EE499 Major Project Report on

A STUDY ON THE PERFORMANCE OF AUTO-ENCODER BASED DENOISING TECHNIQUE

Submitted in partial fulfillment of the requirements for the degree of

BACHELOR OF TECHNOLOGY

in

ELECTRICAL & ELECTRONICS ENGINEERING

by

Harshavardhan D - 191EE123

J Karan Tejas - 191EE126

Shashank Shekhar Panda - 191EE248

Aditya Devanand Kaloji- 191EE102

under the guidance of

Dr. Shubhanga K N

DEPARTMENT OF ELECTRICAL & ELECTRONICS ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE - 575025

DECLARATION

By the B. Tech Undergraduate

We hereby *declare* that the Major Project entitled ***"A Study on the Performance of Auto-encoder based Denoising Technique"*** , which is being submitted to the **National Institute of Technology Karnataka, Surathkal**, for the award of the Degree of Bachelor of Technology in Electrical & Electronics Engineering, is a *bonafide report of the work carried out by me/us.* The material contained in this Major Project Report has not been submitted to any University or Institution for the award of any degree.

Harshavardhan D
(Reg. No.: 191E123)
Dept. of Electrical & Electronics Engineering

Shashank Shekhar Panda
(Reg. No.: 191EE248)
Dept. of Electrical & Electronics Engineering

J Karan Tejas
(Reg. No.: 191EE126)
Dept. of Electrical & Electronics Engineering

Aditya Devanand Kaloji
(Reg. No.: 191EE102)
Dept. of Electrical & Electronics Engineering

Department of Electrical & Electronics Engineering
Place : NITK, Surathkal
Date :

CERTIFICATE

This is to *certify* that the EE499 Major Project Work Report entitled ***"A Study on the Performance of Auto-encoder based Denoising Technique"*** submitted by **Harshavardhan D (Reg. No.: 191EE123), Shashank Shekhar Panda (Reg. No.: 191EE248), J Karan Tejas (Reg. No.: 191EE126) and Aditya Devanand Kaloji (Reg. No.: 191EE102)** as the record of the Major Project work carried out by them, is *accepted as the B.Tech. EE499 Major Project submission* in partial fulfillment of the requirement for the award of degree of **Bachelor of Technology in Electrical & Electronics Engineering in the Department of Electrical & Electronics Engineering, NITK Surathkal Karnataka**.

Research Guide
Dr. Shubhanga K N
Professor
Dept. of Electrical & Electronics Engineering
NITK Surathkal

Head Of Department
Dr. Dattatraya Narayan Gaonkar
Associate Professor and Head of the Department
Dept. of Electrical & Electronics Engineering
NITK Surathkal

ACKNOWLEDGEMENT

We would like to articulate our deep gratitude to our project guide Dr. Shubhanga K N, Professor, Department of Electrical and Electronics Engineering, National Institute of Technology Karnataka, Surathkal, who has always been a source of motivation for carrying out this project. His constant inspiration and ideas have helped us in shaping this project. We are thankful to him for giving his valuable time despite his busy schedule for completion of our project.

A word of thanks goes to the Head of Department Dr. Dattatraya Narayan Gaonkar, for allowing us to use department facilities and gather the literature for carrying out this project, along with all the Faculty, Staffs and Students of Department of Electrical and Electronics Engineering who have helped us in carrying out our work.

We would like to express our gratitude to all those authors whom we have mentioned in the reference section for giving shape to our thoughts through their path breaking endeavours. Last but not the least we would like to thank our friends and family for supporting us in every possible way while carrying out this project work.

Harshavardhan D
(Reg. No.: 191E123)
Dept. of Electrical & Electronics Engineering

Shashank Shekhar Panda
(Reg. No.: 191EE248)
Dept. of Electrical & Electronics Engineering

J Karan Tejas
(Reg. No.: 191EE126)
Dept. of Electrical & Electronics Engineering

Aditya Devanand Kaloji
(Reg. No.: 191EE102)
Dept. of Electrical & Electronics Engineering

ABSTRACT

Signal noise is unwanted interference that degrades the quality of signals. The most frequent and evident issue brought on by signal noise is the distorted interpretation or erroneous display of a process condition by the equipment.

Extreme signal noise, however rare, can cause an apparent signal loss. The majority of contemporary electronic devices have noise filtering built in. This filter won't be enough in excessively noisy surroundings, which can result in the equipment not receiving a signal and no communication at all.

A system that experiences signal noise variations may unintentionally misinterpret the noisy signals, causing relays and alarms to turn on and off at random intervals. An industrial process is improperly controlled in a circumstance like this.

As there are multiple denoising techniques present out there including both traditional methods and learning based methods this project aims to find out the best suitable algorithm given a case.

In this project we are trying to compare different types of denoising techniques by adding white gaussian noise to a clean signal. The parameter on which different denoising techniques are going to be compared is SNR(Signal to Noise Ratio).

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 What is Noise?

The signal is the meaningful information that you're actually trying to detect. The fluctuation or variation that is random, undesirable, which interferes with the original message signal and corrupts the parameters of the message signal is called noise. It is most likely to be entered at the channel or the receiver. Imagine talking on the phone to a friend who is out in the traffic or sitting in a park; aside from your friend's voice (which is regarded as the pure signal in this case), you'll hear a lot of other sounds, such as the sound of vehicles honking, people talking, the traffic police whistling or the birds chirping etc. These sounds can all be categorised as noise the presence of which in a signal can make it difficult to analyze and extract meaningful information from it.

There are two main ways in which noise appears in any system. One is through some external source while the other is created by an internal source, within the receiver section.

- **External Source**: This noise is typically caused by outside factors that could affect the communication channel or medium. It is impossible to stop this noise entirely. Preventing noise from influencing the signal is the best course of action. The most prevalent examples of this form of noise are:
  - Atmospheric noise (caused due to irregularities in the atmosphere)
  - Extra-terrestrial noise, such as Cosmic and Solar noise
  - Industrial noise
- **Internal Source**: When the receiver is operating, its components make this noise. Due to their continual operation, circuit components may emit a variety of noises. This noise can be measured. The impact of this internal noise may be lessened with an appropriate receiver design. The most common examples of this type of noise are:
  - Thermal agitation noise (Johnson noise or Electrical noise)
  - Shot noise (due to the random movement of electrons and holes)
  - Transit-time noise (during transition)

– Miscellaneous noise is another type of noise which includes flicker, resistance effect and mixer generated noise, etc.

1.2 The Need for Noise Removal

Noise is an inconvenient feature which affects the system performance. Following are the effects of noise. The weakest signal that an amplifier can amplify is indirectly limited by noise. Due to noise, the oscillator in the mixing circuit could have a frequency limit. The functionality of a system is dependent on how well its circuits work. The smallest signal that a receiver can process is constrained by noise. Sensitivity is the smallest quantity of input signal required to produce an output with the desired quality. A receiver system's sensitivity is impacted by noise, which eventually has an impact on the output.

1.3 Prior Work

Numerous researchers have utilised a variety of noise removal strategies. We compared the most effective traditional filtering methods and noise-removing algorithms in order to totally remove noise from signals and achieve successful results without compromising the integrity of the signal.By traditional we imply using mathematical models and statistical methods to remove noise patterns. Among the many approaches to denoising a signal, we decided to examine and analyse the wavelet and EMD methods.

1.3.1 Wavelet based denoising

Wavelet analysis divides signals into scaled and shifted versions of a wavelet, as opposed to Fourier analysis, which separates data into sine waves with predetermined frequencies. A wavelet is a rapidly fading oscillation that resembles a wave, unlike a sine wave. As a result, wavelets may now represent data at many scales. Wavelet transform method can be employed for analysis of the signal with respect to approximation and detailed coefficients. Here we utilized the discrete wavelet transform technique as it is more suitable for denoising of audio signals.

The signal which is to be analyzed is taken as input, to which *white gaussian noise* is added and the resulting noisy signal is obtained. White gaussian noise is preferred as it has almost constant PSD (power spectral density) for easy and precise analysis.The noisy signal is then decomposed into two parts, namely detailed coefficients and approximation coefficients.

The number of levels required for decomposition generally depends upon the nature of the signal. Multilevel decomposition is done to repeat the decomposition process so that many lower-resolution signal components can be obtained through wavelet decomposition trees. Wavelet thresholding, the final step, is to reconstruct the original audio signal without much loss of information. In a construction process that involves using the wavelet coefficients and considering the levels of iteration, successful reconstruction of the original audio signal is obtained. The wavelet transform can be summarized as seen in Figure 1.4.1

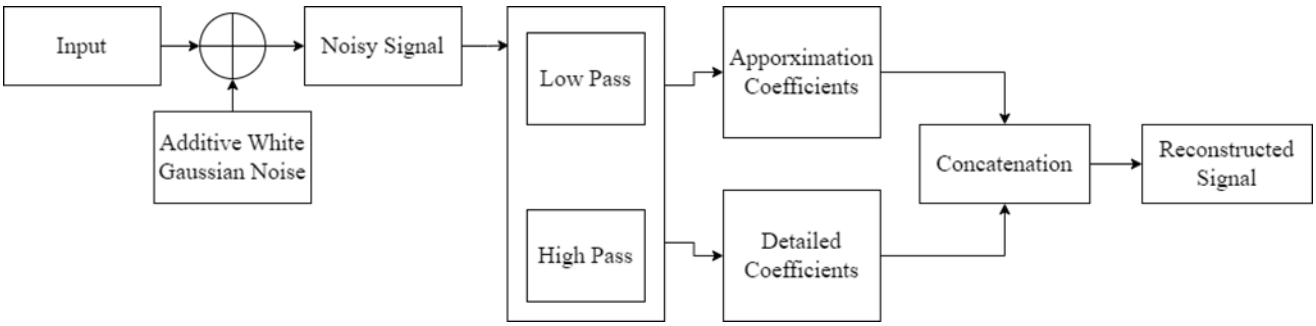

Figure 1.3.1: Wavelet Transform flow diagram

Different wavelets can be used, depending on the application. The wavelets we used are *Daubechies* (db4) wavelets with the highest number N of vanishing moments with the support width 2N-1. dB wave solves the problem of signal discontinuities and is applicable for continuous and discrete wavelet transforms.

1.3.2 Empirical Mode Decomposition (EMD)

A procedure called the empirical mode decomposition (EMD) approach divides multicomponent signals into a set of amplitude and frequency-modulated (AM/FM) zero-mean signals, also known as intrinsic mode functions (IMFs). EMD represents the signal as an expansion of signal-dependent basis functions estimated using an iterative process called sifting. This contrasts conventional decomposition techniques like wavelets, which perform the analysis by projecting the signal under consideration onto a number of predefined basis vectors. EMD-based denoising is particularly useful for removing noise from non-stationary signals, such as biological signals, acoustic signals, and financial signals. It has been applied in a wide range of applications, including speech processing, vibration analysis, and financial time series analysis.

EMD decomposes a signal into a series of intrinsic mode functions (IMFs), which are defined as smoothly varying, amplitude-modulated, narrowband components of the signal.

The basic steps involved in EMD-based denoising are as follows:

1. **Decomposition of the signal**: The noisy signal is decomposed into a set of IMFs using the EMD algorithm. The EMD algorithm decomposes the signal into a series of oscillatory components by iteratively extracting the IMF with the highest frequency content and subtracting it from the original signal.

2. **Selection of relevant IMFs**: The IMFs with the highest frequency content, which are likely to contain the noise, are identified based on their frequency spectra. In our case we utilize a metric called *Spectral Flatness*. By comparing the Spectral Flatness (SF) of each IMF to a threshold T, it is possible to determine the number of noisy IMFs, n. The spectral flatness of a signal is given by the geometric mean of the power spectrum divided by its arithmetic mean as seen in equation below.

$$\text{Spectral Flatness} = \frac{\sqrt[N]{\prod_{n=0}^{N-1} x(n)}}{\frac{1}{N}\sum_{n=0}^{N-1} x(n)} \tag{1.1}$$

   In this equation, $x(n)$ represents the signal sample at time index n, N is the total number of samples in the signal.

3. **Filtering of noisy IMFs**: The noisy IMFs that do not satisfy the threshold (spectral flatness criterion)are filtered using a suitable filter to remove the noise while preserving the relevant signal components. In our case, we used a lowpass butterworth filter of order 5 with a cut-off frequency of suitable value obtained through ad-hoc analysis of the input signal.

4. **Reconstruction of denoised signal**: The denoised signal is reconstructed by summing the filtered IMFs along with a residual signal obtained by subtracting the sum of the filtered IMFs from the original signal.

The above algorithm can be summarized in the form of a flow as seen in Figure 1.4.2.

The effectiveness of EMD-based denoising depends on the quality of the IMFs obtained by the EMD algorithm and the choice of the filter used to remove the noise. The denoising performance can be evaluated using metrics such as signal-to-noise ratio (SNR).

We implemented the above algorithms on MATLAB for four different input signals:

1. Frequency readings from a power system taken at 30 samples/second for 180 seconds (large variations present locally)

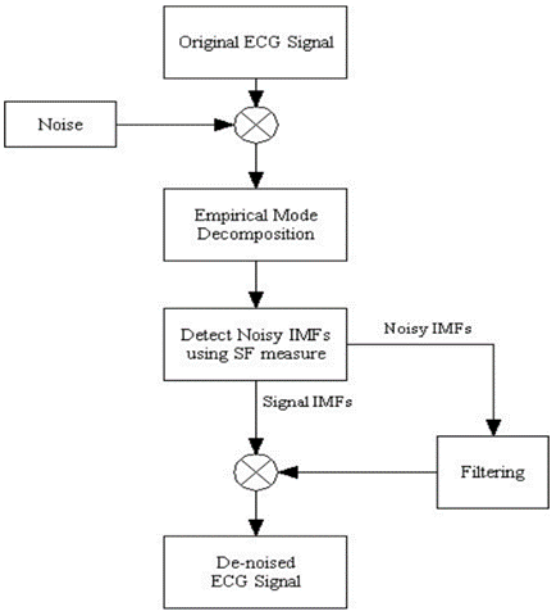

Figure 1.3.2: Flowchart depicting EMD filter based denoising algorithm

2. Sinewave of frequency 60Hz

3. Random audio signal

4. Frequency readings from a power system taken at 30 samples/second for 180 seconds (variations present throughout)

We then, compared the original signal, the signal after adding white gaussian noise (SNR = 15dB), and the denoised output signal in the form of a table shown below.

	EMD-Based		Wavelet-Based	
Input Signal	Noisy Signal SNR	Denoised Signal SNR	Noisy Signal SNR	Denoised Signal SNR
Input-1	15	18.3182	15	20.8425
Input-2	15	24.8620	15	27.8395
Input-3	15	18.2944	15	20.5533
Input-4	15	20.0259	15	22.8226

Table 1.3.1: Input and Output SNR values for Wavelet & EMD based denoising methods for different input signals

The above table illustrates how the each of the algorithms perform for varying input signals for an input SNR of 15 dB. In case of the wavelet approach we notice that it performs considerably better than its EMD counterpart for this particular input SNR. However, it was observed that the wavelet-based approach doesn't should much improvement; rather, the

quality deteriorates as and when we increase the input SNR values further, as seen in Figure 1.4.2. Furthermore, in the case of the EMD-based denoising approach, we can observe that although it performs comparatively worse than the wavelet at lower input SNR values, it is able to maintain signal integrity at higher SNR values, unlike the wavelet.

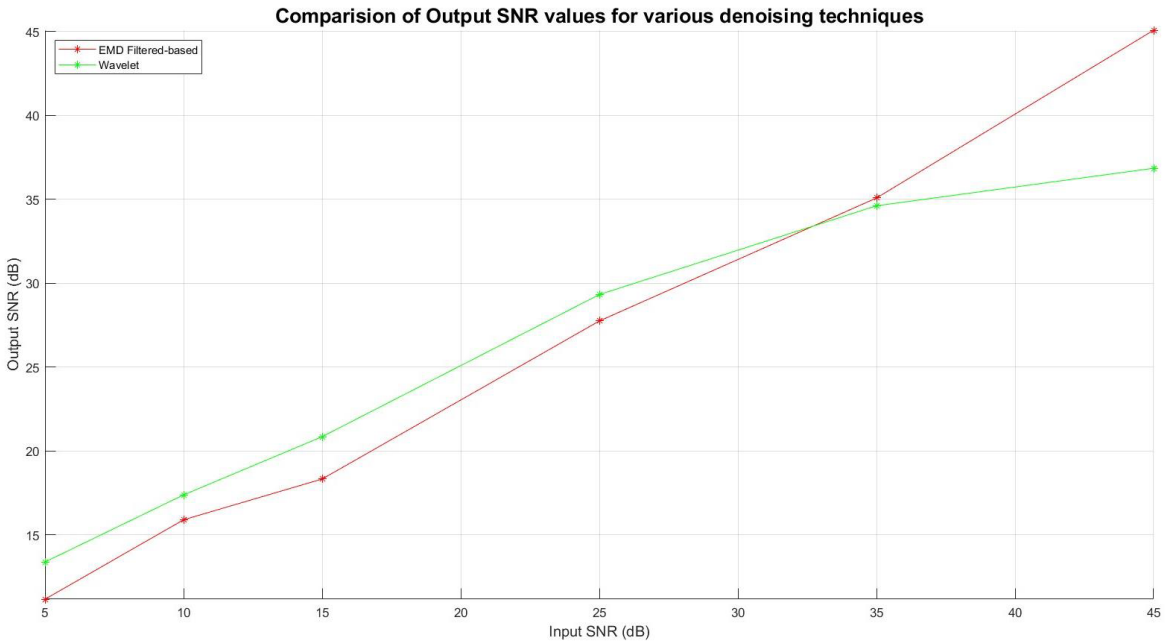

Figure 1.3.3: Comparison of Output SNR values for varying Input SNR values for different denoising methods

1.4 Why ML-Based denoising methods?

Owing to all these noises the need for techniques to denoise signals so they could be interpreted the way they were meant to be becomes inevitable. The process of reducing noise from a signal is known as noise reduction or denoising. Both audio and picture noise reduction methods exist. Algorithms for noise reduction may slightly skew the signal. As with common-mode rejection ratio, noise rejection refers to a circuit's capacity to separate an undesirable signal component from the desired signal component. Numerous researchers have utilised a variety of noise removal strategies recently. We compared the most effective filtering methods and noise-removing algorithms in order to totally remove noise from signals and achieve successful results without compromising the integrity of the signal.

One of the most effective ways to remove noise from signals is by using machine learning

techniques. Machine learning algorithms can learn from the patterns in the signal data and can be trained to identify the noise components and remove them from the signal. Here are some reasons why machine learning techniques are useful for signal denoising:

- **Non-linear relationships**: In many cases, the relationship between the signal and the noise is non-linear, making it challenging to use traditional methods for noise removal. Machine learning techniques can capture non-linear relationships between the signal and the noise, allowing for more accurate noise removal.
- **Adaptive methods**: Machine learning techniques can adapt to changing noise patterns and adjust their denoising strategy accordingly. This is especially useful when dealing with non-stationary signals where the noise characteristics can change over time.
- **Customizable models**: Machine learning models can be customized to specific signal types and noise characteristics. This allows for more accurate and effective noise removal than traditional methods that use fixed filters.
- **Large datasets**: Machine learning models require large datasets to train and optimize their parameters. Signal processing applications typically have large datasets, making them well-suited for machine learning algorithms.

There are several machine learning techniques that can be used for signal denoising, including neural networks, support vector machines, and decision trees. These techniques have been used successfully in various signal processing applications, such as speech recognition, image processing, and medical signal analysis.

In conclusion, machine learning techniques have proven to be effective for removing noise from signals. Their ability to capture non-linear relationships, adapt to changing noise patterns, and customize models to specific signal types and noise characteristics make them a powerful tool for signal denoising. As machine learning continues to advance, we can expect even more sophisticated techniques for signal processing and denoising to emerge.

CHAPTER 2

LITERATURE REVIEW

2.1 Background and Related Works

Neural networks and deep neural networks are becoming more and more well-liked in the modern era as intelligent solutions to practically any computing challenge. Predicting results and classifying objects are two of those. The idea behind an autoencoder helps improve degraded images by reducing Gaussian noise. An artificial neural network called an autoencoder seeks to learn an encoding (representation) for a set of data. The autoencoder is what gathers noise from practice photos and then tries to produce a clean image that closely resembles the input. This proposed model, which is based on an autoencoder, uses deconvolution, convolutional, and convolutional layers to reduce noise. The performance of this suggested model is finally evaluated using the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM).[1]

Autoencoder-based audio denoising is a technique used to remove noise from audio signals by training an autoencoder neural network to learn the underlying structure of the audio data and then use this information to remove the noise. The architecture of the autoencoder used for audio denoising may differ from that used for image denoising due to the nature of the data. Audio signals are typically one-dimensional, and their structure is determined by the frequency components of the signal. Therefore, the autoencoder may use a 1D convolutional neural network (CNN) architecture to extract the relevant features from the audio signals. This paper describes a method for creating an autoencoder to map noisy machine sounds to clean sounds for denoising purposes.

In the scope of [2], the removal of generated noise with Gaussian distribution and the environmental noise with a specific example of the water sink faucet noise from the induction motor sounds has been demonstrated. The suggested autoencoder's denoised sounds and the test set's original sounds were compared using the mean square error (MSE) as the assessment criterion.

A novel Denoising Auto-encoder with a Multi-branched Encoder (DAEME model) is proposed in another study [3] . This autoencoder builds a multi-branched encoder using various component models that is based on a Dynamically-Sized Decision Tree (DSDT). The multi-branched encoder conducts a specific mapping from noisy to clean speech along

each branch in the DSDT based on past knowledge of speech and noisy settings (the speaker, environment, and signal components are taken into account in this research). The multi-branched encoder is then used to train a decoder.

Yet another study offers a new denoiser [4], a noise learning-based DAE, which modifies the structure of the denoising autoencoder (DAE) (nlDAE). The suggested nlDAE discovers the the input data's noise. The denoising process is then carried out by removing the created noise from the noisy input. So, when the noise is easier to regenerate than the original data, nlDAE is more effective than DAE. We offer three case studies—signal restoration, symbol demodulation, and exact localization—to demonstrate the effectiveness of nlDAE. According to numerical findings, nlDAE requires a smaller training dataset and latent space dimension than DAE.

BART is a denoising autoencoder for sequence-to-sequence model pretraining. In order to train BART, text is first corrupted using a random noise function, and then a model is learned to recreate the original text. It employs a typical Tranformer-based neural machine translation architecture that, despite its simplicity, generalises several other more modern pretraining approaches, including GPT with its left-to-right decoder and BERT (owing to the bidirectional encoder). The best result was obtained by randomly rearranging the original sentences' sequence as well as by utilising a cutting-edge in-filling strategy in which long stretches of text are substituted with a single mask token. BART performs admirably for comprehension tasks but is most successful when tailored for text production. It produces new state-of-the-art results on a range of abstractive dialogue, question answering, and summarization tasks, with gains of up to 6 ROUGE, and matches the performance of RoBERTa with equivalent training resources on GLUE and SQuAD. With just target language pretraining, BART also offers a 1.1 BLEU improvement over a back-translation approach for machine translation. Additionally, ablation experiments that mimic different pretraining strategies within the BART framework have been published in order to more accurately assess the variables that have the greatest impact on end-task performance. [5]

Data can be efficiently represented and autoencoder can learn the data structure in a flexible way. These characteristics enable autoencoder to overcome high design costs and subpar generalisation in addition to well coping with vast volume and variety of data. However, using an autoencoder has issues with low resilience and overfitting. This study [6] investigates denoising sparse autoencoder, which involves supplementing classic autoencoder with a corrupting operation and a sparsity restriction in order to extract meaningful features, increase resilience, and avoid overfitting. The findings imply that the various autoencoders

discussed in this study have some close relationships, and the investigated model can extract intriguing features that can accurately reconstruct the original data. Also, these findings point to a promising strategy for constructing deep models using the suggested autoencoder.

2.2 Problem Statement

In the growing world of communication, signal transmission and processing have become integral. In this process of transmission and conversion of signal energy, the effect of noise and distortion is inevitable. Noise reduction and distortion removal have many applications in various fields such as radar, image processing, audio signal processing, sonar, etc. Consequently, finding methods to achieve greater accuracy in the received signals is essential.

2.3 Objectives of the Project

Analyze and compare different denoising techniques for 1D signals based on parameters such as their SNR ratios. Adding to it will be a detailed study of different adaptive filter and learning-based models that will present us with the apt algorithm given a noisy signal sample.

CHAPTER 3

PROPOSED WORK

3.1 Noise Theory

Signal fading, reverberations, echo, multipath reflections, and missing samples are examples of changes in a signal caused by the non-ideal features of the communication channel. The phrase "signal distortion" is frequently used to characterise a systematic undesired change in a signal.

3.1.1 Signal to Noise ratio(SNR)

To evaluate how noise affects a signal, people frequently utilise the signal-to- noise ratio (SNR). The quantized signal xq[n] is a superposition of the unquantized, undistorted signal x[n] and the additive quantization error e[n]. This measurement is based on an additive noise model. The SNR is determined by the ratio of the signal powers of x[n] and e[n]. SNR is typically expressed using a logarithmic scale, measured in decibels, to account for the vast range of possible SNR values and the human ear's logarithmic loudness perception (dB).

$$\text{SNR} = \frac{\sum_{n=0}^{N-1} |x[n]|^2}{\sum_{n=0}^{N-1} |e[n]|^2} \tag{3.1}$$

3.1.2 Additive white gaussian noise

Additive white Gaussian noise (AWGN) is a type of random signal that is commonly used in the field of signal processing and communication theory. It is a mathematical model used to represent the noise that is added to a signal during communication or signal processing.

The term "additive" refers to the fact that the noise is added to the original signal. "White" implies that the noise has a flat power spectral density over a wide frequency range. This means that the noise has equal power at all frequencies, which is why it is called "white". "Gaussian" refers to the fact that the noise is modeled using a Gaussian probability distribution. The Gaussian distribution is a common distribution for many types of random phenomena, and it is relatively easy to work with mathematically.

In communication theory, AWGN is often used as a model for the effects of noise in a communication channel. The presence of noise in a communication channel can degrade the

quality of the transmitted signal and make it harder to detect and decode. By understanding the properties of AWGN, communication engineers can design communication systems that are more robust and can operate effectively even in noisy environments.

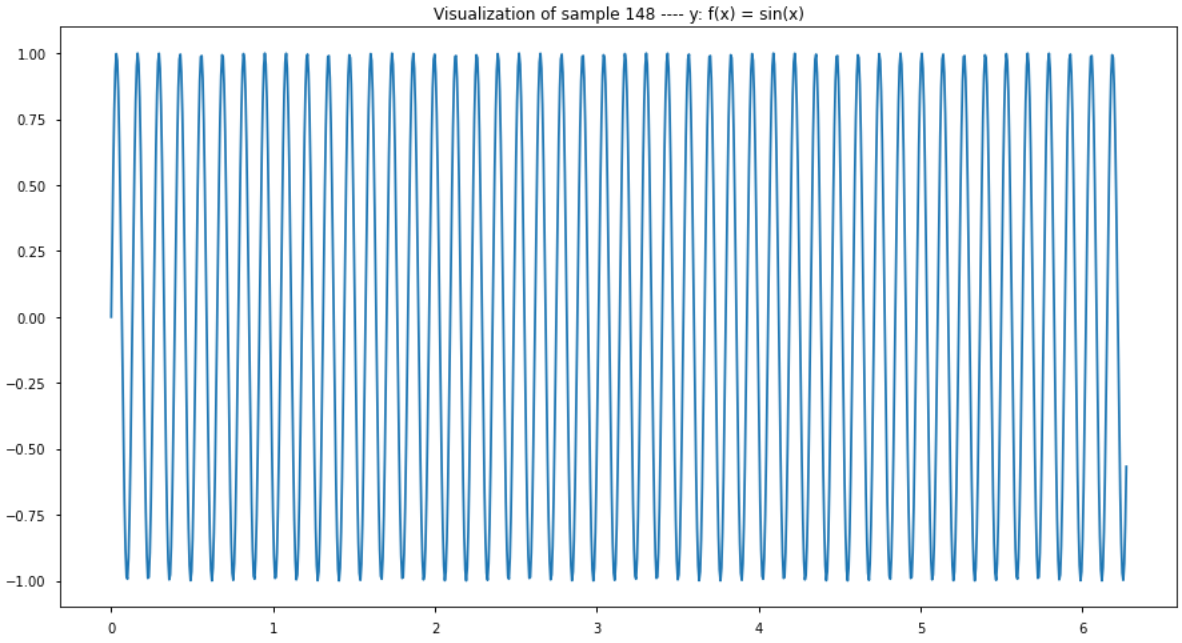

Figure 3.1.1: Visualization of a clean sample of sin(x)

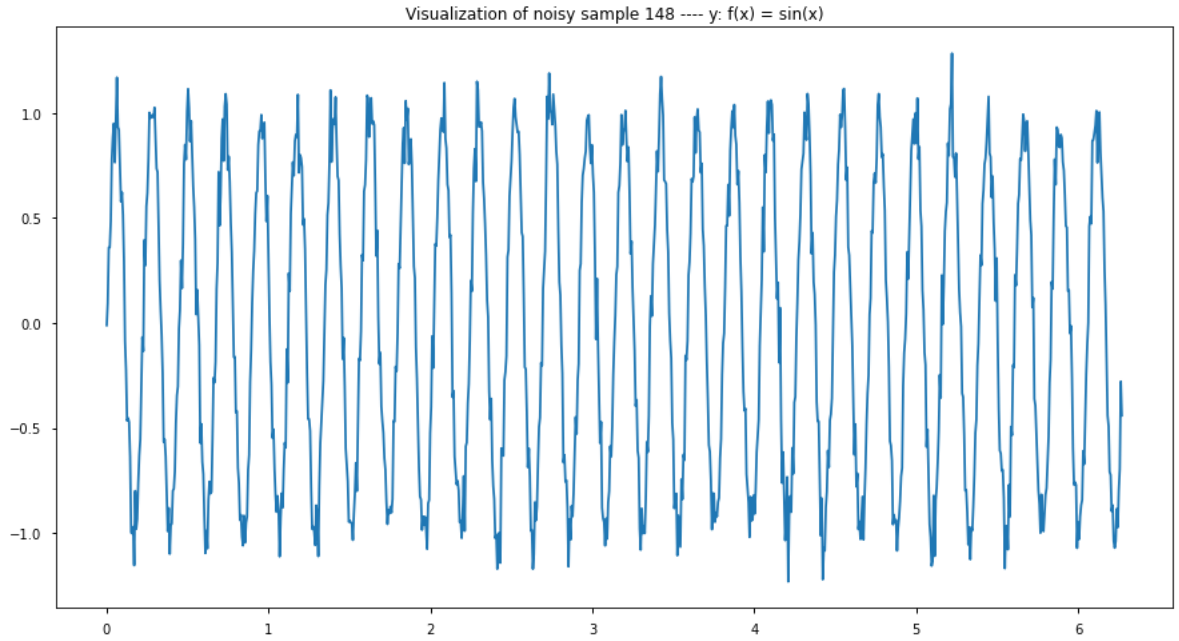

Figure 3.1.2: Visualization of a noisy sample of sin(x)

Figure 3.1.1 depicts a random sine wave before adding noise and the same sine wave after adding white gaussian noise of SNR 17.09 dB as seen Figure 3.1.2.

3.2 Autoencoders

3.2.1 What is an Autoencoder?

Autoencoder is a type of neural network that is designed to learn a compressed representation of input data by training the network to reconstruct its own inputs. In other words, the network learns to encode an input into a lower-dimensional representation (also called a bottleneck), and then decode it back into the original input. They are used for unsupervised learning tasks, such as image compression and denoising.

The autoencoder consists of two main components: an encoder and a decoder. The encoder takes an input, such as an image or signal, and maps it to a compressed representation, usually a vector of lower dimensions. The decoder then takes this compressed representation and reconstructs the original input.

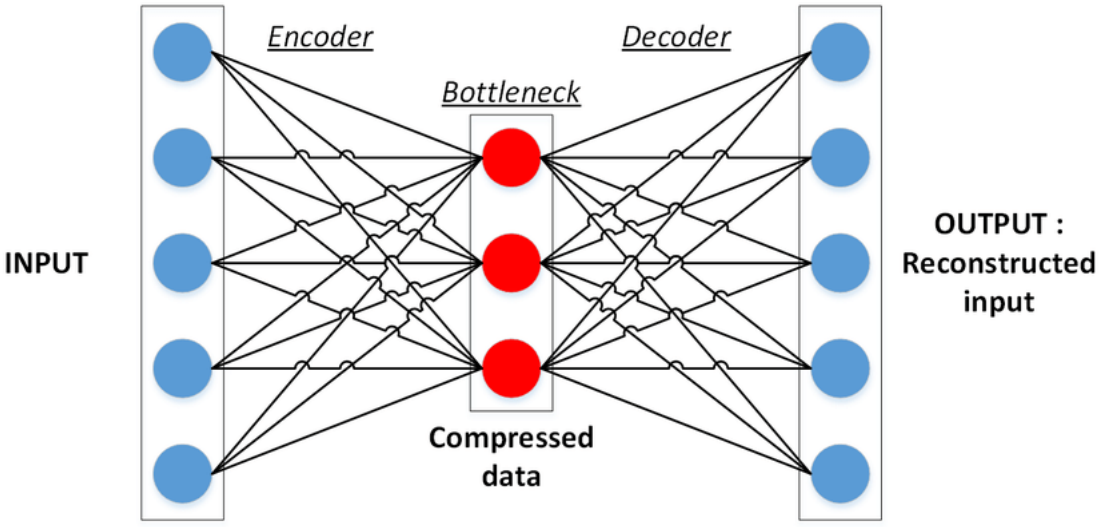

Figure 3.2.1: Basic architecture of a single layer autoencoder

As seen in Figure 3.1.1, the simplest type of autoencoder is a single-layer feedforward neural network with one hidden layer. The input is fed into the network and is compressed into a bottleneck layer with fewer neurons. The bottleneck layer is the layer with the smallest number of neurons in the network. The output of the bottleneck layer is then passed through the decoder, which generates the reconstructed input.

The loss function used to train the autoencoder is typically the *mean squared error* (as in our case as well) or *binary cross-entropy* between the original input and the reconstructed input. During training, the autoencoder learns to minimize this error by adjusting the weights and biases of the neural network. Once the autoencoder has been trained, the encoder can be used to transform new inputs into the compressed representation, and the decoder can be used to generate the reconstructed input.

In summary, autoencoders are a type of neural network that learns to compress input data into a lower-dimensional representation and then reconstructs the original input from this compressed representation. They can be used for unsupervised learning, data compression, data denoising, feature extraction, and data generation.

3.2.2 Why are autoencoders used for noise removal?

The issue of audio denoising is very common. The objective is to remove noise from the input signal while preventing a deterioration in signal quality. Since almost everyone has at some point in their lives failed to comprehend what the presenter is saying in the video due to all the background noises, it is not difficult to understand why the issue is of great importance. The denoising algorithm's main goal in these situations is to improve the spoken signal by reducing background noise.

Autoencoders can be taught to encode a state with an encoder and to decode that state into a different state with a decoder.

Now consider this in terms of signal noise. Assume that you feed the neural network noisy data as features while having access to pure data as targets. According to the illustration above, the neural network will learn an encoded state from the noisy image and make an attempt to decode it to the greatest extent possible. What is the barrier separating the clean data from the noisy data? The noise, for sure. As a result, the autoencoder will develop the ability to identify noise and eliminate it from the input signal.

The Autoencoder takes in the cluttered signal and attempts to produce a clear output of it. The model can really be expressed as a convolutional autoencoder known as a U-Net since it is based on symmetric encoder-decoder components of the basic architecture. Convolution, ReLU, and InstanceNormalization blocks are included in the decoder and encoder components. In addition, skip links between the encoder and decoder blocks are present in the model. The purpose of adding skip connections is to lessen gradient vanishing, similar to what occurs in ResNet. The mean squared error (MSE) between the original-clean audio and the output is being optimised after the input has been fed to the network and the outputs

have been generated.

3.2.3 The Model

Now, the model used for training was built as seen in Figure 3.1.2 and has the following layers:

- The input layer, which takes the input data;
- Two Conv1D layers, which serve as encoder;
- Two Conv1D transpose layers, which serve as decoder;
- One Conv1D layer with one output, a *tanH* activation function and padding, serving as the output layer.

The input layer, which takes the input data; Two Conv1D layers, which serve as encoder; Two Conv1D transpose layers, which serve as decoder; One Conv1D layer with one output, a Sigmoid activation function and padding, serving as the output layer.

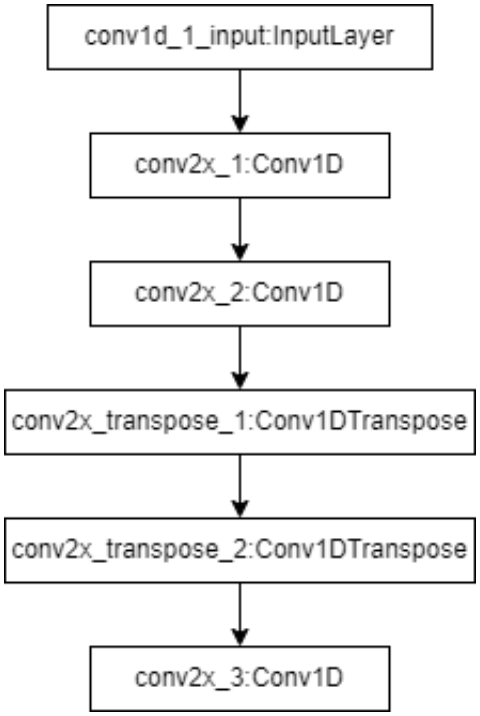

Figure 3.2.2: Model Structure

Following are few functions that were used in the model:

- **Conv1D** : This function creates a temporal convolution layer.In order to generate a tensor of outputs, this layer generates a convolution kernel that is convolved with the layer input over a single spatial (or temporal) dimension.There are few parameters that this function as which are as follows:
  - **filter** : The number of output filters in the convolution
  - **kernel_size** : An integer or tuple of a single integer, specifying the length of the 1D convolution window.(3 in used model)
  - **activation** : The activation function used.(Relu in used model)
  - **kernel_constraint** : Constraint function applied to the kernel matrix(max_norm in used model)
  - **kernel_initializer** : Initializer for the kernel weights matrix(he_uniform in used model)
- **Conv1DTranspose** : This function creates a transposed convolution layer.The requirement for transposed convolutions typically results from the need to use a transformation that goes the opposite way from that of a typical convolution, i.e., from something that has the shape of a convolution's output to something that has the shape of its input while maintaining a connectivity pattern that is compatible with said convolution.Even this function has few parameters that can be tuned which are as follows:
  - **filter** : The number of output filters in the convolution
  - **kernel_size** : An integer or tuple of a single integer, specifying the length of the 1D convolution window.(3 in used model)
  - **activation** : The activation function used.(Relu in used model)
  - **kernel_constraint** : Constraint function applied to the kernel matrix(max_norm in used model)
  - **kernel_initializer** : Initializer for the kernel weights matrix(he_uniform in used model)

We did experimentation with different types of activation functions. The first activation function we considered was **relu**.

$$f(x) = \max(0, x) \tag{3.2}$$

This function in the final layer was giving a clipped output where negative values were completely cut off. So to handle this dilemma we used **tanh**.

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.3}$$

This gave us the desired output when compared with **relu**. This was because no clipping of negative values.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 997, 128)          512

 conv1d_1 (Conv1D)           (None, 995, 32)           12320

 conv1d_transpose (Conv1DTra  (None, 997, 32)          3104
 nspose)

 conv1d_transpose_1 (Conv1DT  (None, 999, 128)         12416
 ranspose)

 conv1d_2 (Conv1D)           (None, 999, 1)            385

=================================================================
Total params: 28,737
Trainable params: 28,737
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.2.3: Model summary

The model can we summarized as seen in Figure 3.1.3

3.3 Model Training & Testing

The above mentioned model was first trained for sine waves of frequency ranging between 40Hz to 60Hz. A total of 1000 samples were generated consisting of frequencies in the aforementioned range, this implies we have 50 groups of sine waves having integer frequencies between 40Hz to 60Hz. White gaussian noise was added to all the signals in the training

dataset such that they had an SNR of 17dB. This model was then tested for 4 different cases :

1. An input sine wave having frequency (50Hz) within the training band (40Hz-60Hz).
2. An input sine wave having frequency (20Hz) before the training band.
3. An input sine wave having frequency (80Hz) after the training band.
4. An input sine wave having a combination of 2 frequencies with only one frequency lying inside the training band (50Hz and 15Hz)

3.3.1 Case-1

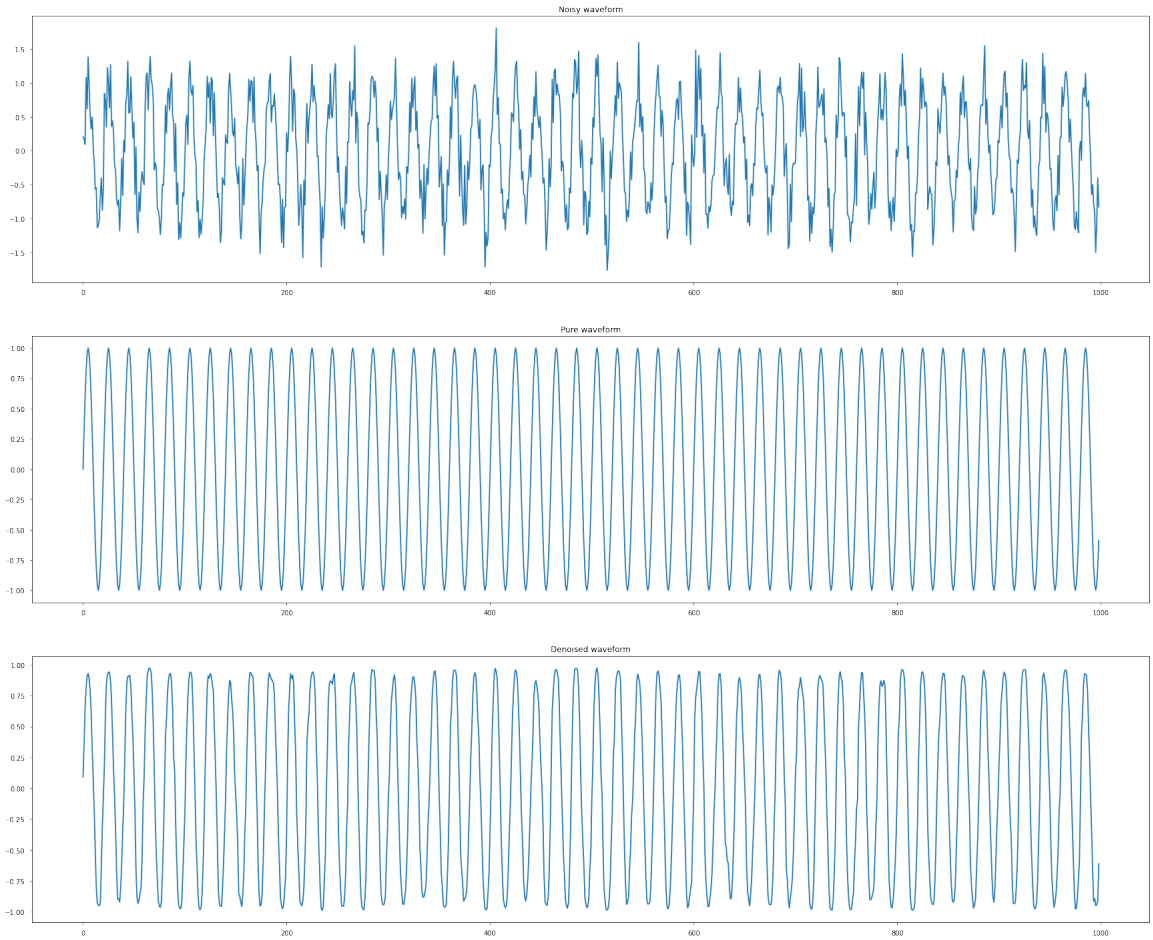

Figure 3.3.1: Input lying inside the training band. Input SNR-8.1946 dB, Output SNR-15.7629 dB (Execution time: 0.0521s)

3.3.2 Case-2

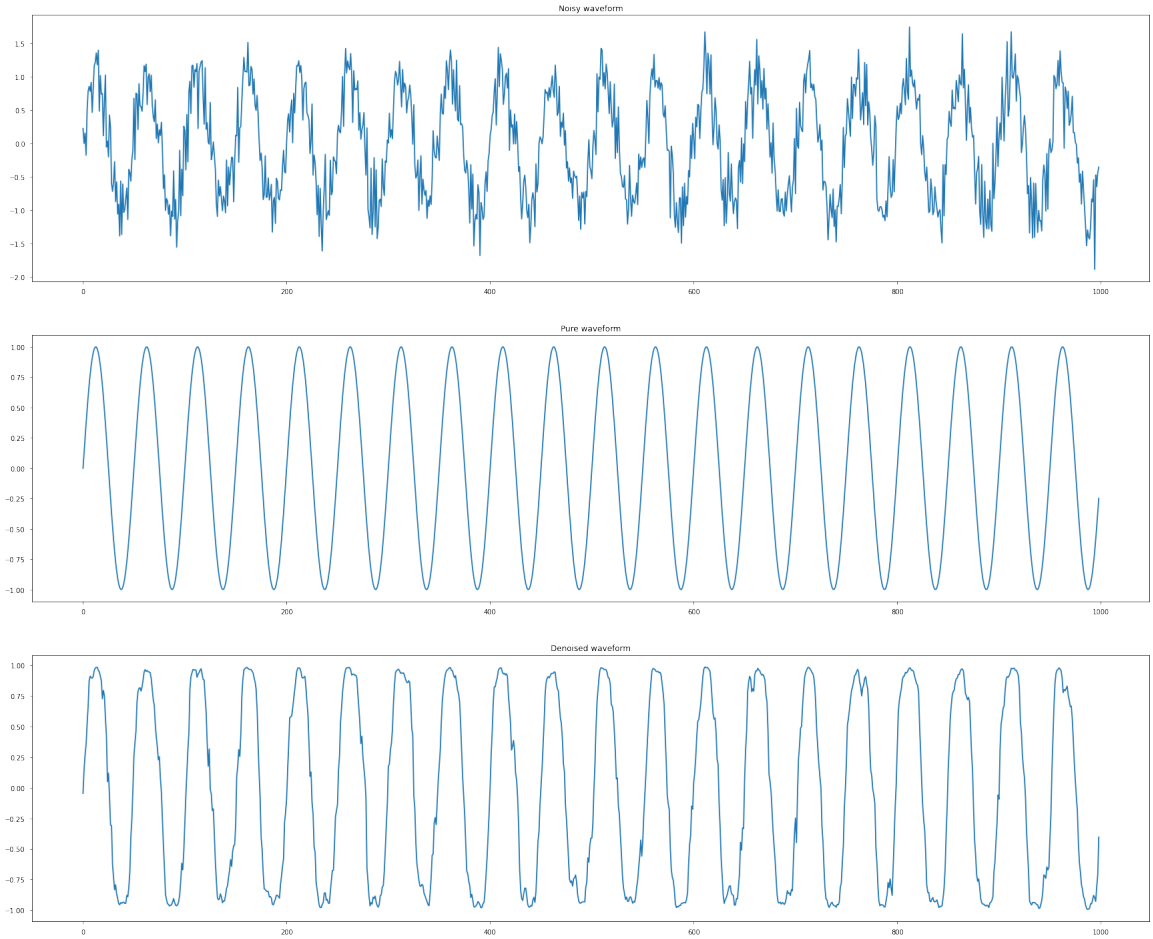

Figure 3.3.2: Input lying before the training band (20Hz). Input SNR-8.1436dB, Output SNR-13.83414dB (Execution time: 0.0565s)

3.3.3 Case-3

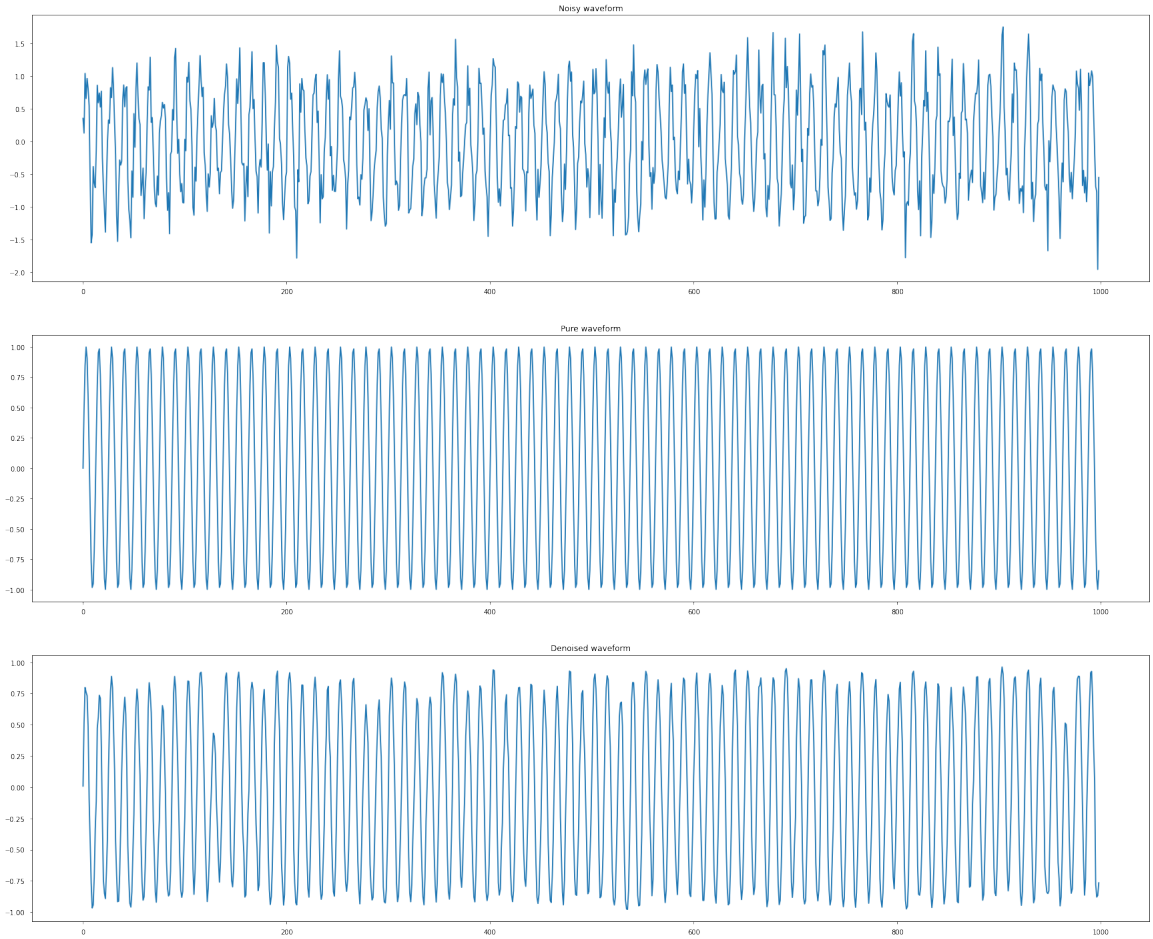

Figure 3.3.3: Input lying after the training band (80Hz). Input SNR-8.16007dB, Output SNR-11.85583dB (Execution time: 0.0440s)

3.3.4 Case-4

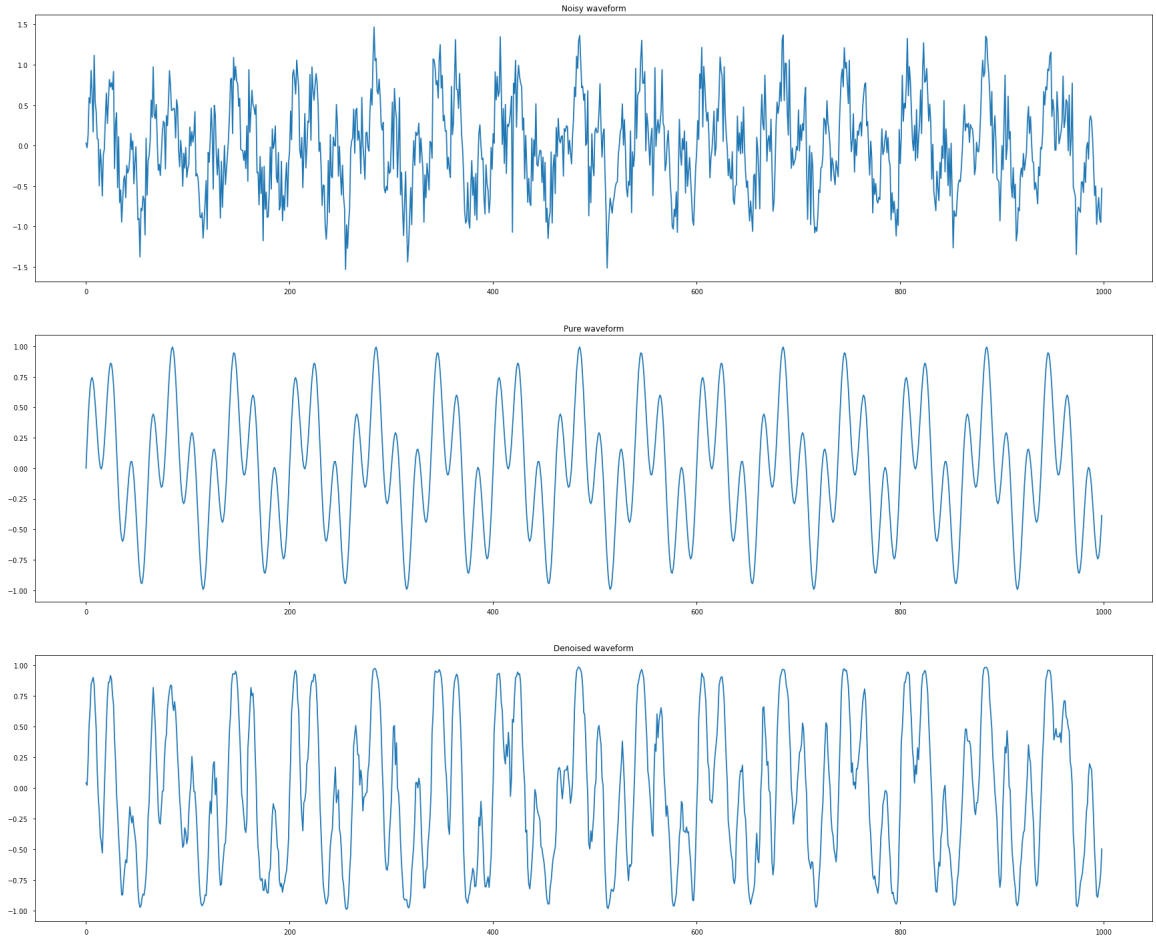

Figure 3.3.4: Input given as a combination of 2 frequencies with only one frequency lying inside the training band (50Hz and 15Hz). Input SNR-5.8243dB, Output SNR-9.7533dB (Execution time: 0.0.0500s)

CHAPTER 4

OBSERVATIONS & CONCLUSIONS

Based on the Signal to Noise Ratio (SNR) the effectiveness of the above-mentioned algorithms for denoising have been assessed. The following is a representation of the SNR:

$$SNR_{in} = 10\log_{10}\left(\frac{\sum_{t=1}^{T}|x[t]|^2}{\sum_{t=1}^{T}|y[t]-x[t]|^2}\right) \tag{4.1}$$

where x(t) is the signal and y(t) is the noisy signal.

$$SNR_{out} = 10\log_{10}\left(\frac{\sum_{t=1}^{T}|x[t]|^2}{\sum_{t=1}^{T}|x[t]-\bar{x[t]}|^2}\right) \tag{4.2}$$

Here $\bar{x(t)}$ is reconstructed signal:

Given below is a table depicting the performance of the autoencoder-based and the EMD-based algorithms for various values for each of the four input signal cases. Note that the values shown in Table 4.0.1 represent the results of training the autoencoder model with a limited dataset range, i.e. (40Hz-60Hz)

	Autoencoder-based (limited dataset)		EMD-Based	
Frequency of Sine wave	Noisy Signal SNR	Denoised Signal SNR	Noisy Signal SNR	Denoised Signal SNR
20Hz	8.14	13.2160	8.15	14.7843
50Hz	8.19	15.4901	8.15	11.6565
80Hz	8.16	11.7393	8.15	11.4618
50Hz+15Hz	5.82	9.7533	5.82	11.1502

Table 4.0.1: Performance of model trained with signals of SNR = 17dB VS EMD-Based method

1. From the above table, we can surmise that for a sine wave of frequency lying in the training band (i.e., 40Hz-60Hz), the auto-encoder-based method performs better compared to its EMD counterpart. However, for the remaining 3 cases, we notice that the former performance is almost identical or slightly inferior compared to the latter. But this is to be expected from the current model as although the model performs denoising to a considerable degree for the frequencies that lie outside the scope of its dataset, it still cannot compete with the traditional approach (i.e., the EMD-based approach).

This brings us to an important limitation of the machine learning-based denoising model, i.e., **the availability of data for training**. The traditional approach, on the other hand, needs no such prior requirement to denoise any given signals, as seen in section 1.4.2.

2. The EMD and wavelet-based approaches could take in any random signal (power frequency signal, audio signals, etc.) as mentioned in section 1.4.2 with additive white Gaussian noise of a particular SNR and denoise it adequately to an extent. In the case of the autoencoder-based approach, in order to perform effective denoising, we would require a sufficient number of data samples in the case of a power frequency signal or an audio signal. This insinuates yet another limitation of this method, its ability to generalize, i.e., this approach can be prone to **overfitting the training data**. This means that the autoencoder may be very good at denoising the specific signals it was trained on but may not perform as well on new, unseen signals.

3. However, the autoencoder-based model can be customized to specific signal types and noise characteristics. This allows for more accurate and effective noise removal than traditional methods that use fixed filters, as observed in Table 4.0.2. Note that the values shown in Table 4.0.2 represent the results of training the autoencoder model with an enlarged dataset range, i.e. (10Hz-90Hz) and increased number of samples i.e. (10000 samples)

	Autoencoder-based (enlarged dataset)		Autoencoder-based (limited dataset)	
Frequency of Sine wave	Noisy Signal SNR	Denoised Signal SNR	Noisy Signal SNR	Denoised Signal SNR
20Hz	7.998	14.7300	8.14	13.2160
50Hz	7.93	14.8251	8.19	15.4901
80Hz	8.00	12.9100	8.16	11.7393
50Hz+15Hz	5.31	10.7189	5.82	9.7533

Table 4.0.2: Input and Output SNR values for different denoising methods tested for various input signals

4. From the above table, we can come to the conclusion that when the model is given access to a broader dataset, it performs considerably better in the cases where it fails to outperform the EMD-based method in contrast to the previous scenario. This speaks to the **customizability of models** of the machine-learning-based approach. Another important point to note is that the band size used for training the model is inversely proportional to the output SNR value for a given number of samples. This can be seen in the case when the sine wave has a frequency of 50HZ, wherein the output SNR value deteriorates slightly on increasing the training band.

5. It is important to note that the training of the auto-encoder model was done using a dataset of signals with **SNR = 17dB** whereas the testing was done with input signals of **SNR = 8dB**. Despite this fact, we can clearly notice that the ML-based model still performs considerably better for signals of varying input SNR values, as seen in the tables above. The following tables show the performance of the autoencoder model trained with a dataset of signals with SNR = 8dB, and it was tested with an input signal of SNR of 8dB as well. It is evident from this table that the performance of the ML model improves even further when trained and tested for the same SNR.

	Autoencoder-based (enlarged dataset)		EMD-Based	
Frequency of Sine wave	Noisy Signal SNR	Denoised Signal SNR	Noisy Signal SNR	Denoised Signal SNR
20Hz	7.99	16.04	8.15	14.7843
50Hz	7.92	15.16	8.15	11.6565
80Hz	8.00	13.767	8.15	11.4618
50Hz+15Hz	5.82	10.36	5.82	11.1502

Table 4.0.3: Performance of model trained and tested with signals of SNR = 8dB VS EMD-Based method

6. Another observation is that Autoencoders can be **computationally expensive** to train, especially when dealing with large datasets. This can be a limitation when working with real-time signals or when dealing with limited computing resources. In contrast, with the traditional approach, the computational power required does not vary proportionally with the complexity of the data, although it might take a modest amount of time to compute the results.

7. The Autoencoder-based denoising method is an **end-to-end learning approach**, meaning that the autoencoder learns to denoise the signal directly from the noisy input. In contrast, EMD-based denoising involves decomposing the signal into intrinsic mode functions (IMFs) and then reconstructing a denoised signal from a subset of the IMFs, as mentioned in section 1.4.2. This multi-step process can introduce additional noise and artifacts into the denoised signal.

8. From Table 1.4.1 we can discern that the EMD and wavelet-based denoising approaches are not flexible, meaning their performance is not consistent for the four different inputs. This implies that it may not work as well in certain domains or for certain types of noise. However, Autoencoder-based denoising is a **flexible approach** that can be adapted to different types of noise and signal domains.

CHAPTER 5

FUTURE WORK

As mentioned in Chapter 4, the current model of the autoencoder works adequately well for only sinusoidal waves and cannot generalize well for other waveforms such as power frequency signals or audio signals due to the lack of sufficient data samples. Thus we aim to acquire relevant and ample data and test our model for other signals as mentioned above in the future.

REFERENCES

[1] Komal Bajaj, Dushyant Kumar Singh, and Mohd. Aquib Ansari. Autoencoders based deep learner for image denoising. *Procedia Computer Science*, 171:1535–1541, 2020. Third International Conference on Computing and Network Communications (CoCoNet'19).

[2] Thanh Tran, Sebastian Bader, and Jan Lundgren. Denoising induction motor sounds using an autoencoder. *arXiv preprint arXiv:2208.04462*, 2022.

[3] Cheng Yu, Ryandhimas E. Zezario, Syu-Siang Wang, Jonathan Sherman, Yi-Yen Hsieh, Xugang Lu, Hsin-Min Wang, and Yu Tsao. Speech enhancement based on denoising autoencoder with multi-branched encoders. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2756–2769, 2020.

[4] Woong-Hee Lee, Mustafa Ozger, Ursula Challita, and Ki Won Sung. Noise learning-based denoising autoencoder. *IEEE Communications Letters*, 25(9):2983–2987, 2021.

[5] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[6] Lingheng Meng, Shifei Ding, and Yu Xue. Research on denoising sparse autoencoder. *International Journal of Machine Learning and Cybernetics*, 8:1719–1729, 2017.