Self-Supervised Speech Representation for Speech Recognition

by

Shreya Sanjay Chaturvedi
202015004

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

MASTER OF TECHNOLOGY
in
ELECTRONICS AND COMMUNICATION

with specialization in
Wireless Communication and Embedded Systems
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

A program jointly offered with
C.R.RAO ADVANCED INSTITUTE OF MATHEMATICS, STATISTICS AND COMPUTER SCIENCE

May 2022

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Master of Technology in Electronics and Communications at Dhirubhai Ambani Institute of Information and Communication Technology & C.R.Rao Advanced Institute of Applied Mathematics, Statistics and Computer Science, and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

[illegible]

Shreya Sanjay Chaturvedi

Certificate

This is to certify that the thesis work entitled "Self-Supervised Speech Representation" has been carried out by Shreya Sanjay Chaturvedi for the degree of Master of Technology in Electronics and Communications at *Dhirubhai Ambani Institute of Information and Communication Technology & C.R.Rao Advanced Institute of Applied Mathematics, Statistics and Computer Science* under our supervision.

[illegible]

Prof. (Dr.) Hemant A. Patil
Thesis Supervisor

Dr. Hardik B. Sailor
Thesis Co-Supervisor
Ex-Scientist at SRIB

Acknowledgments

Firstly, I will express my deepest regards to Almighty God. Then, I would like to express sincere gratitude to my supervisors, Prof. (Dr.) Hemant A. Patil, and Dr. Hardik B. Sailor for their constant support and encouragement throughout the research work. I'm grateful of my supervisors, as they have been extremely patient and had believed in me. They have always helped me by reading, analysing, and correcting the manuscripts, whenever needed. Their presence has always inspired me to come up with the best possible results.

I would like to sincerely thank PRISM team at Samsung R&D Institute, Bangalore (SRI-B) for providing me on opportunity to develop my master's thesis on an industry relevant research problem. Especially I would like to sincerely thank Dr. Hardik B. Sailor, Ms. Garima and Ms. Parina Kolhe for having faith in me and giving freedom to explore the research without any hesitation. Their bonds of appreciation have always inspired me with the best working environment in the lab.

I would like to thank my colleagues Mr. Dipesh Kumar Singh, Mr. Preet Amin, and Mr. Ankur T. Patil for being helping hand hands in completing my thesis work. Especially, I'm blessed to have a friend like Mr. Piyush K. Chondingala, who has always been a motivation for work, harmony, and a great guidance. I would like to thank the Speech Research Lab members Ms. Priyanka Gupta, Ms. Aastha Kachhi and Mr. Anand Therattil, they have been a constant support throughout this journey. I will also thank to Mr. Raushan Ranjhan, Mr. Lalit Sheron, Mr. Tejavardhan Reddy, Ms. Ami Pandat and all my fellow friends to support me in all possible ways, they have made this journey fun, inspiring, and simpler.

I would like to present a special thanks to the institution DA-IICT Gandhinagar, for providing me a quality environment and optimum facilities to carry out research work.

At last, I'm extremely grateful of my parents, they have always been progressive and have inspired me to believe in my potential and having a strong attitude of work.

Contents

Abstract **vi**

List of Principal Symbols and Acronyms **viii**

List of Tables **ix**

List of Figures **x**

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Key Research Challenges . . . 4
- 1.3 Contributions From The Thesis . . . 5
- 1.4 Organization of The Thesis . . . 5
- 1.5 Chapter Summary . . . 7

2 Literature Survey **8**
- 2.1 Introduction . . . 8
- 2.2 Literature Survey for Automatic Speech Recognition (ASR) . . . 8
- 2.3 End-to-End ASR . . . 10
- 2.4 Literature Survey on Children ASR . . . 11
- 2.5 Chapter Summary . . . 12

3 Supervised ASR **13**
- 3.1 Introduction . . . 13
- 3.2 ASR . . . 13
- 3.3 Hybrid Deep Neural Network and Hidden Markov Model (DNN-HMM) ASR . . . 15
  - 3.3.1 Acoustic Modelling (AM) . . . 15
  - 3.3.2 Language Modelling (LM) . . . 17
  - 3.3.3 Decoding . . . 19
- 3.4 Chapter Summary . . . 20

4 Data Augmentation for ASR **21**
4.1 Introduction 21
4.2 Proposed Approach 21
4.2.1 Voice Conversion Using CycleGAN 21
4.2.2 Children-to-Children Voice Conversion 23
4.3 Experimental Setup 24
4.3.1 Details of Dataset 24
4.3.2 CycleGAN 24
4.3.3 ASR Training 26
4.4 Experimental Results 27
4.4.1 Data Augmentation Results Using CycleGAN 27
4.5 Chapter Summary 29

5 Self-Supervised ASR **30**
5.1 Introduction 30
5.2 Transformer 30
5.2.1 Encoder 30
5.2.2 Decoder 35
5.3 Wav2vec 2.0 36
5.3.1 Pre-training 37
5.3.2 Fine-tuning 39
5.4 Experiments 39
5.5 Chapter Summary 40

6 NST Learning **41**
6.1 Introduction 41
6.2 Noisy Student Teacher (NST) Learning 41
6.3 Proposed Work 42
6.4 Experimental Setup 44
6.4.1 Pre-processing 44
6.4.2 Teacher Training 44
6.4.3 Student Training 45
6.5 Experimental Results 45
6.5.1 Performance Evaluation 45
6.5.2 Incorporation of Language Model (LM) 47
6.5.3 Analysis of Loss Curves 47
6.5.4 Analysis of Predicted Text 48
6.6 Chapter Summary 49

7 Beamforming for Replay Attacks 50
7.1 Introduction 50
7.2 Signal Modelling and Beamforming 50
7.2.1 Signal Modelling for Microphone Array Signal 50
7.2.2 Delay and Sum (DAS) Beamforming 52
7.2.3 Minimum Variance Distortionless Response (MVDR) 53
7.3 Experimental Setup 55
7.3.1 Feature Sets and Classifier Used 55
7.3.2 Spectrographic Analysis 55
7.4 Experimental Results 56
7.5 Analysis of Latency Period 58
7.6 Chapter Summary 58

8 Summary and Conclusions 60
8.1 Summary of work presented in the thesis 60
8.2 Limitations of the Current Work 61
8.3 Future Research Directions 62

Appendix A Setup 64
A.1 Kaldi Setup 64
A.2 Espnet Setup 65
A.3 Fairseq Setup 65
A.4 HuggingFace 65

Appendix B Data Preparation 66
B.1 Data File Preparation 66
B.2 Flac to Wav File Conversion 66

Appendix C Voice Activity Detection (VAD) 67

Appendix D Baseline for Pretraining + Finetuning using Fairseq 70

References 72

Abstract

Voice Assistants (VAs) are nowadays an integral part of human's life. The low resource applications of VAs, such as regional languages, children speech, medical conversation, etc are the key challenges faced during development of these VAs. On a broader perspective, VAs consist of three parts, namely, Automatic Speech Recognition (ASR), Natural Language Processing (NLP), and Text-to-Speech (TTS) model. This thesis is focused on one part of them, i.e., ASR. In particular, optimization of low resource ASR is targeted with the application of children's speech. Initially, a data augmentation technique was proposed to improve the performance of isolated hybrid DNN-HMM ASR for children's speech. Hence, we have used CycleGAN-based augmentation technique, where children-to-children voice conversion is performed. Here, for conversion of characteristics, the speech signals were categorized into two classes based on the fundamental frequency threshold of speech. In this work, a detailed experimental analysis of various augmentation, such as SpecAugment, speed perturbation, and volume perturbation are done w.r.t. to ASR.

Further, to optimize low resource ASR, the self-supervised learning, i.e., wav2vec 2.0 have been explored. It is a semi-supervised approach, where pre-training is performed with unlabelled data and then fine-tuned with labelled data. In addition, the fusion of Noisy Student Teacher (NST) learning is done with self-supervised learning techniques. The key achievement of this work was efficient use of unlabelled data and even though the process involves iterative training, redundant training was negligible. The filtering of pseudo labelled data was done before utilizing it for fine-tuning. After Acoustic Model (AM) decoding, the Language Model (LM) was also used to optimize the performance.

Additional work was also done in the direction of replay Spoofed Speech Detection (SSD). In this work, the significance of Delay and Sum (DAS) beamformer was investigated over State-of-the-Art (SoTA) Minimum Variance Distortionless Response (MVDR) beamforming technique for replay SSD.

Keywords: Automatic Speech Recognition, Data Augmentation, Self Supervised Learning, Noisy Student Teacher Learning, Replay Spoof Speech Detection.

List of Principal Symbols and Acronyms

AM	Acoustic Model
ASR	Automatic Speech Recognition
BW	Beam Width
CNN	Convolutional Neural Network
CQCC	COnstant-Q Cepstral Coeffecient
CTC	Connectioniest Temporal Classification
CycleGAN	Cycle Consistent Generative Adversarial Network
d$_k$	dimension of query and key vector
d$_v$	dimension of value vector
d$_{model}$	dimension of embedding of each tokens
DNN	Deep Neural Network
DTFT	Discrete Time Fourier Transform
EER	Equal Error Rate
FFNN	FeedForward Neural network
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
LFCC	Linear Frequency Cepstral Coeffecient
LM	Language Model
LTI	Linear Time-Invariant
LW	Language Weight

MFCC	Mel-Frequency Cepstral Coeffecient
NLP	Natural Language Processing
NST	Noisy Student Teacher
RNN	Recurrent Neural Network
SMCC	Spectral Magnitude Cepstral Coeffecient
SoTA	State-of-The-Art
SSD	Spoofed Speech Detection
SSL	Self-Supervised Learning
TDNN	Time Delay Neural Network
TTS	Text-To-Speech
VAs	Voice Assistants
VC	Voice Conversion
w.r.t.	with respect to
WER	Word Error Rate
WIP	Word Insertion Penalty

List of Tables

2.1 Selected chronological progress of self-supervised learning techniques. 9

4.1 WER (%) comparison for various data augmentation techniques. After [71]. 25
4.2 WER (%) comparison for various configurations of DNN. After [71]. 26
4.3 WER (%) comparison for proposed data augmentation. After [71]. . 28
4.4 WER (%) comparison with combination of proposed data augmentation. Here, $\oplus$ represents system combination operator. After [71]. 28

6.1 Tokenization of raw non-speech labels. After [15]. 45
6.2 Experimental results without LM use in pseudo labels. After [15]. . 46
6.3 Experimental results with LM corrected pseudo labels. After [15]. . 46
6.4 Ablation study of predicted transcription without LM. After [15]. . 48

7.1 Results (in % EER) on ReMASC and its DAS *vs.* MVDR beamformed versions using various feature sets. After [16]. 57

List of Figures

1.1 Functional flow diagram of voice assistants. 2
1.2 Self-supervised ASR *vs.* Supervised ASR 3
1.3 Flowchart of the thesis. 6

3.1 Architecture of conventional ASR system. After [4]. 14
3.2 Illustration of DNN-HMM. After [36]. 16

4.1 Functional block diagram of proposed CycleGAN-based data augmentation for children ASR. After [71]. 23
4.2 Speed Perturbation Example. (a) original speech and (b) Speed perturbed speech. After [31] 27
4.3 Example of SpecAugmentation 27

5.1 Model architecture of the transformer. After [80]. 31
5.2 Encoder architecture of the transformer model. After [80]. 31
5.3 Feedforward neural network. After [74]. 33
5.4 Decoder model architecture. After [80]. 35
5.5 Flowchart of *wav2vec 2.0* architecture. After [7]. 37
5.6 *Wav2vec 2.0* architecture. After [7]. 38

6.1 Functional block diagram of NST learning. After [84]. 42
6.2 Proposed NST approach in tandem with self-supervised learning. After [15]. 43
6.3 Loss and WER curve of Teacher and Student models. Where, (a) is for Teacher, (b) is for Student 1, (c) is for Student 2, while Train loss (Panel I), Dev loss (Panel II), and Dev WER (Panel III) are shown w.r.t. epochs. After [15]. 47

7.1 Functional block diagram of DAS beamformer. Different alignment of signals after the microphones (k=1 to N) indicates difference in time of arrivals. After [48]. 53

7.2 Spectrographic analysis of the genuine (Panel I) *vs.* spoofed (Panel II) speech. Speech signal from (a) ReMASC, and its (b) DAS *vs.* (c) MVDR beamformed versions. After [16]. 56
7.3 DET curves for ReMASC and its beamformed versions: (a) development set, and (b) evaluation set. After [16]. 57
7.4 % EER *vs.* latency period using CQCC feature set for development set of ReMASC and its beamformed versions (i.e., DAS and MVDR). After [16]. 58

CHAPTER 1

Introduction

1.1 Motivation

Recently, the most popular technological developments are Voice Assistants (VAs) and conversational user edges. The VAs are rapidly advancing in various security and household services, such as banking, healthcare, personal usage, etc. Advanced Artificial Intelligence (AI) and machine learning (ML) techniques are used to create VA systems and apps. Users are interacting with digital assistants, which implies that the AI is installing more advanced algorithms to learn from the available data. Hence, the goal is to develop the technology, which can better forecast the user's demands. To that effect, the VAs are designed such that, it can include cognitive technologies that enable digital support for increased comprehension and execution of multistep requests. Google Now, Microsoft's Cortana, Amazon's Alexa, Apple's Siri, and Google Assistance are the most popular VAs. In recent years, the context-based understanding has been a breakthrough in VAs technology. Hence, it is now becoming a crucial component in people's lives, and they want individualized experiences when dealing with technologies like this. Voice technology, in particular, relies heavily on personalization along with security. Breaking down the elements of a VAs, it is consisting of following task

- Automatic Speech Recognition (ASR)
- Natural Language Processing (NLP)
- Desired application logic
- Text-To-Speech (TTS)

In this thesis work, few elements of VAs, such as ASR and security measures have been focused. The goal of ASR, has primarily been to reduce errors while decoding voice inputs and hence, this is what caused systems like Siri, Alexa, and

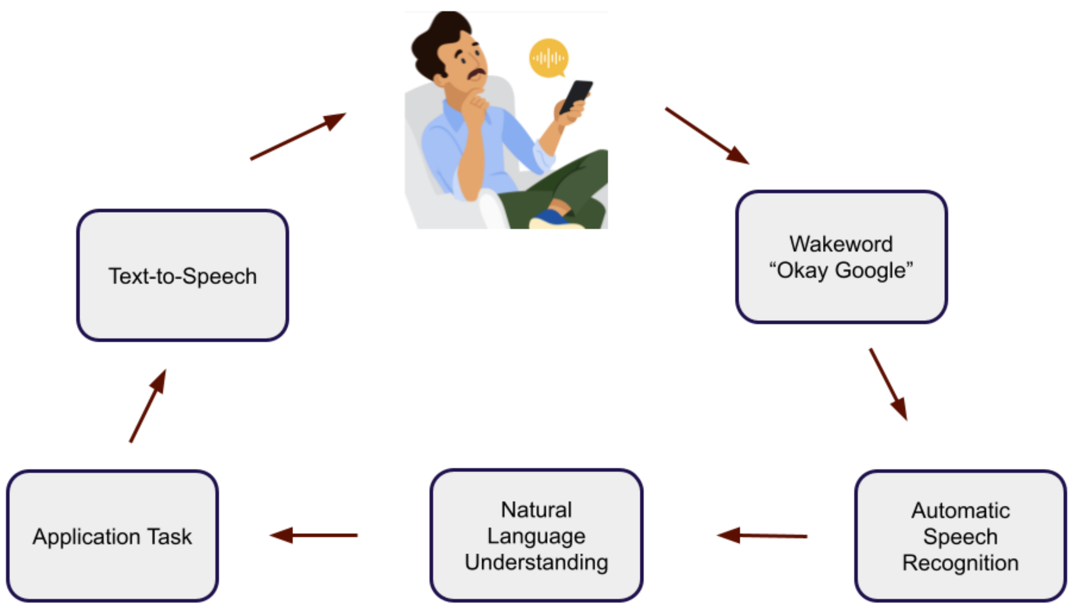
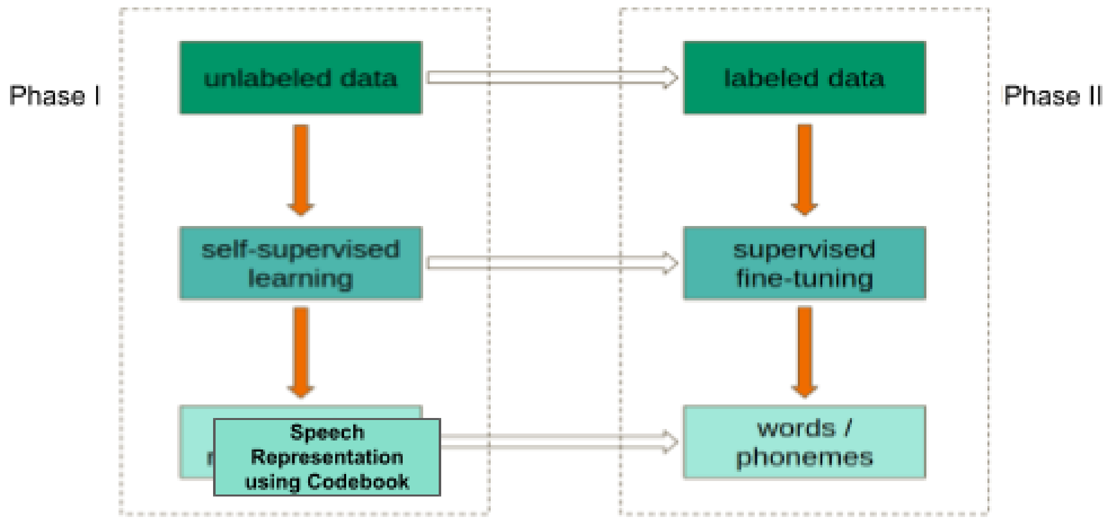


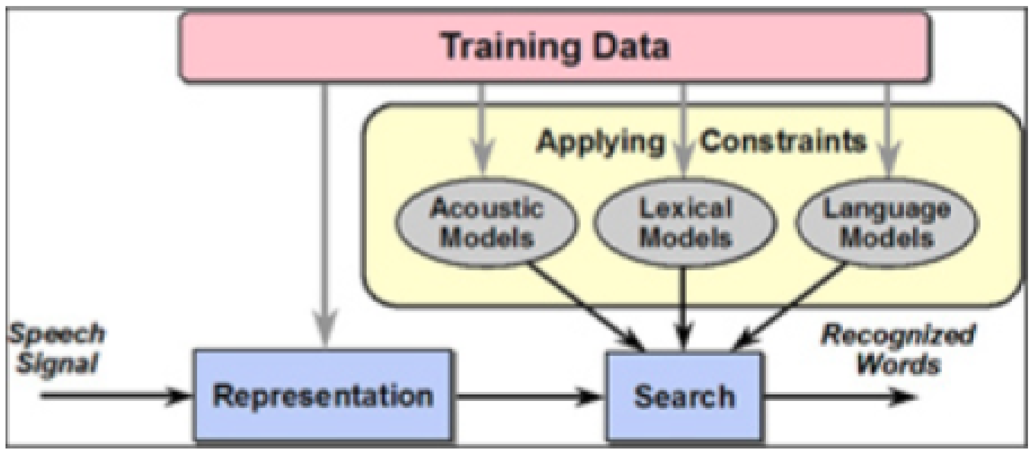
Figure 1.1: Functional flow diagram of voice assistants.

Google Assistant to become so mainstream and commercially successful speech recognition has made its way into our daily lifestyle, due to availability of these VAs. The ASR ought to be facilitated in more languages and for a wider range of activities. Since ASR requires a huge amount of data to operate effectively, and some of it has yet to be captured for certain languages and topics. VAs are typically thought of as independent speakers to which users speak to perform simple activities, such as searching the Internet. VAs are popular because of their ease of use and simplicity, however it's important to note that they're connected to the rest of your digital environment. The fact that these devices have access to a multitude of accounts and passwords makes them difficult to secure in terms of cybersecurity. With so many diverse situations in which voice assistants are used, it's critical to remain up to speed on their numerous vulnerabilities and develop a thorough plan for their use in a remote office or shared workspace.

The self-supervised learning is a robust solution for low resource ASR. The Fig. 1.2 shows the block diagram of SSL-based ASR and supervised ASR. Where SSL ASR is a two stage process, i.e., first pre-training with unlabelled data and then fine-tuning with labelled data. However, Supervised ASR only performs learning with labelled data and also there is a shallow fusion of all three models namely, acoustic model, lexicon model, and language model.



(a) Self-Supervised ASR



(b) Supervised ASR

Figure 1.2: Self-supervised ASR *vs.* Supervised ASR

1.2 Key Research Challenges

ASR is a rapidly growing field, where ASR systems have reached to some reliable system deployment. Supervised ASR gives reliable performance, however, requires extensive information pertaining to the transcribed data. For example, combination of Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM), and combination of Deep Neural Network (DNN) and HMM-based ASR models requires transcription, pronunciation dictionary, and language model (LM), which needs in-depth knowledge and manual efforts to create high quality pronunciation dictionary and LM for all the languages [5]. This constraint was further overcome with the discovery of attention mechanism [8] and Connectionist Temporal Classification (CTC) loss [35, 83], which resulted into an end-to-end (E2E) ASR system. The E2E approach doesn't require pronunciation dictionary and LM, although LM can be added to improve the results. Whereas, E2E approach requires an extensive amount of high quality transcribed data, which is unfeasible for all possible circumstances, such as different languages or age group of speakers. Thus, researchers aims to create a low resource ASR system. To that effect, self-supervised learning (SSL) were introduced, such as the state-of-the-art *wav2vec 2.0* introduced by Facebook AI [7]. In this SSL technique, a significant amount of training is executed with unlabelled audio data and then the model is further fine-tuned with a small amount of transcribed data.

A significant amount of work is reported in the literature for ASR of adult speech, more so for English language. Whereas for children speech, ASR system doesn't perform equally well [60]. There is a significant difference between children and adult speech. Adult speech consist of dialect, tempo, environment variations, whereas children speech is naive. As children are intellectually and physically growing, they have a lot of variability in understanding and expression of speech. Acoustic variability of children's speech includes shift in spectral content and formant frequencies (i.e., higher formants compared to the adult speech), primarily due to lesser size and length of the vocal tract system [40, 69]. All these factors decreases the performance of children ASR. Whereas, children are also major users of assistive speech technology. With the development of virtual learning and interactive courses for children, it has become need of the era to make devices intelligible to children speech [47].

Furthermore, the VAs are highly vulnerable to various spoofing attacks, such as hidden command, self-triggered, Voice Conversion (VC), Speech Synthesis (SS), and replay attacks [86]. In particular, due to the availability of low cost and high quality microphones and playback devices, replay attacks are easy to execute but hard to detect.

1.3 Contributions From The Thesis

- The main contribution of the thesis is for development of a low resource ASR system. Initially, considering SoTA ASR supervised technique, i.e., DNN-HMM ASR system . The performance of supervised ASR was improved by applying CycleGAN-based data augmentation. A detail study of the effect of several augmentation methods is also presented in this part of the thesis.
- Furthermore, working on the challenge of building an ASR for children speech a self-supervised technique, i.e., wav2vec2.0 was used. The performance of this baseline was further improved using unlabelled data and fusing the intuition of NST learning.
- Additionally, some work on developing counter-measures to prevent VA's from replay attacks. In this work, the significance of DAS beamformer was discovered over MVDR for application of replay spoof speech detection.
- Along with results, a detailed discussion on implementation and setup of various frameworks have been explained.

1.4 Organization of The Thesis

Organization of the thesis work is pictorially represented in Fig. 1.3.

Chapter 2

This chapter discusses the detailed background and literature survey of ASR. In particular, various approaches of ASR, along with the augmentation techniques, are discussed. Furthermore, the difference between supervised and self-supervised approach is shown. Thus, pros and cons of all approaches are discussed.

Chapter 3

This chapter explains the technical background of supervised ASR in-depth. In particular, various approaches adopted for building ASR, such as hybrid DNN-HMM, Connectionist Temporal Classification (CTC) , and encoder-decoder models are included along with the pros and cons of each approach.

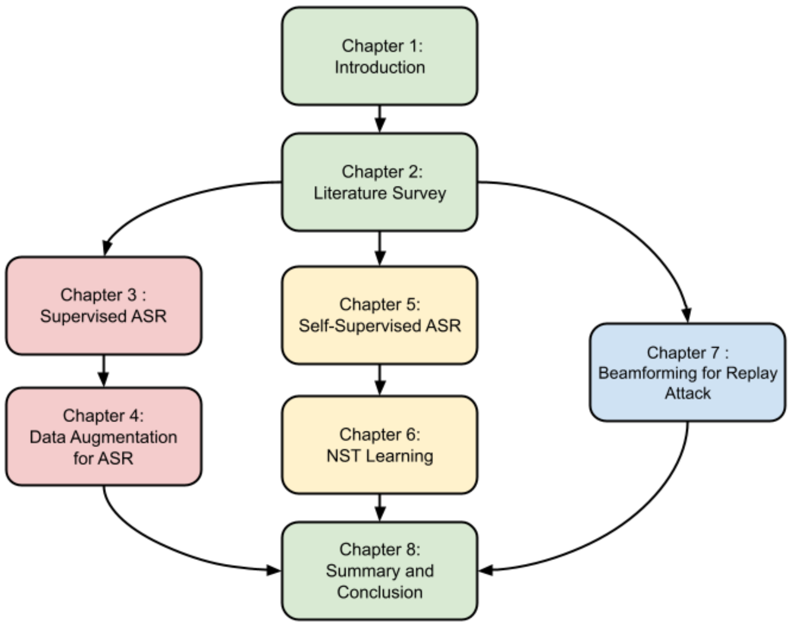


Figure 1.3: Flowchart of the thesis.

Chapter 4

The discussion of various data augmentation techniques is done in this chapter. Later, a CycleGAN-based data augmentation was introduced to improve performance of hybrid DNN-HMM-based ASR system. Furthermore, the experimental results of this work are discussed.

Chapter 5

This chapter includes a detailed study of *wav2vec 2.0*. Its key element, such as Attention model, Transformer model, and its working is also discussed in detail. This chapter will give an overall intuition of how a self-supervised deep learning model works.

Chapter 6

In this chapter, we have discussed the key work done to improve the performance of existing SSL ASR technique. In particular, the concept of NST is introduced, and it's fusion with the SSL architecture is proposed as key work to improve the SSL speech representation.

Chapter 7

In this, an additional work is done for w.r.t. security analysis and voice biometric safety of VA's. Here, the work has targeted to develop countermeasures for Replay Spoofed Speech Detection (SSD). Here, various beamforming techniques are explored to optimize the development of countermeasures.

Chapter 8

Finally, this chapter gives overall summary and conclusions drawn from the work presented in this thesis. The limitations of present thesis work and potential future research directions are also discussed in this chapter.

1.5 Chapter Summary

In this chapter, a brief introduction about VA's is given, where discussion of its utility and challenges was done. Further, key motivation and challenges for ASR development are discussed. Additionally, various approaches have been discussed for implementation of ASR. While some discussion was done for approaches to resolve the limitation of existing baseline. In the following chapter, a brief literature search that is, relevant to this chapter will be discussed.

CHAPTER 2
Literature Survey

2.1 Introduction

This chapter is presents a brief literature survey on ASR. In particular, state-of-the-Art (SoTA) supervised hybrid DNN-HMM is discussed. Further, a brief survey of various unsupervised and self-supervised approaches are done. The comparison between supervised and self-supervised technique is presented, along with a final survey on the work done till now for children ASR is presented. Such literature search helps to position this thesis work into the history of the research problem.

2.2 Literature Survey for Automatic Speech Recognition (ASR)

The first successful conventional model of ASR was GMM-HMM ASR [9], which was later on modified with hybrid model of DNN-HMM, where the generative acoustic model is upgraded with discriminative acoustic hybrid-DNN model [9]. This modified hybrid DNN-HMM model is discussed in detail in Chapter 3. A hybrid DNN-HMM model requires high quality resources such as phonetic dictionary, language model, and speech transcription. It requires huge amount of data, which makes it difficult to deploy in all possible languages. This resulted in discovery of End-to-End ASR, here resources like LM and lexicon dictionary were not required but requires huge amount of data to train the model, it is discussed in detail in next section. Due to limitation of availability of large amount of data, self-supervised ASR techniques were introduced. In supervised ASR techniques a complete one-to-one, from feature to phoneme mapping is done, after the force alignment using HMM. While in self-supervised ASR, the entire training is divided in two major parts, namely, pre-training and fine-tuning. In SSL, the pre-training part has no labels for training and hence an approximate representation is created in the form of codebook via. vector quantization algorithms. Further, in fine-tuning those learning are labelled through training on labelled data, it is in detail discussed in Chapter 5. A literature survey of various SSL techniques is presented in Table (2.1).

Paper Name	**Model**	**Intuition**	**Benchmark dev-clean / dev-other / test-clean / test-other WER (%) [for libri]**	**DataSet / Comment**
Hubert: How much can a bad teacher benefit asr pre-training? [30]	HuBERT	* BERT like masked prediction is used for force learning of both acoustic and language models * Multiple Bad Teacher in pre- training	3.9 / 9.0 / 4.3 / 9.4	Pretrained : 960h Libri Fine tuned :- 10h of Libri Light
The TAL system for the INTERSPEECH2021 Shared Task on Automatic Speech Recognition for Non-Native Children Speech [89]	Wav2vec 2.0	* 11 Fold Data augmentation using 7 techniques, listed as {Speed perturbation, Volume perturbation, Reverberation simulation, various noise augmentation, Pitch augmentation } * Language Model { 4 gram + Transformer }	WER = 23.5%	TLT Data-set English & German Non- native speech
Representation Learning with Contrastive Predictive Coding [78]	Speech representation based in predictive coding	* It extracts useful information of high dimensional data, creating a low dimensional latent space representation		It discards low-level information
wav2vec 2.0: A frame- -work for self-supervised learning of speech representations [7]	Wav2vec 2.0	* Speech input is masked in latent space representation * This latent representation is fed to a Transformer network to build contextualized representation and the model is trained via contrastive task where the true latent is to be distinguished from distractors.	(10h base model) 3.8 / 9.1 / 4.3 / 9.5 (100h large model) 1.9 / 4.0 / 2.0 / 4.0	Libri Speech
Unsupervised Speech [6] Recognition	wav2vec-Unsupervised	* Phonetic segmentation using wav2vec 2.0 representation, k-mean clustering and generator * 1 hot vector representation of unlabelled text. * GAN Training of real or fake	Timit PER = 11.3 Eng Libri Speech on test-other WER = 5.6	Libri Speech
W2v-BERT: Combining Contrastive Learning and Masked Language Modelling for Self-Supervised Speech Pre-Training [17]	Wav2vec+BERT	* Combines these to optimize in an end to end fashion by solving this self-supervised task ->Contrastive Learning ->Masked Language Modeling (MLM) * Relies of iterative re-clustering and re-training process * Conformer blocks for context representation	Without LM Pre- trained only XL: 1.5 / 2.9 / 1.5 / 2.9 XXL:1.5 / 2.7 / 1.5 / 2.8 Pre train+Self training XL: 1.3 / 2.6 / 1.4 / 2.7 XXL: 1.4 / 2.4 / 1.4 / 2.5	Libri-Light 60k corpus
Pushing the limits of semi-supervised learning for automatic speech recognition [92]	Wav2vec + conformer model	* Noisy student training using Spec-Augment	**Conformer XL with no LM Fusion** **1.5 / 3.1 / 1.6 / 3.0**	Libri Speech

Table 2.1: Selected chronological progress of self-supervised learning techniques.

Researchers aim to create a learning mechanism which works on the principle of learning of an infant, thus aims is to create an unsupervised approach of learning a language. While, even children learn gradually with the help of the activities and people around them, on other side infants also take years to grow and learn. Thus, with this intuition, researchers discovered self-supervised learning techniques. The principal approach of self-supervised learning is to perform the major part of learning in pre-training, where supervision of labelled data is not required, in other words for pre-training labels are not required. This pre-training creates an approximate quantized learning of target data. Several such approaches are mentioned in Table (2.1). In this table, several approaches with different application are mentioned. where *wav2vec2.0* is recently the most robust approach in self-supervised approach. While some BERT-based techniques are also mentioned in the Table (2.1), in which masking is efficiently used for prediction learning. In the end of the table Conformer architecture is used, which improves the *wav2vec 2.0* performance, with introduction of CNN in the architecture of transformer.

2.3 End-to-End ASR

There have been several developments in End-to-End ASR overcoming the limitation of hybrid DNN-HMM ASR, following are few of them :

- Requirement of phonetic alignment and state dependent HMM structure for acoustic model. In other words, estimation of HMM and GMMs are required before DNN training.
- It needs lexicon model which is handcrafted pronunciation dictionary that may contain human error.
- Hybrid DNN-HMM ASR uses conditional independence assumption (HMM alignment) between speech embeddings and output transcription. It is not valid in real world, as phonetic context is strongly influenced by pronunciation of a phoneme.
- Integration of several components at decoding makes the whole process complex.

To resolve all above-mentioned drawbacks, End-to-End ASR tries to reduce the number of components into a single network using DNN. In addition, it tries to estimate the distribution directly from the single network, $p(Y|X)$ [27], This way network learns directly the mapping between acoustic signals and transcription without explicitly introducing additional components in-between as introduced in DNN-HMM ASR model. Joint training is used to train the entire network. It uses a single objective function to train the network, which aids in convergence to global optimum points. The use of soft alignments in end-to-end networks improves the system's efficiency by mapping every frame of speech to every possible output with some probability [82]. A typical end-to-end ASR system consists of two major components: an encoder and a decoder. The encoder tries to figure out how to transfer input feature space to latent space representation. The decoder, on the other hand, seeks to learn the mapping between the encoder's latent space representation and the output tokens (transcriptions). For the end-to-end ASR model, two approaches are commonly used: (1) Connectionist Temporal Classification (CTC) and (2) Attention network-based encoder-decoder system. In the next sub-sections, both designs are briefly explained.

2.4 Literature Survey on Children ASR

Various types of automation technologies have brought increased demand on effective interaction between humans and machines. Recently, voice assistants or intelligent personal assistants (IPAs) are widely used for communication, education, consoles, etc. This has become possible due to advancements in automatic speech recognition (ASR) systems. The users of ASR includes both children and adults. Children use ASR for various purposes, such as remote learning, gaming, entertainment, etc. However, the performance of ASR for children's speech was found to be worse than that for adults [60]. Due to increasing demand of speech-based interfaces for children, it is important to address this challenge for children's ASR.

The differences between the acoustic patterns of adult and children speech was studied in [40, 69]. Recent study suggest that, there are two main factors responsible for the variability in the children's speech [69]. First, temporal and spectral variability present in the children speech are due to physiological and developmental variations. Second, variability in pronunciation for children speech is due to lack of understanding about language and linguistic information, resulting into several disfluencies, such as pauses, fillers, repetitions, corrections, etc, [40]. Acoustic variability includes shift in spectral content, therefore higher formant frequencies (F_1 to F_4), and their transition. High variations in spectral content within a subject were found in [39], however, due to developmental changes, a large inter-speaker variability for children was also noted in [23]. Li *et al.* reported that the performance of ASR degraded significantly, when the bandwidth of speech is reduced from 4 kHz to 1.5 kHz [41]. One of the reasons for this was higher (more than 60%) formants (F_1, F_2, and F_3) for children due to much shorter length of vocal tract system than that for the adults.

In the past, some research efforts have been carried out to develop acoustic model (AM) for children's ASR. Adapting acoustic models with Maximum Likelihood Linear Regression (MLLR) and Maximum A-Posteriori (MAP) gained significant attention for this problem [70]. Shivakumar *et al.* investigated different transfer learning adaptation techniques on large vocabulary continuous speech recognition (LVCSR) for children [69]. Effectiveness of data augmentation by artificially augmenting noise was investigated in [43]. Augmentation of adult speech data with children has shown improvement in performance of children's ASR [64]. Multitask learning, transfer learning as well as self-supervised learning framework employed for multilingual data adaptation showed its effectiveness for children's ASR [46, 90]. Generative adversarial networks (GANs) were also used in the literature for data augmentation [68]. In particular, CycleGAN-based data augmentation showed better improvement over the other techniques [68].

2.5 Chapter Summary

In this chapter, a review of evolution of ASR since the beginning was presented. It initially mainly discussed the SoTA supervised ASR techniques such as GMM-HMM and hybrid DNN-HMM. Later their limitation were addressed, in which major limitation is requirement of resources. Thus, the concept of low resource ASR was introduced, which created self-supervised ASR technique. Thus, in the end, several SSL techniques for ASR were analysed. On that effect, the next chapter is the background study of Supervised ASR.

CHAPTER 3
Supervised ASR

3.1 Introduction

The most robust technique for deployment of supervised ASR is hybrid DNN-HMM model. Thus, in this chapter, a brief discussion of hybrid DNN-HMM ASR is done. Initially a brief discussion on application and end product of model, i.e., posterior probability introduction was given. Then, the acoustic modelling and language modelling process will be discussed. For acoustic modelling, first feature extraction is performed to have a low dimensional speech representation. Then this low dimensional speech representation is mapped with the output transcription, before that force alignment of transcription w.r.t. audio using HMM is performed. This results in the posterior probability distribution, which is actually used in decoding.

3.2 ASR

A speech signal is converted into a sequence of words, characters, or phonemes by an ASR system, and then the machine understands using NLP. To execute various tasks, most voice assistants, such as Amazon Alexa, Apple Siri, Google Voice Assistant, and Cortana, employ ASR in conjunction with Text-to-Speech (TTS). Fig. 3.1 depicts the components of a typical ASR system. Using a feature extraction block, the input voice stream is first converted into a sequence of acoustic feature vectors. Let's call the feature vector sequence, $X = \{x_1, x_2, ..., x_T\}$, where T is the number of frames. Following that, the decoder estimates the best word sequence, $\widehat{W}$, using the ASR fundamental equation:

$$\widehat{W} = \underset{w}{argmax}\ P(W/X). \tag{3.1}$$

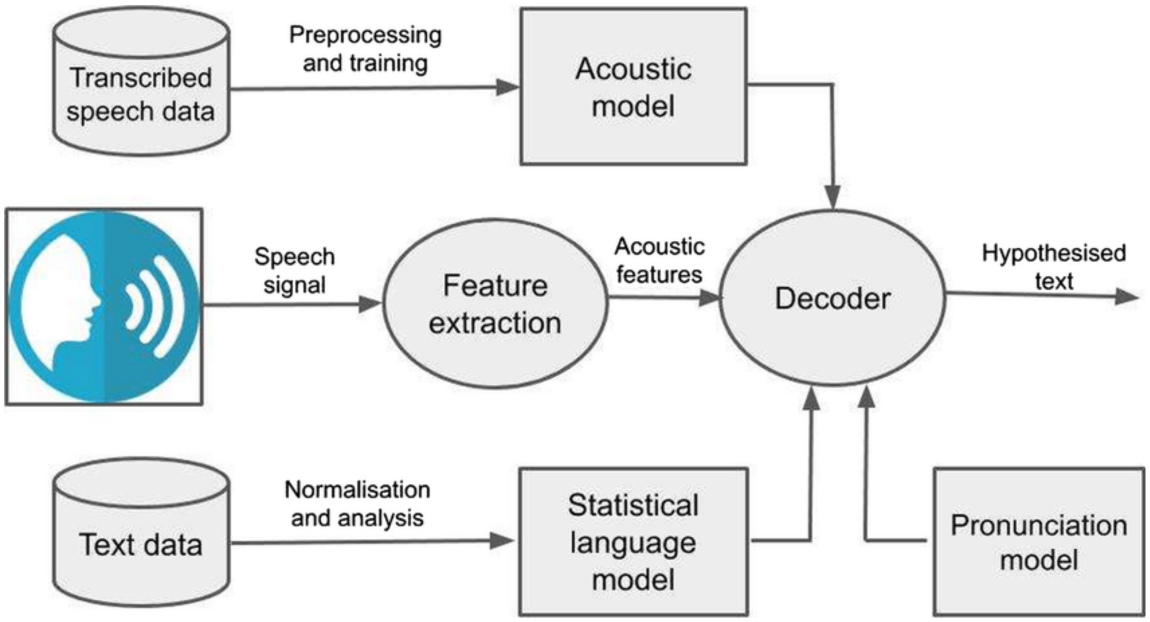


Figure 3.1: Architecture of conventional ASR system. After [4].

Several generative models, including HMM is used for sequence modelling of speech, which is variable in length [22]. The above eq. 3.2 can be rewritten as follows using Bayesian decision theory [21]:

$$\widehat{W} = \underset{w}{argmax} \frac{P(W)P(X/W)}{P(X)}. \tag{3.2}$$

Due to the fixed acoustic feature vector X, the equation can be approximated as:

$$\widehat{W} = \underset{w}{argmax}\ P(W)P(X/W). \tag{3.3}$$

The probability of occurrence of an acoustic feature sequence X given that the word detected is W is defined as $P(X|W)$, and is estimated using an Acoustic Model (AM) . Furthermore, $P(W)$ is the prior probability calculated using a Language Model (LM) for a specific word sequence W, and it is unaffected by the acoustic features seen. The ASR task can be performed in two ways: hybrid DNN HMM [12], in which the AM and LM are trained separately; whereas, the End-to-End approach tries to learn a direct mapping between acoustic feature sequence X, and word sequence W. The sections that follow briefly outline the various components used in these approaches.

3.3 Hybrid Deep Neural Network and Hidden Markov Model (DNN-HMM) ASR

3.3.1 Acoustic Modelling (AM)

Acoustic models are one of the most important components of any ASR system, and they must be resistant to changes in the environment, the speaker, and the context. For each unit of speech, feature vectors are estimated in acoustic models. The choice of these basic speech sound units is critical before training any acoustic model. For example, in a database with a vast vocabulary, words cannot be used as the fundamental unit because this would need a huge number of training utterances and thus, modelling acoustic data fluctuations would be difficult. Before deciding on basic speech sound units, two aspects should be considered: 1) It should be generalizable, allowing any new term to be created from these basic speech sound units, and 2) it should be correct. Phonemes are commonly used as basic speech sound units in hybrid DNN-HMM ASR tasks because they are generalizable. Words, on the other hand, can be utilized as the basic sound unit of a little vocabulary challenge. Phonemes are perceptually different basic units of sound that can model articulatory gesture in any language, regardless of the context. For the big vocabulary ASR job, tri-phones are used as the basic speech unit. Since, speech recognition is context-dependent, the pronunciation of the current phoneme is influenced by the past and future phonemes. As a result, tri-phones are used in the basic speech sound unit, to introduce left and the right context.

The estimation of precise representation of input feature vectors for each basic sound unit is the second stage of acoustic modelling. The acoustic model tries to determine $P(X|W)$, the probability of an acoustic feature X given a phoneme or word W. In the case of an isolated word ASR with an N-word vocabulary, each word should have its own acoustic model. Here, $P(X|W)$ will be the same as $P(X|\beta_i)$ in that case, where B; denotes the i^{th} word acoustic model. To estimate the probability for the acoustic model, a statistical approach is used. Hidden Markov Models (HMM) are often regarded as one of the most effective statistical models [67]. The likelihood of a transition from state i to state j at a time frame t is evaluated in an HMM and is known as transition probability a_{ij}. The observation probability, $b_j(x_t)$, is also computed from the observation vector x_t, for every state j. As shown in Fig. 3.2, an HMM usually comprises two states with no emission probability, assigned to them, one at the beginning of the chain and the other at the conclusion. The likelihood of emission in between states is calculated using

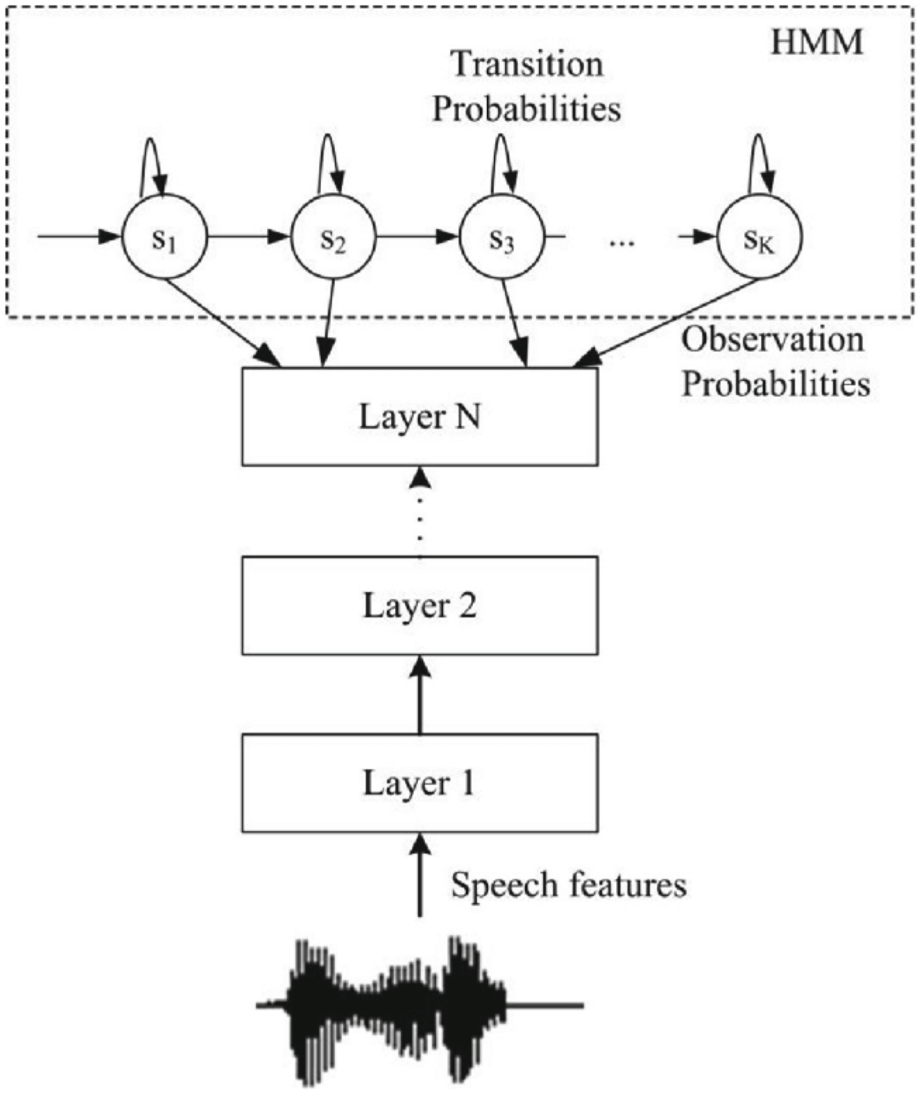


Figure 3.2: Illustration of DNN-HMM. After [36].

a Deep Neural Network (DNN). A DNN that is directly related to the emission probability can be used to determine the posterior probability, $P(\theta|X)$. It denotes the likelihood of a particular state occurring given the observed feature vector. To accomplish this objective, each DNN output layer will correspond to a specific HMM state, and the values collected from DNN will be translated to the emission probability using Bayes' rule. Specifically,

$$P(\theta/X) = \frac{P(X/\theta)P(\theta)}{P(X)}, \tag{3.4}$$

where $P(\theta)$ is a class prior in this case. A forced-alignment in the GMM-HMM training generalizes the relative frequency of each class as determined by the class labels. The eq. 3.4 can be rewritten as:

$$\frac{P(\theta/X)}{P(\theta)} = \frac{P(X/\theta)}{P(X)}, \tag{3.5}$$

The HMM's emission probability can be computed using the term on the right-hand side of the above eq. 3.5. The denominator, P(X), will be used as a scaling factor in this case. Here, Fig. 3.2 shows an illustration of the N-layer DNN used in hybrid DNN-HMM [36].

3.3.2 Language Modelling (LM)

For any language, the language model presents a statistical representation of grammar. There are some guidelines given in [66], which specify the allowed combinations of basic language units. The language model, which uses the word as its basic unit, attempts to calculate the probability, $P(W)$, of each possible sequence of words, $W = \{w_1, w_2, ..., w_n\}$ for a speech utterance. Traditional deterministic grammar was used in the early days of language modelling. It generates a probability value of one for permissible proper structure. It also generates a probability value of zero for any syntactically wrong structure. However generally, n-gram language models are used to assess the likelihood of a given sequence of n words being formed. The probability of any word sequence occurring, $P(W)$, can be computed as follows:

$$P(w) = P(w_1, w_2, ..., w_n), \tag{3.6}$$

$$P(W) = P(w_1)P(w_2|w_1)...P(w_n|w_1, w_2, ..., w_{n-1}), \tag{3.7}$$

$$P(W) = \prod_{i=1}^{n} P(w_i|w_1, w_2, ..., w_{i-1}), \tag{3.8}$$

$P(w_i|w_1, w_2, ..., w_{i-1})$ is the likelihood of any word w_i occurring given the sequence of words $w_1, w_2, ..., w_{i-1}$ that has already been created. It signifies that the estimation of the current word is based on previous predicted words. Probability estimation, due to the high number of alternatives in a broad vocabulary language model, $P(w_i|w_1, w_2, ..., w_{i-1})$ is computationally difficult. The aforementioned challenge is solved by employing Markov's N^{th} order assumption, and the resulting language model is known as the N-gram language model [22]. $P(w_i|wi - N + 1, w_{i-N+2}, ..., w_{i-1})$ uses the last $N - 1$ words to estimate the current word probability. If $N = 2$, the resulting language model is known as the bi-gram language model, and it only uses the last word to compute the probability $P(w_i|w_{i-1})$. Similarly, with $N = 3$, the tri-gram language model can be found, which uses the last two preceding words, $P(w_i|w_{i-1}, w_{i-2})$. The probability $P(w_i|w_{i-1}, w_{i-2})$ is calculated for a specific word w_i as:

$$P(w_i|w_{i-1}, w_{i-2}) = \frac{Count(w_{i-2}, w_{i-1}, w_i)}{Count(w_{i-2}, w_{i-1})}. \tag{3.9}$$

The number of times the words $w_{i-2}, w_{i-1},$ and w appear in this sequence is represented by Count(w_{i-2}, w_{i-1}, w_i). Count(w_{i-2}, w_{i-1}) denotes the number of times the words w_{i-2}, w_{i-1} appear in this specific sequence. The transcriptions used to train the ASR are used to compute the language model.

DNNs have improved in a variety of tasks, including language modelling, as computer resources have increased. Recurrent Neural Networks (RNNs) are commonly used to complete this task due to variable length and consecutive text data [49]. The Long-Short Term Memory (LSTM) network, for example, consistently outperformed the others for language modelling [73]. RNN is a neural network that not only processes its input w at time step t, but also processes its previous hidden state s_{t-1} to construct the current hidden state, s. RNNLM's purpose is to convert the history h into a fixed-dimension real vector, d, i.e.,

$$f(h) \in \mathbb{R}^d, \tag{3.10}$$

where f(.) denotes a mapping function that is dependent on the model used for the task. Assume RNN receives input w from alphabet V and the hidden state s is a d-dimensional real vector, $s \in \mathbb{R}^d$. RNN may now be explained using only two functions. After processing the current word w and the previous state s_{t-1}, the state transition function δ generates a new state, s_t, i.e.,

$$s_t = \delta(w_t, s_{t-1}). \tag{3.11}$$

Then a function $g : s \rightarrow (0,1)^{|V|}$ transform new state s_t with a dimension d to an output distribution over all the words, i.e.,

$$p(w|s_t) = g(s_t). \tag{3.12}$$

The above eq. 3.12 can be rewritten as,

$$p(w_t|w_1, w_2, w_{t-1}) = p(w_t|s_{t-1}). \tag{3.13}$$

Here,

$$\begin{aligned} s_{t-1} &= \delta(w_{t-1}, s_{t-1}), \\ &= \delta(w_{t-1}, \delta(w_{t-2}, s_{t-3}), \\ &= \delta(w_{t-1}, \delta(w_{t-2}, \delta(..., \delta(w_1, s_0). \end{aligned} \tag{3.14}$$

It can be concluded that the calculation of any state s_t, is dependent on all previous histories, beginning with w_1 and ending with w_t. Theoretically, the network can encode limitless history information, however, in practise, because of computational limits, RNNLM is unable to do so. When compared to the vanilla RNNLM, which uses the affine transform for both the functions g and δ, LSTM uses a gated approach that allows the model to learn long-term dependencies.

3.3.3 Decoding

The method of decoding involves guessing the word sequence from the audio data. For the estimate task, this technique employs the acoustic and linguistic models. For this objective, statistical estimation were used. First, the utterance's acoustic feature vectors, $X = x_1, x_2, ...x_t$ are retrieved. The decoder then tries to determine the best words sequence, $\widehat{W} = w_1, w_2, ..., w_n$ for the supplied speech, with the least amount of error. It can be calculated mathematically via maximization of the posterior probability, $P(w|x)$. The entire decoding process can be thought of as a search, in which the decoder looks for possible word sequences and chooses the one with the highest posterior probability. Thus, the new decoding equation is represented by:

$$\widehat{W} = \underset{argmax}{W} P(W)^{LW} WIP^{N(W)} P(X|W). \tag{3.15}$$

Here, LW and WIP are assigned empirically, whereas $N(W)$ represents the total number of words in a particular word sequence, W.

3.4 Chapter Summary

In this chapter, in depth intuition behind the working of ASR system in general as well as hybrid DNN-HMM ASR system was surveyed. Where in the end, at decoding stage for output generation, a shallow fusion of probabilities of both acoustic and language modelling is done. In the following chapter, the performance of hybrid DNN-HMM ASR system is improved using suitable data augmentation techniques.

CHAPTER 4

Data Augmentation for ASR

4.1 Introduction

In this chapter, CycleGAN-based data augmentation technique is investigated without using any additional adult or children speech data for children ASR task. Instead of learning one-to-one mapping for source-target speaker-pairs as in the conventional voice conversion (VC) techniques, speakers are divided into source-target classes and a generalized mapping is learned from one class to another and vice-versa. In particular, CycleGAN-based data augmentation showed better improvement over the other techniques for children ASR task.

4.2 Proposed Approach

GAN frameworks are known for their capability of generating realistic fake outputs via estimation of underlying probability distribution function, a centeral problem in signal processing [37]. The CycleGAN architecture involves the concept of cycle-consistency, which ensures that the reconstructed output features are around the original features [91, 42]. Considering the huge variability of acoustic behaviour of children, we are targeting the diverse behaviour of their F_0. Thus, we explored two-class mapping of children voices using modified CycleGAN architecture, as explained next [26].

4.2.1 Voice Conversion Using CycleGAN

The task of voice conversion based on speaker classes is to convert a generalized class of features $a \in A$ to another generalized class of features $b \in B$. Here, $A, B \subset \mathbb{R}^{D \times T}$ are assumed to be feature spaces for speakers with low and high average F_0 , respectively (details of which are provided in sub-Section 2.2). D is the dimension of the input feature vector, and T is the number of speech frames.

Two generators of the CycleGAN are trained to learn the mapping function of $G_{A\to B} : A \to B$, and $G_{B\to A} : B \to A$. Two discriminators of the CycleGAN, namely, $D_A : \mathbb{R}^{D\times T} \to [0,1]$, and $D_B : \mathbb{R}^{D\times T} \to [0,1]$, are used to discriminate if a feature vector is from their respective class or not. Three different loss functions are used during training in order to learn the above mentioned mapping functions [33].

Adversarial loss: Adversarial loss is used to create adversary between generators and discriminators. For the generator-discriminator-pair, $G_{A\to B}$ and D_B, the adversarial loss is estimated as:

$$\mathcal{L}_{adv}(G_{A\to B}, D_B) = \mathbb{E}_{b\sim P_B(b)}[log D_B(b)] + \mathbb{E}_{a\sim P_A(a)}[log(1 - D_B(G_{A\to B}(a)))], \quad (4.1)$$

where $\mathbb{E}[.]$ is expectation operator.

Cycle-Consistency Loss: To make $G_{A\to B}$ and $G_{B\to A}$ inverse of each other, so that no two inputs are mapped to the same output, cycle-consistency loss is applied, i.e., $G_{B\to A}(G_{A\to B}(a)) \approx a$ [11]. In particular,

$$\begin{aligned}\mathcal{L}_{cyc}(G_{A\to B}, G_{B\to A}) =& \mathbb{E}_{a\sim P_A(a)}[\|G_{B\to A}(G_{A\to B}(a)) - a\|_1] + \\ & \mathbb{E}_{b\sim P_B(b)}[\|G_{A\to B}(G_{B\to A}(b)) - b\|_1],\end{aligned} \quad (4.2)$$

where $\|.\|_1$ is L_1-norm.

Identity-mapping loss: To ensure the integrity of the linguistic content, identity mapping loss is used [88]. Mathematically, identity-mapping loss ensure that any feature belonging to the target class is mapped to itself, i.e., $G_{A\to B}(b) = b$.

$$\mathcal{L}_{id}(G_{A\to B}, G_{B\to A}) = \mathbb{E}_{a\sim P_A(a)}[\|G_{B\to A}(a) - a\|_1] + \mathbb{E}_{b\sim P_B(b)}[\|G_{A\to B}(b) - b\|_1]. \quad (4.3)$$

Discriminators are trained only on adversarial loss. The generator loss function is estimated as a weighted sum (weights given by λ_{adv}, λ_{cyc}, and λ_{id} hyper-parameters) of the three loss functions, which is given by:

$$\mathcal{L}_{gen} = \lambda_{adv}\mathcal{L}_{adv}(G_{A\to B}, D_B) + \lambda_{adv}\mathcal{L}_{adv}(G_{B\to A}, D_A) + \lambda_{cyc}\mathcal{L}_{cyc}(G_{A\to B}, G_{B\to A}) + \lambda_{id}\mathcal{L}_{id}(G_{A\to B}, G_{B\to A}). \tag{4.4}$$

4.2.2 Children-to-Children Voice Conversion

Due to shorter length of vocal tract system and smaller vocal folds (i.e., lesser mass), children have higher formant frequencies (F_1 to F_4) and F_0 than those for the adults [24]. Moreover, children speech contains large variations in F_0, due to the growing age of the children. To avoid scarcity of data per speaker, we propose to use F_0-based threshold to create two sets of speaker classes. An average F_0 of all the utterances present in the training corpus is sorted in an array. Thereafter, the value of median is chosen to be the threshold (F_{th}). The utterances having average F_0 less than this threshold (F_{th}) were put together into one class, say class A, and the utterances having average F_0 more than the threshold were put into another class, say class B. Thereafter, CycleGAN is employed for voice conversion between class A and class B (i.e., two-way conversion). Fig. 4.1 shows the functional block diagram for the proposed data augmentation approach for voice conversion between high F_0 children speakers to low F_0 and vice-versa.

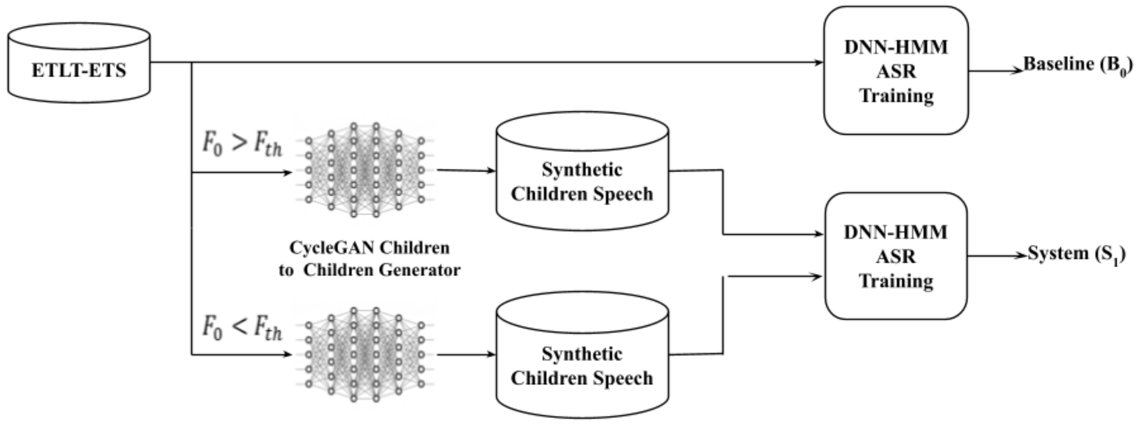

ETLT-ETS
DNN-HMM ASR Training
Baseline (B0)
F0 > Fth
Synthetic Children Speech
CycleGAN Children to Children Generator
DNN-HMM ASR Training
System (S1)
F0 < Fth
Synthetic Children Speech

Figure 4.1: Functional block diagram of proposed CycleGAN-based data augmentation for children ASR. After [71].

4.3 Experimental Setup

4.3.1 Details of Dataset

Experiments are performed on ETS subset of English corpus released during ETLT2021 challenge [28]. The training corpus contains 53.43 hours of speech data from 800 speakers. Each speaker has 4 recordings leading to a total of 3200 audio files. Development (Dev) set contains about 3.3 hours of data belonging to 50 speakers. There are a total of 200 audio files, corresponding to 4 recordings per speaker. The evaluation (Eval) set contains approximately 3.3 hours of audio data from the 50 speakers. Similar to train and Dev sets, there are 4 recordings available per speaker, leading to a total of 200 audio files. These recordings involve both read and spontaneous speech. Considering the challenge of limited data, the proposed experiments only utilize the ETLT2021 dataset and no external speech in the study.

4.3.2 CycleGAN

Input Features

To train CycleGAN, we extracted *24*-dimensional (D) Mel Cepstral Coefficients ($0^{th} + 23$-MCEP coefficients), F_0, and aperiodicity (AP) using WORLD vocoder [50]. MCEP features and F_0 were normalized by estimating their respective means and standard deviations over the entire class of speakers. F_0 was converted using logarithmic Gaussian normalized transformation, while MCEPs were given as an input to the CycleGAN. Speech synthesis was done using WORLD vocoder with converted MCEPs, converted F_0, and original aperiodicity.

Architectural Details

We used a CycleGAN-VC2 architecture based on the description given in [33]. PatchGAN was used for discriminator architecture with a total of three downsampling layers. For generator, *2-1-2*D architecture described in [33] was used, however, instead of using statistical upsampling layer (composed of interpolation and pixel-shuffle layers), we used two-strided transpose-convolution layers, such that network can learn its own upsampling. A total of two downsampling layers, six residual layers, and two upsampling layers were used. Downsampling and upsampling were done using strided *2D* CNNs, while residual layers were composed of 1D CNNs [38]. Features were converted from 1D to 2D, and vice-versa by feature alignment using 1D CNN after reshaping. Gated Linear Units (GELU) were used as activation functions, and instance normalization was used to normalize convolution layers.

Table 4.1: WER (%) comparison for various data augmentation techniques. After [71].

Augmentation Technique	**Dev WER%**
No Augmentation	15.28
SpecAugment (SA)	15.20
Speed Perturbations (SP)	14.19
Volume Perturbation (VP)	15.35
SA + SP	13.67
SA + VP	13.69
VP + SP	13.77
SA + VP + SP	13.31
*Only CycleGAN	16.28
CycleGAN Augmentation	14.63
SA + VP + SP + CycleGAN	**12.11**

*In this experiment, only CycleGAN data was used for training purpose. This proves the synthesized data is intelligible.

Training of CycleGAN

The entire dataset was separated into two classes based on speaker's average F_0. Then, 1590 utterances were taken from both the classes in order to train our network. For the training corpus, variation in average F_0 was observed from 90 Hz to 355 Hz and F_{th} is found to be around 194 Hz. Segments of 256 frames were selected from a randomly selected utterance, which added a second layer of randomness for the training. CycleGAN was trained to convert features from one class to the other, for which it was trained for 5×10^4 iterations with a batch size of 8. To stabilize the training of the GANs, the Least Squares Generative Adversarial Networks (LSGANs) were used [45]. Initial learning rates of 4×10^{-4}, and 2×10^{-4} were used for generators and discriminators, respectively, which were decreased by a factor of two at 3×10^4, and 4×10^4 iterations. Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$ was used to calculate gradients and update the weights of the model. λ_{cyc}, λ_{id}, and λ_{adv} are taken as 10, 5, and 1, respectively. λ_{id} is reduced to 0 after 2×10^4 iterations. To ensure generators does not fall behind in training, discriminators are trained only when their loss value is above the threshold of 0.05, allowing the generator to catch-up on the training because syn-

Table 4.2: WER (%) comparison for various configurations of DNN. After [71].

DNN Configuration	Dev WER%
6 CNN layers + 7 TDNN-F layers	13.27
6 CNN layers + 9 TDNN-F layers	**12.82**
6 CNN layers + 11 TDNN-F layers	13.12

chronization of generator with discriminator is the key challenge in GAN training [26, 25]. After training CycleGAN model, voice conversion is applied for high F_0 speakers to low F_0 and vice-versa, which gives a total of 3180 synthetic audio files.

4.3.3 ASR Training

The hybrid DNN-HMM ASR systems were built in Kaldi toolkit with the training recipe provided by the organizers of the INTERSPEECH ETLT2021 challenge. 40-D high resolution MFCC features are used for training the AM [62]. In addition, Cepstral Mean Variance Normalization (CMVN) is applied to reduce channel noise effects [72]. The AM is a chain model trained with lattice-free maximum mutual information (LF-MMI), and consists of Convolutional Neural Network (CNN) layers followed by factorized TDNN layers (TDNN-F) of sizes 1024 [61], [63]. 100-D *i*-vectors are also fed along with MFCC features to provide speaker adaptation during training. A *n*-gram language model (LM) is built using the training data transcriptions. Order of *n* is explored for the proposed ASR model. Phonetic pronunciations are used from the CMU lexicon, with a G2P system trained on Phonetisaurus [52]. Three-way speed perturbation (i.e., for the speed perturbation factor of 0.9, 1.0, and 1.1), volume perturbation, and SpecAugment are employed for data augmentation [57].

Speed perturbation is the augmentation method where the signal is resampled as per the scale of speed to be perturbed, an example is shown in Fig. 4.2.

Volume Perturbation is an augmentation technique where the signal's amplitude is just scaled with the perturbation factor.

SpecAugment is applied directly to the feature inputs of a neural network (i.e., filter bank coefficients). The augmentation policy consists of warping the features, masking blocks of frequency channels, and masking blocks of time steps, shown in Fig. 4.3.

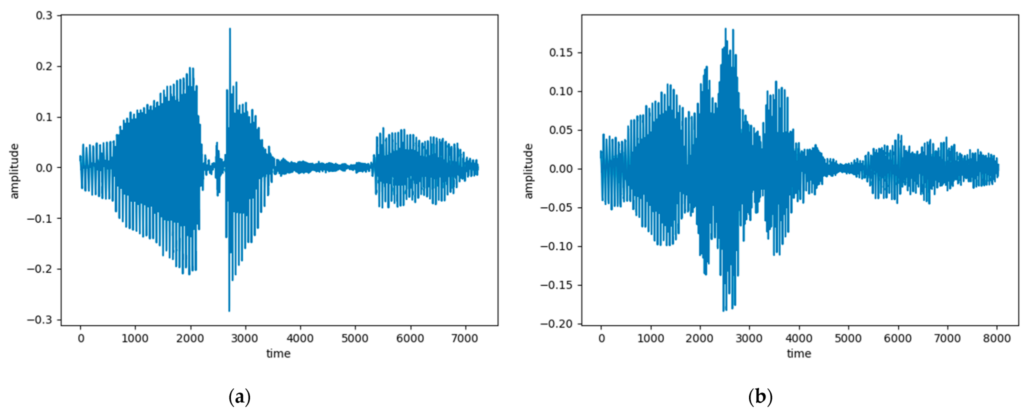


Figure 4.2: Speed Perturbation Example. (a) original speech and (b) Speed perturbed speech. After [31]

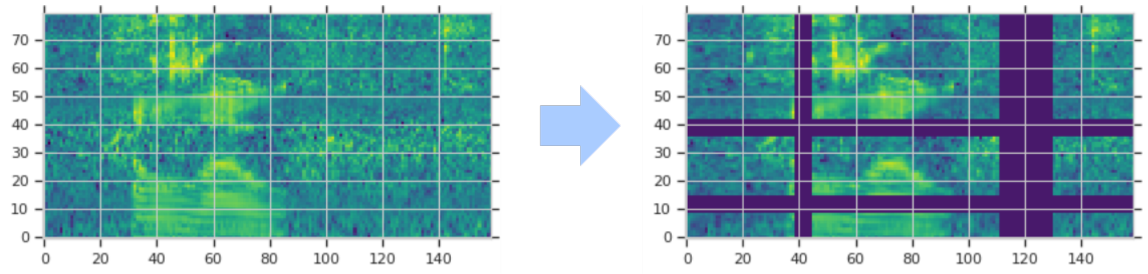


Figure 4.3: Example of SpecAugmentation

4.4 Experimental Results

The ETLT 2021 challenge organizers have provided DNN-HMM-based system as the challenge baseline for English track [28]. The DNN-HMM system has the same configuration as explained in sub-Section 3.3.

4.4.1 Data Augmentation Results Using CycleGAN

The significance of generated data using CycleGAN along with the other conventional augmentation techniques and its various combinations are also shown for dev set in Table 4.1. For the Kaldi system, SpecAugment (SA) did not show significant improvement. However, there is a remarkable relative reduction of 10-12.8 % WER compared to the baseline, when SA is combined with the other conventional data augmentation techniques namely, SP and VP, respectively. The CycleGAN generated data when augmented with the original data (which leads to two folds of data) gives a relative improvement of 4.3 % WER without any other conventional augmentation techniques. Compared to SA+SP+VP baseline, there is a relative reduction of 9 % in WER using CycleGAN generated data with

Table 4.3: WER (%) comparison for proposed data augmentation. After [71].

System	Language Model (LM)	Dev.	Eval.
B_1	3-gram	13.63	–
B_0	4-gram	13.31	33.21
B_2	5-gram	13.73	–
S_1	2-gram	14.44	33.56
S_2	3-gram	**12.11**	**32.21**
S_3	4-gram	12.82	32.64
S_4	5-gram	13.21	32.94

Table 4.4: WER (%) comparison with combination of proposed data augmentation. Here, $\oplus$ represents system combination operator. After [71].

System	Dev.	Eval.
$S_1 \oplus S_2$	12.02	32.11
$S_2 \oplus S_3$	**11.95**	**32.10**
$S_3 \oplus S_4$	13.12	32.68

SA+SP+VP augmentations. Relatively, the best performance of 12.11 % WER is achieved when all the augmentations are combined. An intelligibility of CycleGAN generated speech is verified using ASR model trained only using synthetic data. From Table 1, it is shown that only CycleGAN generated speech data in training gives 16.28 % WER which shows the intelligibility of the synthetic speech is not hampered much. Table shows WER comparison for the three DNN configurations with varying depths on dev set. Experimental results shows that 6 CNN layers followed by 9 TDNN-F layers gave the highest reduction in WER compared with the other configurations. Hence, in our further experiments, we took 6 CNN layers followed by 9 TDNN-F layers as a DNN configuration for AM.

Table 4.3 shows the ASR results of baseline B_0 and the proposed system. Here, S1 to S4 refers to our proposed system with augmented data with different n-gram LM. Similarly, in B1 & B2, the performance of baseline is observed in different n-gram LM. Baseline system B_0 results in 13.31 % WER and 33.21 % WER on Dev and Eval sets, respectively. Experiments were conducted for different configurations of LMs. Bi-gram LM showed increment of 1.13 % and 0.35 % in WER for Dev and Eval sets, respectively. However, tri-gram and 4-gram LMs have shown significant reduction in % WER compared to the baseline B_0. The system S_3 showed the highest reduction of 1.20 % and 1.00 % in WER for dev and eval sets, respectively. In addition, system combination for bi-gram with tri-gram, tri-gram with 4-gram, and 4-gram with 5-gram are investigated. Minimum Bayes Risk (MBR) approach is used for the system combination, keeping uniform weights for all the systems (i.e., hypothesis-level combination). We found that the system combination of $S_3 \oplus S_4$ gave the highest reduction of 1.36 % and 1.11 % in WER for Dev and Eval sets, respectively.

4.5 Chapter Summary

In this chapter, we evaluated the performance of data augmentation for children ASR using the CycleGAN model that learns a generalized mapping between source and target class. Specifically, a group of children's voices are mapped to another (without using one-to-one mapping) based on average F_0. The experimental results show that the proposed CycleGAN-based data augmentation approach gives 3 % relative reduction in WER for children ASR. To achieve mentioned best results, various DNN architectures along with different n-gram LMs were analysed, and the most optimum model was consist of 6 CNN 9 TDNNF layers and 4-gram LM, respectively.

CHAPTER 5

Self-Supervised ASR

5.1 Introduction

The vision of low-resource ASR was proceeded with the development of Self-Supervised Learning (SSL) ASR approach [80]. In this chapter, a detailed discussion of **wav2vec 2.0**, i.e., SoTA self-supervised ASR technique is presented. It has mainly two types of training, namely, pre-training and fine-tuning. The key component of this acoustic model is the *transformer model*. Thus, in this chapter, first transformer is explained followed by complete architecture of SSL acoustic model. Transformers are attention based encoder-decoder based model, which has brought a revolution in sequence-to-sequence modelling.

5.2 Transformer

Introduction of transformer model has redefined the Neural Networks (NN) by outperforming CNN and RNN. It is an attention-based model, that includes of encoder and decoder architecture. The architecture of the transformer model is shown in Fig. 5.1.

5.2.1 Encoder

The input data is provided to the encoder and hence, the speech representation learning from this input data is majorly performed in the encoder part. Fig. 5.2 shows encoder architecture of the transformer model.

Input Encoding

Vectorization (input encoding):
An input sequence cannot be directly feed to any NN-based model. Hence, to

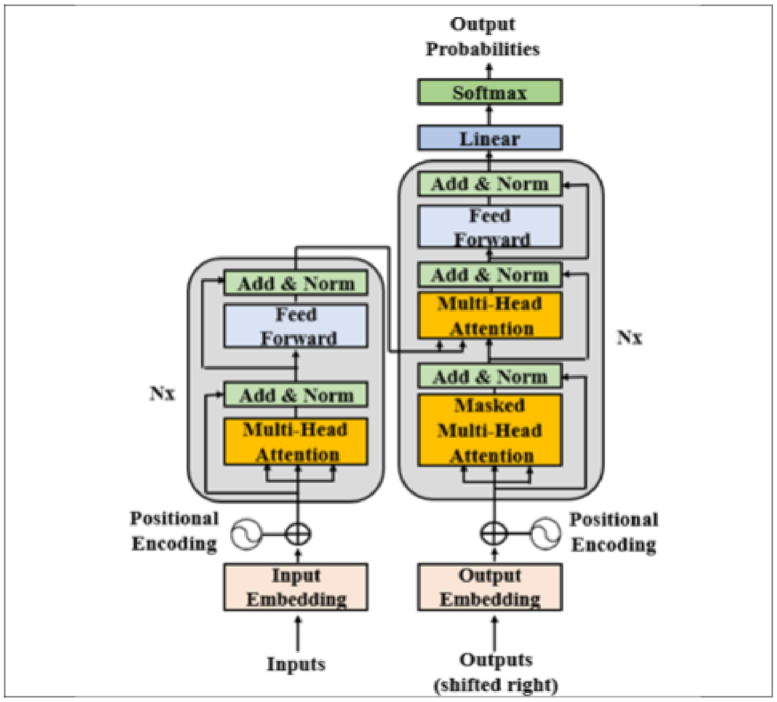


Figure 5.1: Model architecture of the transformer. After [80].

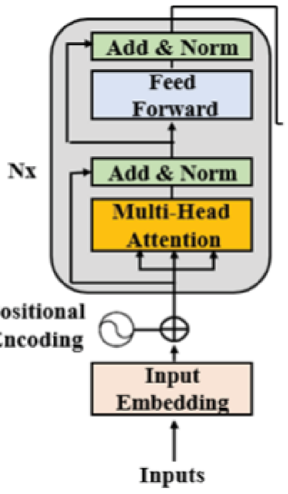


Figure 5.2: Encoder architecture of the transformer model. After [80].

train the NN-based model, the input values should be distinctive. Furthermore, the process of representation is called *vectorization*, if the input entities are in distinctive numerical representation. In Sub-Section 5.3, it will be discussed further w.r.t. *wav2vec 2.0* architecture, where a low-dimensional representation of input values are created.

Positional Embedding

A sequential data is systematically arranged input elements. Hence, the arrangement, in the other words, position of each input element is a crucial information. Thus, while having vector representation of an input value, positional information is also embedded into it. Few methods used for positional encoding are:

$$PE_{pos,2i} = sin(pos/10000^{2i/d_{model}}), \tag{5.1}$$

$$PE_{pos,2i} = cos(pos/10000^{2i/d_{model}}), \tag{5.2}$$

where *pos, d*, and *i* represents position, input sequence length, and dimension of model, respectively.

Attention Model

In deep learning, attention can be interpreted as a vector of significance weights, which memorize the long source sequence. In attention mechanism, long source sequences are memorized by shortcuts between the entire source input and the context vector. This context vector learns and the alignment between source and target. Hence, attention model has outperformed RNN, as RNN doesn't have long memory. Here, attention mechanism mainly works with three principal vectors, namely, *key, query*, and *value*. Where each key, query, and value are assigned a unique weight matrix, with dimension as:

$$\begin{aligned} Q_w &= d_{model} * d_k, \\ K_w &= d_{model} * d_k, \\ V_w &= d_{model} * d_v. \end{aligned} \tag{5.3}$$

Let there be n inputs entities and thus, the input matrix (M_{input}) will be of dimension $n * d_{model}$. Hence, multiplying M_{input} with key, query, and value weights, will provide their respective matrix as:

$$\begin{aligned} Q &= M_{input} * Q_w, \\ K &= M_{input} * Q_k, \\ V &= M_{input} * V_w. \end{aligned} \tag{5.4}$$

The final score matrix can be obtained with the following calculation [53],

$$A = Attention(Q, K, V) = softmax\left(\frac{Q * K^T}{\sqrt{d_k}}\right) V. \tag{5.5}$$

Multi-Headed Attention Model

For better understanding of sequence, multiple attention are considered in parallel. It acts as of having different aspects of understanding. To combine the information of all attention heads, each output score is concatenated as:

$$MultiHead(o/p) = Concat(A_0, A_1, A_2, ..., A_i) * W_o, \tag{5.6}$$

where dimension of W_o is x, d_{model}.

Add & Norm

This layer adds the input and output values of previous layer and then normalize it. A simple addition is performed here, and its result is denoted as V. Hence, this vector V is normalized using the following equation [56],

$$Normalize(V) = \gamma \left(\frac{v - \mu}{\sigma} \right) + \beta, \tag{5.7}$$

where μ is mean, σ is standard deviation, γ is a scaling factor, and β is the regularizing constant.

Feedforward Network

It is the basic neural network, as shown in Fig. 5.3. It consists of two hidden layers, where input and output dimensions are same.

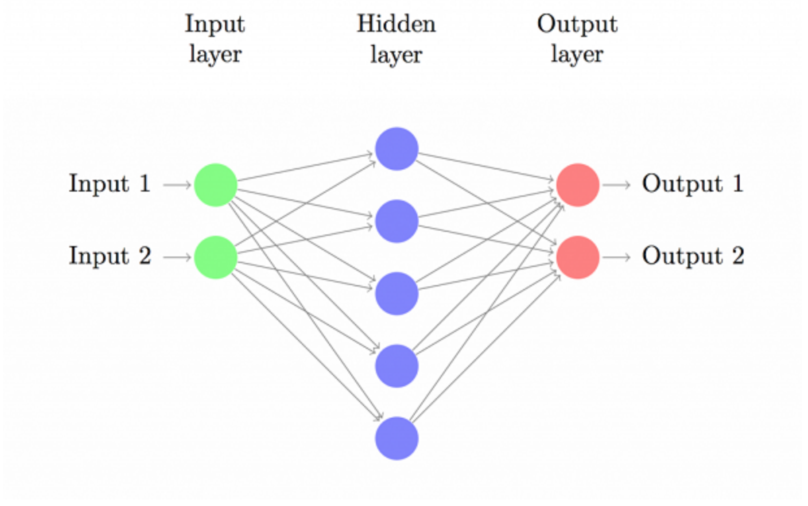


Figure 5.3: Feedforward neural network. After [74].

Algorithm 1: Execution of Encoder model

```
Input: Labeled audio : $L$
Output: Predicted output : $D$
    1. Input embedding
        (a) Vectorization L $\rightarrow$ V          // V = vector representation
        (b) Find P                          // P = Positional Embedding
        (c) X = V + P
    2. Generate $Q_w$, $K_w$ and $V_w$
    3. Calculate Multihead Attention = A
    4. Normalization = $A_N$
    5. $A_N$ passes through FFNN
    6. Normalization $\rightarrow$ D
```

Summary of Encoder Architecture

Following are the steps for execution of encoder model as shown in Fig. 5.2 :

- Input values are tokenized to have vector representation.
- Those vector representation are embedded with the positional embedding.
- With different key, query, and value weight matrix, for different attention heads, attention value is calculated w.r.t. each attention head.
- Attention value of each attention head is further concatenated.
- Output and input of multi-headed attention are summed and then normalized.
- This normalized output is further passed through a feedforward neural network (FFNN).
- input and output of FFNN are added and then normalized.
- This normalized output is carried forward to the decoder.

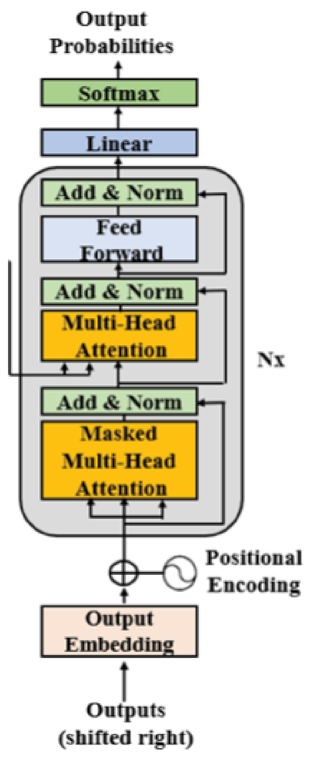


Figure 5.4: Decoder model architecture. After [80].

5.2.2 Decoder

Masked Attention

Masked attention is the one of the key difference between decoder and encoder. Here, in the masked attention model, attention is calculated till the predicted value, instead of the entire sequence. Whereas, in encoder's attention model, the entire sequence is considered for attention output.

Summary of Decoder Architecture

Following are the steps for execution of decoder model as shown in Fig. 5.4.

- Firstly, it creates a vector representation of the output sequence.
- It adds the positional embedding to those representations.
- Those values are feed to masked multi-head attention model.
- Input and output of above mentioned masked multihead attention are summed and then normalized.

Algorithm 2: Execution of Decoder model

```
Input: Labeled transcription : $L$
Output: Predicted probability : $P$
  1. Embedding of output, i.e., transcription
      (a) Vectorization L $\rightarrow$ V          // V = vector representation
      (b) Find P                                   // P = Positional Embedding
      (c) X = V + P
  2. Generate  $Q_w$, $K_w$ and $V_w$
  3. Calculate Multihead Attention = $A_1$
  4. Normalization = $A_{N1}$
  5. Collect $K_W$ and $V_W$ from encoder and $Q_W$ from $A_1$
  6. Calculate Multihead Attention = $A_2$
  7. Normalization = $A_{N2}$
  8. $A_{N2}$ passes through FFNN
  9. Normalization
 10. Linearization and Softmax
```

- For the next multi-head attention model, the **key** and **value** are taken from the encoder output, while the query is taken from the masked multi-headed attention of the decoder.
- The above mentioned inputs and outputs are summed and then normalized.
- This normalized value is given to FFN network.
- FFN input and output are summed and normalized.
- Furthermore, by applying linear operation and softmax normalization, we will achieve probabilistic information of each value.

5.3 Wav2vec 2.0

In this model, the learning is mainly divided in two phases as shown in Fig. 5.5,

1. Self-supervised learning as pre-training

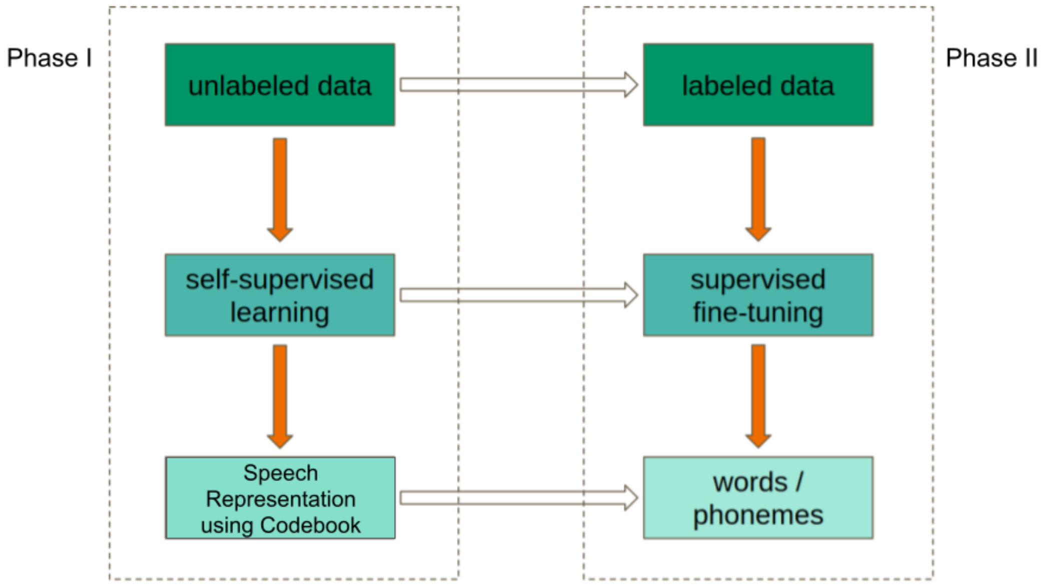


Figure 5.5: Flowchart of *wav2vec 2.0* architecture. After [7].

2. Supervised learning as fine-tuning

The major improvement in this model is pre-training [7]. As in pre-training process, the major part of training is done by generating a codebook similar to tokens for its acoustic representation. This learning is further fine-tuned with the supervised training. The major architecture of model is the same for both pre-training and fine-tuning process. The process of prediction is done in three major parts, namely;

- The raw audio wave is converted into low-dimensional latent representation, i.e., Z
- Transformer then creates contextualized representation, i.e., C
- Linear projection of the output

As shown in Fig. 5.6, for feature encoding task audio is split into small chunks with use of windowing. Each window is feed to a CNN, where CNN provides a latent representation of the input audio signal.

5.3.1 Pre-training

It is a self-supervised method of learning, which is done with unlabelled data. As the data is unlabelled, for learning quantized codebook representation is being created. In pre-training, the linear projection of the output prediction is not performed and training is done on unlabelled data.

Vector Quantization

Quantization is the process where the continuous data is represented as a finite set of values. This arises a question, that how speech can be represented as finite representation? Usually in supervised speech recognition, there are finite set of phonemes, similarly here finite representation are created in the form of codebook. In this model, G codebooks are made with V codewords representation. Here, the quantization is done by finding the best codeword from each codebook and, then concatenated and processed with the linear transformation. The loss calibrated in the process of quantization is known as **diversity loss** [7].

$$\mathcal{L}_d = \frac{1}{G * V} \sum_{g=1}^{G} -H(\bar{p}_g), \tag{5.8}$$

where H(*) represents entropy. In this equation, Entropy of each codebook is combined.

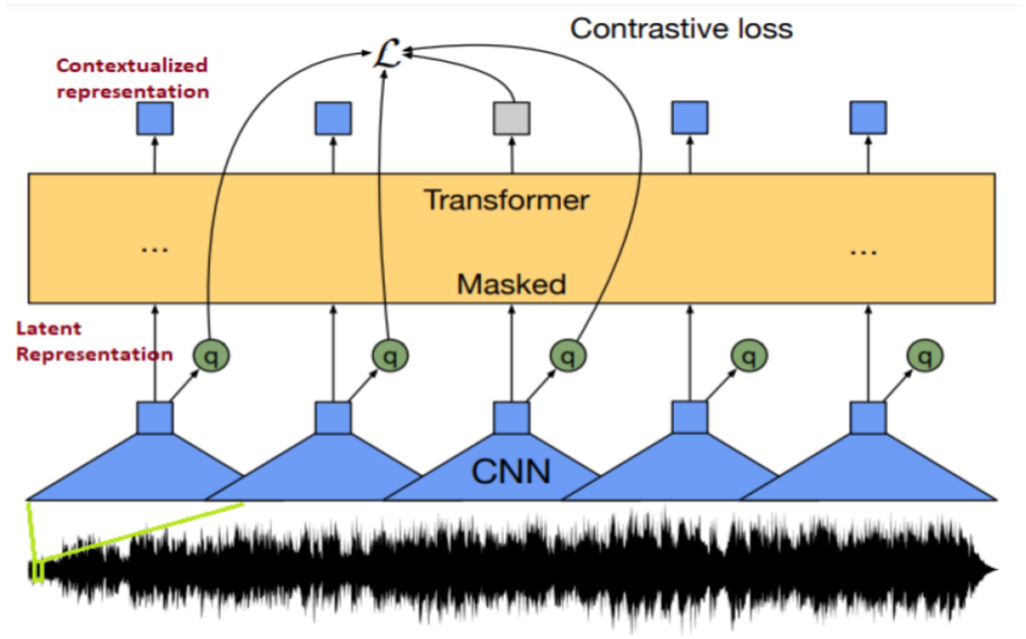


Figure 5.6: *Wav2vec 2.0* architecture. After [7].

Masking

Masking is performed on the input given before to the transformer. For masking, mainly two hyperparameters are defined, namely, p that is amount of masking, and M is number of consecutive time step to be masked. From the latent representation of speech, few time steps are selected and then their M consecutive time steps are masked. Loss calculated in this process is *Contrastive Loss*. Here, c_t is the context network output, which is centred over masked time step t, q_t represents codebook quantized value, and K are the possible distractors in codebook. Thus, the loss can be calculated as [7]:

$$\mathcal{L}_m = -\log \frac{\exp\left(sim(c_t, q_t)/k\right)}{\sum_{\bar{q}\sim Q_t} \exp\left(sim(c_t, \bar{q}_t)/k\right)}. \tag{5.9}$$

Here *sim* represents the scalar product of two vectors.

5.3.2 Fine-tuning

The quantization is not employed at this level of training. On top of the context representation C, a randomly initialized linear projection layer is added. The model is then fine-tuned using a modified version of SpecAugment and a typical CTC loss. CTC assigns the probability of any output given an input. CTC targets key challenges such as different input & output sequence length and alignment of input & output. SpecAugment is a technique, where time and frequency masking is performed on the signal, this delays the issue of overfitting and improves the performance.

5.4 Experiments

TIMIT dataset is used for the reference of experiment. This dataset consist of utterances of numeric value. The WER achieved in this experiment was 24 %, without LM. The codes are mentioned in Appendix 4. In implementation, first the transcription is cleaned, and a vocabulary dictionary is created. This dictionary is used in the process of tokenizer and feature extraction. Then using huggingface API the whole wav2vec architecture is implemented in form of model and processor.

Fairseq was also used for wav2vec 2.0 implementation. Here pretraining was also implemented, data manifestation and finetuning but faced difficulty in decoding. The following were pretraining checkpoints achieved:

```
    epoch: 430,
"train_loss": "2.458",
"train_ntokens": "40890.5",
"train_nsentences": "359.912",
"train_prob_perplexity": "279.368",
"train_code_perplexity": "254.365",
"train_temp": "1.77",
"train_loss_0": "2.358",
"train_loss_1": "0.081",
"train_loss_2": "0.018",
"train_accuracy": "0.56572",
 "train_wps": "1807.5",
"train_ups": "0.04",
"train_wpb": "40890.5",
"train_bsz": "359.9",
"train_num_updates": "24510",
"train_lr": "0.000475304",
"train_gnorm": "0.727",
"train_train_wall": "1244",
 "train_wall": "556557"
```

5.5 Chapter Summary

In this chapter, discussion of SoTA self-supervised ASR technique, i.e., wav2vec 2.0 was done. Wav2vec 2.0 mainly consist of the transformer model, which was also discussed in detail. Transformer is an attention-based encoder decoder architecture. Transformers introduced concept of masked multi-headed attention, which majorly contributed in outperforming the other sequence-to-sequence model. In wav2vec 2.0 the feature encoding was done with CNN and mainly the pretraining was introduced, which improved the performance of acoustic modelling significantly. In next chapter, proposed work based on this will be discussed, where using unlabelled data even fine-tuning process is improved, result in improvement of overall performance.

CHAPTER 6
NST Learning

6.1 Introduction

ASR is a fast-growing field, where reliable systems are made for high resource languages and for adult's speech. However, performance of such ASR systems is inefficient for children speech, due to numerous acoustic variability in children speech and scarcity of resources. To that effect, we propose to use the unlabelled data extensively to develop an ASR system for low resourced children in speech. Furthermore, SoTA *wav2vec 2.0* is used as the baseline ASR technique. The performance of baseline ASR is further enhanced with the intuition of Noisy Student Teacher (NST) learning.

6.2 Noisy Student Teacher (NST) Learning

The Noisy Student Teacher learning is a semi-supervised learning approach explored for various applications as mentioned in these literatures [87, 84, 58, 59, 77]. In this approach, there is a Teacher model, which is as usual trained on hard labelled[1] data. While further similar or larger models are taken for iterative self-training, where model in each iteration is trained by combining hard-labelled and pseudo-labelled data. pseudo labelled data are unlabelled data with logits[2] produced by a preceding model (i.e., either Teacher or $N-1^{th}$ Student model). The model that experience iterative training is known as Student model. The above process is illustrated via Fig. 6.1.

[1]Hard-labelled data is a data whose audio files are provided with its respective reliable transcription

[2]Logits are unnormalized raw prediction generated by a model

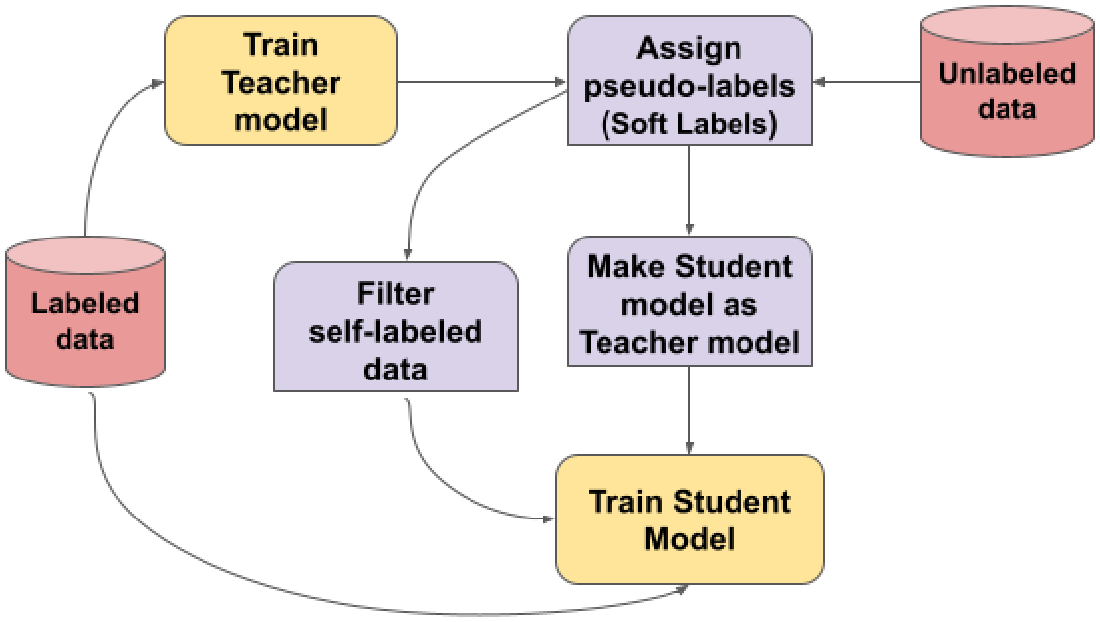


Figure 6.1: Functional block diagram of NST learning. After [84].

6.3 Proposed Work

The proposed method aims to have a fusion of the state-of-the-art self-supervised technique (i.e., *wav2vec 2.0*) and an efficient self-training approach (i.e., NST learning). The *wav2vec 2.0* technique amateurs learning with the least possible data, where combining it with iterative self-learning was not yet explored in the literature. While, the concept of NST is limited to the use of unlabelled data by generating its pseudo labels or adding noise to both (i.e., hard and pseudo labelled audio) data during training. The major limitation subjected to this approach is that, the learning information of the neural network model (i.e., weights or checkpoints) was **not** transferred in the process. In the other words, the drawback of NST learning here is if a model has already learned significantly with some hard-labelled data, repeating the same training again from the beginning will be a redundant task. This redundant training can be optimized by SSL approach *wav2vec 2.0*, where in the supervised fine-tuning, the learning parameters are initialized with the pre-trained model (i.e., trained on unlabelled data). This use of checkpoints of pre-trained model is exploited in our proposed work. As shown in Fig. 6.2 and Algorithm 3, the Teacher model is initialized with pre-trained model checkpoints, while in all Student models, the parameters are initialized by a preceding fine-tuned model. In the other words, if we are training Student1 model, then the checkpoints achieved after complete training of Teacher model will be used to initiate the training parameters for Student. While for training of Student

Algorithm 3: Self-supervised learning with noisy student teacher learning. After [15]

```
Input: Labeled data : $L$, Unlabeled data : $U$, Checkpoints : $C$
Output: Weights : $W$, Model : $f$, Teacher Model : $T$,
        Predicted output : $y$,   Student Model : $S$
  1. Train Teacher model $T_0$ with labelled data $L$ and Pre-trained model
     Checkpoints $C_0$.
  2. Train Student $S_k$ by
     (a) pseudo-label created by teacher $T_k$ model
         $U_{Label} = f^{T_k}(U)$
     (b) Initialize weights of $S_k$ with teacher $T_k$ Checkpoints
         $W_{S_k} = C_{T_k}$
     (c) Apply SpecAugment on hard and pseudo-labels
         $D = Augment(L \cup U_{Label})$
     (d) Trained $S_k$ with CTC Loss
         $y_{S_k}(i) = f_{S_k}(x(i))$ for $x(i) \in D$
  3. Set $T_{k+1} = S_k$ and repeat step 2.
```

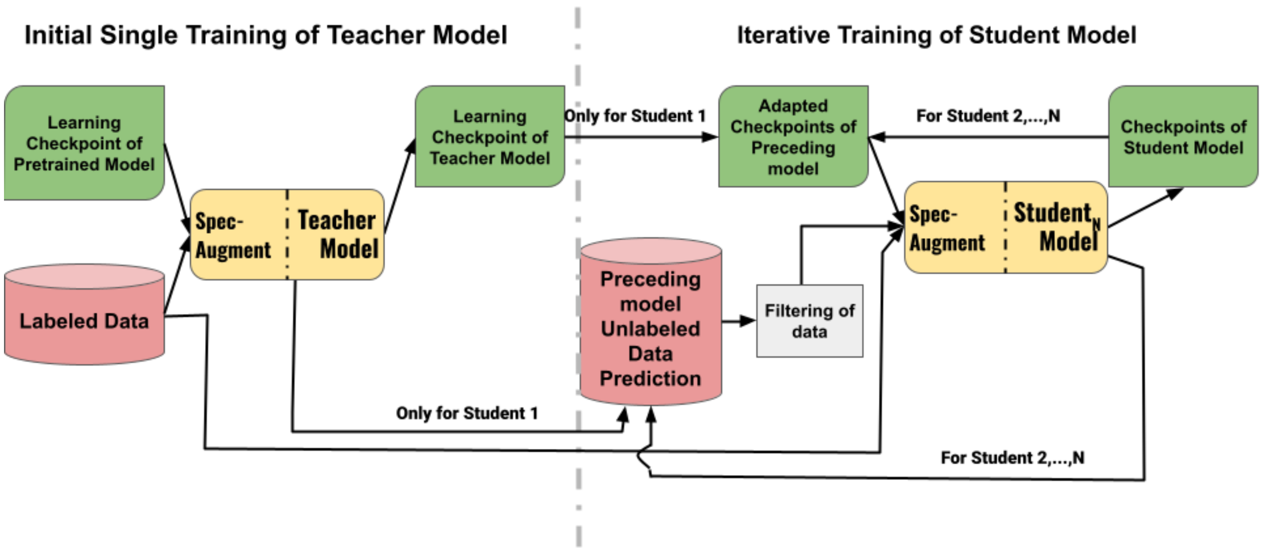

Initial Single Training of Teacher Model
Iterative Training of Student Model
Learning Checkpoint of Pretrained Model
Labeled Data
Spec-Augment
Teacher Model
Learning Checkpoint of Teacher Model
Only for Student 1
Adapted Checkpoints of Preceding model
For Student 2,...,N
Checkpoints of Student Model
Spec-Augment
Student N Model
Preceding model Unlabeled Data Prediction
Filtering of data
Only for Student 1
For Student 2,...,N

Figure 6.2: Proposed NST approach in tandem with self-supervised learning. After [15].

2,3,..,N model, its preceding model's (i.e., Student 1,2,...,(N-1) model) checkpoints are used to initialize its parameters.

In Student training, the training data is also increased due to addition of pseudo-labelled data. Thus, in all Student model training, the training data is a combination of hard and pseudo-labelled data. Here, pseudo-labelled data is unlabelled data with the transcription generated from the parent (i.e., preceding student or teacher) model. The pseudo-labelled transcription is further correct by LM and then the empty transcription files are filtered out. While in experimentation, the effect of LM use for pseudo labels generation is also mentioned. The LM used is a word-level model, which targets the correct prediction of each word using Bayes' theorem [44].

6.4 Experimental Setup

6.4.1 Pre-processing

Intensive processing was required in this corpus. First, all the audio files with only non-speech information were removed, such as distortion, noise, laughter, and giggle. From the remaining audio, all the non-speech tags were removed, or replaced with a single "unknown" tag, as shown in Table 6.1. Here, each audio is consisting of a single sentence and thus, all further punctuation marks apart from apostrophe (′) are removed. To reduce further variability in the list of tokens, all the text were converted into the lower case. After this process, a set of vocabulary dictionary was created, resulting into 26 alphabet characters, apostrophe, Padding ([PAD]), and unknown tag ([UNK]). These 30 tokens were randomly given a numerical representation. From this set of tokens, a tokenizer is created using HuggingFace ASR Processor [1] , which will encode the string with a numerical character-level representation. This encoded string along with padding is dynamically added to each audio representation feed to the learning model. Fine selection of hyperparameters needs to be done empirically.

6.4.2 Teacher Training

Teacher training model is a *wav2vec 2.0* architecture proposed in [7]. In this architecture, checkpoints of a pre-trained model are considered to initialize the parameters of a fine-tuning model. Here, the pre-trained model used is a Facebook's base model of wav2vec 2.0 [2], which was trained on 53k hours of unlabelled Libri-speech data. The learning rate was initialized with 10^{-4}, 30 epochs, batch size of 8, weight decay was 0.005, and warm up step was 1000. SpecAugment was enabled in order to avoid overfitting. Grouping of audio is performed for training, based on length of audio, due to which maximum length of labels for audio is dynamically predicted batchwise.

Table 6.1: Tokenization of raw non-speech labels. After [15].

Raw Labels	**Token**
<BREATH>, <DISCARD>, <SILENCE>, <SNIFF>, <No-Signal>, <COUGH>, <Distortion>, <ECHO>	Tags are removed
<NOISE>, <SIDE_SPEECH>, unknown utterance	[UNK]
Padding	[PAD]

6.4.3 Student Training

Architecturally, there is not much difference in the Student model. The hyperparameters were initialized as learning rate was 10^{-6} with 15 epochs, 2500 warm up steps, and batch size of 12. As mentioned in Section 6.3 and shown in Fig. 6.1, instead of using checkpoint of pre-trained model for N^{th} Student model training, checkpoints of preceding model are used. For example, if $N = 1$ then Teacher model is used to initialize the model parameter, while if $n = 2, ..., N$ then $(n-1)^{th}$ student model was used to initialize the model parameters.

6.5 Experimental Results

In this section an overall evaluation of the experiments is done which is divided in four parts. In first sub-section 6.5.1 key performance measure, WER is analysed. In next sub-section how LM impacts the performance of proposed work. Then, in a further subsection loss behaviour is analysed and in the last sub-section the output transcription are studied in detail.

6.5.1 Performance Evaluation

The experiments are performed in three stages, namely, Teacher, Student1, and Student2, as shown in Table 6.2, where both the results of acoustic modelling and improved results with LM are presented. In this experiment, LM was not used to generate the pseudo labels. It can be observed that the major part of acoustic

Table 6.2: Experimental results without LM use in pseudo labels. After [15].

Model Type	Dev. WER %	Test WER % Without LM	Test WER % With LM
Teacher	33.9	34	32.3 [5.2%]
Student1	28.8	32.6	31.0 [4.9%]
Student2	27.4	33.3	31.5 [5.4%]

Values in [.] refers to the relative improvement occurred due to LM

Table 6.3: Experimental results with LM corrected pseudo labels. After [15].

Model Type	Test WER % Without LM	Test WER % With LM
Teacher	34	32.3
Student1	32.3	31.0
Student2	**32.0**	**30.6**

modelling is performed in Teacher training. The wav2vec2.0 approach for ASR is meant to give a relatively lesser WER % with the least possible data, however, in case of children ASR task, the quality of data transcription and intelligibility of speech both are distorted. Thus, achieving a good accuracy with a fewer data is a difficult proposition. Through our approach, we have tried to address this issue as in Teacher model, major acoustic modelling is achieved. Hence, in our experiments, we achieved 34 % WER without LM. Further, training of the Student model will result in more intuitive learning of our acoustic model. In Student 1, its learning parameters had begun from the checkpoints of Teacher model (shown in Fig. 6.3) and hence, lesser epochs were required. Further, Student model has additional pseudo-labelled data, which will add into the learning of Student model. Thus, we achieved reduction of absolute 5.1 % WER on Dev set in Student 1 training. Similar approach is applied to Student 2 training, where on Dev set, the WER was reduced by absolute 1.4 %. This improvement is significant, however, lesser difference compared to the preceding layers. This reflects the saturation of the underlying model and hence, it will be highly prone to over-fitting. This possibility was reflected in test results, where a significant improvement of 1.4 % without LM was observed in test on Student 2 model compared to the Teacher model, while in Student 2 model, without LM it decreased by 0.7 % WER. This drawback is discussed in the next sub-Section.

In Table 6.3, results are shown with LM, being used for pseudo label generation. It should be noted that the divergence in performance, observed in between Student 1 and Student 2 training is solved here, with the use of LM. This is due to reduction of prediction loss transfer in consecutive training model.

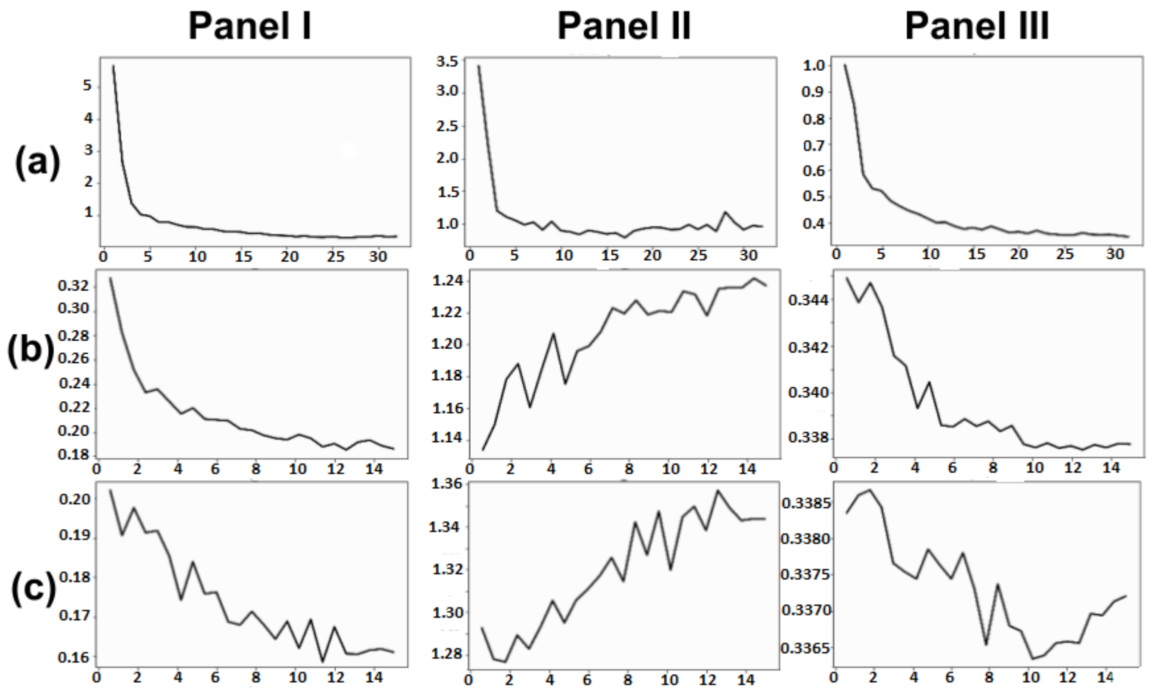


Figure 6.3: Loss and WER curve of Teacher and Student models. Where, (a) is for Teacher, (b) is for Student 1, (c) is for Student 2, while Train loss (Panel I), Dev loss (Panel II), and Dev WER (Panel III) are shown w.r.t. epochs. After [15].

6.5.2 Incorporation of Language Model (LM)

ASR experiments, mentioned in Table 6.2, have neither used LM for training nor for generation of pseudo labels. Due to this, a major limitation faced in this approach was transfer of prediction error. It was noticed that the pseudo labels generated by Teacher model were having some spelling errors. These errors were not rectified in the succeeding training. To overcome this issue, LM is required to avoid this situation of error transfer. Thus, in Table 6.3, LM is used for generation of pseudo labels, and it shows better performance as the transfer of error is reduced.

6.5.3 Analysis of Loss Curves

The projection of loss and results on Dev set are shown in Fig. 6.3. In Fig. 6.3(a) a smooth trend of decrement of train loss, Dev loss, and Dev WER plot w.r.t. epochs can be observed. While in Fig. 6.3(b) and Fig. 6.3(c), a lot of distortion is experienced in all the three cases. In addition, the Dev curve seems diverging, while on the other side, peak difference is just of 0.01 of loss value, which is quite negligible. Graph in Fig. 6.3 depicts high training resolution in Student models. The loss and WER graphs of Student starts nearly from the values, where Teacher model graphs have concluded. This strengthens our hypothesis of eliminating redundant training and emphasizing only on getting better resolution of acoustic modelling.

6.5.4 Analysis of Predicted Text

Table 6.4: Ablation study of predicted transcription without LM. After [15].

Original String	**Predicted String**
energy	energ
to to wind up the wire more on the rivet and you'll have more power	to t to wind up the wire more on the rivet and you'l have more powers
you'll get more washers	ill get more washers
um more electricity	more electricity
well electricity in a circuit they can like you can combine them to make something work light up	he electricity and a circuit they can thike you can mind them to make something work light up

In this study, behavioural analysis of results is done with a few hand-picked examples presented in Table 6.4 in order to justify the analysis. It can be observed that children's style is naive and thus, they repeat the word multiple times. However, repetition of voiced sound, results in repetition of consonants. For example, children is speaking "to to" while it is interpreted as "to tto" or "to t to", although it is an error, however, it reflects a good resolution of acoustic modelling. In addition to this, a major drawback observed is w.r.t. dense sentences, i.e., as the audio are of maximum 10 seconds duration only, while some of them have transcription of more than 300 characters in a string. In this case, the children speaker is speaking so fast that the model is unable to understand. Hence, performance on high tempo speech will be less. Non-speech words, such as "um" and "ahh" were not identified even though they were part of the transcription, because these are like vowel sounds, which represent a small pause and their occurrence in the overall corpus is very less.

6.6 Chapter Summary

In this chapter, we have presented an approach to solve challenges of children ASR by proposing a novel fusion architecture of *wav2vec 2.0* and NST learning. In particular, we have tried to adapt the goodness of both the techniques, i.e., using less labelled data and unlabelled data in an iterative manner. Hence, we were able to relatively improve the results by 19.1 % WER in Dev set and 10 % WER for Test set after using LM for pseudo labels. During ASR experiments, it was experienced that the hyperparameters should be chosen carefully, because this approach is highly prone to overfitting. The use of LM is a must, such that loss transfer while training should be minimum.

CHAPTER 7
Beamforming for Replay Attacks

7.1 Introduction

This chapter investigates the capability of the Delay and Sum (DAS) beamformer to extract the reverberation characteristics in replay speech signals. The replay mechanism consists of the characteristics of the recording, playback devices, and corresponding environments due to which reverberation characteristics are embedded into the replay speech signal. Further, analysis is presented for DAS *vs.* Minimum Variance Distortionless Response MVDR beamformer for replay SSD task. MVDR is a state-of-the-art beamformer for speech enhancement and farfield ASR applications, as it successfully nullify the reverberation effects in distant speech signals. Whereas, DAS suppresses the additive noise and retains the reverberation effect observed in the output signal and hence, DAS is suitable choice for replay SSD task.

7.2 Signal Modelling and Beamforming

7.2.1 Signal Modelling for Microphone Array Signal

Assuming the acoustic paths between the sound source and the microphones in the array to be linear and time-invariant (LTI) [20], the speech signal received by the N-element microphone array is modelled as [65, 13, 55]:

$$\begin{aligned} z_k(n) &= r_k(n) * g(n) + v_k(n), \\ &= y_k(n) + v_k(n), k = 1, 2, ..., N, \end{aligned} \tag{7.1}$$

where n is the discrete-time index, $r_k(n)$ denotes the impulse response of the acoustic medium between desired source $g(n)$ and k^{th} microphone, * denotes the convolution operation, and $v_k(n)$ represents noise at the k^{th} microphone. Here, we have assumed that the speech signal $y_k(n)$ and the noise signal $v_k(n)$ are zero-mean and uncorrelated. Furthermore, for replay speech signal, impulse responses of recording devices ($a(n)$) and environment ($b(n)$) as well as impulse responses of playback devices ($c(n)$) and environment ($d(n)$) are convolved. Let $e(n)$ represents the combination of these impulse responses, and can be represented as [3]:

$$e(n) = a(n) * b(n) * c(n) * d(n). \tag{7.2}$$

Hence, the replay speech signal (z_{kr}) can be represented as:

$$\begin{aligned} z_{kr}(n) &= r_k(n) * e(n) * g(n) + v_k(n), \\ &= y_{kr}(n) + v_k(n), k = 1, 2, ..., N. \end{aligned} \tag{7.3}$$

Thus, the characteristics of the $y_{kr}(n)$ in eq. (7.3) is different from that of $y_k(n)$ because of the additional impulse response $e(n)$ caused by the replay mechanism. Considering this $e(n)$ as distinguishing characteristics of the replay spoof, it can be emphasized using suitable signal processing technique for replay SSD. To that effect, we present the significance of the DAS beamformer over MVDR for replay SSD through mathematical analysis, and it this hypothesis validated using experiments.

The representation of the received signal in eq. (7.1) in frequency-domain can be expressed as [13]:

$$\begin{aligned} Z_k(\omega) &= R_k(\omega) \odot G(\omega) + V_k(\omega), \\ Z_k(\omega) &= Y_k(\omega) + V_k(\omega), \quad k = 1, 2, ..., N, \end{aligned} \tag{7.4}$$

where $Z_k(\omega)$, $R_k(\omega)$, $G(\omega)$, $V_k(\omega)$, and $Y_k(\omega)$ are the discrete-time Fourier transforms (DTFTs) of $z_k(n)$, $r_k(n)$, $g(n)$, $v_k(n)$, and $y_k(n)$, respectively. Here, the symbol $\odot$ represents the componentwise multiplication operation. The frequency-domain representation of N-microphone array can be represented in the matrix form as :

$$\begin{aligned} \mathbf{Z}(\omega) &= \mathbf{R}(\omega) \odot G(\omega) + \mathbf{V}(\omega), \\ &= \mathbf{Y}(\omega) + \mathbf{V}(\omega), \end{aligned} \tag{7.5}$$

where,

$$\begin{aligned}\mathbf{Z}(\omega) &= [Z_1(\omega), .., Z_N(\omega)]^T, \mathbf{R}(\omega) = [R_1(\omega), .., R_N(\omega)]^T, \\ \mathbf{G}(\omega) &= [G(\omega), .., G(\omega)]^T, \mathbf{Y}(\omega) = [Y_1(\omega), .., Y_N(\omega)]^T, \\ \mathit{and}, \mathbf{V}(\omega) &= [V_1(\omega), .., V_N(\omega)]^T.\end{aligned} \tag{7.6}$$

7.2.2 Delay and Sum (DAS) Beamforming

The DAS beamformer is a primitive beamforming technique for noise reduction in the array signal processing literature [32, 34]. This involves reinforcing the desired signal while suppressing the unwanted noise signals. The conventional DAS beamformer will delay all the input signals in time, such that the array sensor can focus on one direction. Hence, the summation of the delayed signals will result in suppression of noise, which is arriving from the other directions. Furthermore, it can be postulated that the summation of the delayed signals leads to cancellation of *additive (random) noise*. Fig. 7.1 shows the functional block diagram of DAS beamformer from the receiver end. Here, weights for corresponding single channel microphone signal in a microphone array are shown. The time-domain representation of DAS beamformer is given by [10]:

$$\mathbf{b}(n) = \frac{1}{\beta}\sum_{k=1}^{N} w_k z_k(n - \tau_k). \tag{7.7}$$

Furthermore, the frequency-domain representation of DAS beamformer is given by taking DTFT of eq. (7.7), i.e., [85]:

$$\begin{aligned}\mathbf{B}(\omega) &= \frac{1}{\beta}\sum_{k=1}^{N} w_k e^{-j\omega\tau_k} Z_k(\omega), \\ &= \mathbf{W}^H \mathbf{Z}(\omega),\end{aligned} \tag{7.8}$$

where

$$\mathbf{W} = \frac{1}{\beta}\sum_{k=1}^{N} w_k e^{-j\omega\tau_k}, \tag{7.9}$$

where w_k is the elementwise weighting for the spatial window, β is the summation of the weights, and $\mathbf{W}$ is the steering vector (optimized wights vector) of desired *linear* phase shift and weights. The superscript H denote the Hermitian transpose. The $\mathbf{B}(\omega)$ represents the frequency response of beamformed signal. The power at the output of the beamformer is calculated by taking autocorrelation of the beamformer output, i.e.,

$$\mathbf{p}(\omega) = \mathbb{E}[|\mathbf{B}(\omega)|^2], \tag{7.10}$$

where $\mathbb{E}[\cdot]$ is the expectation operator.

Algorithm 4: DAS Beamformer. After [48]

```
Input: Speech signal $z(n)$
Output: DAS Beamformed Speech Signal
1 for Every $k = 1, 2 \dots N$ microphones do
2     $Z_k(\omega) = DTFT(z_k(n))$
3     $Z_{kdelay} = Z_k(\omega)e^{-j\omega\tau_k}$          // Signal delayed
4     $Z_{kscaled} = Z_{kdelay} * w_k$                       // Signal Scaled
5 $\sum_{k=1}^{N} Z_{kscaled}$                               // Sum all channel
6 $\mathbf{B}(\omega) = \frac{1}{\beta}\sum_{k=1}^{N} Z_{kscaled}$   // Normalized
```

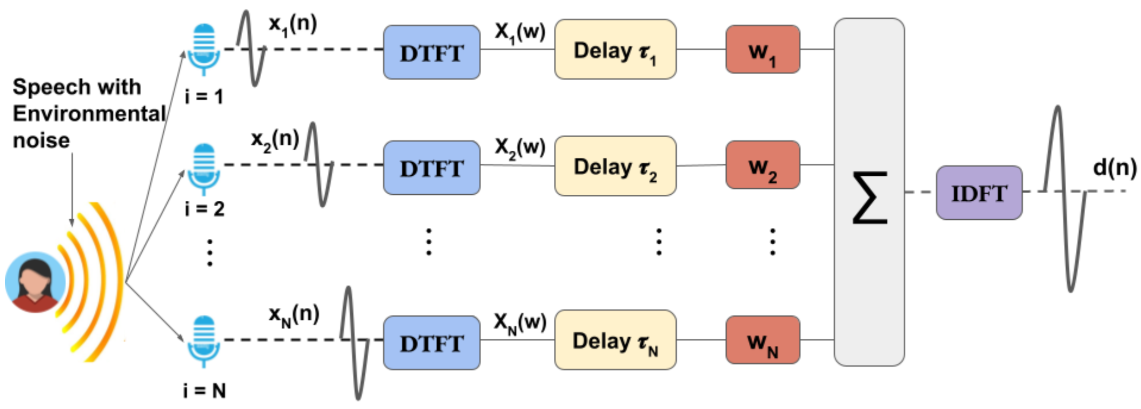


Figure 7.1: Functional block diagram of DAS beamformer. Different alignment of signals after the microphones (k=1 to N) indicates difference in time of arrivals. After [48].

7.2.3 Minimum Variance Distortionless Response (MVDR)

The perfect reverberation cancellation in multi-channel filtering is achieved using MVDR beamformer [20, 29]. In this approach, Signal-to-Noise Ratio (SNR) of the multi-channel audio signal is significantly improved by minimizing the distortion (noise) [14]. Hence, to minimize the noise power, the variance is minimized in such a way that it will retain the all-pass characteristics of the audio signal. The price paid for this minimization of noise, is an increase in the computational complexity of the system. The matrix of output power of MVDR beamformer is given by:

$$\begin{aligned}\mathbf{p}(\omega) &= \mathbb{E}[|\mathbf{B}(\omega)|^2], \\ &= \mathbf{W}^H\mathbf{C}(\omega)\mathbf{W},\end{aligned} \tag{7.11}$$

where $\mathbf{C}(\omega)$ and $\mathbf{W}$ represents the matrix of cross-power-spectral density and initial weight matrix, respectively. Here, for computation of FFT, finite overlapping of time frames are averaged to estimate the sampled co-variance matrix. The co-variance matrix for L time frames is given by [10]:

$$\hat{\mathbf{C}}(\omega) = \frac{1}{L}\sum_{l=0}^{L-1}\mathbf{Z}_l(\omega)\mathbf{Z}_l^H(\omega), \tag{7.12}$$

where $\hat{\mathbf{C}}(\omega)$ is estimated co-variance matrix. The computation of weights is done by minimizing the noise and unity gain of the desired signal, i.e.,

$$\begin{aligned}&\underset{\mathbf{W}}{arg\ min} \quad \mathbf{W}^H(\omega)\hat{\mathbf{C}}(\omega)\mathbf{W}(\omega), \\ &subject\ to \quad \mathbf{W}^H(\omega)\mathbf{s} = 1,\end{aligned} \tag{7.13}$$

where $\mathbf{s}$ represents the steering vector, which is the most crucial matrix for calculation of the beamformer. It provides the directional information of microphone array. During this minimization, it affects the impulse response of the medium. Let d_k be the desired direction representation for element k. Then, s_k is given by :

$$s_k = e^{jwd_k}. \tag{7.14}$$

Eq. (7.13) is solved by utilizing Lagrange multipliers [79]. Hence, through MVDR beamformer, the obtained solution of optimum weight matrix is given by:

$$\mathbf{W}_o(\omega) = \frac{\hat{\mathbf{C}}^{-1}(\omega)\,\mathbf{s}}{\mathbf{s}^H\,\hat{\mathbf{C}}^{-1}(\omega)\,\mathbf{s}}. \tag{7.15}$$

Hence, the optimum weights are used to get beamformed signal from the microphone array signal, i.e.,

$$\mathbf{B}(\omega) = \mathbf{W}_o^H(\omega)\mathbf{Z}(\omega). \tag{7.16}$$

After optimization of weights, the output power ($\mathbf{p}_o$) of MVDR beamformer is given by:

$$\mathbf{p}_o(\omega) = \mathbf{W}_o^H(\omega)\hat{\mathbf{C}}(\omega)\mathbf{W}_o(\omega). \tag{7.17}$$

Algorithm 5: MVDR Beamformer. After [48].

```
Input: Speech signal $z(n)$
Output: MVDR Beamformed Speech signal
1 $\hat{\mathbf{C}}(\omega) = \frac{1}{L}\sum_{l=0}^{L-1} \mathbf{Z}_l(\omega)\mathbf{Z}_l^H(\omega)$.          // Co-variance Matrix
2 $s_k = e^{jwd_k}$.          // Steering Vector
3 $\mathbf{W}_o(\omega) = \frac{\hat{\mathbf{C}}^{-1}(\omega)\,\mathbf{S}}{\mathbf{S}^H\hat{\mathbf{C}}^{-1}(\omega)\,\mathbf{S}}$.          // Optimized Weight
4 $\mathbf{B}(\omega) = \mathbf{W}_o^H\mathbf{Z}(\omega)$.
```

7.3 Experimental Setup

7.3.1 Feature Sets and Classifier Used

This paper aims to analyse relative significance of DAS *vs.* MVDR beamforming. Hence, we utilized generally used feature sets and classifiers for anti-spoofing applications. The state-of-the-art baseline feature sets for anti-spoofing application are the Constant-Q Cepstral Coefficients (CQCC) and Linear Frequency Cepstral Coefficients (LFCC) feature sets [19, 51, 75, 76]. Along with these feature sets, we also explored Mel Frequency Cepstral Coefficients (MFCC) and Spectral Magnitude Cepstral Coefficients (SMCC) feature sets [54, 81]. MFCC is explored in this task as it is the state-of-the-art feature set for speech and speaker recognition task [18]. SMCC feature set is extracted from the log-magnitude spectrogram followed by DCT operation. All the feature sets consists of static, Δ, and $\Delta\Delta$ coefficients. The dimension of CQCC, LFCC, MFCC, and SMCC are *90, 120, 42,* and *90*, respectively.

Furthermore, for classification task, we have used Gaussian Mixture Model (GMM)-based classifier. In particular, we have used 512 mixture components to train the GMM for replay SSD. As this choice of mixtures gives relatively better performance.

7.3.2 Spectrographic Analysis

To analyse the effect of beamforming on ReMASC dataset, we observe Energy Spectral Density (ESD) using spectrogram. Fig. 7.2 shows the spectrographic analysis of genuine (Panel I) and its corresponding spoofed speech utterances (Panel II) from ReMASC and its beamformed versions (i.e., DAS and MVDR). Furthermore, the Fig. 7.2(a) represents that the speech signal is from ReMASC dataset. Whereas the Fig. 7.2(b), and Fig. 7.2(c) shows the speech signal corresponding to DAS and MVDR beamformed versions of ReMASC dataset, respectively. It can be clearly observed from Fig. 7.2 (b) and (c) that, due to beamforming, the ESD of the genuine as well as spoofed speech signals get suppressed in the high frequency region. In particular, the rectangular boxes of Panel II shows that using MVDR beamformer, the noise gets suppressed efficiently as compared to the original ReMASC and its DAS beamformed version. Hence, this observation can lead to inference, that the MVDR beamformer suppresses the reverberation noise, whereas the DAS beamformer retains the reverberation characteristics primarily.

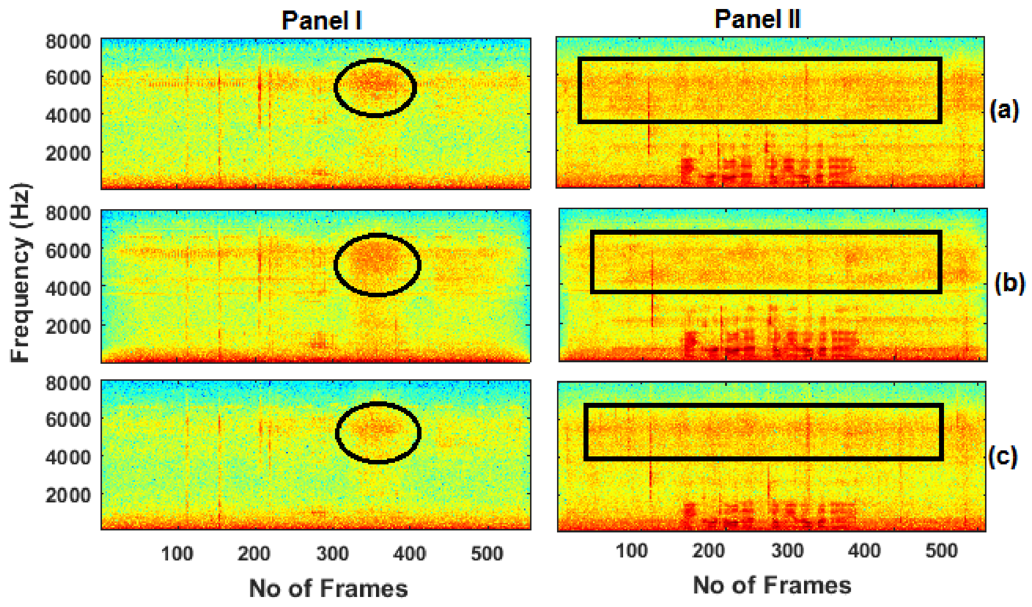


Figure 7.2: Spectrographic analysis of the genuine (Panel I) *vs.* spoofed (Panel II) speech. Speech signal from (a) ReMASC, and its (b) DAS *vs.* (c) MVDR beamformed versions. After [16].

7.4 Experimental Results

We have evaluated performance of DAS *vs.* MVDR beamformers using % EER. The SSD systems are developed for CQCC, MFCC, LFCC, and SMCC feature sets using GMM-based classifier for all the three datasets, i.e., ReMASC and its DAS *vs.* MVDR beamformed versions. The % EER on development and evaluation sets are shown in Table 7.1 for all the three variants of datasets. The experiments are performed using static, Δ, and $\Delta\Delta$ features. However, it can be observed that *only static* features performed better than all the other combinations. Hence, all the results reported in Table 7.1 are obtained using only static features. It can be observed from Table 7.1 that the improved performance is obtained on the DAS beamformed ReMASC than that for the ReMASC and its MVDR beamformed version, for all the feature sets considered in this study. In particular, using DAS beamformer, the absolute reduction in % EER over ReMASC, is 3.25%, 4.61%, 12.37%, and 4.70% for CQCC, LFCC, MFCC, and SMCC feature sets, respectively, on the evaluation set. It is important to note that results for MVDR are not better than that for the case of without beamforming. This suggests that the DAS beamforming can be potentially utilized to improve the performance of the replay SSD system for VAs. Furthermore, the performance of all the systems are also shown using Detection Error Trade-off (DET) curves in Fig. 7.3. In particular, the Fig. 7.3(a) is for development set and Fig. 7.3(b) is for evaluation set of all the three datasets using CQCC and LFCC features.

Table 7.1: Results (in % EER) on ReMASC and its DAS *vs.* MVDR beamformed versions using various feature sets. After [16].

Beamforming Technique →	**Without Beamforming**		**DAS**		**MVDR**	
Feature Set ↓	**Dev.**	**Eval.**	**Dev.**	**Eval.**	**Dev.**	**Eval.**
CQCC	19.15	22.12	**17.30**	**21.40**	38.15	23.61
LFCC	22.03	22.97	**22.01**	**21.91**	37.64	24.59
MFCC	25.11	26.58	**23.99**	**23.29**	40.73	33.95
SMCC	30.88	26.36	**28.74**	**25.12**	45.72	39.35

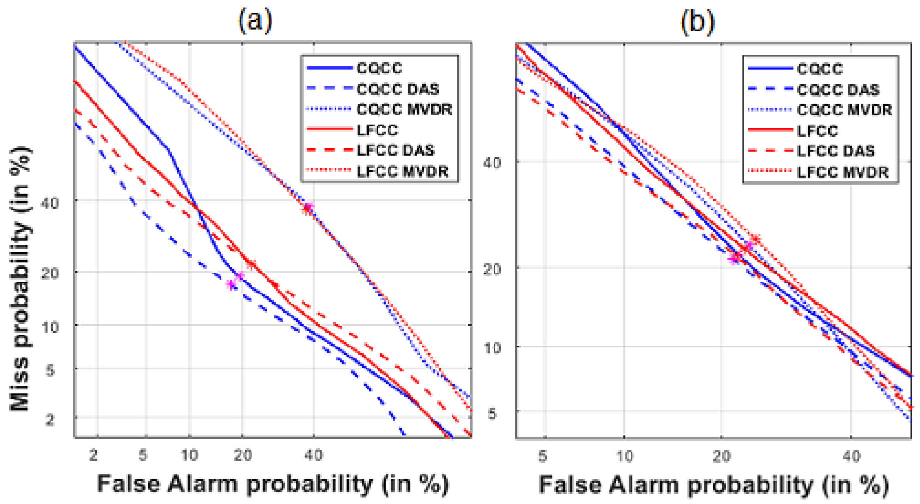


Figure 7.3: DET curves for ReMASC and its beamformed versions: (a) development set, and (b) evaluation set. After [16].

7.5 Analysis of Latency Period

In this study, we have analysed the trade between % EER and latency period (as shown in Fig. 7.4), using CQCC feature for development set of ReMASC and its beamformed version (i.e., DAS and MVDR). Here, the latency is variation of performance evaluation (in % EER) w.r.t. varying duration of an input speech signal. Hence, the variation of time duration, ranges from 20 to 2000 ms, with an interval of 200 ms. It can be observed from the Fig. 7.4 that even for short latency period, DAS is performing better than the other two approaches and hence, it shows the significance of DAS beamformer for practical SSD system deployment for voice assistants.

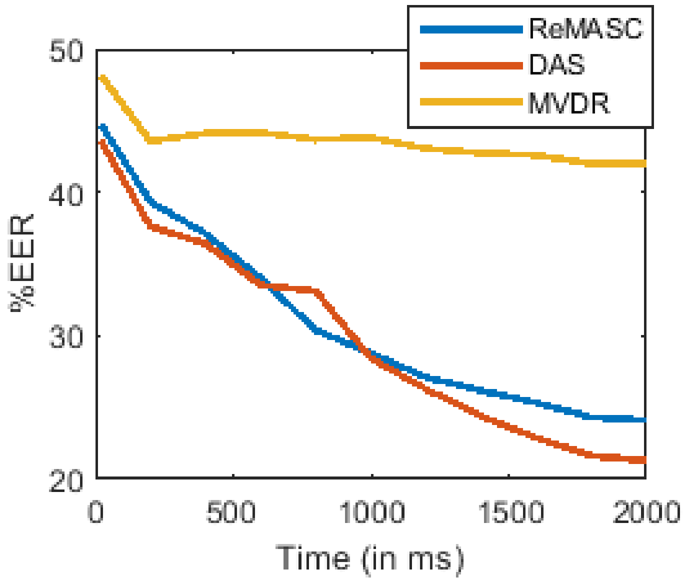


Figure 7.4: % EER *vs.* latency period using CQCC feature set for development set of ReMASC and its beamformed versions (i.e., DAS and MVDR). After [16].

7.6 Chapter Summary

In this study, the significance of the beamforming techniques for the replay SSD task for VAs is investigated. Hence, for replay CM development for VAs the experiments are performed on ReMASC, which is specially designed for this purpose. Also, four state-of-the-art feature sets, namely, CQCC, LFCC, SMCC, and MFCC are used for the anti-spoofing experiments on VAs. The improved performance is observed for the SSD systems developed by using the beamformed dataset over the original ReMASC dataset. The beamforming suppresses the noise in the speech signal. This suppressed noise might be the common for original and its beamformed versions. Hence, the discriminative acoustic cue in replay mechanism might get enhanced in the DAS beamformed dataset than the MVDR. The fact that MVDR gives relatively larger % EER than DAS beamformer indicating there is more confusion between genuine and spoofed signal, which is beamformed using MVDR than its DAS counterpart.

CHAPTER 8

Summary and Conclusions

8.1 Summary of work presented in the thesis

Data Augmentation for ASR

In this thesis, we evaluated the performance of data augmentation for children ASR using the CycleGAN model that learns a generalized mapping between source and the target class. Specifically, a group of children's voices are mapped to another (without using one-to-one mapping) based on average F_0. The experimental results show that the proposed CycleGAN-based data augmentation approach gives 3 % relative reduction in WER for children ASR. To achieve mentioned best results, various DNN architectures along with different n-gram LMs were analysed, and the most optimum model was consist of 6 CNN & 9 TDNN-F layers and 4-gram LM, respectively. The intelligibility of the CycleGAN generated speech is also verified by using the synthetic-only training data. However, voice conversion of one class of children's speech into another contains some outliers. In addition, from Fig. 2, the spectral smoothening effects can be observed in GAN-based synthetic speech spectrogram.

Improved SSL with NST Learning

In this paper, we have presented an approach to solve challenges of children ASR by proposing a novel fusion architecture of *wav2vec 2.0* and NST learning. In particular, we have tried to adapt the goodness of both the techniques, i.e., using less labelled data and unlabelled data in an iterative manner. We were able to relatively improve the results by 19.1 % WER in Dev set and 10 % WER for Test set after using LM for pseudo labels. During ASR experiments, it was experienced that the hyperparameters should be chosen carefully, because this approach is highly prone to overfitting. The use of LM is a must, such that loss transfer while training should be minimum. Further work can be extended on a larger dataset, especially on the unlabelled data, that might avoid the issue of overfitting.

Replay SSD for VAs

In this study, the significance of the beamforming techniques for the replay SSD task for VAs, is investigated. Hence, for replay CM development for VAs the experiments are performed on ReMASC, which is specially designed for this purpose. Also, four state-of-the-art feature sets, namely, CQCC, LFCC, SMCC, and MFCC are used for the anti-spoofing experiments on VAs. The improved performance is observed for the SSD systems developed by using the beamformed dataset over the original ReMASC dataset. The beamforming suppresses the noise in the speech signal. This suppressed noise might be the common for original and its beamformed versions. Hence, the discriminative acoustic cue in replay mechanism might get enhanced in the DAS beamformed dataset than the MVDR. The fact that MVDR gives relatively larger % EER than DAS beamformer indicating there is more confusion between genuine and spoofed signal, which is beamformed using MVDR than its DAS counterpart. In the other words, MVDR beamformed replay signal is very similar to its genuine counterpart, indicating overall impulse response $e(n)$ (which itself is the impulse response of four components) are being ideally nullified. This finding is in agreement with the basic design structure of MVDR beamformer.

8.2 Limitations of the Current Work

Data Augmentation for ASR

There are some outliers in audio conversion using CycleGAN architecture. This has resulted in reduction of intelligibility of converted speech by 1% compared to the original speech.

Improved Self-Supervised Learning with Noisy Student Teacher Learning

This approach is highly prone to overfitting. The use of LM is a must, such that loss transfer while training should be minimum. Further work can be extended on a larger dataset, especially on the unlabelled data, that might avoid the issue of overfitting.

Replay Spoof Speech Detection (SSD) for VAs

Only two beamforming methods were explored. While the accuracy received is not practically deployable.

8.3 Future Research Directions

Data Augmentation for ASR

Other voice conversion methods can be explored for data augmentation. While speech synthesis can also be explored using GAN with different architectures and set of feature mapping. With a decent amount of augmentation, state-of-the-art end-to-end ASR technique can also be explored. Apart from data augmentation with limited data, self-supervised learning can also be implemented utilizing *wav2vec2.0* the technique.

Improved Self-Supervised Learning with Noisy Student Teacher Learning

Other SSL approaches can also be explored instead of wav2vec 2.0. LM reduces the error of acoustic modelling, as the better the LM, the lower will be the prediction error transferred from one layer to another. In this work, a depth study of LM was not performed, thus in further work it can be focused. There are some popular RNN and LSTM based LM, while transformer based LM can also be explored.

Replay Spoof Speech Detection (SSD) for VAs

In the future, other techniques such as 3D neural networks or modified MVDR available for beamforming can be explored on microphone arrays signals for SSD task on VAs.

Publications From The Thesis

Conferences

1. **Shreya Chaturvedi**, Hemant A. Patil, Hardik B. Sailor, " Noisy Student Teacher Training with Self – Supervised Learning for Children ASR" **submitted** in IEEE International Conference on Signal Processing and Communications (SPCOM), Bengaluru, India, 2022.

2. Piyushkumar K. Chodingala, **Shreya S. Chaturvedi**, Ankur T. Patil, Hemant A. Patil, "DAS *vs.* MVDR: Which Beamformer is Suitable For Replay Attack Detection On Voice Assistants?," **submitted** to INTERSPEECH, Incheon, Korea, September 18 to 22, 2022.

3. Dipesh k. Singh, Preet Amin, Hemant A. Patil, Hardik B. Sailor, **Shreya Chaturvedi**, "Voice Conversion Based Data Augmentation Using CycleGAN for Children's ASR," **submitted** to INTERSPEECH, Incheon, Korea, September 18 to 22, 2022.

4. Piyushkumar K. Chodingala, **Shreya S. Chaturvedi**, Ankur T. Patil, Hemant A. Patil, "Robustness of DAS Beamformer Over MVDR for Replay Attack Detection On Voice Assistants," **submitted** in IEEE International Conference on Signal Processing and Communications (SPCOM), Bengaluru, India, 2022.

5. Piyushkumar K. Chodingala, Ankur T. Patil, Hemant A. Patil, **Shreya S. Chaturvedi** "Significance of DAS *vs.* MVDR Beamformer for Replay Spoof Detection On Voice Assistants," **rejected** to 30th European Signal Processing Conference (EUSIPCO), Belgrade, Serbia, August 29-September 2, 2022.

Chapter A
Setup

ASR frameworks toolkits, such as Kaldi, Espnet and Fairseq needs systematic sequence of steps to be executed for setup. The system configurations that were used were:

- OS - Ubntu 18
- Python = 3.6 or 3.9
- gcc = 9
- cuda toolkit = 11.4
- torch = 1.10.2

Note recommendation is to first create virtual environment and then do all setup into it.

A.1 Kaldi Setup

`https://kaldi-asr.org/doc/install.html` After following the above instructions, SRILM language model toolkit also needs to be setup additionally. Follow the following instructions for that:

1. Download a zip file from URL `http://www.speech.sri.com/projects/srilm/download.html`.
2. Change the name of zip of extension of "*.tar.gz" to "srilm.tar.gz"
3. Copy the file to ".../kaldi/tools"
4. open terminal at location ".../kaldi/tools" and execute this command on terminal "bash install_srilm.sh"

A.2 Espnet Setup

For installation, follow the steps mentioned at this URL https://espnet.github.io/espnet/installation.html

Note:

- Prefer to have prior kaldi setup, it will give you flexibility to use fbank features and some additional functionalities of kaldi with Espnet.
- In Step 3 of above-mentioned link, perform any one of the given option, prefer the virtual environment method.
- Make sure the CUDA path is taken properly, if there is any uncertainty related to that, in Step 4 of above-mentioned link, make sure to mention Torch and CUDA version, otherwise, it will be CPU only setup and neural networks won't run properly.

A.3 Fairseq Setup

Installation of Fairseq was successful, a part of decoding script. In the following installation Pretraining, data preparation and fine-tuning was done quite successfully. Following two URL links were used for setup:

- For Fairseq : https://github.com/facebookresearch/fairseq/issues/url
- For Python binding and wav2letter for decoding https://github.com/flashlight/wav2letter/wiki/Building-Python-bindings

A.4 HuggingFace

This is very simple, it is just like using an Application Programming Interface (API). Main library to install is "Transformers" and "jiwer" to calculate %WER . Write this commands on terminal in your created environment:

- pip install transformers=4.17.0
- pip install jiwer
- pip install numpy

There is a possibility while execution one might face issue for different library and packages, it can be simply solved by using "pip install" and googling over stack overflow is a great practice to have.

CHAPTER B
Data Preparation

B.1 Data File Preparation

In data preparation for ASR, we have to provide a proper input and output information in a document. For supervised frameworks such as Kaldi and espnet, there can be various types of files, while usually four are main as follows:

- text : <Utternce ID>   <Transcription>
- wav.scp : <utterance ID>   <Audio Path>
- utt2spk : <utterance ID>   <speaker ID>
- spk2utt : <speaker ID>   <utterance 1 ID>, <utterance 2 ID>, ...

With suitable example, it can be studied in detail from https://kaldi-asr.org/doc/data_prep.html

While for most of self-supervised ASR framework, only one file is required, where every row will have unique audio information, and each row will consists of "<audio path>   <transcription>"

B.2 Flac to Wav File Conversion

Flac files are usually slow and are not compatible in all the frameworks, while wav format files are the most commonly used format. Hence, in case of any complication related to type of audio file and such uncertainty, "flac" or other formats should be converted to "wav format". **Do not prefer Python-based conversion or rewrite method for audio format change.** As it might just change the extension or disrupt the audio file. Hence, use SOX command for such task. "sox infile.flac outfile.wav" [1]

[1]For more reference of codes made by me look to my GitHub repositary https://github.com/Pixeliate?tab=repositories

CHAPTER C

Voice Activity Detection (VAD)

Due to computational limitation, long audio cannot be processed. Thus, this method was explored of Voice Activity Detection (VAD). It is the energy-based classification of speech, where every segment is segregated in category of either speech or non-speech audio. While this can be only useful where we want to only work with audio but not the transcription. Because just based on this information, transcription cannot be chunked w.r.t. audio. While in Kaldi due to force alignment using HMM, the segmentation of audio is possible with their respective transcription, however this process also reduces the performance to a certain extent.

By implementing this, we will receive a True(1) or False(0) label w.r.t. every sample or frame. Some initial attempts are mentioned in the following code. While I have noted that the sample-based VAD might work for similar environment and the same speaker, but not in general. That's why energy-based (frame) dependent seems to be more useful. Code of VAD is provided in this repository https://github.com/Pixeliate/Voice-Activity-Detection-Sample-based-.git

```
#!/usr/bin/bash
import collections
import contextlib
import sys
import wave

import webrtcvad


def read_wave(path):

    with contextlib.closing(wave.open(path, 'rb')) as wf:
        num_channels = wf.getnchannels()
        assert num_channels == 1
        sample_width = wf.getsampwidth()
        assert sample_width == 2
        sample_rate = wf.getframerate()
        assert sample_rate in (8000, 16000, 32000, 48000)
        pcm_data = wf.readframes(wf.getnframes())
        return pcm_data, sample_rate


def write_wave(path, audio, sample_rate):

    with contextlib.closing(wave.open(path, 'wb')) as wf:
        wf.setnchannels(1)
        wf.setsampwidth(2)
        wf.setframerate(sample_rate)
        wf.writeframes(audio)


class Frame(object):

    def __init__(self, bytes, timestamp, duration):
        self.bytes = bytes
        self.timestamp = timestamp
        self.duration = duration


def frame_generator(frame_duration_ms, audio, sample_rate):
    n = int(sample_rate * (frame_duration_ms / 1000.0) * 2)
    offset = 0
    timestamp = 0.0
    duration = (float(n) / sample_rate) / 2.0
    while offset + n < len(audio):
        yield Frame(audio[offset:offset + n], timestamp, duration)
        timestamp += duration
        offset += n
```

```
def vad_collector(sample_rate, frame_duration_ms,
                  padding_duration_ms, vad, frames):
    #num_padding_frames = int(padding_duration_ms / frame_duration_ms)

    for frame in frames:
        is_speech = vad.is_speech(frame.bytes, sample_rate)

        sys.stdout.write('1' if is_speech else '0')


sample = "/home/speechlab/Desktop/Shreya/TLT_2021/ETLT2021_CAMBRIDGE_EN_baseline/
audio, sample_rate = read_wave(sample)
vad = webrtcvad.Vad(int(1))
frames = frame_generator(30, audio, sample_rate)
frames = list(frames)
segments = vad_collector(sample_rate, 30, 300, vad, frames)
print(segments , "\n ============================================\n")
```

CHAPTER D
Baseline for Pretraining + Finetuning using Fairseq

```
#!/usr/bin/bash
##########################################################################
#  THIS SCRIPT TO BE RUN ON Folder: self-supervsed-speech-recognition   #
##########################################################################
flac_to_wav=true
partition=true

stage=3

raw_set="/Desktop/Shreya/myst-v0.4.2/data/test/test_Trans_wav.txt"
finetune_set="/home/priyankag/ssl/baseline_trial/test_finetune.txt"
test_set="/home/priyankag/ssl/baseline_trial/test_test.txt"
data_dir="data"

if [ $stage -eq 1 ] ; then
  echo "stage 1 : Data Preparation"
  for part in finetune test ; do
     mkdir -p baseline_trial/data/$part
     python3 baseline_trial/data_prep.py $flac_to_wav $raw_set...
             ...$part $partition
  done
  #stage=2
fi


if [ $stage -eq 2 ] ; then
```

```
echo "stage 2 : Create Dictionary "
python3 gen_dict.py
    --transcript_file baseline_trial/data/finetune/finetune_set.txt
    --save_dir baseline_trial/dictionary/
#stage=3
fi

if [ $stage -eq 3 ] ; then
echo "stage 3 : FineTuning on pretrained model"
#python3 finetune.py
    --transcript_file path/to/transcript.txt
    --pretrain_model path/to/pretrain_checkpoint_best.pt
    --dict_file path/to/dict.ltr.txt
python3 finetune.py
    --transcript_file baseline_trial/data/finetune/finetune_set.txt
    --pretrain_model baseline_trial/Pre-Trained_model/430_checkpoint.pt
    --dict_file baseline_trial/dictionary/dict.ltr.txt
fi
```

References

[1] *HuggingFace ASR based transformer*, 2022. {Last Accessed : March 2022}.

[2] *HuggingFace wav2vec 2.0 based Pretrained model*, 2022 (Last Accessed March 30, 2022).

[3] F. Alegre, A. Janicki, and N. Evans. Re-assessing the threat of replay spoofing attacks against automatic speaker verification. In *International Conference of the Biometrics Special Interest Group (BIOSIG)*, pages 1–6, Darmstadt, Germany, 10-12 September, 2014.

[4] E. Alsharhan and A. Ramsay. Investigating the effects of gender, dialect, and training size on the performance of arabic speech recognition. *Language Resources and Evaluation*, 54(4):975–998, 2020.

[5] R. F. Astudillo and J. P. d. S. Neto. Propagation of uncertainty through multilayer perceptrons for robust automatic speech recognition. In *INTERSPEECH, Denver, Colorado, USA*, pages 461–464, 2011.

[6] A. Baevski, W.-N. Hsu, A. Conneau, and M. Auli. Unsupervised speech recognition. *Advances in Neural Information Processing Systems (NIPS)*, 34, December 2021.

[7] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems (NIPS)*, 33:12449–12460, 2020.

[8] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. {Last Accessed : 15-04-2022}.

[9] J. R. Bellegarda and C. Monz. State of the art in statistical methods for language and speech processing. *Computer Speech & Language*, 35:163–184, 2016.

[10] L. G. Bezanson. *The Subarray MVDR Beamformer: A Space-Time Adaptive Processor Applied to Active Sonar*. University of California, San Diego, 2013.

[11] A. Bhattad, A. Dundar, G. Liu, A. Tao, and B. Catanzaro. View generalization for single image textured 3d models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6081–6090, Nashville, TN, USA, 2021.

[12] H. Bourlard and N. Morgan. Continuous speech recognition by connectionist statistical methods. *IEEE Transactions on Neural Networks*, 4(6):893–909, 1993.

[13] M. Brandstein and D. Ward. *Microphone Arrays: Signal Processing Techniques and Applications*. Springer Science & Business Media, 2001.

[14] J. Capon. High-resolution frequency-wavenumber spectrum analysis. *Proceedings of the IEEE*, 57(8):1408–1418, 1969.

[15] S. S. Chaturevedi, H. B. Sailor, and H. A. Patil. Noisy student teacher training with self supervised learning for children asr. ***submitted** to International Conference on Signal Processing and Communications (SPCOM), iisC Bengaluru, India*, 11-15 July 2022.

[16] P. K. Chodingala, A. T. Patil, H. A. Patil, and S. S. Chaturvedi. Significance of das *vs.* mvdr beamformer for replay spoof detection on voice assistants. ***rejected** in* 30th *European Signal Processing Conference (EUSIPCO), Belgrade, Serbia*, August 29-September 2, 2022.

[17] Y.-A. Chung, Y. Zhang, W. Han, C.-C. Chiu, J. Qin, R. Pang, and Y. Wu. W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. *arXiv preprint arXiv:2108.06209*, 2021.

[18] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 28(4):357–366, 1980.

[19] H. Delgado, M. Todisco, M. Sahidullah, N. Evans, T. Kinnunen, K. A. Lee, and J. Yamagishi. ASVSpoof 2017 Version 2.0: Meta-data analysis and baseline enhancements. *Odyssey- The Speaker and Language Recognition Workshop, Les Sables d'Olonne, France*, page 296–303, 26-29 June 2018.

[20] S. Doclo, W. Kellermann, S. Makino, and S. E. Nordholm. Multichannel signal enhancement algorithms for assisted listening devices: Exploiting spatial diversity using multiple microphones. *IEEE Signal Processing Magazine*, 32(2):18–30, 2015.

[21] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

[22] M. Gales and S. Young. *The application of hidden Markov models in speech recognition*. Now Publishers Inc, 2008.

[23] M. Gerosa, D. Giuliani, and S. Narayanan. Acoustic analysis and automatic recognition of spontaneous children's speech. In *Ninth International Conference on Spoken Language Processing*, pages 1082–1086, 2006, Pittsburgh, PA, USA.

[24] S. Ghai. Addressing pitch mismatch for children's automatic speech recognition. *Ph. D. Thesis, Department of Electronics and Electrical Engineering, Indian Institute of Technology (IIT) Guwahati, India*, 2011.

[25] I. Goodfellow. NIPS 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2017. {Last Accessed: 15-01-2021}.

[26] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in Neural Information Processing Systems (NIPS)*, 27, 2014.

[27] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[28] R. Gretter, M. Matassoni, D. Falavigna, A. Misra, C. W. Leong, K. Knill, and L. Wang. ETLT 2021: Shared task on automatic speech recognition for non-native children's speech. pages 3845–3849, Brno, Czech Republic, INTERSPEECH 2021.

[29] E. A. Habets, J. Benesty, S. Gannot, and I. Cohen. The MVDR beamformer for speech enhancement. In *Speech Processing in Modern Communication*, pages 225–254. Springer, 2010.

[30] W.-N. Hsu, Y.-H. H. Tsai, B. Bolte, R. Salakhutdinov, and A. Mohamed. Hubert: How much can a bad teacher benefit asr pre-training? In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6533–6537. IEEE, 2021.

[31] J. Jeong, S. I. M. M. R. Mondol, Y.-W. Kim, and S. Lee. An effective learning method for automatic speech recognition in korean ci patients' speech. *Electronics*, 10:807, 03 2021.

[32] D. H. Johnson and D. E. Dudgeon. *Array Signal Processing: Concepts and Techniques*. Simon & Schuster, Inc., 1992.

[33] T. Kaneko, H. Kameoka, K. Tanaka, and N. Hojo. CycleGAN-VC2: Improved CycleGAN-based non-parallel voice conversion. In *The International Conference on Acoustics, Speech, Signal Processing (ICASSP)*, pages 6820–6824, Brighton, UK, 2019.

[34] M. Karaman, P.-C. Li, and M. O'Donnell. Synthetic aperture imaging for small scale systems. *IEEE Transactions on Ultrasonic, Ferroelectrics, and Frequency Control*, 42(3):429–442, 1995.

[35] S. Kim, T. Hori, and S. Watanabe. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA*, pages 4835–4839, 2017.

[36] I. Kipyatkova and A. Karpov. Dnn-based acoustic modeling for russian speech recognition using kaldi. In *International Conference on Speech and Computer*, pages 246–253. Springer, 2016.

[37] F. Lago, C. Pasquini, R. Böhme, H. Dumont, V. Goffaux, and G. Boato. More real than real: A study on human visual perception of synthetic faces. *IEEE Signal Processing Magazine*, 39(1):109–116, 2021.

[38] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[39] S. Lee, A. Potamianos, and S. Narayanan. Acoustics of children's speech: Developmental changes of temporal and spectral parameters. *The Journal of the Acoustical Society of America (JASA)*, 105(3):1455–1468, 1999.

[40] Q. Li and M. J. Russell. An analysis of the causes of increased error rates in children's speech recognition. In *Seventh International Conference on Spoken Language Processing*, pages 2337–2340, 2002, Denver, Colorado, USA.

[41] Q. Li and M. J. Russell. Why is automatic recognition of children's speech difficult? In *Seventh European Conference on Speech Communication and Technology*, pages 2671–2674, Aalborg Congress and Culture Centre, Aalborg, Denmark, 2001.

[42] X. Li, N. Dong, J. Huang, L. Zhuo, and J. Li. A discriminative self-attention cycle gan for face super-resolution and recognition. *IET Image Processing*, 15(11):2614–2628, 2021.

[43] H. Liao, G. Pundak, O. Siohan, M. K. Carroll, N. Coccaro, Q. Jiang, T. N. Sainath, A. W. Senior, F. Beaufays, and M. Bacchiani. Large vocabulary automatic speech recognition for children. In *INTERSPEECH , Dresden, Germany, September 6-10, 2015*, pages 1611–1615.

[44] S. Loria et al. textblob documentation. *Release 0.15*, 2:269, 2018.

[45] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley. Least squares Generative Adversarial Networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2794–2802, Venice, Italy, 2017.

[46] M. Matassoni, R. Gretter, D. Falavigna, and D. Giuliani. Non-native children speech recognition through transfer learning. In *The International Conference on Acoustics, Speech, Signal Processing (ICASSP)*, pages 6229–6233, Calgary, AB, Canada, 2018.

[47] P. McGrath, J. Goodman, S. Cunningham, B. MacDonald, T. Nichols, and A. Unruh. Assistive devices: utilization by children. *Archives of Physical Medicine and rehabilitation*, 66(7):430–432, 1985.

[48] A. Meyer, D. Döbler, J. Hambrecht, and M. Matern. Acoustic mapping on three-dimensional models. In *Proceedings of the* 12^{th} *International Conference on Computer Systems and Technologies*, pages 216–220, Vienna, Austria, 2011.

[49] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Cernocky. Rnnlm-recurrent neural network language modeling toolkit. In *Proc. of the 2011 ASRU Workshop*, pages 196–201, 2011.

[50] M. Morise, F. Yokomori, and K. Ozawa. WORLD: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE Trans. on Information and Systems*, 99(7):1877–1884, 2016.

[51] A. Nautsch, X. Wang, N. Evans, T. H. Kinnunen, V. Vestman, M. Todisco, H. Delgado, M. Sahidullah, J. Yamagishi, and K. A. Lee. ASVSpoof 2019: spoofing countermeasures for the detection of synthesized, converted, and replayed speech. *IEEE Trans. on Biometrics, Behavior, and Identity Science*, 3(2):252–265, 2021.

[52] J. R. Novak, N. Minematsu, and K. Hirose. Phonetisaurus: Exploring grapheme-to-phoneme conversion with joint n-gram models in the WFST framework. *Natural Language Engineering*, 22(6):907–938, 2016.

[53] W. Ocasio. Attention to attention. *Organization science*, 22(5):1286–1296, 2011.

[54] A. V. Oppenheim. Speech spectrograms using the fast fourier transform. *IEEE Spectrum*, 7(8):57–62, 1970.

[55] A. V. Oppenheim, R. W. Schafer, and T. Stockham. Nonlinear filtering of multiplied and convolved signals. *IEEE Transactions on Audio and Electroacoustics*, 16(3):437–466, 1968.

[56] A. Ortiz, C. Robinson, D. Morris, O. Fuentes, C. Kiekintveld, M. M. Hassan, and N. Jojic. Local context normalization: Revisiting local normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11276–11285, 2020.

[57] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le. SpecAugment: a simple data augmentation method for automatic speech recognition. In *INTERSPEECH*, pages 2613–2617, Graz, Austria, 2019.

[58] D. S. Park, Y. Zhang, Y. Jia, W. Han, C.-C. Chiu, B. Li, Y. Wu, and Q. V. Le. Improved noisy student training for automatic speech recognition. *arXiv preprint arXiv:2005.09629*, 2020. {Last Accessed : 14-04-2022}.

[59] H.-J. Park, P. Zhu, I. L. Moreno, and N. Subrahmanya. Noisy student-teacher training for robust keyword spotting. *arXiv preprint arXiv:2106.01604*, 2021.

[60] A. Potamianos and S. Narayanan. Robust recognition of children's speech. *IEEE Transactions on Speech and Audio Processing*, 11(6):603–616, 2003.

[61] D. Povey, G. Cheng, Y. Wang, K. Li, H. Xu, M. Yarmohammadi, and S. Khudanpur. Semi-orthogonal low-rank matrix factorization for deep neural networks. In *INTERSPEECH*, pages 3743–3747, Hyderabad, India, 2018.

[62] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, et al. The kaldi speech recognition toolkit. In *Automatic Speech Recognition and Understanding (ASRU)*, Waikoloa, HI, USA, 2011.

[63] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur. Purely sequence-trained neural networks for ASR based on lattice-free MMI. pages 2751–2755, San Francisco, USA, INTERSPEECH 2016.

[64] M. Qian, I. McLoughlin, W. Quo, and L. Dai. Mismatched training data enhancement for automatic recognition of children's speech using DNN-HMM. In *International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5, Calgary, Alberta, Canada, 2016.

[65] T. F. Quatieri. *Discrete-Time Speech Signal Processing: Principles and Practice*. Pearson Education, 3rd edition, India, 2006.

[66] L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*. Prentice-Hall, Inc., 1993.

[67] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[68] S. Shahnawazuddin, N. Adiga, K. Kumar, A. Poddar, and W. Ahmad. Voice conversion based data augmentation to improve children's speech recognition in limited data scenario. *INTERSPEECH,* Shanghai, China, pages 4382–4386, 2020.

[69] P. G. Shivakumar and P. Georgiou. Transfer learning from adult to children for speech recognition: Evaluation, analysis and recommendations. *Computer Speech & Language*, 63:101077, 2020.

[70] P. G. Shivakumar, A. Potamianos, S. Lee, and S. S. Narayanan. Improving speech recognition for children using acoustic adaptation and pronunciation modeling. In *Workshop on Child Computer Interaction (WOCCI)*, pages 15–19, Singapore, 2014.

[71] D. K. Singh, P. P. Amin, H. B. Sailor, H. A. Patil, and S. S. Chaturevedi. Voice conversion based data augmentation using cyclegan for children's asr. ***submitted*** *to INTERSPEECH, Incheon, South Korea,*, 18-22 September 2022.

[72] O. M. Strand and A. Egeberg. Cepstral mean and variance normalization in the model domain. In *COST278 and ISCA Tutorial and Research Workshop (ITRW) on Robustness Issues in Conversational Interaction*, Minneapolis, MN, USA, 2004.

[73] M. Sundermeyer, R. Schlüter, and H. Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.

[74] D. Svozil, V. Kvasnicka, and J. Pospichal. Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems*, 39(1):43–62, 1997.

[75] M. Todisco, H. Delgado, and N. Evans. Constant q cepstral coefficients: A spoofing countermeasure for automatic speaker verification. *Computer Speech & Language*, 45:516–535, 2017.

[76] M. Todisco, X. Wang, V. Vestman, M. Sahidullah, H. Delgado, A. Nautsch, J. Yamagishi, N. Evans, T. H. Kinnunen, and K. A. Lee. ASVspoof 2019: Future Horizons in Spoofed and Fake Audio Detection. In *INTERSPEECH, Graz, Austria*, pages 1008–1012, Sep. 15-19, 2019.

[77] Y.-H. Tu, J. Du, and C.-H. Lee. Speech enhancement based on teacher–student deep learning using improved speech presence probability for noise-robust speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(12):2080–2091, 2019.

[78] A. Van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807, 2018.

[79] H. L. Van Trees. *Optimum Array Processing: Part IV of Detection, Estimation, and Modulation Theory*. John Wiley & Sons, 2004.

[80] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[81] R. Vergin, D. O'Shaughnessy, and A. Farhat. Generalized mel frequency cepstral coefficients for large-vocabulary speaker-independent continuous-speech recognition. In *IEEE Transactions on Speech and Audio Processing*, volume 7, pages 525–532, 1999.

[82] D. Wang, X. Wang, and S. Lv. An overview of end-to-end automatic speech recognition. *Symmetry*, 11(8):1018, 2019.

[83] Y. Wang, X. Deng, S. Pu, and Z. Huang. Residual convolutional CTC networks for automatic speech recognition. *arXiv preprint arXiv:1702.07793*, 2017. {Last Accessed : 15-04-2022}.

[84] Z. Wang, R. Giri, U. Isik, J.-M. Valin, and A. Krishnaswamy. Semi-supervised singing voice separation with noisy self-training. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, Ontario, Canada*, pages 31–35, 2021.

[85] M. Wölfel and J. McDonough. *Distant Speech Recognition*. John Wiley & Sons, 2009.

[86] Z. Wu, N. Evans, T. Kinnunen, J. Yamagishi, F. Alegre, and H. Li. Spoofing and countermeasures for speaker verification: A survey. *Speech Communication*, 66:130–153, 2015.

[87] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020.

[88] Z. Xie, G. Zhang, C. Yu, J. Zheng, and B. Sheng. CFMNet: Coarse-to-fine cascaded feature mapping network for hair attribute transfer. In *Computer Graphics International Conference*, pages 406–417, Geneva, Switzerland, 2021. Springer.

[89] G. Xu, S. Yang, L. Ma, C. Li, and Z. Wu. The tal system for the interspeech2021 shared task on automatic speech recognition for non-native childrens speech. *INTERSPEECH*, pages 1294–1298, 2021.

[90] G. Xu, S. Yang, L. Ma, C. Li, and Z. Wu. The TAL system for the INTERSPEECH2021 shared task on automatic speech recognition for non-native childrens speech. In *Proc. INTERSPEECH*, pages 1294–1298, Brno, Czech Republic, 2021.

[91] T. Yin and J. Yang. Detection of steel surface defect based on faster r-cnn and fpn. In *2021 7th International Conference on Computing and Artificial Intelligence*, pages 15–20, Tianjin, China, 2021.

[92] Y. Zhang, J. Qin, D. S. Park, W. Han, C.-C. Chiu, R. Pang, Q. V. Le, and Y. Wu. Pushing the limits of semi-supervised learning for automatic speech recognition. *arXiv preprint arXiv:2010.10504*, 2020.