12-10-2018 10:30 AM

Objective Assessment of Machine Learning Algorithms for Speech Enhancement in Hearing Aids

Krishnan Parameswaran
The University of Western Ontario

Supervisor
Parsa Vijay
The University of Western Ontario

Graduate Program in Electrical and Computer Engineering
A thesis submitted in partial fulfillment of the requirements for the degree in Master of Engineering Science

Abstract

Speech enhancement in assistive hearing devices has been an area of research for many decades. Noise reduction is particularly challenging because of the wide variety of noise sources and the non-stationarity of speech and noise. Digital signal processing (DSP) algorithms deployed in modern hearing aids for noise reduction rely on certain assumptions on the statistical properties of undesired signals. This could be disadvantageous in accurate estimation of different noise types, which subsequently leads to suboptimal noise reduction. In this research, a relatively unexplored technique based on deep learning, i.e. Recurrent Neural Network (RNN), is used to perform noise reduction and dereverberation for assisting hearing-impaired listeners. For noise reduction, the performance of the deep learning model was evaluated objectively and compared with that of open Master Hearing Aid (openMHA), a conventional signal processing based framework, and a Deep Neural Network (DNN) based model. It was found that the RNN model can suppress noise and improve speech understanding better than the conventional hearing aid noise reduction algorithm and the DNN model. The same RNN model was shown to reduce reverberation components with proper training. A real-time implementation of the deep learning model is also discussed.

Keywords

Machine learning, Deep learning, Digital signal processing, Hearing aid, Recurrent Neural Network, Noise reduction

Acknowledgments

I would like to thank my supervisor, Associate Professor, Dr. Vijay Parsa, Principal Investigator, National Centre for Audiology, for guiding me through my research work during the last two years. I would also like to thank the faculty members of National Centre for Audiology for their direct/indirect support during my research. I shall extend my gratitude to Ontario Research Fund group which offered financial support for the project.

I take this opportunity to thank my peers and friends who supported me and motivated me throughout my research period. Lastly, I express my indebtedness to my family members for encouraging me throughout my graduate program.

Table of Contents

List of Tables

List of Figures

List of Acronyms

HA	Hearing Aid
DSP	Digital Signal Processing
dB	Decibels
SNR	Signal to Noise Ratio
CMOS	Complementary Metal Oxide Semiconductor
ML	Machine Learning
RNN	Recurrent Neural Network
VLSI	Very Large Scale Integrated
ASIC	Application Specific Integration Circuit
ADC	Analog-to-Digital Converter
IIR	Infinite Impulse Response
FIR	Finite Impulse Response
FFT	Fast Fourier Transform
IFFT	Inverse Fast Fourier Transform
SPL	Sound Pressure Level
ILD	Interaural Level Difference
BTE	Behind-The-Ear
BSS	Basic Spectral Subtraction
SOS	Spectral Over Subtraction
NSS	Non-Linear Spectral Subtraction
MBSS	Multi-Band Spectral Subtraction
WF	Weiner Filter
WGN	White Gaussian Noise
PSD	Power Spectral Density
ISS	Iterative Spectral Subtraction
SSPP	Spectral Subtraction Based on Perceptual Properties
DNR	Digital Noise Reduction
DNN	Deep Neural Network
DL	Deep Learning

ANN	Artificial Neural Network
ReLU	Rectified Linear Unit
DAG	Directed Acyclic Graph
MLP	Multi-Layer Perception
MSE	Mean Squared Error
MAE	Mean Absolute Error
BP	Back Propagation
BPTT	Back Propagation Through Time
LSTM	Long Short-Term Memory
GRU	Gated Recurrent Unit
VAD	Voice Activity Detector
DCT	Discrete Cosine Transform
BFCC	Bark Frequency Cepstral Coefficients
HINT	Hearing In Noise Test
STOI	Short-Time Objective Intelligibility
OIM	Objective Intelligibility Measure
AI	Articulation Index
SII	Speech Intelligibility Index
STI	Speech Transmission Index
CSII	Coherence-Based Speech Intelligibility Index
ITFS	Ideal Time Frequency Segregation
TF	Time Frequency
DFT	Discrete Fourier Transform
HASPI	Hearing Aid Speech Perception Index
TFS	Temporal Fine Structure
IBM	Ideal Binary Mask
OHC	Outer Hair Cell
IHC	Inner Hair Cell
MFCC	Mel-Frequency Cepstral Coefficients
BM	Basilar Membrane
HASQI	Hearing Aid Speech Quality Index

CC	Cepstral Correlation
GT	Gammatone
ERB	Equivalent Rectangular Bandwidth
MHA	Master Hearing Aid
LPC	Linear Predictive Coding
NLMS	Normalised Least Mean Squared
MHANR	Master Hearing Aid Noise Reduction
CLA	Command line Application
IO	Input-Output
JACK	Jack Audio Connection Kit
TCP/IP	Transmission Control Protocol/Internet Protocol
SCNR	Single Channel Noise Reduction
RBM	Restricted Boltzmann Machine
CD	Contrastive Divergence
MMSE	Minimum Mean Squared Error
AIR	Aachen Impulse Response
OS	Operating System
IDE	Integrated Development Environment

Chapter 1

1. Introduction

This chapter deals with the background, a brief history of hearing aids (HAs), conventional signal processing algorithms in hearing aids, machine learning in hearing aids and thesis outline.

1.1 Background

Individuals who are unable to listen clearly like those with normal hearing ability i.e. hearing thresholds (sound level below which a person's ear is unable to detect any sound) of 25 dB (decibels) or better in both the ears are known to be suffering from hearing impairment [1]. In other terms, hearing loss is characterized by an increase in hearing, threshold for a sound frequency. It implies that the hearing sensitivity decreases, eventually, making it difficult for the listener to detect soft sounds. The magnitude of hearing loss can range from mild to moderate, severe, or profound, affecting either a single ear or both the ears thereby making it difficult to hear conversational speech or loud sounds. The reasons for loss of hearing or deafness may be congenital or acquired.

Statistics reveal that approximately 466 million individuals around the world have problems of disabling healing loss of which 34 million are reported to be children. Disabling hearing loss indicates *hearing loss greater than 40 decibels (dB) in the better hearing ear in adults and a hearing loss greater than 30 dB in the better hearing ear in children*. Studies reveal that by the year 2050 roughly nine hundred million individuals (one in every ten people) are likely to suffer from disabling hearing loss [1].

The exposure of the younger generation aged between 12 and 35 years, to loud noise in recreational settings have posed a risk of hearing loss to about 1.1 billion people in this cohort. The annual world-wide cost of unaddressed hearing impairment is estimated to be US$750 billion [1]. Preventing, identifying and addressing hearing impairment issues are cost-effective, bring huge benefit to people. Majority of these hearing-impaired people require custom assistive hearing devices.

The intelligibility of human speech is quite significant in hearing and communication. It is not only a measure of comfort but also that of comprehension. Various factors such as the physical

characteristics, conditions in communication and capacity of information as well as the ability to gather information from a specific context, mimics and gesture determine the quality and intelligibility of speech [2]. Intelligibility can be better understood by discussing the distinction between real and recorded speech. In a real conversation, individuals will be able to identify/distinguish the sounds in the background and focus on the speech of the person conversing, thus filtering the desired information from the audio environment. This helps to significantly increase the speech intelligibility and comprehension although the communication may happen in a noisy environment [2].

The environment in which speech signals are transmitted are typically noisy in real-time situations. Persons with normal hearing may understand speech in a moderately noisy environment since speech is an extremely redundant signal [3]. Therefore, even if part of the signal is masked by noise, the unmasked part may still allow the speech segments to be intelligible for effective communication. Comparatively, there is reduced redundancy in speech signal for individuals suffering from hearing impairment because the either the speech is partially audible or is severely distorted. Background noise that masks even a small portion of the remaining speech signal will degrade intelligibility significantly. Thus, people with hearing loss have a greater difficulty in understanding speech in a noisy environment than the people with normal hearing [3].

The noise potential for masking speech is expressed by the Signal-Noise Ratio (SNR) which is the ratio of the power of speech signal to the power of noise, when both are presented concurrently. Lower the value, bigger is the loss in understanding of speech for both normal hearing and hearing impaired [4]. Consequently, as the SNR increases, the speech intelligibility improves significantly.

1.2 A brief history of hearing aids

Hearing aids are small electronic devices that amplifies sound and makes it easier to understand speech. They are designed in a way that they can capture sound waves using a very small microphone, modify softer sounds into audible sounds, and finally pass the same to the ear by means of a tiny speaker, enabling individuals with hearing disability to perceive sounds again, and enhance their ability to hear. Thus, it can acquire, process and feedback acoustic signal in real time in noisy conditions [2]. In the modern day, with the help of microchips, hearing aids are not only smaller in size but also are able to demonstrate a better quality.

The invention of hearing aids is centred on no single person. Though there are no recorded dates, it is believed that ear horns have existed in every civilization for thousands of years since humans first created carved objects. Giovanni Battista Porta was most likely the first to describe ear trumpet, a metal version of the ear horn, around 1588 [5]. The 17th century saw the growth of speaking tubes as assistive hearing devices until pre-electric horns and trumpets became popular.

In the 1800s the introduction of decorative and functional hearing aids which could be concealed and made less obvious, addressed the concerns regarding public perception about hearing aids [5]. Finally, in the nineteenth century the idea of electrically-powered hearing aid instruments was introduced which consisted a battery box, earpiece and microphone, made of carbon dust which was to be refined later.

The first electronic hearing aids were devised after the invention of the telephone (the technology within which increased how acoustic signal could be altered) and microphone in the 1870s and 1880s. In 1898 Miller Reese Hutchison created the first electric hearing aid and named it Akouphone [6]. The use of carbon transmitter assisted in amplifying the sound by taking a weak signal and making the same stronger with the help of electric current.

The invention of microprocessor led to the miniaturization of hearing aids. In 1987, the first commercial digital hearing aids was invented. Until then, the hearing aids were analog in nature. In analog hearing aids, incoming noise are dealt with by taking electrical sound waves emerging from the microphone, amplifying the same ‘as is’ and transmitting it to the ear while digital hearing aids transforms soundwaves from a microphone into digital binary code which is in turn altered by a microchip within the hearing aids prior to transforming back into analog signals and delivering to the speaker [7].

The functionalities of both analog and digital hearing aids are basically similar i.e taking sounds to amplify them so that they can assist with better hear. Both analog and digital hearing aids are programmable, whereby the microchips within can be customised to improve the sound quality and matched with the individual user. At the same time, they can be customised to develop several settings for use in various environments. For example, the settings can be adjusted in a

quiet environment and can be readjusted while in a noisy restaurant and further readjusted in large auditoriums.

Compared to analog hearing aids, digital hearing aids contain additional features and flexibility which are perceived to be user-configurable as it has the ability to modify sound in digital form. For example, digital hearing devices offer numerous channels and memories, providing advantage of being able to save more location-specific profiles. Additionally, digital hearing aids possess the capability to decrease surrounding noise automatically, eliminate feedback or whistling, or the capability to favour human voices over any other noise.

1.3 Signal processing in hearing aids

Digital Signal Processing (DSP) is a relatively recent technique that involves the sampling of an analog signal and the processing of these samples in digital form. This processing can be accomplished using standard digital integrated circuit technologies such as complimentary metal oxide semiconductor (CMOS), which is space and energy efficient. The digital processing of speech is a mature technology and a wide range of algorithms is available for filtering, compression, noise reduction, dereverberation, feedback reduction and special effects as well as coding for bandwidth reduction. Many of these techniques would prove beneficial to the hearing impaired if tailored correctly to their needs and implemented in a hearing aid [8].

Signal processing research on digital hearing aids encompasses signal acquisition, amplification, transmission, measurement, filtering, parameter estimation, separation, detection, enhancement, modeling, and classification [9]. They can be broadly categorised into four areas. The first area adopts advanced signal processing methods to characterize and compensate for hearing loss like loudness and frequency selectivity loss. The second area focuses on effective target signal improvement and reducing noise, including adaptive microphone array technologies, spectral subtraction algorithms, and blind source separation methods. The third area consists of practical usage of hearing aids to address problems like flexibility, convenience, feedback cancellation, and artifact reduction. Finally, the fourth area concentrates on developing hearing aid technology into devices, the functions of which can also be used for hand phones and music players [9]. Here the focus is mainly placed on problems like echo cancellation, bone conductive microphones, and wireless voice link. However, faced with constraints pertaining to hardware

necessities, computational speed, consumption of power and other pragmatic issues, the advancement and application of signal processing methods in digital hearing aids have faced numerous challenges in the last decade [9].

1.4 Introduction to machine learning

Machine learning is a technique of analysing data, which helps to automate building of analytical models. It is that branch of artificial intelligence, which adopts the idea that data can be learnt effectively from the systems, patterns can be identified, and decisions can be made with little human involvement. This field of computer science uses statistical methods and provides computer systems a learning ability to improve performance, gradually on specific tasks, either with or without explicitly programming the data.

Machine learning has applications in various areas such as image recognition (face detection and recognition in Facebook), speech recognition (as in Siri in Apple phones and Alexa for home automation), medical diagnosis (prediction of disease progression), product recommendation (as in Amazon, Netflix or Youtube websites), basket analysis (learning association between products people buy), classification and prediction of potential customer behaviours in terms of loan repayment in banks, real-time information extraction from web pages, data security (predicting which files are malware), fraud detection (in PayPal), traffic and route recommendations (as in Google Maps) etc.

While implementing conventional signal processing algorithms in digital hearing aids, the speech characteristics need to be learnt, analysed and then hard coded to segregate speech in audio segments. Similarly, the types of noise and various environments that the hearing-impaired listeners are subjected to, must be known during design to be able to program the noise characteristics into the hearing aids. Recently, machine-learning approaches to enhancement of speech have been promising in refining speech intelligibility for people with hearing impairment. A machine learning model understands the underlying details and learns to differentiate speech from noise when trained with features of speech and noise separately. The recent researches also indicate that deep learning can be used for automatic speech recognition where latency is not important [18]. This project focuses on speech enhancement using a combination of signal processing and deep learning techniques in real-time, where latency is of paramount significance.

1.5 Problem statement and contributions

Conventional noise reduction techniques in modern hearing aids work well with stationary noise. They may also provide improvement in SNR. However, the improvement in SNR does not necessarily enhance speech intelligibility as per behavioural studies. The conventional methodologies have limitations processing non-stationary noise. Recently, deep learning-based models have been found to perform better than conventional algorithms for hearing aid applications. This thesis focuses on training and implementation of a deep neural network for the purposes of noise reduction and dereverberation, evaluation of its performance using objective predictors of speech intelligibility and quality and real-time implementation of the algorithm in a portable computing platform.

1.6 Thesis outline

This thesis is organized in chapters that discuss conventional signal processing algorithms in a bit more detail (chapter 2), Machine Learning (ML) and its speech applications so far (chapter 3), the implementation of speech enhancement algorithm, database used to evaluate it, the objective metric and comparison between Gammatone and Bark filters (chapter 4), results from the objective assessment of pretrained and trained Bark RNN and their comparison with a couple of reference algorithms (chapter 5), followed by conclusion and future work (chapter 6).

Chapter 2

2 Conventional signal processing algorithms

Signal processing in hearing aids has come a long way over the past few decades. This chapter briefly discusses the advancement in digital signal processing (DSP) over the past few decades, outlines the various DSP algorithms incorporated into hearing aids, and finally focuses on the conventional noise reduction DSP algorithms and their effectiveness in enhancing speech perception by hearing impaired listeners.

2.1 DSP over the years

The world is becoming more and more digital in our daily activities since the invention of integrated circuits. Not only did the size of microelectronics shrink, but the computational power of the microelectronic systems doubled almost every two years. This follows the "Moore's law of microelectronics" which led to a remarkable influence on the technology within hearing aids. An all-digital hearing aid was reported to be developed in the 1980s [10]. Researchers created several prototype hearing aids with the help of customised DSP chips using less power as well as very large scale integrated (VLSI) chip technology. These hearing aids were put to use for research on individuals with hearing-impairment. Now, most of the commercially available hearing aids are digital. Modern hearing aids have become intelligent systems providing a wide variety of algorithms for specific and diverse hearing and communication challenges of users in various acoustic environments [10].

Numerous hearing aids, in the modern times, comprise application-specific integrated circuits (ASICs) as well as reprogrammable DSP cores and microcontroller cores providing advantages of flexible reuse of the microelectronics for different signal processing algorithms. There has been significant improvement in the performance of digital hearing aids during the last two decades [10]. The objectives of hearing aids have been upgraded to improve quality of sound by reducing the artifacts and providing more natural sound output, in addition to improving speech intelligibility. These requirements must be met in dynamically changing listening environments in everyday life situations.

Newer DSP platforms provide computational abilities to implement advance functions like reducing frequency, cancelling noise impulses and adaptive directional microphones. Another area

where there has been a huge improvement is in systems integration [10]. Adaptive algorithms could sometimes counteract each other, just like algorithms for noise-cancellation and acoustic feedback controlling algorithms are likely to decrease gain and work in opposition to amplification schemes. These effects are taken into consideration during design, implementation as well as in choice of parameters for signal processing algorithms. A more detailed description of individual DSP algorithms within hearing aids is given next.

2.2 Signal processing in hearing aids

The types of hearing loss are so diverse that each of them needs custom prescriptions. One listener may need frequency-dependent gain, the other may need level-dependent and the third listener may need a combination of both. Thus, technology in hearing aids must be flexible enough to be able to meet revised requirements and digital hearing aid is the solution available today. Figure 2.1 provides a framework of signal processing algorithms in digital hearing aids of modern times. The signals from the environment that come to listener's ear are picked up by one or more omnidirectional microphones. These signals would be analog in nature. Hearing aids in the early stages used to process analog signals only which means amplification of input in simple terms. This may not help the hearing-impaired listeners much and in fact amplification could just cause distress.

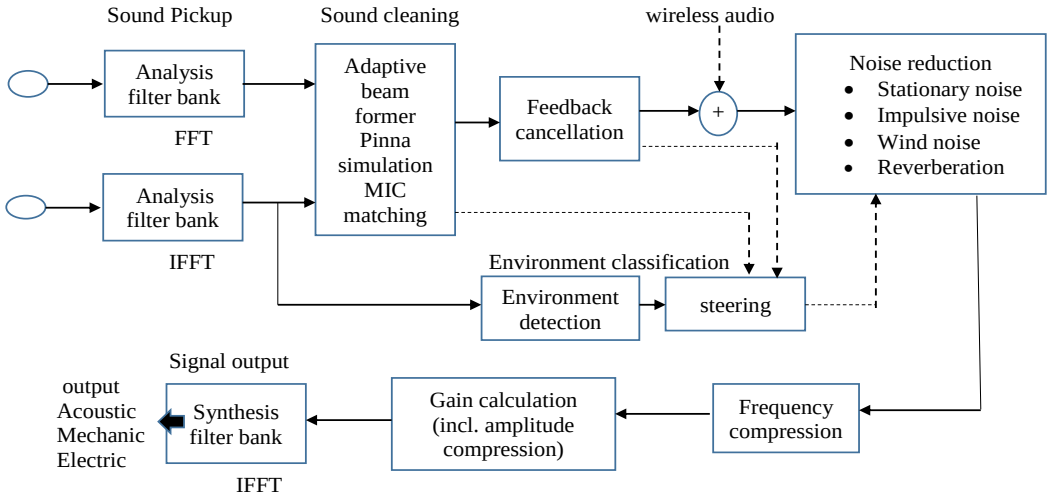

Figure 2.1: Signal processing in hearing aids

The analog signal (sound waves) is converted to electrical signal by the vibration of diaphragm in microphones which is caused by the changes in air pressure. This electrical signal is then sampled and digitized using an Analog-to-Digital Converter (ADC), which usually provides a usable audio bandwidth of 8 to 16kHz. The resultant discrete time signal may be analyzed spectrally either in frequency domain or in time domain depending on the signal processing algorithm used in the device [10]. Either of the methods will involve trade-offs in time-frequency resolution. The hearing devices these days are expected to have a delay no more than 10-12ms to avoid incongruity between the speaker's lip movement, echo and the audio signal [10]. The spectral resolution is dependent on frequencies with bandwidth proportional to center frequency.

The time domain analysis of speech signals can be done using a bank of infinite impulse response (IIR) filters as: Signal from microphone → Filter bank (Combination of low pass, band pass and high pass filters) → Sound cleaning → Loudness adjustment → Recombination of signals into single output. Each of the filters in filter bank may allow or stop a specific range of frequencies that form a band. The channel width increases with center frequency with more frequency resolution at lower frequencies. IIR filters require much lesser computation to achieve desired filter slopes when compared to FIR filters. However, a cautious design is required to evade the artifacts that may be produced otherwise. It is recommended to process a block of samples instead of processing each sample when it comes for computational efficiency [10].

The analysis in frequency domain is done by taking Fourier transform of the signal: Store specific number of samples in a buffer → Multiply the values with a window function (to avoid discontinuities at the edges) → Apply Fourier transform (FFT) → Sound cleaning → Loudness adjustment → Inverse FFT → Output synthesis [10].

Hybrid analysis systems combine the time domain online processing with frequency domain offline processing. The signal collected by microphone is processed first in time domain up to sound cleaning, which is summed across channels before passing it on to the filter bank in frequency domain. Such systems join lower quantization sound as well as the time domain filter distortion with the higher frequency resolution of FFT systems. At the same time there is little need to smoothen the gain across frequency or have large temporal overlap between consecutive blocks. Time-frequency analysis schemes form the basis for different adaptive algorithms though the scope is limited in hearing aids when compared to human auditory system [10].

The signal processing algorithms are broadly classified into three types based on their functionality:

1) Frequency and level-dependent gain application to provide a satisfactory volume for the respective hearing profile (usually audiogram)
2) Sound cleaning by removal of stationary and non-stationary noises and reducing acoustic feedback through the use of various algorithms for noise reduction.
3) Environment classification for automatic adjustment of hearing aid settings to choose suitable signal processing algorithm for a given condition

No single algorithm may provide optimal performance in all these aspects. Though different manufacturers may have or come up with entirely different signal processing strategies and solutions the underlying class of these strategies remain same with premises and scope clearly defined for each of these solutions.

2.3 Audibility

There are a couple of important strategies to restore audibility. Most people suffering from sensorineural hearing impairment endure loudness recruitment, a phenomenon which causes the loudness to grow rapidly for any sound above the absolute threshold. The gain should reduce when input levels increase to compensate this effect, which means the input-output function is compressive and the process by which we achieve it is called multichannel compression. For lower inputs, up to about 40 dB SPL, the gain is kept constant. The applied gain decreases with further increase up to 100 dB SPL which compensates loudness recruitment and the compression ratio is made infinite beyond 100 dB SPL for limiting [10]. The compression algorithms can be slow-acting (like in Adaptive Dynamic Range Optimization) based on input sound level which will help maintain signals for localization of sound based on Interaural Level Differences (ILD) or fast-acting which will have comparatively shorter attacks and release times and are able to make loudness perception level closer to normal.

Many a times it is quite tough to restore audibility at high frequencies for individuals suffering from extreme hearing loss. Applying gains at these frequencies could cause acoustic feedback or loudness discomfort, especially when they have narrow dynamic range. Frequency lowering is an alternative approach in such situations. This would be perceptually beneficial because of the lower threshold in lower frequencies. Frequency lowering methods can be classified

into three categories: a) frequency transposition – a block of higher frequency component is shifted downward to a frequency in destination band, b) frequency compression or frequency division – frequency components up to a frequency stay unaffected and the components above are shifted downward with a lower slope and c) spectral envelope warping - lowering of part of the spectral envelope, keeping the spectral components unchanged.

2.4 Sound cleaning

It is important to note that restoring audibility alone will not suffice for speech perception in acoustically challenging environments. It is indeed challenging for hearing-impaired people to understand speech (intelligibility) in noisy environments. Though reducing noise may sound an easier task, this has been a prime area of focus since the invention of digital hearing aids. There are so very divergent types of speech and background noise to deal with. What makes it more difficult is the constantly changing speech and noise in realistic sound scenarios. The target speech differs due to the characteristics of speaker, vocal effort, distance and orientation of the speaker, the amount of reverberation in the listening environment and the presence/absence of visual cues. Due to this variability, a single algorithm for optimal noise reduction is not available.

The most successful approach for noise reduction uses directional microphone, that can help when there is a spatial separation of target speech from source of the noise. Hearing aids using this technique will have two omnidirectional microphones – one at the front and the other toward the back side. The signal collected from one microphone is delayed and added with the other to form a beam pattern. In this manner, the target signal from the beam's direction will be captured and others will be attenuated. The shape of beam pattern is constant in a static beamformer whereas it dynamically adapts based on environment in an adaptive beamformer. When the direction of highly interfering sound changes, the adaptive beamforming system shall adjust itself to maintain the good suppression of the interference. The beamformer may be independently applied to various frequency bands instead of the entire broadband signal.

Directional microphones provide 2-5 dB improvement in the threshold of speech reception in realistic conditions [10]. Though use of these microphones is advantageous in many ways, these may not help in understanding of speech coming from side or back like a vehicle coming from behind the listener or sounds coming from side and back at movies. The placement of these

microphones such as in the Behind-The-Ear (BTE) hearing aids, far from the pinna may also affect their directivity.

Binaural beamformer is a technique combining the four microphones of two bilaterally worn hearing aids that forms a four-microphone directional system. The outputs of two microphones on both sides are initially combined before exchanging over a wireless link with the hearing aid on the other side. Thus, the ipsilateral and contralateral signals in each hearing aid are combined using a frequency dependent weighting function to create binaural directivity pattern, which is narrower than the monoaural one. This static beamformer can be made adaptive by combining the resulting signal in one hearing aid with the current spatiotemporal distribution of undesired signals. Researches earlier have demonstrated the way this approach can enhance speech intelligibility while reducing the effort made to hear.

The above two approaches assume that source of speech is directly at the front side of the listener. However, this is not the case in many of the communication scenarios, just like the target could be on the side while driving. A solution for such a situation is to pick up signal from the ear that is ipsilateral (better ear) to the target and transmit the same to the contralateral side. This will allow both the ears to receive a reasonably clean representation of the target.

In situations where speech and noise are spatially co-located, or in small-size custom hearing aids which preclude the placement of two microphones due to size restrictions, additional strategy is required for noise reduction. Single microphone noise canceling algorithms are such strategies and represent one of the most widely used techniques in digital hearing aids today. It works on the premise that speech signal has different temporal properties than that of the interfering sounds and that low-rate temporal fluctuations are relatively scarce in case of the background sounds. These temporal fluctuations in speech signal are used to estimate the SNR, which is used in computationally efficient algorithms like spectral subtraction, harmonic extraction, and Wiener filtering techniques to reduce interference. These algorithms are briefly described next.

One of the initial noise reduction algorithms that was first proposed for single channel speech improvement was spectral subtraction (BSS) [11]. It is a technique to restore power or the magnitude spectrum observed signals in additive noise, by subtracting the estimated average noise spectrum from a noisy signal spectrum. An estimation of the noise spectrum is done and updated,

from the periods when there is no signal while the noise alone exists. In this manner, the estimation of noise spectrum is made when there are pauses in speech and is deducted from the spectrum when the speech is noisy. While doing so, an assumption is made that the noise is not correlated with speech and that the noise is additive in nature. Many variants of spectral subtraction are present, and the principle for all the variants is to estimate short time speech spectral magnitude by deducting sound from noisy speech or by multiplying the noisy spectrum with gain functions and combine with the phase of noisy speech.

Spectral over-subtraction (SOS) introduced two extra parameters such as over-subtraction factor and noise spectral floor in basic spectral subtraction method to reduce remnant noise (processing distortion), which is one of the two drawbacks of BSS, the other being narrow band musical noise [11]. The over-subtraction factor restricts the quantum of noise power spectrum subtracted from the noisy speech power spectrum in each frame. The spectral floor parameter helps to dissuade the ensuing spectrum from dipping beyond a pre-determined minimum level instead of setting to zero (spectral floor). This implementation considers that the speech spectrum is affected by noise evenly and the subtraction factor subtracts an over-estimate of noise from noisy spectrum. In order to draw a balance between the surrounding noise and remnant noise elimination, numerous combinations of over-subtraction factor α, and spectral floor parameter β provides a trade-off between the quantity of residual noise in the background and the extent of perceived remnant noise. Because noise is perceived to influence speech spectrum equally and that the over-subtraction feature is constant across frequencies, the enhanced speech may be distorted.

In reality, noise is colored and influences speech signal in a different way throughout the spectrum and subtraction factor is expected to be frequency dependent to explain various types of noise. The non-linear spectral subtraction (NSS) idea, extends this ability by forcing the dependency of the over-subtraction factor frequency while the subtraction process remains non-linear. Bigger values are deducted at frequencies with low SNR levels, and lesser values are deducted at frequencies with high SNR levels giving better flexibility for compensation of errors in the estimation of noise energy in various frequency bins [11].

Multi-band spectral subtraction (MBSS) groups frequencies into bands and multiple bands constitute the speech spectrum. It implements four uniformly spaced frequency bands and spectral subtraction in each band is independent of the other [11]. Since the real-world sound is extremely

random in nature, enhancement in MBSS algorithm is required for the reduction of WGN. The MBSS algorithm usually performs better than other subtractive-type algorithms.

In a Wiener filter (WF) design it is assumed that speech and noise are not correlated [11]. They are considered to be in normal distribution and this filter aims to reduce the mean squared error criterion. The gain function of WF is fixed at all frequencies and it requires power spectral density (PSD) of clean signal and noise for filtering, which are the main drawbacks of WF. It is not possible to apply non-causal WF directly in order to estimate clean speech since it is not possible to assume that speech is stationary. An adaptive WF implementation decreases each component of frequency by a certain amount that depends on the power of noise at that frequency.

The gain function in WF is calculated using power spectrum of noisy speech, rather than clean speech which will degrade its accuracy. An iterative algorithm is used to solve this problem. In the iterative spectral subtraction (ISS) algorithm, the improved speech output is taken to be the input for following iteration and consequently for spectral subtraction, the remnant noise is re-estimated in every iteration. Therefore, a larger number of iterations will provide a better enhanced speech when compared to WF.

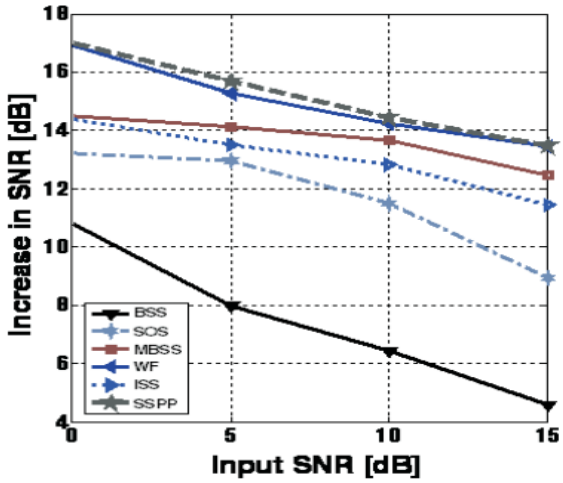

Figure 2.2: A comparison of performance of different spectral subtraction methods [11]

The use of fixed subtraction parameter is one of the major drawbacks of spectral subtraction while the noise spectrum in real time is changing constantly. Although MBSS will allow the adaptation of parameters, the remnant noise is not completely suppressed especially at low SNRs. The speech signal quality and intelligibility can be improved by the perceptual

properties of spectral subtraction [11]. By incorporating the masking properties of human auditory system into enhancement process, it is possible to attenuate components of noise that were previously inaudible as a result of masking. Based on the masking properties, the subtraction parameters are adapted in the algorithm. Through the calculation of noise masking threshold, masking properties are modelled.

The spectral subtraction effectiveness is highly depending on estimation of accurate noise that is mostly a tough task. Since the power spectrum and magnitude are non-negative variables, the negative estimates of these variables, if any, must be mapped into a non-negative value. The distribution of the restored signal is distorted by this nonlinear rectification method and the distortion is more visible when the SNR reduces. Spectral subtraction that is based on perceptual properties (SSPP) performs better in comparison to above discussed algorithms for speech improvement. WF leads to reduced remnant noise, however, the structure of noise is musical and speech regions, especially fricative consonants, are reduced. Such spectral subtraction can lead to distortion of speech. Thus, using SSPP algorithm for enhanced speech is more pleasant reducing the remnant noise with least speech distortion, if any. A comparison of various spectral subtraction methods is demonstrated in figure 2.2, which indicates the advantage of SSPP algorithms [11].

It is important to note that difference exist among manufacturer-specific implementations of the aforementioned spectral subtraction and Wiener filtering algorithms. Estimation of speech and noise spectra and rules for engaging/disengaging noise are proprietary. As such, assessment of noise reduction feature, either objectively or through subjective measurements, is of great importance. Few studies have investigated the effectiveness of noise reduction algorithms as implemented in commercial hearing with hearing impaired listeners. For example, Ricketts et al. studied the effect of digital noise reduction (DNR) in hearing aids on speech recognition and quality [12]. This subjective study with 14 adults wearing hearing aids presented that there is no impact on speech understanding with the use of conventional noise reduction techniques although there may be an improvement in speech quality. Another study by Bentler et al. showed that DNR can improve listening comfort but has no effect on speech perception of even with visual cues [13]. Desjardins et al. found that speech recognition in noise score did not change much with noise reduction in place for 12 hearing-impaired listeners with BTE hearing aids [14]. Brons et al.

observed different perceptual effects among hearing aids and that none of the noise reduction algorithms actually improved speech intelligibility while they may reduce annoyance of noise [15].

Single microphone noise canceling is usually effective for noise spectrum that hardly changes over time and does not take into account the undesirable effects from an enclosed environment [10]. In environments such as office, lecture rooms and big halls, the target speech will have multiple paths from the source to listener. Apart from shortest direct path from source to listener, the target signal could hit any part of the room (floor, ceilings or walls) and objects in the room and reflect from them before it adds on to the direct path to listener. This process is called reverberation and can cause severe impact on the ability to perceive speech. The performance of beamformers depreciates when distance between target and listener increases and with increasing reverberation. The reverberated speech can be represented as the convolution of original speech and the impulse response of the environment. Dereverberation would need an algorithm to estimate and segregate the parts of speech that are reflected off the objects in the reverberating environment from the speech collected by the hearing aid.

Thus, dereverberation is a highly complex task in real-time due to its need to estimate the constantly changing original speech target and the unknown impulse response of the room or hall. The dereverberation approaches are generally very complicated to implement them in hearing aids leading to time delays, which may not be acceptable for hearing aids. Lebart et al. [16] took a simpler approach to this problem by estimating parts that are decaying over time and attenuating them. This signal processing was done by converting the time-domain signal into frequency domain and comparing the decay rate with respect to standard reverberation time. This method definitely has shown improvement in quality of sound and hearing ease but does not enhance speech intelligibility [17].

Signal processing algorithms provide a wide range of solutions to improve speech intelligibility in adverse listening environments. However, there are limitations for every algorithm based on environment in which it is assessed. The objective and subjective results may vary most of the time because of the change in premises in realistic environments. Hence it is important to specify the testing conditions and every result would be slightly or largely different from the other based on these conditions.

To summarize, signal processing in hearing aids, the conventional signal processing techniques to restore audibility and sound cleaning are discussed. Though there are many noise reduction techniques available, the increase in SNR provided by these methods does not necessarily improve speech intelligibility. The same is observed with dereverberation though there are evidences of improvement in speech quality. Emerging evidence suggests that algorithms incorporating machine learning are effective in enhancing speech understanding by hearing-impaired listeners in noisy environments. Machine learning techniques are discussed in detail in the next chapter.

Chapter 3

3. Machine learning in hearing aids

Speech enhancement in noisy conditions has always been a challenging task for better speech recognition and further processing, not just in hearing aids but also in all telecommunication systems. The objective of speech enhancement is to reduce noise and to increase intelligibility of the input noisy speech. However, the results using conventional signal processing methodologies are not always satisfactory in terms of speech intelligibility and speech quality, in adverse conditions. Hence there is always a need for better performing adaptive noise reduction algorithms because of the non-stationarity of speech and noise signals.

Machine learning is that branch of artificial intelligence which has been recently found useful in various speech recognition and natural language processing tasks. Specifically, deep learning has the capability to learn complex patterns underlying speech data. Healy et al. used deep neural network (DNN) to segregate speech from noise for hearing-impaired listeners [18]. This work focused on ideal binary mask estimation with a classification approach using a Restricted Boltzmann Machine (RBM). Chen et al. extended the work with ideal ratio mask estimation instead using a deep neural network [19]. This chapter explains a bit about machine learning, deep learning, artificial neural networks and their learning process.

3.1 Types of machine learning

ML algorithms are generally classified into three, based on the way the models learn.

1) Supervised learning - This algorithm includes a target or outcome variable (also known as dependent variable) which will be predicted from a set of predictors (independent variables). The algorithm attempts to model the relationship and dependencies between the dependent and independent variable with the help of the given set of variables, in a way that the output values for a new data set based on those relationship can be predicted This training process goes on till the model is able to achieve the desired accuracy level on the training data. Example: Regression and classification tasks
2) Unsupervised learning - In this algorithm, no target or outcome variable is available to make predictions. It is used for detecting patterns and descriptive modeling. Clustering of

population in different groups generally uses unsupervised technique for segmenting customers in different groups for specific intervention.

3) Semi-supervised learning – This method falls in between the above two. The cost of getting labeled dataset is usually high and, in such situations, we obtain some labeled data along with lots of unlabeled data for training.
4) Reinforcement learning - Here, the machine is exposed to an environment where it trains itself repeatedly through trial and error and has to determine the ideal behaviour within the given context to maximize performance. Q-learning, temporal difference and deep adversarial networks are some of them.

ML can be further categorized into three based on the objective the model is trying to achieve.

1) Regression: For prediction of a continuous real number
2) Classification: For prediction of a discrete number
3) Clustering: For segmentation

Deep Learning (DL) uses different architectures of Artificial Neural Networks (ANNs) to learn from data. These ANNs are a family of machine learning models that have taken inspiration from human brain and the biological nervous system.

3.2 Artificial neural networks (ANNs)

An ANN is composed of neurons, which are the basic functional units inspired from neurons in biological neural network. These neurons are interconnected to form a network of neurons that takes in and processes a set of inputs to provide a set of outputs. The neurons (also called as nodes) are organised in layers containing specific number of them in every layer and connected with specific ones in another layer. Each of these connections (called edges) has a weight that is multiplied with the conveyed value. Figure 3.1 illustrates a neuron for which θ_1, $\theta_2,\ldots,\theta_N$ are the weights for incoming values with θ_0 and 1 being the weight and input representing the bias to the neuron. The sum of weighted inputs to the neuron is passed to an activation function that computes the output or activation of the neuron.

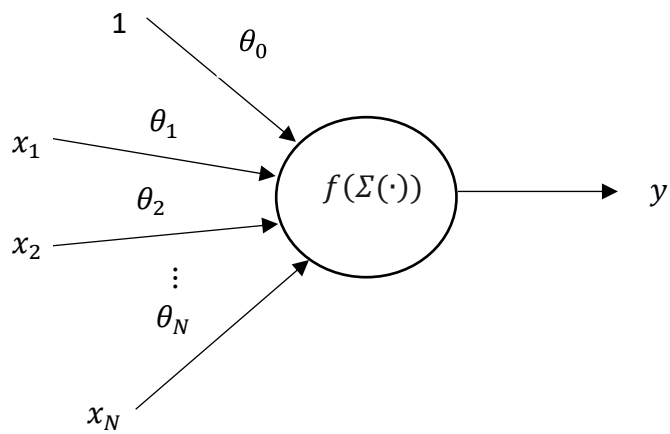

Figure 3.1: Artificial neuron

3.2.1 Activation function

An activation function is a function that transforms the input value to an output to determine whether the neuron must be activated or not.

$$y = f(\sum_{i=1}^{N} \theta_i x_i + \theta_0) \tag{3.1}$$

$$y = f(\theta^T x) \tag{3.2}$$

where, θ is the weight vector, x is the input vector and w_0 is the bias

If $\mu = \theta^T x$, a sigmoid activation function would have the following representation:

$$f(\mu) = \frac{1}{1+e^{-\mu}} \tag{3.3}$$

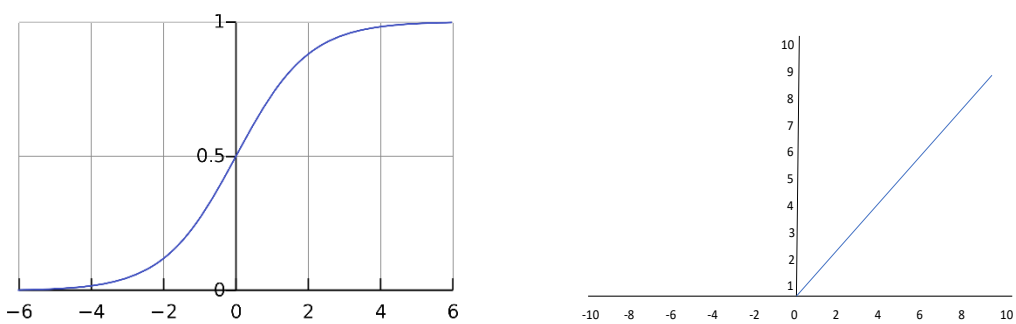

Figure 3.2 (from left to right) a) sigmoid activation [20] b) ReLU activation function

A Rectified Linear Unit activation (ReLU) function and a leaky ReLU are given by

$$ReLU: f(\mu) = \max(\mu, 0) \tag{3.4}$$

$$Leaky\ ReLU: f(\mu) = \max(\mu, 0) + \alpha \min(\mu, 0) \tag{3.5}$$

The sigmoid activation function is linear when the weights are small and the corresponding weighted sum is close to zero. However, it will turn non-linear as the weighted sum grows in both the positive and negative directions as in figure 3.2a. This would result in neuron be considered on or off as the weighted sum tends to infinity. The magnitude of bias value acts as the threshold weighted sum that the inputs must provide to activate the neuron and contribute to output. Recent years have seen the development of rectified non-linear activation functions like Rectified Linear Unit (ReLU, figure 3.2b), which increases linearly for positive inputs and gives zero output for any negative input. The drawback with ReLU is that it is not differentiable for negative inputs, which provides no further learning. Leaky ReLU is a variant of ReLU that has a non-zero gradient for any negative input, which is more popular for image classification tasks than conventional ones.

Rectified non-linearity is observed to provide significant boost to the performance of object recognition system as per [21]. [22] found that types of ReLU can provide a better model of biological neurons than conventional hyperbolic tangent or sigmoid functions. [23] recommends replacement of sigmoid with ReLU in feedforward neural networks.

3.2.2 Feedforward neural network

Feedforward neural networks are a subclass of ANNs in which the nodes are organised in a directed acyclic graph (DAG) from input to output. A unique type of these networks is the multilayer perceptron (MLP) as in figure 3.3. Here the nodes are arranged in layers where connections are provided only between neighbouring layers. The features are fed in to the first layer which is the input layer. The output is taken from the final layer of the network termed as the output layer. The layers between input and output layers are termed as the hidden layers. The number of layers is identified by counting the number of layers in the network and neglecting the input layer since it does not perform any computations. By this convention, the network in figure 3.3 is a 2-layer MLP.

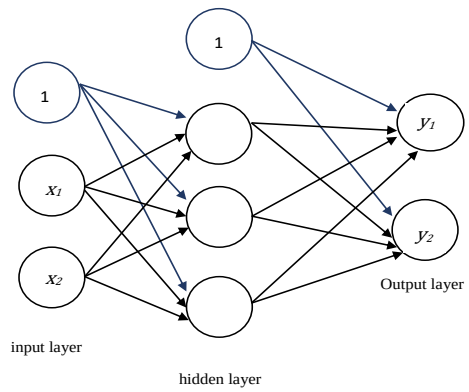

Figure 3.3: Multilayer Perceptron (MLP)

3.3 Learning process in neural networks

Multilayer perceptrons are usually trained in supervised manner using labeled data. Problem identification is the first step in any machine learning task, followed by collecting data. The training data is a collection of input-output pairs [{x^k, y^k}, k = 1 to m] where x^k is feature set of kth training example, y^k is the corresponding label and m is the total number of examples.

3.3.1 Forward propagation

The inputs are multiplied with respective weights progressively layer after layer. The output from final layer is the prediction by the network. A cost/loss function is defined to measure the difference between actual (target) output and the output predicted by the network. The training process calculates the loss value after every pass of specific number of examples specified by batch size from input to output (forward propagation) and updates the parameters so as to minimize the loss value [24]. For a 3-layer network (2 hidden layers and an output layer),

$$\left.\begin{array}{ll} a^{(1)} = x^{(i)} & \text{(inputs)} \\ z^{(2)} = \theta^{(1)} a^{(1)} & \\ a^{(2)} = g(z^{(2)}) & \text{(add } a_0^{(2)}\text{)} \\ z^{(3)} = \theta^{(2)} a^{(2)} & \\ a^{(3)} = g(z^{(3)}) & \text{(add } a_0^{(3)}\text{)} \\ z^{(4)} = \theta^{(3)} a^{(3)} & \\ a^{(4)} = h_\theta(x) = g(z^{(4)}) & \text{(prediction)} \end{array}\right\} \quad \boxed{\begin{array}{ll} \text{Forward propagation} & (3.6) \\ \text{(Vectorized implementation)} & \end{array}}$$

The (MSE) cost/loss function is defined as:

$$J(\theta) = \sum_{i=1}^{m}[h_\theta(x^i) - y^{(i)}]^2 \tag{3.7}$$

A few widely used loss functions are mean squared error (MSE), sum of squared error (SSE) mean absolute error (MAE) and cross entropy error. The objective of the network is to find the set of weights or parameters that minimize the loss function:

$\min_{\theta}$ J(θ) ⟹ Need code to compute $J(\theta)$, $\frac{\partial}{\partial\theta_{ij}^{l}} J(\theta)$

The most critical part of training is the update of parameters which is done using backpropagation in MLPs. The neural network is initialized with small weights and once the loss value is calculated, the derivates of loss with respect to the weights of the networks are calculated and the resulting value times learning rate (a factor that determines the rate of update of parameters) is subtracted from the previous weights.

$$\theta := \theta - \propto \frac{\partial}{\partial\theta} J(\theta) \tag{3.8}$$

θ is the weight vector,
$\propto$ is the learning rate and
$\frac{\partial}{\partial\theta}$ J(θ) is the gradient vector

3.3.2 Backpropagation (BP)

BP uses a gradient based update rule and hence the cost and activations must be differentiable. The differentiation of the objective cost function with respect to output will result in an error term. Derivatives are calculated with respect to all output elements and these terms are propagated backwards in the network using chain rule.

$\delta_j^{(l)}$ = error of node j in layer l
$\delta_j^{(4)} = a_j^{(4)} - y^4$ or $\delta^4 = a^{(4)} - y$

$\delta^3 = (\theta^{(3)})^T \delta^4 .* g'(z^{(3)})$ $\quad a^{(3)} .* (1 - a^{(3)})$
$\delta^2 = (\theta^{(2)})^T \delta^3 .* g'(z^{(2)})$
No (δ^1)

Back Propagation (3.9)

In general, the partial derivatives can be given by

$$\frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta) = a_j^{(l)} \delta_j^{(l+1)} \text{ (ignoring regularization part)} \tag{3.10}$$

$$\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)} \tag{3.11}$$

$$D_{ij}^{(l)} := \frac{1}{m} \Delta_{ij}^{(l)} + \lambda \theta_{ij}^{l} \quad if\ j \neq 0 \tag{3.12a}$$

$$D_{ij}^{(l)} := \frac{1}{m} \Delta_{ij}^{(l)} \quad if\ j = 0 \tag{3.12b}$$

$$\frac{\partial}{\partial \theta_{ij}^{l}} J(\theta) = D_{ij}^{(l)} \tag{3.13}$$

The training set size is usually large (>>1) and a summation over the gradient term is necessary to get the derivatives of objective function. The update of weights is sometimes done using the entire training data, called as full batch learning. It is recommended to update weights more frequently using a subset of the entire training data called mini-batch or after every training example is passed through the network which can be considered as batch size = 1.

3.4 Regularization

The key to successful training of a neural network is to have large enough training examples proportional to the dimensionality of features. This will enable the network to learn various combinations of features and generalize well for unseen data. However, if the neural network has too large number of parameters compared to the number of training examples or if the training data is not representative enough, then there is a risk of overfitting the network parameters with the limited number of training examples. This would mean that the model will memorize the feature mapping from training data and perform less better when it is exposed to unseen data. This results in high accuracy or performance with training data and low accuracy or performance with testing data. Reducing the complexity of the model will obviously make overfitting less likely but this can limit the ability of the network to learn complex pattern and cause underfitting.

The solution to overfitting is to have a complex enough model with large number of training examples. Though this looks like a straight forward solution, the size of data that can be obtained is domain dependent and getting a huge labeled dataset may be costly or unrealistic.

Fortunately, there are many alternatives in place to avoid overfitting in complex models. These are collectively called as regularization. One of the regularization methods aims to avoid the weights of the network from growing too big. This is done by incorporating an additional term in loss function which penalizes the loss when the values of weights increase, which will essentially make the learning smoother and less susceptible to outlier occurrences.

An example of such a regularization function is L2 that adds a parameter times the sum of square of weights to the cost function. If the regularization parameter has a big value, the only way to reduce loss and optimize the network is by reducing the sum of square of weights small, which means the weight values must be small. L2 or weight decay ensures that the weights with small or zero gradient terms are slowly scaled towards zero, reducing the effect of unnecessary parameters and simplifying the model. Thus, regularization is a method to improve performance on unseen test data by probably making the performance on training data conservative.

Another example of regularization is L1 which penalizes the sum of absolute values of weights rather sum of squared values. This method is chosen when the result is desired to be sparse and many weights will have an optimal value of zero.

3.5 Recurrent neural network

Artificial neural networks typically process a set of features and predict the dependent variable without considering time. They assume that subsequent inputs are independent of each other. However, there are applications like speech recognition, natural language processing and time series forecasting where the values of input at different time instants may be important for prediction of future output. A recurrent neural network is a type of artificial neural network which uses its internal state or memory to process sequences of input and this makes it applicable to such time dependent tasks.

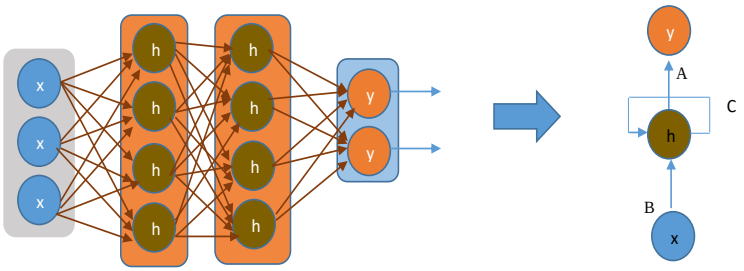

Figure 3.4: Recurrent Neural Network

Recurrent Neural Networks (RNN) were developed in 1980s. A recurrent neuron has connection to itself apart from the connections to those in previous and next layers as in ANNs. This will form a directed graph along a sequence. The idea here is to use sequential information. If the task is to estimate the subsequent word in the sentence, it is important to know which words came prior to it. RNNs can store the information captured from the previous inputs in its memory.

In figure 3.4, x is the input, h is the hidden state and y the output in an ANN. The figure b shows its recurrent representation – A is the set of weights from hidden layer to output layer, B the weights from input to hidden layer and C being the weights for recurrent connections.

3.5.1 Back propagation through time

Backpropagation through time (BPTT) is a gradient-based method used to train RNNs. BPTT applies backpropagation training algorithm to recurrent neural network to sequence data like that of time series. A recurrent neural network represents one input each timestep and estimates one output by unrolling every input timestep. Every timestep has single input, a copy of the network, and a single output. In figure 3.5, x and y are the inputs and outputs for different time instants 0, 1, …, t-1, t, t+1, … to/from RNN with parameters A, B and C and hidden state h. Errors are calculated and accrued for every timestep. The network is rolled back up and the weights are revised. Spatially, every timestep of the unrolled RNN may be viewed as an added layer, given the order dependence of the problem and the internal state from the previous timestep is accepted as input for the succeeding timestep.

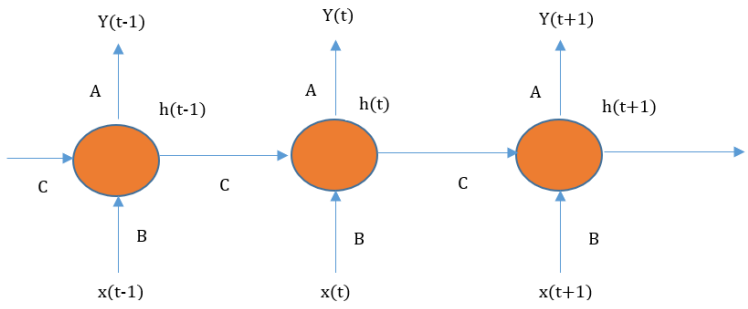

Figure 3.5: RNN unrolled in time

The algorithm can be summarized as: Present a sequence of timesteps of input and output pairs to the network →Unroll the network then calculate and accumulate errors across each timestep → Roll-up the network and update weights → Repeat.

3.5.2 Vanishing gradient problem

BPTT can get costlier with the increase in the number of timesteps. If input sequences constitute thousands of timesteps, then the number of derivatives required for one weight update will also count into thousands. As the length of time interval between latest output and its oldest dependent input increases, the number of error values that need to be calculated increases and these error values have to be propagated all the way to the first layers. If the gradient values are smaller and close to zero, any multiplication between the error values and weight matrix will result in a set of even lower values for the layers in the beginning during backpropagation. This can cause weights to vanish or explode (go to zero or overflow) in the early layers and make slow learning and model skill noisy.

There are a few solutions to overcome the vanishing gradient problem:

1) Global search methods that do not use gradient information investigated by Bengio et al. [25] such as simulated annealing, multi-grid random search, discrete error propagation and random weight guessing (though it is unclear how it can handle complex problems). Angeline et al. proposed a genetic approach that also avoids gradient computation.

2) Methods that force higher gradients by time-weighted pseudo-Newton optimization, but they have restrictions in learning to store accurate real-value information over time [26].

3) Ring proposed a method to add a higher order unit influencing appropriate connection every time a unit in the network receives contradictory error signals [27]. However, Ring's net was not able to generalize well for unobserved lag durations.

4) Mozer used time constants that influence unit activations, however these time constants must be fine tuned externally for extended time lags [28]. Sun et al proposed to update the activation of a recurrent unit by summation of old activation and scaled current net input [30]. The net input, however, might disturb the stored information. Lin et al. proposed alternatives of time-delay networks like NARX networks in which the reduction of error is slacked due to the shortcuts taken for back propagation [31]. Nevertheless, the suggested alternatives are unable to solve the general problem since they can increase only by a constant multiplicative factor, the duration of temporal dependencies that can be learned.

5) The hierarchical chunker systems by Schmidhuber could possibly bridge random time lags although the success depends only on the local predictability through the sequences leading to time lags [32]. The performance of hierarchical recurrent networks with self-organizing time scales for grammar learning tasks has been found to deteriorate when noise level intensifies, and the sequences of input becomes less compressible.

3.5.3 Long short-term memory

RNNs come with a great advantage of being able to connect previous information to the present task. This is especially useful when we need only the previous word to predict the next one. For example, if we are attempting to predict the next word in "the clouds are in the …"(sky), we will not require additional information to predict the same. RNNs can learn to predict such cases where the gap between required information and the place where it is needed is small. However, there are instances where more information is required. For example, in order to predict the last word in "I grew up in Quebec, Canada……. I speak fluent … "(French), more information would be needed since the word to be predicted in the last line is a language. The additional information is obtained from previous timesteps that is then being narrowed down to the exact language. There are many such situations where the gap between relevant information and the place it is required can be large. Though RNNs were considered to be capable of handling such situations, [25] and [29] found the results otherwise. The problem of vanishing gradient makes the

conventional RNNs hard to train. Long Short-Term Memory (LSTM) networks can address this problem [33].

LSTMs are novel, efficient, gradient-based models that are developed by [33] to mitigate the vanishing gradient problem. They observed that BPTT in RNNs may blow up or vanish the error signals that flow backwards in time. This may cause oscillation of weights or take an undesirably long time to learn the underlying pattern or does not work at all. LSTMs can learn to bridge gaps of the order of 1000 time steps even during noisy sequences. This is achieved by enforcing constant error flow through internal states of special units.

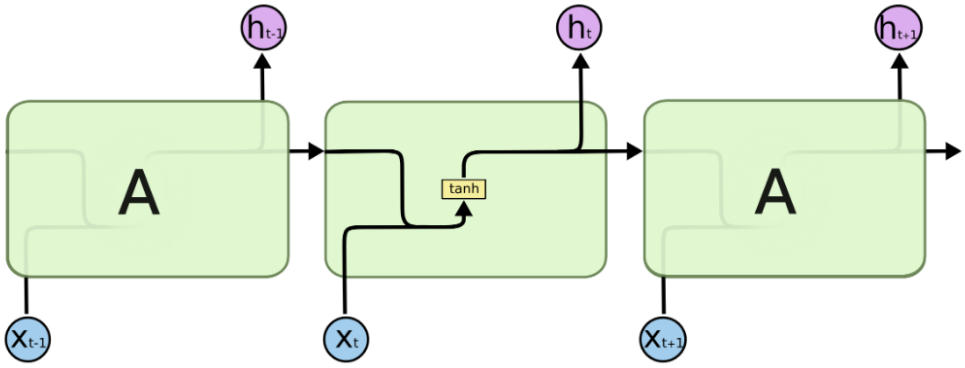

Figure 3.6: Mathematical model of RNNs [34]

Recurrent neural networks are composed of recurrent neurons, the mathematical model of the internal components of which is given in figure 3.6. This repeating module will have a simple structure in RNNs, such as a tanh layer. LSTMs also have such structure depicting chain, although the repeating modules have a different structure. There are four neural network layers interacting in a unique manner, refer figure 3.7 and 3.8.

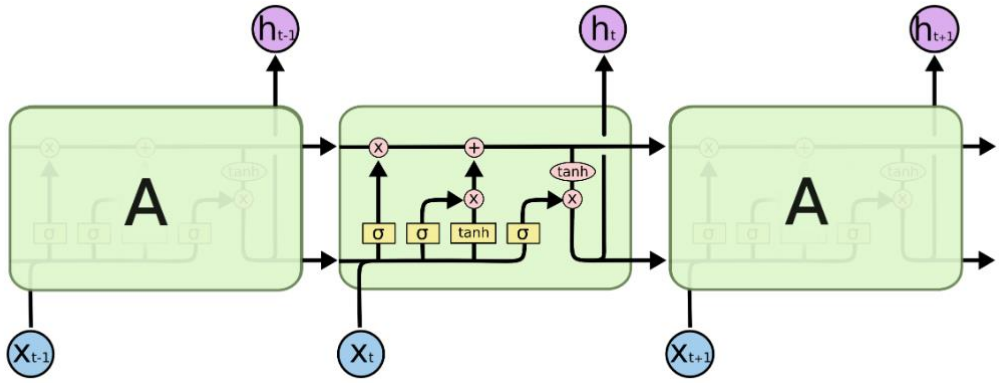

Figure 3.7: Mathematical model of LSTMs [34]

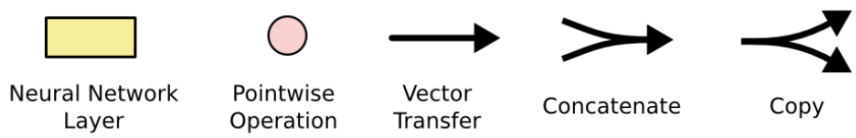

Figure 3.8: Operations

The important part of LSTMs is the cell state. It cannot be manipulated but can be regulated to allow information to pass through. This is done using a sigmoid layer. There are three ways this regulation happens. The first sigmoid layer acts as forget gate which decides the information that can be let through from previous cell state:

$$f_t = \sigma\ (\theta_f * [h_{t-1},\ x_t] + b_f) \tag{3.14}$$

The second one known as input gate which decides the values to be updated. This is done in combination with a vector of new candidate values.

$$i_t = \sigma\ (\theta_t * [h_{t-1},\ x_t] + b_i) \tag{3.15}$$

$$\check{C}_t = tanh\ (\theta_{\check{C}} * [h_{t-1},\ x_t] + b_{\check{C}}) \tag{3.16}$$

Actual update of cell state requires multiplication of old state by forget gate value and adding the result to the product of new candidate values and input gate values.

$$C_t = f_f * C_{t-1} + i_t * \check{C}_t \tag{3.17}$$

The third sigmoid layer decides parts of the cell state that will go to output.

$$\sigma_t = \sigma\ (\theta_\sigma * [h_{t-1},\ x_t] + b_\sigma) \tag{3.18}$$

$$h_t = \sigma_t\ * tanh\ \ (C_t) \tag{3.19}$$

The updated cell state is put through a tanh function to ensure values be between -1 and 1and multiply by the output of the sigmoid gate.

3.5.4 Gated recurrent unit

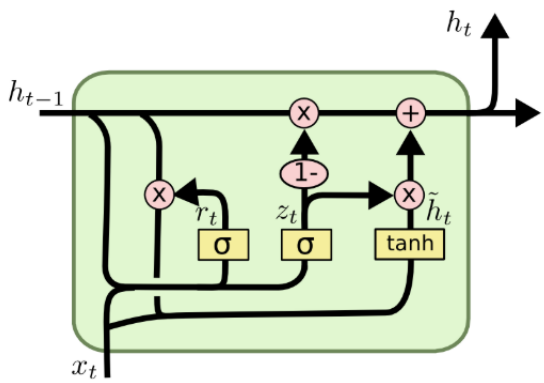

Figure 3.9: Gated Recurrent Unit [34]

Researches involving LSTM use different versions of this architecture. [35] used the one with peephole connections in all the sigmoid layers. Another variant uses coupled connection of forget and input gates. A more interesting version is the gated recurrent unit (GRU) proposed by [36] (figure 3.9). In this variant, the forget and input gates are combined to form an update gate.

$$z_t = \sigma\ (\theta_z * [h_{t-1},\ x_t]) \tag{3.20}$$

$$r_t = \sigma\ (\theta_r * [h_{t-1},\ x_t]) \tag{3.21}$$

It combines the cell state and hidden state as well.

$$\hat{h}_t = tanh\ (\theta\ * [h_{t-1},\ x_t]) \tag{3.22}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t\ * \hat{h}_t \tag{3.23}$$

GRUs are much simpler than LSTMs and hence are becoming more popular these days.

To summarize, machine learning and its types, applications of deep learning in speech enhancement, what are recurrent neural networks and their drawback and how these drawbacks can be addressed in modeling time sequences are discussed. In the next chapter, a specific deep learning model is discussed in detail for speech enhancement application in hearing aids.

Chapter 4

4. Implementation

This chapter explains the implementation details of the machine learning model including feature extraction, neural network architecture, training, the database used for training, evaluation metrics and speech enhancement with another set of features using Gammatone filterbank.

A typical noise reduction technique will include a noise spectral estimator, driven by voice activity detector (VAD) or a similar algorithm. There are mainly three estimators (figure 4.1) that need to be accurate but are difficult to tune [37]. The VAD will check if there is speech content in the given segment and if yes, it gives update to noise spectral estimator to estimate the noise spectrum. The noise spectral estimator has to estimate the spectral contents of noise and convey the information to spectral subtractor to adapt accordingly. If there is no speech content, the noise spectral estimator has no work to be done and spectral subtractor remains deactivated.

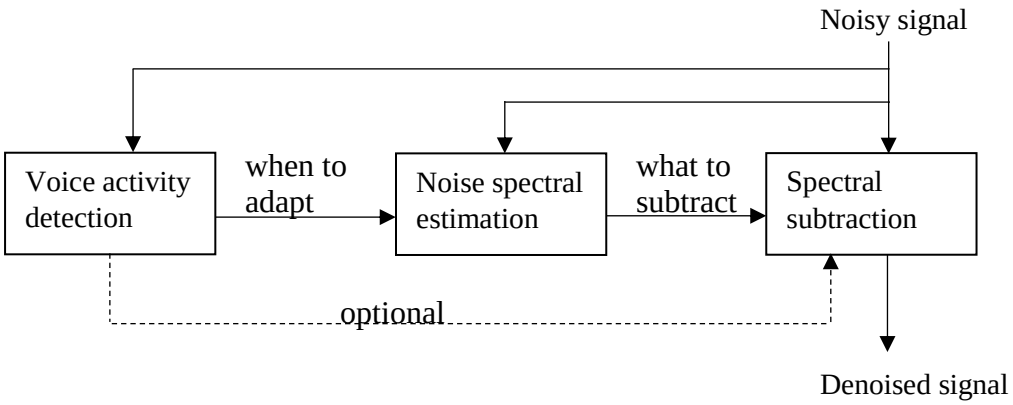

Figure 4.1: Noise reduction scheme

Though improvements have been made in conventional noise and spectral estimators, it remains difficult to design them and requires a lot of time to manually tune it. This is why advanced technologies like deep learning have found application in speech enhancement. Deep learning is found useful in applications like automatic speech recognition where latency and computational factors are of less importance [37]. The selected approach focuses on evaluation of deep learning in real-time applications with low complexity. The conventional end-to-end systems use signal processing for noise reduction. This results in increased complexity of the solution and requires a lot of manual efforts.

Instead, a hybrid approach proposed by JM Valin is chosen which uses deep learning in place of most of the conventional signal processing steps and estimators [37]. Some basic signal processing blocks were used where they are considered to be more straightforward. In this method, full-band (48kHz) speech is used for further processing. Majority of the noise reduction is done using gains computed by a GRU RNN on a low-resolution envelope. A pitch comb filter is used to suppress the noise that remains between pitch harmonics.

The entire algorithm may be seen as having the following three components:

4.1 Feature extraction

The input noisy signal would be in time domain which needs to be converted to frequency domain for any processing. This is done on a frame level and the frame size is kept at 480. A window function is applied before transforming to frequency domain for analysis/synthesis based on time domain aliasing cancellation:

$$w(n) = sin\,[\frac{\pi}{2} sin^2\,(\frac{\pi n}{N})] \tag{4.1}$$

where N is window length.

This is in accordance with Princen-Bradley criterion and achieves efficient reconstruction of the signal from frequency domain after processing. Fourier transform is applied to convert the windowed signal into spectral domain, based on opus codec. The FFT size is kept at twice the frame size i.e, 960. The main processing is based on 20 ms with 50% overlap (10ms offset). Figure 4.2 provides a summary of processes involved in construction of cleaner speech from a noisy one using the RNN model.

There are tens of thousands of frequency bins in the audible range of 20Hz to 2kHz. Research shows that estimating all of them would require thousands of neurons in the neural network with millions of weights and around 400 outputs. This would jeopardize the very objective of reducing complexity of the model. This can be addressed by assuming that the envelopes of speech and noise have less resolution than frequency bins. Here, Bark scale is used in grouping frequencies and form a scale of 22 bands in the audible range.

The bandwidth of frequency bands grows as we move from low to high frequency regions. The energy in each band, E(b), in Bark scale is calculated using the amplitudes of signal in frequency domain and band size:

$$E(b) = \sum_b w_b\ (k)\ |X(k)|^2 \tag{4.2}$$

where, w_b (k) is amplitude of band b at frequency k

$$\sum_b w_b(k) = 1 \tag{4.3}$$

X (k) is magnitude of signal at frequency k.

The bands are constrained with the condition of at least four bins at low frequencies as in Opus codec. The frame analysis is followed by pitch down sampling and pitch search. A pitch shifted version of the input signal is created based on the pitch values and values in buffer. The window function is applied to this signal as well before transforming into Fourier domain and then energy

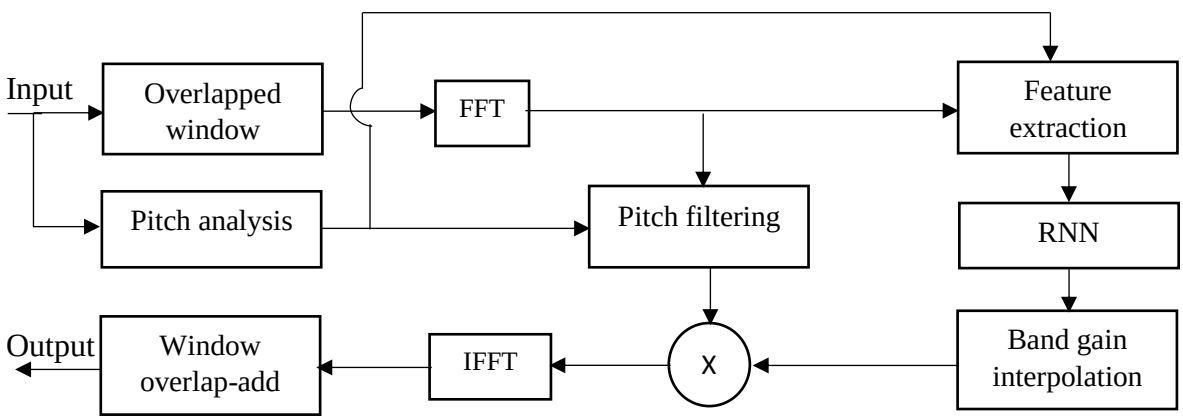

Figure 4.2: Speech enhancement using RNN model

is computed. The product of real parts of original and pitch shifted signals for a given bin in each band and the product of corresponding imaginary parts are added. The resultant is multiplied by the fraction of band size and the process is repeated with all segments of band to get the band correlation. This band correlation is scaled by the energies in both the original signal and pitch shifted signal.

$$p_b = \frac{\Sigma_k w_b(k)\ \ R\ [X\ (k) P^*(k)]}{\sqrt{\Sigma_k w_b(k) |X(k)|^2 .\ \Sigma_k w_b(k)\ |X(k)|^2}} \tag{4.4}$$

where X(k) and P(k) are the original and its pitch shifted version in frequency domain.

Discrete Cosine Transform (DCT) is applied to the log spectrum which will give 22 Bark Frequency Cepstral Coefficients (BFCC). The other features include first and second temporal derivatives of first six BFCCs, six coefficients of the result of DCT of pitch correlation across frequency bands, pitch period and spectral non-stationarity metric. The 42 features extracted from one of the frames in training set are plotted in figure 4.3 for understanding.

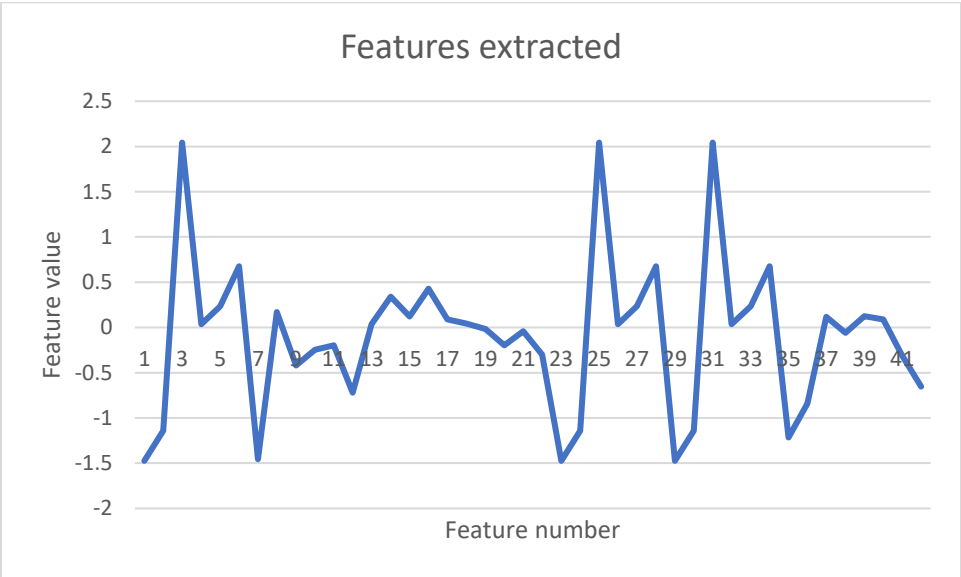

Figure 4.3: Features from an audio frame

The target values are the gains in each band and VAD output. The gain in each band is calculated as the ratio of energy in clean speech to that in noisy speech:

$$g_b = \sqrt{\frac{E_s\,(b)}{E_x\,(b)}} \tag{4.5}$$

where, E_s (b) is the energy in clean speech and E_x (b) is the energy in noisy speech.

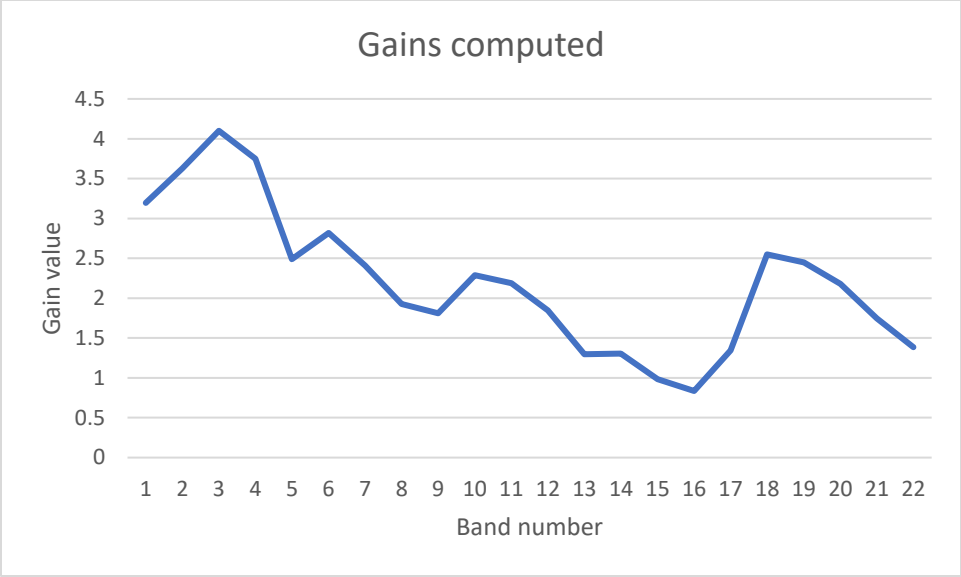

Figure 4.4: Gains computed for an audio frame during Bark frequency analysis

Figure 4.4 shows the gain values computed for the 22 bands during feature extraction process that will be applied to input to reconstruct a cleaner version of the noisy frame.

4.2 Neural network architecture

The neural network designed for training consists of feedforward, fully connected (dense) layers and the recurrent layers (figure 4.5). The input layer has 42 neurons to accept the Bark features and 22 neurons at the output layer to provide an estimate of the gai values in each band. The architecture may be looked at as two sequential models – one for voice activity detection and the other for gain estimation. It is a 6-layer network with 4 hidden layers and 2 output layers.

The VAD section of the neural net consists of 3 layers besides the input layer. The 42 features are transformed to another set of features using a dense layer, which are fed to the GRU layer. The GRU layer uses ReLU activation function to capture non-linearity. During feature extraction, the speech content in a frame is checked based on the energy of clean speech in that frame. If the frame energies indicate speech content over consecutive frames, VAD activation is estimated with a fully connected dense layer.

The gain estimation section can be viewed as a combination of noise estimation and subtraction. It has 3 layers – 2 GRU recurrent layers and 1 output dense layer. The first GRU layer takes in the features and the outputs of dense and GRU layers in VAD section as input. This layer

acts as a noise spectrum estimator. The next GRU layer considers input features and the outputs of previous two GRU layers as essential features. This layer together with the fully connected dense layer at the output learns to estimate gain values based on ideal critical band gains calculated in the feature extraction process.

The GRU layers together are the backbone of this architecture. This deep learning network has a total of 215 units with 49 units in VAD section and 166 units in the other. Though there were trials done with different number of units and layers, the outputs suggested that there is no significant improvement in performance with further increasing complexity and that this architecture is the optimal one. The way we construct the training data has a larger impact on final quality.

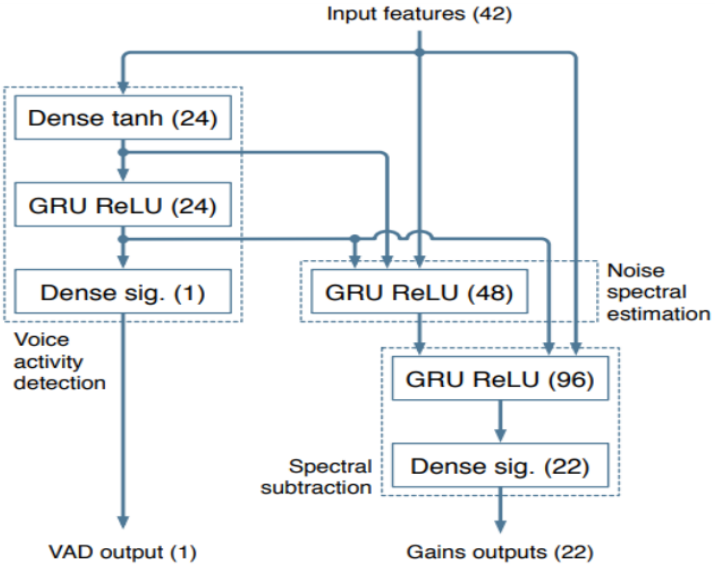

Figure 4.5: RNN model [37]

4.3 Training

There are in total 42 features that are extracted using the feature extraction procedure. These features are extracted from both clean speech and noise-only signals. During this feature extraction, the clean and noise signals are mixed at different SNRs randomly. The number of features required for training can be changed based on the duration of clean and nose-only contents available. The 22 gain values are also extracted and written into a file. These features/gains together written in a binary file are then exported to a h5 file for efficient use of main memory.

The neural network is defined using Sequential, Dense and GRU modules in Keras library. The h5 file will have 65 columns – 42 features, 22 gain values and 1 vad ouput. The number of rows depends on the number of speech/noise segments used for feature extraction. This data has to be converted to 3 dimensions to create inputs for recurrent layers at different time steps. Hence a window size is defined which decides the duration of noisy sequence considered at once. The ratio of the number of training examples in h5 file to window size will give us the number of sequences for training.

4.3.1 Loss function

The first 42 columns in the file are considered for input and the last 23 columns (gains and VAD) are taken as true values for training. A form of the difference between predicted and true values can be used to update the weights as the network learns. Typically, mean squared error is the loss function used in wide range of applications. This metric performs very well in any given offline application or where noise is relatively stationary. However, a modified form is recommended for speech perception application.

To be able to match the perceptual effect, a perceptual parameter γ is proposed by JM Valin that can control the degree of noise suppression. Having calculated the true value of gain during feature extraction, the loss function for a given gain estimate is given by:

$$L(g_b, \tilde{g}_b) = (g_b^{\gamma} - \hat{g}_b^{\gamma})^2 \quad (4.6)$$

The loss function minimizes the MSE on log-energy, which would make the suppression way too aggressive and may cause speech distortion. The value of γ is recommended to be ½ to have a good trade-off between noise suppression and speech distortion. A smoothing function is applied to the gain values estimated by the model to provide minimum expected level of reverberation. Standard cross entropy loss function is used for VAD output.

4.3.2 Optimization

Training on a full dataset via batch gradient descent is time consuming and the weight update will be less frequent. On the other hand, stochastic gradient descent may make progress more frequently, but the speedup obtained by vectorization is lost to the time required for processing of each example separately. It does not guarantee to converge too. Here, a version of

the mini-batch gradient descent is used during training. This algorithm will pull a mini-batch specified by batch size from the entire dataset randomly for training update. This process is repeated until the number of samples equal to size of the dataset is trained per epoch. This will preserve the vectorization benefits and makes good progress towards minimum.

Gradient descent may cause oscillations (which can be imagined as oscillations perpendicular to the plane in which contour plot of the cost function is drawn) that slows down the learning process [38]. In addition, gradient descent will overshoot if learning rate is chosen high. Being conservative in learning rate selection will make the learning significantly slower. To curb the vertical oscillations and make progress in the horizontal direction, exponentially weighted average concepts called momentum and RMSProp were introduced.

$$v_{dw} = \beta_1 v_{dw} + (1 - \beta_1)\, dW \tag{4.7a}$$

$$v_{db} = \beta_1 v_{db} + (1 - \beta_1)\, db \tag{4.7b}$$

$$s_{dw} = \beta_2 s_{dw} + (1 - \beta_2)\, dW^2 \tag{4.8a}$$

$$s_{db} = \beta_2 s_{db} + (1 - \beta_2)\, db \tag{4.8b}$$

$$W = W - \propto \frac{V_{dw}^{corrected}}{\sqrt{s_{dw}^{corrected}} + \varepsilon} \tag{4.9a}$$

$$b = b - \propto \frac{V_{db}^{corrected}}{\sqrt{s_{db}^{corrected}} + \varepsilon} \tag{4.9b}$$

where, W and b represent the horizontal and vertical movement components, $\propto$ is the learning rate, β_1 and β_2 are the weighted average components of momentum and RMSProp respectively,

Adam, an optimization algorithm that combines the advantages of momentum and RMSProp, is used during training.

4.4 Database

The objective of this project is to evaluate/predict the performance of machine learning algorithms with hearing impaired individuals under a variety of noisy conditions. Hence the dataset chosen is as close to realistic situations as possible. There have been different trials with various

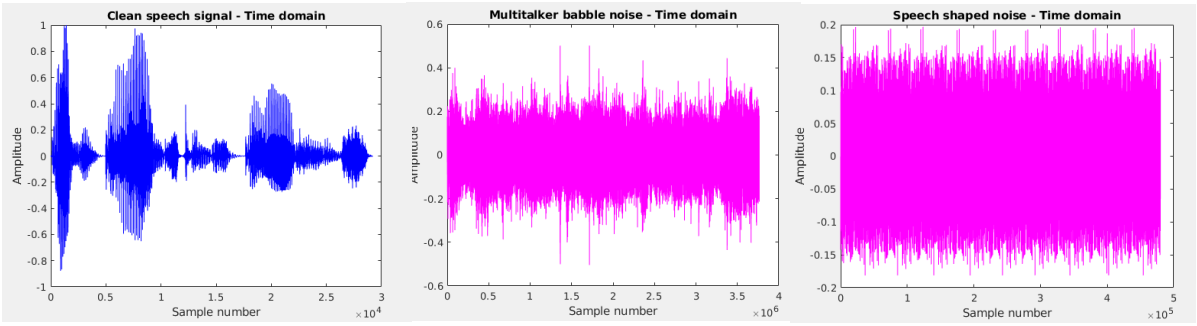

Figure 4.6: (from left to right) Time domain representation of a) clean speech b) babble noise c) speech shaped noise

combinations of speech (figure 4.6a) and noise (4.6b, 4.6c) for training and testing. In general, the clean speech and noise only segments are obtained from Hearing in Noise Test (HINT) database and Edinburgh datashare [39].

There are 250 clean utterances in the HINT database overall. They are arranged in 25 lists with 10 sentences in each list (one of them given in figure 4.6a and 4.7a). Edinburgh datashare is a digital repository of data produced with the infrastructure available at University of Edinburgh. The last 42 of the clean speech files (out of 292) are carefully chosen from this datashare so that they are understandable for a normal hearing individual before evaluating with hearing loss settings. 47 different noise-only segments have been used for training. 42 of them come from the same Edinburgh datashare by subtracting the clean speech part from the noisy ones. Besides, noises like cafeteria, traffic, white, multitalker babble (figure 4.6b, 4.7b) and speech shaped noise (figure 4.6c, 4.7c) are used in combination. Testing is done with speech from HINT database and babble, speech shaped noise and traffic and the results are shown for speech with babble noise.

The majority of the speech and noise files were originally sampled at a sampling frequency of 44.1 kHz. They were resampled to 48 kHz as the RNN model is developed to work with that

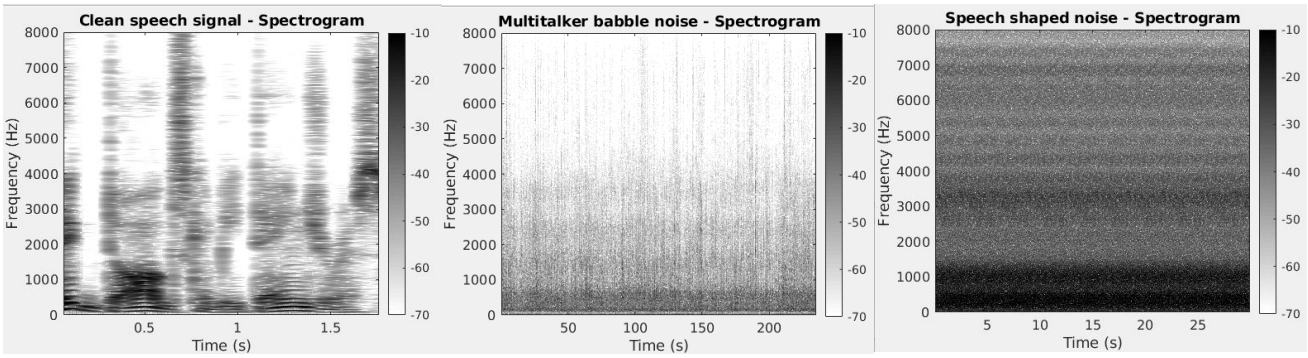

Figure 4.7: (from left to right) Spectrogram of a) clean speech b) babble noise c) speech shaped noise

frequency. All the clean speech files were then added together to form a single file. The same process is followed for noise-only files too. The feature extraction is coded in C language. The clean speech and noise-only portions have to be in separate files for the model to learn their features during training and detach them during testing. These files are in 16-bit pcm formats for efficient use of memory and are provided as parameters for the program executable.

The features are extracted in a binary format which are then transferred to an h5 file again for better use of memory. The features for further processing are extracted from this file during training. The neural network to learn the gain values in each band of Bark scale based on frame representation is defined in Python. Keras provides suitable APIs to establish complex models and is a highly popular library that is in use today.

4.5 Pretrained RNN model

As a preliminary study, the pretrained model was used to evaluate how well the model, as it is, performs with new types of noise. The feature extraction and gain computation using RNN developed from scratch are combined in a single code for real-time purposes. The weights of saved model from training are used in the RNN developed in C for testing. Testing is done with clean speech mixed with babble noise at -3dB, 0dB and 3dB respectively. The input noisy signal is close to 2 seconds long, amplitude is non-zero for many frequencies though it is very low for frequencies above 2000Hz.

4.6 Evaluation metrics

Speech enhancement algorithms in assistive hearing devices aim to improve either the speech intelligibility, quality or both. In a noise reduction system, this is done by estimating the characteristics of noise and remove the segments of speech where similar characteristics are observed. In this process, the signal degradation due to noise may be improved but it can often create modifications to speech in the form of artifacts. These artifacts can further affect intelligibility.

Subjective listening test is usually the methodology used to confirm the effectiveness of any speech enhancement algorithm. However, it can often be time consuming and costly, especially in algorithm development stage. The other systematic way of evaluating the performance of algorithm is to have an objective measure that estimates the speech intelligibility or quality.

4.6.1 Short-time objective intelligibility (STOI) measure

The objective intelligibility measures (OIMs) like articulation index (AI), which is one of the first ones, speech intelligibility index (SII) and speech transmission index (STI) are found to be suitable for various types of degradations like additive noise, reverberation, filtering and clipping. Nevertheless, they are less appropriate for speech enhancement systems where noisy speech is processed by a time-frequency weighting. For instance, when spectral subtraction is applied, the STI-based measures estimate an improvement in intelligibility though the subjective tests suggest otherwise. Similarly, coherence SII (CSII) and covariance-based STI show low correlation with the actual intelligibility of ideal time frequency segregation (ITFS) processed speech [40].

Most of the OIMs estimate intelligibility based on long-term statistics of entire speech signal. It is important to have an intermediate measure that takes the short time statistics in ITFS processed speech into consideration. Consequently, Taal et al. proposed STOI [40], which is based on short-time (~400ms) TF regions. STOI assumes that the clean and processed signal are time- aligned. The model resamples both the signals to 10 kHz frequency before processing. The signals are segmented, Hanning-windowed with 256 samples zero padded up to 512 samples and then a short-time DFT is applied.

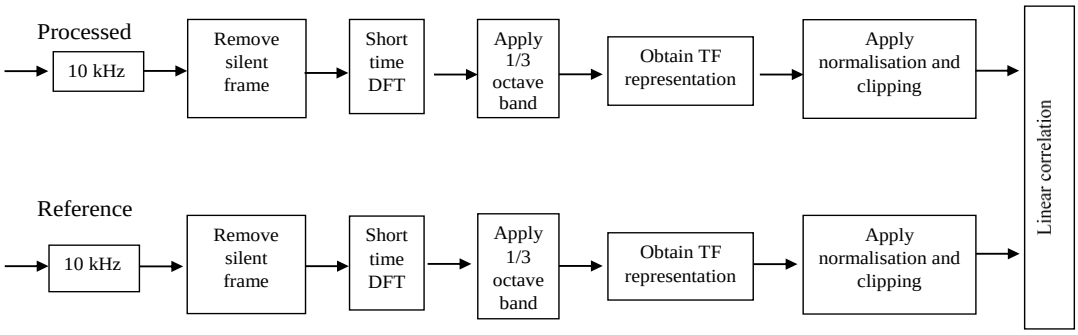

Figure 4.8: STOI model

STOI works with 15 1/3 octave bands by grouping DFT bins. The TF-representation of both the clean and processed speech are essentially the square root of sum of the square of the DFT bin values in a band. A normalization factor is applied to every TF-region of processed signal in order to equalize the energy of processed speech with that of clean speech. This signal is applied with a bound for signal-distortion ratio. The intermediate intelligibility measure is the linear correlation between the clean and modified processed TF-units (figure 4.8). And finally, STOI is calculated by taking the mean of the intermediate measure over all bands and frames.

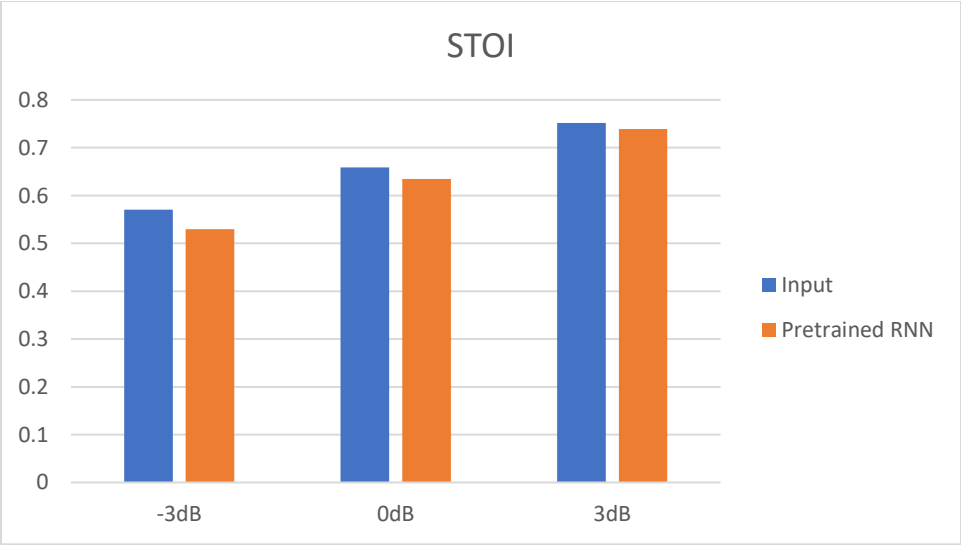

Figure 4.9: Comparison of STOI values between noisy speech and the output of pretrained RNN model at SNRs -3 dB, 0 dB and 3 dB

4.6.2 Hearing aid speech perception index (HASPI)

Additive noise degrades speech by filling the short silent segments in speech with the random fluctuation of noise. This can also corrupt the temporal fine structure (TFS) of the speech. The speech processing in hearing aids involve suppression of this noise and dynamic range compression that would adapt amplification/attenuation over time. This time-varying gain can cause distortion of signal envelope and introduce modulation sidebands as well.

Changes in both TFS and signal envelope are being used to predict speech intelligibility. TFS changes are measured using indices like coherence-based SII. Coherence is the ratio of cross correlation of clean unprocessed and noisy processed signals to the product of their RMS intensities. This is then converted to SDR which can be used like SNR for CSII calculation. Changes to signal envelope are measured using STI, speech-based STI or STOI, the latest of all.

Christiansan et al. [41] and Kates et al. [42] observed that the intelligibility predicted by CSII with IBM-processed speech was low whereas the actual intelligibility was high. In our case, the STOI results in figure 4.9 suggested that intelligibility of speech input to the pretrained network is higher than expected. However, there is this poor correlation between objective predictions by STOI and actual subjective tests as explained by other researchers.

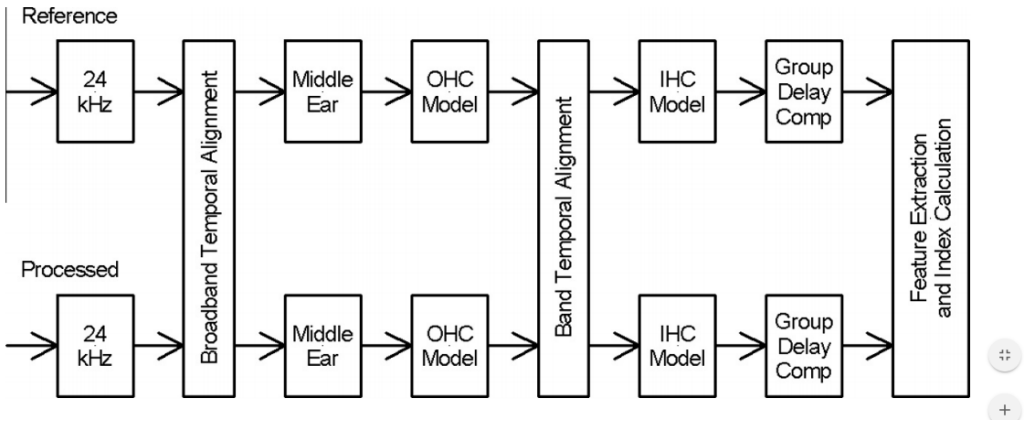

Figure 4.10: HASPI model [42]

The coherence-based speech intelligibility indices do not evaluate the envelope modifications well and the envelope correlation-based measures do not consider the short-time variations. A procedure that combines coherence with envelope modulation changes called HASPI was developed by [42] to get accurate results. It uses an auditory model (figure 4.10) that combines

coherence and envelope fidelity and incorporates aspects of normal as well as hearing-impaired auditory functions.

HASPI resamples both the reference and target signal to 24 kHz. It requires the signals to be time aligned. There are three time alignment steps. The first time alignment step is after resampling, followed by a middle ear filter to capture the low-frequency and high-frequency attenuations in equal loudness contours at low signal levels. The resultant signal is passed through a linear auditory filter bank with the filter bandwidth adjusted to input signal intensity and outer hair cell (OHC) damage. Dynamic compression is provided based on the signal from filter bank. A parallel gammatone filter-bank is used to cover frequencies in the range 80 to 8000 Hz with 32 bands. It processes the signal in two ways: one that takes envelope and incorporates the hearing loss due to OHC damage from the control filter bank and the other that considers the basilar membrane motion. The last temporal alignment will compensate the frequency-dependent group delay of gammatone filters. Thus, both the envelope and vibration level are modeled for speech intelligibility and they are converted to dB above the auditory threshold.

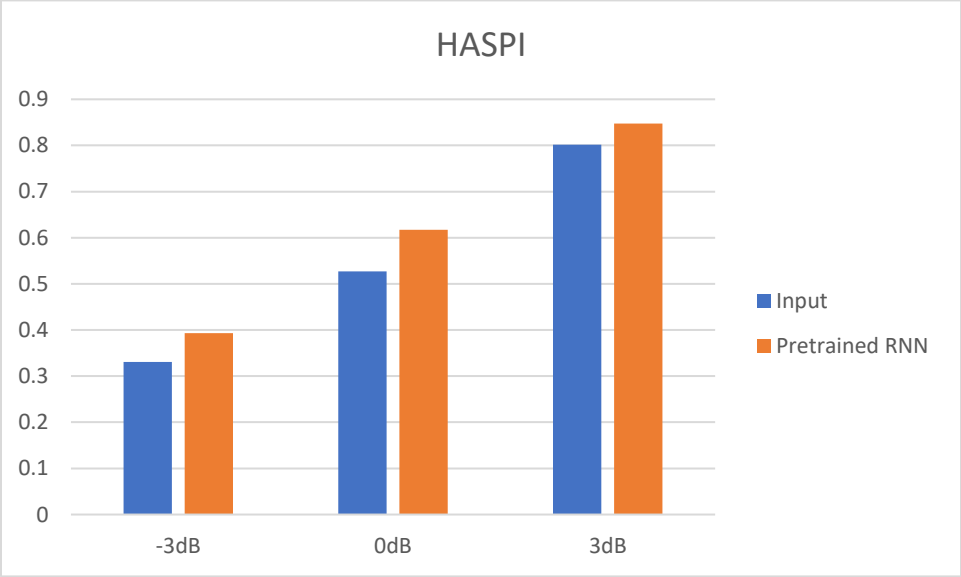

Figure 4.11: Comparison of HASPI values between noisy speech and the output of pretrained RNN model at SNRs -3 dB, 0 dB and 3 dB

The envelope output from the auditory model in frequency domain is the short-time log magnitude spectrum. A set of half-cosine basis functions are applied to the short-time spectrum of each segment of the envelope output to get the cepstral sequences. The inverse Fourier transform

will give cepstral coefficients equivalent to MFCC. If the low frequencies of the signal have more energy than high frequencies, the envelope samples after fitting the basis function for spectral tilt would give a positive value in time domain and vice versa. The basilar membrane (BM) output of auditory model is divided into 16-ms segments as well. The short-time normalized cross correlations for the low, mid and high-level segments (classified based on intensities) are averaged across time and frequency to get auditory coherence values. These three coherence values linearly weighted together with the normalized cross-correlation of the cepstral sequences (after removing the silences) for the reference and processed signals, followed by a logistic transformation will give the HASPI intelligibility index value. The HASPI values for input and output of pretrained RNN model (figure 4.11) look more closer to subjective perception.

4.6.3 Hearing aid speech quality index (HASQI)

HASQI is a quality index developed by [43] that combines envelope and TFS characteristics to provide a more accurate and robust prediction. It works with both normal hearing and hearing-impaired situations. It has four components: the envelope-based cepstral correlation, TFS-based vibration correlation, a cepstral-vibration product and a spectral shape component. The cepstral correlation (CC) is calculated in a similar way as that of HASPI. The third power of the average cepstral correlation will give the CC component.

The vibration correlation part is similar to the short-time coherence used in HASPI. The BM vibration signals in each band of the segment in clean and processed speech are cross correlated and then a frequency-dependent weight is applied. Finally, the vibration correlation component is the third power of the value obtained after summing all the weighted cross-correlations across segment and frequency bands and weighting the sum by the sum of weights. The cepstral-vibration product takes the product of square of cepstral correlation index and the BM vibration index. The above three components constitute the non-linear model.

The linear index is based on changes in long-term spectrum of the signal. Specifically, it uses the differences in excitation pattern and differences in slopes of excitation patterns. The standard deviation is calculated for both changes in excitation pattern and their slopes and a linear combination of them is subtracted from 1 to get the output of linear model. HASQI version 2 is the product of the index from non-linear processing and linear processing, which is found to be

the most accurate in estimating the subjective ratings. The HASQI values of output of pretrained RNN model in figure 4.12 show improvement in speech quality at all SNRs.

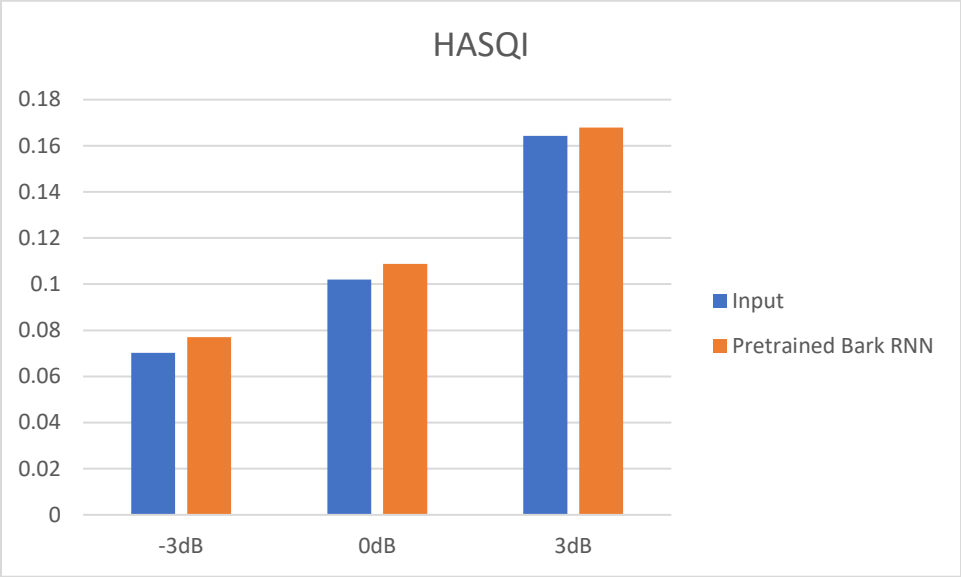

Figure 4.12: Comparison of HASQI values between noisy speech and the output of pretrained RNN model at SNRs -3 dB, 0 dB and 3 dB

4.7 Gammatone filter bank-based speech enhancement

The RNN-based model is built with Bark scale as reference. This architecture looks at energy in 22 different frequency bands for the deep learning model to estimate gains in these bands. The Bark scale has just 10 bands in the low frequency region up to 2000Hz. The speech content will be usually more in low frequencies. Therefore, it is important to have higher spectral resolution in low frequencies to recover clean speech from noisy ones.

There are other scales in which the number of bands are higher and can improve the frequency resolution significantly. Choosing a scale with too many numbers of bands would increase the complexity of the model. GammaTone (GT) filter bank with 32 bands can accommodate 10 bands up to 1000Hz and 15bands up to 2000Hz. Hence Gammatone filter bank is selected to evaluate if the increase in frequency resolution provides better speech intelligibility.

GT filter bank is a widely used scheme to model human auditory system. It was introduced by [44] to explain the impulse response of auditory system based on neural firing times. The impulse response of GT filter is given by

$$g_t(t) \quad \alpha \quad t^{n-1} exp(-2\pi bt)\, cos(2\pi f_0 t + \emptyset) \quad (t \geq 0) \tag{4.10}$$

n is the order of GT filter

b controls the relative shape of envelope and duration

f_0 frequency of amplitude modulated carrier tone

$\emptyset$ phase of the carrier tone

It is equivalent to an amplitude modulated carrier tone with an envelope proportional to the magnitude indicated in the above equation. Increasing n will make the envelope less skewed and the duration of impulse response decreases as b increases. The GT scale has 32 bands (Table 4.1) with following center frequencies:

Band	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Center frequency	50	92	140	195	258	331	415	512
Band	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Center frequency	622	750	896	1065	1259	1481	1738	2032
Band	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
Center frequency	2370	2760	3207	3721	4313	4993	5775	6673
Band	**25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**
Center frequency	7707	8895	10261	11832	13637	15714	18100	20845

Table 4.1: Gammatone frequencies

In this study, the GT filterbank is designed using Malcolm Slaney's Auditory toolbox. MakeERBFilters function can be used to create the filter coefficients. As per Nyquist criterion, the coefficients are in the frequency range 50 Hz to 24000 Hz since we use full-band speech. The ERBFilterBank in the toolbox extracts the filter output for an impulse response with FFT size of 960 based on the filter coefficients created. The frequencies and the filter output will be created in the descending order and hence, fliplr (Matlab) is applied to the output to set it right.

4.7.1 Band energy

The time domain values of each frame have to be transformed into frequency domain for computation of energy. Fourier transform is applied to the time domain value and 481 values are

considered for further processing. The real and imaginary parts are squared and added for every sample of the frame. For a given band, the square of the signal and GT filer response for each sample are multiplied and added. Here again, only 481 values of filter response are considered since the other 479 values form a mirror image. The energy in each band is calculated using equations (4.2) and (4.3) but with a different set of w_b vectors specific to Gammatone filterbank.

4.7.2 Band correlation

The band correlation is dependent on the original signal and a pitch delayed version of it, both in Fourier domain. The product of real parts and that of imaginary parts are added together, for every sample in the frame. The resultant is a 1x481 vector which is weighted with the filter response (481x32) to get the band correlation for each band (refer equation 4.4).

4.7.3 Interpolated band gain

The gain values for each band is calculated using equation (4.5). The resultant values are brought to the range -1 to 1. During testing, these 32 values estimated by the neural network have to be interpolated to create 481 values and are then multiplied with the input signal values in frequency domain. These interpolated gain values are used to amplify or attenuate the speech signal:

$$k(b) = \frac{cf}{fs} * N \tag{4.11}$$

cf is center frequency, fs is sampling frequency and N is the FFT size

$$g[k(b) + j] = m * [k(b) + j] + c \tag{4.12}$$

$m = \frac{E(b+1) - E(b)}{k(b+1) - k(b)}$ is the slope,

$c = E(b) - m * k(b)$ is the intercept component,

k(b) is the band start and j is the frequency component within the band for which gain must be interpolated.

Inverse transform is applied to this signal followed by windowing and overlap add to resynthesize the time domain signal. The comparison of HASPI values indicating the change in speech intelligibility with the use of Gammatone features is given in figure 4.13. RNN trained with GT features gives inferior intelligibility index values than respective inputs at -3, 0 and 3 dB SNRs.

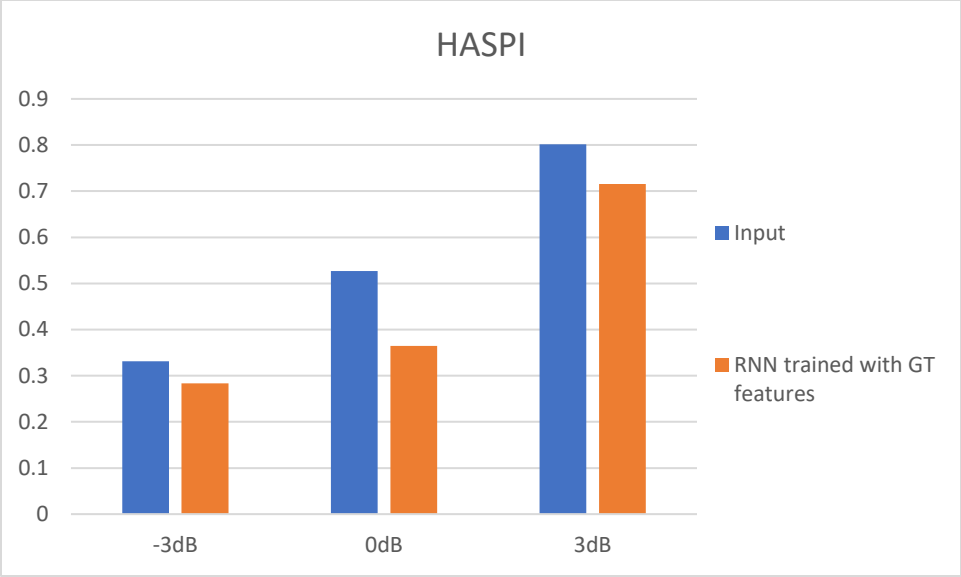

Figure 4.13: Comparison of HASPI values between input and output of RNN trained with Gammatone features at SNRs -3dB, 0dB and 3dB

The implementation of the RNN model, database, evaluation metrics and results for Gammatone filterbank based gain estimation are discussed in this chapter.

Chapter 5

5. Experimental results

Chapter 5 elaborates the noise suppression results obtained using two machine learning-based models and compares them with a more conventional digital signal processing technique in openMHA. It also describes the portable implementation that would be helpful for subjective listening tests. The chapter also has the results for dereverberation trials.

5.1 Open Master Hearing Aid

Open Master Hearing Aid (OpenMHA) is an open source software platform for real-time hearing aid software development [45]. It is developed by Hoertech gGmbH together with University of Oldenburg. The large rate of hearing loss (~13% of US population) and its high economic impact without adequate rehabilitative solutions tell us that this is an area that needs more attention. Therefore, National Institutes of Health came up to fund and facilitate a project that provides new technical solutions through research and development.

This open-source platform comes with a software development kit (C/C++ SDK) including a library for signal processing for algorithm development. It also has a set of Matlab and Octave tools to support offline testing. OpenMHA provides real-time runtime environments for PC platforms that have standard sound hardware and ARM platforms. It has some baseline algorithms that can be used individually or in combination for signal processing.

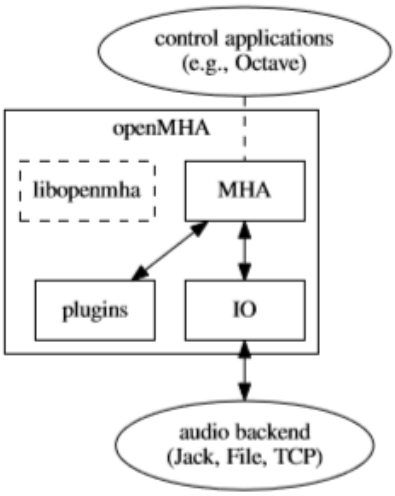

Figure 5.1: OpenMHA framework [45]

This 5-year project started in 2017 with an upgrade plan for every year. The first-year version of this program aimed at utilities for algorithm development and real-time support under Linux on PC platforms and for Beaglebone black ARM platforms. OpenMHA can be split into four components (figure 5.1). MHA is a command line application (CLA) that can be used to start, process and quit the audio processing. The second component is plugins part which provides functionalities of built-in signal processing algorithms. The MHA command line application can incorporate various built-in plugins during algorithm development. The other components are audio input-output modules (IO) and openMHA toolbox library (libopenmha).

The MHA CLA is a plugin host and can load other plugins and IO modules. The IO module for real-time signal processing is "MHAIOJack" which interfaces with Jack Audio Connection Kit, whereas "MHAIOFile" and "MHAIOTCP" for audio file access and TCP/IP based signal exchange respectively. Every plugin implements one algorithm and in combination, they can serve like a virtual hearing aid. The openMHA toolbox library provides a user-friendly mechanism to integrate real-time safe runtime configuration updates into every plugin. The entire framework can be controlled using Matlab or Octave environments.

The audio in and out can be offline or real-time based on the audio backend required. The real-time processing in openMHA requires JACK to be installed and activated. The framework and plugins require configuration of settings for a given signal processing task. This can be done through a text-based configuration language. The number of input channels could be single or multiple based on the type of waveform used or the number of microphones that the hearing support system uses in real-time. The number of output channels, "nchannels_in", is automatically learnt by MHA and can be read from the read-only variable "nchannels_out". Signal processing in real-time is done with chunks of audio data just like any other conventional algorithm and it can be done in either time-domain or frequency domain. In case of spectral domain, the frame size, specified by "fragsize", is kept at 64 samples, window length to be 128 and FFT length at 256 for a sample rate "srate" of 44100 Hz in many applications.

An openMHA plugin can behave like a plugin host. Plugins that can load other plugins in this way are called bridge plugins. When a bridge plugin is called by openMHA, it will perform its analysis and invoke the particular signal processing part in loaded plugin. The processed signal is cleanly transferred to next plugin until call to that plugin returns and then signal is resynthesized.

This allows combination of analysis and synthesis methods in a single plugin. The processed signal is then returned in the original domain back to the caller of bridge plugin.

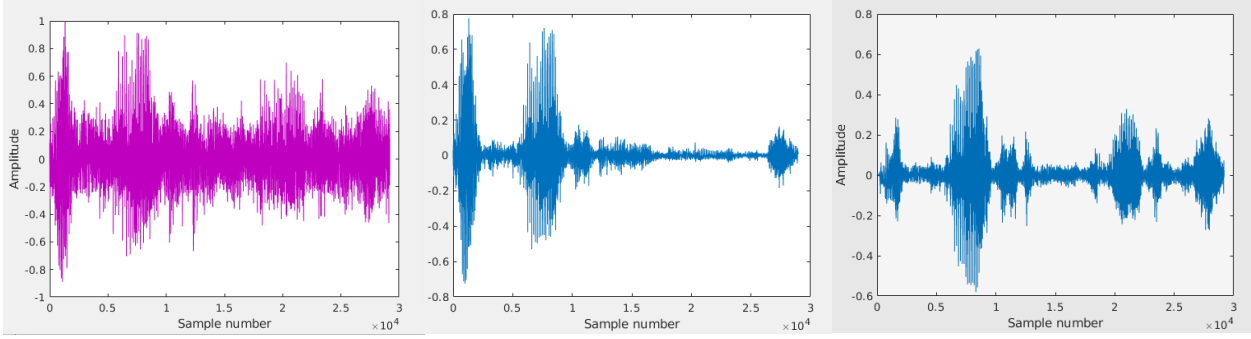

Figure 5.2: (from left to right) Waveform representation of a) Input noisy speech b) Output of pretrained RNN c) Output of MHANR algorithm

There are a variety of baseline algorithms in openMHA. These include linear predicted coding (LPC), beamforming, adaptive feedback cancellation, FFT filter bank, combining filter channels, IIR filter, NLMS algorithm, dynamic compression, resample, overlapadd and so on. Of these the single channel noise reduction scheme is used here for evaluation and comparison.

Now, speech enhancement in noisy speech (figure 5.2) is tested with the more conventional signal processing method used in openMHA. OpenMHA comes with a built-in algorithm for single channel noise reduction. It works at a sampling frequency of 16 kHz. It breaks the entire noisy file into fragments of 256 samples. The single channel noise reduction (SCNR) scheme is developed to work in frequency domain. The window length is 512 with FFT size equal to window length. The output of SCNR has improved speech intelligibility as well.

The first plugin that is loaded in SCNR scheme is overlapadd. MHA can support different audio backends like sound cards, JACK, sound files or the network. The backend is specified to be MHAIOFile in this scheme. The mhachain plugin acts as bridge plugin for processing. It loads algorithms like noisePowProposedScale and timoSmooth for noise reduction and sends the processed frame back IO plugin after overlapadd operation.

The speech intelligibility of the processed speech from pretrained RNN network and MHA noise reduction algorithm are compared with that of input using HASPI in figure 5.3. The HASPI

values are improved by 4-9% with both the algorithms and the conventional signal processing method and the pretrained machine learning based model are comparable at this stage.

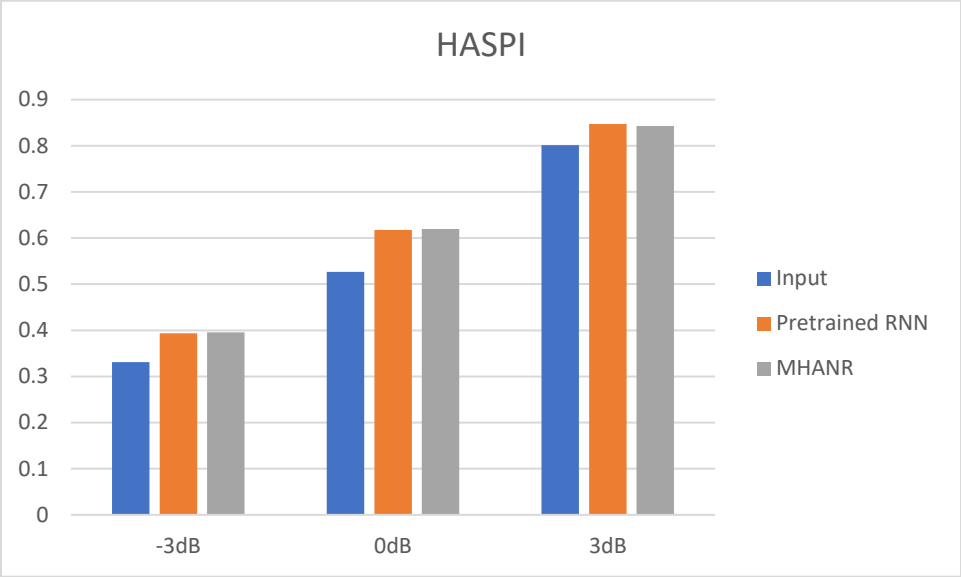

Figure 5.3: HASPI comparison between pretrained RNN and MHANR

5.2 DNN-based speech enhancement

In this method proposed by [46], a DNN is modeled as mapping function from noisy to clean speech features. Log-power spectral features from pairs of noisy and clean speech data are used as they are found to offer perceptually relevant characteristics. These are calculated by taking short-time Discrete Fourier transform of each windowed frame of input signal.

The framework is developed in two stages, refer figure 5.4. In the training stage, a DNN-based regression model is trained using these log-power spectral features. The model architecture for this training is a feed-forward neural network with components to learn non-linearity. Specifically, a deep generative model is pretrained with these features of noisy speech, which are all normalized to zero mean and unit variance. This is done by stacking multiple restricted Boltzmann machines (RBMs), after training in an unsupervised greedy fashion as indicated by [57] Contrastive Divergence (CD) is the optimization algorithm used to update parameters during this pre-training process. Then BP with MMSE-based objective metric between the log-power spectral features of the estimated and reference clean speech is used to train the DNN.

In this method, there is no assumption about the relationship between clean and noisy speech during training. It can learn to separate speech from noise automatically. This model also captures acoustic context information along the time-axis as well as frequency-axis while assuming the standard independence among different dimensions. The post processing involves global variance equalization to address the over-smoothing problem in the reconstructed signal based on network estimates. Dropout regularization is used to improve generalization. This DNN is also provided with the online noise information to better predict clean speech. The reconstruction of the enhanced signal is done by taking exponent of enhanced features, followed by inverse Fourier transform and overlap-add method.

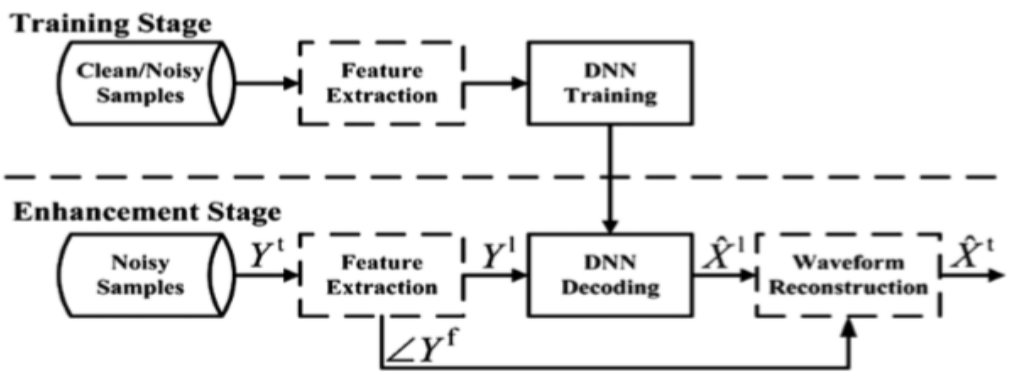

Figure 5.4: Speech enhancement by DNN model [46]

The DNN has 3 hidden layers with 2048 units in each of them. It is trained with clean speech from TIMIT database. The noise types used are babble, restaurant, street, white Gaussian

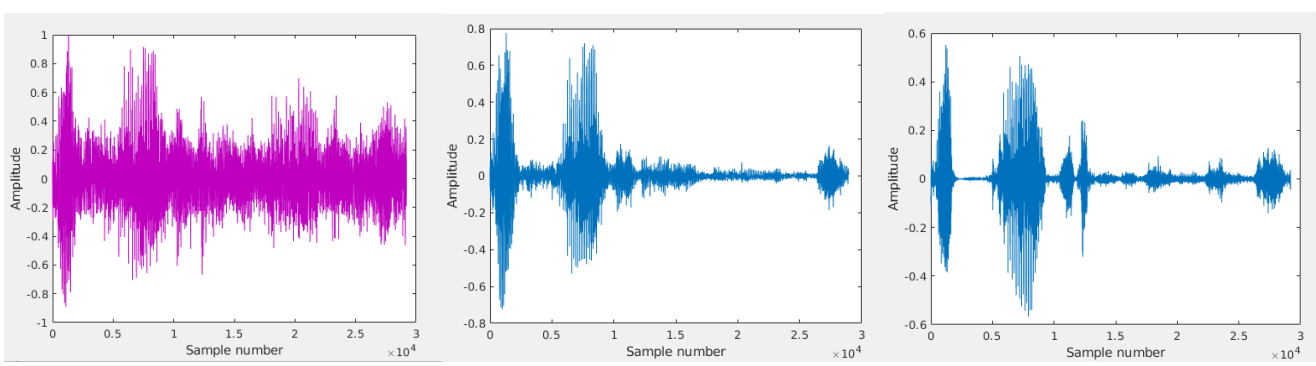

Figure 5.5: (from left to right) Waveform representation of a) input noisy speech b) output of pretrained RNN model c) output of RBM-based DNN model

noise and 100 other environmental noises. The speech and noise were mixed at six different SNRs ie, 20, 15, 10, 5, 0 and -5 dB to build a variety of training examples. In total, 2500 hours of noisy training data were created. The clean speech and noise signals were sampled at 8kHz with frame length of 32 msec and an overlap of 50%. The log-power spectral feature set had a dimensionality of 129.

This RBM-based model has shown great promise in noise reduction in unseen sounds. The research work performed by authors suggested that it can perform much better than conventional techniques in non-stationary noise conditions. The model is trained in different configurations and an optimal configuration was chosen by the developers after evaluating their performances carefully. The waveform representation of output of pretrained model and RBM model are given in figure 5.5. The HASPI values of RBM model (figure 5.6) point out that it can perform better than the pretrained RNN-based model for -3 dB SNR and SNRs in that range while pretrained RNN gets a slight advantage as the SNR of input increases above zero.

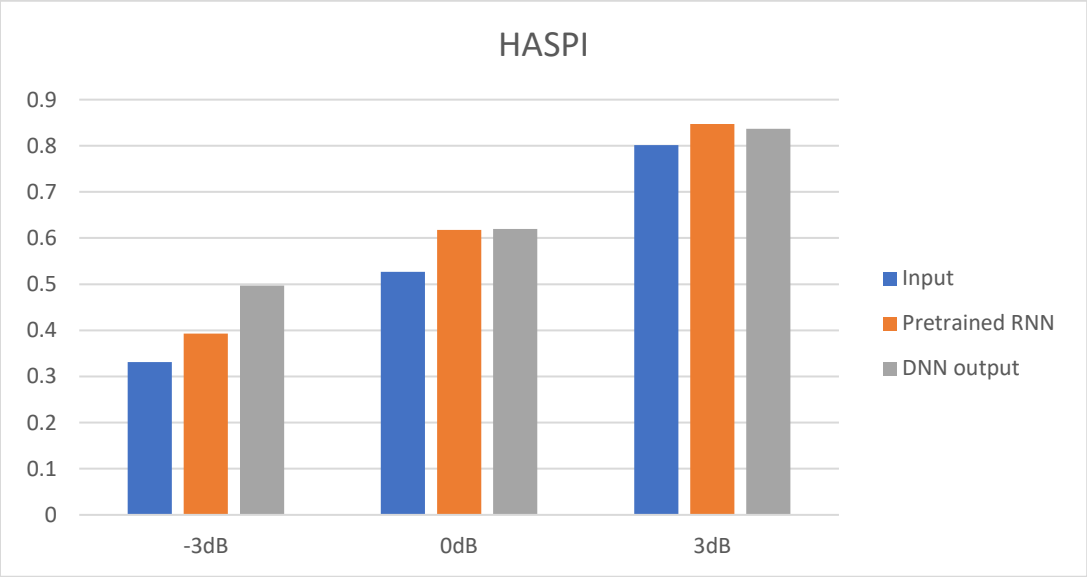

Figure 5.6: HASPI comparison between pretrained RNN and DNN models

5.3 Bark filter bank-based speech enhancement

The speech segments are usually concentrated in the frequencies up to 10 kHz though the audible range extends up to 20 kHz. The gammatone filter bank provided better spectral resolution

at lower frequencies and was expected to support better speech enhancement than the Bark scale, with the help of more features from speech-specific frequencies. However, our results (figure 23) suggest that Bark filter bank is still able to provide better performance for this dataset. Therefore, the Bark filter-bank based enhancement is further explored in this section.

5.3.1 Scaling of input waveforms

Here, the RNN model is trained with different scales of input. The input waveforms were scaled by max value in one iteration and in the range -1 to 0.9 in another. The results as in figure 5.7 indicate that the unscaled waveforms provide better results when compared to scaled ones.

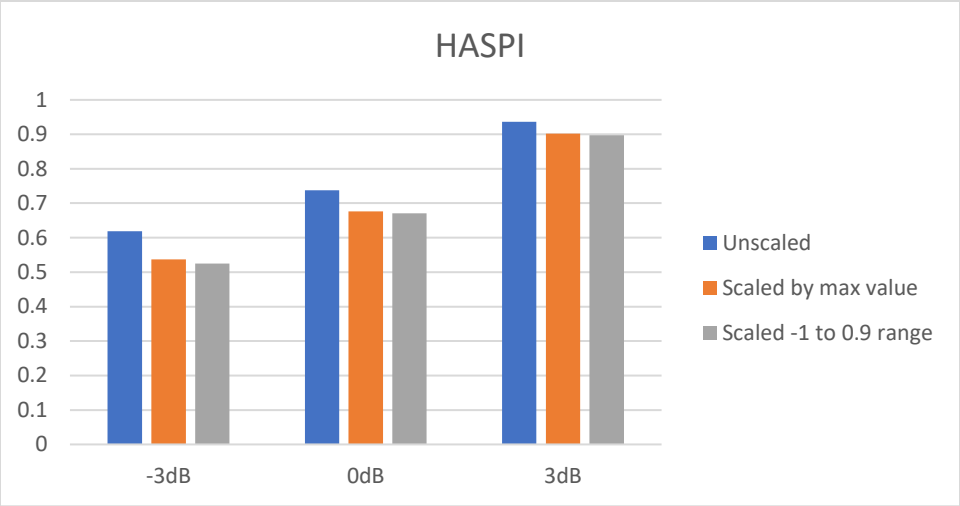

Figure 5.7: HASPI values for different scaling configurations

5.3.2 Different SNRs

The level of noise in realistic situations could be arbitrary. It is important to eliminate or reduce the background noise to a level in which speech can be understood. The energy of speech with respect to noise is indicative of the speech intelligibility and it can be specified by SNR. Many different SNRs were tried for training in this section (figure 5.8).

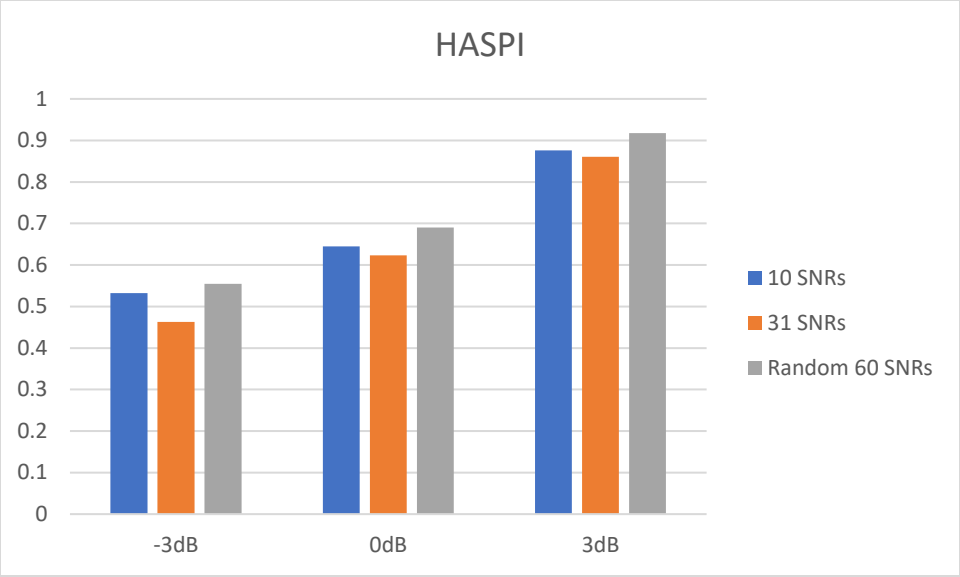

Figure 5.8: HASPI values for different SNR configurations

Speech and noise were mixed at 10 different SNRs manually during the initial trial. The following trial had 31 SNRs in the range -10dB to 20dB. For more SNRs, speech and noise were mixed based on a random number generator that would randomly select an SNR between -40dB and 20dB. It is seen that the performance gets better as more and more SNRs are included during training.

5.3.3 Acoustic context

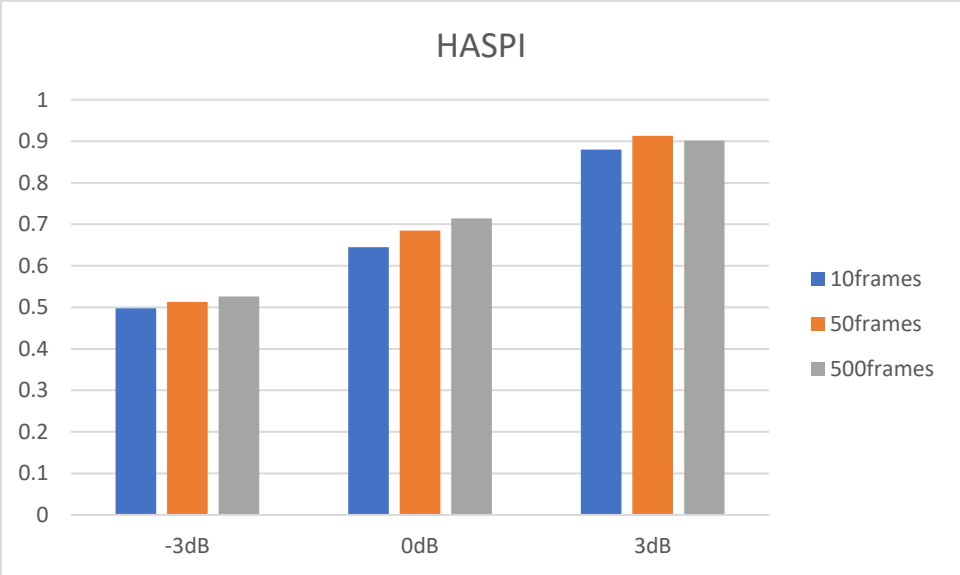

Figure 5.9: HASPI values for different window size

A noise estimation algorithm usually needs time to estimate the noise spectrum and help spectral subtraction or any equivalent algorithm to extract clean speech out of noisy speech. It is again dependent on a given dataset to decide the duration of context required and is defined by window size. It is the number of frames that needs to be considered per sequence for training. The window size is varied in steps from 10 to 2000 and found similar performance in training and validation sets with 2000 frames (figure 5.9).

5.3.4 Final configuration

A final iteration of training is done with the parameters learnt in previous trials. This is followed by testing with a combination of clean speech in HINT database and multitalker babble noise. The HASPI value comparison (figure 5.10 and table 5.1) and the spectrogram of output are compared with that of outputs of previously discussed models.

SNR	Input	Pretrained Bark RNN	MHANR	DNN output	Trained GT RNN	Trained Bark RNN
-3dB	0.331	0.3932	0.3953	0.4967	0.2833	0.6189
0dB	0.5269	0.6174	0.6194	0.6199	0.3645	0.7372
3dB	0.8014	0.8474	0.8431	0.8369	0.7159	0.9364

Table 5.1: Intelligibility (HASPI) index values - Speech with multitalker babble for different models

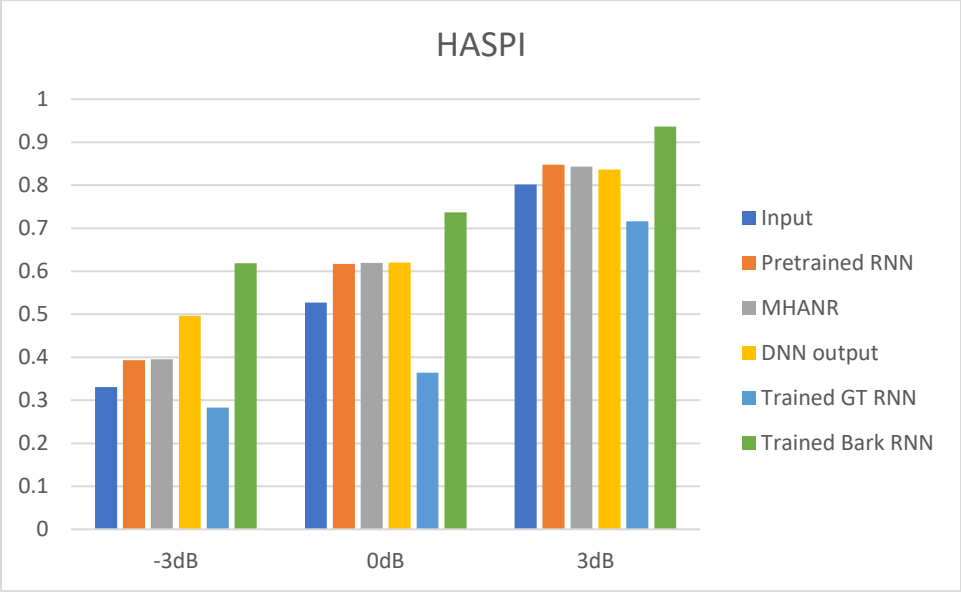

Figure 5.10: Graphical representation of HASPI values - Speech with multitalker babble for different models

There is an improvement in intelligibility index of at least 15% with trained Bark RNN model. In case of SNRs of the range -3dB, the improvement goes up to 85%.

SNR	Input	Pretrained Bark RNN	MHANR	DNN output	Trained GT RNN	Trained Bark RNN
-3dB	0.0703	0.0771	0.078	0.0797	0.0643	0.1237
0dB	0.102	0.1088	0.1185	0.0972	0.0762	0.1599
3dB	0.1643	0.1679	0.1742	0.1445	0.1385	0.2513

Table 5.2: Quality (HASQI) index values - Speech with multitalker babble for different models

The HASQI values are calculated too (table 5.2 and figure 5.11) and they indicate a minimum of 53% increase (at 3dB) when compared to input and it goes up to 76% for -3dB SNR. The spectrograms (figure 5.12) reflect this prediction as well with the modulation in lower frequencies recovered better by this trained RNN than others. This makes the model clearly a preferred one out of the five algorithms compared here.

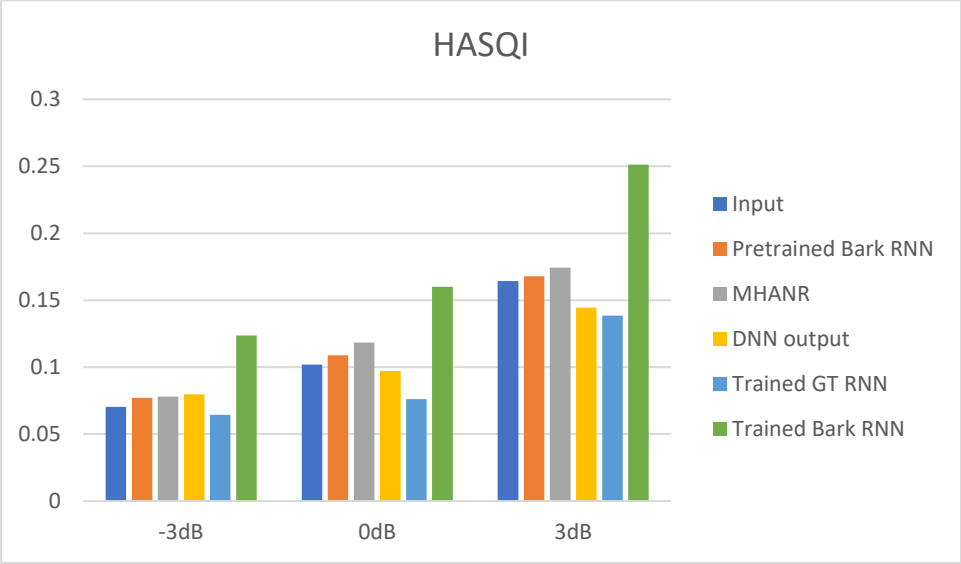

Figure 5.11: Graphical representation of HASQI values - Speech with multitalker babble for different models

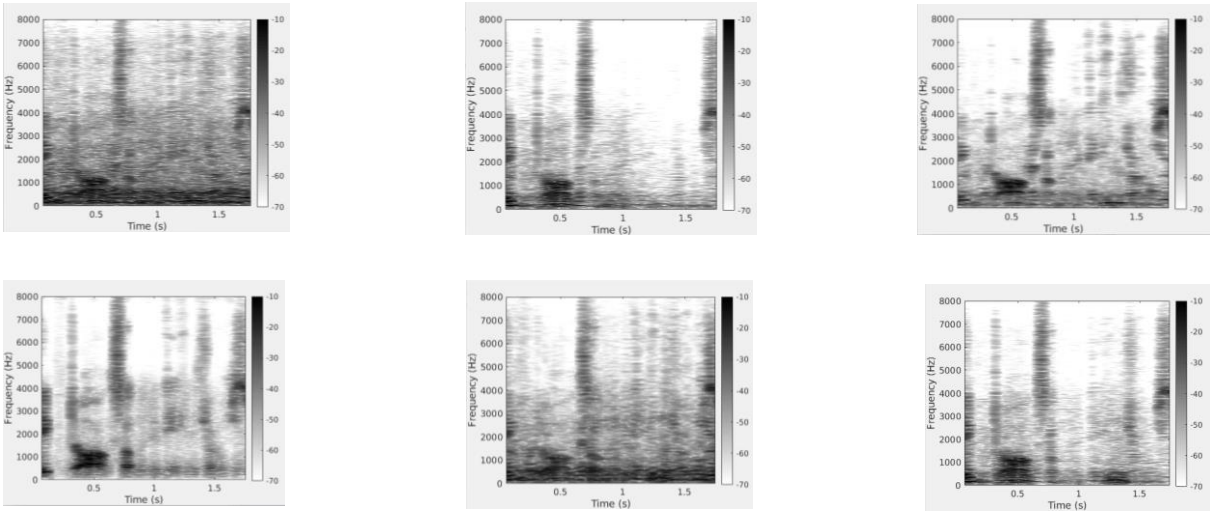

Figure 5.12 (Top from left to right): Spectrogram of a) Noisy input at 0dB SNR, b) Output from pretrained RNN model, c) OpenMHA output (Bottom from left to right) d) RBM model output, e) Output from trained GT RNN model and f) Output from trained Bark RNN model

5.4 Dereverberation

Reverberation is a phenomenon that causes reflection of sounds from surfaces to add to the signal when it is being listened to at a distance from source. It is prominent in enclosed environments where sounds may reflect from hard surfaces like walls, ceiling, and other objects. Hence the design of environments that can cause reverberation may take the inverse square law drop-off of sound intensities. However, there exists many reverberating environments in realistic scenarios that can deteriorate speech intelligibility.

The low frequency components are the ones that can usually travel farther than high frequencies during reverberation. This would mean that the direct signal is overlapped by the low frequency reflected components and may have a larger impact on hearing-impaired listeners. Reverberation is specified by the reverberation time, which is the time taken by the signal to drop its intensity by 60 dB. It is important to reduce the reverberation time and limit it to minimum required during speech enhancement.

Different versions of RNN model are tested with reverberated speech to evaluate their performance under reverberating conditions. The impulse response for every type of enclosed environment was calculated based on data in Aachen Impulse Response (AIR) database [47].

The data is collected using a setup shown in figure 5.13, which consists of a dummy head that is placed at a specific distance from the sound source. The two microphones placed close to the ear

Figure 5.13: Measurement for Room Impulse Response calculation [47]

position of the dummy head will collect the reverberated speech for that environment. The impulse response for the environment was extracted based on a suitable deconvolution technique.

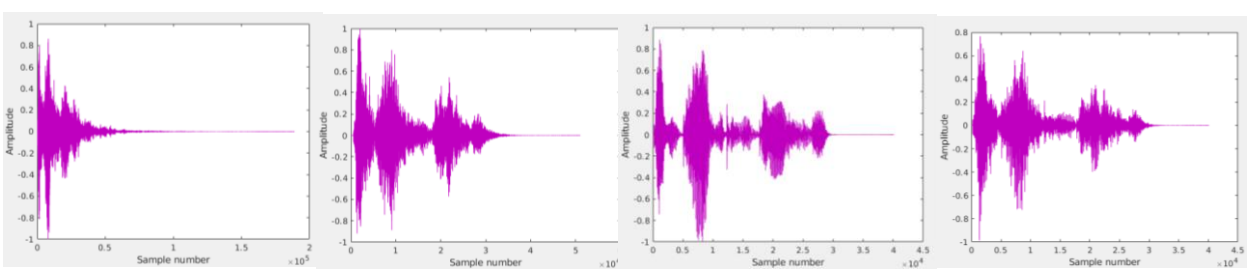

Figure 5.14 (Left to right): a) Speech with impulse response of Aula Carolina b) Speech with lecture c) Speech with booth d) Speech with office

Reverberated speech for training is created by convolving the signal with the impulse response of the environment (figure 5.14). A dataset was created using the speech samples from

HINT database and the impulse response from AIR database. AIR database includes data for Aula Carolina, Lecture, Booth and Office environments.

5.4.1 Convolution and addition

In this experiment, three experiments were done to check the difference in performance with the way speech is convolved with impulse response. The initial trial was done by convolving every sentence of each list in the HINT database with impulse response of the environment and then adding them together. This would create a file with $N_1 + L_1 - 1$ samples for training where N_1 and L_1 are the length of individual clean speech file and impulse response respectively. The training data includes environments like aula Carolina, lecture and booth. In order to compare this reverberated speech with equivalent clean speech, L_1-1 zeros are added at the end of each sentence of clean speech. A single clean speech and reverberated speech are created by adding all these convolved files respectively.

The second trial involved addition of all individual clean speech files in each list together and then applying convolution. Here again, the length of the single reverberated speech file created after convolution would be N+L-1 samples where N is the length of the single file created after adding all individual clean speech files and L is the length of impulse response respectively. The length of the single clean speech file is also adjusted to N+L-1 using zero padding at the end.

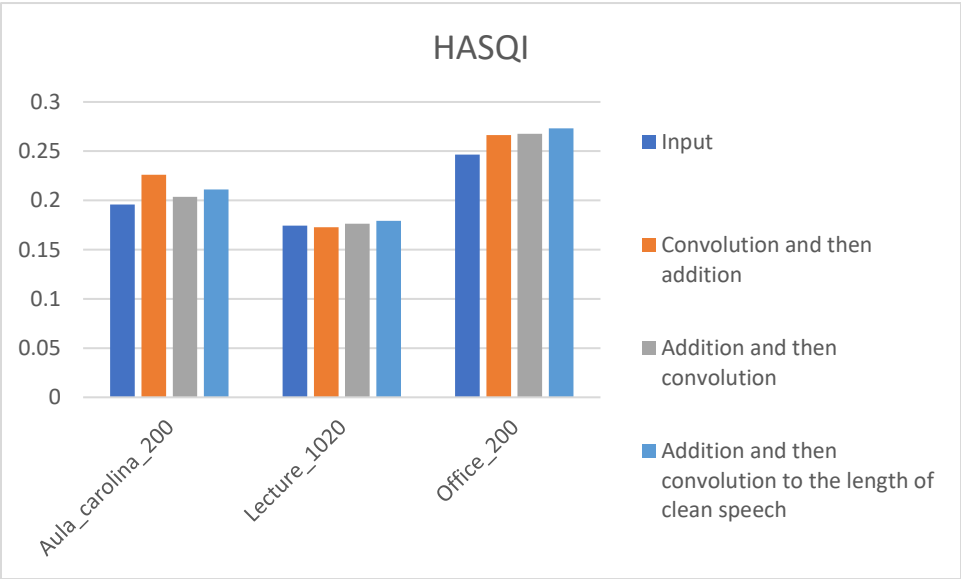

Figure 5.15: HASQI values with different methods for convolution

HASQI is used to evaluate the quality of input and output. The results in figure 5.15 indicate that convolving individual file and comparing them with respective clean speech file zero padded at the end facilitates better learning of the network and yields better results during testing. The third trial was using the same reverberated speech file from second trial but only to the length of clean speech and the quality index considered the clean speech length as well. The results from third trial indicate an improvement over the second one. However, it is the convolution of individual file and zero padding of clean speech that gave the best results.

5.4.2 Acoustic context

We have seen that the context in which speech is listened to has been found to have an impact in the effectiveness of noise reduction. Therefore, different iterations of training were conducted by varying number of frames of reverberating speech. The window size is varied from 100 frames to 2000 frames. As in figure 5.16, the window size with 2000 frames yields the best results for all kinds of reverberating conditions.

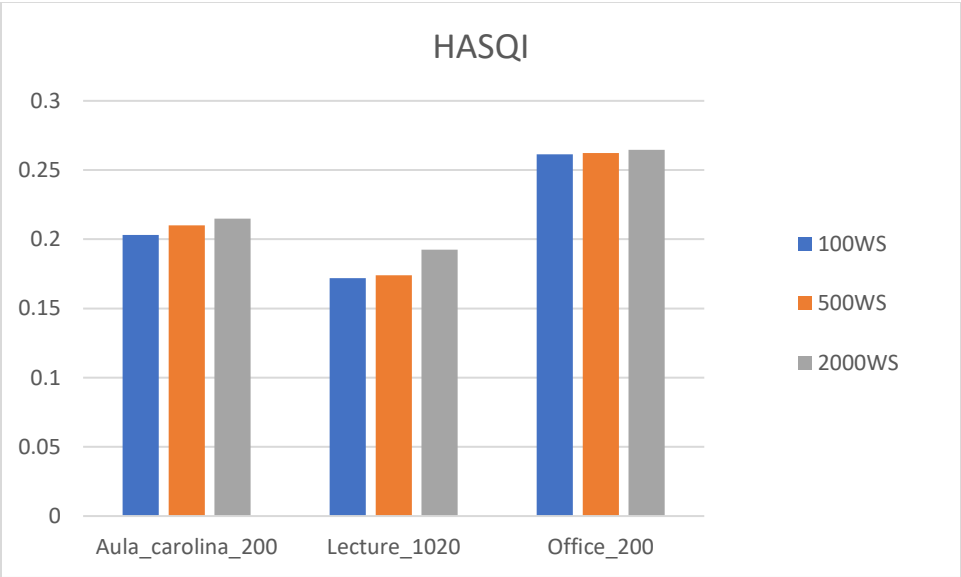

Figure 5.16: HASQI values for different window size

5.4.3 Comparison of results

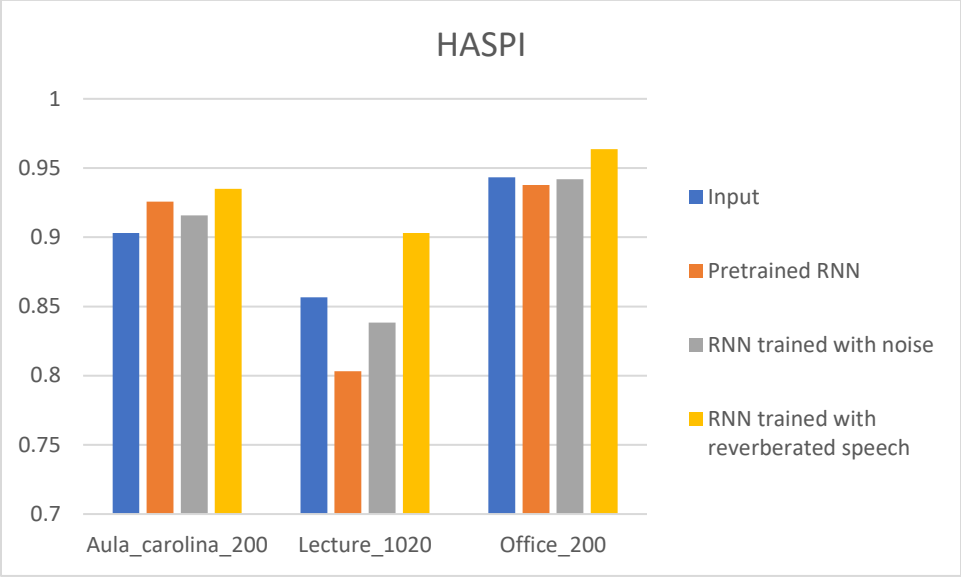

Figure 5.17: Graphical representation of HASPI values - Speech with reverberation for different models

	Input	Pretrained RNN	RNN trained with noise	RNN trained with reverberated speech
Aula_carolina_200	0.9031	0.9257	0.9158	0.935
Lecture_1020	0.8565	0.8031	0.8383	0.903
Office_200	0.9432	0.9378	0.9418	0.9636

Table 5.3: Intelligibility (HASPI) index values - Speech with reverberation for different models

In this section, the performance of RNN trained with reverberated speech is compared with other RNN models dealt with so far, in terms of speech intelligibility and quality. In general, the intelligibility in reverberating environments will be high (figure 5.17 and table 5.3). However, both HASQI and HASPI values are evaluated for the final configuration.

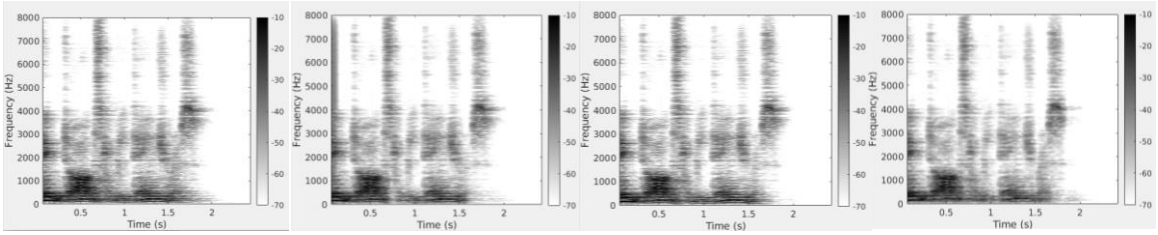

Figure 5.18 (from left to right) a) Input reverberating (office) speech, b) Output of pretrained RNN model, c) Output of RNN model trained with noise, d) Output of RNN model trained with reverberated speech

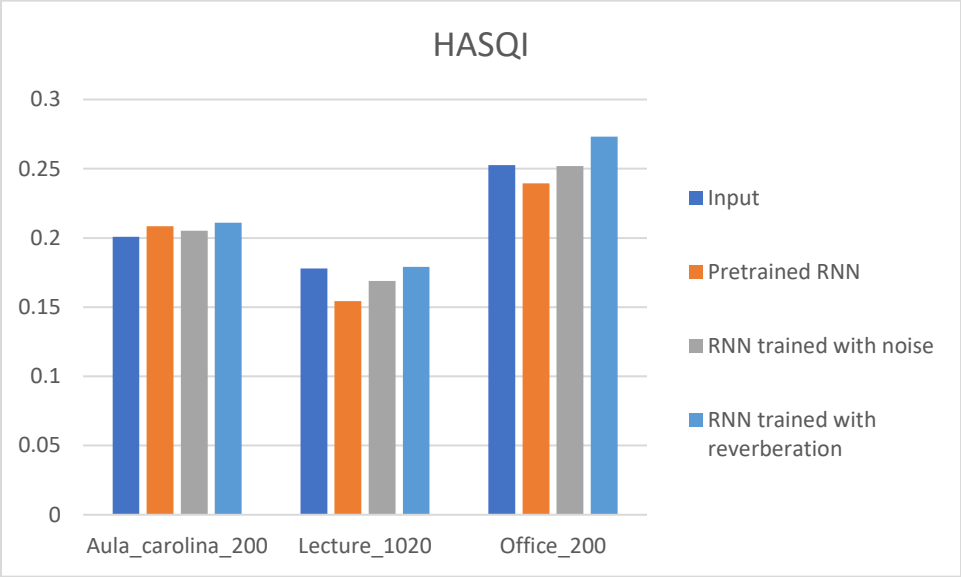

Figure 5.19: Graphical representation of HASQI values - Speech with reverberation for different models

	Input	Pretrained RNN	RNN trained with noise	RNN trained with reverberation
Aula_carolina_200	0.2008	0.2084	0.2053	0.211
Lecture_1020	0.178	0.1543	0.169	0.1792
Office_200	0.2527	0.2394	0.2519	0.2731

Table 5.4: Quality (HASQI) index values - Speech with reverberation for different models

The HASPI values indicate an improvement of up to 5% (in case of lecture). Since the HASPI values are already high even with reverberation, the main focus here is on speech quality. In figure 5.19 (table 5.4), the HASQI values for pretrained RNN when subjected to babble noise

is compared with that of RNN trained with noise and RNN trained with reverberation. The improvement in speech quality is the best with RNN trained with reverberation, providing 5-8% HASQI increase when compared to input. It is also interesting to see good generalization of this model for the unseen office reverberation data.

5.5 Portable implementation

Speech enhancement algorithms that are being developed need to be incorporated in a small assistive hearing device for the listeners to get benefited. Besides, subjective listening test is the validation procedure that can evaluate the performance of an algorithm based on user experience. This is usually done with a lab setup and all listeners need to be in the lab for testing. It is often a difficult task to get many listeners to a particular facility. However, if the testing device can be physically taken to the listener, just like an assistive hearing device, that would be a great assistance to the listener and would help getting more listeners attend the test. The present work also focuses on these aspects and make a portable implementation of the noise reduction algorithm.

Intel Minnowboard is chosen for the portable implementation. It is a compact Single Board Computer with 2.9” x 3.9” form factor and has a 64-bit Intel Atom processor for intensive computations. The main advantages are it can support a wide range of OSs and IDEs and comes with a built-in firmware for basic operations until an OS is in place. The hardware is open source and all the design specs can be accessed.

The Minnowboard is setup with Linux Ubuntu OS and is installed with all frameworks to run a real-time application. It is also installed with openMHA and a few built-in plugins were executed. In addition, a plugin has been developed that incorporates the RNN-based model into openMHA and is successfully implemented. This will enhance the list of algorithms that can be executed with openMHA. There are provisions to add peripheral devices like display and keyboard, which can be done as future work.

Here, the concept of openMHA and the reference algorithm based on DNN are explained. The performance of trained Bark RNN with different configurations are evaluated and compared with these two algorithms apart from the pretrained and Gammatone versions of the same model for noise reduction. The same is done for dereverberation as well.

Chapter 6

6. Conclusion

Noise reduction has been a challenging area of research since the invent of hearing aids. Most of the realistic scenarios in which communication happens is noisy. This affects both the intelligibility of speech and quality. This will have more impact on hearing-impaired listeners than the normal hearing ones because of the increased threshold for hearing. Therefore, it is important to have a robust noise reduction system in assistive hearing devices.

Digitization of analog signals offered a lot of opportunities to have custom speech enhancement for listeners. It has options to provide frequency or level-dependent gain based on audiogram. Digital signal processing (DSP) was found effective for many specific hearing impairments. However, DSP algorithms must be built for every type of undesired characteristic (noise) that needs to be reduced or removed from noisy or reverberating speech. This requires manual efforts to tune the algorithm and is time consuming. Statistical assumptions are also made about unseen signals which can be detrimental for accurate estimation of noise and thereby its reduction. Research shows that machine learning, a cutting-edge technology, is effective in many speech enhancement applications.

6.1 Summary

This thesis focused on exploring and evaluating deep learning models that can help reduce noise in different conditions for hearing-impaired listeners. An RNN model was trained with speech and noise from HINT and Edinburgh database and the speech intelligibility and quality were calculated using HASPI and HASQI respectively. The RNN model was trained with reverberating speech as well. The performance was compared with that of openMHA (a framework based on conventional digital signal processing), RBM-based model and a pretrained RNN model. The speech intelligibility of RNN trained with noise is found to be better than all other models along with improvement in speech quality. The speech intelligibility and quality were enhanced to a larger extent with the model trained with reverberating speech.

Sound cleaning is an integral component of speech enhancement in assistive hearing devices. Conventional signal processing techniques try to address stationary noise, impulsive/wind

noise or reverberation. The main focus of this project was on noise reduction using machine learning which can handle non-stationary noise apart from the ones that do not change over time. This methodology reduces the huge manual effort involved in studying different noise characteristics, developing custom algorithms that can reduce or remove different noise types and fine tune them for realistic environments. The performance of the deep learning model was evaluated objectively for hearing impaired listeners, which was based on both the intelligibility and quality aspects. Besides, the performance under reverberation was evaluated as well.

Hearing aids are expected to have fast acting algorithms while causing no artifacts or distress because of the resulting rapid switch of algorithms within it. The design of digital signal processing algorithms needs a careful tuning to find an optimum balance between noise reduction and the consequent artifacts. Machine learning techniques do not make any assumptions about noise properties to reduce noise and are purely based on the target signals in case of supervised learning. This would cause less speech distortion when compared to conventional algorithms. The project also demonstrated real-time implementation in a portable platform.

6.2 Future work

This research has witnessed impact of acoustic context and other hyperparameters on training and performance during testing. Large scale training has a bigger effect since a dataset with many features need to be accompanied with many rows of training examples for a deep learning model to learn complex patterns underlying the data. The data available for the project was limited and therefore, training with a bigger dataset is one of the primary objectives in future work.

The future work may include extending the same methodology to address environment classification. This in combination with machine learning-based noise reduction can make a robust speech enhancement approach. This is particularly suitable for real-time implementation in small hearing devices when the deep learning model is made with low complexity (just like the model used here).

References

[1] "Deafness and hearing loss". Internet: http://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss 15 March. 2018 [7 October 2018]

[2] Prajapati, Pooja G., and Devani, Anupam N. "Review Paper on Noise Reduction Using Different Techniques" *International Research Journal of Engineering and Technology*. 4.3 (2017) 522-524

[3] Levitt, Harry. "Noise reduction in hearing aids: A review." *Journal of rehabilitation research and development* 38.1 (2001): 111-122.

[4] Quintino, Camila Angélica, Maria Fernanda Capoani Garcia Mondelli, and Déborah Viviane Ferrari. "Directivity and noise reduction in hearing aids: speech perception and benefit." *Brazilian journal of otorhinolaryngology* 76.5 (2010): 630-638

[5] "A brief history of hearing aids" https://www.beltone.com/hearing-aid-guides/hearing-aid-history [10 October 2018]

[6] "The Road to the First Electric Portable Hearing Aid….and Beyond" https://hearinghealthmatters.org/hearinginternational/2015/the-road-to-the-first-electric-portable-hearing-aid-and-beyond/ [10 October 2018]

[7] "An Introduction to Digital versus Analog Hearing Aids" https://www.salemaudiologyclinic.com/an-introduction-to-digital-versus-analog-hearing-aids/ [7 November 2018]

[8] Murray, D. J., and J. V. Hanson. "Application of digital signal processing to hearing aids: a critical survey." *Journal of the American Academy of Audiology* 3.2 (1992): 145-152.

[9] Luo, Fa-Long, and Arye Nehorai. "Recent developments in signal processing for digital hearing aids." IEEE Signal Processing Magazine 23.5 (2006): 103-106.

[10] Popelka, Gerald R., et al. "Hearing Aids", Springer International Publishing (2016)

[11] Upadhyay, Navneet, and Abhijit Karmakar. "Speech enhancement using spectral subtraction-type algorithms: A comparison and simulation study." Procedia Computer Science 54 (2015): 574-584.

[12] Ricketts, Todd A., and Benjamin WY Hornsby. "Sound quality measures for speech in noise through a commercial hearing aid implementing." Journal of the American Academy of Audiology 16.5 (2005): 270-277.

[13] Bentler, Ruth, and Li-Kuei Chiou. "Digital noise reduction: An overview." Trends in Amplification 10.2 (2006): 67-82.

[14] Desjardins, Jamie L., and Karen A. Doherty. "The effect of hearing aid noise reduction on listening effort in hearing-impaired adults." Ear and Hearing 35.6 (2014): 600-610.

[15] Brons, Inge, Rolph Houben, and Wouter A. Dreschler. "Effects of noise reduction on speech intelligibility, perceived listening effort, and personal preference in hearing-impaired listeners." Trends in hearing 18 (2014): 2331216514553924.

[16] Lebart, Katia, Jean-Marc Boucher, and P. N. Denbigh. "A new method based on spectral subtraction for speech dereverberation." Acta Acustica united with Acustica 87.3 (2001): 359-366

[17] Fabry, David, and Jürgen Tchorz. "Results from a new hearing aid using "acoustic scene analysis"." The Hearing Journal 58.4 (2005): 30-36.

[18] Healy, Eric W., et al. "An algorithm to improve speech recognition in noise for hearing-impaired listeners." The Journal of the Acoustical Society of America 134.4 (2013): 3029-3038.

[19] Chen, Jitong, et al. "Large-scale training to increase speech intelligibility for hearing-impaired listeners in novel noises." The Journal of the Acoustical Society of America 139.5 (2016): 2604-2612.

[20] "Logistic-Curve.Svg" Internet: https://commons.wikimedia.org/wiki/File:Logistic-curve.svg [20 December 2018]

[21] Jarrett, K., et al. "What is the best multi-stage architecture for object recognition" In Proceedings of the International Conference on Computer Vision (ICCV'09). (2009).

[22] Glorot, Xavier, Antoine Bordes, and Yoshua Bengio. "Deep sparse rectifier neural networks." *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 2011.

[23] Goodfellow, Ian, et al. Deep learning. Vol. 1. Cambridge: MIT press, 2016.

[24] "Machine learning". Internet: https://www.coursera.org/learn/machine-learning [10 October 2017]

[25] Bengio, Yoshua, Patrice Simard, and Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult." *IEEE transactions on neural networks* 5.2 (1994): 157-166.

[26] Hochreiter, Sepp. "The vanishing gradient problem during learning recurrent neural nets and problem solutions." *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6.02 (1998): 107-116.

[27] Ring, Mark. "Learning sequential tasks by incrementally adding higher orders." *Advances in neural information processing systems*. 1993.

[28] Mozer, Michael C. "Induction of multiscale temporal structure." *Advances in neural information processing systems*. 1992.

[29] Hochreiter, Sepp, et al. "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies." (2001).

[30] G. Sun, H. Chen, and Y. Lee. Time warping invariant neural networks. In J. D. Cowan S. J. Hanson and C. L. Giles, editors, *Advances in Neural Information Processing Systems* 5, pages 180–187. San Mateo, CA: Morgan Kaufmann, 1993.

[31] T. Lin, B. G. Horne, P. Ti˜no, and C. L. Giles. Learning long-term dependencies in NARX recurrent neural networks. *IEEE Transactions on Neural Networks*, 7(6):1329–1338, November 1996.

[32] J. Schmidhuber. Learning complex, extended sequences using the principle of history compression. *Neural Computation*, 4(2):234–242, 1992.

[33] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation 9.8* (1997): 1735-1780.

[34] Olah, Christopher. "Understanding LSTM networks." *GITHUB blog*, posted on August 27 (2015): 2015.

[35] Chung, Junyoung, et al. "Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555 (2014).

[36] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[37] Valin, Jean-Marc. "A hybrid dsp/deep learning approach to real-time full-band speech enhancement." arXiv preprint arXiv:1709.08243 (2017).

[38] "Improving Deep Neural Networks: Hyperparameter tuning, Regularization and Optimization". Internet: https://www.coursera.org/learn/deep-neural-network [10 September 2018]

[39] Valentini-Botinhao, Cassia. "Noisy speech database for training speech enhancement algorithms and TTS models." (2017).

[40] Taal, Cees H., et al. "A short-time objective intelligibility measure for time-frequency weighted noisy speech." *Acoustics Speech and Signal Processing (ICASSP)*, 2010 IEEE International Conference on. IEEE, 2010.

[41] Christiansen, Claus, Michael Syskind Pedersen, and Torsten Dau. "Prediction of speech intelligibility based on an auditory preprocessing model." Speech Communication 52.7-8 (2010): 678-692.

[42] Kates, James M., and Kathryn H. Arehart. "The hearing-aid speech perception index (HASPI)." Speech Communication 65 (2014): 75-93.

[43] Kates, James M., and Kathryn H. Arehart. "The hearing-aid speech quality index (HASQI) version 2." Journal of the Audio Engineering Society 62.3 (2014): 99-117

[44] Holdsworth, John, et al. "Implementing a gammatone filter bank." Annex C of the SVOS Final Report: Part A: The Auditory Filterbank 1 (1988): 1-5.

[45] "Open community platform for hearing aid algorithm research". Internet: www.openmha.org [10 March 2018]

[46] Xu, Yong, et al. "A regression approach to speech enhancement based on deep neural networks." IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) 23.1 (2015): 7-19.

[47] "Aachen Impulse Response Database". Internet: https://www.iks.rwth-aachen.de/en/research/tools-downloads/databases/aachen-impulse-response-database/ [2 October 2018]

Curriculum Vitae

Name:	Krishnan Parameswaran
Post-secondary Education and Degrees:	SASTRA University Thanjavur, Tamil Nadu, India 2006-2010 B.Tech
	The University of Western Ontario London, Ontario, Canada 2017-2018 MESc
Honours and Awards:	Ontario Research Fund scholarship 2017-2018
	Outstanding presentation award in Graduate Symposium 2018
Related Work Experience	Teaching Assistant The University of Western Ontario 2017-2018
Publications:	*Machine learning technique to predict the relationship between CG and gearing for banks in Malaysia* (under review with IEEE Intelligent Systems)