

8-2018

Comparison of Neural Networks and Least Mean Squared Algorithms for Active Noise Canceling

Samuel Kyung Won Park
Clemson University, skpark@g.clemson.edu

Follow this and additional works at: https://tigerprints.clemson.edu/all_theses

Recommended Citation

Park, Samuel Kyung Won, "Comparison of Neural Networks and Least Mean Squared Algorithms for Active Noise Canceling" (2018).
All Theses. 2920.
https://tigerprints.clemson.edu/all_theses/2920

This Thesis is brought to you for free and open access by the Theses at TigerPrints. It has been accepted for inclusion in All Theses by an authorized administrator of TigerPrints. For more information, please contact kokeefe@clemson.edu.

COMPARISON OF NEURAL NETWORKS AND LEAST MEAN SQUARED ALGORITHMS FOR ACTIVE NOISE CANCELING

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Electrical Engineering

by
Samuel Kyung Won Park
August 2018

Accepted by:
Dr. Carl Baum and Dr. Eric Patterson, Committee Chair
Dr. Harlan Russell
Dr. Robert Schalkoff
Dr. Apoorva Kapadia

Abstract

Active Noise Canceling (ANC) is the idea of using superposition to achieve cancellation of unwanted noise and is implemented for many applications such as attempting to reduce noise in a commercial airplane cabin. One of the main traditional techniques for noise cancellation is the adaptive least mean squares (LMS) algorithm that produces the anti-noise signal, or the 180 degree out-of-phase signal to cancel the noise via superposition. This work attempts to compare several neural network approaches against the traditional LMS algorithms. The noise signals that are used for the training of the network are from the Signal Processing Information Base (SPIB) database. The neural network architectures utilized in this paper include the Multilayer Feedforward Neural Network, the Recurrent Neural Network, the Long Short Term Neural Network, and the Convolutional Neural Network. These neural networks are trained to predict the anti-noise signal based on an incoming noise signal. The results of the simulation demonstrate successful ANC using neural networks, and they show that neural networks can yield better noise attenuation than LMS algorithms. Results show that the Convolutional Neural Network architecture outperforms the other architectures implemented and tested in this work.

Table of Contents

Title Page	i
Abstract	ii
Dedication	iii
Acknowledgments	iii
List of Tables	iv
List of Figures	v
1 Introduction	1
2 Background	4
2.1 Adaptive Filtering	4
2.2 ANC Systems	7
2.3 Neural Network Architectures	11
3 Previous Work	17
4 Experiment	21
5 Results	24
6 Conclusion	37
Bibliography	39

List of Tables

3.1	Noise reduction using NLMS filters [18]	17
3.2	RNN and MLF noise reduction for F16	20
3.3	RNN and MLF noise reduction for destroyer	20
3.4	RNN and MLF noise reduction for Volvo	20
3.5	RNN and MLF noise reduction for Leopard	20
5.1	NLMS noise reduction	24
5.2	dB reduction using MLF	24
5.3	dB reduction using RNN	25
5.4	dB reduction using LSTM	25
5.5	dB reduction using CNN	26

List of Figures

2.1	Feedforward ANC System	7
2.2	Feedback ANC System	8
2.3	Feedforward ANC block diagram	9
2.4	Feedback ANC block diagram	9
2.5	Feedback ANC with $\hat{S}(z) = S(z)$	10
2.6	Multilayered Perceptron Diagram	11
2.7	Recurrent Neural Network diagram	14
2.8	Long Short Term Memory Neural Network	15
2.9	From Krizhevsky et al. (2012) [15] Deep Learning Convolutional Neural Network	16
5.1	Destroyer Noise Signal	26
5.2	LSTM Destroyer Output	27
5.3	LSTM Residual Destroyer Noise	27
5.4	F-16 Noise Signal	28
5.5	LSTM F-16 Output	28
5.6	LSTM Residual F-16 Noise	29
5.7	Leopard Noise Signal	29
5.8	LSTM Leopard Output	30
5.9	LSTM Residual Leopard Signal	30
5.10	Volvo Noise Signal	31
5.11	LSTM Volvo Output	31
5.12	LSTM Residual Volvo Noise	32
5.13	CNN anti noise signal with Destroyer noise signal	32
5.14	CNN anti noise signal with F-16 noise signal	33
5.15	CNN anti noise signal with Leopard noise signal	33
5.16	CNN anti noise signal with Volvo noise signal	34
5.17	CNN Residual Destroyer Noise	34
5.18	CNN Residual F-16 noise	35
5.19	CNN Residual Leopard Noise	35
5.20	CNN Residual Volvo Noise	36

Chapter 1

Introduction

In everyday activities people experience the effects of numerous sources of noise. Loud noises in the workplace can adversely affect persons with prolonged exposure. Passive techniques such as barriers and silencers can be effective, but they are often bulky, costly, and ineffective at low frequencies. Active Noise Canceling (ANC) is a successful technique implemented in many commercial applications such as commercial airplane cabins where ANC reduces engine noise so that passengers experience less effects from the loud decibel levels of the engine noise. A primary traditional technique for ANC is the adaptive least mean squares (LMS) algorithm that produces the anti-noise signal, or the 180 degree out-of-phase signal, to cancel the noise via superposition [22].

LMS filtering is based on the steepest descent algorithm that employs an instantaneous estimate of the gradient of the mean squared error. It is a stochastic gradient descent method in that the filter is adapted from the error at the present time. The weight update strategy of the algorithm is based on the “direction” of the error. When the mean squared error (MSE) gradient is positive, it implies that the error is going towards a positive direction. Because of this, the weights to the filter are changed to prevent the increase of error, and vice versa when the MSE gradient is negative. The weight update equation is as follows: $w_{n+1} = w_n - \mu \Delta \epsilon[n]$, where ϵ represents the MSE and μ represents the step size, or learning rate [7]. While the adaptive method of error correction that the LMS algorithm employs reaches a local minima, the algorithm is not guaranteed to reach a global minima to ensure the best possible solution. There are many variations to the LMS algorithms that attempt to compensate for various shortcomings of the algorithm. The pure LMS algorithm is changed and manipulated to accommodate for instability, nonlinearity, and non-convergence at the

trade-off of increased complexity of implementation.

In ANC systems two popular LMS algorithms are often employed. The first algorithm takes into account path functions that exist in an environment. There is a primary path where the reference signal travels to the loud speaker and there is a secondary path where the signal needs to travel from the microphone to the input of the filter, or when the output signal needs to travel from the output of the filter to the loud speaker. To account for the secondary path effect, a filtered version of LMS, called Filtered X LMS (FXLMS) is commonly used in ANC systems. The X is short for auxiliary path and in this paper this auxiliary path is referred as the secondary path. The algorithm utilizes FIR filter coefficients to model the secondary path. One of the weaknesses of this is that the FXLMS is limited to linear control or filtering problems [2]. In other words, the control input signal as well as the measured associated error signal used in the adaptation process must be linear function of the reference signal used by the adaptive filter to derive the control signal [26]. The second algorithm that is popular with ANC application is the normalized least mean squares (NLMS) algorithm. One of the drawbacks of using a pure LMS algorithm is that it is sensitive to the scaling of its input. This makes the system less stable and is challenging to choose an appropriate step size, μ , so the algorithm normalizes the power of the input to allow for more stability. While the NLMS is simple and less of a load on computation, the convergence of the algorithm can suffer. While the traditional ANC system works, the LMS algorithms are tailored more towards lower frequency bands and one specific noise environment [4].

Neural Networks and especially the convolutional neural network (CNN) have recently risen in popularity because of the increase in capabilities of graphical-processor-unit (GPUs), computation for training much larger training sets with millions of labeled examples, and better architecture regularization strategies for preventing overfitting [12].

Neural networks can be used to tackle complicated problems with nonlinearity and instability while providing potentially better convergence or performance [2] [6]. Neural Networks also employ a gradient descent algorithm but in updating the weights of the network base on *a priori* knowledge of an input signal. Published neural network architecture experiments have shown success in noise reduction. The neural network replaces the LMS algorithm and is trained to perform its own filtering to meet the criteria of a proper ANC system.

This work attempts to exceed the traditional LMS algorithm using neural networks such as the Multilayered Feedforward Neural Network (MLF), the Recurrent Neural Network (RNN), the

Long Short Term Memory Neural Network (LSTM), and the Convolutional Neural Network (CNN) for ANC. Also, this work investigates whether current architectures, untried with ANC, such as the LSTM and the CNN can achieve better results than established findings with more traditional networks such as MLFs or RNNs.

Chapter 2

Background

2.1 Adaptive Filtering

The following derivations are a summary from Widrow *et al.* on the adaptive filtering using the Least Mean Squared algorithm [28]. The filter updates the inputs simultaneously by adjusting weights in response to a desired output. The input vector is defined to be

$$\mathbf{X}_f = \begin{bmatrix} x_{0f} \\ x_{1f} \\ \vdots \\ x_{nf} \end{bmatrix} \quad (2.1)$$

The input components are all at discrete intervals and are indexed by subscript j . The component x_{0f} is a constant usually a 1 and used only in cases of needing a bias. The weighting coefficients are defined by the following:

$$\mathbf{W} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix} \quad (2.2)$$

The \mathbf{w}_0 is a bias usually a 1 in case of needing a bias. The output, \mathbf{y}_j , is equal to the inner product of \mathbf{X}_j and \mathbf{W} .

$$\mathbf{y}_j = \mathbf{X}_j^T \mathbf{W} = \mathbf{W}^T \mathbf{X}_j \quad (2.3)$$

The error, ϵ_j , is defined as the error between the output, \mathbf{y}_j , and the desired response, \mathbf{d}_j .

$$\epsilon_j = \mathbf{d}_j - \mathbf{X}_j^T \mathbf{W} = \mathbf{d}_j - \mathbf{W}^T \mathbf{X}_j \quad (2.4)$$

The LMS adaptive algorithm adjusts the weights to minimize the mean-square error. Assuming that the input signals, \mathbf{X}_j , and the desired response, \mathbf{d}_j , is statistically stationary and the weights are also fixed, a general expression for the mean-square error as a function of weight values can be derived.

$$\epsilon_j^2 = \mathbf{d}_j^2 - 2\mathbf{d}_j \mathbf{X}_j^T \mathbf{W} + \mathbf{W}^T \mathbf{X}_j \mathbf{X}_j^T \mathbf{W} \quad (2.5)$$

Taking the expected value of both sides yields

$$E[\epsilon_j^2] = E[\mathbf{d}_j^2] - 2E[\mathbf{d}_j \mathbf{X}_j^T] \mathbf{W} + \mathbf{W}^T E[\mathbf{X}_j \mathbf{X}_j^T] \mathbf{W}. \quad (2.6)$$

Since \mathbf{d}_j is a scalar and \mathbf{X}_j is a vector, the cross correlation, \mathbf{A} , between them is

$$\mathbf{A} = E[\mathbf{d}_j \mathbf{X}_j] = E \begin{bmatrix} \mathbf{d}_j x_{0j} \\ \mathbf{d}_j x_{1j} \\ \vdots \\ \mathbf{d}_j x_{nj} \end{bmatrix}. \quad (2.7)$$

The input correlation matrix, \mathbf{R} , is

$$\mathbf{R} = E[\mathbf{X}_j \mathbf{X}_j^T] = E \begin{bmatrix} x_{0j} x_{0j} & \dots & x_{0j} x_{nj} \\ \vdots & \ddots & \vdots \\ x_{nj} x_{0j} & \dots & x_{nj} x_{nj} \end{bmatrix}. \quad (2.8)$$

The input correlation matrix is symmetric, positive definite, or in rare cases positive semidefinite. Thus the mean-square error can be expressed as

$$E[\epsilon_j^2] = E[d_j^2] - 2\mathbf{A}^T \mathbf{W} + \mathbf{W}^T \mathbf{R} \mathbf{W}. \quad (2.9)$$

The error function is a quadratic function that never goes negative. Adjusting the weights to minimize the error involves using gradient descent. The gradient of the error is

$$\nabla = \begin{bmatrix} \frac{\partial E[\epsilon_j^2]}{\partial w_0} \\ \vdots \\ \frac{\partial E[\epsilon_j^2]}{\partial w_n} \end{bmatrix} = -2\mathbf{A} + 2\mathbf{R}\mathbf{W}. \quad (2.10)$$

The optimal weight vector, \mathbf{W}^* , generally called the Wiener weight vector is obtained by setting equation 2.10 to zero which yields, $\mathbf{W}^* = \mathbf{R}^{-1}\mathbf{A}$. By using the gradient, the weights can be updated by the weight updating equation,

$$\mathbf{W}_{j+1} = \mathbf{W}_j - \mu \nabla_j. \quad (2.11)$$

The parameter μ is the step size and it influences stability and convergence. The LMS algorithm assumes that ϵ_j^2 is an estimate of the mean-square error and thus differentiates ϵ_j^2 with respect to \mathbf{W} .

$$\hat{\nabla}_j = \begin{bmatrix} \frac{\partial \epsilon_j^2}{\partial w_0} \\ \vdots \\ \frac{\partial \epsilon_j^2}{\partial w_n} \end{bmatrix} = 2\epsilon_j \begin{bmatrix} \frac{\partial \epsilon_j}{\partial w_0} \\ \vdots \\ \frac{\partial \epsilon_j}{\partial w_n} \end{bmatrix} \quad (2.12)$$

The partial derivative of the instantaneous error with respect to the weight components can be obtained by differentiating equation 2.5, thus the estimate of the gradient can be simplified to

$$\hat{\nabla}_j = -2\epsilon_j \mathbf{X}_j \quad (2.13)$$

Using the gradient estimate, the weight update equation becomes the Widrow-Hoff LMS algorithm [28].

$$\mathbf{W}_{j+1} = \mathbf{W}_j + 2\mu\epsilon_j \mathbf{X}_j \quad (2.14)$$

2.2 ANC Systems

The basic principle of Active Noise Cancellation is superposition of signals to reduce sound pressure in an environment. There exists a noise signal and that noise signal can be canceled by an opposing anti-noise signal that is the exact replica but opposite of the original noise signal in that it is with 180-degrees out-of-phase from the original noise signal. By superposition, the two signals then can cancel each other [8] [22]. There are two methods that utilize ANC. One method is the feedforward approach with a two microphone and single loudspeaker set up. The first microphone records the reference signal or the noise signal as shown in Figure 2.1. The second microphone measures the resulting signal or error signal produced from the loudspeaker. The reference signal and the error signal travel through a path that consists of a preamplifier, anti-aliasing filter, and an analog-to-digital converter (ADC) [3]. While this method is viable, the loud speaker and the error microphone can have feedback and the input microphone may not have information for the system as ANC occurs. A second method is to eliminate one of the microphones and use only a single microphone. This is referred to as the feedback method as shown in Figure 2.2. The microphone is used to measure the error signal, and the loudspeaker produces the anti-noise signal. For this method, the reference signal can be produced by adding the 180-degree phase shift from the anti-noise signal. The feedback method is preferred because in the feedforward method the input sensor may not provide enough information about the noise and there exists some feedback from the cancellation speaker to the input sensor [19].

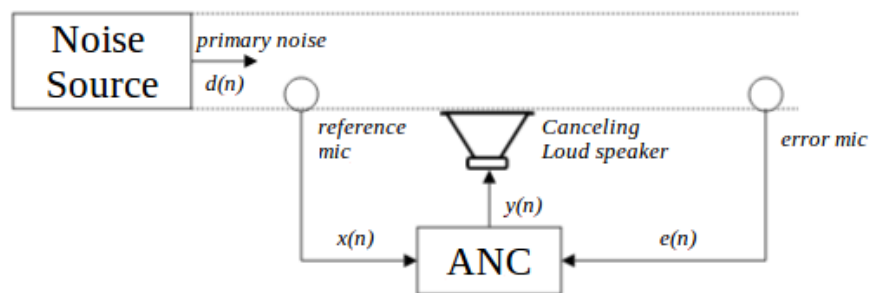


Figure 2.1: Feedforward ANC System

The feedforward approach is shown as a block diagram in Figure 2.3. This figure is characterized by a primary path function, $P(z)$, the secondary path function, $S(z)$, and the finite impulse

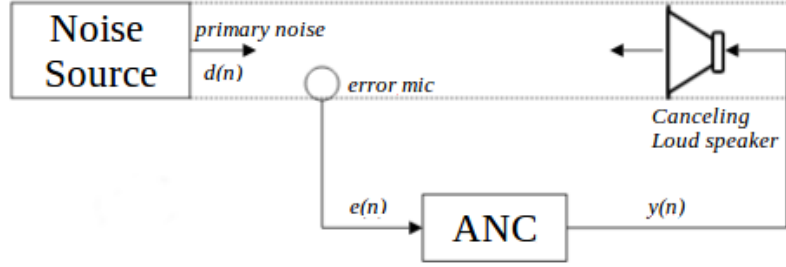


Figure 2.2: Feedback ANC System

response (FIR) filter, $W(z)$. The primary path function models the path from the noise source to the reference microphone. $P(z)$ is treated as a negligible function because of the assumption that the noise source and the reference microphone is close. The secondary path function is the model of the path from the cancellation speaker to the error microphone. The FIR filter coefficients are updated by the LMS algorithm to output the anti-noise signal for the ANC system by using the error signal and the reference noise signal.

In the feedback approach, the primary noise signal, $d(n)$, is not available. The main point as expressed in the feedback block diagram (Figure 2.4) is to regenerate the $d(n)$ signal from the error signal. It is shown in figure 2.2 that the primary noise can be expressed in the frequency domain as $D(z) = E(z) + S(z)Y(z)$, where $E(z)$ is the error signal, $S(z)$ is the secondary path transfer function, and $Y(z)$ is the output signal. The secondary path transfer function, $S(z)$, is the path for the signal to travel from the loud speaker to the error microphone. The estimate for the secondary path transfer function will be denoted as $\hat{S}(z)$. Thus, the estimate of $d(n)$ from the estimate of the secondary path transfer function, $\hat{S}(z)$, can be used to synthesize the reference signal, $x(n)$, as $X(z) \equiv \hat{D}(z) = E(z) + \hat{S}(z)Y(z)$.

From Figure 2.4, the reference signal, $x(n)$, and the output signal, $y(n)$, can be expressed as,

$$x(n) \equiv \hat{d}(n) = e(n) + \sum_{m=0}^{M-1} \hat{s}_m y(n-m) \quad (2.15)$$

$$y(n) = \sum_{l=0}^{L-1} w_l(n)x(n-l) \quad (2.16)$$

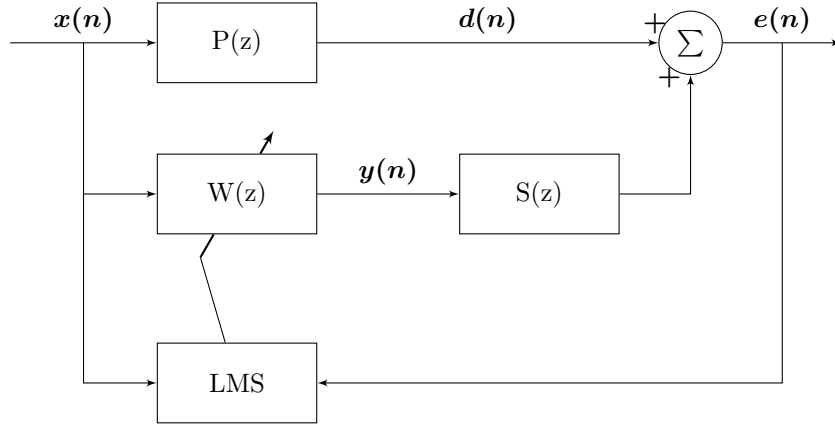


Figure 2.3: Feedforward ANC block diagram

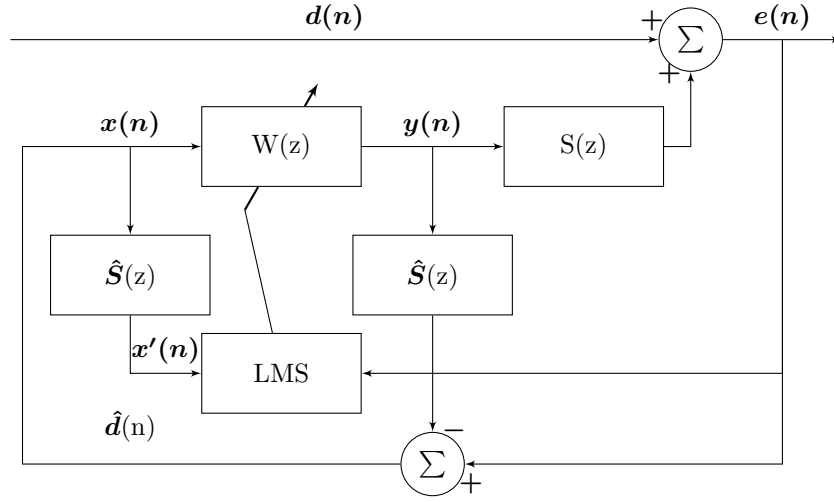


Figure 2.4: Feedback ANC block diagram

In Equation 2.15, $\hat{s}_m, m = 0, 1, \dots, M - 1$ is the M^{th} order Finite Impulse Response (FIR) filter which is used to approximate the secondary path transfer function. In Equation 2.16, $w_l(n), l = 0, 1, \dots, L - 1$ are the L^{th} order adaptive FIR filter coefficients of $W(z)$ at time n . These coefficients, \hat{s}_m and $w_l(n)$, are then updated by the filtered-X LMS algorithm (FXLMS) as,

$$w_l(n + 1) = w_l(n) + \mu x'(n - l)e(l) \quad (2.17)$$

In Equation 2.17, μ is the step size and $x'(n)$ is the output of $\hat{S}(z)$ from the synthesized reference signal, $x(n)$, in Figure 2.4. It can be shown that $x(n) = d(n)$ if $\hat{S}(z) = S(z)$. Assuming this condition is satisfied, then this feedback ANC system can be transformed into a feedforward system

as the feedback system can be expressed as

$$\frac{E(z)}{D(z)} = \frac{1 + W(z)S(z)}{1 + W(z)(\hat{S}(z) - S(z))}. \quad (2.18)$$

So when $\hat{S}(z) = S(z)$, the transfer function becomes

$$\frac{E(z)}{D(z)} = 1 + W(z)S(z). \quad (2.19)$$

The operation of Equation 2.19 is shown in Figure 2.5 below.

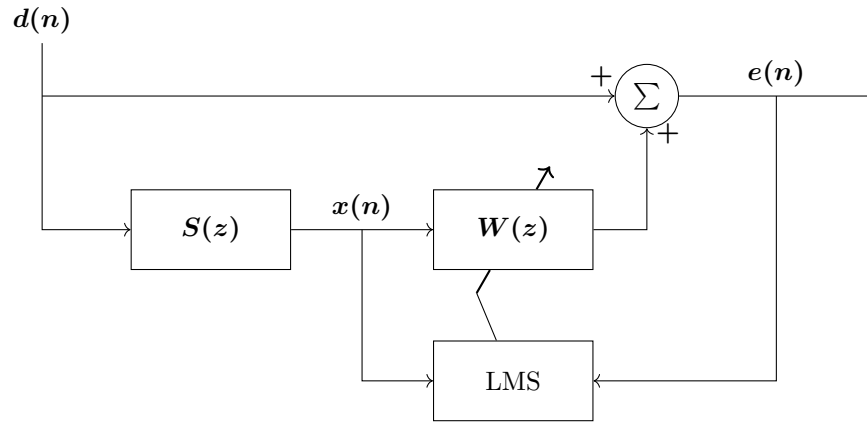


Figure 2.5: Feedback ANC with $\hat{S}(z) = S(z)$

The estimate of the secondary filter, $S(z)$, is usually a delay [16] [21]. So, when $S(z)$ is assumed to be a pure delay i.e. $S(z) = z^{-\Delta}$, then the feedback ANC system will be equivalent to a standard adaptive predictor. For this research, the secondary path is modeled as a pure delay where $S(z) = z^{-1}$.

For this work, the LMS algorithm and $W(z)$ are replaced with a neural network that predicts and approximates the appropriate anti-noise signal. The neural network is trained to recognize different types of noise environments and to correspond accordingly. The neural network will be trained with noises from the Signal Processing Information Base (SPIB). The selected noise for training will include noises from the destroyer engine room, the cockpit of an F16 fighter jet, the cabin of a Leopard military vehicle, and a cabin of a Volvo. These four datasets were chosen based on previous works dealing with these datasets, availability, and a good representation of different types of noise from high frequency to low frequency.

2.3 Neural Network Architectures

2.3.1 Multilayer Feedforward Neural Network

The Multilayer Feedforward Neural Network is one of the older and simpler networks. It is usually comprised of an input layer, a hidden layer, and an output layer as shown in Figure 2.6. The units are all fully connected, and it is trained using the backpropagation algorithm [25] [23]. This network does well on estimating functions and other processing tasks [24]. While using gradient descent, the conceptual difference between LMS and Neural Networks is that LMS is using gradient descent to adapt the FIR filter and the MLF uses it to estimate weights based on target versus error. The MLF also has the local minima problem. Having convergence with the network does not guarantee that it is the best solution because of the uncertainty that the network converged to the global minima. The backpropagation uses the generalized delta rule or the GDR. The way it uses gradient descent procedure is from the output layer backward to the input layer attempting to minimize error, hence the name backpropagation.

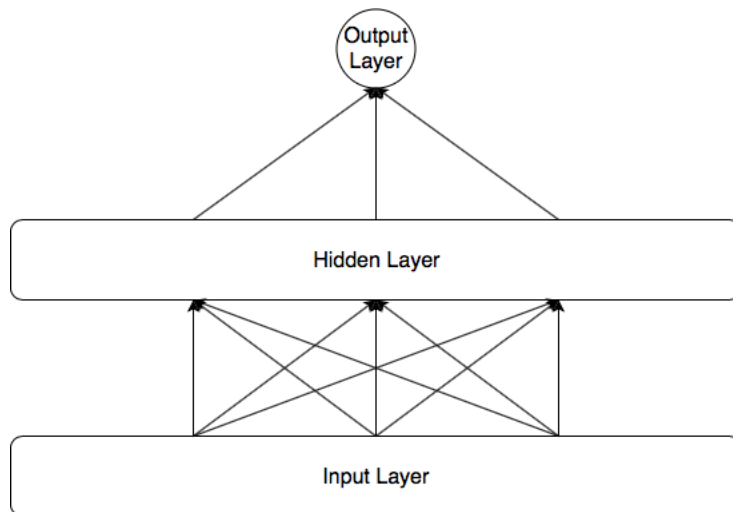


Figure 2.6: Multilayered Perceptron Diagram

2.3.2 Backpropagation

The backpropagation algorithm for a multilayered network is summarized in the following steps [25]. The experiments here are performed in epoch iterations so the derivations are also done in epoch. The target output from the training set is t_j^p , and o_j^p is the output from the individual j^{th}

unit in the output layer where j is the number of output units. The overall error is E , and the epoch error is denoted as E^p . To get pattern/epoch error measure, $E^p = \frac{1}{2} \sum_j (t_j^p - o_j^p)^2$. The total error is $E = \sum_p E^p$. The weights is denoted as w_{ji} where the weight goes from the previous layer unit i to the current layer unit j . The sum of all the weights times the output from the previous layer at the current j^{th} unit is call the net activation which is denoted as $net_j = \sum_i w_{ji} o_i$, or if there is a bias included, $net_j = \sum_i w_{ji} o_i + bias_j$. At each unit there is a designated activation function that can vary with application, and they include functions such as relu, softmax, sigmoidal, tanh, etc [25] [10]. The purpose of an activation function is to squash the incoming inputs between a range of values typically between 0 and 1, or -1 and 1 as all the inputs are multiplied to the weights and are summed at each unit. This activation function will be denoted as f_j , where it is the activation function at the entire layer and that it is nondecreasing and differentiable. The objective is to minimize the error of the network in respect to the output of the network. Therefore, backpropagation starts at the outermost layer, in this case the j^{th} layer. The chain rule,

$$\frac{\partial E^p}{\partial w_{ji}} = \frac{\partial E^p}{\partial o_j^p} \frac{\partial o_j^p}{\partial net_j^p} \frac{\partial net_j^p}{\partial w_{ji}} = \frac{\partial E^p}{\partial net_j^p} \frac{\partial net_j^p}{\partial w_{ji}}, \quad (2.20)$$

is formally used to develop the delta rule and the generalized delta rule. From the definition of net activation, the derivative of the net activation in respect to the weights can be taken from the previous definition and as such,

$$\frac{\partial net_j^p}{\partial w_{ji}} = o_i^p \quad (2.21)$$

where the o_i^p is the output from the previous i^{th} unit, or if in the case of an input layer, it will be the input value alone. Through the chain rule and minimizing the output layer of the MLF, the sensitivity of the pattern error on the net activation of the j^{th} unit is:

$$\delta_j^p = -\frac{\partial E^p}{\partial net_j^p} \quad (2.22)$$

By substituting Equations 2.5 and 2.6 into 2.4, one obtains

$$\frac{\partial E^p}{\partial w_{ji}} = -(\delta_j^p) o_i^p. \quad (2.23)$$

Equation 2.7 is essentially the update strategy needed for the weight updates. Furthermore, because

$$\frac{\partial E^p}{\partial net_j^p} = \frac{\partial E^p}{\partial o_j^p} \frac{\partial o_j^p}{\partial net_j^p} \quad (2.24)$$

the derivative of the output in respect to the net activation is simply the derivative of the activation function at that unit, so $\frac{\partial o_j^p}{\partial net_j^p} = f'_j(net_j^p)$. If the layer for the units 1 through j is the output, then the derivative of the pattern error in respect to the output can be derived from the previous definition of the pattern error, E^p .

$$\frac{\partial E^p}{\partial o_j^p} = -(t_j^p - o_j^p) \quad (2.25)$$

Using Equations 2.6 and 2.8, the sensitivity of the pattern error at the output layer can be expressed as

$$\delta_j^p = (t_j^p - o_j^p) f'_j(net_j^p) \quad (2.26)$$

By minimizing the error in respect to the weights, $\frac{\partial E^p}{\partial w_{ji}}$, the change of the weights from previous layer to current layer is

$$\Delta^p w_{ji} = -\epsilon \left(\frac{\partial E^p}{\partial w_{ji}} \right) = \epsilon \delta_j^p o_i^p \quad (2.27)$$

where ϵ is the learning rate. If i is the number of hidden units in the hidden layer, and j is the number of units in the next or output layer, then the chain rule can once again be employed as the following equation.

$$\frac{\partial E^p}{\partial o_i^p} = \sum_j \frac{\partial E^p}{\partial net_j^p} \frac{\partial net_j^p}{\partial o_i^p} = \sum_j (-\delta_j^p w_{ji}) \quad (2.28)$$

So the sensitivity of the pattern error at the hidden layer can be formulated from Equation 2.8 by substituting known variables.

$$\delta_i^p = -\frac{\partial E^p}{\partial o_i^p} f'_i(net_i^p) = f'_i(net_i^p) \sum_j \delta_j^p w_{ji} \quad (2.29)$$

There are other variations to the optimizers other than the stochastic gradient descent (SGD). There are techniques such as momentum [25] in SGD that can help the gradient descent push past a local minima, and other variations that help with computational efficiency and convergence [14].

2.3.3 Recurrent Neural Networks

Recurrent Neural Networks (RNN), or Elman's Network [24] [19], are often better at time dependent inputs and outputs. Recurrent neural networks have hidden layers where the outputs are tied back to the inputs. With such a feedback loop, the loop contains the hidden layer from the previous iteration shown in Figure 2.7, the network itself has the capabilities of holding information or holding memory. Because of such properties, the RNN has the potential to do well in time-dependent applications. As such, ANC applications built using RNN based networks have potential for better results. Backpropagation is used in a RNN as well but with one key difference. The feedback of the hidden layer can be unrolled into a feedforward network with time variation. The first layer with its own hidden layer will be the starting network, and the next network will be the next time step with the hidden layer of the next time step and the previous time step. This can go on until specified time steps have been satisfied. The backpropagation used here is called backpropagation through time (BPTT) where it utilizes the backpropagation from the last time step to the first time step [27]. The backpropagation updates the weights from last to first.

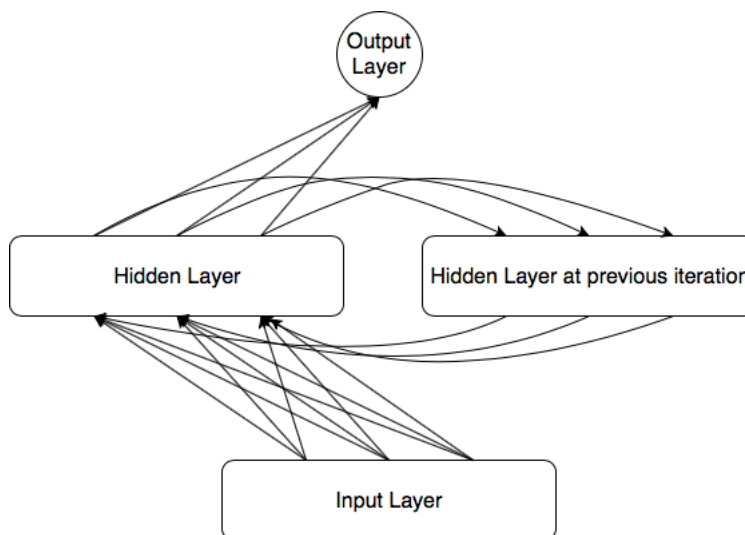


Figure 2.7: Recurrent Neural Network diagram

2.3.4 Long Short Term Memory Neural Network

A more recent architecture extended from the simple RNN is the Long Short Term Memory (LSTM) neural networks. They have been used in speech recognition applications and language

models among other applications [11]. LSTMs are unique because they have specific gates that allow for memory retention beyond just short term memory. RNNs are in a sense dependent on short term memory where the size of the hidden layer greatly influence memory capacity. LSTM networks provide a way to have both the long term memory and short term memory. The network forgets the unimportant information and makes room for important memory thus allowing for long term and short term memory. In the LSTM, there are three gates that allow for this type of functionality. These gates are units with sigmoidal activation functions. The first gate is a sigmoid layer that outputs numbers from 0 to 1 with 0 meaning forget incoming information and a 1 meaning let everything through. The information from the first gate is then multiplied into the cell state, C_{t-1} from Figure 2.8 to either remember or forget. The second gate is a layer with a sigmoid to decide whether to forget or remember and a tanh to fit the incoming input as part of a new cell state, \tilde{C}_t . The new update for the new cell state is multiplied with the a scaled input to determine how much impact the new cell state will have. The output of the second gate is then added with the old cell state. The third gate takes the new cell state squeezes it with tanh and outputs it to the next time step and as the output of that LSTM layer.

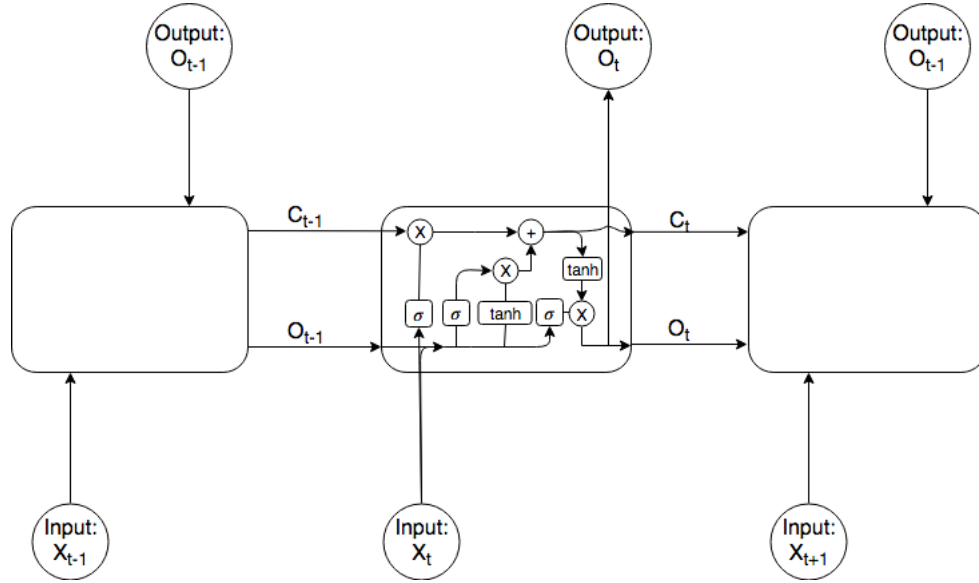


Figure 2.8: Long Short Term Memory Neural Network

2.3.5 Convolution Neural Network

The convolution neural network (CNN) performs a feature organization or reorganization of input values and name is based on its similarity in structure to a signal convolution [15] [29]. However, in the first layer instead of doing a mathematical convolution, it performs more of a mathematical correlation, and it could be said that it is a convolution with a kernel or filter that is invariant to inversion or flips. The layer thus changes the feature map with either a specified kernel or self-organizing kernel, and this will organize the input with its features. Then, the CNN has a layer for max pooling which is where the layer takes the maximum of features over its designated region. After pooling the necessary features, there is a flattening layer where the features are flatten and outputted as inputs to the following layer. The following layers can be comprised of either feedforward network, a recurrent network, etc. From there layers with different capabilities allow the network to become a deep learning network [29]. Deep learning allows computational models that are composed of multiple processing layers to learn representation of data on multiple levels of abstraction [17]. Krizhevsky published a 5 convolution layer and three feedforward layers in a image classification project and it is shown in Figure 2.9 [15].

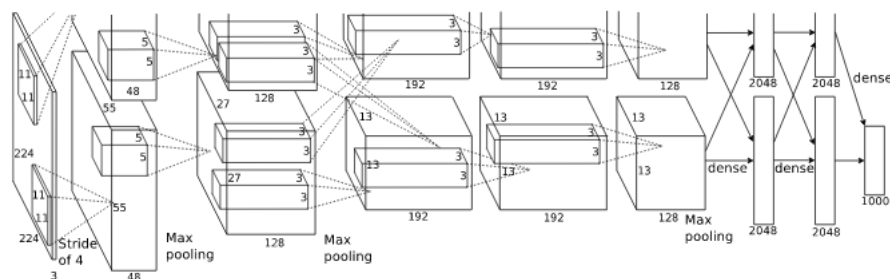


Figure 2.9: From Krizhevsky et al. (2012) [15] Deep Learning Convolutional Neural Network

Chapter 3

Previous Work

The first Active Noise Canceling system proposed produced a destructive anti noise signal through an adaptive algorithm called the Least Mean Squared algorithm [8]. The two main LMS algorithms for ANC systems have been the Normalized LMS (NLMS) and the filter-X LMS (FXLMS) [8]. Milani *et al.* [18] used the NLMS algorithm to determine the level of noise attenuation in a feedforward ANC system, a feedback ANC system, and a hybrid system consisting of both the feedforward and the feedback systems to achieve still higher attenuation. They used the datasets of the Volvo car cabin, the F-16 cockpit, and the Leopard Military vehicle noises from the Signal Processing Information Base (SPIB) as a basis for comparison. Their hybrid ANC system provided the highest attenuation using the NLMS algorithm. The results are in Table 3.1 below. The NAL stands for Noise Attenuated Level, FF stands for feedforward ANC systems, and the FB stands for feedback ANC systems. The NAL was calculated using the energy of the noise signals and the energy of the attenuated signals and their unit of measure is decibels (dB).

Table 3.1: Noise reduction using NLMS filters [18]

Structure	Volvo (dB)	F-16 (dB)	Leopard (dB)
NAL _{NLMS,FF}	17.6	21.62	34.81
NAL _{NLMS,FB}	3.7	6.23	8.7
NAL _{NLMS,hybrid}	17.61	22.25	34.88

While LMS algorithms have achieved successful noise attenuation, several experiments using neural networks have been completed to study alternative approaches. Neural Networks could be trained to predict the next sample or several samples instead being a step behind such as in the

adaptive filter approach of the LMS algorithms. Chen *et al.* [4] proposed using the multilayered perception (MLP), otherwise known as the multilayered feedforward network, to predict the anti-noise from the incoming noise signal. This was done theoretically with simplified models. Their focus was on the feedback ANC system. As described in the background section if it is assumed that the estimate of the secondary path transfer function, $\hat{S}(z)$, equals the secondary path, $S(z)$, a feedback ANC system becomes a feedforward system. Also if that secondary path is a pure delay, the system becomes a standard predictor. Chen *et al.* made a MLP with a 40 unit input layer, 40 unit hidden layer, and single unit output layer. They took the noise dataset of the destroyer engine room, and the cockpit of the F-16 fighter jet. Firstly, they concluded that the network was able to reduce the noise uniformly by 20 dB. Secondly, they concluded that traditional ANC systems must be tailored to each specific noise environment, but the neural network is capable of functioning in various noise environments. Lastly, conventional ANC systems are good at removing narrow band pure tone noises, but the proposed MLP can deal with broad-band noises.

Along with Chen *et al.* using a MLP, Na *et al.* [19] built on the work of Chen *et al.* and experimented with ANC using the MLP, the RNN or specifically Elman’s Neural Network, and the filtered-X LMS algorithm. Their ANC system was also a feedback ANC system with specified constraints like Chen. For their simulation and training, they took noise data from a moisture-removing machine. Their experiment yielded a 14.35 dB reduction from the FXLMS, a 20.83 dB reduction from their MLP predictor, and a 22.35 dB reduction from their RNN predictor.

In a more recent paper, Salmasi *et al.* [24] experimented with the MLP and the RNN to predict more noise signals with more variable frequencies for ANC applications. They took four noise environments from SPIB which were the destroyer engine room, the F-16 fighter jet, the military vehicle Leopard, and the cabin of a Volvo. These noise signals represented a wider range of frequencies amongst them. The destroyer engine room and the F-16 are a relatively high-frequency noise signal. The Leopard is a relatively mid-range frequency noise signal, and the Volvo is a relatively low-frequency noise signal. They trained on 2,000 samples from each noise environment and used the rest of the noise samples from a sound file for testing. They took three tests from each noise environment and compared which network type did the best. Their MLP predictor was structured with a 20 unit input layer, a 20 unit hidden layer, and a single unit output layer. Their RNN predictor was structured with a 20 unit input layer, a 20 unit hidden layer, and a single unit output layer. Their activation function for the hidden layer for both of the architectures were the

hyperbolic tangents, and the activation function for the output layers for both architectures was a pure linear function. Their experiment focused on the feedback ANC system with the same criteria as Chen previously. Salmasi *et al.* concluded that the RNN architecture is a better predictor than the MLP architecture for ANC applications. Their results are tabulated in Tables 3.2 to 3.5 below.

The work of Salmasi *et al.* provided a good launching point for this experiment. It provided details and versatility not only to replicate or emulate their experiment but an avenue to build and improve ANC applications using Neural Networks with more recent applications and techniques. The results from the outputs of this experiment’s CNN predictor and the LSTM predictor are compared with their results to see if they serve as better architectures for performing active noise cancellation.

Table 3.2: RNN and MLF noise reduction for F16

	Noise Attenuation (dB)	
	Feed-Forward Neural Network	Recurrent Neural Network
1 st test	23.7335	25.1345
2 nd test	24.75	25.5037
3 rd test	24.478	25.3954

Table 3.3: RNN and MLF noise reduction for destroyer

	Noise Attenuation (dB)	
	Feed-Forward Neural Network	Recurrent Neural Network
1 st test	22.9461	23.9076
2 nd test	23.5132	24.0004
3 rd test	22.9781	23.5339

Table 3.4: RNN and MLF noise reduction for Volvo

	Noise Attenuation (dB)	
	Feed-Forward Neural Network	Recurrent Neural Network
1 st test	47.1487	49.5744
2 nd test	45.2066	48.2081
3 rd test	47.8032	50.6561

Table 3.5: RNN and MLF noise reduction for Leopard

	Noise Attenuation (dB)	
	Feed-Forward Neural Network	Recurrent Neural Network
1 st test	40.8741	42.79
2 nd test	41.2916	43.2439
3 rd test	40.4198	42.6719

Chapter 4

Experiment

Four datasets were pulled from the Signal Processing Information Base (SPIB)[13] which are noise environment recordings from a Destroyer Engine Room, the cockpit of an F16 Jet traveling at 500 knots between altitudes 300 to 600 feet, the cabin of military vehicle Leopard traveling at 70 km/hr, and the interior cabin of a Volvo traveling 120 km/hr in the rain. All the noise signals are 235 seconds long with sampling rate of 19.98 kHz, and had analog to digital conversion of 16 bit precisions. All of the noise signals are upsampled to 44.1 kHz. The noise signals are all normalized by dividing 2^{15} because the quantization of the signals are 16 bits. The noise sets are also chosen to try results with different frequencies of noise. The destroyer engine and the F-16 jet cockpit noises are relatively high frequency noises. The Leopard produces mid ranged frequency noises, and the Volvo produces a relatively low-frequency noises.

The first experiment setup was to replicate results of an LMS approach. The noise signals are each combined with a reference audio or music signal. The LMS algorithm inputs the combined signals and the target output of the LMS algorithm is to output the audio or music signal. The algorithm needed a reference signal to target for and having zeros as a reference signal provided poor performance for the algorithm. The need to upsample the noise signals was required here where the audio file had a sample rate of 44.1 kHz while the noise signals all had 19.98 kHz. The results here are compared with the results of the neural networks.

The remaining experimental setups used different neural network architectures. The neural networks were setup was done through Keras [5] using Tensorflow [1] as the backend. This allowed GPU implementation in training the neural networks. The code was written in Python [9] and the

datasets was prepared using numpy [20] arrays. The neural networks were trained as a predictor. The neural network would accept N number of samples as its input and by using the N number of samples, the neural network would output the $(N+1)$ 'th sample. The inputs to the neural networks were all set to take in 20 samples. The training set was comprised of 2,000 samples of a noise signal, and the test set was the 10,000 samples from the rest of the signal. The training was done in epoch iterations, and all the networks were trained to learn and predict all four of the noise signals. To validate results of this experiment, three 10,000 samples were taken from three separate places in the future part of the signal for each noise signal. This would ensure that the testing samples were never seen by the network. The samples were run through the network to check to see the amount of noise reduction the network could achieve. This technique of validation is referred to as the out-of-sample validation as the dataset was all sequential and time dependent. The optimizer chosen for this experiment was the Adam optimizer. While Adam is similar to stochastic gradient descent (SGD), Adam is a first order gradient based optimizer that is computationally efficient, little memory requirements, is invariant to the diagonal rescaling of gradients, and does well with large datasets and/or parameters [14]. Hence, Adam was chosen over SGD to optimize the networks.

Four neural networks were trained to attenuate the noise signals by outputting the 180 out-of-phase noise signal. The first neural network is the multilayered feedforward (MLF) neural network, also known as the multilayered perceptron (MLP). The structure of this neural network has a 20 unit input layer, a 20 unit hidden layer, and a 1 unit output layer. All of these layers are fully connected. The activation function used in the output layer is a linear function, and the activation function used in the hidden layer is the hyperbolic tangent (\tanh) function because of its range from -1 to 1.

The second neural network is the recurrent neural network or the Elman's network. This network is designed to have 20 inputs, 20 hidden layer neurons, and 1 neuron in the output layer. The activation function for the output layer is a linear function, and the activation function for the hidden layer is the \tanh function.

The third neural network is the Long Short Term Memory Neural Network (LSTM). The activation function in its hidden layer is the \tanh function, and the output layer's activation function is a linear function. This network takes in 20 inputs and outputs a single predicted value at each sample.

The fourth neural network is the Convolutional Neural Network. The structure of the CNN

is that it has an input convolutional layer with a linear activation function, a second convolutional layer with tanh activation function, a max pooling layer that grabs the maximum feature, a flattening layer, and a single unit output layer with a linear activation function. The number of kernels or filters for the two convolutional layers are 4 chosen to match four different noise environments and the length of those kernels is 5.

The noise attenuation of the experiment is calculated as in earlier cited works using the following equation [18] [24]:

$$E = \sum_{-n}^n |x[n]|^2 \quad (4.1)$$

E is energy of a discrete signal. E_{input} is the energy of the input signal and E_{atten} is the energy of the attenuated signal.

$$NoiseAttenuation = 10\log_{10}\left(\frac{E_{input}}{E_{atten}}\right) \quad (4.2)$$

The attenuation takes the energy of the input noise and divides it by the energy of the attenuated noise. The results are converted in decibels.

Chapter 5

Results

The ANC results using NLMS are given in Table 5.1 below. The results are comparative to previous findings using traditional LMS techniques for ANC. While noise attenuation was evident with the high-frequency noise signals, NLMS performed worse in these environments versus the relatively lower frequency noise environments of the Leopard and the Volvo. The results show that NLMS are constrained to a relatively narrow band of frequencies and that it has relatively weak noise reduction at higher frequencies.

Table 5.1: NLMS noise reduction

	NLMS (dB)
Destroyer	9.961
F-16	9.961
Leopard	12.786
Volvo	12.028

The results for the MLP predictor, Table 5.2, were similar to the MLP predictor of Salmasi *et al.* except that the predictors for the destroyer and F-16 noise environment were higher while the Leopard and the Volvo stayed relatively the same.

Table 5.2: dB reduction using MLF

	MLF Result			Salmasi <i>et al.</i>		
	1st test	2nd test	3rd test	1st test	2nd test	3rd test
Destroyer	28.517	28.301	28.615	22.9461	23.5132	22.9781
F-16	26.986	28.278	27.102	23.7335	24.75	24.478
Leopard	41.144	40.580	40.777	40.8741	41.2916	40.4198
Volvo	46.974	47.782	45.258	47.1487	45.2066	47.8032

The results for the RNN predictor, Table 5.3, were as expected except for the Volvo values. The RNN was able to exceed the values from the MLP predictors except the Volvo values. A likely reason for such a case is that during training the gradient descent is stuck in a local minima.

Table 5.3: dB reduction using RNN

	RNN results			Salmasi <i>et al.</i>		
	1st test	2nd test	3rd test	1st test	2nd test	3rd test
Destroyer	28.933	28.821	29.237	23.9076	24.0004	23.5339
F-16	27.597	28.981	27.740	25.1345	25.5037	25.3954
Leopard	42.641	42.331	42.482	42.79	43.2439	42.6719
Volvo	45.399	46.320	44.431	49.5744	48.2081	50.6561

The results for the LSTM predictor, Table 5.4, demonstrate that the results for the destroyer and F-16 noise environment are better, but the results for the Leopard and the Volvo remain close with RNN predictor of Salmasi *et al.*. The LSTM predictor did better than this experiment's MLP predictor except for the Destroyer and the F-16 noise environment where they both remained relatively similar. The output of the LSTM are compared to the noise environments separately in Figures 5.1 to 5.12 below. The residual signal which are added components of the LSTM output and the signal from each of the noise environments are presented after each corresponding LSTM output. From Figures 5.1 and 5.2, when the two signals are added via superposition, the resulting residual signal is produced shown in Figure 5.3. For the LSTM examples, the figures are organized to provide a step by step visual representation. The figures are organized in the following order: Destroyer, F-16, Leopard, and Volvo. As the results suggest from Table 5.4, the residual signal shown for Destroyer and the F-16 is thicker than the residual signals of the Leopard and the Volvo. This is congruent with the dB reduction shown in Table 5.4.

Table 5.4: dB reduction using LSTM

	LSTM results			Salmasi <i>et al.</i> RNN		
	1st test	2nd test	3rd test	1st test	2nd test	3rd test
Destroyer	28.440	27.825	28.700	23.9076	24.0004	23.5339
F-16	27.0589	28.227	27.075	25.1345	25.5037	25.3954
Leopard	42.517	42.031	42.285	42.79	43.2439	42.6719
Volvo	49.300	49.441	47.193	49.5744	48.2081	50.6561

The CNN predictor, results shown on Table 5.5, revealed better performance than all the networks in this experiment and the results presented in the paper of Salmasi *et al.* The CNN predictions out perform all the previous networks' noise reduction, and it does so encompassing a

wide range of frequency bandwidths. For the CNN results, the plot of the CNN output and the signal of the noise environment is plotted together to give a visual representation that the CNN output is the anti-noise signal of the corresponding noise signal. This is shown in Figures 5.13 to 5.16. The corresponding residual signals are presented in Figures 5.17 to 5.20. The figures are organized showing the concatenated plots of the CNN output and noise signals. The figures are organized showing in the following order: Destroyer, F-16, Leopard, and Volvo. They are also separated by showing all the concatenated plots first then the all the figures for the residual signals. Similar to the results of the LSTM, the residual signals of the Destroyer and F-16 is thicker than the residual signals of the Leopard and the Volvo.

Table 5.5: dB reduction using CNN

	CNN results			Salmasi <i>et al.</i> RNN		
	1 st test	2 nd test	3 rd test	1 st test	2 nd test	3 rd test
Destroyer	34.732	34.990	35.443	23.9076	24.0004	23.5339
F-16	32.917	34.333	33.152	25.1345	25.5037	25.3954
Leopard	44.814	46.418	45.542	42.79	43.2439	42.6719
Volvo	51.989	48.220	50.822	49.5744	48.2081	50.6561

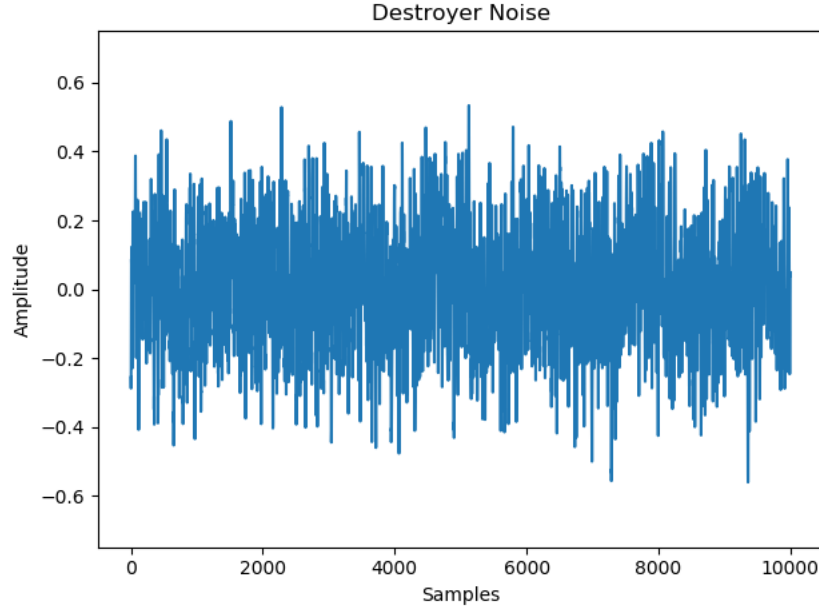


Figure 5.1: Destroyer Noise Signal

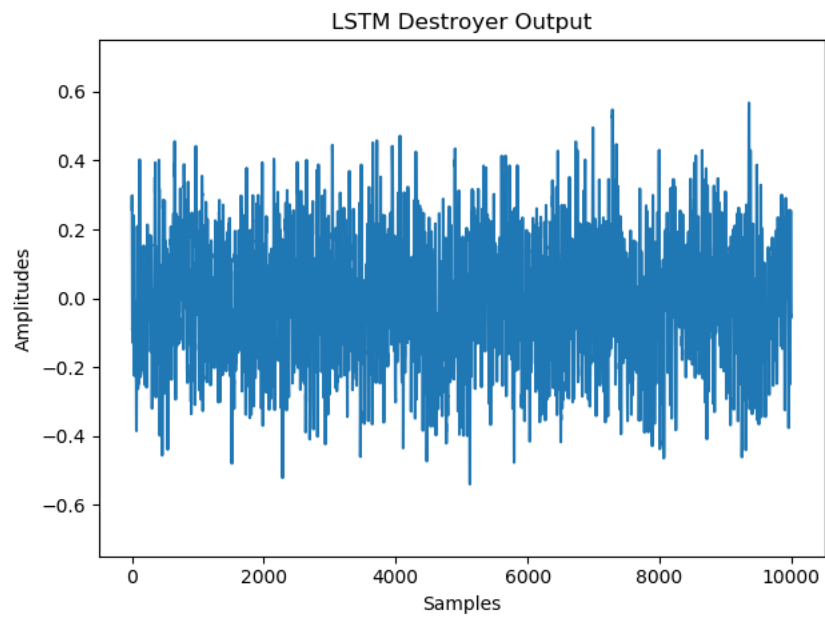


Figure 5.2: LSTM Destroyer Output

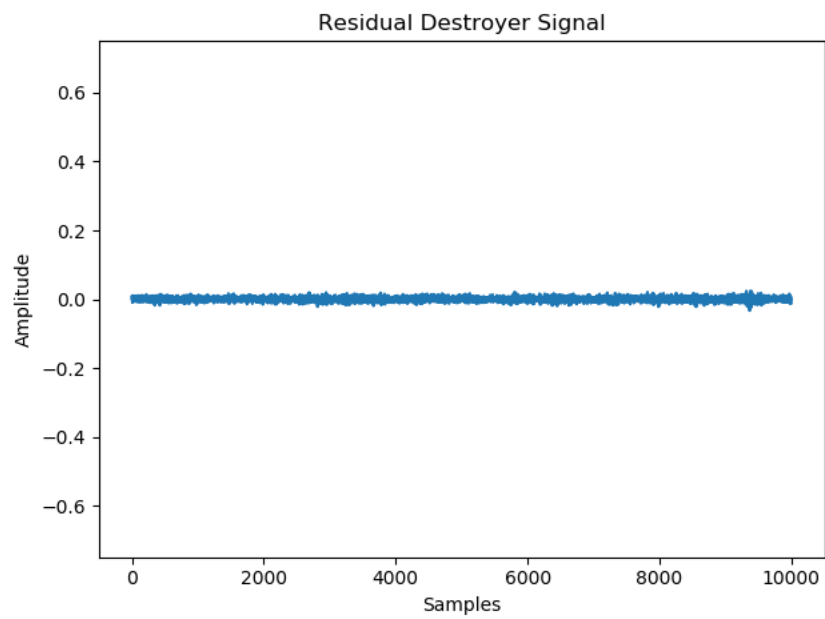


Figure 5.3: LSTM Residual Destroyer Noise

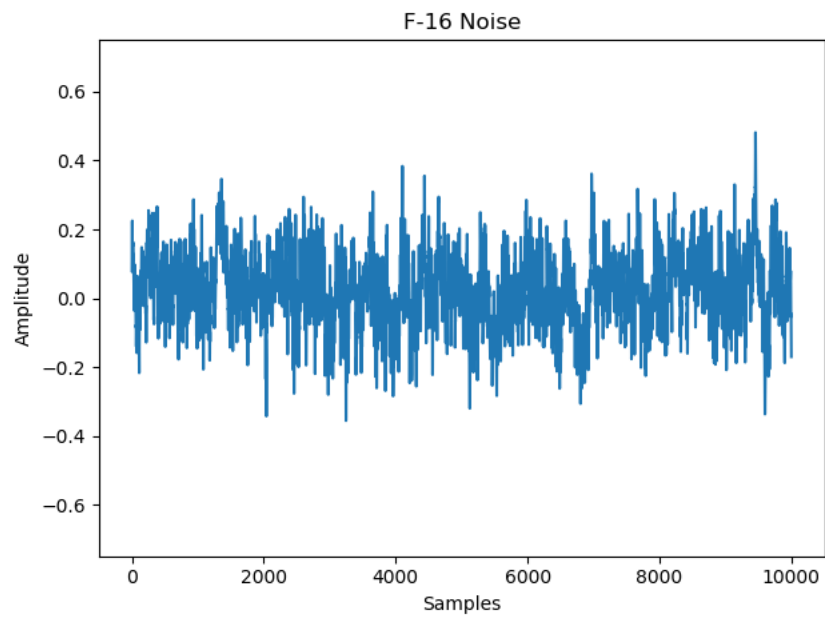


Figure 5.4: F-16 Noise Signal

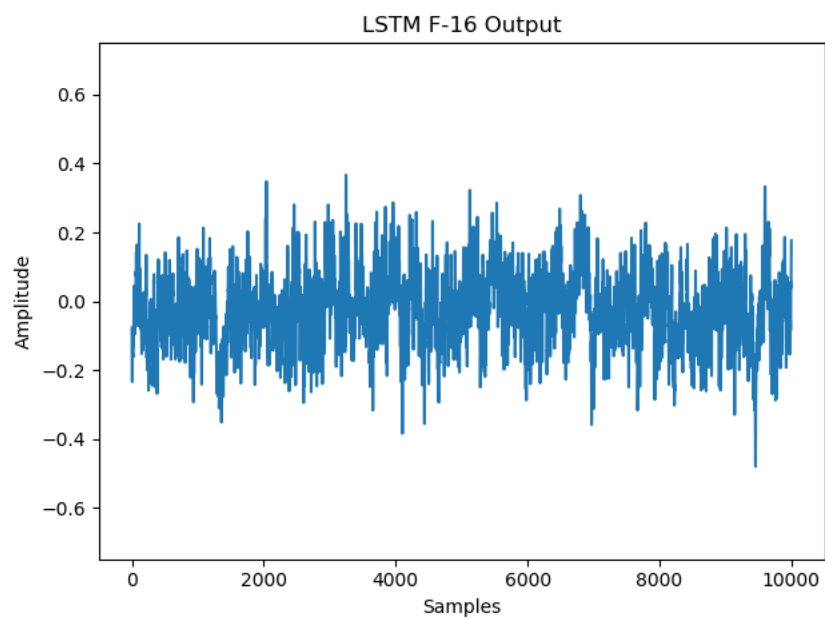


Figure 5.5: LSTM F-16 Output

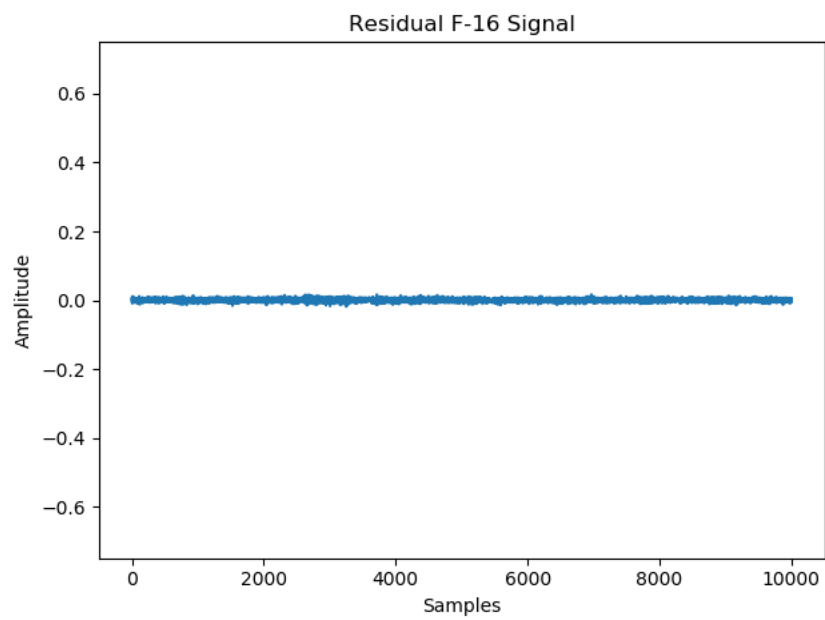


Figure 5.6: LSTM Residual F-16 Noise

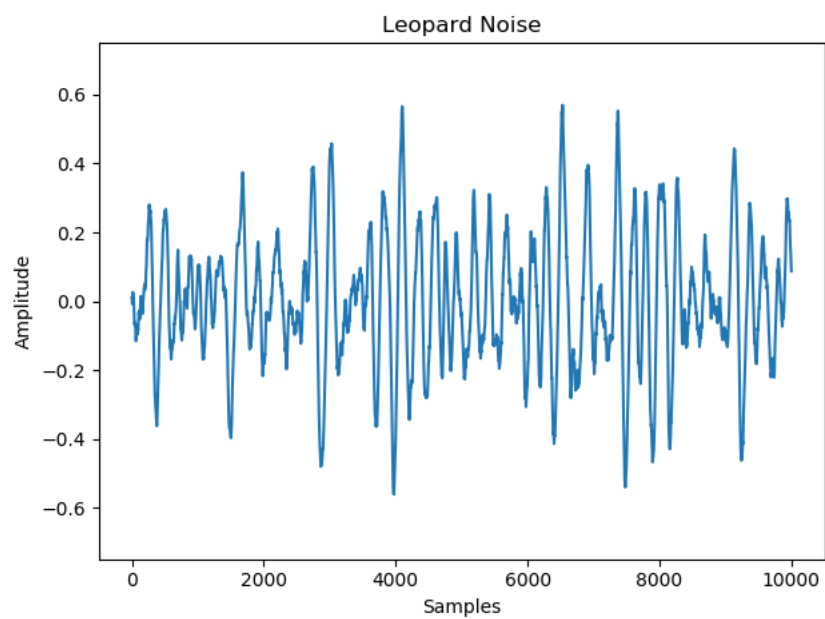


Figure 5.7: Leopard Noise Signal

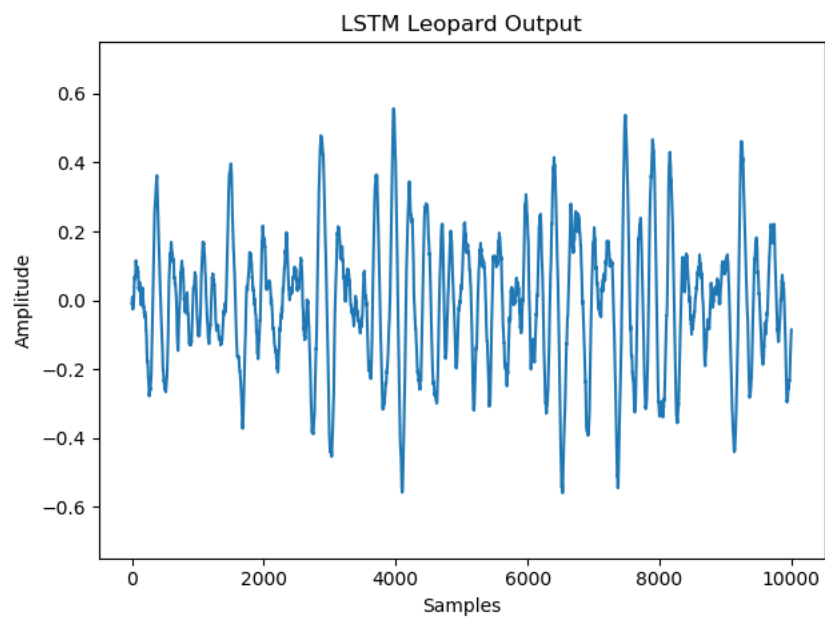


Figure 5.8: LSTM Leopard Output

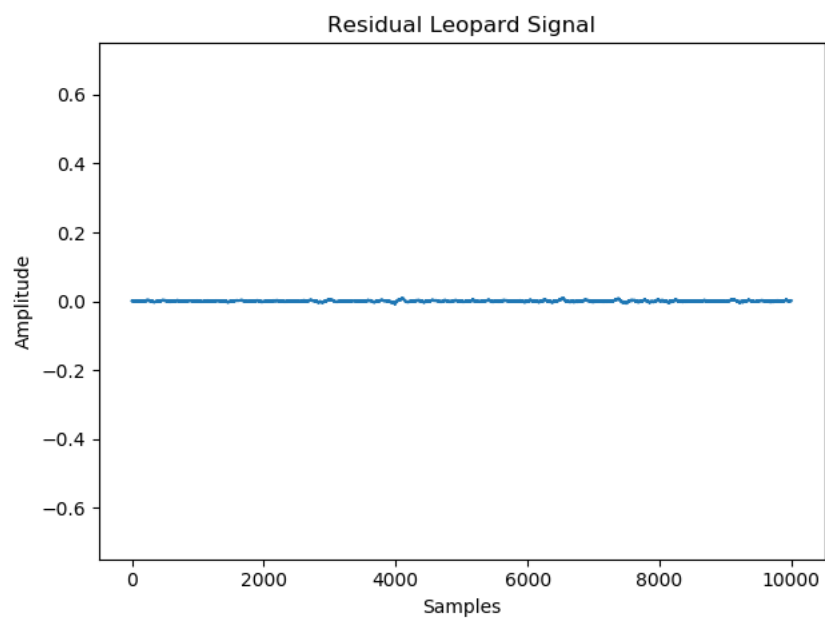


Figure 5.9: LSTM Residual Leopard Signal

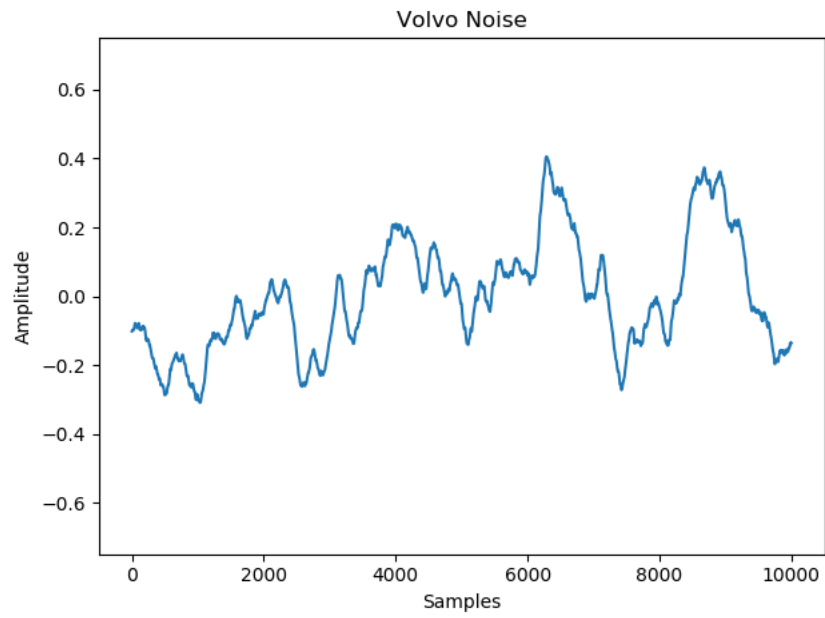


Figure 5.10: Volvo Noise Signal

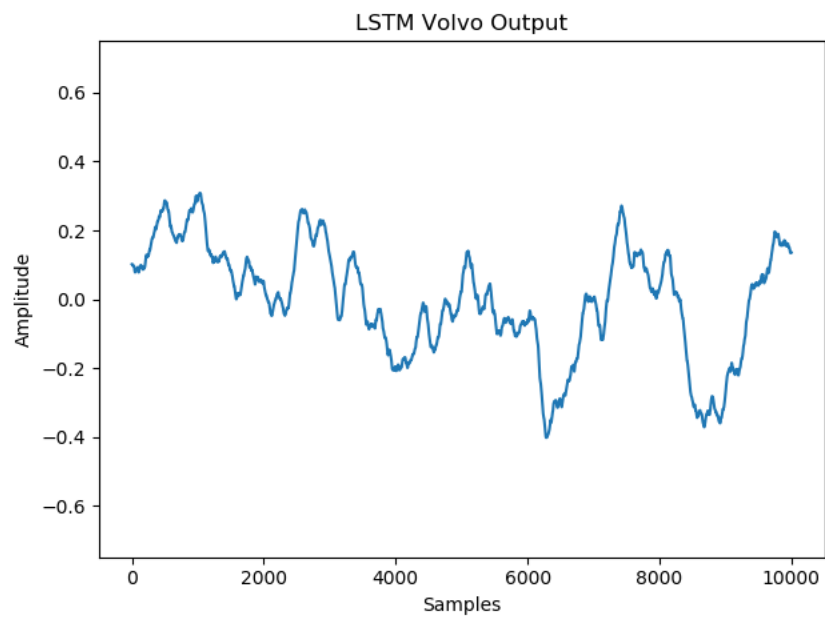


Figure 5.11: LSTM Volvo Output

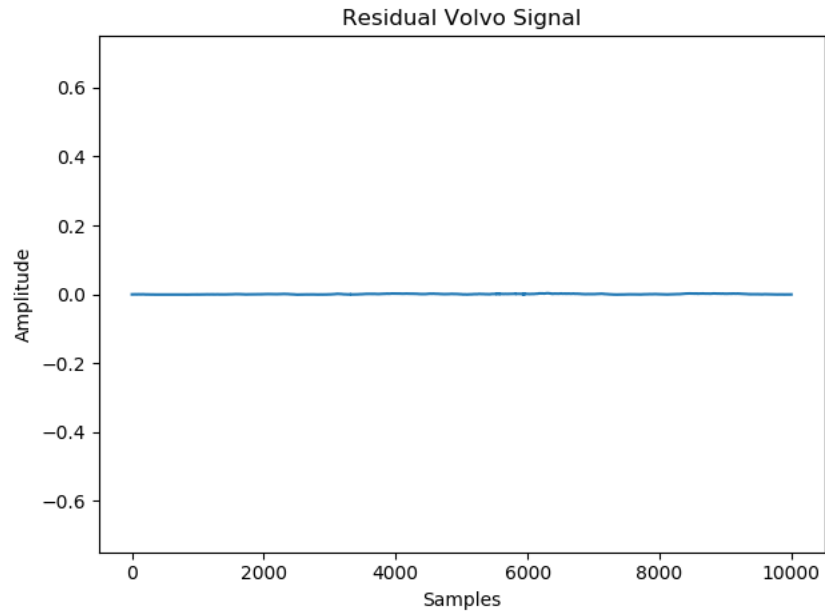


Figure 5.12: LSTM Residual Volvo Noise

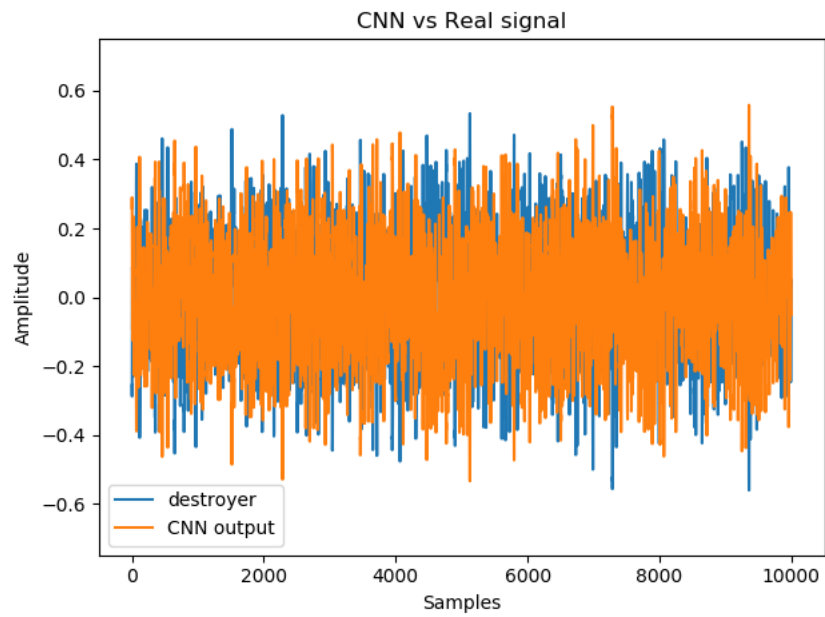


Figure 5.13: CNN anti noise signal with Destroyer noise signal

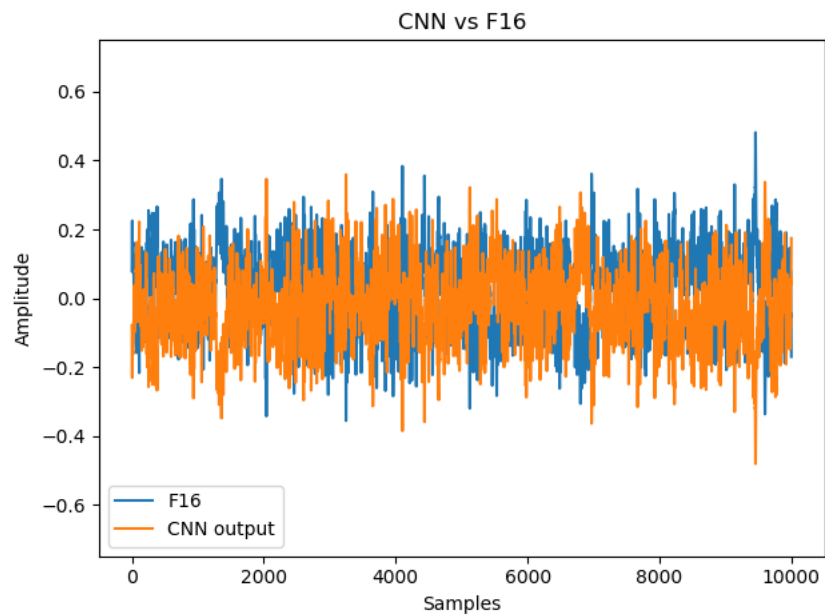


Figure 5.14: CNN anti noise signal with F-16 noise signal

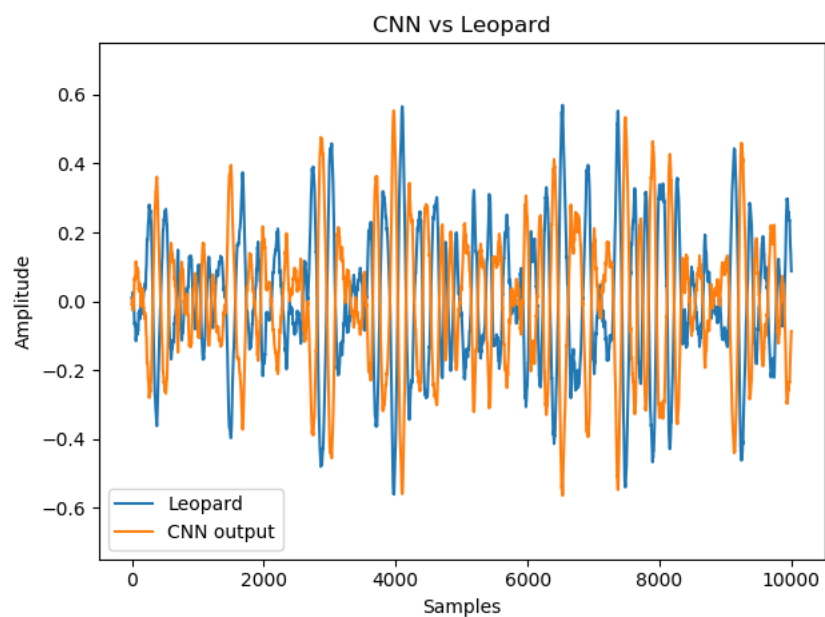


Figure 5.15: CNN anti noise signal with Leopard noise signal

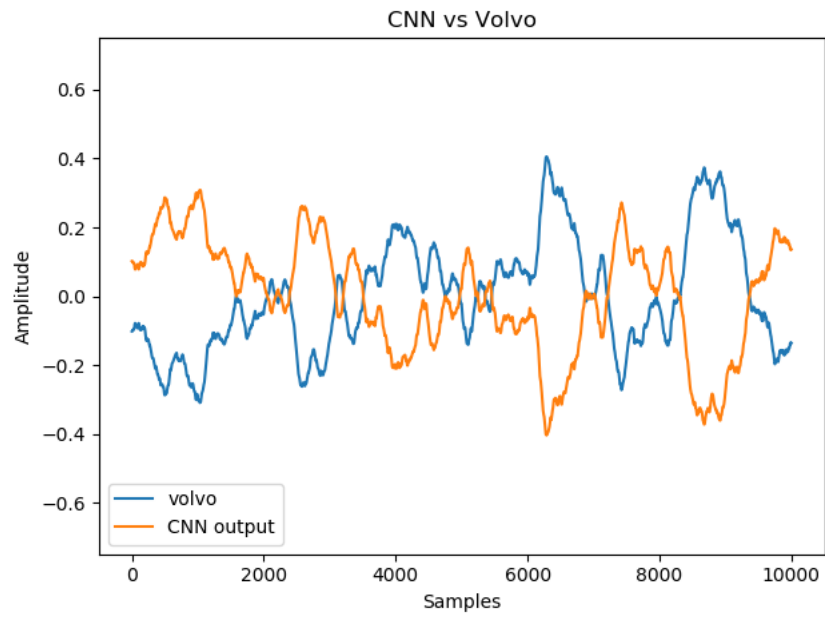


Figure 5.16: CNN anti noise signal with Volvo noise signal

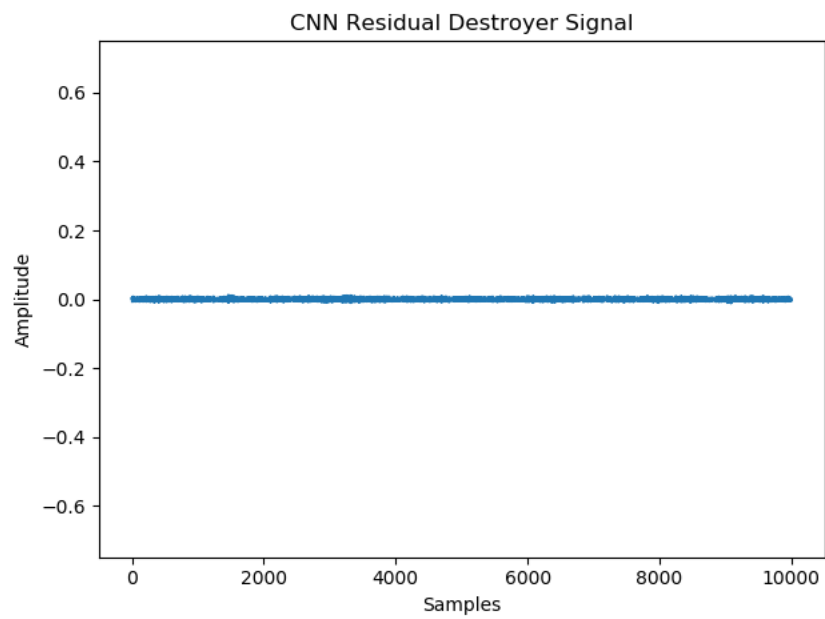


Figure 5.17: CNN Residual Destroyer Noise

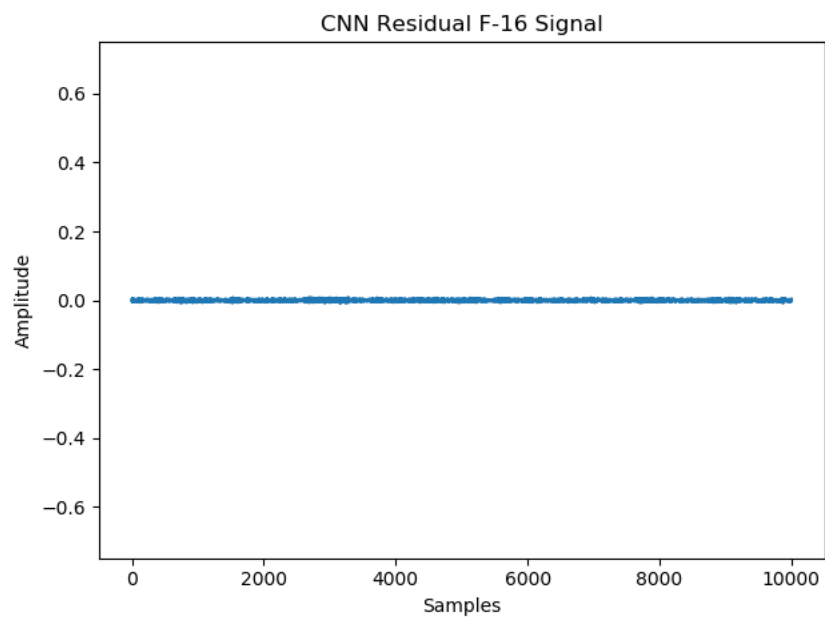


Figure 5.18: CNN Residual F-16 noise

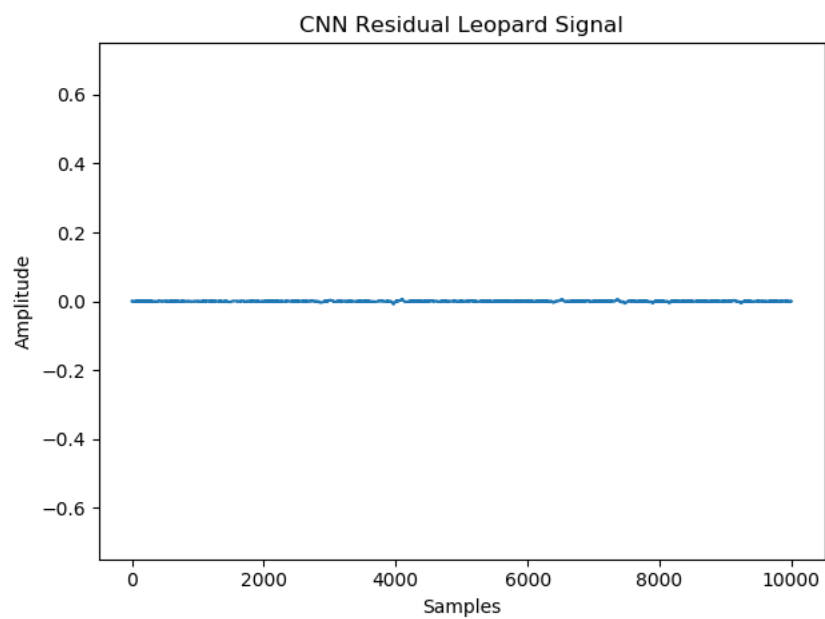


Figure 5.19: CNN Residual Leopard Noise

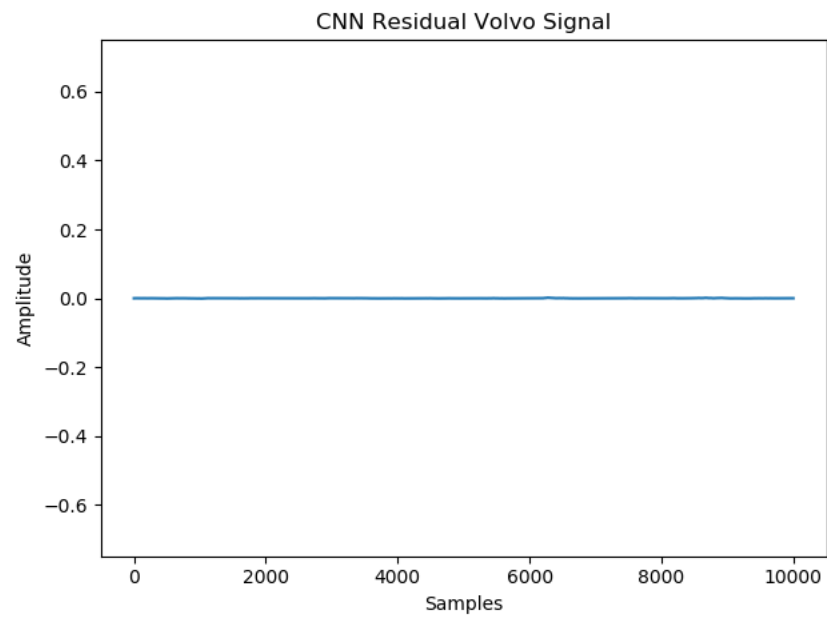


Figure 5.20: CNN Residual Volvo Noise

Chapter 6

Conclusion

All the neural networks were able to perform better than the experimented and published values of the NLMS and the FXLMS algorithms. While the LSTM did perform better in the noise environments of the Destroyer and the F-16 than other published networks, the attenuation at those environment were similar to the attenuation achieved in the MLP and the RNN. While the LSTM predictor achieves the proper noise attenuation, the architecture itself performs similarly to other networks. One of the possible reason for the LSTM performing similarly to the MLP and the RNN is that its strength is in the field of speech recognition and language models where long term dependencies are much more prevalent. In other words, the network's strength is in recognizing and retaining long term dependent applications. In this experiment, the predictor was only required to remember the very next sample because the simplified model's delay was by one sample. When the problem statement become more complex and the delay becomes even more significant, the long term dependency would have more affect on the problem set and the predictor would have the potential to perform better than the MLF and the RNN. Out of all the networks discussed in this experiment and after all the comparisons with previous published results, this paper's convolutional neural network performed the best noise attenuation. One of the reasons for the success of the CNN in this application is potential in the fact that the kernels are also trained.

An area for future work is to incorporate additional real world elements to the ANC system model. The input signal could have more delay than one sample and outside elements could be added that affect the incoming noise signal beyond a pure delay. The system could also incorporate nonlinearity in that the secondary path may be modeled as a nonlinear path. Other future extensions

could include more complex environments in which the noise environment has more variability present in the noise signal. For example, the amplitude of the noise might change drastically with the frequency of the noise. Extensions such as these would test the neural networks capabilities in more complex instances and scenarios.

Bibliography

- [1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.
- [2] Riyanto Bambang. Active noise cancellation using recurrent radial basis function neural networks. In *Circuits and Systems, 2002. APCCAS'02. 2002 Asia-Pacific Conference on*, volume 2, pages 231–26A. IEEE, 2002.
- [3] Cheng-Yuan Chang, Antonius Siswanto, Chung-Ying Ho, Ting-Kuo Yeh, Yi-Rou Chen, and Sen M Kuo. Listening in a noisy environment: Integration of active noise control in audio products. *IEEE Consumer Electronics Magazine*, 5(4):34–43, 2016.
- [4] Casper K Chen and Tzi-Dar Chiueh. Multilayer perceptron neural networks for active noise cancellation. In *Circuits and Systems, 1996. ISCAS'96., Connecting the World., 1996 IEEE International Symposium on*, volume 3, pages 523–526. IEEE, 1996.
- [5] François Chollet et al. Keras. <https://keras.io>, 2015.
- [6] Kunal Kumar Das and Jitendriya Kumar Satapathy. Legendre neural network for nonlinear active noise cancellation with nonlinear secondary path. In *Multimedia, Signal Processing and Communication Technologies (IMPACT), 2011 International Conference on*, pages 40–43. IEEE, 2011.
- [7] Jyoti Dhiman, Shadab Ahmad, and Kuldeep Gulia. Comparison between adaptive filter algorithms (lms, nlms and rls). *International Journal of Science, Engineering and Technology Research (IJSETR)*, 2(5):1100–1103, 2013.
- [8] SJ Elliott and PA Nelson. The active control of sound. *Electronics & communication engineering journal*, 2(4):127–136, 1990.
- [9] Python Software Foundation. Python language reference, version 3.6, 2018. Available at <http://www.python.org>.
- [10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.

- [11] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2017.
- [12] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.
- [13] Don H. Johnson. Signal processing information databse. Data retrieved from website <https://spib.linse.ufsc.br>.
- [14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.
- [15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.
- [16] Sen M Kuo and Dennis R Morgan. Review of dsp algorithms for active noise control. In *Control Applications, 2000. Proceedings of the 2000 IEEE International Conference on*, pages 243–248. IEEE, 2000.
- [17] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- [18] Ali A Milani, Govind Kannan, and Issa MS Panahi. On maximum achievable noise reduction in anc systems. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 349–352. IEEE, 2010.
- [19] Kyungmin Na and Soo-Ik Chae. Single-sensor active noise cancellation using recurrent neural network predictors. In *Neural Networks, 1997., International Conference on*, volume 4, pages 2153–2156. IEEE, 1997.
- [20] Travis E. Oliphant. A guide to numpy, 2006.
- [21] Alan V Oppenheim, Ehud Weinstein, Kambiz C Zangi, Meir Feder, and Dan Gauger. Single-sensor active noise cancellation. *IEEE Transactions on Speech and Audio Processing*, 2(2):285–290, 1994.
- [22] L Poole, G Warnaka, and R Cutter. The implementation of digital filters using a modified widrow-hoff algorithm for the adaptive cancellation of acoustic noise. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’84.*, volume 9, pages 215–218. IEEE, 1984.
- [23] David E Rumelhart and James L McClelland. Parallel distributed processing: explorations in the microstructure of cognition. volume 1. foundations. 1986.
- [24] Mehrshad Salmasi, Homayoun Mahdavi-Nasab, and Hossein Pourghassem. Comparison of feed-forward and recurrent neural networks in active cancellation of sound noise. In *Multimedia and Signal Processing (CMSP), 2011 International Conference on*, volume 2, pages 25–29. IEEE, 2011.
- [25] Robert J Schalkoff. *Artificial neural networks*, volume 1. McGraw-Hill New York, 1997.
- [26] Scott D Snyder and Nobuo Tanaka. Active control of vibration using a neural network. *IEEE Transactions on Neural Networks*, 6(4):819–828, 1995.

- [27] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.
- [28] Bernard Widrow, John R Glover, John M McCool, John Kaunitz, Charles S Williams, Robert H Hearn, James R Zeidler, JR Eugene Dong, and Robert C Goodlin. Adaptive noise cancelling: Principles and applications. *Proceedings of the IEEE*, 63(12):1692–1716, 1975.
- [29] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.