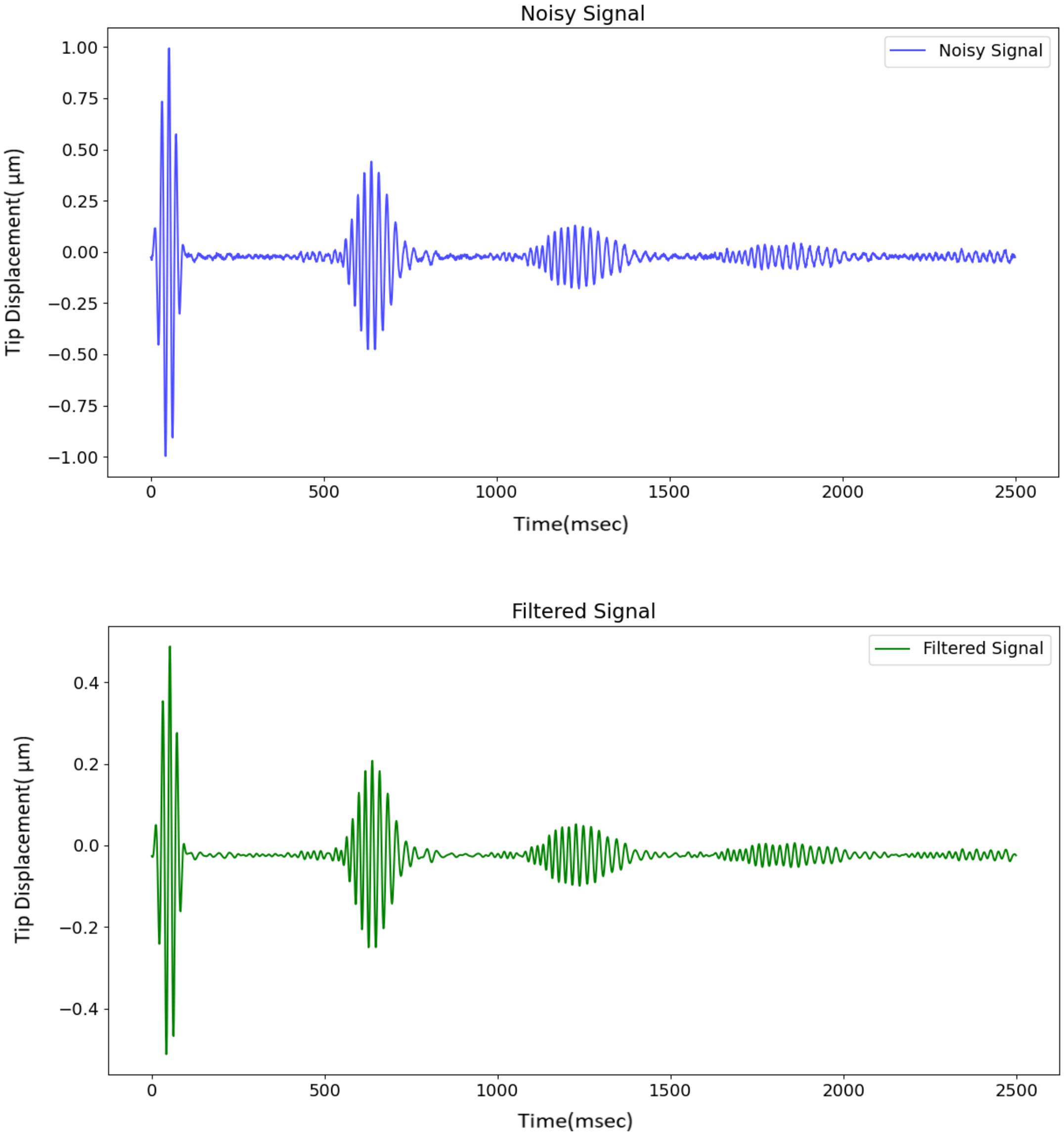

Figure 5.1: Noisy and reconstructed wave response using FFT features.

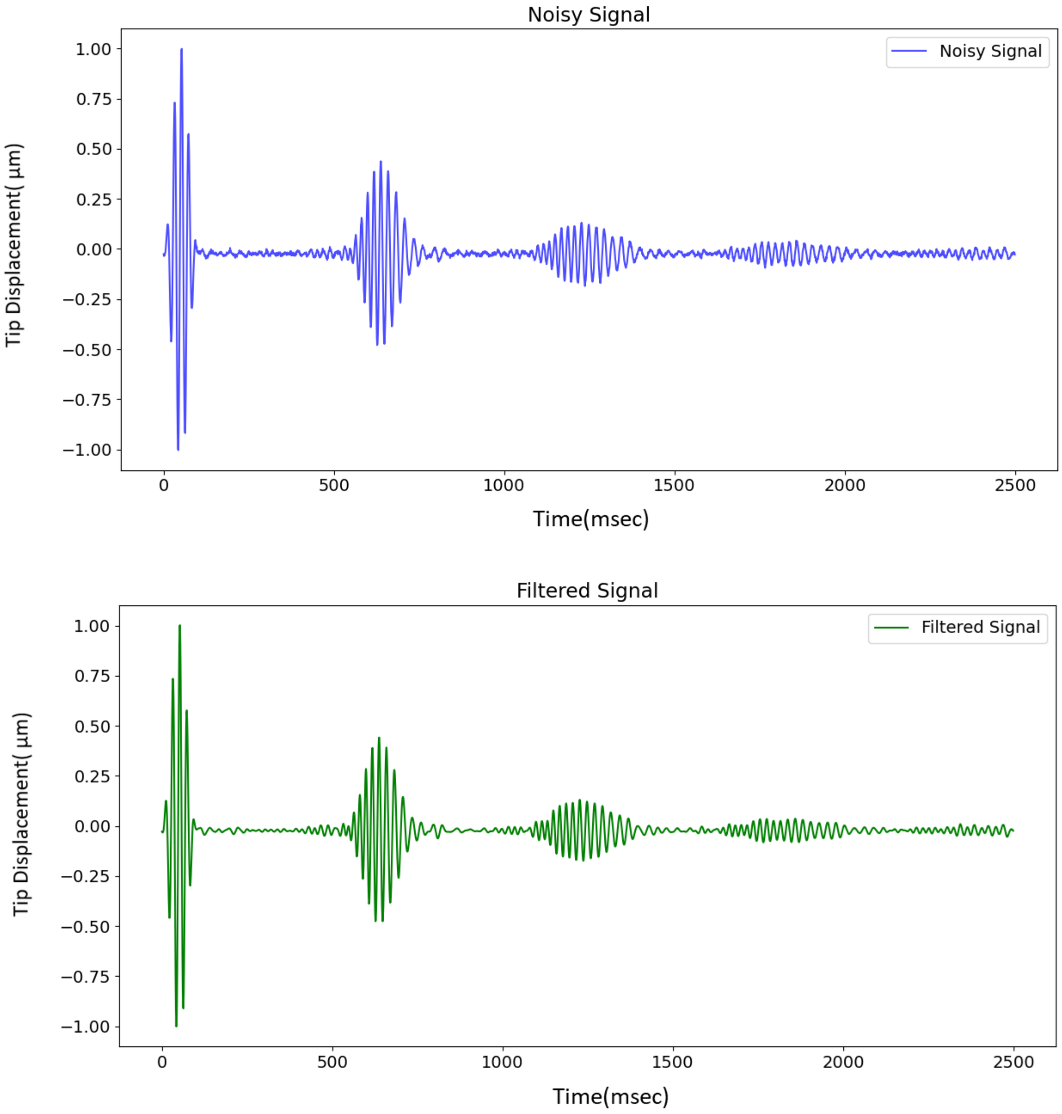

Figure 5.2: Noisy and reconstructed wave response using autoencoder.

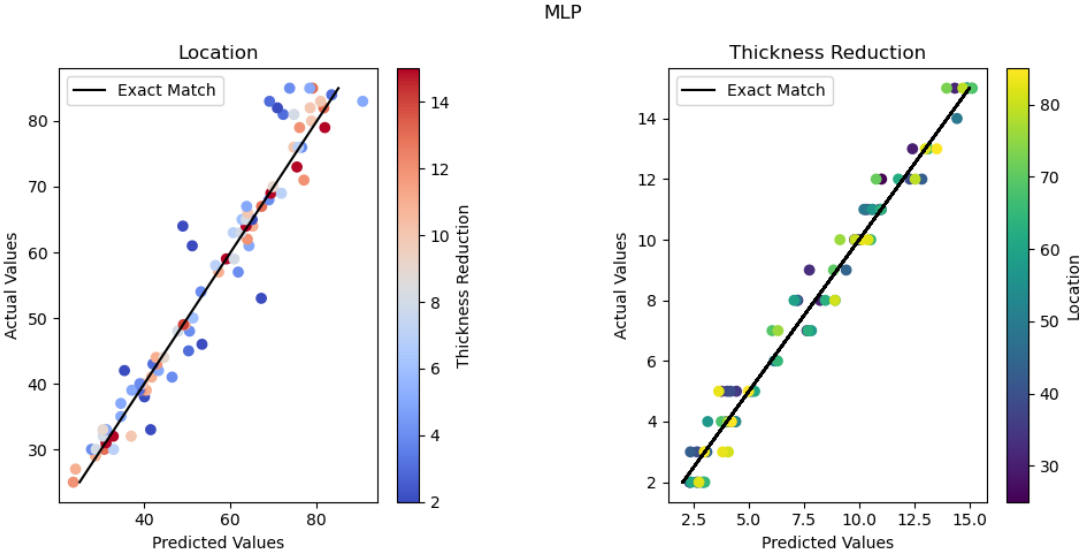

Figure 5.3: Predicted vs Actual values for MLP at 15% noise level.

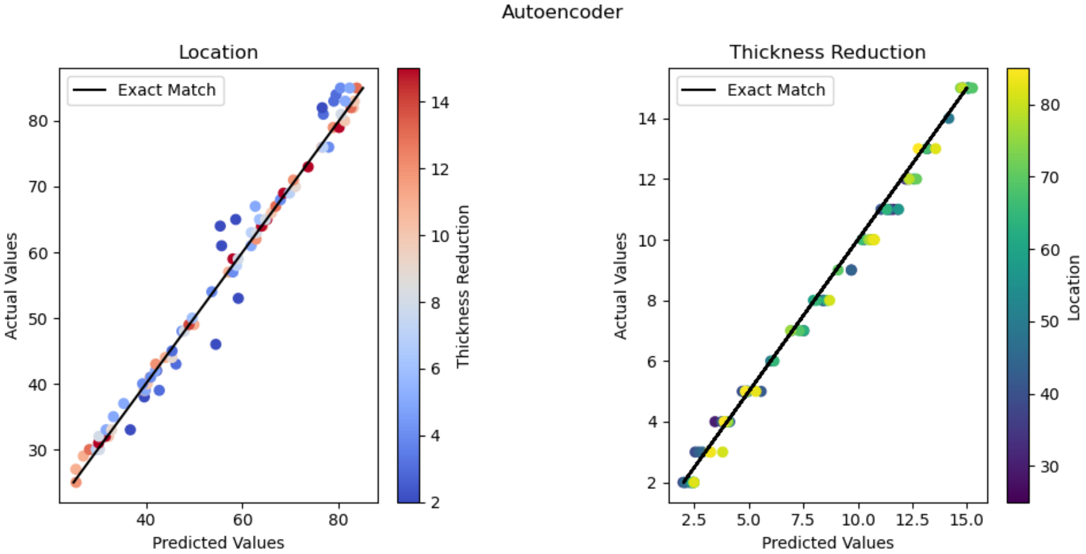

Figure 5.4: Predicted vs Actual values for Autoencoder at 15% noise level.

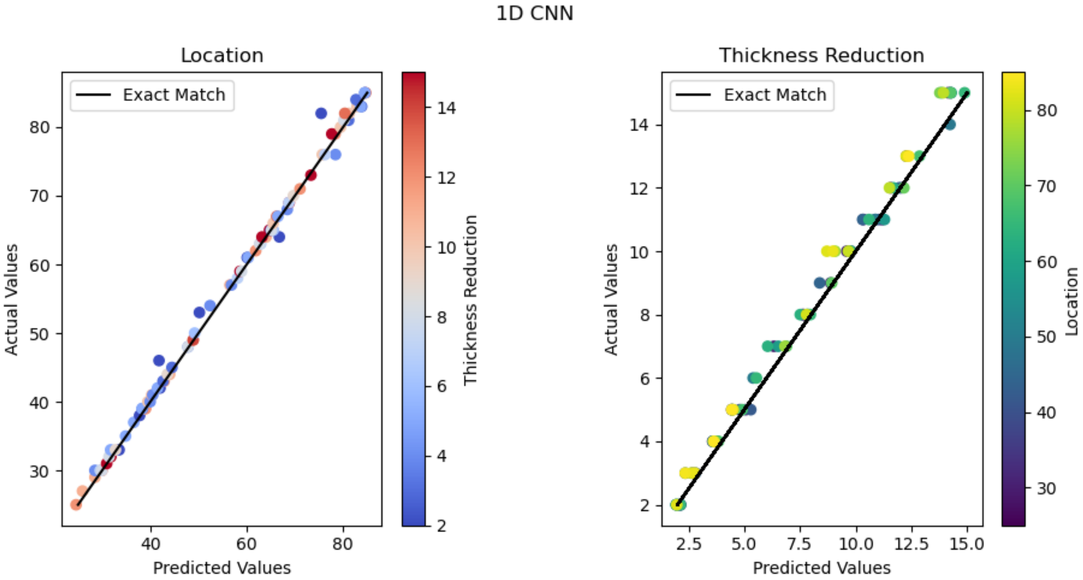

Figure 5.5: Predicted vs Actual values for 1D CNN at 15% noise level.

Chapter 6. Conclusion and Future Work

6.1 Conclusion

This research investigated deep learning architectures, including MLP, 1D CNN, and autoencoders for the SHM application using guided Lamb wave responses. The models demonstrated robust and accurate damage detection, even in the presence of significant noise levels. They successfully extracted critical information regarding damage location and extent, achieving a strong correlation with the actual damage values. The key findings are summarized below:

- Frequency domain features were able to provide a mechanism for signal representation and noise filtering. The MLP model trained with these frequency domain features achieved an R^2 score of 0.9586 at the 15% noise level, as compared to an R^2 score of 0.9152 when using raw time domain features, demonstrating better performance in handling increased noise levels.

- The autoencoder improved the performance and used only 50 dimensional latent vector for data representation which is 63% less than the 135 dimensional features extracted from frequency domain transformation. Mean percentage errors were within 2.7% for localization and 5.5% for thickness reduction at the maximum noise level.

- The 1D CNN model achieved the best performance, demonstrating robustness to noise. It achieved R^2 scores ranging from 0.9964 to 0.9908, with mean percentage errors remaining within 1.2% for damage localization and 5.7% for thickness reduction prediction, even at the highest noise level of 15

These results suggest that end-to-end deep learning models, particularly the 1D CNN and autoencoder, can effectively process noisy guided Lamb wave responses for SHM tasks without relying on manual feature engineering. The ability of these models to automatically learn robust representations and extract relevant information directly from the raw signal data enables accurate damage detection and quantification, even in the presence of noise. This paves the way for robust SHM methodology development using deep learning techniques.

6.2 Future Work

While this research demonstrated the effectiveness of deep learning for damage detection using simulated Lamb wave responses, there are many avenues for further exploration. One key challenge was predicting damage location for extremely low-level damage in the presence of high-intensity noise. Future studies could investigate advanced architectures like transformer-based models, which use attention mechanisms and could improve feature learning. This could lead to improved localization performance, particularly in scenarios with very small levels of damage. Similarly, hybrid approaches could also be explored. The autoencoder used fully connected layers for the encoder and decoder network. A CNN-based autoencoder architecture could be explore to enhance feature extraction, dimensionality reduction, and noise resilience.

Other research directions could focus on improving the robustness and generalizability of the models. The dataset could integrate diverse noise generation techniques and include real-world sensor data alongside simulated responses. Our research focused on only single damage scenarios. So another possibility is to augment the dataset with multiple damage scenarios. This can improve models' ability to generalize to real-world conditions.

Future work could also focus on developing interpretable architectures. For example, the future could investigate the connection between the latent representation learned by the autoencoder and frequency domain features. Furthermore, explainable AI techniques, such as SHAP analysis could be employed to gain insights into the decision-making process of the models. This could provide more transparency and foster trust in the SHM system.

By pursuing these future research directions, we can further refine deep learning models for Lamb wave-based SHM, overcoming current limitations and achieving greater accuracy, robustness, and interpretability. This will lead to more reliable and effective SHM systems, assuring the safety and integrity of infrastructure in real-world applications.

References

[1] H Amini Tehrani, Ali Bakhshi, and Moustafa Akhavat. An effective approach to structural damage localization in flexural members based on generalized s-transform. *Scientia Iranica*, 26(6):3125–3139, 2019.

[2] Onur Avcı, Osama Abdeljaber, Mustafa Serkan Kiranyaz, Boualem Boashash, Henry Angelo Sodano, and Daniel J. Inman. Efficiency validation of one dimensional convolutional neural networks for structural damage detection using a shm benchmark data. 2018.

[3] Onur Avcı, Osama Abdeljaber, Serkan Kiranyaz, and Daniel J. Inman. Convolutional neural networks for real-time and wireless damage detection. *Conference Proceedings of the Society for Experimental Mechanics Series*, 2019.

[4] Pierre Baldi. *Autoencoders*, page 71–98. Cambridge University Press, 2021.

[5] Vineetha Bettaiah and Heggere S Ranganath. An analysis of time series representation methods: data mining applications perspective. In *Proceedings of the 2014 ACM Southeast Regional Conference*, pages 1–6, 2014.

[6] Prabhav Borate, Gang Wang, and Yi Wang. Data-driven structural health monitoring approach using guided lamb wave responses. *Journal of Aerospace Engineering*, 33(4):04020033, 2020.

[7] Thanh Bui-Tien, Dung Bui-Ngoc, Hieu-Huy Nguyen-Tran, Lan Nguyen-Ngoc, Hoa Tran-Ngoc, and Hung Tran-Viet. Damage detection in structural health monitoring using hybrid convolution neural network and recurrent neural network. *Frattura ed Integrità Strutturale*, 2021.

[8] R Dhanapal and D Bhanu. Electroencephalogram classification using various artificial neural networks. *J. Crit. Rev*, 7:891–894, 2020.

[9] Charles R Farrar, Scott W Doebling, and David A Nix. Vibration–based structural damage identification. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 359(1778):131–149, 2001.

[10] Charles R Farrar and Keith Worden. *Structural health monitoring: a machine learning perspective*. John Wiley & Sons, 2012.

[11] Victor Giurgiutiu. *Structural health monitoring: with piezoelectric wafer active sensors*. Elsevier, 2007.

[12] Yingying He, Hongyang Chen, Die Liu, and Likai Zhang. A framework of structural damage detection for civil structures using fast fourier transform and deep convolutional neural networks. *Applied Sciences*, 11(19):9345, 2021.

[13] SeyyedPooya HekmatiAthar, Mohammad Taheri, Jameson Secrist, and Hossein Taheri. Neural network for structural health monitoring with combined direct and indirect methods. *Journal of Applied Remote Sensing*, 14(1):014511–014511, 2020.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[15] Azadeh Noori Hoshyar, Bijan Samali, Ranjith Liyanapathirana, Afsaneh Nouri Houshyar, and Yang Yu. Structural damage detection and localization using a hybrid method and artificial intelligence techniques. *Structural Health Monitoring*, 19(5):1507–1523, 2020.

[16] Nikhil Ketkar. *Recurrent Neural Networks*, pages 79–96. Apress, Berkeley, CA, 2017.

[17] Serkan Kiranyaz, Turker Ince, and Moncef Gabbouj. Real-time patient-specific ecg classification by 1-d convolutional neural networks. *IEEE Transactions on Biomedical Engineering*, 63(3):664–675, 2015.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[19] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[20] Usik Lee. *Spectral element method in structural dynamics*. John Wiley & Sons, 2009.

[21] Zhiwei Lin, Yonggui Liu, and Linren Zhou. Damage detection in a benchmark structure using long short-term memory networks. In *2019 Chinese Automation Congress (CAC)*, pages 2300–2305. IEEE, 2019.

[22] Arman Malekloo, Ekin Ozer, Mohammad AlHamaydeh, and Mark Girolami. Machine learning and structural health monitoring overview with emerging technology and high-dimensional data source highlights. *Structural Health Monitoring*, 21(4):1906–1955, 2022.

[23] Marvin Minsky and Seymour A Papert. *Perceptrons, reissue of the 1988 expanded edition with a new foreword by Léon Bottou: an introduction to computational geometry*. MIT press, 2017.

[24] Tanmai Sree Musalamadugu and Hemachandran Kannan. Generative ai for medical imaging analysis and applications. *Future Medicine AI*, (0):FMAI5, 2023.

[25] Thanh Q Nguyen. A data-driven approach to structural health monitoring of bridge structures based on the discrete model and fft-deep learning. *Journal of Vibration Engineering & Technologies*, 9(8):1959–1981, 2021.

[26] Joy Pal, Shirsendu Sikdar, and Sauvik Banerjee. A deep-learning approach for health monitoring of a steel frame structure with bolted connections. *Structural Control and Health Monitoring*, 29(2):e2873, 2022.

[27] Chathurdara Sri Nadith Pathirage, Jun Li, Lingjun Li, Hong Hao, and Wanquan Liu. Application of deep autoencoder model for structural condition monitoring. *Journal of Systems Engineering and Electronics*, 29:873–880, 2018.

[28] Liv Rittmeier, Natalie Rauter, Andrey Mikhaylenko, Rolf Lammering, and Michael Sinapius. The guided ultrasonic wave oscillation phase relation between the surfaces of plate-like structures of different material settings. In *Acoustics*, volume 5, pages 136–164. MDPI, 2023.

[29] Luca Rosafalco, Andrea Manzoni, Stefano Mariani, and Alberto Corigliano. Fully convolutional networks for structural health monitoring through multivariate time series classification. *Advanced Modeling and Simulation in Engineering Sciences*, 7, 2020.

[30] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[31] Laurie A. Schintler. High dimensional data. *Encyclopedia of Big Data*, 2021.

[32] Alex Shenfield and Martin Howarth. A novel deep learning model for the detection and identification of rolling element-bearing faults. *Sensors*, 20(18):5112, 2020.

[33] Sandeep Sony, Sunanda Gamage, Ayan Sadhu, and Jagath Samarabandu. Vibration-based multiclass damage detection and localization using long short-term memory networks. In *Structures*, volume 35, pages 436–451. Elsevier, 2022.

[34] Gang Wang. Analysis of bimorph piezoelectric beam energy harvesters using timoshenko and euler–bernoulli beam theory. *Journal of Intelligent Material Systems and Structures*, 24(2):226–239, 2013.

[35] Ruhua Wang, Chencho, Senjian An, Jun Li, Ling Li, Hong Hao, and Wanquan Liu. Deep residual network framework for structural health monitoring. *Structural Health Monitoring*, 20(4):1443–1461, 2021.

[36] SM Warren and P Walter. A logical calculus of the ideas immanent in nervous activity. 1943. *Bull. Math. Biol*, 52:99–115, 1990.

[37] Qingchen Zhang, Laurence T Yang, Zhikui Chen, and Peng Li. A survey on deep learning for big data. *Information Fusion*, 42:146–157, 2018.

[38] Youqi Zhang, Yasunori Miyamori, Shuichi Mikami, and Takehiko Saito. Vibration-based structural state identification by a 1-dimensional convolutional neural network. *Computer-Aided Civil and Infrastructure Engineering*, 34(9):822–839, 2019.

Appendix A. Training LSTM

This thesis also explored the possibility of using LSTM networks because of their capability to capture temporal dependencies within sequential data. However, the training results for LSTM were unsatisfactory. LSTM models with different parameters such as several hidden layers and hidden unit sizes were trained and tested.

For the LSTM network with two hidden layers with 512 hidden units which was followed by fully connected layers with sizes 256 and 128, it only achieved a low R^2 score of 0.0017. This is equivalent to predicting the average values.

Several factors could have resulted in the poor performance of the LSTM model:

- Limited temporal dependence: The wave response for different damage scenarios varies by small differences in the notch response. This might not have contained significant temporal correlations. So, the model only learns to predict the mean value of different damage scenarios. Exploring different types of time-series preprocessing techniques could potentially extract significant temporal features for the LSTM.

- Small Data size: The LSTM network uses different gating mechanisms due to which it has a complex architecture. This means it usually requires a substantial amount of training data to generalize well. However, this research used a relatively small dataset and as such there are limitations in performance. Further data augmentation and increasing datasets could enhance the performance of LSTM.

- Further Hyperparameter tuning: The selected hyperparameters, such as the number of layers, hidden unit sizes, activation functions, learning rate, and training epoch, might not be sufficient for training the LSTM to the dataset. Applying further hyperparameter tuning, utilizing bi-directional LSTM with attention, and using better optimization algorithms could potentially lead to better results.

Despite the poor performance results of LSTM in this study, they are still a viable option for SHM applications. LSTM can learn temporal dependence in the time-series data which can be applied to complex damage scenarios. By conducting further study on the above-mentioned areas, it could lead to the effective application of LSTMs for Lamb wave-based SHM.