THE COOPER UNION
ALBERT NERKEN SCHOOL OF ENGINEERING

# A Deep Partitioned Autoencoder for De-Noising Live Audio

by

Ethan Lusterman

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

September 2016

Professor Sam Keene, Advisor

THE COOPER UNION FOR THE ADVANCEMENT OF SCIENCE AND ART

ALBERT NERKEN SCHOOL OF ENGINEERING

This thesis was prepared under the direction of the Candidate's Thesis Advisor and has received approval. It was submitted to the Dean of the School of Engineering and the full Faculty, and was approved as partial fulfillment of the requirements for the degree of Master of Engineering.

Dean, School of Engineering    Date

Prof. Sam Keene, Thesis Advisor    Date

# Acknowledgements

Ack example

# Abstract

Abstract

# Contents

# List of Figures

Table of Nomenclature

# 1 Introduction

Advances in smartphone technology have led to smaller devices with more powerful audio hardware, allowing for common consumers to make higher quality recordings. However, recorded speech and music are subject to noisy conditions, often hampering intelligibility and listenability. The goal of denoising audio recordings is to improve intelligibility and perceived quality. A variety of applications of audio denoising exist, including listening to a recording of a band or an artist's live performance in a noisy crowd, or listening to a recorded conversation or speech under noisy conditions.

A common technique for denoising involves the use of deep neural networks (DNN). [PARIS] Advances in parallel graphics processing units (GPU) and in machine learning algorithms have allowed for training deeper networks faster, utilizing more hidden layers with more neurons.

Prior work in denoising audio has involved access to noise-free training data. Since common consumers do not often have access to clean audio, we seek to denoise without the use of clean audio.

In this thesis, we compare several neural network architectures and problem scenarios, ranging from data input types, level of noise, depth of network, training objectives, and more. In Chapter 2, we present background information on machine learning, neural networks, and signal processing as well as prior work in audio denoising. In Chapter 3, we detail all considered network architectures. In Chapter 4, we compare results from different data inputs, levels of noise, network architectures, and training objectives and discuss methods of evaluation. Finally, we make conclusions and recommendations for future work in Chapter 5.

# 2 Background

## 2.1 Machine Learning

Machine learning involves the use of computer algorithms to make decisions based on training data. Generally, this falls into categorizing input data (classification) or determining a mathetmatical function to determine a continuous output given an input (regression). Popular classification examples include recognizing handwritten digits (MNIST) as well as determining whether an image contains a cat or a dog. (REF) An example of a regression problem is determining the temperature given a set of input features (humidity, latitude, longitude, date, etc.).

Problems where training data contain input data vectors as well as the correct output vectors (targets) are known as supervised learning problems. Training a model to denoise audio where noise was introduced to the clean audio would be a supervised learning problem. On the other hand, training a model to denoise audio where the underlying clean signal is not known is an unsupervised learning problem. Different loss functions and neural network architectures can be exploited to accomplish denoising without the clean data.

For the purposes of this thesis, we use machine learning to determine an underlying nonlinear function that removes noise from time slices of audio (i.e. regression). These slices can then be pieced back together through overlap-add resynthesis.

### 2.1.1 Regression

### 2.1.2 Overfitting and Curse of Dimensionality

### 2.1.3 Loss functions and Regularization

### 2.1.4 Gradient Stuff?

## 2.2 Neural Networks

### 2.2.1 Dense Layer

### 2.2.2 Convolutional Layer

### 2.2.3 Nonlinearity Choice

## 2.3 Signals and Systems

Domain knowledge of discrete audio signals and systems better informs our decisions for an audio denoising system, so some background information on signals and systems as it pertains to this thesis is detailed below.

### 2.3.1 Signals

We deal exclusively with discrete-time audio signals in this thesis. A discrete-time audio signal $x[n]$ is represented as a sequence of numbers (samples), where each integer-valued slot $n$ in the sequence corresponds to a unit of time based on the sampling frequency $f_s$. This comes from sampling the continuous-time audio signal $x_c(t)$:

$$x[n] = x_c(nT) \tag{1}$$

where $T = 1/f_s$. For example, a 1-second speech signal sampled at 8kHz has 8000 samples. Furthermore, digital signals also have discrete valued sample amplitudes. For the purposes of this thesis, the bit depths of computers we use for analysis are high enough to allow for perfect reconstruction between continuous-time signals and digital signals.

We also assume signals collected have been properly sampled according to the Nyquist-Shannon sampling theorem, which states that a discrete-time signal must be sampled at at least twice the highest frequency present in the signal to prevent aliasing of different frequencies. For example, speech signals genearlly have information up to 8kHz, so many speech signals are sampled at 16kHz. Music is more complex in that signals often span up to about 20kHz, so CD quality recordings are often sampled at 44.1kHz or higher. For this thesis, we use recordings sampled at 44.1kHz or lower.

### 2.3.2 Convolution

The discrete-time convolution operation takes two sequences $x[n]$ and $h[n]$ and outputs a third sequence $y[n] = x[n] * h[n]$:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \tag{2}$$

Convolution is commutative, so $x[n] * h[n] = h[n] * x[n]$ holds true.

A linear, time-invariant (LTI) system is characterized by its impulse response $h[n]$, which allows us to determine samples $y[n]$ when $x[n]$ is subject to $h[n]$. For the purposes of this thesis, our underlying clean signal $x[n]$ might be subject to the conditions of an acoustic environment $h[n]$ and crowd noise $N[n]$:

$$y[n] = h[n] * x[n] + N[n] \tag{3}$$

In this scenario, our system would attempt to recover $h[n] * x[n]$ and possibly even $x[n]$ if the acoustic environment were deemed "noisy enough" due to echo and reverberation.

One of our proposed systems also incorporates convolutional neural networks (CNN) which use convolutions between frames of samples instead of simple linear combinations (discussed later).

### 2.3.3 Frequency Transforms

In some of our proposed systems, we use a frequency transformed version of the input signal as a preprocessing step to the system input. While no new information is gained from transforming the input, networks often respond better to determining the value of the magnitude of varying frequencies at a time slice instead of the individual time samples.

The frequency transform we use in this thesis is the discrete-time Fourier transform (DTFT). A sequence of $N$ discrete-time samples is transformed into another sequence of $N$ samples where each index then corresponds to a frequency bin. The DTFT $X[k]$ of a signal $x[n]$ is given by the following:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \tag{4}$$

where the twiddle factor $W_N$ is given by $W_N = e^{-j(2\pi/N)}$. Then the reconstruction of $x[n]$ from $X[k]$ is given by:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn} \tag{5}$$

In this thesis, we also exploit the main duality between the time and frequency domain using the convolution theorem, which states that convolution in time is equivalent to multiplication in frequency and vise versa:

$$\mathscr{F}\{h[n] * x[n]\} = H[k]X[k] \tag{6}$$

$$\mathscr{F}^{-1}\{H[k] * X[k]\} = h[n]x[n] \tag{7}$$

This allows us to effectively treat our network as a non-linear filter that can denoise small time/frequency slices of our noisy signal, which can then be pieced back together using overlap-add resynthesis. We detail this in the next section.

### 2.3.4 Windowing and Perfect Reconstruction

To window a signal is to multiply a window function $w[n]$ by the frame, i.e. $w[n]x[n]$ over the frame length $N$. Because we are training a network to denoise small segments of a larger audio signal, we window the signal segments. This accommodates the finite-length requirement of the DTFT and helps to prevent spectral leakage. [DSPBOOK]

Also, to be able to properly reconstruct our signal, we use a window function and corresponding overlapping frame percentage to accomplish perfect reconstruction. The corresponding overlapping frame percentage is set such that the window sums to a constant for all time. For example, a rectangular window $w[n] = 1$ over an interval of length $N$ has an overlap of 0% to sum to a constant 1 for all time. Another popular window is the Hanning window, defined over an interval $N$ by the following:

$$w[n] = \frac{1}{2}\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) \tag{8}$$

For the Hann window, the perfect reconstruction overlap is a frame length of $N = 50\%$.

### 2.3.5 Noise and Signal-to-Noise Ratio

Since we are trying to denoise audio signals, we must discuss how we measure noise. One of the most common measures of degradation of signal quality from additive noise is signal-to-noise ratio (SNR), defined as the ratio of signal variance to noise variance. [DSP] For the signal $y[n] = x[n] + N[n]$, where $x[n]$ is the signal of interest and $N[n]$ is the additive noise, the SNR is defined as

$$SNR = \frac{\sigma_x^2}{\sigma_n^2} \tag{9}$$

where $\sigma^2$ refers to the variance of the signal in question over some time interval. For the purposes of this thesis, we achieve desired a desired SNR for a simulation by scaling the noise to match the variance to the signal, then scaling the noise or the signal to achieve the desired SNR.

### 2.3.6 Magnitude and Phase Spectrum

# 3 System Description

# 4 Results

# 5 Conclusions and Future Work

## 5.1 Conclusions

While more work is needed, deep partitioned neural network architectures using time and frequency data seem promising in long-term solutions for denoising speech and music signals.

## 5.2 Future Work

### 5.2.1 Models

Make network deeper. Consider gradual partitioning instead of hard.

### 5.2.2 Data

Get more data. Consider different noise levels and types of signals.