

Towards Building an Automatic Speech Recognition Systems in Indian Context using Deep Learning

Thesis submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

in

Electronics and Communication Engineering

by

Sai Ganesh Mirishkar

2018802003

mirishkar.ganesh@research.iiit.ac.in



International Institute of Information Technology, Hyderabad

(Deemed to be University)

Hyderabad - 500 032, INDIA

November 2023

Copyright © Sai Ganesh Mirishkar, 2023

All Rights Reserved

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled “**Towards Building an Automatic Speech Recognition Systems in Indian Context using Deep Learning**” by Sai Ganesh Mirishkar, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Advisor: Dr. Anil Kumar Vuppala

I would like to dedicate this thesis to my parents



Sri. Narasinga Rao & Smt. Saraswathi

My guide

Dr. Anil Kumar Vuppala

&

My Family, Teachers, Friends, and Well-wishers

Acknowledgments

I would like to convey my profound admiration and heartfelt appreciation to my mentor, Dr. Anil Kumar Vuppala, for his priceless counsel, motivation, and support throughout my research journey. I am deeply grateful that he took me under his wing as a student. I also appreciate the personal attention and the precious time he invested in determining my research topic. His dedication and work ethic have continually inspired me to persevere in the realm of research. I am genuinely grateful for the outstanding research atmosphere he has fostered, enabling students to learn and engage in research with total autonomy. I also express my gratitude for his guidance in shaping my future professional path.

I am immensely grateful to Prof. Yegnanarayana Bayya, an extraordinary educator and researcher, whose profound impact on my academic journey cannot be overstated. His unwavering dedication to teaching and research has served as a powerful catalyst in my growth. Attending his enlightening lectures on speech signal processing has been an absolute privilege, an experience that has left an indelible mark on my intellectual development. Despite his numerous groundbreaking research achievements, it is his exceptional teaching style that continues to resonate with me, igniting a lifelong passion for learning. Moreover, I express my sincere appreciation for the invaluable discussions we have shared on various occasions, which have further enriched my understanding of the subject.

With deep appreciation, I extend my gratitude to the remarkable researchers, both India and abroad, whose support has been instrumental in my journey. Firstly, I would like to express my sincere thanks to Prof. Hema Murty and Anusha Prakash for generously sharing their Common Phone Set lexicon source codes. Their contribution has been invaluable to my work. I am also immensely grateful to Prof. Umesh and Vasista Lodogala for their invaluable assistance in rectifying anomalies encountered during the collection of the Telugu Speech corpus. Additionally, I owe a special debt of gratitude to Prof. Umesh for providing me with invaluable feedback

during my Proposal presentation, which has greatly influenced the direction of my research. His insights and guidance have been transformative in shaping my work.

I am profoundly indebted to the esteemed individuals who have played pivotal roles in shaping my research journey: Prof. Rajeev Singal, Prof. Dipti Misra Sharma, Dr. Radhika Mamidi, Dr. Vineet Gandhi, Dr. Manish Shrivastava, Prof. Jayanthi Sivaswamy, Dr. Vinoo Alluri, Dr. Santosh Nannuru, and Dr. Chiranjeevi Yarra. Their sage advice and unwavering encouragement have been instrumental in my pursuit of knowledge. Each of them has imparted invaluable insights and expertise, enriching my research in profound ways. I am sincerely grateful for their guidance, which has propelled me towards greater heights in my academic endeavors.

Immersed in the vibrant atmosphere of the speech processing laboratory, I stand in awe of the exceptional individuals whose unwavering dedication has shaped my journey. It is with profound gratitude that I extend my heartfelt thanks to Dr. Sudarsana Reddy Kadiri, Dr. Gangamohan, Dr. Vishala Pannala, and Dr. Hari Krishna Vydana for their invaluable contributions. Their timely advice and unwavering guidance have been pivotal in my growth. Their profound expertise has not only illuminated my path but also fueled my determination to achieve excellence. I am truly indebted to them for their remarkable support and unwavering commitment to my success.

With great reverence, I express my profound gratitude to an exceptional group of individuals - Aditya, Shelly Jain, Krishna, Haala, Vishnu, Ravi Garu, Purva Madam, Ruchi Pandey, Pruthwik Mishra, Vandan Mujadia, Saumitra Yadav, Annaya Mukherjee, Palash, Vamshi, Anuprabha Madam, Priyanka Madam, and Keshav Raj. Their unwavering presence has fostered an incredibly warm and inviting atmosphere within and beyond the confines of the lab. I hold dear the cherished moments of collective learning and joy we have shared, and I extend my heartfelt thanks to each and every one of them. Their cooperation, understanding, and assistance have been invaluable to me on this journey, and I am truly grateful for their unwavering support.

I wish to acknowledge and express my heartfelt appreciation to the remarkable administrative staff at IIIT. Prathima Garu, Appaji Garu, Pushpalatha Garu, Karthik Garu, and G Srinivas Garu have been invaluable pillars of support throughout my tenure. Their unwavering dedication, relentless efforts, and selfless assistance have played a pivotal role in enhancing my experience at IIIT. I am truly indebted to their remarkable professionalism, guidance, and unwavering

commitment to facilitating a conducive environment for learning and growth. Their invaluable contributions have been instrumental in shaping my journey, and I am sincerely grateful for their exceptional support.

With utmost gratitude, I extend my sincere thanks to IITH-IHUB for their unwavering support in the form of a fellowship throughout my PhD journey. Their generous assistance has been instrumental in fueling my research endeavors and enabling me to make significant strides in my field. Furthermore, I express my heartfelt appreciation to IEEE and ISCA for their invaluable travel grant support, which has granted me the opportunity to attend and actively contribute to numerous conferences. Their backing has not only broadened my knowledge but also allowed me to forge meaningful connections with experts in my domain. I am truly indebted to these organizations for their unwavering encouragement and assistance, which have been pivotal in shaping the trajectory of my academic pursuits.

Initiating with utmost gratitude, I humbly acknowledge the immeasurable debt I owe to my beloved parents, M Narasinga Rao and M Saraswathi. Their unwavering love, unwavering support, and boundless sacrifices have been the foundation of my journey. Equally deserving of my heartfelt gratitude are my sisters Subhashini and Gauthami, my brother-in-law Subba Rao, and Hari Krishna, along with my niece Piyushi and Haruhi. Their unwavering support in every aspect of my life has been nothing short of remarkable.

I reserve a special place in my heart for my wife, Mrs. Dr. Ketaki, and our little son, Sai Divij Bhushan. Their unwavering understanding and acceptance of my late nights in the lab is a testament to their exceptional patience and support. I am profoundly grateful to my wife for her remarkable tolerance and unwavering belief in my aspirations.

Additionally, I extend my heartfelt appreciation to all my friends and teachers who have played an instrumental role in shaping my journey thus far. Their guidance, encouragement, and unwavering belief in my potential have propelled me to this significant stage in life. It is with deep reverence that I dedicate this thesis to all my esteemed teachers and, in particular, to my mentor and guide, Dr. Anil Kumar Vuppala. Their invaluable insights and unwavering support have been instrumental in my growth and academic achievements.

-Sai Ganesh Mirishkar

Abstract

Automatic Speech Recognition (ASR) systems are increasingly prevalent in our daily lives, with commercial applications such as Siri, Alexa, and Google Assistant. However, the focus of these systems has been largely angled towards English, leaving a considerable portion of non-English speakers underserved. This is particularly evident in India, a linguistically diverse country with many languages classified as low-resource in the context of ASR due to the scarcity of annotated speech data. This thesis aims to bridge this gap, focusing on enhancing ASR systems for Indian languages using deep learning methodologies. India is a land of language diversity. There are approximately 2000 languages spoken around, and among those officially registered are 23. Of those, very few have ASR capability. This is because building an ASR system requires thousands of hours of annotated speech data, a vast amount of text, and a lexicon that can span all the words in the languages. The necessity for a comprehensive presence in the diverse Indian markets demands the development of multilingual Automatic Speech Recognition (ASR) systems. It's a common scenario where ASR systems for Indian languages have to be implemented in low-resourced contexts. Furthermore, the complexity of the linguistic landscape is amplified due to the high prevalence of bilingualism in the Indian population, leading to frequent instances of code-switching and linguistic borrowing between languages. Operating concurrent ASR systems that can handle code-switching in the Indian context presents a considerable challenge. This predicament has spurred our research endeavors, driving us to focus on constructing a large corpus for one language and leveraging its phonetic space on other language families in monolingual and multilingual ASR scenarios.

This thesis incorporates a crowd-sourcing strategy to collect an extensive speech corpus, particularly for Telugu. Using this approach, around 2000 hours of Telugu speech data, capturing regional variations through three modes: spontaneous, conversational, and read, under various

background conditions, has been collected. This data served as a foundation for developing and evaluating neural network architectures tailored to the characteristics of Indian languages.

We also explored the potential of self-supervised learning to understand and enhance learned representations, fine-tuning them to suit different language families and data sizes. This approach led to insights into the shared phonetic space among Indian languages, allowing for the development of a multilingual ASR system utilizing a joint acoustic model approach. These studies mark a significant stride towards overcoming the challenges of multilingualism in the Indian context, setting a path for creating more inclusive and effective ASR systems.

The research findings presented in this thesis not only contribute towards building efficient and accurate ASR systems for low-resource Indian languages but also underscore the power of deep learning approaches in linguistic technology. It is our hope that this work will motivate and aid further research in this direction, promoting linguistic diversity and broadening access to information and communication technologies for speakers of low-resource languages.

Keywords: *Automatic Speech Recognition, Indian Languages, Crowd-Sourced, Low-Resource, Spontaneous Speech, Conversational Speech, Read Speech, Word Error Rate, Self-Supervised Learning, Common Phone Set, Common Label Set, Joint Acoustic Model, Time-Delay Neural Network, Transformer, Conformer, Wav2Vec.*

Contents

Chapter	Page
1 Introduction	1
1.1 Scope of the thesis	5
1.2 Organization of the thesis	5
2 Background and Related Work	8
2.1 Automatic Speech Recognition	9
2.1.1 Problem Formulation	10
2.1.2 End-to-End ASR	17
2.1.3 Semi-Supervised ASR	21
2.2 Overview of Neural Networks for Speech Processing	22
2.2.1 Recurrent Neural Networks	22
2.2.2 Self-Attention Networks	23
2.2.3 Convolution-Augmented Self-Attention Networks	25
2.3 Neural Network Pre-Training: A Pathway to Self-Supervised Learning	27
2.3.1 Background	27
2.3.2 Self-Supervised Learning Approaches for Speech Representation	28
2.4 Past works in Multilingual speech recognition	30
2.5 Challenges in building ASR systems for Indian Languages	31
2.6 Databases	36
2.6.1 Multilingual and code-switching ASR challenges for low-resource Indian languages (MUCS) Corpus	36
2.6.2 MILE Corpus (Kannada,Tamil)	37
2.7 Summary and Conclusion	37
3 Crowd-sourced strategies for the collection of a large-scale speech corpus	39
3.1 Introduction	39
3.2 Overview of Telugu Language	41
3.3 Proposed data acquisition workflow for speech corpus collection	43
3.4 Guidelines followed throughout the data collection process	46
3.4.1 Audio Guidelines	46
3.4.2 Transcription Guidelines (see Figure 3.5)	46
3.5 Approaches in IIITH-Crowdsourced Telugu Data corpus	48
3.5.1 Approach 1: Spontaneous mode speech (Collection through web and mobile applications from the crowd)	49

3.5.2	Approach 2: Conversational mode Speech (Data pooling through freely available resources)	53
3.5.2.1	Mobile-based support for Transcription correction	55
3.5.2.2	Web-based support for Transcription correction:	55
3.5.3	Approach 3: Data pooling through WhatsApp campaigns	56
3.5.3.1	Telugu text corpus	57
3.5.3.2	Audio Verification process	60
3.5.3.3	Payments for Verification	60
3.6	Implementation Considerations and Quality Analysis	61
3.7	CSTD corpus statistics	62
3.8	Summary & Conclusion	64
4	Investigation of Hybrid & End-to-End Speech Recognition systems on CSTD database	66
4.1	Introduction	66
4.2	Hybrid ASR baseline	67
4.2.1	Front-end features:	67
4.2.2	Acoustic model training	67
4.2.2.1	GMM modeling	67
4.2.2.2	DNN modeling	68
4.2.2.3	TDNN models	69
4.2.3	Language model training	70
4.3	End-to-End ASR baseline	70
4.3.1	Evaluation Results & Discussion	71
4.3.1.1	Hybrid ASR baseline results	73
4.3.1.2	End-to-End models baseline results	73
4.4	Ablation studies on IITH-CSTD corpus	74
4.4.1	Different Speaking Styles	74
4.4.2	Different dataset sizes	75
4.4.3	Cross-dataset benchmarking of speech recognition task	77
4.5	Summary & Conclusion	78
5	Exploration of self-supervised based approach for improving the performance of speech recognition system on CSTD corpus	80
5.1	Introduction	80
5.1.1	Data pre-processing	81
5.2	Telugu Wav2Vec pretrained model	82
5.2.1	Model Architecture for Pre-training	83
5.2.2	Contrastive Objective and Codebook Diversity During Model Pre-training	85
5.3	Pretraining Experimental Setup	88
5.3.1	Layer-Wise Analysis of CSTD-5k Pretrained Model	89
5.4	Summary & Conclusion	91

6	Building ASR Systems for Low-Resource Languages of the Indian Subcontinent Using Telugu Pretrained Model	92
6.1	Introduction	92
6.2	Model Architecture for Fine-tuning	93
6.2.1	Decoding	94
6.2.2	Rescoring	94
6.3	Methodology and Experimentation	95
6.3.1	Procedure for Fine-tuning	95
6.3.2	Language Model Setup	96
6.3.3	Rescoring Setup	97
6.4	Results and Analysis	97
6.4.1	Performance of Telugu Pretrained Model Across different language Families	97
6.4.2	Impact of Multilingual Fine-tuning on the CSTD-5k Telugu Pretrained Model	101
6.4.3	Impact of Dataset Size on Pretrained Model Performance in Speech Recognition Tasks	102
6.5	Summary & Conclusion	103
7	Exploring the Shared Phonetic Space of Indian Languages for Multilingual Speech Recognition Systems	106
7.1	Introduction	106
7.1.1	Comparison of Common Phone Set vs. Common Label Set	108
7.2	Proposed Work	112
7.3	Results & Analysis	114
7.3.1	Comparison of CPS Vs. CLS	114
7.3.2	Experimental results on different variants of TDNN architecture for both monolingual and multilingual ASR systems	116
7.3.2.1	Monolingual Experimental Results	116
7.3.2.2	Multilingual Experimental Results	117
7.3.3	Experimental Results on proposed architecture	119
7.4	Summary & Conclusion	121
8	Summary and Conclusions	122
8.1	Summary	122
8.2	Contributions in the thesis	123
8.3	Directions for Future work	124
	Bibliography	128

List of Figures

Figure	Page
1.1 Acoustic variations in speech signal	2
2.1 A typical block diagram of an ASR system.	10
2.2 High-level architecture of transformer encoder	24
2.3 High-level architecture of conformer encoder	25
3.1 Languages spoken across Indian Subcontinent	42
3.2 Three different dialects among the Telugu-speaking regions	43
3.3 The working pipeline developed for corpus collection task	44
3.4 Illustration of the Speech fragmentation algorithm	45
3.5 Instructions to be followed while performing the transcription task to achieve golden standards.	47
3.6 The workflow of crowd procurement	48
3.7 Illustration of the workflow for Approach 1	51
3.8 Administrator console for monitoring the campaigns	52
3.9 The workflow of Approach 1 (Spontaneous speech)	53
3.10 The workflow of Approach 2 (Conversational speech)	54
3.11 Transcription interface designed to correct the transcriptions provided to the crowd.	55
3.12 status of the completed jobs displayed on the transcriber’s web-based screen.	56
3.13 Illustration of text normalization	58
3.14 The flow chart for WhatsApp campaign (Approach3)	59
3.15 Audio verification web interface designed for WhatsApp campaigns.	60
4.1 WER evaluation on different architectures (like TDNN, Transformer, & Conformer) of different speaking styles sizes (Spontaneous, Conversational, and Read) on IITH-CSTD corpus	74
4.2 WER evaluation on different architectures (like TDNN, Transformer, & Conformer) of different dataset sizes (10 hrs, 50 hrs, 100 hrs, 1000 hrs, & 2000 hrs) on IITH-CSTD corpus	76
5.1 Illustration of Wav2Vec 2.0 architecture.	84
5.2 Illustration of CCA similarity with the features extracted from the CNN module	90

7.1	Working pipeline of a unified parser.	108
7.2	An example of a Telugu utterance	109
7.3	An example of a Telugu word (English translation of it is “Sanskrit”).	111
7.4	An example of a Telugu word after passing through SB-CLS parser.	111
7.5	An example of a Telugu word after passing through AB-CLS parser.	112
7.6	Proposed Multistream CNN TDNN-LSTM architecture with time-restricted self-attention	113
7.7	Comparison of CPS and different variants of CLS on Telugu, Tamil, and Gujarati languages	115

List of Tables

Table	Page
2.1 Contrastive comparison of existing speech database reported in the literature . .	35
2.2 Summary of databases used in thesis (where R, C, S indicates Read, Conversational, and Spontaneous mode speeches (speaking styles))	36
3.1 Database statistics for collected CSTD corpus	63
4.1 Evaluation of IITH-CSTD Corpus on hybrid ASR systems	72
4.2 Evaluation of IITH-CSTD Corpus on End-to-End ASR systems	72
4.3 Benchmarked comparison of IITH-CSTD corpus with MSR Telugu, Youtube	78
6.1 Comparison of the word error rates (WER) on CSTD-5k for different databases (MUCS, OpenSLR-126, OpenSLR-127, and IITH-CSTD).(where WOLM, WLM, and WLME indicate without language model, with a language model, and with language model using extra text, respectively.)	98
6.2 Comparison of multilingual fine-tuning on CSTD-5k (Telugu pretrained model)	101
6.3 WER evaluation of different pretrained models of different dataset sizes on the IITHCSTD database	103
7.1 Results of monolingual models in terms of WER (%) over a variety of acoustic models	116
7.2 Results of multilingual models in terms of WER (%) over a variety of acoustic models	118
7.3 Benchmarked comparison of MSCNN over MSCNNLSTM (where MSCNN corresponds to Multistream CNN architecture and MSCNNLSTM represents Multistream CNN-LSTM architecture (The evaluation metric considered here is word error rate (WER(%)))).	119
7.4 Benchmarked comparison of MSCNN_WA over MSCNNLSTM_WA (where MSCNN corresponds to Multistream CNN + attention architecture and MSCNNLSTM_WA represents Multistream CNN-LSTM + attention architecture (The evaluation metric considered here is word error rate (WER(%)))).	120

List of Symbols and Abbreviations

List of Symbols:

$p(x, y)$	– Joint Probability Distribution
$p(x/y)$	– Conditional Probability
λ	– Model parameters of i-vector system
T	– Total variance matrix
$E[.]$	– Expectation operator

List of Abbreviations:

AB-CLS	– Akshara Based Common Label Set
AED	– Attention based Encoder-Decoder
AM	– Acoustic Model
AI	– Artificial Intelligence
ASR	– Automatic Speech Recognition
BERT	– Bidirectional Encoder Representations from Transformers
BLSTM	– Bidirectional LSTM

BN	– Batch Normalization
BPTT	– Back Propagation Through Time
CA	– Coastal Andhra
CC	– Canonical Correlation
CC-BY-SA	– Commons Attributions Share-Alike
CCA	– Canonical Correlation Analysis
CLS	– Common Label Set
CLSRIL-23	– Cross Lingual Speech Representations for Indic Languages
CNN	– Convolutional Neural Networks
CPS	– Common Phone Set
CSTD	– Crowd Sourced Telugu Database
CTC	– Connectionist Temporal Classification
dB	– Decibels
DNN	– Deep Neural Network
DNN-HMM	– Deep Neural Network-Hidden Markov Model
EM	– Expectation-Maximization
FFN	– Feed Forward Network
fMLLR	– Feature-based Maximum Likelihood Linear Regression
GMM	– Gaussian Mixture Model
GMM-HMM	– Gaussian Mixture Model-Hidden Markov Model

GPT	– Generative Pre-trained Transformer
GPU	– Graphics Processing Unit
HCI	– Human Computer Interaction
HMI	– Human Machine Interaction
HMM	– Hidden Markov Model
JAM	– Joint Acoustic Model
kHz	– Kilo Hertz
KS	– Knowledge Sources
LDA	– Linear Discriminant Analysis
LID	– Language Identification
LM	– Language Model
LSTM	– Long Short-Term Memory
MB-CLS	– Monophone Based Common Label Set
mBERT	– Multilingual BERT
MFCC	– Mel-Frequency Cepstral Coefficients
MILE	– Medical Intelligence and Language Engineering Lab)
MLLT	– Maximum Likelihood Linear Transform
MSCNN	– Multistream CNN
MSCNNLSTM	– Multistream CNN-LSTM
MSR	– Microsoft Research

MUCS	– Multilingual and code-switching
NLP	– Natural Language Processing
NLTM	– National Language Translation Mission
OpenSLR	– Open Speech and Language Resources
PCM	– Pulse Code Modulation
RBM	– Restricted Boltzmann Machine
ReLU	– Rectified Linear Unit
RNN	– Recurrent Neural Networks
RNN-T	– Recurrent Neural Network Transducer
RNNLM	– Recurrent Neural Network Language Model
RY	– Rayalaseema
SAT	– Speaker Adaptive Training
SB-CLS	– Syllable Based Common Label Set
SGD	– Stochastic Gradient Descent
SGMM	– Subspace Gaussian Mixture Model
SNR	– Signal to Noise Ratio
STT	– Speech-To-Text
SVD	– Singular Value Decomposition
TDNN	– Time Delay Neural Network
TDNNF	– Low-rank TDNN

TG	–	Telangana
TTS	–	Text To Speech
UBM	–	Universal Background Model
USD	–	US Dollars
UTF	–	Unicode Transformation Format
VAD	–	Voice Activity Detector
WA	–	With Attention
WADA-SNR	–	Waveform Amplitude Distribution Analysis - Signal-to-Noise-Ratio
WebRTC	–	Real-Time Communication for the web
WER	–	Word Error Rate
WLM	–	With Language Model
WLME	–	Language Model with Extra text
WOLM	–	With Out Language Model
XLSR-53	–	Cross-lingual Speech Representations

Chapter 1

Introduction

Speech serves as a distinctive and natural means of communication among humans. It is one of the most potent tools possessed by human beings. It is multifaceted, encompassing information about the message, language, and speaker-specific attributes such as gender, age, and emotion. All these attributes are interwoven into a single signal, which characterizes a typical speech signal, making the field of building speech systems a fascinating area of study. However, the complexity of speech signal analysis is compounded by the quasi-periodic or non-stationary nature of speech signals, coupled with their high temporal variations.

Over the past decade, research in speech signal processing has seen significant progress. The focal point of most of this research has been the development of effective speech recognition systems. The primary function of an automatic speech recognition system (ASR) is to process the input speech signal and convert it into the corresponding text, bridging the gap between human speech and machine understanding.

The importance of these systems cannot be overstated, particularly as we continue to push the boundaries of human-machine interaction (HMI). Speech scientists predict an era of substantial growth in the speech recognition industry in the years to come. The global market potential is staggering, with projections reaching into the realm of several billion dollars.

In an increasingly digitized world, voice-based interfaces are rapidly integrating into our everyday lives. From mobile assistants to home automation systems, these interfaces transform how we interact with technology, making it more intuitive and personalized. The promise of HMI is not just about enhancing technology but also about enriching human lives by facilitating seamless interaction with our digital environment. The narrative of speech recognition is not

just about the evolution of machines but a testament to the adaptability and inventiveness of human communication.

The performance of an ASR system may be impacted if there are excessive acoustic variations in speech signals [1]. The variations in acoustic characteristics in speech signals can stem from a variety of factors, which are as follows:

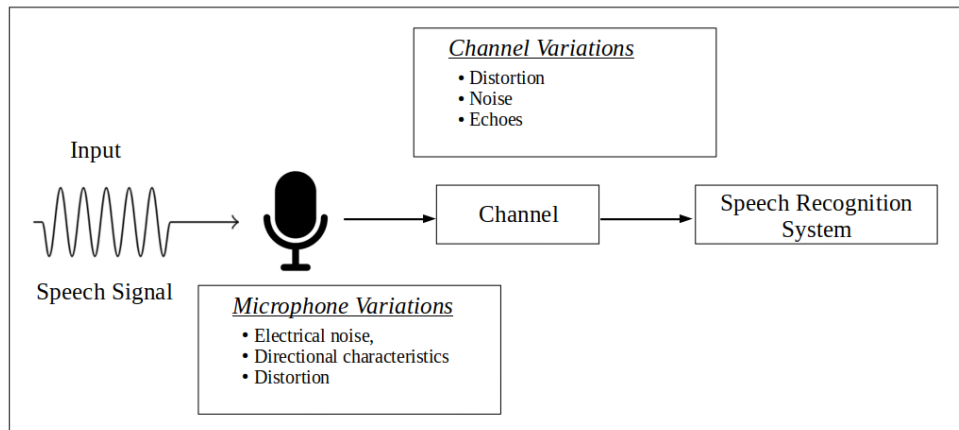


Figure 1.1: Acoustic variations in speech signal

- **Environmental conditions:**

- Background noise is a critical factor in ASR as it effectively diminishes the Signal-to-Noise Ratio (SNR). Various sources, such as a fan's whirring, the car's hum, or chatter in a "babble" setup, can contribute to this noise. This unwanted acoustic input mingles with targeted speech sounds, complicating clear and accurate speech recognition.
- During the recording of speech samples, it's essential to mitigate the presence of echo paths, which can introduce further distortions. Echoes, essentially repeated reflections of sound that reach the microphone with a delay, can add another layer of complexity, particularly in environments not acoustically designed for recording.
- Therefore, the careful management and consideration of recording environments, including minimizing background noise and controlling for echo paths, is paramount in capturing high-quality speech data, leading to improved performance in subsequent speech recognition tasks.

- **Impact of Transducer/channel:**

- This particular effect is predominantly attributed to the idiosyncrasies of the recording microphone. In its traditional role, a microphone's core function involves the transformation of an acoustic waveform, a vibrational representation of sound, into an analogous electrical signal. This process, however, is not always characterized by consistent linearity and instead exhibits a degree of variation that is often contingent on the particular microphone in use.
- This non-linear transformation can cause spectral distortion, changing the inherent properties of the recorded speech. Such inconsistencies can impact the performance of speech recognition systems which often assume a linear relationship between the original and recorded signal.
- In practical applications, we have observed variations in microphone response that span a considerable range, extending up to 20 dB. This considerable deviation can have a profound effect, significantly modifying the spectral attributes inherent to the speech signal.
- These alterations, while potentially subtle to human listeners, can significantly affect automated processes relying on spectral features. As such, understanding and accounting for the influence of the transducer/channel is of paramount importance in improving the performance and robustness of speech recognition systems.

- **Physiological:**

- Each human possesses a unique vocal tract length, which inherently influences the resonance frequencies of their speech. It is observed that the resonance frequencies typically decrease with the increase in the size of the vocal tract. As such, male adults generally exhibit lower resonance frequencies compared to females, who, in turn, typically display lower resonance frequencies than children.
- This interesting phenomenon implies that even if two individuals possess the same pitch, the spectra of their speech can exhibit discernible differences due to variances in their head sizes, which influence the length and shape of their vocal tracts.

These physical disparities are among the many factors that contribute to the rich complexity and individuality of human speech and pose unique challenges and considerations in the field of speech recognition.

- **Behavioral:**

- The pace of speech notably varies among individuals, with influences stemming from a multitude of factors. It has been observed that the speaker's regional and social background significantly impacts their accent and lexical choices.
- For example, people from different geographical areas or social communities may speak at different rates and utilize unique accents, idioms, or phrasing. These variations in speaking rate, accent, and language use can create additional challenges in speech recognition, as systems must be capable of understanding a broad range of speech patterns and rates. This highlights the importance of designing and training speech recognition systems that are adaptable and flexible in handling diverse speaking styles and accents, ensuring their applicability across a wider range of real-world scenarios.

- **Phonetic context:**

- The fundamental building block of speech is the “phoneme,” with a combination of phonemes forming words and, subsequently, sentences. The articulation of any given phoneme depends on its surrounding phonetic context, that is, the phonemes that precede and follow it.
- Take, for instance, the words “bat” and “pat.” The phonetic difference resides solely in the initial phonemes, /b/ and /p/. This demonstrates how a minor change in the phonetic composition of a word can lead to an entirely different word, emphasizing the importance of accurately recognizing phonetic variabilities.
- If an ASR system struggles to distinguish such nuances accurately, it could result in significant errors in transcription and understanding. Therefore, a nuanced grasp of the contextual dependencies of phonemes is essential for the development of robust and reliable ASR systems.

1.1 Scope of the thesis

The primary objective of this thesis is to develop ASR systems tailored for Indic languages. An essential component in constructing these systems is the availability of a well-curated database accompanied by comprehensive transcripts. Recent research [2, 3] has indicated that English ASR systems have achieved human-level performance in public domains. The quantity of annotated data accessible for English ASR tasks is approximately 30,000 hours [4, 5]. Conversely, the situation for Indic languages is quite different, with available corpora rarely exceeding 200 hours. As a result, it is imperative to establish a method or framework for curating large-scale databases for ASR tasks specifically designed for Indic languages. Additionally, it is important to investigate the acoustic sharing properties between resource-rich and resource-poor languages within the Indian context.

The scope of the thesis is summarized as follows:

- This thesis provides a platform/tool for collecting speech corpora for the speech recognition task. Furthermore, strategies for collecting the database in a crowdsourced approach are discussed.
- This thesis investigated the importance of dialect-based information for speech recognition tasks.
- Exploration of self-supervised approach for improving the performance of ASR system.
- In this thesis, the importance of acoustic sharing among the resource-rich and resource-poor languages is investigated.
- Building an ASR system in the Indian context for low-resource settings by investigating different neural network architectures.

1.2 Organization of the thesis

The remainder of this dissertation will be structured in the following manner:

- **Chapter 2**, presents relevant background information associated with this thesis. This chapter aims to equip readers with the full comprehensive content of the thesis.

- **Chapter 3**, unveils the International Institute of Information Technology Hyderabad-Crowd Sourced Telugu Database (IIITH-CSTD), which is a compilation of Telugu speech samples gathered via crowd-sourcing. The primary purpose of this chapter is to demonstrate how crowd-sourcing strategies can be applied to alleviate the scarcity of resources for the Telugu language. In this context, we've established a crowd-sourcing pipeline and discussed the protocols implemented for collecting this corpus.
- **Chapter 4**, delves into various hybrid and end-to-end frameworks employed on the IIITH-CSTD corpus. The chapter aims to establish definitive baseline measures for the Telugu ASR task, paving the way for future work and developments in this field.
- **Chapter 5**, provides an introduction to the self-supervised representation learning methodology. This chapter focuses on training a wav2vec2.0 model, leveraging an extensive 5000-hour of unlabeled Telugu audio. Firstly, the intricacies of the pretraining process involved are discussed. The latter portion of this chapter shifts focus to a comprehensive layer-by-layer analysis of the Telugu pretrained model. (The Telugu pretrained model in this thesis is referred to as CTSD-5k).
- **Chapter 6**, presents a detailed analysis of CSTD-5k, Cross Lingual Speech Representations for Indic Languages (CLSRIL-23), XLSR-53 by performing fine-tuning on different dataset sizes varying from 10 minutes, 1 hour, 10 hours and 100 hours respectively using IIITH-CSTD corpus. Later part of the chapter, CSTD-5k performance across Language Families is investigated. Lastly, an impact of Multilingual Fine-tuning on the CSTD-5k is explored.
- **Chapter 7**, investigates the development of multilingual Automatic Speech Recognition (ASR) systems. This chapter guides the training of a joint acoustic model using a common phone set (CPS) and a common label set (CLS). Initially, it begins by evaluating the effectiveness of different lexicon units, which includes CPS and CLS, such as monophones, aksharas, and syllable-based systems, for constructing monolingual ASR systems. Later, the investigation is performed on various acoustic models designed to develop monolingual and multilingual ASR systems employing a CLS.

- **Chapter 8**, delivers a comprehensive overview and conclusion of the dissertation. Additionally, it discusses potential pathways for subsequent research directions.

Chapter 2

Background and Related Work

In this chapter, we lay down a comprehensive groundwork touching upon several topics integral to this dissertation. Our discussion begins with an overview of automatic speech recognition (ASR) in Section 2.1, a pivotal application in speech processing and the main evaluative criterion for the systems discussed in this thesis. Following this, we turn to three predominant neural network architectures used in speech processing in Section 2.2. These include Long Short-Term Memory (LSTM), Transformer, and Conformer models, each offering distinct capabilities and advantages in processing speech data. Section 2.3 offers a succinct review of the evolution of neural network pre-training, subsequently delving deeper into the contemporary advancements in representation learning within speech domains, facilitated by self-supervised techniques. Section 2.4 encapsulates a survey of the pertinent studies in the area of multilingual speech recognition, shedding light on the strides made in this domain and the challenges yet to be overcome. In Section 2.5, we discuss the challenges associated with building ASR systems specifically for Indian languages, which present unique linguistic and socio-cultural characteristics that pose interesting problems and opportunities for ASR research. Section 2.6 outlines the databases employed in this thesis for constructing speech recognition models and conducting various studies. These databases form the backbone of our empirical work and provide a rich source of speech data to train and evaluate our models. Finally, we conclude this chapter with a summary and some concluding remarks, synthesizing the key points discussed and setting the stage for the detailed exploration in the subsequent chapters are presented in Section 2.7.

It is important to emphasize that the review presented in this chapter does not aim to be exhaustive, encompassing all existing research (indeed, not even all cutting-edge papers will be

referenced) or delve into the intricate mathematical aspects of the topics discussed. Instead, the objective is to furnish readers with adequate background knowledge to facilitate a thorough understanding of the contents of this thesis.

2.1 Automatic Speech Recognition

Automatic Speech Recognition (ASR) represents a crucial technology enabling the transformation of spoken language into written text through computational means. ASR systems process speech signals as input and generate corresponding textual output, effectively handling diverse accents, dialects, speaking styles, and ambient noise. The primary objective of ASR tasks is to enhance human-computer interactions by enabling seamless communication. ASR has found widespread use across various domains, including call centers, voice dialing, data entry, dictation, command and control, and computer-assisted language learning. In contemporary applications, ASR is combined with text-to-speech synthesis (TTS) to develop intelligent personal assistants (e.g., Apple Siri, Google Home, Amazon Echo) in chatbots and mobile phones and accessibility tools for individuals with hearing or speech impairments. Continuous progress in machine learning and natural language processing has notably boosted the accuracy and effectiveness of ASR systems, making them indispensable in modern human-computer interactions.

In recent times, ASR has evolved into a critical component of information processing systems. Initially, speech recognition systems were limited to recognizing simple, isolated words and relied on acoustic models trained to minimize acoustic distance. To create large vocabulary ASR systems capable of operating in unconstrained environments, it is crucial to incorporate information from various knowledge sources (KS) into the ASR system [6, 7, 8]. Developing an ASR system involves drawing from multiple knowledge sources, including acoustics, phonetics, phonology, prosodics, lexical, syntax, semantics, and pragmatics. These sources provide valuable insights into speech variability, speech sound properties, stress and intonation patterns, language patterns, grammatical structures, word meanings, and conversational context. Integrating these knowledge sources into the ASR system enhances its accuracy and effective-

ness in speech signal recognition. This comprehensive understanding of speech aspects enables the development of more robust ASR systems.

2.1.1 Problem Formulation

A representative block schematic of an ASR system is depicted in Figure 2.1. The block

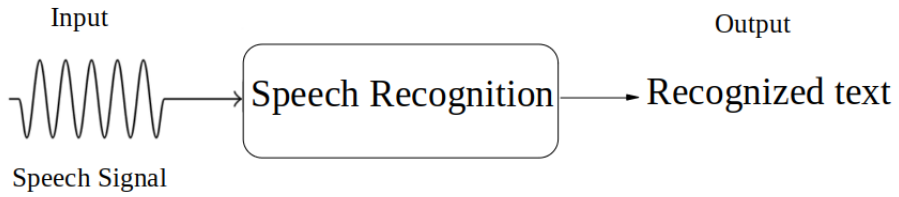


Figure 2.1: A typical block diagram of an ASR system.

schematic demonstrates that the speech recognition system seeks to identify or predict the corresponding text based on a given input speech signal. Mathematically, the speech recognition problem can be formulated as:

$$\text{recognized text} = \mathbf{H}(\text{speech signal}) \quad (2.1)$$

Here from the Equation 2.1, the role of $\mathbf{H}(\bullet)$ is to properly/correctly estimate the function ($\mathbf{H}(\bullet)$) in the training procedure. Cutting-edge statistical frameworks facilitate the conversion of speech signals into corresponding textual sequences. The transformation function $\mathbf{H}(\bullet)$ possesses a probabilistic characteristic in these frameworks. Consequently, the speech recognition problem can be more explicitly expressed using the statistical formulation, as highlighted in prior studies [7, 9, 10, 11]. The statistical formulation involves estimating the probabilities of the word sequence given the acoustic features, which can be represented as $P(\mathbf{W}|\mathbf{X})$. The objective is to find the most probable word sequence that maximizes this probability, which can be expressed as:

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w}} P(\mathbf{W}|\mathbf{X}) \quad (2.2)$$

here $\hat{\mathbf{w}}$, is the predicted word sequence and $P(\mathbf{W}|\mathbf{X})$ is the posterior probability distribution. The Equation 2.2 can be further decomposed by applying the Bayes rule, which can be written

as:

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w}} P(\mathbf{W}|\mathbf{X}) = \arg \max_{\mathbf{w}} \frac{P(\mathbf{W})P(\mathbf{X}|\mathbf{W})}{P(\mathbf{X})} \quad (2.3)$$

In Equation 2.3, \mathbf{X} represents the sequence of observation feature vectors extracted from the input speech signal, where $x_1, x_2, x_3, \dots, x_T$ correspond to the T frames of the signal, and \mathbf{W} represents the sequence of words, where $w_1, w_2, w_3, \dots, w_T$ correspond to the words in the sequence. The denominator of Bayes rule, (from Equation 2.3) $P(\mathbf{X})$, is ignored as it has no bearing on the maximum posterior probability. The product of the prior probability of the word sequence $P(\mathbf{W})$ and the likelihood of the observation feature vectors given the word sequence $P(\mathbf{X}|\mathbf{W})$ is maximized to obtain the hypothesized word sequence. So the Equation 2.3, can be simplified as:

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w}} P(\mathbf{W}|\mathbf{X}) = \arg \max_{\mathbf{w}} \frac{P(\mathbf{W})P(\mathbf{X}|\mathbf{W})}{P(\mathbf{X})} = \arg \max_{\mathbf{w}} P(\mathbf{W})P(\mathbf{X}|\mathbf{W}) \quad (2.4)$$

In speech research, Equation 2.4 is frequently denoted as the “fundamental equation of speech recognition.” The statistical models employed for estimating $P(\mathbf{X}|\mathbf{W})$ and $P(\mathbf{W})$ are generally referred to as the *acoustic model* (AM) and *language model* (LM), respectively. Consequently, the ASR problem has evolved into identifying the optimal parameterizations for acoustic and language models.

Acoustic Model evaluates $P(\mathbf{X}|\mathbf{W})$ for a given text sequence W , which represents the likelihood of a speech utterance X , given W . For instance, consider a speech sample of someone uttering the phrase “my name is ganesh.” An ideal acoustic model would assign a higher likelihood score to $P(\mathbf{X}|\text{my name is ganesh})$ compared to a phrase that sounds significantly different, such as “increase the tv volume.”

The likelihood $P(\mathbf{X}|\mathbf{W})$ is determined by the acoustic model (AM), which is responsible for improving the recognition accuracy by learning high-level statistics and adjusting for differences in speakers, dialects, environments, and noise. To acquire such a statistical model of this nature generally necessitates a dataset containing audio recordings and their corresponding textual transcripts $\{(X^{(i)}, W^{(i)})\}_{i=1}^N$, with N representing the total number of audio-transcript pairs in the dataset. Consider the exemplification of speech utterance denoted as \mathbf{X} . Conventionally, \mathbf{X}

is characterized as a continuum of spectral attributes, symbolized as a series of spectral feature vectors $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$, including the logarithmic Mel-filterbank features and Mel-frequency cepstral coefficients (MFCCs) [12]. The sequence length, denoted by T , possesses variability and is contingent upon the length of the individual utterance. The textual transcript, symbolized as \mathbf{W} , comprises a series of textual elements that vary in their levels, depending upon the final application of the ASR system during the training phase of acoustic models. A prevalent method of representing \mathbf{W} involves employing a meticulously designed pronunciation model. This model facilitates the mapping of each word within \mathbf{W} to its respective pronunciation, concomitant with its associated probability. Each spoken word \mathbf{W} can be decomposed into a sequence of N_w basic speech sound units, known as base phones, denoted by $\mathbf{q}_w = q_1, q_2, q_3, \dots, q_T$. This sequence corresponds to the pronunciation of that particular word. In practice, the pronunciation sequence is provided by a lexicon (pronunciation dictionary) that ideally contains the phonetic decomposition of the words. Multiple pronunciations for a given word can be accommodated in the dictionary by computing the different pronunciations, and mathematically it can be expressed as:

$$P(\mathbf{W}|\mathbf{X}) = \sum_{\mathbf{Q}} P(\mathbf{X}|\mathbf{Q})P(\mathbf{Q}|\mathbf{W}) \quad (2.5)$$

In conventional practice, a Hidden Markov Model (HMM) [6] is utilized to model $P(\mathbf{X}|\mathbf{Q})$. Each phonetic unit, or phone, possesses an individual HMM assigned for its representation. Typically, this model comprises three hidden states designed to encapsulate the transitory acoustic dynamics within a phone. This can be conceptualized as an attempt to model the phonetic unit's beginning, middle, and end. In HMM-based modeling, each base phone q is represented by a continuous density HMM, and the sequence of base phones \mathbf{Q} is obtained by concatenating all the base phones for a given the word (as discussed in Equation 2.5), such that $\mathbf{Q} = \mathbf{q}_{w1}, \mathbf{q}_{w2}, \mathbf{q}_{w3}, \dots, \mathbf{q}_{wL}$. The acoustic model can be expressed mathematically as a summation of all valid pronunciation sequences for a given the word \mathbf{W} , where \mathbf{Q} represents a particular pronunciation sequence. This is expressed as follows:

$$P(\mathbf{W}|\mathbf{Q}) = \sum_{\mathbf{s}} P(\mathbf{X}, \mathbf{s}|\mathbf{Q}) \quad (2.6)$$

The state sequence through the composite hidden Markov model (HMM) has been represented by \mathbf{s} . The speech features have been effectively modeled and represented by concatenating a

sequence of HMM phone models. It is important to note that context-dependent variations in speech have not been considered in the current formulation. This is particularly evident in the case of vowels, such as the phoneme /a/, where variations between the words "bat" and "cat" are observed. It is worth mentioning that context-independent phone models are referred to as monophones. To address this issue, a simple approach of incorporating context information in the phone models has been proposed in the past. Specifically, a unique HMM phone model has been employed for every possible pair of left and right neighbors of the phones. This approach creates triphones, which are HMM models that incorporate context-dependent variations in speech.

Within the realm of acoustic feature vectors denoted as \mathbf{X} , these are perceived as state emissions. The densities of these vectors are typically represented by either a cluster of Gaussian Mixture Models (GMM) [13], or a Deep Neural Network (DNN) [14], with the DNN often demonstrating superior capabilities. In the case of GMM-HMM acoustic models, parameter determination can be achieved by implementing maximum likelihood estimation via the forward-backward algorithm, further discussed in [15]. The parameter estimation for DNN-based acoustic models usually begins with a pre-trained foundational GMM-HMM speech recognition system. This system calculates the target state label for each frame in the audio sequence, resulting in what is commonly termed as forced alignment. Once the target state sequence is acquired, the DNN-based acoustic model can be trained to employ backpropagation alongside traditional gradient descent techniques. The exploration and application of varied neural architectures, including feed-forward neural networks [16, 17], recurrent neural networks [18, 19], and convolutional neural networks [20], have been thoroughly examined in the field.

Language Model The statistical model employed to estimate $P(\mathbf{W})$ in Equation 2.4 is commonly referred to as the language model within the field. It provides an a priori probability, indicating the likelihood of a textual sequence \mathbf{W} appearing initially in human language (i.e., being spoken or written). To illustrate the function of a language model in speech recognition, let's consider an example featuring a recording of the phrase "how to recognize speech" and another recording of the phrase "how to wreck a nice beach." A robust acoustic model might

struggle to attribute distinct likelihood scores $P(\mathbf{X}|\mathbf{W})$ to the two recordings. This challenge arises due to the strikingly similar pronunciations of the two phrases. Consequently, their \mathbf{Q} (the phone sequence converted from the original text sequence \mathbf{W} using the pronunciation model) would also exhibit substantial similarity, resulting in strikingly similar \mathbf{X} . At this juncture, a language model intervenes and settles the impasse. It would simply assign a higher likelihood score to $P(\text{"how to recognize speech"})$ than to $P(\text{"how to wreck a nice beach"})$ because the former is empirically more likely to be spoken or written by an individual.

The primary function of the language model in ASR is to provide the value of $P(\mathbf{W})$ for the basic ASR equation. The probabilistic relationship between a sequence of words can be established and effectively modeled through the utilization of a text corpus containing a large number of words. Mathematically, a language model can be expressed as a probability distribution over a sequence of words. Specifically, for a sequence of words \mathbf{W} consisting of N words, the language model can be expressed as:

$$\begin{aligned}
 P(\mathbf{W}) &= P(w_1, w_2, w_3, \dots, w_N) \\
 &= P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \dots P(w_N|w_1, w_2, \dots, w_{N-1}) \\
 &= \prod_{i=1}^N P(w_i|w_1, w_2, \dots, w_{i-1})
 \end{aligned} \tag{2.7}$$

The factorization outlined in Equation 2.7 suggests a combinatorial expanse of possible word histories. This magnitude frequently results in language models demonstrating suboptimal generalization capabilities when encountering novel combinations of words. To circumvent this issue, it is common practice to formulate n -gram models with a modest n , generally between two to five. This approach guarantees that the models only account for the immediate $n - 1$ word when forecasting the upcoming word. For instance, a bi-gram model, where $n = 2$, would offer an approximation to Equation 2.7 as follows:

$$\begin{aligned}
 P(\mathbf{W}) &= P(w_1, w_2, w_3, \dots, w_N) \\
 &= P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \dots P(w_N|w_1, w_2, \dots, w_{N-1}) \\
 &\approx P(w_1)P(w_2|w_1)P(w_3|w_2) \dots P(w_N|w_{N-1})
 \end{aligned} \tag{2.8}$$

Depending on the number of words used to predict the likelihood of the current word, the n-gram model can be extended to bi-gram and tri-gram models, where the current word depends on the previous word $P(w_i|w_{i-1})$ or two previous words $P(w_i|w_{i-2}, w_{i-1})$, respectively. The estimation of bi-gram probabilities, which involves determining the likelihood of one word w_i preceding another word w_j , is executed by merely enumerating the occurrences of the sub-sequence (w_i, w_j) within the text corpus.

$$P(w_j|w_i) = \frac{\text{Count}(w_i, w_j)}{\text{Count}(w_i)} \quad (2.9)$$

Lately, Recurrent Neural Networks (RNNs) have emerged as an alternative to n-gram models in the parameterization of language models [21]. Due to the intricate workings of an RNN, while processing each word w_i in a text sequence $W = (w_1, w_2, \dots, w_N)$, the network retains a hidden state h_i at every time step, which naturally encodes the information of all prior words from w_1 to w_{i-1} . It is mathematically expressed as:

$$\begin{aligned} P(W) &= \prod_{i=1}^N P(w_i|w_1, w_2, \dots, w_{i-1}) \\ &= \prod_{i=1}^N P(w_i|h_i), \\ h_i &= \text{RNN}(w_i, h_{i-1}). \end{aligned} \quad (2.10)$$

This attribute empowers RNN-based language models to more adeptly capture the long-term context dependencies present in text sequences compared to n-gram models. Thus, when utilized, they often contribute to enhancing ASR system performance. In contrast to the process of training acoustic models, training a language model relies exclusively on text data $W^{(i)}_{i=1}^N$. Typically, the maximum likelihood estimation method is utilized, which instructs the language model to optimize the probabilities of producing the text sequences found within the text corpus.

$$\theta^* = \arg \max_{\theta} \prod_{i=1}^N P(W^{(i)}; \theta) = \arg \max_{\theta} \sum_{i=1}^N \log(P(W^{(i)}; \theta)) \quad (2.11)$$

In the Equation 2.11, the symbol θ signifies the parameters of the language model.

Decoding The speech recognition procedure can be conceptualized as a search challenge, wherein the objective is to identify the sequence of words that optimizes Equation 2.4. Two primary techniques are employed to discover the optimal path: breadth-first search and depth-first search. In

a depth-first search, the most likely hypothesis is pursued until the speech utterance concludes, while in a breadth-first search, all potential hypotheses are explored concurrently. Viterbi decoding, a specific form of breadth-first search, is the prevalent method in speech recognition systems. A decoding graph is generated to conduct decoding, consisting of branches leading to every possible initial word, branches connecting the starting word to every potential subsequent word, and so forth. When fully populated, this graph encompasses all feasible word sequences. Pronunciation sequences are substituted for each word within the graph, with multiple pronunciations combined in parallel. Common phone models with similar contexts are merged and share the same entry points. Ultimately, the decoding graph is comprised of nodes (phones, characters, HMM-states) linked by node transitions and word end nodes connected by word transitions. It is possible to reach any path from the starting node, and the score is assessed by summing all log state transition probabilities along the path. These paths can be obtained through a movable token positioned at the conclusion of each node and each word. This token carries both the score and the history of traversed words. The graph can navigate any path by shifting the node from its current location to an adjacent node. Consequently, the search challenges can be viewed as a token-passing algorithm.

Initially, a singular token is placed at the tree's starting node and then duplicated to all connecting nodes. Only the best token is preserved if multiple tokens arrive at a node. Once all the acoustic vectors have been processed, the end words are considered, and the word token with the highest score signifies the optimal path. The history within the token discloses the best possible word sequence. Nevertheless, the token-passing algorithm necessitates considerable space and time for computation. To render it feasible, the token with the highest score is selected at every frame, and tokens with scores falling below the beam width from the top score are discarded. Calculating the paths within the beam width from the optimal path becomes efficient and manageable.

Evaluation The efficacy of an Automatic Speech Recognition (ASR) system is assessed utilizing the Word Error Rate (WER) as the primary metric. The significance of this measure becomes particularly crucial in cases where the reference and hypothesized speech segments

differ in length. To calculate the WER, an initial alignment of the hypothesized and reference speech samples is performed. WER is mathematically expressed as follows:

$$WER(\%) = \frac{S + I + D}{N} \times 100 \quad (2.12)$$

Where S , I , and D represent the quantities of substitutions, insertions, and deletions, respectively. The total number of words in the reference utterance is denoted by N .

2.1.2 End-to-End ASR

Drawing upon the fundamental speech recognition equation, referred in Equation 2.4. ASR systems have been traditionally constructed in a “hybrid” manner for several decades. This traditional method involves the independent development of distinct components, such as acoustic, pronunciation, and language models, which are subsequently amalgamated to compose the complete ASR systems. In contrast, recent trends show a gradual transition from this hybrid modeling approach toward end-to-end modeling. This innovative method aspires to convert speech utterances directly into textual tokens employing a singular, integrated model.

Within the framework of end-to-end Automatic Speech Recognition (ASR), the conditional probability function, represented as $P(W|X)$, is typically conceptualized via a single model, predominantly in the form of a neural network. This end-to-end approach affords numerous advantages when compared to conventional hybrid methodologies:

1. The primary advantage of end-to-end ASR models lies in their use of a unified objective function. This function aligns with the ASR objective and is crucial for optimizing the entire network. Conversely, hybrid models develop each component independently, increasing the likelihood of compounded errors across the components.
2. Moreover, end-to-end models are engineered to directly convert spoken language into textual tokens. Unlike conventional hybrid systems, this streamlined process significantly simplifies the speech recognition pipeline. The latter are often intricate in design and necessitate extensive expertise and years of experience in ASR.

3. The compact nature of end-to-end systems, facilitated by using a single neural network for modeling, sets them apart from traditional hybrid systems. This compactness enables these systems to be deployed on devices with high accuracy and low latency.

The inherent benefits of end-to-end ASR have sparked considerable attention in recent times [22, 23, 24, 25, 26, 27, 28, 29, 30]. Indeed, some end-to-end systems have already outperformed hybrid systems, even though the latter have undergone optimization at the production level over several decades [31, 32, 33].

The following sections briefly introduce three prevalent techniques in modeling end-to-end ASR: Connectionist Temporal Classification, Attention-Based Encoder-Decoder, and Recurrent Neural Network Transducer.

Connectionist Temporal Classification is an innovative approach utilized in end-to-end ASR models. It was devised to overcome the alignment problem, a prevalent challenge in sequence prediction tasks, particularly speech recognition. In traditional methods, explicit alignment is required between the input and output sequences. This necessitates a detailed mapping, which can be highly complex for speech data due to the variable nature of speech speed and style across individuals. The CTC algorithm elegantly circumvents this issue by integrating all possible alignments, automatically allowing the model to learn the optimal alignment. CTC introduces a “blank” symbol to the output alphabet, denoted as “ ϵ ”. This symbol represents the possibility of no output at a particular timestep. Mathematically, CTC defines a probability distribution over all possible alignments by summing them. If “ X ” is the input sequence and “ Y ” is the output sequence, CTC models the conditional probability $P(Y|X)$. This probability is computed by summing over all possible alignments “ A ” of “ Y ” onto “ X ” :

$$P(Y|X) = \sum_A P(A|X) \quad (2.13)$$

where A represents all possible alignments of output Y on input X .

CTC uses a recurrent neural network (RNN) to model $P(A|X)$. The RNN takes the input sequence X and outputs a probability distribution over the extended alphabet (original alphabet plus the blank symbol) at each timestep. The probabilities of all possible alignments are then

computed by a forward-backward algorithm, a dynamic programming method similar to the one used in the Hidden Markov Model. The objective of CTC is to maximize the log probability of the correct output sequence given the input sequence, i.e., maximize $\log P(A|X)$. This is usually done using gradient-based methods like stochastic gradient descent (SGD). Mathematically CTC loss is defined as follows:

$$\begin{aligned}\mathcal{L}_{CTC} &= -\log \sum_{\tilde{A}} P(\tilde{A}|X) \\ &= -\log \sum_{\tilde{A}} \prod_{t=1}^T P(\tilde{a}_t|X)\end{aligned}\tag{2.14}$$

here in Equation 2.14, $\tilde{A} = (\tilde{a}_1, \tilde{a}_2, \tilde{a}_3, \dots, \tilde{a}_T)$ represents the valid CTC path.

Attention-Based Encoder-Decoder is another effective technique utilized in end-to-end ASR models. This approach, in essence, is a sequence-to-sequence model with the addition of an attention mechanism, allowing the model to focus on different parts of the input sequence at each step of the output sequence generation. The model is structured into two primary components: the encoder and the decoder. The encoder processes the input sequence (such as an audio waveform in ASR) and maps it into a higher-dimensional, fixed-length context vector. The decoder then uses this context vector to generate the output sequence (like a transcript in ASR) one token at a time.

The attention mechanism plays a crucial role in this model. As the decoder generates each token, the attention mechanism determines which parts of the encoded input sequence are most relevant. This relevance is computed as a weighted sum of the encoder's outputs. The weights, often referred to as attention scores, indicate the importance of each input element for the current output token. Mathematically, if we denote the encoder's outputs as h_1, h_2, \dots, h_T and the attention scores at time step t as $a_1^t, a_2^t, \dots, a_T^t$ the context vector c_t for output token at time t can be computed as:

$$c_t = \sum_i a_i^t h_i\tag{2.15}$$

The attention scores α_i^t are usually computed using a softmax function over some measure of relevance (such as a dot product) between the decoder's previous hidden state and each of the encoder's outputs:

$$\alpha_i^t = \text{softmax}(\text{score}(s_{t-1}, h_i)) \quad (2.16)$$

where s_{t-1} is the decoder's previous hidden state. One key advantage of the attention-based approach is that it allows the model to handle inputs and outputs of different lengths and varying length ratios. This is particularly useful in tasks like ASR, where the input sequence (audio waveform) and output sequence (transcript) length can vary greatly. However, it also introduces challenges like handling long input sequences, leading to further research and development of various attention mechanisms.

Recurrent Neural Network Transducer (RNN-T) is prominent approach in end-to-end ASR systems. This model was designed to provide a natural and flexible way to perform sequence-to-sequence tasks without needing pre-specified alignment between the input and output sequences. The RNN-T model comprises three main components: the encoder, the prediction network, and the joint network. The encoder processes the input sequence (such as a sequence of acoustic features in ASR) and converts it into a higher-level representation. The prediction network generates the output sequence (like a sequence of phonemes or words in ASR). The joint network combines the outputs of the encoder and the prediction network to compute a distribution over the output labels at each timestep. Mathematically, the RNN-T model defines a probability distribution over the output sequence Y given the input sequence X as follows:

$$P(Y|X) = \prod_t P(y_t | y_{1:t-1}, x_{1:t}) \quad (2.17)$$

where y_t is the output at timestep t , $y_{1:t-1}$ are the previous outputs, and $x_{1:t}$ are the inputs up to timestep t . The probability $P(y_t | y_{1:t-1}, x_{1:t})$ is computed by the joint network as a softmax over the output labels. The RNN-T model can be trained end-to-end using the maximum likelihood principle, i.e., by maximizing the log probability of the correct output sequence given the input sequence. This is typically done using backpropagation through time (BPTT) and stochastic gradient descent (SGD).

One of the key advantages of the RNN-T model is its ability to output symbols as soon as it's confident about its prediction. This makes it suitable for streaming ASR applications, where low latency is crucial. However, training RNN-T models can be computationally demanding due to the need to sum over all possible output sequences, which has been addressed in recent research using various approximation methods.

2.1.3 Semi-Supervised ASR

So far, ASR has been treated as an exclusively supervised learning matter: whether applying hybrid or end-to-end methodologies, it invariably demands audio-transcript pairs to train systems in acquiring proficient speech representations. Meanwhile, using extensive and effortlessly accessible unlabeled speech data to augment supervised ASR performance constitutes an ongoing research concern. Two principal avenues exist for harnessing unlabeled speech data to address semi-supervised ASR tasks.

The initial investigation approach involves self-training [34, 35, 36], commonly referred to as pseudo-labeling. This process initiates with training a teacher model utilizing available labeled data. Following this, the teacher model annotates the unlabeled data. Consequently, a student model is trained using both labeled data and the newly pseudo-labeled data. The pseudo-labeling method can be reapplied iteratively to augment the quality of the teacher model. Historically, self-training has been confirmed to be a highly effective approach, garnering extensive research in the field of ASR [37, 38, 39, 40, 41, 42].

An additional strategy to leverage unlabeled speech data involves unsupervised pre-training, also referred to as self-supervised pre-training. This method initially guides the model to handle a proxy task, specifically designed to interact exclusively with unlabeled data. Empirical evidence from such a proxy task indicates its potential to set the model's parameters at a beneficial starting point prior to the initiation of supervised data training. A significant amount of current research has focused on creating proxy tasks that promote optimal model performance when fine-tuned for ASR tasks [43, 44, 45, 46, 47, 48, 49]. Furthermore, studies suggest that

the improvements resulting from self-training and unsupervised pre-training have a synergistic effect on downstream ASR applications [50, 51].

2.2 Overview of Neural Networks for Speech Processing

In this section, we examine two prominent neural network architectures utilized in speech processing: recurrent neural networks and self-attention networks. We assume that readers possess a foundational understanding of their basic mechanisms and variations (e.g., long short-term memory) and concentrate on reviewing their applications in speech processing, emphasizing end-to-end ASR modeling.

2.2.1 Recurrent Neural Networks

Recurrent Neural Networks (RNN) have been widely adopted in speech processing tasks due to their ability to model sequential data effectively. In the context of end-to-end ASR, RNNs have demonstrated remarkable performance by directly converting speech signals into text without requiring intermediate representations. In end-to-end models like CTC (Connectionist Temporal Classification), AED (Attention-based Encoder-Decoder), and RNN-T (Recurrent Neural Network Transducer), the encoder network plays a critical role. The primary objective of the encoder network is to convert the input audio sequence $X = (x_1, x_2, \dots, x_T)$ into a high-level feature representation $H = (h_1, h_2, \dots, h_T)$.

The encoder network processes the input audio sequence and extracts relevant features, enabling the model to make accurate predictions. Various neural network architectures can be employed as encoder networks, such as Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and Self-Attention Networks (Transformers). When end-to-end models first emerged [52, 53, 27], Long Short-Term Memory (LSTM) networks [54], a type of Recurrent Neural Network (RNN), were predominantly utilized to construct encoder networks. There are two primary configurations for assembling encoder networks using LSTMs:

1. **Multi-Layer Unidirectional LSTM:** The encoder comprises multiple layers of unidirectional LSTM cells in this arrangement. Each layer processes the input audio sequence or

the output from the previous layer sequentially, from the beginning to the end of the sequence. Unidirectional LSTMs are proficient at capturing temporal dependencies within the input audio sequence.

$$h_t^l = LSTM(x_t^l, h_{t-1}^l), \quad (2.18)$$

2. **Multi-Layer Bidirectional LSTM:** As an alternative, the encoder can be assembled using multi-layer bidirectional LSTM cells. In this setup, each layer consists of two distinct LSTM networks: one processes the input sequence from the beginning to the end, while the other processes it in reverse order, from the end to the beginning. The outputs from both networks are combined at each timestep. Bidirectional LSTMs enable the encoder to capture past and future context within the input audio sequence, yielding a more robust feature representation.

$$h_t^l = [LSTM(x_t^l, h_{t-1}^l), LSTM(x_t^l, h_{t+1}^l)], \quad (2.19)$$

Here $LSTM(\bullet)$ denotes the standard LSTM unit, h_t^l denotes the hidden output of the l^{th} layer at time t , and x_t^l as the input vector for the l^{th} layer. The input vector x_t^l is equal to x_t when $l = 1$ (i.e., the first layer input). Otherwise, it equals h_t^{l-1} . The output of the final layer in the LSTM network serves as the encoder output.

2.2.2 Self-Attention Networks

In the early stages of end-to-end model development, LSTMs were the preferred choice for encoder networks due to their capacity to model short-range dependencies effectively in sequential data. However, with the introduction of newer architectures such as Self-Attention Networks or Transformers, the variety of encoder networks in end-to-end models has expanded, offering researchers a range of options tailored to the specific demands of their ASR tasks. In the literature, many of these networks have proven to be more adept at capturing long-term dependencies within sequences. This can be attributed to the self-attention mechanism employed by Transformers, which permits access to the entire sequence when deriving h_t for each x_t in X . As a result of the impressive modeling capabilities offered by Transformers, there is an emerging tendency to substitute LSTM encoder networks with Transformer-based architectures in end-

to-end ASR models [55, 56, 57, 58, 59]).

The encoding architecture of a Transformer-centric end-to-end model encompasses an array of Transformer units. Each unit houses a multi-head self-attention layer alongside a feed-forward network (FFN). In order to establish connections between distinct layers and units, residual connections and layer normalization are integrated. Within every Transformer unit, the input vector \mathbf{x}_t is subjected to linear transformation, resulting in a query vector \mathbf{q} , a key vector \mathbf{k} , and a value vector \mathbf{v} through the use of matrices \mathbf{W}_q , \mathbf{W}_k , and \mathbf{W}_v respectively. The same is portrayed in Figure 2.2.

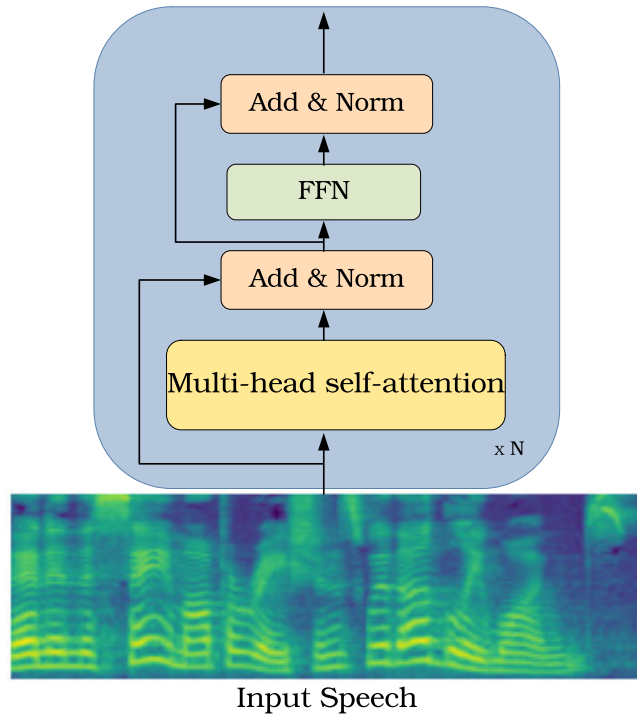


Figure 2.2: High-level architecture of transformer encoder

The self-attention mechanism leverages the dot-product similarity function to compute the attention distribution over the input sequence. It is mathematically expressed as follows:

$$\alpha_{t,\tau} = \frac{\exp(\beta(\mathbf{W}_q \mathbf{x}_t)^T (\mathbf{W}_k \mathbf{x}_\tau))}{\sum_{\tau'} \exp(\beta(\mathbf{W}_q \mathbf{x}_t)^T (\mathbf{W}_k \mathbf{x}_{\tau'}))} \quad (2.20)$$

here $\beta = \frac{1}{\sqrt{d}}$ which is scaling factor, τ represents the input sequence, and $\alpha_{t,\tau}$ gives us the attention weight for x_τ . Later, all the attention weights are used to combine the value vectors to generate the layer output at the current time step. It is mathematically expressed as :

$$z_t = \sum_{\tau} \alpha_{t,\tau} \mathbf{W}_v x_\tau = \sum_{\tau} \alpha_{t,\tau} v_\tau \quad (2.21)$$

The input vector z_t at time step t is utilized for the subsequent Transformer block. To bolster the model's capabilities, multi-head self-attention is implemented by incorporating multiple concurrent self-attention mechanisms that act upon the input sequence. The resulting outputs of each attention component are then amalgamated.

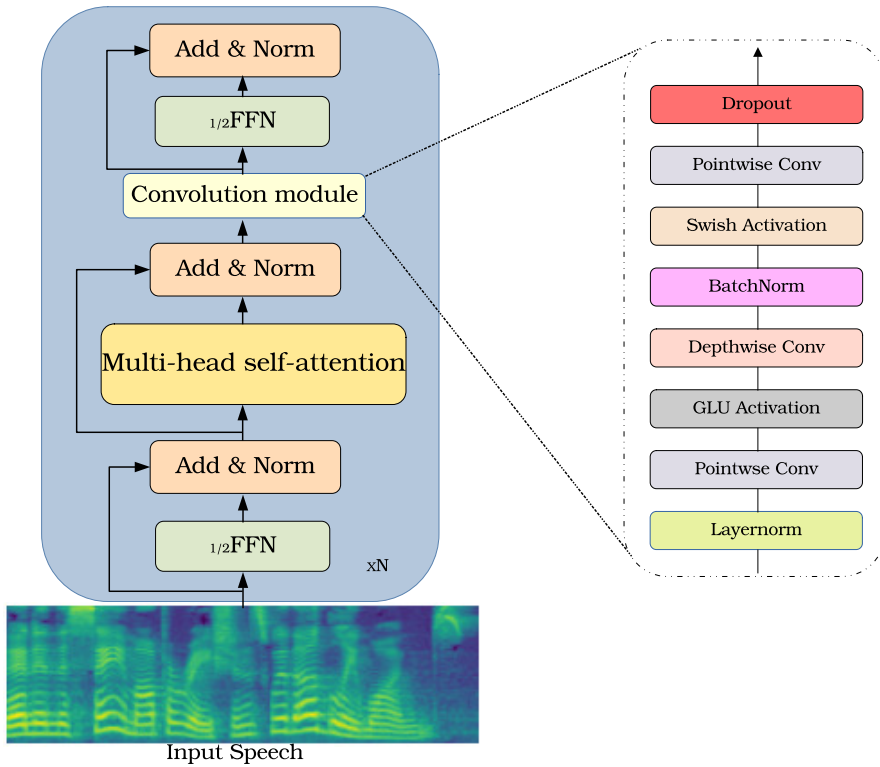


Figure 2.3: High-level architecture of conformer encoder

2.2.3 Convolution-Augmented Self-Attention Networks

Convolution-Augmented Self-Attention Networks is an advanced development in the field of neural networks, combining the strengths of Convolutional Neural Networks (CNNs) and Self-Attention mechanisms. This hybrid approach has been designed to leverage the benefits of

local and global feature extraction inherent in CNNs and Self-Attention mechanisms. A Convolutional Neural Network processes data with a grid-like topology (such as an image or a sequence) by applying a series of filters (or “kernels”) that detect local patterns. This operation is handy for tasks where the spatial or temporal relationships between nearby data points are important, such as image or speech recognition.

On the other hand, the Self-Attention mechanism, popularized by the Transformer model, calculates the relevance (or “attention”) of each data point in the sequence to every other data point. This mechanism excels in tasks where long-range dependencies between data points need to be captured, such as in natural language processing or time-series analysis. The Convolution-Augmented Self-Attention Network works on these two concepts, integrating the convolution operation into the self-attention mechanism. The same is shown in Figure 2.3. In this model, a convolutional layer is applied before the self-attention layer. The convolutional layer operates on local windows of the input sequence, capturing local patterns and dependencies. The processed sequence is then fed to the self-attention layer, which models the global dependencies between the different parts of the sequence. Mathematically, the output of the convolutional layer can be expressed as:

$$H' = \text{Conv}(H), \quad (2.22)$$

where H is the input sequence, and $\text{Conv}(\bullet)$ represents the convolution operation. The output H' is then fed to the self-attention layer:

$$O = \text{SelfAttention}(H'), \quad (2.23)$$

where O is the final output, and $\text{SelfAttention}(\bullet)$ represents the self-attention operation. This architecture allows the network to capture the local patterns and the global dependencies in the input data, making it a versatile tool for many sequence processing tasks. It has been successfully used in various domains, including ASR, where capturing both local (e.g., phonetic) and global (e.g., syntactic or semantic) information is crucial.

2.3 Neural Network Pre-Training: A Pathway to Self-Supervised Learning

As previously outlined in Section 2.1.3, it delineates on semi-supervised methodologies. Neural Network Pre-training and Self-Supervised Learning are powerful strategies employed in the field of deep learning, which has proven to significantly enhance the performance of various models, especially in scenarios where labeled data is scarce. The self-supervised Learning involves two principal paths, (1) pre-training, (2) fine-tuning. The focal point of this thesis is to address the self-supervised approach to solving low-resource problems. In this section, we initially present a succinct overview of neural network pre-training. We then review some of the most emblematic self-supervised pre-training architectures used in speech processing.

2.3.1 Background

Neural network pre-training is a foundational process that typically involves exposing the network to an extensive, often labeled dataset before transitioning into a fine-tuning phase on a smaller, task-specific dataset. This methodology is rooted in the idea that the broad patterns and features uncovered during this initial phase can be leveraged advantageously for the specific task. A crucial distinction between supervised and self-supervised pre-training concerning the nature and quantity of information encapsulated within the learned representation is observed. In a supervised context, the ensuing representations preserve only the information relevant to the task while disregarding nonessential data. Hence, representations sculpted via supervised pre-training are predominantly compatible with tasks akin to the initial supervised tasks. It explains why, for instance, representations trained on image classification prove beneficial for tasks such as object detection and semantic segmentation, reflecting the similarity of these tasks.

Conversely, representations derived from self-supervised tasks typically encompass various facets of the input data. They are not confined by the specific demands of a single task, particularly when the self-supervised task revolves around reconstructing input segments. This feature underscores the broad utility of representations derived from self-supervised learning, which typically exhibit greater adaptability and superior generalization across diverse tasks,

underscoring the broad applicability and versatility of self-supervised learning methodologies. The “pre-training followed by fine-tuning” paradigm has gained widespread adoption in semi-supervised learning, largely spurred by advances in computer vision. A notable illustration of this paradigm is supervised pre-training, wherein representations are gleaned through a supervised task. These representations, learned from tasks like image classification, have demonstrated their efficacy in object detection and semantic segmentation fields.

This methodology has been extended to other supervised tasks that eschew manual annotations, a concept known as self-supervised learning. Various tasks in the vision domain, including colorization, solving jigsaw puzzles, and inpainting, have been proposed to facilitate the self-supervised learning of visual representations. The approach has also marked significant success in natural language processing, where models like BERT and GPT are pre-trained on comprehensive text corpora before fine-tuning for specific tasks.

2.3.2 Self-Supervised Learning Approaches for Speech Representation

Methods for learning speech representations through self-supervised tasks can broadly be bifurcated into two categories: those grounded in contrastive learning principles and reconstructive (generative) learning.

Contrastive learning is a self-supervised learning strategy that revolves around distinguishing between similar and dissimilar instances in a dataset. The fundamental idea is to learn representations that are similar for instances from the same class (positive pairs) and different for instances from different classes (negative pairs). This strategy aims to derive representations of speech data that bring similar instances closer together in the representation space while pushing dissimilar instances further apart.

In a typical speech processing context, contrastive learning involves presenting a model with pairs of speech segments. Positive pairs consist of segments that are considered similar in some way, such as two segments from the same speaker or two segments sharing the same phonetic

content. Negative pairs, in contrast, are composed of segments that are considered dissimilar or unrelated, such as segments from different speakers or segments with different phonetic content. The objective of the model is to generate representations that minimize the distance between positive pairs and maximize the distance between negative pairs in the representation space. This is commonly achieved using a contrastive loss function, which encourages the model to reduce the similarity between representations of negative pairs and increase the similarity between representations of positive pairs. Through this method, the model learns to extract meaningful features from speech data that are useful for distinguishing between different speakers, different phonetic content, or other relevant factors. This makes contrastive learning a powerful tool for tasks such as speaker verification, speech recognition, and other downstream tasks that require robust and discriminative speech representations.

The effectiveness of contrastive learning in speech representation has been demonstrated in various applications, leading to improvements in the performance of many downstream tasks. It represents a significant step forward in leveraging large amounts of unlabeled speech data to learn meaningful representations, thereby contributing to speech processing technology advances.

Reconstructive learning, referred to as generative learning, is a type of self-supervised learning approach applied in speech representation learning. This methodology is centered around predicting or recreating parts of the input speech data. In a typical reconstructive learning scenario, a model is trained to predict missing or obscured parts of a speech signal or, in some cases, reconstruct the entire signal. This is often achieved by creating a task where parts of the input data are deliberately masked or removed, and the model is tasked with predicting the missing parts based on the remaining unmasked data. Techniques such as denoising autoencoders or masked language models in natural language processing are typical examples of this approach.

The objective of this strategy is to encourage the model to learn a representation that captures as much information as possible about the structure and content of the speech data. By doing

so, the model is expected to develop a comprehensive understanding of the various aspects of the speech signal, such as phonetic, prosodic, and speaker-specific characteristics. By learning to reconstruct the input data, models can extract meaningful features and patterns that can be used for various downstream tasks.

2.4 Past works in Multilingual speech recognition

Multilingual speech recognition aims to build ASR systems that can recognize speech from multiple languages within a single system or by sharing resources across languages. There has been considerable research and development in this area, with several notable works contributing to the advancement of multilingual ASR systems. Some key efforts include:

1. **Shared acoustic models:** Early approaches to multilingual ASR focused on sharing acoustic models across languages. In these methods, researchers trained a single acoustic model on data from multiple languages, leveraging similarities in phonetic and phonological structures. These shared models led to improvements in ASR performance for low-resource languages by borrowing knowledge from high-resource languages [60, 61].
2. **Multilingual bottleneck features:** Another approach involves training a deep neural network (DNN) on multiple languages to extract bottleneck features. These features, representing language-independent speech characteristics, can be used to train ASR systems for individual languages. This technique has shown significant improvements in performance for low-resource languages [62, 63].
3. **Transfer learning:** Transfer learning has emerged as a popular technique for leveraging pre-trained models from related tasks or languages. Researchers have used pre-trained models, such as multilingual BERT (mBERT) or XLSR (Cross-lingual Speech Representations), to improve ASR performance in low-resource languages by fine-tuning them with limited target language data [64, 65].
4. **Multilingual end-to-end ASR:** Recent developments in end-to-end ASR have been extended to multilingual scenarios. Researchers have experimented with training end-to-end ASR models on multiple languages simultaneously, using techniques such as joint

CTC/attention mechanisms and multi-task learning. These approaches have shown promising results in improving ASR performance across various languages [66, 67, 68].

5. **Cross-lingual adaptation:** Cross-lingual adaptation involves using knowledge from one language to improve recognition in another. Researchers have explored techniques such as cross-lingual transfer learning, language-adaptive training, and unsupervised adaptation to improve ASR performance in low-resource languages using data from related high-resource languages [69, 70].
6. **Code-switching:** Given the prevalence of code-switching in multilingual environments, researchers have also explored methods to handle code-switching in ASR systems. These approaches include language identification, language modeling, and deep learning techniques to improve recognition performance in code-switched speech [71, 72].

2.5 Challenges in building ASR systems for Indian Languages

Automatic Speech Recognition (ASR) systems play a vital role in processing speech signals by converting them into text format. With over 7000 languages spoken globally, 90% of them are considered low-resource languages, representing the speech of over 3 billion people. Most of these low-resource languages are not English. A language is deemed low-resource when there is limited web presence or a lack of availability of speech, text, transcribed data, linguistic expertise, or a pronunciation dictionary [73]. In India, approximately 2000 languages are spoken, including dialects, sociolects, pidgins, accents, etc., of which only 22 are officially registered, and 13 have different orthographies. Additionally, approximately 74% of the country's population is literate, and most only read and write in their native language. As a result, most Indians are excluded from the mainstream, as only 12.16% of the population can read and write in English. In such scenarios, speech interfaces that use vernacular languages are advantageous.

In the context of Indian scenarios, developing an Automatic Speech Recognition (ASR) system presents unique challenges and opportunities. Among the 22 majorly spoken Indian languages, only a few of them have Automatic Speech Recognition (ASR) capability. A substantial amount of annotated speech data is required to build an ASR system that performs well. However,

most Indian languages are considered low-resource, and annotating data is challenging, time-consuming, and expensive. Despite a large volume of speech content in Indian languages being available online for public use, developing ASR systems for Indian languages has been complex. It has received little attention over the last several decades. However, since the launch of the Digital India Mission in 2015, efforts have been made to build speech recognition technologies capable of handling low-resource languages.

The primary challenges in implementing ASR for Indian scenarios include:

1. Orthography

- India's linguistic diversity also extends to its writing systems. Several scripts are used to represent the numerous languages spoken across the country. Understanding these writing systems is essential for tasks of building speech systems. Some of the prominent writing systems in India include:

Devanagari: Used for languages such as Hindi, Marathi, Nepali, and Sanskrit. Devanagari is one of the most widespread writing systems in India. It is an abugida ¹ script, where consonants have an inherent vowel sound that can be altered using diacritical ² marks.

Bengali-Assamese: This script is employed for Bengali, Assamese, and several other languages which are spoken in eastern India and Bangladesh. Like Devanagari, it is also an abugida script featuring distinct consonant symbols and diacritical marks for vowels.

Gurmukhi: Primarily used for the Punjabi language, the Gurmukhi script is another abugida writing system, with characters representing consonants and vowel signs modifying the inherent vowel sound.

Gujarati: Utilized for the Gujarati language, the Gujarati script is closely related to Devanagari but has some unique characters and diacritical marks.

¹An abugida, also known as an alphasyllabary, is a type of writing system in which consonant symbols (or base characters) carry an inherent vowel sound. This inherent vowel can be modified or suppressed using additional diacritical marks or symbols.

²Diacritical marks, also known as diacritics or accents, are supplementary symbols or glyphs added to a base letter in a writing system to indicate a modification or variation in the pronunciation, meaning, or function of the letter.

Tamil: The Tamil script is employed for the Tamil language and is characterized by its rounded letter shapes. As a syllabic alphabet or abugida, each consonant symbol carries an inherent vowel sound that can be modified using vowel signs.

Telugu: Used for the Telugu language, the Telugu script is another example of an abugida writing system, with unique symbols for consonants and vowel signs for modifying the inherent vowel sound.

Malayalam: The Malayalam script is employed for the Malayalam language spoken in the southern Indian state of Kerala. It is an abugida script with distinct consonant symbols and diacritical marks for vowels.

Kannada: The Kannada script is used for the Kannada language, primarily spoken in the southern Indian state of Karnataka. This abugida writing system features unique symbols for consonants and diacritical marks for vowels.

Oriya: Utilized for the Odia language, the Oriya script is another abugida system with distinct symbols for consonants and diacritical marks for vowels.

Perso-Arabic (Urdu): The Perso-Arabic script, adapted for the Urdu language, is an alphabet with consonant and vowel symbols. Unlike the other scripts mentioned, which are written from left to right, the Perso-Arabic script is written from right to left.

2. Multilingual society

- India is a multilingual country with rich linguistic diversity. As a result, an Automatic Speech Recognition (ASR) system must be equipped to support all official Indian languages to ensure effective communication and accessibility for everyone.

3. Limited Resources

- Many Indian languages suffer from a scarcity of annotated speech data, which is essential for training ASR models. Developing or expanding existing datasets is necessary to ensure the models are well-equipped to handle the nuances of each language.

4. Code-switching

- Indian speakers often switch between languages during conversations, which adds complexity to ASR system development. Handling code-switching in speech recognition requires advanced modeling techniques and a thorough understanding of the languages involved.

5. Accents and dialects

- There are significant variations in accents and dialects within each Indian language. ASR systems must be designed to handle these variations and recognize speech from speakers with different regional accents and dialects.

6. Socio-cultural factors

- Indian languages are heavily influenced by socio-cultural factors, such as caste, religion, and geography. These factors can impact language use and introduce variations that are difficult for ASR systems to handle.

ASR systems for Indian languages: researchers faced challenges due to the lack of resources and smaller datasets. Table 2.1, reveals a noticeable scarcity of freely available data for building speech recognition systems in Indian languages, particularly when compared to the availability of such data in English. The table shows the language, duration in hours, and mode of speech in each corpus. The analysis reveals that most of the available speech corpora for English ASR systems in the literature are read speech, with the exception of Gigaspeech and People's Speech which consist of read and spontaneous speech collected from public forums like YouTube. It is worth noting that Gigaspeech [5] and People's Speech [4] are built using data extracted from sources such as YouTube and other public forums. Building high-quality ASR systems for languages with limited resources poses a significant challenge. The table highlights the lack of freely available data for building speech recognition systems in Indian languages, particularly compared to the availability of such data in English.

Table 2.1: Contrastive comparison of existing speech database reported in the literature

Dataset	Language	Hours	Mode
WSJ [74]	English	80	Read
Librispeech [75]	English	1,000	Read
Gigaspeech [5]	English	10,000	Read & Spontaneous
People’s Speech [4]	English	30,000	Conversational
TIMIT [76]	English	5.4	Read
Common voice [77]	English	2953	Read
Switch board [78]	English	260	Conversational
MSR Telugu [79]	Telugu	50	Conversational & Read
MSR Tamil [79]	Tamil	50	Conversational & Read
MSR Gujarati [79]	Gujarati	50	Conversational & Read
Bengali Open SLR [80]	Bengali	120	Read
Gramvani [81]	Hindi	100	Spontaneous
MUCS Hindi [82]	Hindi	100	Read
MUCS Odia [82]	Odia	100	Read
MSR Marathi [82]	Marathi	109.3	Read
MILE Kannada [83, 84]	Kannada	350	Read
MILE Tamil [83, 84]	Tamil	350	Read

2.6 Databases

This section delineates in-depth specifics of the different databases employed for the experiments conducted in this thesis. A summary of the specifics related to each dataset is presented in Table 2.2 for a more structured and accessible understanding.

Table 2.2: Summary of databases used in thesis (where R, C, S indicates Read, Conversational, and Spontaneous mode speeches (speaking styles))

Database	Attributes	Languages						
		Te	Ta	Kn	Hi	Ma	Gu	Od
MUCS [82]	Duration (hours)	50	50	-	100	100	50	100
	Sampling Rate (kHz)	16	16	-	8	8	16	8
	Speaking Style	R&C	R&C	-	R	R	R&C	R
MILE Kannada [83, 84]	Duration (hours)	-	-	350	-	-	-	-
	Sampling Rate (kHz)	-	-	16	-	-	-	-
	Speaking Style	-	-	R	-	-	-	-
MILE Tamil [83, 84]	Duration (hours)	-	150	-	-	-	-	-
	Sampling Rate (kHz)	-	16	-	-	-	-	-
	Speaking Style	-	R	-	-	-	-	-

2.6.1 Multilingual and code-switching ASR challenges for low-resource Indian languages (MUCS) Corpus

The MUCS Speech corpus [82] provides approximately 450 hours of data across six languages: Hindi, Marathi, Odia, Telugu, Tamil, and Gujarati. The duration varies among these languages, with 50 hours for Telugu, Tamil, and Gujarati and 100 hours for Hindi, Marathi, and Odia. The sampling rates also differ, with 16 kHz for Telugu, Tamil, and Gujarati and 8 kHz for Hindi, Marathi, and Odia. In terms of speaking styles, the MUCS database contains Read (R) and Conversational (C) speech data for Telugu, Tamil, and Gujarati. In contrast, only Read (R) style is available for Hindi, Marathi, and Odia. The Hindi, Marathi, and Odia data were collected from native speakers performing a reading task. The choice of speakers and text aimed to cover

different language variations to ensure better generalizability of the ASR systems developed using this data.

2.6.2 MILE Corpus (Kannada,Tamil)

The MILE (Medical Intelligence and Language Engineering Lab) Speech corpus [83, 84] is a valuable resource for developing ASR systems for Tamil and Kannada languages. This publicly available transcribed speech data provides significant data for both languages. The corpus consists of 150 hours of data for Tamil and 347 hours for Kannada. The Tamil data was recorded from 531 native Tamil speakers, while the Kannada data was recorded from 915 native speakers of Kannada. This wide range of speakers ensures diverse accents and speaking styles are captured, contributing to a more robust ASR system. All the MILE Speech corpus data was recorded in a clean, noise-free environment using USB microphones. This high-quality recording setup ensures clear and consistent audio data for training and testing ASR models. The data is divided into (i) training data - 152 hours for Tamil and 275 hours for Kannada, and (ii) test data - 65 hours for Tamil and 72 hours for Kannada.

2.7 Summary and Conclusion

This thesis comprehensively explores various themes central to the Automatic Speech Recognition (ASR) field. The study focuses on three primary aspects: (i) the gathering of a speech corpus through a crowd-sourced approach, (ii) the investigation of self-supervised representation learning for model training, and (iii) the exploitation of different acoustic models within a multilingual speech recognition framework.

A review of the existing literature reveals a noteworthy void in the construction of speech recognition tools specifically designed for Indian languages. Recognizing this gap, the research aims to address it by establishing a user-friendly platform. This unique platform encourages the collection of a large-scale corpus for speech recognition tasks using a crowd-sourced approach. This approach facilitates a broad and diverse database from various speakers, thus ensuring a

more representative and accurate speech recognition system.

This thesis embarks on an in-depth investigation of a pre-trained model built entirely from scratch utilizing a self-supervised learning approach. This method enables the model to learn directly from raw data, eliminating the need for extensive labeled data and fostering a more efficient learning process. In addition, the study probes into the captivating realm of language adaptation in speech recognition systems. It investigates the effects of using a pre-trained model, originally trained in one language, on its subsequent performance when fine-tuned to other languages. This exploration might unlock the key to creating more efficient and adaptable multilingual speech recognition systems.

The study also explores the intriguing concept of the Joint Acoustic Model (JAM). It capitalizes on a common phone set (CPS) and a common label set (CLS), aiming to standardize the speech recognition process across various languages. This could lead to more accurate recognition and interpretation of spoken language, irrespective of its linguistic origin. The thesis then turns its attention to an array of diverse acoustic models. These models are designed to advance monolingual and multilingual ASR systems, emphasizing enhancing accuracy and reducing language biases. This approach has the potential to increase the accuracy of recognition and interpretation of spoken language, regardless of its linguistic origin. Following this, the thesis focuses on a spectrum of diverse acoustic models. These models are meticulously crafted with the purpose of advancing both monolingual and multilingual ASR systems, placing a strong emphasis on improving accuracy while simultaneously minimizing language biases.

The subsequent chapter in this thesis will provide a detailed explanation of the strategies adopted for the collection of a large-scale speech corpus. It presents the novel idea of using a crowd-sourced approach, an effective method for obtaining a vast and diverse dataset crucial for developing a robust and accurate ASR system.

Chapter 3

Crowd-sourced strategies for the collection of a large-scale speech corpus

3.1 Introduction

The tremendous advancements in deep learning have allowed us to build speech recognition systems robust to language, speaker, and environmental variations. However, building high-quality automatic speech recognition (ASR) systems is still challenging. To achieve state-of-the-art performance, it is crucial to have a database that covers diverse speakers from different environments. Over the past decade [75, 74], academia and industry have made many efforts to create corpora for English speech recognition tasks. The performance of speech recognition systems depends mainly on the quality and size of training corpora. A corpus is considered low-resource if it has a limited web presence and lacks linguistic expertise and digital resources, such as acoustic and text. Building ASR systems for Indian languages is a challenge because most of these languages are low-resourced. To achieve human parity [85] in Indian language ASR systems, a significant amount of training data is required. Despite the availability of many resources for Indian languages, collecting annotated speech data from them is still challenging and expensive.

In [86, 87, 88, 89, 90, 91], an attempt to build annotated speech datasets using traditional data acquisition techniques, researchers have made smaller-scale efforts in various studies. Researchers from Microsoft released 45 hours each of speech data for Telugu, Tamil, and Gujarati languages as part of the Interspeech 2018 multilingual challenge [79]. With the growing

interest to collect data at a larger scale, the Speech Lab at the Indian Institute of Technology, Madras (IIT-Madras) has conducted ASR challenges in three Indian languages: Hindi, Tamil, and Indian-English¹. As part of the challenge, 490 hours of transcribed speech data were made available for all three languages under the national language translation mission (NLTM) by MeitY, Government of India. All these corpora listed above have been collected using traditional data acquisition techniques. The Microsoft Research team has also utilized crowd-sourced strategies [92] for Marathi language ASR at a smaller scale. These efforts are aimed at encouraging the advancement of ASR in Indian Languages.

Collecting quality speech data is challenging for several reasons. First, speech data is complex and requires significant expertise and resources to annotate. It involves transcribing audio data and aligning it with the corresponding text, which is time-consuming and labor-intensive. Second, speech data is highly variable, and differences in speakers, dialects, accents, and environmental conditions can cause variations in speech. This variability makes building models that can generalize well to new speakers or environments difficult. Third, for many languages, especially low-resource ones, speech data is scarce, which limits the training data available to build ASR systems. Finally, collecting speech data at scale involves a lot of complexity and is expensive, especially when working with diverse languages and dialects. These factors collectively make collecting quality speech data a challenging task.

This thesis presents an innovative method for collecting speech data using crowdsourcing techniques. Crowdsourcing is a process that involves outsourcing tasks to a vast group of individuals known as the “crowd.” The term “crowdsource” is derived from the combination of two words: “crowd” and “outsourcing,” which implies the outsourcing of tasks to online-based workers. Therefore, the primary goal of this study was to harness the potential of crowdsourcing by utilizing the collective intelligence of diverse individuals to collect speech data. Crowdsourced methods can address the challenges of obtaining high-quality speech data by facilitating the collection of a large amount of data from a diverse population in a relatively short period and at a lower cost than traditional methods. This approach involves recruiting a large group of

¹<https://sites.google.com/view/indian-language-asrchallenge/home>

non-expert individuals compensated for their contributions to recording speech samples in their native language or dialect. The crowdsourcing approach also allows data collection from speakers who may not be easily accessible or available through traditional means. However, one of the challenges of using a crowdsourced approach is ensuring the quality of the collected data. Several techniques can address this challenge, such as providing clear instructions to contributors, validating data through quality control measures, and using machine learning techniques to filter out low-quality data. Despite its limitations, the crowdsourced approach has proven effective in gathering data for low-resource languages and dialects, contributing to the development of robust speech recognition systems.

Technology plays a crucial role in creating a clean, annotated, crowdsourced corpus, as it enables engagement with a vast number of people to perform complex and creative tasks at a significantly lower cost. This process involves collecting information or opinions from a large group, typically through the Internet and smartphones. It offers cost savings, speed, and the ability to work with people possessing skill sets that an in-house team within a company may not have. Researchers have made significant efforts to use crowdsourcing methods to collect data for global English and European languages, as evidenced by previous studies [87, 93]. Recently, Google has partnered with local universities and communities in Southeast Asia, Africa, Europe, and South America to release 38 data sets for building text-to-speech (TTS) and automatic speech recognition (ASR) applications [88]. This crowdsourcing approach has contributed to high-quality data. Moreover, the recent release of People’s Speech Corpus, a large-scale diversified English speech recognition dataset, for commercial usage [4], is the most extensive open dataset to date that can be used for both research and commercial purposes under the Commons Attributions Share-Alike (CC-BY-SA) licenses.

3.2 Overview of Telugu Language

In previous studies, speech corpora for several Indian languages, such as English, Hindi, and Tamil, have been made available online [79, 94, 95, 96, 97]. The size of these datasets is approximately 200 hours for each language, providing valuable resources for researchers and

developers of automatic speech recognition (ASR) systems. This thesis presents various strategies which could be incorporated while collecting speech corpus using the crowd-sourcing approach. So here we have opted for one of the low-resource Indic languages, which is Telugu.

Telugu is India's fourth most widely spoken language, with 80 million speakers worldwide. However, to the best of our knowledge, the largest publicly available Telugu speech corpus is 50 hours, as collected by [79], which is inadequate for state-of-the-art ASR architectures that require larger amounts of training data. Telugu, a Dravidian language, is predominantly spoken in two South Indian states, namely Telangana and Andhra Pradesh. The language has also extended its influence in neighboring regions and is spoken by individuals in regions of Tamil Nadu, Karnataka, Orissa, and Chattisgarh, which share their borders with the Telugu-speaking states. A visual representation of the geography of Indian states and the languages spoken in each state is provided in Figure 3.1.

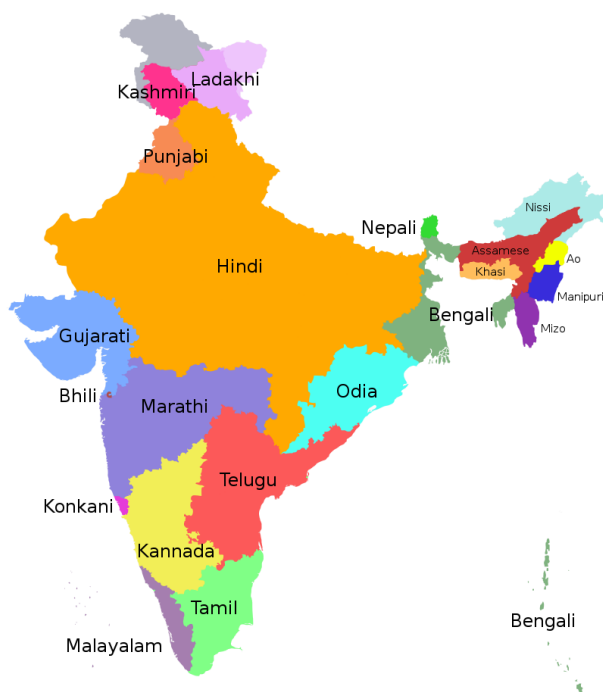


Figure 3.1: Languages spoken across Indian Subcontinent

It has been observed that there exist notable distinctions among Telugu dialects spoken by different groups of individuals. In light of this observation, our research initiative endeavors to capture the diversity of Telugu dialects by sourcing speech samples from individuals hailing

from three distinct geographical regions, namely Coastal Andhra (CA), Rayalaseema (RY), and Telangana (TG). As highlighted earlier, the dearth of annotated Telugu speech corpora has stimulated our endeavor to develop a framework that enables recording a substantial and well-balanced corpus comprising spontaneous and natural interactions sourced from a diverse group of speakers.



Figure 3.2: Three different dialects among the Telugu-speaking regions

3.3 Proposed data acquisition workflow for speech corpus collection

In this subsection, we delineate the methodology employed for collecting the Telugu corpus ², including the pre-processing steps and the verification procedures implemented to ensure the quality of the collected data. The same is depicted in Figure 3.3.

1. **Sampling:** We use FFmpeg library ³ to make sure that all the collected audio is sampled at 16kHz and is mono-channel.
2. **Fragmentation:** In this step, the speech/audio signal is truncated based on the short, long, and extended pauses. For performing the fragmentation, we use WebRTC (Real-Time Communication for the web) - Voice Activity Detector (VAD) ⁴ library. WebRTC-VAD

²For replication of the CSTD methodology, kindly refer to the link https://github.com/mirishkarganesh/tallip/tree/main/CSTD_pipeline

³<https://ffmpeg.org/>

⁴<https://github.com/wiseman/py-webrtcvad>

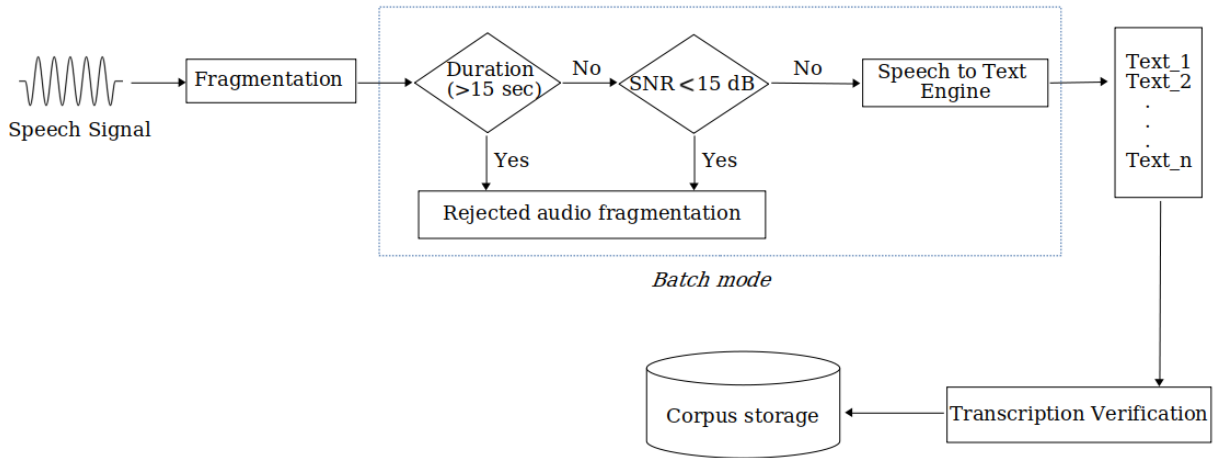


Figure 3.3: The working pipeline developed for corpus collection task

is a module that is part of WebRTC⁵ developed and maintained by Google. Previously, [98] has shown that this open-source module can be effectively used to quickly fragment the raw speech in order to collect speech corpus. Therefore, we opt to use WebRTC-VAD to collect this corpus. A snippet of fragmentation is depicted in Figure 3.4.

3. **Duration Filter:** A duration filter is commonly used in building annotated speech corpora, where audio samples are divided into smaller segments, called chunks or utterances, for easier annotation. After the audio is segmented, it is common practice to filter out chunks that are too long to be useful for annotation purposes. In practice, the audio chunk greater than 15 seconds are filtered out. So that the resulting corpus will consist of more homogeneous audio segments, which human annotators or machine learning algorithms can more easily annotate. This can lead to higher-quality annotations and better performance in speech recognition and other related tasks. This is a standard practice [77] for building an annotated speech corpus.
4. **SNR Filter:** Since it is a crowd-sourced database, it is possible for audio files to be corrupted due to the injection of noise from external sources, which can affect the quality of the audio samples. The chunked audio files are passed through the SNR filter to avoid high noise content in the database we collect. The threshold of a filter is 15 dB. The

⁵<https://webrtc.org>

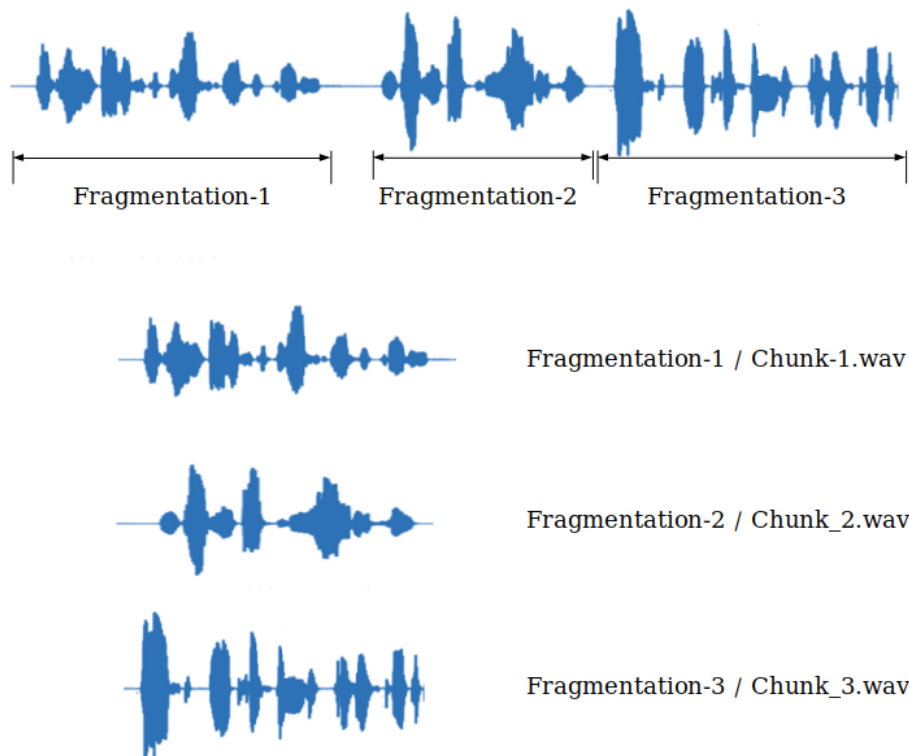


Figure 3.4: Illustration of the Speech fragmentation algorithm

role of this filter is to filter out the audio samples less than 15 dB. We use Waveform Amplitude Distribution Analysis - Signal-to-Noise-Ratio (WADA-SNR) to perform this operation [99].

5. **Rough Transcripts:** In this step, we use a Speech to Text engine to get rough transcripts of the accepted audio samples. We showed the pipeline used in Figure 3.3. The modules present in the blue dashed box work in batch mode.
6. **Verification:** The rough transcripts are sent for four level verification process by human intervention. Finally, the verified transcripts are stored in the database.

The above-mentioned data acquisition workflow for speech collection is referred to as the CSTD pipeline.

3.4 Guidelines followed throughout the data collection process

Maintaining consistency in the data collection process is essential to ensure the quality and reliability of the resulting speech corpus. One key aspect of consistency is ensuring that both the audio contributors and transcribers are given the same guidelines and instructions. In this case, the guidelines include the requirement that the audio contributors should have native speaker-level fluency in Telugu, which is the language being recorded. This ensures that the audio samples are of high quality and represent natural speech patterns and accents. Additionally, by providing consistent guidelines to both the audio contributors and transcribers, it helps to ensure that the transcriptions accurately reflect the speech in the audio samples, which is essential for building a high-quality speech corpus.

3.4.1 Audio Guidelines

The audio guidelines are designed to ensure that the audio samples collected are of high quality and accurately represent the spoken language. Some of the key guidelines include:

1. Ensuring that the recording device is free from any external glitches, such as faults in the microphone or background noise, which could impact the audio quality.
2. Speaking clearly, distinctly, loudly, and avoiding whispering, to ensure the speech is easily understandable and can be transcribed accurately.

3.4.2 Transcription Guidelines (see Figure 3.5)

The instructions passed to the crowd while transcribing are as follows:

1. Carefully listening to the audio file to ensure that all words and phrases are transcribed accurately.
2. Editing and correcting the transcript, including adding punctuation marks to match the speech.

ఉపయోగించాల్సిన వ్యాకరణ చిహ్నాలు :

Symbol	Telugu name	English name	Usage telugu
(.)	బిందువు	Full stop	వాక్యం పూర్తి అయినప్పుడు పూర్తి అయినట్లుగా సూచించినదానికి బిందువు పెడతారు.
(,)	వాక్యఅంశ బిందువు	Comma	చెప్పవలసిన అంశం ముగియనప్పుడు, అసమాపక క్రియలను వాడే వాక్యం రాస్తున్నప్పుడు వాక్యం బిందువు లేదా కామా ఉపయోగిస్తాము.
(;)	అర్ధ బిందువు	Semi colon	ఒక్క పెద్ద వాక్యం లో భాగంగా ఉండే చిన్న వాక్యాలు చివర అర్ధ బిందువు వస్తుంది.
(:)	న్యూన బిందువు	Colon	వాక్యాలు లో వరుసగా కొన్ని పదాల పట్టిక ఇచ్చుట ముందు న్యూన బిందువు ఉపయోగిస్తారు .
(" ")	అనుకరణ చిహ్నాలు	Quotation marks	ఒకరు అన్న మాట ఇంకొకరు చెప్పుచున్నప్పుడు, వేరే గ్రంథం నుండి తీసిన వాక్యాలు చెప్పుచున్నప్పుడు అనుకరణ వాడతారు.
(?)	ప్రస్నార్థకము	Question marks	ఏదైనా విషయాన్ని గురించి వేరే వాళ్ళు అడిగేటప్పుడు ఆ వాక్యం చివర ప్రస్నార్థకము వస్తుంది.
(!)	ఆశ్చర్యార్థకము	Exclamatory Mark	ఆశ్చర్యాన్ని, ఆనందాన్ని, భయాన్ని, వింతని, మెచ్చుకోలును తెలిపే పదాల చివర ఆశ్చర్యార్థకము ను వాడతాము.
(-)	పొడవు గీత	EmDash/ Hyphen	వాక్యంలో వచ్చే విషయాలు వివరణ ఇచ్చేటప్పుడు పొడవు గీత వాడతాము.

Figure 3.5: Instructions to be followed while performing the transcription task to achieve golden standards.

3. Transcribing the audio verbatim in Telugu, including any numbers and words in other languages.
4. Not translating any words and keeping the transcriptions as they are in the audio verbatim.
5. Keeping informal words and any false starts or repairs as they are in the audio verbatim.
6. Using only the punctuation marks provided in the list and adding them where necessary.

Overall, these guidelines are designed to help ensure that the resulting speech corpus is high quality and accurately represents the spoken language. By providing clear and consistent instructions to audio contributors and transcribers, the project can create a useful and reliable speech corpus for research and development purposes.

3.5 Approaches in IITH-Crowdsourced Telugu Data corpus

This section will discuss the various approaches employed during the collection of the CSTD speech corpus. The corpus comprises a diverse range of speaking styles, including read, spontaneous, and naturalistic. Given that the corpus was obtained through crowd-sourcing, the first step in its collection was identifying and selecting an appropriate crowd. This crowd was recruited from various sources, such as agencies, educational institutions, companies, Telugu Kootami⁶ (a non-profit organization promoting Telugu language and culture), as well as non-resident Indians (NRIs) living abroad.

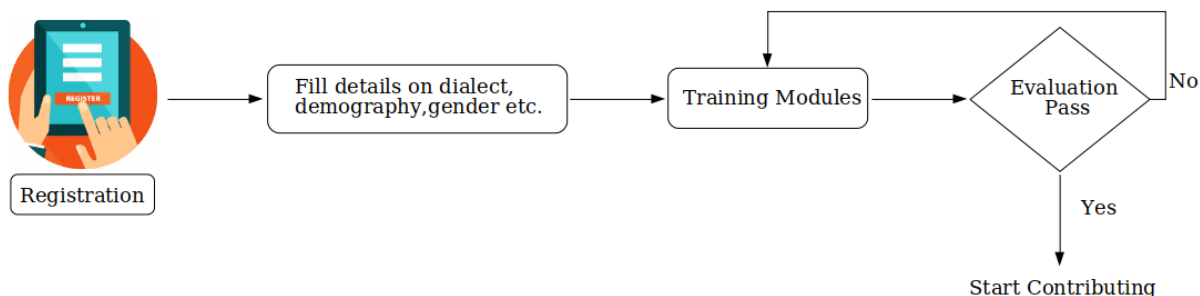


Figure 3.6: The workflow of crowd procurement

⁶<https://telugukootami.org/>

The procured crowd is not associated with any commercial forum or a tech-based company. Once the crowd is procured, they have been asked to register with their credentials, i.e., age, gender, dialect, and demographic information, to a registration portal. We train our crowd-workers accordingly. Upon completion of the training, the contributors are assigned a simple task as an evaluation. Based on the evaluation, the contributor is certified with a score for audio contribution and transcription verification tasks. If the contributor fails to clear the training module, he/she will be sent back for training. The flow of crowd scrutiny is depicted in Figure 3.6.

The IIITH-CSTD Corpus has been collected using three distinct approaches, each with a specific purpose and methodology.

1. **Approach 1** involves the collection of spontaneous speech through web and mobile applications from the crowd. This approach is useful for capturing natural speech patterns and variability in different types of speech, including regional and dialectal variations. This approach is often used in large-scale data collection efforts, such as those aimed at building speech recognition.
2. **Approach 2** involves the collection of conversational speech from open resources such as YouTube and podcast channels. This approach is useful for capturing informal and unscripted speech patterns, such as those used in everyday conversations between friends and family members.
3. **Approach 3** involves the collection of read speech through WhatsApp campaigns. This approach is useful for capturing speech patterns that are more controlled and predictable, such as those used in scripted presentations or announcements. This approach may also be useful for capturing speech from speakers who may not be comfortable with spontaneous or conversational speech, such as those with speech impairments or non-native speakers.

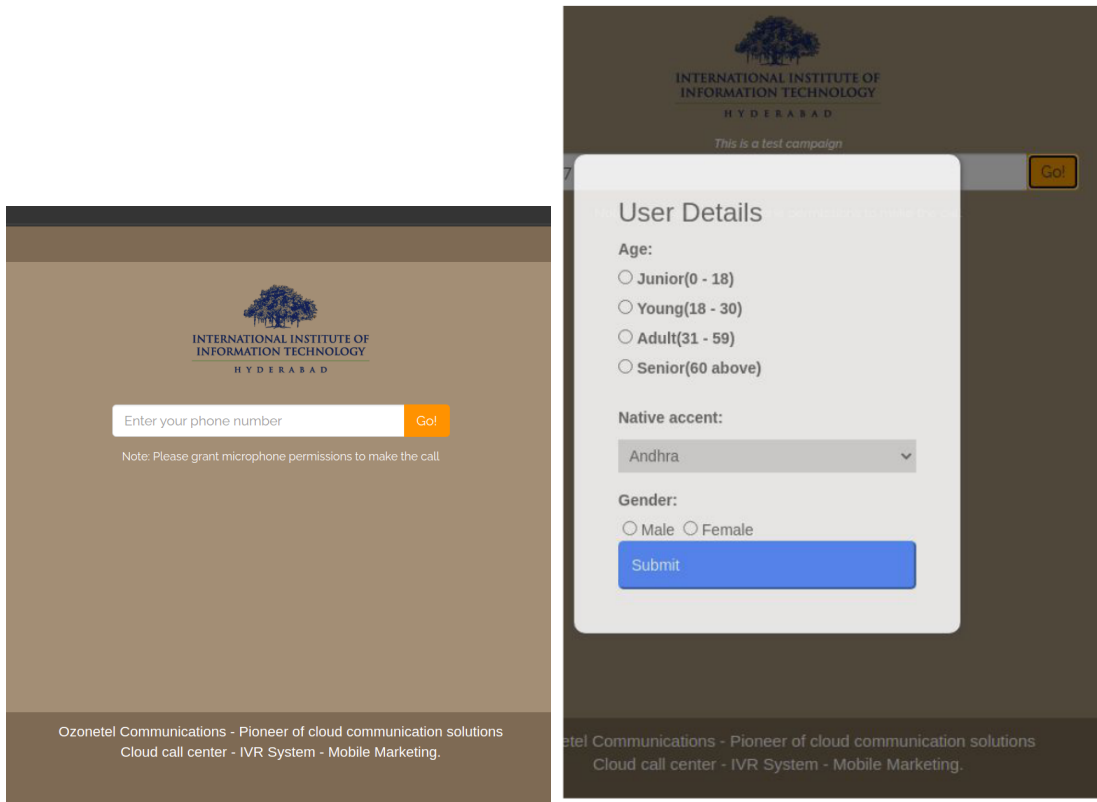
3.5.1 Approach 1: Spontaneous mode speech (Collection through web and mobile applications from the crowd)

The first approach involves collecting spontaneous speech data from crowd-workers on specific topics (approximately 2000). The crowd-workers can use either their mobile phone or laptop

to contribute their audio recordings. To ensure audio quality, guidelines are provided to the participants. In this approach, participants have the freedom to choose the topics they want to talk about and what to say. This makes the data more natural and diverse because it reflects the participants' personal interests, experiences, and perspectives. However, it is important to provide clear guidelines and instructions to ensure the quality and consistency of the data collected. This can include guidance on the recordings' format and the topics that should be avoided. In this method, topics related to caste, religion, and similar sensitive issues are avoided.

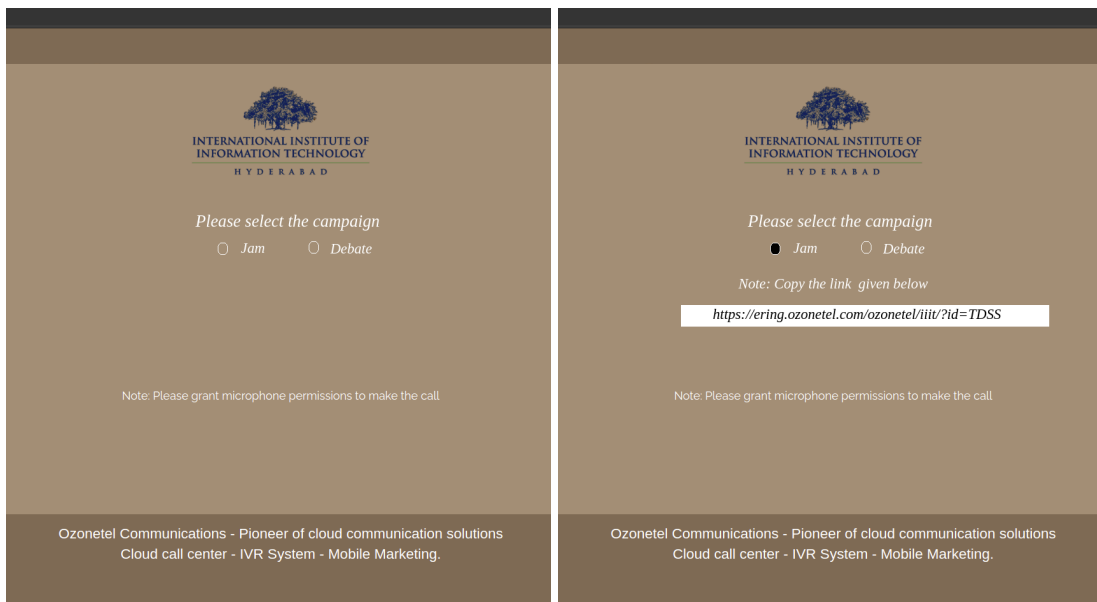
A platform has been developed wherein a participant has to register by providing a phone number, age, gender, and demographic information before the start of their Just a Minute (JAM) or debate campaign session. For the JAM sessions, the user count is restricted to 1, but for debate sessions, a minimum of 2 users are assigned to a room for conversations. The mobile number is entered on the home page when the participants open it, as shown in Figure 3.7(a). Once the mobile number is submitted, the participants are redirected to the demographic information page as shown in Figure 3.7(b). The user is asked to enter his or her credentials, i.e., age, native accent (like Coastal Andhra (CA), Rayalaseema (RY), Telangana (TG)), and gender. The age groups of options from junior level (0-18 years), young (18-30 years), adults (31-59 years), and seniors (60 years and above) are provided in the entries. Several students and faculties from different universities and colleges have been approached across the Telugu-speaking Andhra Pradesh and Telangana states. Later, based on the campaign, i.e., Debate / JAM, the URL links (see Figure 3.7(c)) are created and shared with the participants. URLs shared with the participants are supported by mobile and web-based applications based on their campaign (see Figure 3.7(d)). The same is depicted in Figure 3.7.

In this approach, once the participant completes the registration process, they are presented with a list of topics to choose from. The campaign is designed in such a way that each topic is only assigned to one participant/debate room to ensure that there is no repetition or duplication. By preventing topic repetition, the approach can ensure that the data collected is diverse and representative of a range of perspectives and experiences. It can also mitigate potential biases that may arise if certain topics are repeatedly chosen or avoided by participants. Crowd-sourced



(a) Campaign links shared to the participants in the crowd.

(b) Demographic information entries of the crowd from his/her mobile or laptop.



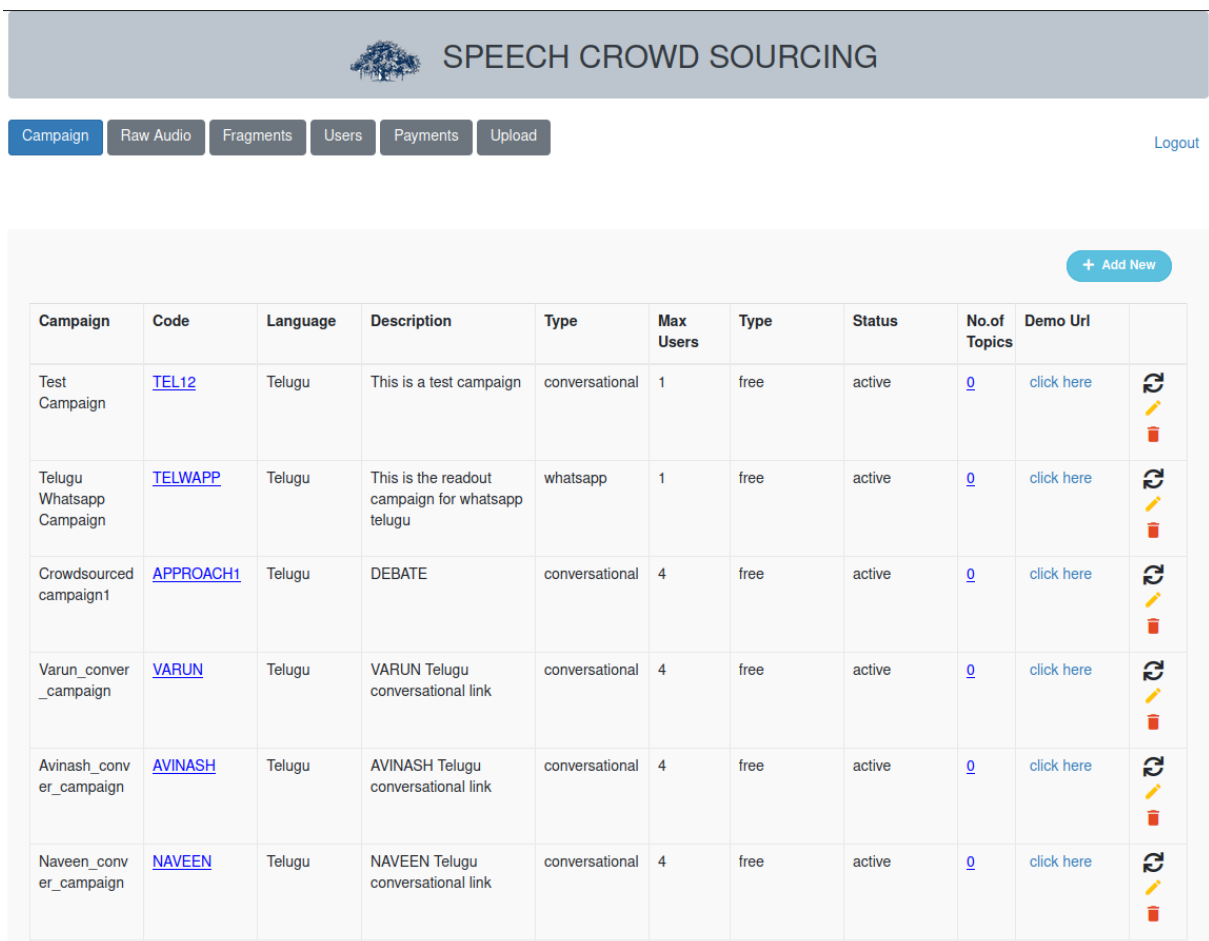
(c) Selecting the campaign

(d) link for the selected campaign

Figure 3.7: Illustration of the workflow for Approach 1

data collected from the contributors is stored on the Google Cloud at the back end. The recording interface supports both the mobile and web versions of the application with a VOIP-based framework. The VOIP-based technology has been obtained from Ozonetel Communication Pvt. Ltd. It captures the users' recordings at 16 kHz sampling frequency using the audio codecs of 16-bit Pulse code modulation (PCM).

An admin console has been designed to monitor the data collection process. Here admin console serves as a "human in the loop" mechanism, providing a way for the administrator to review and approve the data collected before it is sent to the CSTD pipeline (see section 3.3). The role of the human (administrator) is to review, create, delete, or modify the list of campaigns at any point of time if it doesn't meet the guidelines provided. The snippet of the admin console is shown in Figure 3.8.



The screenshot shows the 'SPEECH CROWD SOURCING' administrator console. At the top, there is a navigation bar with tabs for 'Campaign', 'Raw Audio', 'Fragments', 'Users', 'Payments', and 'Upload'. A 'Logout' link is visible on the right. Below the navigation bar is a table with the following columns: Campaign, Code, Language, Description, Type, Max Users, Type, Status, No. of Topics, Demo Url, and a column with action icons (refresh, edit, delete). The table contains six rows of campaign data.

Campaign	Code	Language	Description	Type	Max Users	Type	Status	No. of Topics	Demo Url	
Test Campaign	TEL12	Telugu	This is a test campaign	conversational	1	free	active	0	click here	
Telugu Whatsapp Campaign	TELWAPP	Telugu	This is the readout campaign for whatsapp telugu	whatsapp	1	free	active	0	click here	
Crowdsourced campaign1	APPROACH1	Telugu	DEBATE	conversational	4	free	active	0	click here	
Varun_conver_campaign	VARUN	Telugu	VARUN Telugu conversational link	conversational	4	free	active	0	click here	
Avinash_conver_campaign	AVINASH	Telugu	AVINASH Telugu conversational link	conversational	4	free	active	0	click here	
Naveen_conver_campaign	NAVEEN	Telugu	NAVEEN Telugu conversational link	conversational	4	free	active	0	click here	

Figure 3.8: Administrator console for monitoring the campaigns

As mentioned earlier, in this approach, the final raw data that humans validate is passed through the CSTD pipeline for automatic validation of duration, SNR parameters, and raw transcripts for the obtained speech segments. This suggests that the data is subjected to a rigorous quality check before it is considered for further analysis.

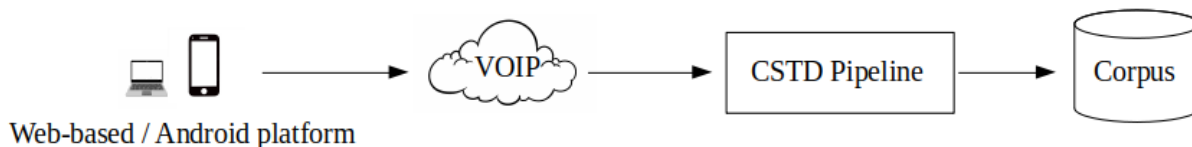


Figure 3.9: The workflow of Approach 1 (Spontaneous speech)

It is interesting to note that this approach has considered more than 1500 speakers across different age groups (20-50) because the audio recordings from the age groups 50+ (which is ≥ 50) have some type of creakiness and shivering nature in their sample so we had to discard such samples. And for the age group below 20, most of the data was non-verbal (laughter, stutter), so we were forced to discard such samples by human intervention by using the admin console. So in this way, we ensured that high-quality speech is only picked and considered for the data collection process. The low-quality data is discarded by following the guidelines discussed in Section 3.4. Later, all the accepted speech samples from both the campaign (JAM and debate) having the dialect-specific information (Telangana (TG), Rayalaseema (RY), and Coastal Andhra (CA) dialects) are stored accordingly.

3.5.2 Approach 2: Conversational mode Speech (Data pooling through freely available resources)

There is a lot of abundant audio content with transcripts available on the internet, but it is limited to English. The majority of this audio is Creative Commons (CCS) licensed by the author, who wants to give other people the right to use it. It is unclear how much non-English Creative Commons licensed content exists in total. To pool the data at a larger scale, it is always difficult to engage the crowd for a longer duration, irrespective of the language they speak. In those scenarios, these CCS audios can be used in the pooling of data as they have very high-quality audio content from YouTube and podcasts.

In order to collect larger volumes of conversational speech, high-quality speech data from freely available resources of YouTube platforms and podcasts are selected for pooling. These platforms contain a lot of high-quality audio and videos from different domains. To maintain the diversity, high-quality speech data from different domains are downloaded by using the youtube-dl library ⁷. Once the raw data is downloaded, it is passed through the CSTD pipeline (see Figure 3.3). The same is depicted in Figure 3.10.

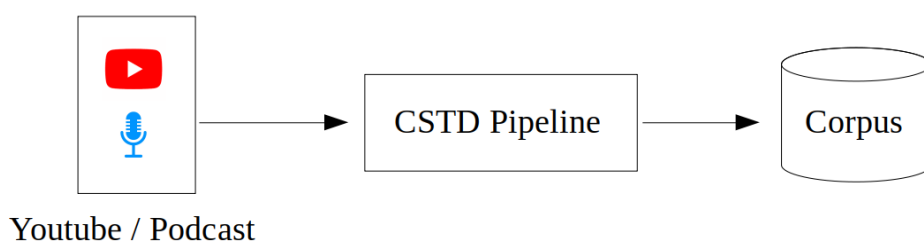


Figure 3.10: The workflow of Approach 2 (Conversational speech)

The advantage of employing this approach is that it is not as tedious as Approach 1, as there is no requirement to put in extra effort to collect speech data. High-quality speech data is pooled through the open forums as the raw audio data is readily available. This approach focuses on fragmenting and transcribing the pooled audio content. The collected speech data is filtered from the pooled audio through Approaches 1 and 2, fragmented, and finally shared with the transcribers for correcting the textual part. The transcriber’s role is to listen to and transcribe the speech fragments. Hence, we designed and built an application where the participants can transcribe the audio seamlessly. We provide both mobile and web versions for user convenience. Additionally, we utilize freely available Google cloud speech API. The rough transcript generated from the Google API is displayed on the user console, where the transcriber can edit and submit the corrected text. The fragments from both Approach 1 and 2 are loaded into this interface. The screenshot of the developed transcription interface for mobile phones is shown in Figure 3.11.

⁷<https://github.com/tpikonen/youtube-dl>

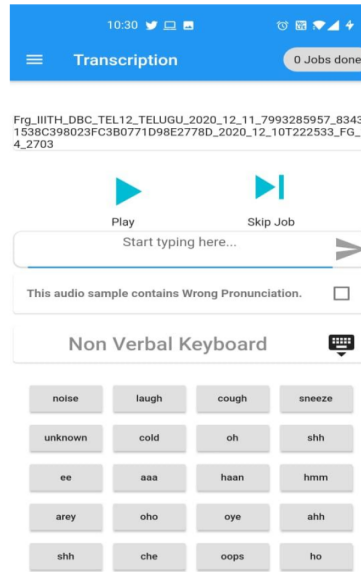


Figure 3.11: Transcription interface designed to correct the transcriptions provided to the crowd.

3.5.2.1 Mobile-based support for Transcription correction

Each fragmented audio file is displayed on the transcriber’s mobile screen. A play button is provided for the transcribers to listen to the audio files. An editable text window is provided to edit the rough transcription provided by Google Cloud API, after which the transcriber is prompted to submit the transcription. The next fragment is only displayed after the approval of the previous fragment. If there is any unwanted audio content in the fragment, then that audio fragment is discarded and not counted as a successful job. Only the approved fragments are counted toward the successful jobs. The transcribers are paid according to the count of their successful jobs. The transcribers are selected based on their expertise in the Telugu language. We have specifically assigned some sample transcription tasks to the transcribers and selected them based on the accuracy of the submitted transcripts. Most of these transcribers have a good command of the Telugu language (written and spoken). To increase the speed of transcription tasks, more transcribers are recruited to meet the monthly targets.

3.5.2.2 Web-based support for Transcription correction:

Transcribers are provided with similar web-based support, which we described in Section 3.5.2.1. Instructions were passed to the transcribers to use this web-based support to the maximum ex-

tent possible to reduce the stress on the transcribers' eyes. This web-based application was able to extract the maximum output from the transcribers. The major obstacle in the design of a

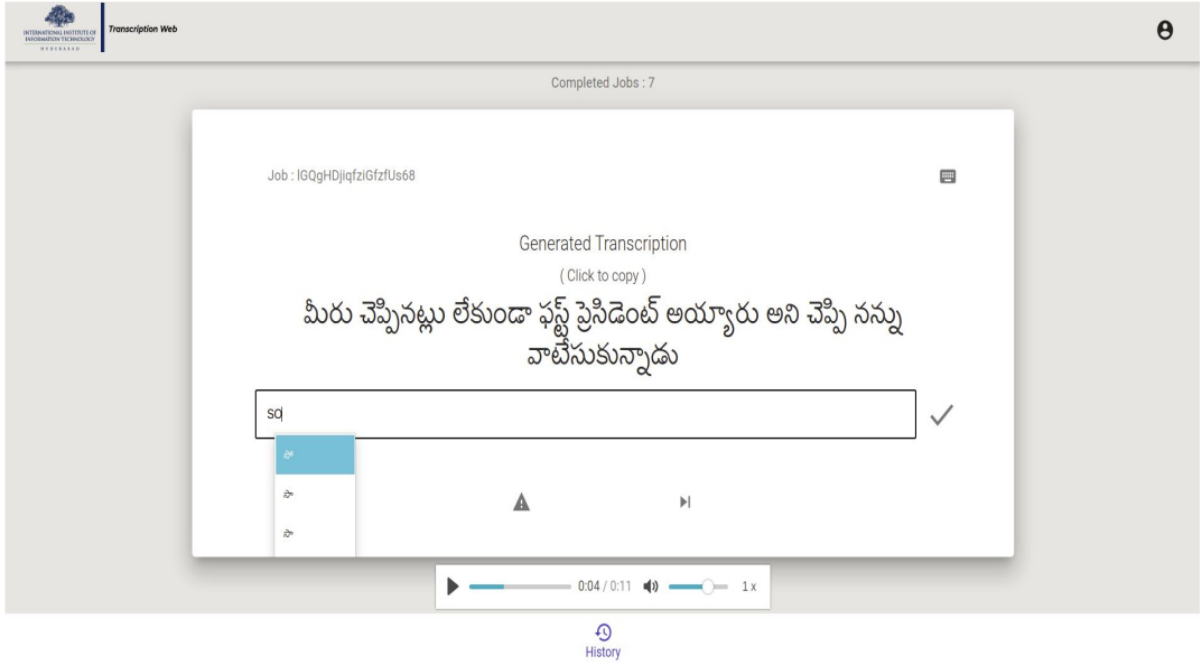


Figure 3.12: status of the completed jobs displayed on the transcriber's web-based screen.

web-based interface is the process of incorporating a Telugu keyboard. In mobiles, a Telugu keyboard can be selected easily as support for typing in multiple languages in Android versions. It is challenging to bring those options to this web-based interface. The option of equivalent Telugu prompts is displayed on the screen when the transcriber types the English keyboard option. The layout of the web-based transcription support is shown in Figure 3.12.

3.5.3 Approach 3: Data pooling through WhatsApp campaigns

In Approaches 1 and 2, the campaigns to pool the data are run so that the contributors can select their topics of interest. In all these topics, there is no control over the text to be spoken by the contributors. There are more chances for the crowd to opt for topics from a particular domain leading to un-diversified domain content. To counter this, a separate WhatsApp campaign is designed so that we have control over the domain text. This Section describes the process we followed to collect Telugu text.

3.5.3.1 Telugu text corpus

Firstly, the Telugu text has been crawled from various blogs on the internet, newspapers (like Eenadu ⁸, Andhra Jyothi ⁹ and Namasthe Telangana ¹⁰), and repository of Telugu Wikipedia ¹¹. The collected text is normalized and then converted to meaningful phrases, and the steps involved in text normalization are explained below:

1. Firstly, all the extra spaces, punctuation marks, special symbols, emoticons, etc., are removed.
2. All numbers, mathematical expressions (+,-, etc.,) and currency symbols (\$,₹, etc.) are replaced with their respective spellings in Telugu as per requirement.
3. Finally, all the meaningful phrases are structured and stored in the database.

In Figure 3.13, we show a snippet of a sample from our database for the text normalization task. To access the Telugu normalized text used in approach 3 (read mode speech) can be found here ¹².

⁸<https://www.eenadu.net/>

⁹<https://www.andhrajyothy.com/>

¹⁰<https://www.ntnews.com/>

¹¹<https://te.wikipedia.org>

¹²telugu normalized text used in read mode speech collection can be accessed here https://drive.google.com/file/d/1jqnnwjwp2EKRIusSUUIq8s3slv2jVw-S/view?usp=share_link

Original Text:

భోజనాలు వంటివి 16న, దాదాపు 1:30 గంటలలోపు ముగించుకోవాలి. * అస్వస్థులు, ఆహార సేవనాన్ని, తేలికైన విధంగా, దాదాపు 8:45 లోపుగా ముగించుకోవాలి.## గర్భిణీలు చక్కగా పూలు ధరించి, వీలుచేసుకుని గ్రహణ సమయంలో ఇంట్లో దీపార్చన చేస్తే ఇంటి యజమానికి స్త్రీయస్కరం. మేషరాశివారు గ్రహణాన్ని గమనించి, శ్రద్ధగా & పరిశీలనచేయడానికి అనువైన సమయం...వృషభరాశివారు గ్రహణాన్ని గమనించకుండా వారికి ఉపదేశానుసారంగా మంత్ర/యంత్ర/తంత్ర జపాలను సాధనలను చేసుకోవాలి. గ్రహణానంతరం తలస్నానం చేసి, సూర్యోదయం తరువాత చంద్రునికి ఓ నూలుపోగుని సమర్పించాలి. పసుపు పూసినదైతే శ్రేష్ఠం.. మిథున రాశి వారు గ్రహణాన్ని చూడరాదు. సాధనకి ఇది అనువైన సమయం? తమ ఉద్దేశం కోరుకుని, ఇష్ట దైవానుగ్రహంగా, సూర్యులకు గాని వైశ్యులకు గాని దానమిస్తే శ్రేష్ఠం..@ కర్కాటక రాశి వారికి ఈ గ్రహణం నిమిత్త మాత్రంగా %6% ఉంటుంది. కావున అశ్రద్ధ "చేయకుండా," సత్యర్థాలు చేయకపోయినా సమ్మతమే కానీ ప్రయత్న పూర్వకంగా వికల్పాలకు దూరంగా ఉంటే వారికి శ్రేయస్కరం..సింహరాశి వారికి **ఈ గ్రహణ ప్రభావం దుష్కరమల నిర్మూలనకు పనికి వస్తుంది. కావున అంతవరకూ తెలిసి చేసిన దుష్కర్మాలకు పశ్చాత్తాపం చెంది, పునరావృత్తం కాకుండా ఉండటానికి అనువైన సమయంగా పనిచేస్తుంది.%% ఈ సమయంలో వీరు క్షత్రియులకు దానం ఇస్తే శ్రేష్ఠం..కన్య రాశి వారు కూడా ఈ గ్రహణ సమయానికి పూజా మందిరం దగ్గరలో ఉంటే శ్రేష్ఠం. వీరు గ్రహణాన్ని దర్శించ రాదు. <url>మరునాడు తల స్నానం చేసి వారి ఇష్ట దైవ ప్రీత్యర్థం వైశ్యులకు దానం ఇస్తే మంచిది. <url> గ్రహణానంతరం మీన రాశి వారితో గ్రహణ విషయాలను చర్చించి అధ్యయనం గ్లి చేస్తే ఇరువురికీ మంచిది..తులారాశి కూడా ! ఈ గ్రహణాన్ని దర్శించ (రాదు. మరునాడు తల స్నానం @ చేసి రావి చెట్టునకు ప్రదక్షణలు చేస్తే మంచిది. గ్రహణానంతరం, వృశ్చిక రాశి వారితో గ్రహణ విశేషాలను చర్చించి అధ్యయనం చేస్తే ఇరువురికీ మంచిది....వృశ్చిక రాశి వారికి ^ విధ్యార్థన నిమిత్తం గ్రహణాన్ని పూర్తిగా అభ్యసించాలి. మరునాడు, వీరు వృషభరాశి వారితో చర్చించడం ఇరువురికీ శ్రేయస్కరం...ధనురాశి వారు కూడా ఈ గ్రహణాన్ని దర్శించరాదు. మరునాడు, వీరు వృషభరాశి వారితో కలిసి మేషరాశి వారితో గ్రహణ విశేషాలను "చర్చించి" అధ్యయనం చేస్తే మువ్వరికీ మంచిది....మకర రాశి వారుకూడా గ్రహణాన్ని దర్శించకపోవడమే శ్రేయస్కరం. వీరు సింహ రాశి వారికి దానం ఇస్తే లేదా పుచ్చుకుంటే ఇరువురికీ మంచిది.....కుంభరాశి వారుకి ఈ గ్రహణం అశుభాన్ని కలిగించడానికి అవకాశాలు ఎక్కువగా ఉన్నాయి. కావున ఈ రాశి వారు ఈ గ్రహణానికి అత్యంత దూరంగా ఉంటే వారికి శ్రేయస్కరం. పూర్తిగా ఉపవాశం ఉండి సాధన చేసుకోవాలి.. మీన రాశివారుకూడా %% ఈ చంద్రగ్రహణాన్ని అధ్యయన / సాధనలకు అనుకూలమైన సమయంగా తీసుకుంటే మంచిది. కావున వీరు తప్పని సరిగా దర్శించి విషయాలను యధావిధిగా ఇతరులతో పంచుకోవాలి.

Normalized Text:

<eos> భోజనాలు వంటివి వదలారన దాదాపు ఒక్కటిన్నర గంటలలోపు ముగించుకోవాలి <eos>
 <eos> అస్వస్థులు ఆహార సేవనాన్ని తేలికైన విధంగా దాదాపు ఎనిమిది గంటల నలభై ఐదు నిమిషాల లోపుగా ముగించుకోవాలి <eos>
 <eos> గర్భిణీలు చక్కగా పూలు ధరించి వీలుచేసుకుని గ్రహణ సమయంలో ఇంట్లో దీపార్చన చేస్తే ఇంటి యజమానికి స్త్రీయస్కరం <eos>
 <eos> మేషరాశివారు గ్రహణాన్ని గమనించి శ్రద్ధగా పరిశీలనచేయడానికి అనువైన సమయం <eos>
 <eos> వృషభరాశివారు గ్రహణాన్ని గమనించకుండా వారికి ఉపదేశానుసారంగా మంత్రయంత్రతంత్ర జపాలను సాధనలను చేసుకోవాలి గ్రహణానంతరం తలస్నానం చేసి సూర్యోదయం తరువాత చంద్రునికి ఓ నూలుపోగుని సమర్పించాలి పసుపు పూసినదైతే శ్రేష్ఠం <eos>
 <eos> మిథున రాశి వారు గ్రహణాన్ని చూడరాదు సాధనకి ఇది అనువైన సమయం తమ ఉద్దేశం కోరుకుని, ఇష్ట దైవానుగ్రహంగా, సూర్యులకు గాని వైశ్యులకు గాని దానమిస్తే శ్రేష్ఠం <eos>
 <eos> కర్కాటక రాశి వారికి ఈ గ్రహణం నిమిత్త మాత్రంగా ఉంటుంది కావున అశ్రద్ధ చేయకుండా సత్యర్థాలు చేయకపోయినా సమ్మతమే కానీ ప్రయత్న పూర్వకంగా వికల్పాలకు దూరంగా ఉంటే వారికి శ్రేయస్కరం <eos>
 <eos> సింహరాశి వారికి ఈ గ్రహణ ప్రభావం దుష్కరమల నిర్మూలనకు పనికి వస్తుంది కావున అంతవరకూ తెలిసి చేసిన దుష్కర్మాలకు పశ్చాత్తాపం చెంది పునరావృత్తం కాకుండా ఉండటానికి అనువైన సమయంగా పనిచేస్తుంది ఈ సమయంలో వీరు క్షత్రియులకు దానం ఇస్తే శ్రేష్ఠం <eos>
 <eos> కన్య రాశి వారు కూడా ఈ గ్రహణ సమయానికి పూజా మందిరం దగ్గరలో ఉంటే శ్రేష్ఠం వీరు గ్రహణాన్ని దర్శించ రాదు మరునాడు తల స్నానం చేసి వారి ఇష్ట దైవ ప్రీత్యర్థం వైశ్యులకు దానం ఇస్తే మంచిది గ్రహణానంతరం మీన రాశి వారితో గ్రహణ విషయాలను చర్చించి అధ్యయనం చేస్తే ఇరువురికీ మంచిది <eos>
 <eos> తులారాశి కూడా ఈ గ్రహణాన్ని దర్శించ రాదు మరునాడు తల స్నానం చేసి రావి చెట్టునకు ప్రదక్షణలు చేస్తే మంచిది గ్రహణానంతరం వృశ్చిక రాశి వారితో గ్రహణ విశేషాలను చర్చించి అధ్యయనం చేస్తే ఇరువురికీ మంచిది <eos>
 <eos> వృశ్చిక రాశి వారికి విధ్యార్థన నిమిత్తం గ్రహణాన్ని పూర్తిగా అభ్యసించాలి మరునాడు వీరు వృషభరాశి వారితో చర్చించడం ఇరువురికీ శ్రేయస్కరం <eos>
 <eos> ధనురాశి వారు కూడా ఈ గ్రహణాన్ని దర్శించరాదు మరునాడు వీరు వృషభరాశి వారితో కలిసి మేషరాశి వారితో గ్రహణ విశేషాలను చర్చించి అధ్యయనం చేస్తే మువ్వరికీ మంచిది <eos>
 <eos> మకర రాశి వారుకూడా గ్రహణాన్ని దర్శించకపోవడమే శ్రేయస్కరం వీరు సింహ రాశి వారికి దానం ఇస్తే లేదా పుచ్చుకుంటే ఇరువురికీ మంచిది <eos>
 <eos> కుంభరాశి వారుకి ఈ గ్రహణం అశుభాన్ని కలిగించడానికి అవకాశాలు ఎక్కువగా ఉన్నాయి కావున ఈ రాశి వారు ఈ గ్రహణానికి అత్యంత దూరంగా ఉంటే వారికి శ్రేయస్కరం పూర్తిగా ఉపవాశం ఉండి సాధన చేసుకోవాలి <eos>
 <eos> మీన రాశివారుకూడా ఈ చంద్రగ్రహణాన్ని అధ్యయన సాధనలకు అనుకూలమైన సమయంగా తీసుకుంటే మంచిది కావున వీరు తప్పని సరిగా దర్శించి విషయాలను యధావిధిగా ఇతరులతో పంచుకోవాలి <eos>

Figure 3.13: Illustration of text normalization

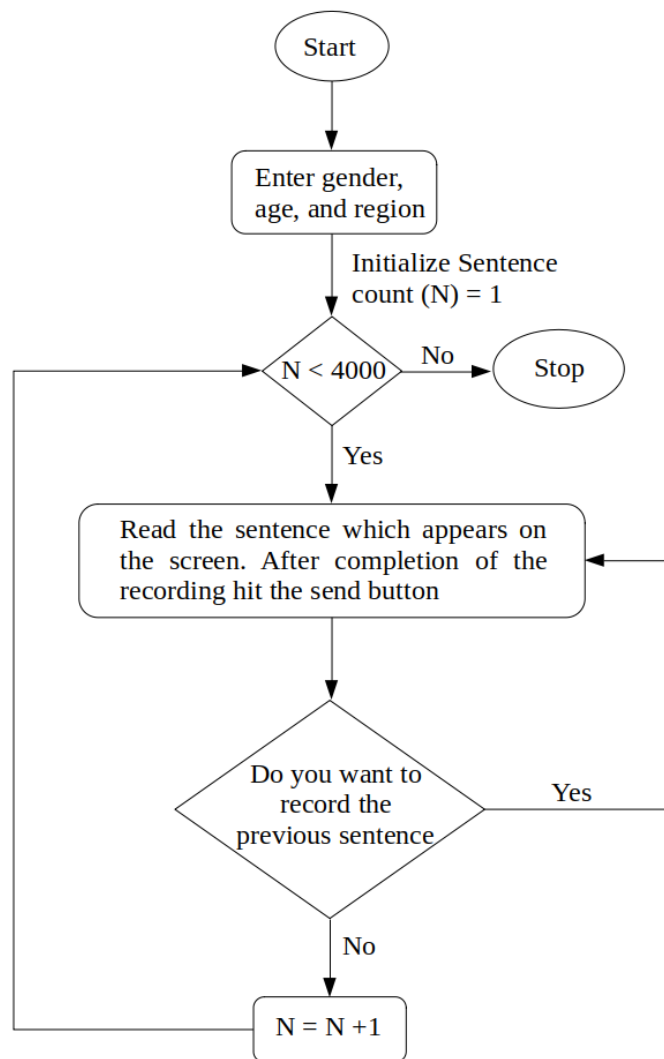


Figure 3.14: The flow chart for WhatsApp campaign (Approach3)

The students from IIIT-H, various other universities, and volunteers (of different age groups) were considered for this campaign. Each contributor is allowed to speak 4000 sentences. To contribute to this campaign, the crowd has to save the contact number (campaign contact number is +1(510)681-0104) on his/her mobile. Later he/she has to send a test message saying "hi/hello" to the campaign contact number to initiate the process. If the user is new to the campaign, the bot asks for specific basic questions like age, gender, and nativity (Telangana, Rayalaseema, and Coastal Andhra). The flow of the WhatsApp campaign is shown in Figure 3.14. The sentences stored in the database are assigned randomly to the contributor. Once it is assigned, the sentences are displayed on the contributor's WhatsApp screen, and instructions are passed to them to read out the sentences. Before giving their voice samples, all demographic

information is collected from the users. They are given clear instructions before participating in this campaign. Many speakers have volunteered for this campaign. Flexibility has been given to them to contribute during their free time. Only one sentence is displayed on their mobile screen at a time. The moment the user uploads his/her voice sample of that corresponding sentence, he/she is prompted to check his sample before confirming the submission. Once the confirmation is given from the user's end, the voice samples are stored in Google Cloud at the back end, and the following sentence is displayed on his/her screen. The campaign is designed so that the minimum number of words in each sentence displayed is 6, and the maximum is 22. The average duration of audio content collected from the sentences is 15 seconds.

3.5.3.2 Audio Verification process

A web-based interface is built to verify the audio data collected in this campaign. The liberty is given to the reviewer to accept or reject the recorded audio file. Here, the reviewer has to check (listen) the content of the recorded audio file with the corresponding transcription provided. The snippet of the audio verification web interface is shown in Figure 3.15.

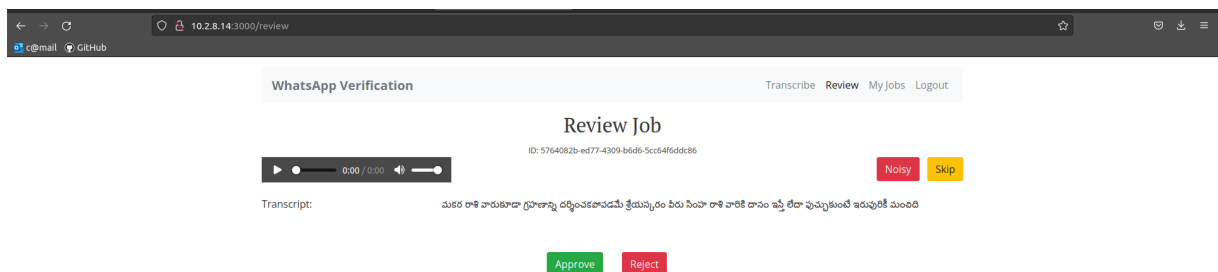


Figure 3.15: Audio verification web interface designed for WhatsApp campaigns.

3.5.3.3 Payments for Verification

As the crowd has volunteered to provide audio data, the payments have only been made for transcription verification tasks. As mentioned earlier, the crowd is from different communities (college students, homemakers, graduate students, etc.) The payment is made only for those transcribers who have contributed and submitted the task successfully. The total number of certified transcribers involved throughout this corpus verification task is 253. Once the transcribers

are onboard, they have been given training and specific golden standards for verification tasks (refer to Section 3.4). The payment is made based on the number of hours of successful transcriptions verified and corrected. Approximately 1200 to 1300 (Indian Rupees) (USD 16.17 to 17.52) for an hour of successful transcription verification for each reviewer.

3.6 Implementation Considerations and Quality Analysis

A major bottleneck to crowdsourcing the collection of annotated speech corpus is maintaining the quality of incoming data sets and manually verified transcripts. Monitoring the speech quality regularly is done to handle the speech quality collected from the crowds. There is no separate algorithm designed to sort the collected data. Once the raw data is uploaded on the user's screen feedback from the crowd is taken on a regular basis on the audio quality before approval from their side. The confidence levels of the participating crowd are also considered before the final approval. When the participant speaks on his or her relevant topics and approves them with higher confidence, only those audio files are passed and stored in the database. The other files are discarded. The first 10% of the duration of each collected audio content is sent for manual verification for the crowd. After receiving feedback from the audio annotators, the process of this data collection continued to meet the expectations for better audio quality control.

There have been several issues addressed to maintain the quality of the collected annotated speech corpus.

1. **Managing Cognitive Overload:** There has been a maximum cap put on the user contributing towards their speech samples. The duration each speaker can contribute is between 60 to 90 minutes.
2. **Recruiting Highly-Rated Crowd-Workers:** Users willing to contribute to the transcription work are made to take the transcription tests to ensure speed and quality on laptops and mobile.
3. **Reviews:** The transcripts displayed on the screens of the users are generated from the Google Cloud speech API, where the mistakes in the transcripts are corrected by the

transcribers with less effort. In this way, we could remove the duplicate transcriptions with the human-based cross-checks on the final submitted transcriptions.

4. **Reward & Recognition:** All the transcribers are paid based on the count of successfully approved transcripts.
5. The data collection platform designed by us helps in capturing the speech data through VOIP-based calling in a 16 kHz, 16-bit, wav format, and higher.

We found that we can overly depend on manual verification when collecting the data on a large scale. Hence, the support of the free version of ASR has opted to filter out noisy and unwanted content. An error message appears on the transcriber's screen whenever noisy or unwanted speech data is inputted. These audio fragments are discarded, and only the text output of high-quality audio files is considered for manual correction. This process helps in providing the gold standard transcripts by reducing the burden on the participating crowd.

3.7 CSTD corpus statistics

The table 3.1 provides database statistics for a corpus of speech collected using three different approaches: Approach 1 (spontaneous mode speech), Approach 2 (conversational mode speech), and Approach 3 (read mode speech). As mentioned earlier, approach 1 is the dialect-specific corpus (Telangana (TA), Coastal Andhra (CA), and Rayalaseema(RY)). The table provides information on the number of utterances, the number of male and female utterances, the number of utterances by age group, the minimum, maximum, and average duration of an utterance, the total number of words, and the number of unique words.

Table 3.1: Database statistics for collected CSTD corpus

Parameters		Approach 1 (Spontaneous Mode Speech)			Approach 2 (Conversational Mode Speech)	Approach 3 (Read Mode Speech)
		CA	RY	TG		
# utterances		72864	68172	89271	298100	102110
# males utterances		38617	36131	47313	157993	54118
# females utterances		34246	32040	41957	140107	47991
# Utterances age group wise	20-30	36432	34086	44636	149050	51055
	30-40	29146	27269	35708	119240	40844
	40-50	7286	6817	8927	29810	10211
minimum duration of an utterance		3 seconds	3 seconds	3 seconds	3 seconds	3 seconds
maximum duration of an utterance		15 seconds	15 seconds	15 seconds	15 seconds	15 seconds
average duration of an utterance		~12.1 seconds	~11.5 seconds	~12.1 seconds	~7 seconds	~5.5 seconds
# words		1826893	1791630	2305724	247985	107985
# Unique words		90416	88854	114958	42151	10291
# Total audio collected (hrs)		668.44	489.34	696.27	840.58	60
# Rejected audio duration (hrs)		260.32	109.28	210.15	167.88	6.2
# Accepted audio collected (hrs)		408.12	380.06	486.12	672.7	53.8

The number of male and female utterances is roughly equal for all three speakers, with slightly more male utterances overall. The number of utterances by age group shows that the corpus has a greater number of utterances from speakers aged 20-30, followed by speakers aged 30-40 and speakers aged 40-50. The minimum, maximum, and average duration of an utterance are consistent across all three approaches, with a minimum duration of 3 seconds and a maximum duration of 15 seconds. The average duration of an utterance ranges from approximately 5.5 seconds for approach 3, approximately 12.1 seconds for approach 2, and 7 seconds for approach 2. The percentage distribution of the corpora is as follows, Approach-1 (i.e., Spontaneous mode speech) (*viz.* 63.1% of data), Approach-2 (i.e., Conversational mode speech) (*viz.* 34.9% of data) and Approach-3 (i.e., Read mode speech) (*viz.* 2% of data).

3.8 Summary & Conclusion

Due to a scarcity of large, annotated speech corpora, numerous under-resourced Indian languages have been unable to leverage the recent advancements in deep neural network architectures for Automatic Speech Recognition (ASR) tasks. The creation of expansive databases often proves to be expensive and time-consuming. Traditional methods that lean heavily on extensive expert-based data acquisition guidelines are perceived as convoluted and burdensome. In response, this study introduces the International Institute of Information Technology Hyderabad-Crowd Sourced Telugu Database (IIITH-CSTD). This Telugu corpus has been collected through a crowd-sourcing methodology, primarily devised to address the resource constraints affecting the Telugu language. The study provides insights into the sources, crowd-sourcing pipeline, and protocols used to assemble the corpus. Spanning approximately 2000 hours of transcribed audio, the corpus encompasses three major regional dialects of Telugu. It presents content in three distinct speaking styles, namely read, conversational, and spontaneous, covering a wide array of topics, including politics, sports, arts, and science.

The study underscores the resource paucity commonly associated with most Indic languages, contrasting this with the resource-rich English language. Utilizing a crowd-sourcing strategy, this chapter demonstrates the collection of an extensive and annotated Telugu speech cor-

pus. Notably, the corpus was gathered virtually amidst the pandemic, capitalizing on specially crafted platforms. This research showcases the efficacy of crowd-sourcing in collating high-quality annotated datasets, offering a cost-effective alternative to traditional data procurement and curation methods.

In essence, the primary benefits of crowd-sourcing include substantial cost reduction and easier data collection compared to conventional methodologies. The subsequent chapter will delve into the experimental setups and the experiments conducted on the collected CSTD corpus.

Chapter 4

Investigation of Hybrid & End-to-End Speech Recognition systems on CSTD database

4.1 Introduction

The IIITH-CSTD corpus was meticulously crafted with a specific aim: to investigate the practicality of using a crowd-sourced data collection platform to facilitate Automatic Speech Recognition (ASR) tasks associated with Indic languages, all while maintaining cost-effectiveness. The current chapter is devoted to conducting a comprehensive evaluation of the IIITH-CSTD corpus, particularly its applicability to the Telugu Speech Recognition Task. A critical part of our evaluation revolves around analyzing the corpus's quality across various speech modes. This includes spontaneous speech, which is data collected through the first approach. Conversational speech, obtained via the second approach, and read speech, gathered through the third approach, are also considered. To achieve a detailed evaluation, we have utilized a set of well-established toolkits, such as ESPnet [100] and Kaldi [101]. These toolkits provide a robust framework for testing and analysis. The underlying objective driving this extensive evaluation process is twofold: Firstly, to establish a reliable performance benchmark for the IIITH-CSTD corpus that future efforts can compare against. Secondly, to verify its adaptability and efficiency for deployment in various Telugu speech recognition scenarios. This comprehensive analysis aims to ascertain the full extent of the corpus's utility and effectiveness.

4.2 Hybrid ASR baseline

4.2.1 Front-end features:

In this study, front-end feature parameterization was carried out using conventional Mel-frequency cepstral coefficients (MFCCs) [102, 103, 7] and high-resolution features (i-vector) [104]. The process of feature extraction is explicated in the following paragraph. The MFCC feature extraction was performed with a Hamming window of 25 milliseconds (ms), maintaining an overlap of 10 ms frame-shift. Each frame was represented as a 13-dimensional MFCC feature vector. To encapsulate the dynamic features of the vocal tract system, velocity, and acceleration components were appended to the original 13-dimensional features, thereby forming a comprehensive 39-dimensional feature vector employed for acoustic modeling. Previous studies [105, 106, 107] have considered GMM-UBM-based i-vectors for speech recognition tasks. The extraction process for i-vectors began with splicing the 13-dimensional MFCC features with respect to time, capturing dynamic variations across frames. Consequently, a 117-dimensional feature (13×9) was derived, considering four frames to the left and four to the right of the central frame, totaling nine frames. The 117-dimensional features were subsequently projected onto a 40-dimensional space using Linear Discriminant Analysis (LDA) [108]. A GMM-UBM was trained on these features to extract i-vectors. The total variability matrix was initially randomized and trained via the expectation-maximization (EM) algorithm [109], creating 150-dimensional i-vectors. These i-vectors were instrumental in building acoustic models.

4.2.2 Acoustic model training

4.2.2.1 GMM modeling

Initially, a simple, straightforward Gaussian Mixture Model (GMM) aimed at the training of context-independent monophone models. Individual phonetic units were represented here through a three-state left-to-right Hidden Markov Model (HMM). The ensuing stage of this process concentrated on developing context-dependent phonetic modeling based on speaker-adaptive training. This was accomplished by considering the alignments established during the

preceding phase. Subsequently, the forty-dimensional LDA features previously obtained were subjected to a de-correlation process facilitated by the Maximum Likelihood Linear Transform (MLLT). After this step, these features underwent normalization using the Feature-based Maximum Likelihood Linear Regression (fMLLR) technique, setting the stage for executing Speaker Adaptive Training (SAT). When estimating the GMM-HMM, the necessity arises to determine numerous unique model parameters. To streamline this complex task, we ventured to investigate the application of a Subspace Gaussian Mixture Model (SGMM)-based acoustic model, as cited in [110]. This model conveniently encapsulates a multifaceted distribution in a compressed format. The SGMM model parameters, adept at adjusting for variations in phonetics and speakers, are derived from a low-dimensional model in combination with a speaker subspace, thus constructing the model. The resultant effect of this methodology is a significant reduction in the number of model parameters, which leads to an enhancement in the model's performance. This enhancement is particularly beneficial when the availability of training data is restricted. Lastly, the unit distribution is determined using the UBM model learned during the process.

4.2.2.2 DNN modeling

In addition, a DNN-based acoustic model that has a number of hidden layers has been investigated for this work. To train the DNN, the features are spliced together with their respective times using LDA+MLLT+fMLLR. The result of the computation is the posterior probability estimated across the HMM states. In this particular investigation, the following tuning parameters were taken into consideration:

1. the number of hidden layers is 3,
2. the number of epochs is 20,
3. the dimension of the hidden layer is 1024,
4. the batch size is 64, and
5. the initial learning rate is 0.01, and the final learning rate is 0.0015.

During training, the standard DNN architecture aims to learn an affine transformation dependent on the current time environment. The need for a significant quantity of training data to

learn the proper transformation is one of the primary limitations here. This issue was addressed in this [105].

4.2.2.3 TDNN models

In the realm of speech recognition systems, Time-Delay Neural Network (TDNN) frequently finds their application in acoustic modeling. This specialized network system transforms acoustic data into a corresponding phonetic equivalent. The initial development and conceptual groundwork of the TDNN framework can be traced back to the research presented by [111], while its practical application in constructing acoustic models was pioneered by [112].

At its operational core, the TDNN processes input in the form of frame-level acoustic features. The system's output is a statistical distribution over individual phonemes, as per the defined parameters of the target language. A unique trait of this network lies in its methodical, sequential processing of the acoustic frames, which it strives to categorize into the corresponding phonemes that are most likely to match. Each input frame to the TDNN is conceptualized as a column vector, with each vector representing a unique time step within the audio signal. The individual rows of the vector denote the corresponding feature values. The TDNN employs a streamlined weight matrix, often referred to as a kernel or filter. This matrix performs the crucial task of traversing the audio signal and transfiguring the signal into output through convolution. Progressive strides in this field led to the inception of the low-rank TDNN. This advanced model features a bottleneck linear layer post each affine transformation of batch-normalized ReLU, supplemented with skip connections for enhanced performance. This design is dubbed 'low-rank TDNN' due to the application of factorization at every linear layer pair of each ReLU unit, thereby giving it a defining characteristic.

As discussed above, the following are tuning parameters for the TDNN architecture that have been considered in this work:

1. Linear bottleneck dimensions in low-rank TDNN is 256
2. The skip connection chooses further preceding layers to be added to the previous ones based on the previous layer's inputs and outputs.

3. Three non-sequential layers are taken as a single skip connection.

4.2.3 Language model training

The Language Model (LM) can make estimations and predictions about the word sequence by using a variety of statistical and probabilistic methods. One of the LM's responsibilities is analyzing the text data based on their word assumption. While performing the ASR job of decoding a series of words, the LM is beneficial so that it helps to reduce the search area. In addition, it helps determine the word sequence's joint probability, denoted by the notation $P(W)$. Both N-gram and Recurrent Neural Networks (RNN) have been investigated here. The following is a list of the parameters whose specifications were utilized in the training of RNNLM:

1. It consists of a single hidden layer composed of 150 nodes, and
2. The activation function is considered sigmoid.

4.3 End-to-End ASR baseline

In this subsection, the Conformer (Convolution-Augmented Transformer) based architecture has been utilized as the ESPnet baseline. This architecture combines the long-range interactions of transformers with the local sensitivities of convolutional neural networks to create a conformer architecture (i.e., convolution + transformer) (The overview of conformer can be found in Chapter 2). Specific parameters have been selected during the training process to optimize the model's performance. Specifically, the encoder block count has been set to 12, the output dimension to 512, and the kernel size to 31. Moreover, the decoder block count has been set to 6, and each encoder and decoder has eight attention heads. The feed-forward dimension has been set to 2048, and the model has been trained for 20 epochs with a learning rate of 0.0015. In addition, a time mask of 5 and a frequency mask of 2 have been considered. The last checkpoint has been replaced with an older one, and the best checkpoint has been stored.

4.3.1 Evaluation Results & Discussion

This subsection presents the evaluation results of hybrid Automatic Speech Recognition (ASR) models, including the Gaussian Mixture Model-Hidden Markov Model (GMM-HMM), Sub-space Gaussian Mixture Model (SGMM), Deep Neural Network-Hidden Markov Model (DNN-HMM), and Time-Delay Neural Network (TDNN), as well as End-to-End models, such as Transformer and Conformer, trained on the IITH-CSTD corpus.

The Hybrid models, including GMM-HMM, DNN, and TDNN, are reported in Table 4.1. In parallel, the outcomes from our evaluation of the End-to-End baseline models, which include the advanced Transformer and Conformer architectures, are detailed with precision in Table 4.2. The tables include three approaches, A1, A2, and A3, for spontaneous, conversational, and read speech. The evaluation metrics are word error rate (WER) in percentages.

In Table 4.1 and Table 4.2, the first column represents the toolkits used for the evaluation, which includes KALDI and ESPnet. The second column shows the model used for the evaluation, which includes GMM (tri), SGMM, DNN, TDNN, and TDNN-RNN for KALDI and Transformer and Conformer for ESPnet. The third column represents the front-end features: MFCC + LDA + SAT and MFCC + i-vector for KALDI, and Mel-filter bank and Raw for ESPnet. The fourth column represents the language model, which is 5-gram for KALDI and no language model for ESPnet. The remaining columns show the WER results for each approach, i.e., A1, A2, A3, and A1+A2+A3, respectively.

Table 4.1: Evaluation of IIITH-CSTD Corpus on hybrid ASR systems

Toolkit	Model	Front-End Features	LM	Approach 1 (A1) (Spontaneous)				Approach 2 (A2) (Conversational)	Approach 3 (A3) (Read)	A1+A2+A3
				TG	CA	RY	TG+CA+RY			
Kaldi	GMM	MFCC + LDA + SAT	5-gram	23.05	27.16	30.89	26.93	45.98	44.63	33.12
	SGMM	MFCC + LDA + SAT	5-gram	17.75	20.07	21.39	20.03	37.68	36.59	28.89
	DNN	MFCC + LDA + SAT	5-gram	14.24	15.98	19.21	17.47	34.25	33.64	24.04
	TDNN	MFCC + i-vector	5-gram	13.04	14.64	17.56	15.48	31.98	31.14	22.69
	TDNN	MFCC + i-vector	RNN	12.41	13.58	15.85	14.94	30.12	30.71	21.98

Table 4.2: Evaluation of IIITH-CSTD Corpus on End-to-End ASR systems

Toolkit	Model	Front-End Features	LM	Approach 1 (A1) (Spontaneous)				Approach 2 (A2) (Conversational)	Approach 3 (A3) (Read)	A1+A2+A3
				TG	CA	RY	TG+CA+RY			
ESPnet	Transformer	Mel-filter bank	-	11.94	13.08	14.24	13.89	26.70	25.89	18.09
	Conformer	MFCC + LDA + SAT	-	11.68	12.79	13.97	13.11	23.45	22.27	15.72

4.3.1.1 Hybrid ASR baseline results

The ASR system has been developed using the collected database, individually and in combination. Preliminary experiments have been conducted on the Approach 1 database, which is diverse and consists of different dialects. A GMM-HMM model has been built using the Approach 1 data, and its alignments are considered for building SGMM, DNN, and TDNN models. Among all the hybrid models, TDNN with RNNLM has performed better for all the cases, i.e., for each dialect (TG, CA, RY) and combining all dialects (TG+CA+RY) in a single model. The statistics of Approach 1 are reported in Table 3.1. The WERs for the best model (TDNN with RNNLM) on TG, CA, RY, and TG+CA+RY are 12.41%, 13.58%, 15.85%, and 14.94%, respectively. The same experiments have been carried out on corpus collected using other approaches. It is observed that TDNN with RNNLM outperforms all the hybrid systems on all the databases (i.e., Approach 1, Approach 2, Approach 3, and A1+A2+A3).

4.3.1.2 End-to-End models baseline results

This subsection presents an End-to-End model baseline trained on the database collected using the ESPnet toolkit. Table 4.2 tabulates the Word Error Rates (WERs) (%) of the models trained on the database collected using three approaches. The study observes that among Transformers and Conformers, the Conformers outperformed the Transformers for all the approaches in the End-to-End baselines. Specifically, for Approach 1, the overall WERs (%) on Transformer and Conformer are 13.89 and 13.11, respectively, for the TG+CA+RY task. The Transformer-based and Conformer-based ASRs trained on data collected using Approach 2 achieved a WER of 26.70 and 23.45 (WER), respectively. For the read mode speech (i.e., A3), the WERs of Transformer-based and Conformer-based models trained on read mode speech are 25.89 and 22.27, respectively. Finally, the WERs of Transformer architecture and Conformer trained on pooled data (i.e., A1 + A2 + A3) are 18.09% and 15.72%, respectively.

4.4 Ablation studies on IITH-CSTD corpus

4.4.1 Different Speaking Styles

In this subsection, we provide a detailed overview of the ablation studies performed on an assortment of architectures, taking into account the distinct speaking styles present in the training data. These comprehensive evaluations aid in discerning the individual contributions of each component within these architectures.

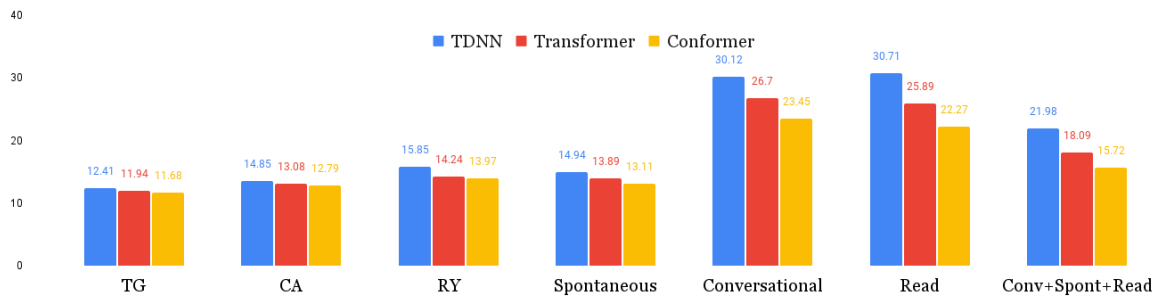


Figure 4.1: WER evaluation on different architectures (like TDNN, Transformer, & Conformer) of different speaking styles sizes (Spontaneous, Conversational, and Read) on IITH-CSTD corpus

The Figure 4.1, compares the performance of three different models, namely Time Delay Neural Network (TDNN), Transformer, and Conformer, across various dialects (Telangana - TG, Coastal Andhra - CA, Rayalaseema - RY) and different speaking styles (Conversational, Read, Spontaneous, and a combined set of all three). The metric used is Word Error Rates (WER), which are commonly used in the evaluation of ASR models, with lower values indicating better performance.

Starting with the dialect-based comparison, across all dialects, the Conformer model appears to perform the best, with the lowest error rates ranging from 11.68% for TG to 13.97% for RY. The Transformer model follows closely, and the TDNN model has the highest error rates among the three models for each dialect. Interestingly, all models seem to struggle more with the RY dialect, suggesting it might have unique characteristics making it more challenging to model.

As for the speaking styles, again, the Conformer model consistently outperforms the other two models. However, it's worth noting that the error rates significantly increase when the models are tested with Conversational and Read speech styles compared to the Spontaneous style. This could be due to factors like increased speed, varied tone, or less clear articulation often found in read and conversational speech compared to spontaneous speech.

In the combined set of all speaking styles (Conv+Spont+Read), the Conformer model again leads with a WER of 15.72%, followed by the Transformer model at 18.09%, and TDNN at 21.98%. This suggests that the Conformer model best handles the diversity of speaking styles among the three.

Overall, this analysis suggests that the Conformer model tends to have superior performance across different dialects and speaking styles compared to the TDNN and Transformer models. However, it also underscores the challenges ASR models face when handling various dialects and speaking styles, with clear variations in performance across these factors. Further research might be needed to improve the models' ability to deal with dialectical and speaking style variations effectively.

4.4.2 Different dataset sizes

In this subsection, we present the results of ablation studies conducted on various architectures with different dataset sizes ranging from 10 to 2000 hours of training data.

Figure 4.2 provides a comparative analysis of the Word Error Rate (WER) for three different speech recognition models (TDNN, Transformer, and Conformer) trained on varying amounts of data. The results show a consistent trend of decreasing WER as the number of training hours increases for all models, which is a well-understood pattern in machine learning. More data tends to provide the model with more examples to learn from, leading to a better generalization of unseen data and, thus, lower error rates.

Looking at the 10-hour training data mark, TDNN has the lowest WER at 60.69%, while the Conformer model has the highest error rate at 68.1%. However, as the amount of training data

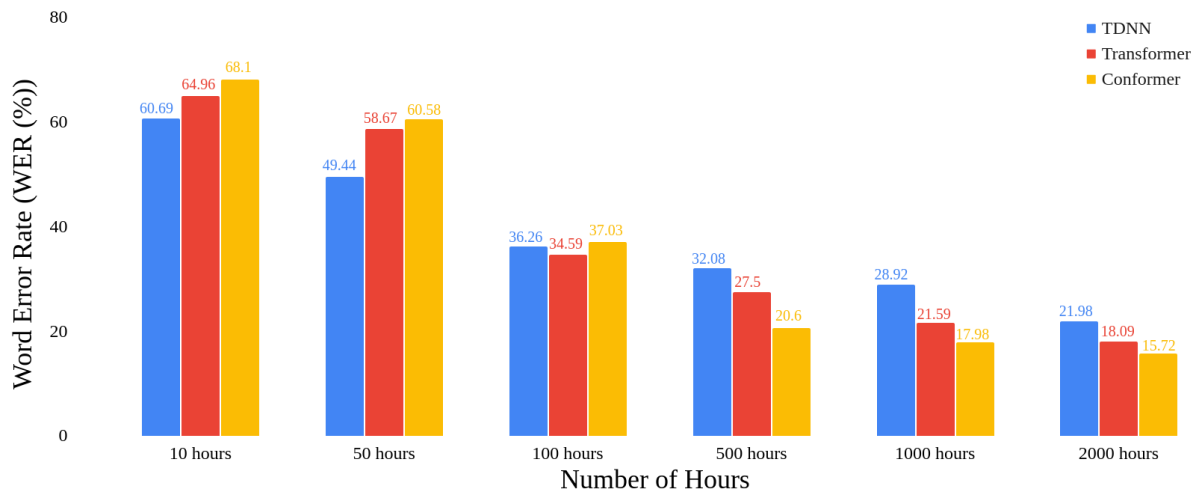


Figure 4.2: WER evaluation on different architectures (like TDNN, Transformer, & Conformer) of different dataset sizes (10 hrs, 50 hrs, 100 hrs, 1000 hrs, & 2000 hrs) on IITH-CSTD corpus

increases, the trend changes. The Conformer model steadily outperforms the other two models, starting from the 500-hour mark and maintaining this lead through the 2000-hour mark. At 2000 hours, the Conformer model achieves the lowest WER at 15.72%. The Transformer model also demonstrates a similar but more pronounced pattern. It starts with the highest WER at the 10-hour and 50-hour marks. Still, it significantly improves as training data increases, surpassing the TDNN model at the 100-hour mark and maintaining its lead for the remaining data points. At the 2000-hour mark, the Transformer model achieves a WER of 18.09%. On the other hand, the TDNN model, while starting with the best performance at lower data volumes, seems to benefit less from additional data compared to the other two models. Its performance improvement rate slows as the training data increases, and it consistently ranks third in performance from the 100-hour mark onwards.

Our study reveals that both the Conformer and Transformer models perform admirably, particularly when presented with an increasing volume of training data. Among them, the Conformer model outshines, skillfully using additional data to fine-tune its accuracy. The Conformers have a winning edge over TDNNs, primarily due to their ability to model long-term dependencies and unravel complex patterns hidden in the data.

The TDNN model, though showing an initial upper hand with smaller datasets, falls short as the amount of training data multiplies. This indicates that the TDNN model may not be as adept as its Conformer and Transformer peers in learning from extensive datasets. Both Transformers and Conformers harness the power of self-attention mechanisms, which aids them in tracking long-term dependencies in the input sequence. Yet, the Conformers set themselves apart by weaving in convolutional layers that grasp local sequence patterns, making them better for sequential modeling tasks like speech recognition. Users can evaluate the ASR system based on the Conformer architecture through the following link: <https://asr.iiit.ac.in/ganesh>. This study underlines the crucial role of model selection, taking into account the available training data. After all, different models possess different learning abilities depending on the volume and characteristics of the data.

4.4.3 Cross-dataset benchmarking of speech recognition task

In this subsection, we present a benchmarked comparison of the IIITH-CSTD corpus with the MSR Telugu dataset. Table 3.1 shows that the MSR telugu dataset is only 50 hours (45 hours of training + 5 hours of testing). But in this case, the IIITH-CSTD corpus is of 2000.8 hours, so to have a fair comparison across the datasets, a subset of the IIITH-CSTD corpus with a duration of 50 hours was selected with each speech mode being 15 hours (spontaneous speech, conversational speech, and read mode speech) which in turn, constituted 45 hours. Five hours were randomly selected for testing. Further, a 5-hour test set from YouTube was included to ensure uniformity in the testing data. The YouTube data were randomly selected and not biased towards any particular domain or speaker. Including this real-world data allowed the evaluation of how well the models trained on IIITH-CSTD and MSR Telugu performed on a more diverse and challenging test set. The table 4.3 reports the word error rate (WER) results obtained for two training (MSR Telugu, IIITH-CSTD (subset)) and three testing scenarios (MSR Telugu, IIITH-CSTD, and Youtube data). The ASR systems are trained on TDNN-based acoustic models. Table 4.3 infer that the IIITH-CSTD corpus outperforms when compared to MSR Telugu in all three testing scenarios (IIITH-CSTD, MSR Telugu, and Youtube). Additionally, the WER for the YouTube dataset on MSR Telugu is higher than the IIITH-CSTD corpus, indicating that the IIITH-CSTD corpus contains a diverse range of speech representative of real-world scenarios.

Table 4.3: Benchmarked comparison of IITH-CSTD corpus with MSR Telugu, Youtube

		Test		
		IITH-CSTD (5 hours)	MSR Telugu (5 hours)	Youtube (5 hours)
Train	IITH-CSTD	17.98	11.81	21.03
	MSR Telugu	25.68	19.78	43.98

The findings of this study suggest that even the subset of the IITH-CSTD corpus presents a more challenging task for speech recognition due to its greater diversity of speech styles and vocabulary. This highlights the importance of evaluating speech recognition models across various datasets to assess their generalization ability. Additionally, the study underscores the potential benefits of training speech recognition models on different datasets that capture various recording conditions, speaker characteristics, and linguistic content. Training on diverse datasets can enhance the robustness and performance of models when faced with new datasets. This study’s results also emphasize the critical role of training on diverse datasets in improving the generalization performance of speech recognition models. Such training enables the model to adapt to variations in recording conditions, speaker characteristics, and other sources of variability commonly encountered in real-world scenarios. Furthermore, it can enhance the model’s ability to handle various vocabularies and linguistic structures that may significantly differ across different regions and dialects.

4.5 Summary & Conclusion

This study evaluates an Automatic Speech Recognition (ASR) system using various models, including GMM-HMM, SGMM, DNN, TDNN, Transformers, and Conformers. Among them, the TDNN model coupled with RNNLM performs the best, while the Conformer model consistently outperforms the others in all test scenarios. The Conformer model exhibits remarkable improvement as the training data increases, highlighting the advantages of utilizing diverse datasets. To further emphasize the importance of dataset diversity, we compare the performance of the IITH-CSTD corpus and the MSR Telugu dataset. Our findings reveal that the

TDNN model excels when working with smaller datasets but reaches a plateau as the training data grows. On the contrary, the Transformer and Conformer models demonstrate significant performance enhancements with larger datasets. Notably, the Conformer model stands out for its superior performance across different dialects and speaking styles, benefiting greatly from larger and more diverse training datasets. This model effectively handles complex data patterns and long-term dependencies.

Furthermore, this study underscores the essential role of diverse datasets in training ASR models. The superior performance of the IITH-CSTD corpus compared to the MSR Telugu dataset proves the advantages of training ASR models on diverse data. Such diverse training data enhances the robustness and generalization capabilities of ASR models, enabling them to adapt to a wide range of vocabularies, dialects, and linguistic structures. The next chapter will delve into a self-supervised approach for speech recognition tasks in the Indian context, building upon the insights gained from this research.

Chapter 5

Exploration of self-supervised based approach for improving the performance of speech recognition system on CSTD corpus

5.1 Introduction

In the realm of speech recognition research, the procurement of extensive and diverse datasets plays a crucial role in model development and performance. Chapter 3 elucidates the strategies and methodologies employed to collect a sizable speech corpus for the Automatic Speech Recognition (ASR) task using a crowd-sourced approach. Through this method, a comprehensive platform was developed and consequently collected an expansive corpus of Telugu speech data. Following this, Chapter 4 presents a series of experiments conducted to establish baseline results for the collected corpus. Overall, while there are still challenges to be addressed, the recent advancements in deep learning have shown promising results in improving ASR for low-resource Indian languages. A potential solution to mitigate the gap between high and low-resource languages is the acquisition of labeled datasets for the latter. However, creating such datasets can be challenging due to the logistical issues in collecting data across various languages and dialects, rendering this approach expensive. An alternative approach to narrow this gap involves self-supervised learning. Models such as Wav2Vec can be trained on larger, unlabelled datasets that are relatively easier to obtain and fine-tuned with smaller labeled datasets. Another approach is a cross-lingual transfer of knowledge from high to low-resource languages. Moreover, transfer learning relies on non-native speakers and content, which may lead to issues

with robustness when deployed on various benchmarks. Thus, although self-supervised and transfer learning can serve as practical techniques to narrow the WER gap, a more extensive dataset remains crucial to reduce the considerable disparity between low and high-resource languages.

This chapter explores the possibility of building effective ASR systems for low-resource languages of the Indian subcontinent using wav2vec2.0. Numerous existing wav2vec models owe their performance to substantial training efforts conducted on considerable volumes of meticulously gathered raw audio data, predominantly in languages such as English. Past research efforts have led to the online availability of speech corpora for various Indian languages, including English, Hindi, and Tamil. These speech corpora, each encompassing roughly 200 hours of language data, serve as an invaluable repository for researchers and developers vested in the progression of Automatic Speech Recognition (ASR) systems [79, 94, 95, 96, 97]. Despite these resources, the Telugu language still witnesses a significant deficit in data representation. The most extensive publicly available Telugu speech corpus, currently standing at 50 hours of data collected by [79], is insufficient to meet the needs of modern ASR architectures. These contemporary ASR systems are known for their necessity for significantly larger volumes of training data.

5.1.1 Data pre-processing

Initially, to build a comprehensive speech corpus for Telugu, we employed a data collection methodology that involved various keywords across different domains. We utilized these keywords to search for suitable channels, playlists, and individual videos on YouTube. Our primary consideration for selection was the quality of the audio, which had to be free of any background music and predominantly in Telugu. Nonetheless, we acknowledged that some degree of code-mixing with English is unavoidable in Telugu speech content. Once we had identified relevant videos, we conducted a manual inspection to verify that the audio was predominantly in Telugu. We also ensured that the videos were available under a Creative Commons License. It is worth noting that our approach, while ensuring high-quality data, reduced the amount of available data, particularly for the Telugu language. Nonetheless, we believe that the data we collected

provides a solid foundation for building a comprehensive Telugu speech corpus that can aid in developing ASR systems. The filtered URLs are sent to the CSTD pipeline for obtaining the raw speech data (unlabeled)(The functionality of the CSTD pipeline is discussed in Chapter 3). The total amount of raw audio data utilized for pretraining is 5000 hours. To ensure the model’s reliability, 4% of the data was assigned as validation data to monitor the validation loss during the pretraining phase. The remaining 96% of the data was employed to pre-train the model to learn robust and powerful representations from the speech audio.

The raw audio data utilized in this study encompassed various speech domains such as news, debates, movies, interviews, speeches, etc., so it covers all three modes: read, conversational, and spontaneous. This approach was adopted to incorporate diverse linguistic contexts, styles, and accents to the pretraining data, which helps in enhancing the model’s ability to generalize across multiple domains and improve recognition accuracy.

5.2 Telugu Wav2Vec pretrained model

Initially, raw speech audio is fed into a multi-layered convolutional network called the encoder. This sophisticated network is designed to extract meaningful, high-level features from the speech audio, converting these complex acoustic signals into a series of latent speech representations. Each latent representation encapsulates the essential acoustic-phonetic properties of the corresponding speech fragment. Following the extraction of latent representations, a masking strategy is employed. Herein, we draw inspiration from the “masked language modeling” concept in Natural Language Processing (NLP). Certain portions of the latent representations are concealed in this step, thus setting up a prediction task for the model to restore these masked portions. The masked latent representations then serve as the input to a Transformer network. The Transformer, renowned for its capacity to generate contextually rich representations, processes these inputs. It uses its self-attention mechanism to weigh the influence of different contextual elements within the sequence, thereby generating context-aware representations that reflect the content of each speech fragment and its relation to the surrounding context. The

training regimen of our model utilizes a contrastive task. The model is tasked to distinguish the actual (masked) latent representation from a set of distractor representations. This task encourages the model to produce accurate and distinctive representations, which is essential for its performance on downstream tasks. The latent representations are processed through a Gumbel softmax function during the contrastive task. This allows us to model the latent representations as discrete speech units, thereby simulating speech categorization into phonemes, words, and phrases humans naturally perform.

5.2.1 Model Architecture for Pre-training

The present study investigates an architecture that closely mirrors the principles of the wav2vec 2.0 framework [113], incorporating three essential components that collaboratively contribute to the system’s overall performance. The same is depicted in Figure 5.1 These components are detailed as follows:

1. **Feature Encoder:** Serving as the initial stage of the process, the feature encoder is a sophisticated multi-layered convolutional neural network (CNN) designed to process raw audio signals. This CNN ingests the input audio and generates a sequence of frames, denoted as $Z = z_1, z_2, \dots, z_T$, where T represents the total number of timesteps. Each z_i corresponds to a vector in the d -dimensional space. The feature encoder effectively translates the raw audio data into structured, latent representations that can be more readily analyzed and manipulated.
2. **Context Network:** Building upon the latent representations produced by the feature encoder, the context network, based on a Transformer architecture, plays a pivotal role in learning context-aware representations for each T unit. This is achieved by implementing a self-attention mechanism, which captures the inherent dependencies and relationships between various elements of the sequence. Consequently, the context network enriches the initial latent representations with contextual information, yielding a more expressive and comprehensive representation of the input audio. The output of the context network

is a sequence of contextualized representations $C = c_1, c_2, \dots, c_T$, where c_i denotes the contextual representation for the i^{th} input (i.e., Z_i).

3. **Quantizer:** The final component in the architecture, the quantizer, is responsible for discretizing the continuous representations generated by the feature encoder. This is accomplished by employing product quantization, which maps each continuous representation to the nearest element in a predefined codebook, resulting in a discrete set of indices. The quantizer effectively compresses the data while retaining crucial information, thus providing a suitable set of targets for self-supervised learning. The output of the quantizer is a sequence of representations $Q = q_1, q_2, \dots, q_T$, where each q_i corresponds to the quantized representation for the i^{th} input (i.e., Z_i).

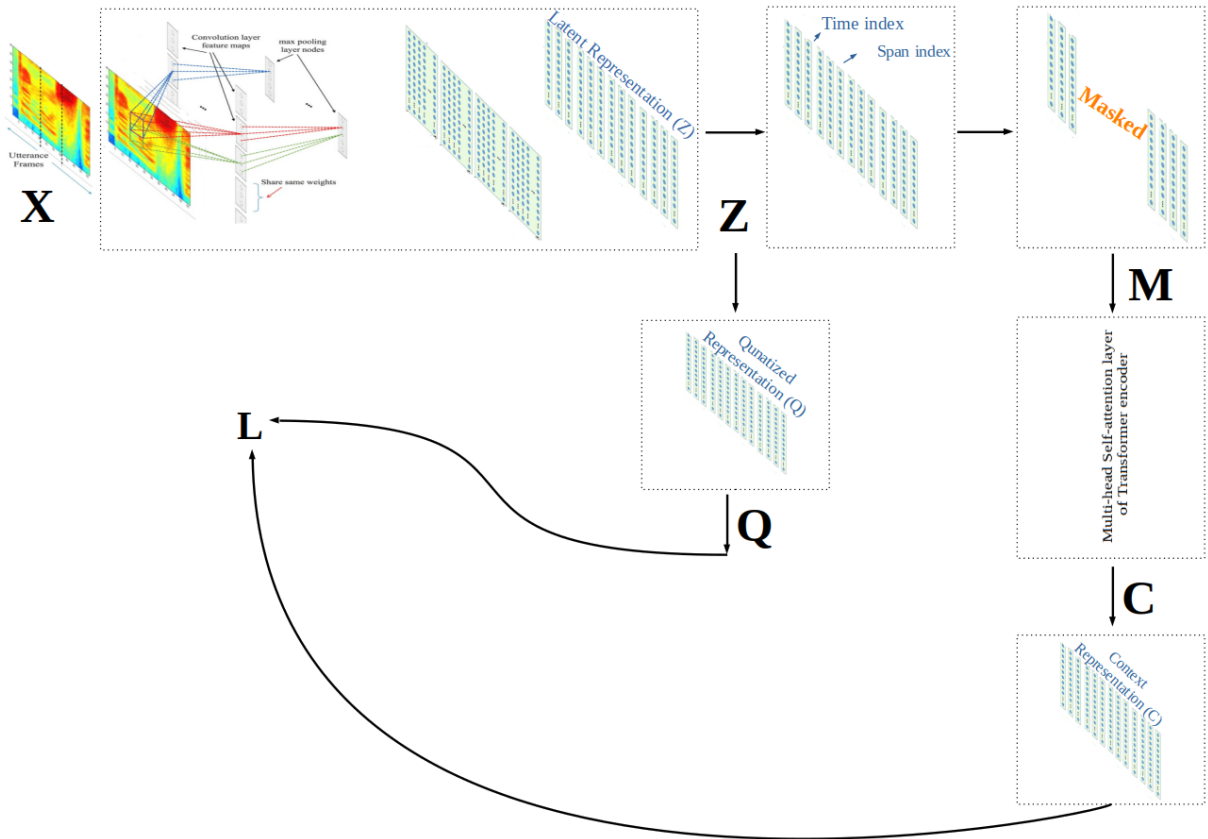


Figure 5.1: Illustration of Wav2Vec 2.0 architecture.

The steps followed in pretraining are listed in Algorithm 1. Masking a certain proportion of time steps in latent space is much similar to the BERT-based Language model.

Algorithm 1 Steps followed to have a learned representation

Input: Input Speech Waveform (X)

[1] speech signal (X) is passed through a multi-layer convolutional neural network (CNN) to encode into a latent representation which is denoted as “ Z ”. It is mathematically represented as,

[2]

$$Z = CNN(X) \quad (5.1)$$

[3] The latent space representations are then quantized and denoted as “ Q ”.

[4]

$$Q = \text{quantization}(Z) \quad (5.2)$$

[5] Later the output of quantization is randomly masked (M)

[6]

$$M = \text{masking}(Z) \quad (5.3)$$

[7] Lastly, the masked output is passed onto the multi-head self-attention layers of the Transformer encoder backbone.

[8]

$$H = \text{Transformer}(M) \quad (5.4)$$

Output: Learned representations

5.2.2 Contrastive Objective and Codebook Diversity During Model Pre-training

During the pre-training phase of our model, we aim to construct robust representations of speech audio by addressing a contrastive objective, denoted as \mathcal{L}_m . This objective compels the model to correctly discern the genuine quantized latent speech representation for a concealed time step amidst various distractors. The model navigates this challenge by drawing upon its cultivated contextual understanding. The representations of speech audio that the model learns through this process are remarkably potent, providing a foundation upon which numerous downstream tasks can effectively build. Parallel to this contrastive objective, we also incorporate a codebook diversity loss, represented as \mathcal{L}_d , into our loss function. This auxiliary loss is instrumental in consistently utilizing all entries within the codebook. One of the prevalent challenges when learning discrete representations is ensuring fair usage of all the learned symbols. Absent this auxiliary loss, the model risks a disproportionate usage of certain symbols, favoring them excessively while neglecting others. The introduction of the \mathcal{L}_d loss effectively counteracts this tendency, prompting a more balanced utilization of the codebook entries. This, in turn, allows the model to canvass the representational space more effectively, facilitating the learning of a

diverse and rich array of speech representations. The ultimate objective is to bolster the model’s robustness and adaptability, enabling it to manage a broad spectrum of acoustic environments and speaker characteristics. The amalgamation of these two losses, the contrastive objective \mathcal{L}_m and the diversity loss \mathcal{L}_d , ensures that the model not only learns high-quality speech representations but also maximizes the full potential of the learned codebook. This comprehensive approach leads to a model capable of performing sophisticated recognition tasks across various conditions and contexts. Mathematically loss is expressed as follows:

$$\mathcal{L} = \mathcal{L}_m + \alpha \mathcal{L}_d \quad (5.5)$$

Here α is a tunable hyperparameter. A significant dimension of our model’s training architecture revolves around contrastive loss. Conceptually, this is rooted in identifying the actual quantized latent speech representation, referred to as q_t , from many alternatives. Within this multitude, denoted by $\tilde{q} \in Q_t$ are $K + 1$ quantized candidate representations. This set encompasses q_t and K distractors. We uniformly sample these distractors from other masked time steps within the same utterance. The mathematical formulation of the contrastive loss, denoted as \mathcal{L}_m , is as follows:

$$\mathcal{L}_m = -\frac{\log \exp(\text{sim}(c_t, q_t)/\kappa)}{\sum_{\tilde{q} \in Q_t} \exp(\text{sim}(c_t, \tilde{q})/\kappa)} \quad (5.6)$$

The variable c_t in this equation represents the output from the context network centered over the masked timestep, t . The sim function computes the cosine similarity between the context representations and quantized latent speech representations. This operation takes the form $\text{sim}(a, b) = a^T b / \|a\| \|b\|$, as detailed in prior studies. In this context, a and b denote distinct feature vectors, and their similarity measure aids in differentiating between the vectors, contributing significantly to the model’s performance. In tandem with contrastive loss, the model also incorporates a codebook diversity loss \mathcal{L}_d to promote uniform usage of the codebook entries. This ensures that each entry contributes equitably to the speech data’s overall representation, enhancing the robustness and comprehensiveness of the trained model. Balancing contrastive loss and codebook diversity loss during training is critical to the model’s successful operation and ability to deliver accurate speech recognition performance.

By employing the diversity loss function, we encourage the model to generate a broad range of quantized speech representations that maximally exploit the available codebook entries. We argue that the richness and diversity of these representations are directly correlated with the model’s overall performance and capacity to generalize across varied speech inputs. The diversity loss function focuses on ensuring the balanced use of each codebook’s entries, which is integral to a well-performing model. In essence, the diversity loss function operates as a regulatory mechanism that promotes uniformity across the codebook usage. This function integrates an entropy maximization strategy applied to the averaged softmax distribution across each codebook’s entries, calculated over a batch of utterances. The softmax distribution, devoid of Gumbel noise or a temperature component, is averaged over the entries of each codebook. Expressed mathematically, the diversity loss function can be written as:

$$\mathcal{L}_d = \frac{1}{GV} \sum_{g=1}^G -H(\tilde{\rho}_g) = \frac{1}{GV} \sum_{g=1}^G \sum_v^V \tilde{\rho}_{g,v} \quad (5.7)$$

Where:

- G represents the total number of codebooks,
- V symbolizes the total number of entries in each codebook,
- $\tilde{\rho}_g$ denotes the averaged softmax distribution over the codebook entries for each codebook,
- H represents the entropy function.

The entropy $H(\tilde{\rho}_g)$ measures the uncertainty or randomness in the usage of the codebook entries. Maximizing entropy in the context of our model implies that we aim to enhance the diversity or richness of the learned speech representations. In effect, it encourages the model to utilize all parts of the learned representation space. Implementing this diversity loss function, ensures an efficient spread across the feature space, thus encouraging the model to learn a comprehensive, diversified range of speech characteristics. As a result, we effectively optimize the model’s performance, contributing to its improved efficiency in handling automatic speech recognition tasks. This innovative approach to the loss function sets a promising avenue for future research and applications in the realm of speech recognition technology.

5.3 Pretraining Experimental Setup

The architecture for the pretraining phase of our experiments, specifically a *BASE* variant model, draws inspiration from the model described in the paper by [113]. The structure of our *BASE* model encompasses:

- Seven convolutional layers: Each layer has 512 channels and employs strides of (5, 2, 2, 2, 2, 2, 2) and kernel widths of (10, 3, 3, 3, 3, 2, 2) respectively. This arrangement ensures that the network can effectively extract high-level features from the input audio.
- Twelve transformer blocks: Each block consists of a model dimension of 768, a feed-forward network (FFN) dimension of 3072, and 8 attention heads. This configuration allows the model to capture the contextual information within the data with high accuracy.

In the quantization module, we use $G = 2$ codebooks, each comprising $V = 320$ entries. This setup aids in discretizing the continuous latent representations, enabling our model to learn discrete speech units. The curated Telugu language dataset serves as the pretraining material for the model. We employ temperature-based sampling to account for data skew, a strategy that helps us create balanced training batches. Based on experiments involving five distinct values of $\alpha \in 0.6, 0.7, 0.8, 0.9, 1$, we discerned that $\alpha = 0.7$ yields the optimal model accuracy for identifying the correct masked unit amid distractor units. For the *BASE* model, we limit the audio segments to 250k samples, corresponding to approximately 15 seconds of audio. To manage memory effectively, we cap the maximum tokens per GPU at 3M and train the model using four A100 GPUs, with a gradient accumulation of 2 steps. In the optimization phase, we utilize the Adam optimizer with a learning rate 0.0005, which undergoes polynomial decay after a 32k step warm-up period. We maintain a consistent random seed setting of 1 across all our experiments and adhere to the default hyperparameter values from the original wav2vec 2.0 codebase. Our *BASE* model consists of 95 million parameters. For implementation, we rely on [114], which enables us to use mixed-precision training with fp16 operations, thereby speeding up the training process and minimizing memory usage. Additional insights regarding the training process of the ASR system, such as learning rate, batch size, and the number of training epochs, are available in our training logs¹. These logs also provide crucial information

¹https://wandb.ai/mirishkarganesh/5000h_Telugu_pretrained_model

on the validation loss and accuracy, integral for tracking the performance of the ASR system during the training phase.

5.3.1 Layer-Wise Analysis of CSTD-5k Pretrained Model

In this subsection, we delve into the intricate task of analyzing the layers of the pre-trained model (CSTD-5k) which was built. Our approach relies on canonical correlation analysis (CCA), a robust methodology that enables us to extract profound insights from the CSTD-5k model. The main purpose of CCA is to derive the linear combination of the variables from each set that maximizes the correlation between the two. That means, for two multivariate datasets, X and Y , CCA aims to find the directions (i.e., canonical vectors or canonical variates) in which both datasets express maximum correlation.

Formally, given two random vectors $X \in \mathbb{R}^p$ and $Y \in \mathbb{R}^q$, CCA seeks vectors $a \in \mathbb{R}^p$ and $b \in \mathbb{R}^q$ such that the correlation between the linear combinations $a^T X$ and $b^T Y$ is maximized, subject to the constraint that these combinations have unit variance. This leads to simultaneous generalized eigenvalue problems, typically solved using Singular Value Decomposition (SVD) or similar matrix factorization techniques. The canonical correlation (CC) is the correlation between the two optimally weighted sets of variables. Its value ranges between -1 and +1, where +1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 indicates no correlation. The number of CCs equals the minimum number of variables in the two sets. Canonical loadings are the correlation coefficients between the variables and the canonical variates. These loadings provide information about the contribution of each variable to the canonical correlation. The squares of the canonical loadings are known as communality estimates, indicating the proportion of the variance in a variable that the canonical variates can explain.

In the illustration provided in Figure 5.2, the similarity between the transformer layer representations and features extracted from Convolutional Neural Network (CNN) module is assessed via Canonical Correlation Analysis (CCA). An interesting pattern in the pre-trained model appears to emulate the behavior of an autoencoder. As we traverse deeper into the model, the

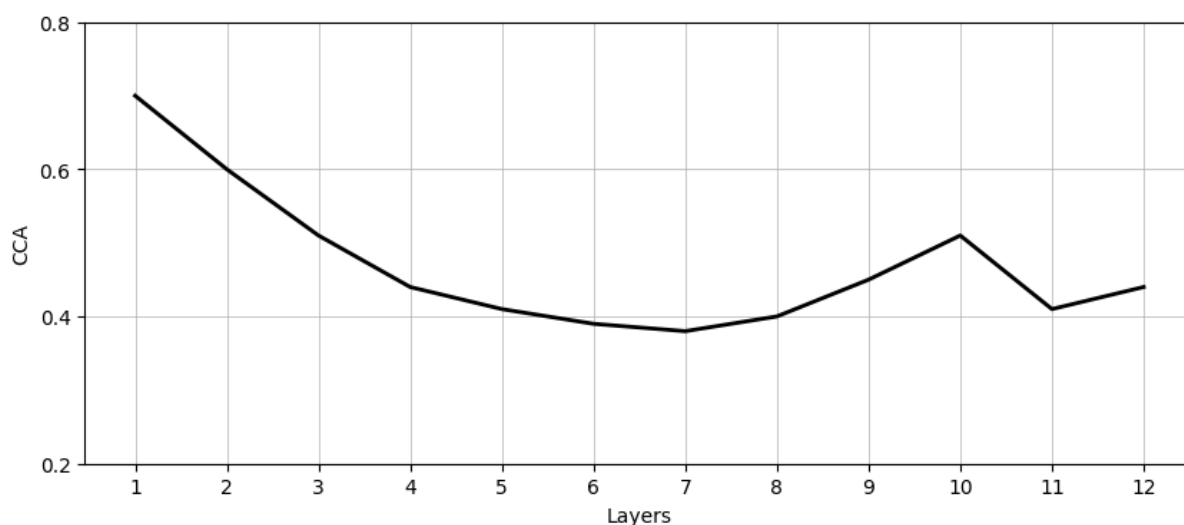


Figure 5.2: Illustration of CCA similarity with the features extracted from the CNN module

representation progressively diverges from the characteristics of the input features. However, this trend reverses itself as we probe further into the network. The deeper layers start displaying increased similarity to the input features as if they are endeavoring to reconstruct the input. This observation aligns well with the training objective of the model, which aims to differentiate the masked input segment from potential distractors.

Therefore, it is expected that the final layers of the model would exhibit properties akin to the input, as these layers play an integral role in determining the output. This autoencoder-like behavior can be seen as encoding the context followed by its reconstruction, capturing the most significant features from the input and using them to generate the output.

This pattern of divergence and subsequent convergence resembles observations made for the BERT text model. In the case of BERT, the objective is also centered around the reconstruction of masked inputs. However, it differs because the task is grounded in masked reconstruction rather than a contrastive prediction. These comparisons provide intriguing insights into the operational dynamics of deep learning models and their potential similarities, regardless of the differences in their specific tasks or applications.

5.4 Summary & Conclusion

This study presents a comprehensive approach to improve Automatic Speech Recognition (ASR) performance for low-resource Indian languages, particularly Telugu. It explores strategies of a self-supervised approach to overcome challenges associated with data scarcity in respective languages. The study applies the wav2vec2.0 model, incorporating a novel blend of contrastive objective and diversity loss, to generate robust and enriched speech representations and builds a pretrained model trained on 5000 hours of unlabeled Telugu data. Later Canonical Correlation Analysis (CCA) of the model layers discloses a characteristic autoencoder-like behavior contributing significantly. These findings obtained in this chapter provide essential insights into enhancing ASR systems for low-resource languages. The forthcoming chapter will utilize the pre-trained CSTD-5k model for executing downstream Automatic Speech Recognition (ASR) tasks. This exercise is not confined to a single language but rather extends to an array of diverse languages. We intend to evaluate the versatility and efficiency of the CSTD-5k model, particularly its capability to adapt and learn from different linguistic structures and phonetics. We will consider how the model processes these languages, its accuracy levels, and the overall efficacy of transcribing speech to text. The investigation aims to provide insights into the practical challenges and benefits of deploying such a model in a multilingual environment.

Chapter 6

Building ASR Systems for Low-Resource Languages of the Indian Subcontinent Using Telugu Pretrained Model

6.1 Introduction

The previous chapter, as delineated in Chapter 6, concentrated on building a Telugu pretrained model using the wav2vec2.0 framework. In the broader deep-learning context, pretraining is an effective strategy for model initialization, which can significantly expedite the learning process and improve model performance. The concept of transfer learning, wherein a model trained on one task is repurposed on a related task, underlies this strategy. The idea behind transfer learning is that if a model learns certain patterns on one task, it can expedite learning-related patterns on another task. This approach is particularly beneficial when dealing with tasks involving languages with low resource availability, such as Telugu. Given the pretrained model's success in understanding the nuances of the Telugu language, it is intuitive to repurpose this model for ASR tasks. By applying transfer learning, we aim to harness the rich linguistic representations ingrained in the pretrained model to recognize and transcribe Telugu speech accurately.

This chapter provides a detailed exploration of the methodologies and techniques employed to adapt the Telugu pretrained model (CSTD-5K) for ASR. This will include an examination of the fine-tuning process, an analysis of various hyperparameters, and a comprehensive evaluation of the performance of our ASR system in a range of settings and scenarios. This study serves as a roadmap for further research in the field, contributing to the ongoing endeavor to address the challenges posed by low-resource languages in ASR. The exploration in this chapter high-

lights the potential of pretrained models in the realm of ASR and underscores the importance of linguistic diversity in our increasingly interconnected global community.

6.2 Model Architecture for Fine-tuning

Once the pretraining phase is completed, the model is subjected to a fine-tuning process tailored for Automatic Speech Recognition (ASR) using task-specific supervised datasets. This tailored adaptation incorporates a randomly initialized projection layer into the context network. The role of this layer is to transform each d -dimensional output generated by the context network into a C -dimensional output, where C corresponds to the vocabulary size. In this study, the vocabulary comprises all distinct characters present in the Telugu language.

Following the transformation, a softmax function is employed to derive a probability distribution spanning the entire vocabulary. The function assists the model in making more precise character predictions during the ASR task. The fine-tuning process can be conducted in one of two ways: jointly, by utilizing data from various languages, or individually, focusing on a single language. The Connectionist Temporal Classification (CTC) loss function, a commonly recognized method, optimizes the model's performance throughout the fine-tuning phase. Integrating a modified variant of the SpecAugment data augmentation technique enriches the data set, enhancing the model's robustness against a wide range of audio inputs and improving its ability to generalize.

Fine-tuning the pretrained model using the aforementioned strategy effectively modifies its underlying architecture to meet the specific challenges and nuances associated with the ASR task. This tailored adaptation allows the model to better handle accents, pronunciations, and background noise variances. Consequently, the fine-tuned model showcases an improved performance in ASR applications, which attests to the effectiveness of this methodology for extracting context-aware representations usable across numerous downstream tasks.

6.2.1 Decoding

The decoding of emissions generated by the softmax layer necessitates the integration of a lexicon alongside a separately trained word-level n -gram language model. A candidate sequence, denoted by $y = y_1, y_2, \dots, y_M$, are assigned probabilities $p_{AM}(y)$ and $p_{LM}(y)$ by the fine-tuned network and the language model, correspondingly. The formula utilized to discern the optimal word sequence, y^* , is as follows:

$$y^* = \operatorname{argmax}_y [\log p_{AM}(y) + \alpha \log p_{LM}(y) + \beta |y|] \quad (6.1)$$

Within Equation 6.1, the representation $|y|$ refers to the length of the sequence, while the variables α and β function as hyperparameters that adjust the influence of the language model and the sequence length, respectively. The process incorporates an efficient beam search decoder to investigate potential sequences, amalgamating network scores, language model scores, and word insertion bonuses.

The incorporation of this decoding method assures the identification of the most likely word sequence, taking into account both the predictions arising from the fine-tuned network and the contextual information supplied by the language model. The incorporation of the language model amplifies the ability of the model to generate transcriptions that are not only more coherent but also grammatically precise. Furthermore, the beam search decoder introduces a highly efficient methodology for scrutinizing potential sequences. Consequently, this integrative approach culminates enhanced ASR performance and more accurate transcription outcomes.

6.2.2 Rescoring

Notably, the aforementioned decoding procedure adopts an n -gram KenLM language model. However, the recent strides in language modeling have unveiled the superior performance exhibited by transformer-based language models. To exploit these cutting-edge advancements, an optional amalgamation of an external transformer-based language model is proposed to rescore the n -best hypotheses generated by the initial process. Specifically, for each contender in the n -

best list, a new score is calculated, which merges the decoder log-probability with the weighted log-probability of the external language model, symbolized as $p_{ELM}(y)$. The ideal sequence is then determined using the subsequent equation:

$$y^*_{ext} = \operatorname{argmax}_y [\log p_{AM}(y) + \alpha_1 \log p_{LM}(y) + \alpha_2 \log p_{ELM}(y) + \beta |y|] \quad (6.2)$$

In this context, $|y|$ represents the sequence length, whereas α_1 , α_2 , and β function as hyperparameters modulating the impact of the primary language model, the external language model, and the sequence length, correspondingly. This enhanced decoding approach efficaciously merges the benefits of n-gram and transformer-based language models, resulting in more precise and contextually relevant transcriptions. Incorporating the latest language modeling techniques within this collaborative approach further bolsters ASR performance, leading to refined transcription results that more accurately encapsulate the intricacies of human speech.

6.3 Methodology and Experimentation

This section provides an exhaustive overview of the following: (i) the datasets engaged for our experimentation, (ii) the chosen hyperparameters for both the pretraining and fine-tuning phases and (iii) the text corpora and corresponding hyperparameters selected for language model training. The dataset enlisted for the fine-tuning process has been elaborated upon in Chapter 2.

6.3.1 Procedure for Fine-tuning

Throughout the fine-tuning stage, all network parameters are subjected to updating, except the convolutional feature encoder. For the *BASE* model, the employment of the Adam optimizer is opted for, with a learning rate set at 1×10^{-4} , coupled with a tri-stage learning rate schedule as follows:

1. A warm-up phase encompassing the initial 10% of the steps,
2. A phase with a constant learning rate for the ensuing 40% of the steps, and
3. A phase with the learning rate undergoing exponential decay for the remaining steps.

The maximum limit of frames per GPU is fixed at 1 Million, with the models being fine-tuned on 4 A100 GPUs devoid of gradient accumulation. This results in an effective batch size comprising 8 Million samples. The *BASE* model undergoes training spanning 80k steps. In the first 200 steps, only the parameters within the final layer experience updates. After this, all model parameters are updated except those tied to the feature encoder. For data augmentation, the feature encoder’s outputs are subjected to masking, consistent with the principles of the SpecAugment methodology. A masking probability of 0.05 and a LayerDrop rate of 0.1 are defined for both models. Early stopping is implemented, featuring a patience setting of 30 epochs. The fine-tuning experiments are conducted employing the Fairseq framework.

6.3.2 Language Model Setup

The present study employs a lexicon-based beam search decoder to generate predictions, as operationalized in the Flashlight library ¹. To enhance the precision of our predictions, the study integrates a 5-gram KenLM language model with pruning applicable to singletons of the fifth order and doubletons of the sixth order. The system’s performance is appraised in three distinct modes: (i) in the absence of a language model (WOLM), (ii) with the training text functioning as a text corpus for the language model’s development (WLM), and (iii) with an external text corpus, sourced from the internet and complemented by text from ai4bharath, for the language model’s creation (WLME).

The hyperparameters α and β in Equation 6.1 are fine-tuned on the validation set through grid search, employing a value range from -4 to +4 for α and from 0 to 5 for β . During this grid search, a beam size of 64 is used. It is crucial to highlight that α and β are fine-tuned on a per-language basis. After identifying the optimal values for α and β , these hyperparameters are applied to the test data, and the beam size is expanded to 1024. This methodology maximizes our system’s performance while accommodating the variability inherent in different languages.

¹<https://github.com/flashlight/flashlight>

6.3.3 Rescoring Setup

In the rescoring process, the employment of an external Transformer language model (LM) is explored to rescale the top-k hypotheses propagated by the decoder. To facilitate this, a BPE-tokenized Transformer LM is trained to utilize the Fairseq toolkit ². The model’s architecture encompasses 12 decoder layers, 16 attention heads, an embedding dimension size of 1024, and an FFN dimension of 4096. The training dataset for the Transformer LM is a mirror image of that used for the KenLM model, inclusive of an external corpus. However, the vocabulary size is confined to the top 50k sub-words. The model undergoes training for 160k steps with the Adam optimizer and a tri-stage learning rate scheduler. This scheduler comprises 16k steps of linearly increasing warm-up, followed by a hold-up until 40k steps at a learning rate of 0.0001, succeeding which, the learning rate undergoes exponential decay over the final 40k steps.

Leveraging the trained language model, the beams propagated by the decoder are rescaled according to Equation 6.2. To pinpoint the ideal values for α and β , a grid search is conducted over a range from 0 to 5 and -4 to +4, respectively. This methodology permits the identification of the optimal hyperparameters for our rescaling process, significantly enhancing our predictions’ accuracy.

6.4 Results and Analysis

This section initially elaborates on the results derived from the CSTD-5k model for four distinct fine-tuning datasets, namely MUCS, OpenSLR-126, OpenSLR-127, and IITH-CSTD.

6.4.1 Performance of Telugu Pretrained Model Across different language Families

Table 6.1 presents a comparison of the word error rates (WER) for various languages fine-tuned on a Telugu pretrained model across several databases (MUCS, OpenSLR-126, OpenSLR-127, and IITH-CSTD). The languages have been categorized into two linguistic families: Dravid-

²<https://github.com/pytorch/fairseq>

ian and Indo-Aryan. The assessment metric is the word error rate (WER), and three diverse language model setups have been contemplated: WOLM (absence of language model), WLM (presence of language model), and WLME (presence of language model supplemented with extra text).

Table 6.1: Comparison of the word error rates (WER) on CSTD-5k for different databases (MUCS, OpenSLR-126, OpenSLR-127, and IIITH-CSTD).(where WOLM, WLM, and WLME indicate without language model, with a language model, and with language model using extra text, respectively.)

Database	Finetuning	WER(%)		
		WOLM	WLM	WLME
MUCS	Telugu	20.3	14.9	9.89
	Tamil	23.65	18.89	13.18
	Hindi	37.98	30.49	29.03
	Marathi	35.49	28.1	27.19
	Gujarati	35.86	27.89	26.32
	Odia	30.61	25.78	22.68
OpenSLR-126	Kannada	22.32	16.89	11.26
OpenSLR-127	Tamil	26.10	23.95	17.82
IIITH-CSTD	Telugu	18.54	13.24	9.86

The key findings from this study can be summarized as follows:

1. Across all four databases, incorporating a language model (WLM) consistently leads to a reduction in WER compared to not using a language model (WOLM). This highlights the effectiveness of integrating language models in improving the performance of the speech recognition system using the CSTD-5k model as a foundation model.
2. Further improvements in WER are observed when using a language model with extra text (WLME) compared to using a language model without extra text (WLM). This demonstrates the importance of leveraging additional textual data to enhance the language model's performance.

3. The differences in WER across the various languages and databases suggest that the performance of the CSTD-5k model can be influenced by multiple factors, such as the complexity of the languages, quality of the training data, and the effectiveness of the language models used.
4. The results also indicate that the most significant gains in performance are achieved when transitioning from WOLM to WLM, with additional improvements observed when moving from WLM to WLME.

From the finetuning dataset, we have two language families, ie. Dravidian Language (Telugu, Tamil, and Kannada) and Indo-Aryan Language (Hindi, Marathi, Gujarati, and Odia) Family. The average relative improvements within each language families as follows:

1. Dravidian Language Family

- Telugu
 - WOLM to WLM: 26.6%
 - WLM to WLME: 33.6%
- Tamil
 - WOLM to WLM: 20.1%
 - WLM to WLME: 30.2%
- Kannada
 - WOLM to WLM: 24.3%
 - WLM to WLME: 33.3%

Average WOLM to WLM improvement: $(26.6 + 20.1 + 24.3)/3 \approx 23.7\%$

Average WLM to WLME improvement: $(33.6 + 30.2 + 33.3)/3 \approx 32.4\%$

2. Indo-Aryan Language Family

- Hindi
 - WOLM to WLM: 19.7%

- WLM to WLME: 3.2%
- Marathi
 - WOLM to WLM: 20.8%
 - WLM to WLME: 3.2%
- Gujarati
 - WOLM to WLM: 22.2%
 - WLM to WLME: 5.6%
- Odia
 - WOLM to WLM: 15.8%
 - WLM to WLME: 12.0%

Average WOLM to WLM improvement: $(19.7 + 20.8 + 22.2 + 15.8)/4 \approx 19.6\%$

Average WLM to WLME improvement: $(4.8 + 3.2 + 5.6 + 12.0)/4 \approx 6.4\%$

The results suggest that the Dravidian language family benefits more from the CSTD-5k model, as the average relative improvement for WOLM to WLM (23.7%) and WLM to WLME (32.4%) is higher compared to the Indo-Aryan language family, which has an average relative improvement of 19.6% for WOLM to WLM and 6.4% for WLM to WLME. These findings indicate that using a pretrained model from a language within the same language family may improve performance. However, various factors such as language complexity, quality of the training data, and effectiveness of the language models may influence these results. Further investigation and analysis would be required to understand the underlying reasons for the differences in relative improvements between the language families.

Additionally, it's important to note that some languages within the same family may exhibit better improvements than others. For instance, within the Dravidian language family, Telugu and Kannada show higher relative improvements between WLM to WLME compared to Tamil. Similarly, within the Indo-Aryan language family, Odia has a higher relative improvement between WLM to WLME compared to Hindi, Marathi, and Gujarati. This suggests that pretrained

models and transfer learning techniques can effectively improve the performance of automatic speech recognition systems for low-resource languages, particularly those that share linguistic similarities with the pretrained model’s language.

6.4.2 Impact of Multilingual Fine-tuning on the CSTD-5k Telugu Pre-trained Model

In this study, we conducted an in-depth evaluation of the MUCS corpus, which encompasses six languages and has an aggregate duration of approximately 450 hours (refer to Table 2.2). Our approach involved jointly fine-tuning the pretrained model by amalgamating the training data from all six languages. This strategy draws inspiration from [115, 96, 94], who demonstrated that the joint fine-tuning of a multilingual model with task-specific data across multiple languages results in enhanced performance.

In our experimental setup, we incorporated a multi-softmax layer at the final stage of the model, wherein each language was assigned a separate softmax layer. The output vocabulary contained unique tokens from each respective language. During the training process, each minibatch consists of data randomly sampled from one of the languages.

Table 6.2: Comparison of multilingual fine-tuning on CSTD-5k (Telugu pretrained model)

Model	MUCS					
	Te	Ta	Hi	Ma	Od	Gu
Monolingual Finetune on CSTD5k	14.9	18.89	30.49	28.1	25.78	27.89
Multilingual Finetune on CSTD5k	13.04	19.10	31.04	29.68	26.84	26.98

The table compares the performance of two fine-tuning approaches: Monolingual Fine-tuning and Multilingual Fine-tuning, both applied to the CSTD-5k model. It is important to note that the monolingual fine-tuning is focused solely on the Telugu language. In contrast, the multilin-

gual fine-tuning approach includes all six languages in the MUCS dataset. Upon analyzing the results, it is evident that the multilingual fine-tuning approach yields slightly better performance for most languages than monolingual fine-tuning. The improvements are particularly notable in Telugu, with a decrease in the error rate from 14.9 to 13.04. However, the improvement is marginal for some languages, such as Tamil. In a few cases, such as Hindi and Marathi, the multilingual fine-tuning results in a slightly higher error rate. The table suggests that multilingual fine-tuning can lead to competitive performance compared to monolingual fine-tuning. This finding supports the idea that jointly fine-tuning a model with task-specific data across multiple languages can improve its adaptability and generalization capabilities, making it more suitable for practical applications in multilingual speech recognition tasks.

The outcomes obtained from our approach are promising, as they reveal that a single joint model can deliver performance levels on par with those of individual models tailored for specific languages. Owing to their ease of maintenance and deployment, joint models of this nature are preferred for practical applications in multilingual speech recognition tasks.

6.4.3 Impact of Dataset Size on Pretrained Model Performance in Speech Recognition Tasks

This investigation entails an evaluation of the IIITH-CSTD corpus, which comprises approximately 2000.8 hours of Telugu speech. For this specific endeavor, we have randomly selected diverse subsets of various durations (10 minutes, 1 hour, 10 hours, and 100 hours). By delving deeper into the analysis of Table 6.3, further insights and implications concerning the performance of the pretrained models, specifically XLSR-53, CLSRIL-23, and CSTD-5k, on the IIITH-CSTD database can be extracted. An in-depth review of performance trends and relative enhancements will foster a comprehensive understanding of these models' proficiency in managing speech recognition tasks across various dataset sizes.

Upon examination of the results, it is evident that the CSTD-5k model consistently achieves lower WERs than the XLSR-53 and CLSRIL-23 models for all dataset sizes. This observation is particularly noticeable for the 100-hour dataset, where the CSTD-5k model attains a WER of

Table 6.3: WER evaluation of different pretrained models of different dataset sizes on the II-IHCSTD database

Pretraining model	Duration			
	10mins	1 hour	10 hours	100 hours
XLSR-53	96.2	68.23	57.98	33.68
CLSRIL-23	85.26	60.48	51.24	27.6
CSTD-5k (our)	67.29	44.68	27.86	10.2

10.2%, far superior to the 27.6% and 33.68% WERs exhibited by the CLSRIL-23 and XLSR-53 models, respectively.

All three models show substantial reductions in WER as the dataset size increases. The relative improvement from the 10-minute dataset to the 100-hour dataset is as follows:

1. **XLSR-53:** WER decreases from 96.2% to 33.68%, representing a relative reduction of 65.02%.
2. **CLSRIL-23:** WER decreases from 85.26% to 27.6%, corresponding to a relative reduction of 67.66%.
3. **CSTD-5k:** WER decreases from 67.29% to 10.2%, resulting in a remarkable relative reduction of 84.85%.

Notably, the improvement rate is not consistent across models. The performance gap between the CSTD-5k model and the other two models widens as the dataset size increases, suggesting that the CSTD-5k model exhibits enhanced capability in leveraging larger datasets for superior performance.

6.5 Summary & Conclusion

In this chapter, we presented a comprehensive evaluation of various aspects of speech recognition, including fine-tuning, language models, and dataset sizes. We discussed the performance

of different models across diverse datasets and language families, analyzing the impact of incorporating language models and pretrained models. The evaluation encompassed multiple metrics, such as word error rate (WER), to assess the effectiveness of the models. Furthermore, we explored the implications of dataset sizes on model performance and investigated the relative improvements achieved through different approaches.

The key findings of this chapter are as follows: (i) Language models play a crucial role in enhancing the performance of speech recognition systems. The integration of language models consistently led to reduced WERs across different languages and databases, highlighting their effectiveness in improving accuracy and precision. (ii) Pretrained models, particularly those within the same language family, exhibited superior performance compared to models trained solely on specific languages. Transfer learning techniques utilizing pretrained models showcased improved adaptability and generalization capabilities across diverse linguistic contexts. (iii) Dataset size significantly impacted model performance, with larger datasets resulting in lower WERs. The relative improvements in WER were more pronounced when transitioning from smaller to larger datasets, indicating the importance of data volume in training robust speech recognition models. (iv) The analysis of different language families demonstrated variations in relative improvements. Some languages within the same family showcased greater enhancements than others, suggesting the influence of language complexity and dataset quality on model performance.

This chapter provides valuable insights into the performance and adaptability of speech recognition models across various scenarios. Evaluating different fine-tuning approaches, language models, and dataset sizes offers guidance for optimizing the performance of automatic speech recognition systems. The findings emphasize the significance of incorporating language models and leveraging pretrained models for improved accuracy and precision. The outcomes highlight the potential of transfer learning techniques and the benefits of utilizing larger datasets in training robust speech recognition models. Furthermore, the evaluation across diverse languages and language families demonstrates the versatility and effectiveness of the proposed approaches.

Overall, this study contributes to the advancement of speech recognition technology, providing a foundation for developing more accurate and efficient systems in real-world applications. The findings pave the way for future research in optimizing model architectures, refining language models, and further exploring additional techniques to enhance speech recognition performance.

Chapter 7

Exploring the Shared Phonetic Space of Indian Languages for Multilingual Speech Recognition Systems

7.1 Introduction

Developing ASR systems for low-resource languages presents a unique challenge due to the dearth of annotated speech data and text. India, with its diverse linguistic landscape, poses an intriguing yet formidable task for ASR researchers. To address this challenge, we present a multilingual ASR system that exploits the shared phonetic space of several Indian languages. Our system employs deep neural architectures during acoustic model training and integrates multiple languages into a single group. In addition, we introduce an attention-based multi-stream convolutional neural network for ASR. We use a time-restricted self-attention mechanism to enhance feature robustness while concatenating the output from multiple streams. We assess the performance of the proposed system using state-of-the-art tests and demonstrate superior results over baseline models for databases and low-resource languages. Our ASR system caters to six Indian languages: Gujarati, Hindi, Marathi, Odia, Tamil, and Telugu. We conduct various tests using different audio and language modeling approaches to evaluate the system's performance. The proposed multilingual ASR system for low-resource languages holds significant promise for bridging the digital divide and enabling technology access for under represented communities. Integrating several Indian languages into a single system marks a crucial step towards providing a comprehensive solution for speech recognition in India. Furthermore, our system's architecture and methodology can extend to other low-resource language environments, providing ASR researchers with a robust tool for addressing challenges in linguistically challenging

settings.

This study aims to develop speech-to-text (STT) systems for Indian languages with limited resources. In the past, various methods have been proposed to build speech recognition systems for low-resource settings. For instance, in [116], acoustic data from multiple languages were used to establish deep neural network (DNN) based acoustic models via unsupervised restricted Boltzmann machine (RBM). The study found that unsupervised pretraining is necessary for hybrid setups, especially when dealing with limited transcribed training data. Additionally, other studies have leveraged transfer learning and data augmentation approaches to address low-resource scenarios, such as [117, 118, 119, 120, 121, 122]. In previous studies [123, 124, 122, 121, 115], multilingual features were extracted to improve the performance of low-resource ASR systems. To handle multiple languages in a single acoustic model, a phonetic sharing approach was proposed in [94]. Recently, the authors in [125] used phonetic sound principles and transliterated all pooled languages of training into one language, mapping each sound to the Latin space. These approaches aim to improve the accuracy and robustness of LID and ASR systems for low-resource languages.

In this study, a rule-based parser was employed to generate the pronunciation dictionary for Indian languages, given that most of them are syllabic. Prior research has shown that Indian languages share a common phonetic space but have distinct phonotactic structures. This unique characteristic of Indian languages can be leveraged to develop more effective ASR systems [126, 127, 128, 129]. By taking advantage of the shared phonetic space, ASR systems can be better optimized to recognize speech accurately and efficiently. The distinct phonotactic structures of each language can also be incorporated into the system to improve its performance on a per-language basis.

Language is a continuum phenomenon across the demography. But the written form of a language is termed orthography. Indian language character set is distinctly coined as aksharas. Aksharas are the smallest unit in the writing system of a language in the Indian context. For building a speech recognition system, one of the primitive blocks is the lexicon (also known

as the dictionary). The lexicon has valid entries for a given word and corresponding phoneme sequence. In the literature, many works have been proposed to build a parser for a lexicon. This work will exploit the Common Label set (CLS) and Common Phone set (CPS) approaches.

Many studies have shown that the Indian languages share a common phonetic space. In [128], the authors have proposed the common phone set approach for Indian languages and adopted it in building HMM-DNN-based text-to-speech synthesis (TTS). The phone set mapping for 13 Indian languages can be found in the below footnote ¹. To leverage its advantage, a parser



Figure 7.1: Working pipeline of a unified parser.

has been developed wherein a UTF8 sequence is converted to IT3 [128] format. The invariant features of Indian languages are identified and defined as the interim step to a unified parser. In this parser, all the similar sounds among all languages are mapped to a single label. The labels are designed so that most of them are transliterated forms of the native script. The working pipeline of a unified parser is illustrated in Figure 7.1. Here we have two approaches to extract the phoneme sequence, and they are as follows:

- Common Phone Set (CPS)
- Common Label Set (CLS)

Initially, in either of the approaches, the raw text is taken and passed for normalization, wherein it filters out the unwanted characters (like special symbols, emoticons, etc.). So, let's try to understand the working flow of the common phone set and common label set by the pseudo-codes mentioned in Algorithm 2.

7.1.1 Comparison of Common Phone Set vs. Common Label Set

Initially, let us consider a sentence which is illustrated in Figure 7.2, (meaning “the Delhi Capitals team has bought Indian hitting batsman Verma.” in Telugu. Telugu is one of the

¹Phone set Mapping [Click here](#)

Algorithm 2 Working flow for Common Phone set

```
1: procedure CPS(raw text)                                ▶ This raw text can be a sentence
2:   System Initialization
3:   Read the input (sentence/word)
4:   IF sentence = True
5:     Convert the text into IT3 format
6:     split IT3 format text into unique words
7:     The unique words in IT3 are used to generate the phoneme mapping
8:   Else
9:     Convert the text into IT3 format
10:    The text in IT3 is used to generate the phoneme mapping
11:  EndIf
```

Dravidian languages, which is spoken majorly in the southern part of India). Let's understand how the Common Phone Set approach applies to the sentence mentioned in Figure 7.2. (The working flow of the Common Phone Set has been mentioned in Algorithm 2).

భారత హిట్టింగ్ బ్యాటర్ వర్మను దిల్లీ క్యాపిటల్స్ జట్టు కొనుగోలు చేసింది.

Figure 7.2: An example of a Telugu utterance

1. Initially, we will be converting the given input text into the nearest Latin format with respect to the input language, which is popularly called IT3.

Input (UTF8) Refer Figure 7.2

Output (IT3) bhaarata hitxtxiqq byaatxar warmanu dillii kyaapitxals jatxtxu konugoolu ceesiqdi.

2. The unique words from the IT3 output is taken for phoneme mapping.

Unique words [bhaarata, byaatxar, ceesiqdi, dillii, hitxtxiqq, jatxtxu, konugoolu, kyaapitxals, warmanu].

3. Later, these unique words list is passed through the indic language parser ([126, 127]) to obtain the phoneme sequence

Output IT3 word and corresponding phoneme sequence:

bhaarata → /bh/ /aa/ /r/ /a/ /t/ /a/
byaatxar → /b/ /y/ /aa/ /tx/ /a/ /r/
ceesiqdi → /c/ /e/ /e/ /s/ /i/ /q//d/ /i/
dillii → /d/ /i/ /l/ /l/ /ii/
hitxtxiqg → /h/ /i/ /tx/ /tx/ /i/ /q/ /g/
jatxtxu → /j/ /a/ /tx/ /tx/ /u/
konugoolu → /k/ /o/ /n/ /u/ /g/ /oo/ /l/ /u/
kyaapitxals → /k/ /y/ /aa/ /p/ /i/ /tx/ /a/ /l/ /s/
warmanu → /w/ /a/ /r/ /m/ /a/ /n/ /u/

The main drawback of the CPS approach is while converting the input text to IT3 format, language-level information of the input text has to be incorporated.

Now let's try to understand the working flow for *Common Label set (CLS)*, even here we will consider the same example which was considered above (refer Figure 7.2). There is a standardized Common Label Set (CLS) [127, 96] for all the different sounds found in Indian languages. This common label set uses a conventional set of labels for speech sounds found in various Indian languages. It works on the principle of letter-to-sound mapping. The Common label set mapping for 13 Indian languages can be found in the below footnote². (The working flow of the Common Label Set is mentioned below in Algorithm 2). This study uses a global parser for all languages, which takes advantage of the syllable structure. In this approach, we found that there are three possible ways where a given word can obtain the phoneme(sound) sequence, which are as follows:

1. Syllable based
2. Aksharas based
3. Monophone based

Initially, let us understand the working of syllable-based-common label set (SB-CLS). The output of the SB-CLS parser gives us all the syllable tokens present in the given word. In this case, it is as follows:

²Common Label Set [Click here](#)

Algorithm 3 Working flow for Common Label set

```
1: procedure CLS(raw text) ▶ This raw text can be a sentence
2:   System Initialization
3:   User input for phoneme mapping (monophone/akshara/syllable based)
4:   Split the input text into unique words
5:   If monophone = True
6:     Each word is split into its letter to sound mapping (Latin space)
7:   EndIf
8:   If akshara = True
9:     Each word is split into its corresponding character unit
10:  EndIf
11:  If syllable = True
12:    The input word is passed through the syllable-based parser
13:  EndIf
```

1. Firstly, let us exploit the artifacts of *syllable based-common label set (SB-CLS)*. The output of the SB-CLS parser gives us all the syllable tokens present in the given word. For example, consider the Telugu word refer Figure 7.3 (meaning “Sanskrit”).

సంస్కృతం

Figure 7.3: An example of a Telugu word (English translation of it is “Sanskrit”).

- Let’s pass the word through the SB-CLS parser. The output achieved is illustrated in Figure 7.4.

/సంస్/ /కృ/ /తం/

Figure 7.4: An example of a Telugu word after passing through SB-CLS parser.

- It is observed that each output token from SB-CLS is the smallest spoken unit by the human being.
2. Secondly, the same word is passed through the *akshara based-common label set (AB-CLS)* parser. The output of the AB-CLS parser gives us characters as separate units.

- The Output of AB-CLS is illustrated in Figure 7.5

సం/ సే/ క్య/ తం/

Figure 7.5: An example of a Telugu word after passing through AB-CLS parser.

2.1. In the monophone based-common label set (MB-CLS), the output of the MB-CLS parser gives us phoneme mapping, which is the common label across all the languages.

- The Output of MB-CLS is /s/ /a/ /q/ /s/ /k/ /rɔ/ /t/ /a/ /q/.

So, in this work, we have exploited SB-CLS, AB-CLS, MB-CLS, and CPS for building low-resource speech recognition tasks in the Indian context.

7.2 Proposed Work

In this work, we extend the idea of time-restricted self-attention layers in TDNN and TDNN-LSTM (which is nothing but the interleaving TDNN and its projection on Long Short-Term Memory layers) across multiple streams. The proposed architecture is shown as a block diagram in Figure 7.6. We used the SpecAugment technique provided in the reference [130] to prevent the network from over-fitting during the pre-processing stage. At a given time step t , the input speech frame is passed through a series of CNN layers to obtain a latent representation denoted as h_t . At this point, h_t is passed through different independent streams, wherein each stream is a stack of TDNN-LSTM layers with varying dilation rates d_m . This is mathematically expressed as,

$$z_t^m = \text{Stacked-TDNN-LSTM}_m(h_t[-d_m, d_m]), \quad (7.1)$$

From the above Equation, m denotes the number of streams, and z_t^m indicates the outputs of all the streams. $[-d_m, d_m]$ is 3×1 kernel given the dilation rate of d_m .

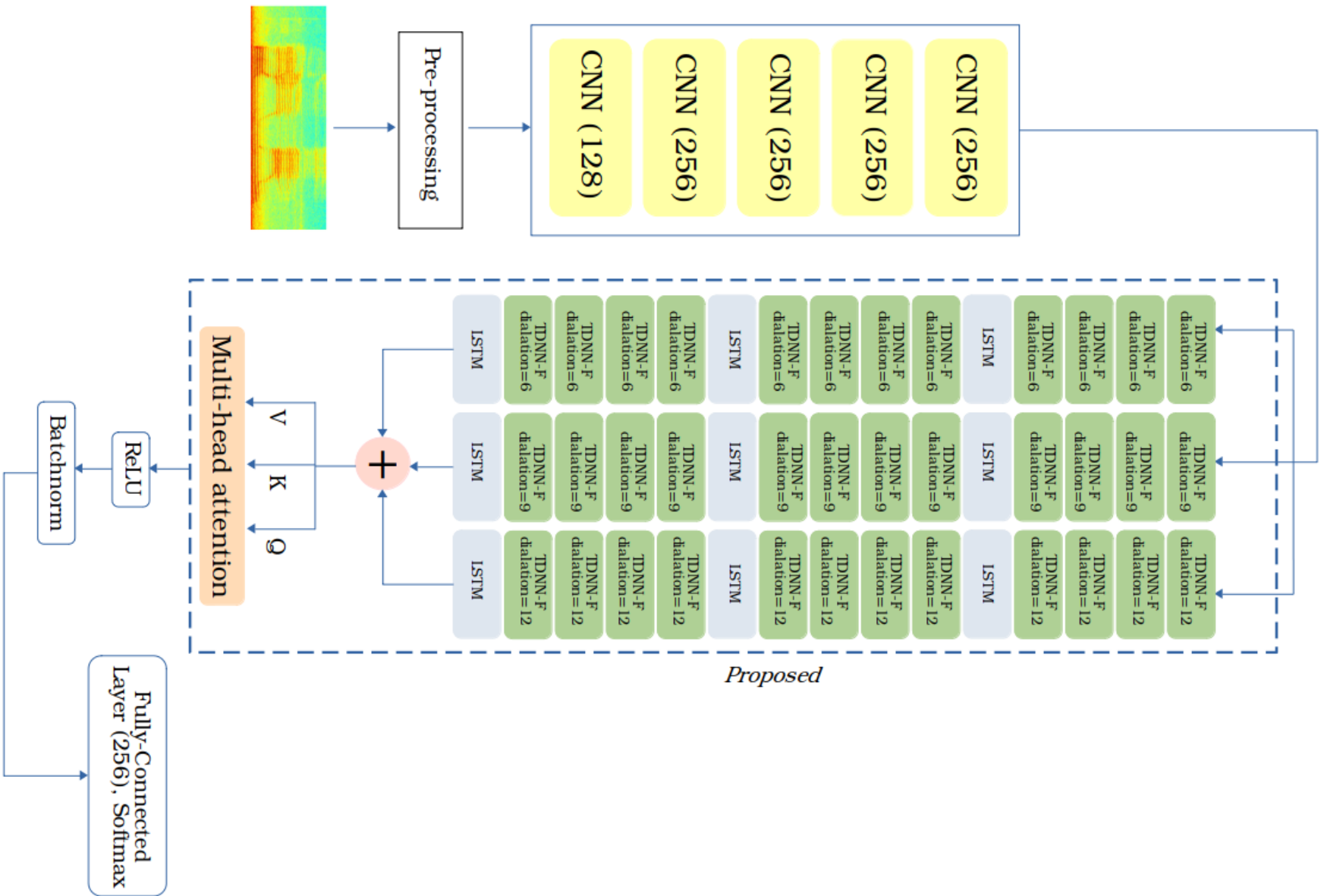


Figure 7.6: Proposed Multistream CNN TDNN-LSTM architecture with time-restricted self-attention

After that, the output of each of the distinct m streams will be concatenated and sent via a dropout layer, ReLU, and batch normalization (BN), respectively. It is represented as:

$$e_t = \text{Dropout}(\text{BN}(\text{ReLU}(\text{Concat}(z_t^1, z_t^2, \dots, z_t^m))))). \quad (7.2)$$

Afterwards, the multi-head attention layer is used to refine this learned feature embedding (e_t) before it is sent to the output layer. Let's assume that at time step t , the input to the attention layer is the three-part sequence query (q_t), key (k_t), and value (v_t), denoted by e_t . The model's inability to foresee the temporal order of values and keys necessitates using a positional encoding scheme. Therefore, every random vector x at time step t may be extended with one hot encoding for the relative position of τ versus t , which can be expressed as $\text{extend}(x, \tau, t)$. This extension can be performed on any random vector x at any time step t . At every given time step t , the attention output y_t is a weighted sum of the values v_t received throughout time for the left (L) and right (R) contexts, as presented in [112]:

$$y_t = \sum_{\tau=t-L}^{t+R} w_t(\tau) \text{extend}(v_t, \tau, t), \quad (7.3)$$

where the weight matrix is found by taking the dot product of the queries and the keys, which is then normalized using softmax and can be obtained as [112]:

$$w_t(\tau) = \frac{\exp(q_t, \text{extend}(k_t, \tau, t))}{S_t}, \quad (7.4)$$

where S_t makes the $\sum_{\tau} w_t(\tau) = \mathbf{1}$ condition hold. In a manner analogous to this, the attention process may be extended to several heads even though each head functions in isolation. In contrast to prior time-restricted attention, both input and output will be of equal duration in our situation. As a result, it is reasonable to consider the present moment to be the focal point of attention. In addition, the attention mechanism is flexible in that it may simultaneously attend to a variety of moments in time, each of which has a unique amount of importance.

7.3 Results & Analysis

7.3.1 Comparison of CPS Vs. CLS

This subsection initially used several lexicons (CPS, CLS (Syllable, Aksharas, Monophone)), and certain ablation studies were carried out on Telugu, Tamil, and Gujarati languages. Thus

we could conclude which lexicon is the most effective when constructing ASR systems using Low resource settings. Identical setups have been considered while feature extraction, building acoustic models, and language modeling.

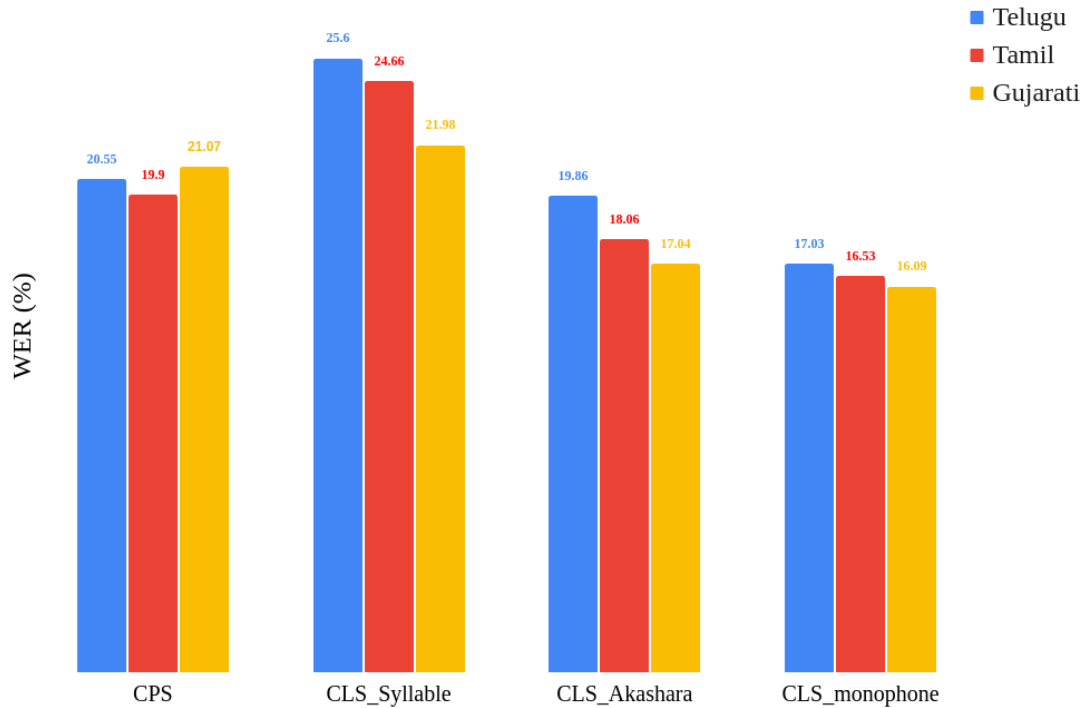


Figure 7.7: Comparison of CPS and different variants of CLS on Telugu, Tamil, and Gujarati languages

The bar chart 7.7 illustrates the comparison of CPS and different variants of CLS on three different language monolingual systems (Telugu, Tamil, and Gujarati). Among all the four lexicons, MB-CLS (monophone based-common label set) performs better when compared to all others, and SB-CLS (syllable based-common label set) performs worse.

In this study, going forward, all the experiments conducted will utilize the MB-CLS (Monophone Based-Common Label Set) lexicon for our evaluation.

7.3.2 Experimental results on different variants of TDNN architecture for both monolingual and multilingual ASR systems

7.3.2.1 Monolingual Experimental Results

The experiments were performed on six Indian languages, and the database statistics are reported in Chapter 2. In this work, we built GMM-HMM, SGMM, 7-layer TDNN, 8-layer TDNN, 13-layer TDNN, low-rank TDNN (also known as TDNNF), TDNNF-LSTM, and TDNN-BLSTM for both monolingual and multilingual systems. The experimental results for six monolingual languages are reported in Table 7.1.

Table 7.1: Results of monolingual models in terms of WER (%) over a variety of acoustic models

Model	Language					
	Hindi	Marathi	Odia	Tamil	Telugu	Gujarati
GMM-HMM	69.03	33.22	55.78	48.81	47.27	28.33
SGMM	61.01	26.41	51.36	39.68	38.08	28.61
7-Layer TDNN	48.02	24.46	33.07	34.58	33.29	25.01
8-Layer TDNN	43.67	22.31	29.44	30.28	31.78	23.12
13-Layer TDNN	30.16	19.65	35.58	21.89	21.67	21.05
TDNNF	18.26	16.23	18.34	16.53	15.87	16.09
TDNNF-LSTM	17.07	15.98	16.08	14.89	13.65	15.10
TDNNF-BLSTM	18.73	17.26	18.02	16.23	15.18	17.97

The Table 7.1 compares the performance of different monolingual models over a variety of acoustic models. The metric used for evaluation is Word Error Rate (WER), which is a common metric in speech recognition systems. Lower WER values indicate better performance. The table provides WER results for seven different monolingual models: GMM-HMM, SGMM, 7-Layer TDNN, 8-Layer TDNN, 13-Layer TDNN, TDNNF, TDNNF-LSTM, and TDNNF-BLSTM. These are evaluated in six different languages: Hindi, Marathi, Odia, Tamil, Telugu, and Gujarati.

From the results, we observe that the TDNNF-LSTM model outperforms all other models across all languages, yielding the lowest WER in each case. The WER ranges from a low of 13.65% in Telugu to a high of 17.07% in Hindi. This suggests that the integration of LSTM (Long Short-Term Memory) layers into the Time Delay Neural Network Feature (TDNNF) model has enhanced its ability to capture complex patterns and dependencies in speech data, leading to better transcription accuracy. Contrastingly, the GMM-HMM model shows the highest WER across most languages, indicating its relative inferiority in transcription accuracy. The poor performance could be due to the fact that this model may not capture the non-linear patterns in the speech data as effectively as deep learning-based models like TDNNF and its variants.

It is also worth noting that models with more layers (such as 13-Layer TDNN) tend to perform better than those with fewer layers (such as 7-Layer TDNN), indicating that model complexity can play a role in transcription accuracy. In terms of language-wise comparison, Telugu consistently shows the lowest WER across different models, indicating that these models may be particularly well-suited for this language or that the Telugu data used was of high quality. On the other hand, Hindi consistently has the highest WER, which could mean these models struggle more with this language, or there could be issues with the data quality or representativeness of Hindi.

7.3.2.2 Multilingual Experimental Results

To train a multilingual ASR system, we combined all six languages into one and used it as a common acoustic model. It is known as Joint Acoustic Model (JAM). The performance of models trained using SGMM has been lower than that of models trained with TDNN. GMM-HMM-modeled triphones are not evenly distributed throughout the six languages. As a result, when the JAM is trained with SGMM, it performs poorly. Better results have been achieved by JAM trained with TDNNF compared to TDNN and SGMM-based systems. TDNNF and other TDNNF acoustic models, in contrast to SGMMs, are taught to predict changes over a relatively long period. Because of features like skip connection, bottleneck linear transformation layer, and batch normalization, TDNNF has successfully dealt with the variations introduced by six languages.

Table 7.2: Results of multilingual models in terms of WER (%) over a variety of acoustic models

Model	Language					
	Hindi	Marathi	Odia	Tamil	Telugu	Gujarati
SGMM	38.69	30.6	39.68	36.96	35.68	33.65
7-Layer TDNN	36.28	22.08	30.50	33.84	31.07	23.27
8-Layer TDNN	35.16	21.98	29.86	35.98	34.89	21.62
13-Layer TDNN	21.48	19.32	20.03	18.06	19.01	18.21
TDNNF	17.07	12.90	17.25	15.35	14.07	16.02
TDNNF-LSTM	15.97	11.08	16.92	13.59	12.86	14.32
TDNNF-BLSTM	17.62	14.18	18.58	17.38	15.48	16.61

Even in the case of multilingual systems, the TDNNF-LSTM architecture has performed better across all the languages. The results of it have been Tabulated in Table 7.2. For example, in the Hindi language, TDNNF-LSTM achieves a WER of 15.97% while the next best model, TDNNF, has a higher WER of 17.07%. This pattern of TDNNF-LSTM superiority is consistent across all the languages considered. This indicates that the incorporation of Long Short-Term Memory (LSTM) layers into the Time Delay Neural Network Feature (TDNNF) model results in superior performance due to LSTM’s ability to capture long-term dependencies in speech data.

On the other end of the spectrum, the SGMM model tends to have higher WER values, indicating less accurate performance relative to the other models. This could be due to the inability of SGMM to capture non-linear patterns in the data, a capability at which deep learning-based models excel. Looking at the results across different layers of TDNN, the performance generally improves with an increase in layers. For example, a 13-Layer TDNN performs significantly better than a 7-Layer and 8-Layer TDNN across all languages. This may suggest that increasing the model’s complexity and capacity can enhance its ability to capture intricate patterns in speech data, thereby improving transcription accuracy.

Considering the language-wise comparison, Marathi has the lowest WER across all the models, implying that these models are well-optimized or perform better with this language. However, Hindi consistently has the highest WER, indicating that these models might have difficulties handling this language or the quality of the data set for Hindi might be lower.

Table 7.3: Benchmarked comparison of MSCNN over MSCNNLSTM (where MSCNN corresponds to Multistream CNN architecture and MSCNNLSTM represents Multistream CNN-LSTM architecture (The evaluation metric considered here is word error rate (WER(%)))).

Lang	Monolingual		Multilingual	
	MSCNN	MSCNNLSTM	MSCNN	MSCNNLSTM
Hindi	17.26	16.98	17.42	16.06
Marathi	15.32	14.89	13.25	12.51
Odia	17.03	15.89	14.68	13.08
Tamil	16.4	13.67	13.59	13.01
Telugu	15.01	13.07	13.23	13.21
Gujarati	17.09	14.86	12.62	11.06

7.3.3 Experimental Results on proposed architecture

Based on our observations in Section 7.3.2.1 and Section 7.3.2.2, we concluded that the TDNNF-LSTM architecture achieves superior results in monolingual and multilingual speech recognition systems that operate with limited resources. Previous research has demonstrated that the multistream CNN architecture can enhance the robustness of recognition tasks. This provides the impetus for us to include the TDNNF-LSTM architecture in a multistream approach to capitalize on its architectural advantages.

Our study involved conducting experiments on both monolingual and multilingual ASR systems for six Indian languages. The conventional multistream CNN is denoted as MSCNN in Table 7.3 and Table 7.4, while the proposed architecture discussed in section 4 is denoted as MSCNNLSTM. In Table 7.4, the abbreviation WA refers to "with attention." A multi-head attention layer is applied in all of the architectures after concatenating all the streams and be-

fore the output layer. This enhances the architecture’s robustness by more effectively capturing learned representations.

Table 7.4: Benchmarked comparison of MSCNN_WA over MSCNNLSTM_WA (where MSCNN corresponds to Multistream CNN + attention architecture and MSCNNLSTM_WA represents Multistream CNN-LSTM + attention architecture (The evaluation metric considered here is word error rate (WER(%)))).

Lang	Monolingual		Multilingual	
	MSCNN_WA	MSCNNLSTM_WA	MSCNN_WA	MSCNNLSTM_WA
Hindi	17.03	16.07	16.85	15.71
Marathi	14.85	12.85	11.59	10.78
Odia	16.78	15.07	14.1	12.81
Tamil	15.48	12.06	11.61	10.98
Telugu	14.06	12.65	12.51	11.15
Gujarati	15.54	13.69	11.25	10.96

To evaluate the effectiveness of our proposed system, we trained it and the baselines using the same MUCS database as in our previous experiments. We quantified the systems’ performance by measuring their word error rates (WER), and the results are presented in Table 7.3 and Table 7.4, as a percentage. Our proposed system outperforms the baseline multi-stream CNN database with relative WER improvements of 1.02% and 2.05% across all languages, as shown in Table 7.3. This indicates the effectiveness of our proposed system in improving ASR accuracy.

Furthermore, our analysis revealed that the multilingual system consistently outperforms the monolingual system in most cases, even after incorporating attention, underscoring the significance of the proposed system in enhancing speech recognition performance, particularly in multilingual contexts. These findings hold important implications for advancing the field of speech recognition, particularly in applications requiring multilingual support, where the proposed system’s superior performance may offer benefits in terms of efficiency and accuracy.

7.4 Summary & Conclusion

The present study examines a TDNN-based multilingual automatic speech recognition (ASR) system for six Indian languages, specifically Hindi, Marathi, Odia, Tamil, Telugu, and Gujarati. The objective of the study is to investigate the effectiveness of the Joint Acoustic Model (JAM) for building a multilingual ASR system and to analyze the impact of different lexicon variants, like CPS and SB-CLS, AB-CLS, MB-CLS, on the system's performance.

The experimental results indicate that the multilingual models can produce results comparable to those of the monolingual models. The study further finds that the low-rank TDNN (TDNNF) architecture outperforms conventional TDNNs in most cases. Given that most Indian languages share a common phonetic space and are syllabic, the CLS approach can be extended to include more Indian languages in future research. The study also explores the TDNNF architecture by stacking more TDNNF and LSTM layers after a sequence of TDNNF layers. The TDNNF-LSTM architecture is shown to perform better in most cases. Moreover, the study proposes the use of a multi-stream CNN architecture followed by a LSTM layer and a multi-head time-restricted self-attention layer, which outperforms the baseline TDNN and multi-stream CNN architecture by 3-5% on the MUCS dataset.

The study finds that the six-language multilingual model can handle inter-sentential cases, but recognizing intra-sentential code-switching utterances requires exploring different architectures and utilizing monolingual data. In conclusion, the experimental findings from this study are valuable for researchers interested in building multilingual ASR systems for syllabic languages.

Chapter 8

Summary and Conclusions

8.1 Summary

This thesis delves into the pressing issue of resource paucity in the context of Automatic Speech Recognition (ASR) for low-resource Indian languages, with a specific focus on Telugu. It introduces the International Institute of Information Technology Hyderabad-Crowd Sourced Telugu Database (IIITH-CSTD), a large annotated speech corpus collected through a novel crowdsourcing methodology. This methodology represents a cost-effective and efficient alternative to traditional, often cumbersome, data acquisition methods.

The work extensively evaluates the effectiveness of various ASR models such as GMM-HMM, SGMM, DNN, TDNN, Transformers, and Conformers on the gathered corpus. The empirical results underscore the superior performance of the Conformer model as the training dataset size expands, illustrating the significance of large and diverse datasets. The study demonstrates that models trained on diverse datasets exhibit improved generalization capabilities, enabling them to adapt to different vocabularies, dialects, and linguistic structures.

Furthermore, the thesis explores a self-supervised approach for speech recognition tasks using the wav2vec2.0 model. This model, trained on 5000 hours of unlabeled Telugu data, provides a powerful tool to generate robust speech representations beneficial for languages with limited resources. The effectiveness of language models, dataset sizes, fine-tuning methods, and transfer learning techniques is critically examined to provide insights into enhancing the performance

of ASR models.

The study further extends its scope by examining the effectiveness of a TDNN-based multilingual ASR system for six Indian languages (Hindi, Marathi, Odia, Tamil, Telugu, and Gujarati), testing the viability of the Joint Acoustic Model (JAM) and evaluating different lexicon variants.

8.2 Contributions in the thesis

- **Crowd-sourcing Approach for Data Collection:** It introduces a novel data collection method that can address the issue of resource scarcity prevalent in under-resourced languages. The crowd-sourcing approach aids in efficiently collecting large and diverse datasets, offering a significant leap from traditional data procurement and curation methods.
- **Highlighting the Importance of Dataset Diversity:** The thesis underscores the critical role of diverse datasets in training ASR models. It demonstrates how models trained on diverse data can generalize better and adapt to a wide variety of dialects, vocabularies, and linguistic structures.
- **Ablation of different ASR Models:** A comprehensive assessment of various ASR models is conducted. When exposed to larger and diverse training datasets, the Conformer model performs remarkably well, handling complex data patterns and long-term dependencies efficiently.
- **Implementation of a Self-Supervised Approach:** The study explores and applies the wav2vec2.0 model in a self-supervised context, providing a potential strategy for overcoming challenges associated with data scarcity in low-resource languages.
- **In-depth Examination of ASR system on various scenarios:** The thesis presents a thorough evaluation of different aspects critical to the performance of ASR systems, including the influence of fine-tuning, language models, dataset sizes, and the impact of pretrained models on accuracy and precision.

- **Insight into Multilingual ASR Systems:** It investigates the Joint Acoustic Model (JAM) within a TDNN-based multilingual ASR system, offering valuable insights for researchers interested in developing ASR systems for multiple syllabic languages.

8.3 Directions for Future work

- **Application of Crowdsourcing Methodology to Other Languages:** The proven effectiveness of the crowdsourcing approach for Telugu can be extended to other low-resource languages, thereby creating an opportunity for the development of large, annotated speech corpora for a variety of languages.
- **Further Improvement of ASR Models:** With the advancements in deep learning and AI, the opportunity for the continued enhancement of ASR models is significant. Exploring models capable of handling complex data patterns and long-term dependencies more efficiently can lead to more accurate ASR systems.
- **Focus on Multilingual Models:** The study provides insights into constructing a multilingual ASR system for six Indian languages. Further research should explore code-switching at the inter-sentential and intra-sentential levels and the effectiveness of multilingual models in these contexts.
- **Enhancement of Self-Supervised Approaches:** The encouraging results using the wav2vec2.0 model signify the potential of self-supervised approaches. Further research should explore novel methodologies for self-supervised learning in ASR systems, particularly for low-resource languages.
- **Examination of Lexicon Variants in Multilingual ASR Systems:** Future work should focus on exploring the effectiveness of different lexicon variants such as CPS and SB-CLS, AB-CLS, and MB-CLS in the context of a multilingual ASR system. Such an investigation can provide insights into building more robust and adaptable ASR systems.

Publications

Related publications

Journals:

1. **Ganesh S Mirishkar**, and Anil Kumar Vuppala, “IITH-CSTD corpus: Crowd-sourced strategies for the collection of a large scale Telugu speech corpus”, IEEE ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 2023. (<https://doi.org/10.1145/3600228>)
2. **Ganesh S Mirishkar**, and Anil Kumar Vuppala, “Exploring the Shared Phonetic Space of Indian Languages for Multilingual Speech Recognition Systems”, (**under review** Speech Communication).
3. **Ganesh S Mirishkar**, and Anil Kumar Vuppala, “Towards Building ASR Systems for Low-Resource Languages of the Indian Subcontinent Using Telugu Pretrained Model”, (**under review** IEEE ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)).

Conferences & Book Chapters:

1. Shelly Jain, Aditya Yadavalli, **Ganesh Mirishkar** and Anil Kumar Vuppala,, “How Do Phonological Properties Affect Bilingual Automatic Speech Recognition?”, IEEE SLT 2022, Doha Qatar, 2022.
2. **Ganesh S Mirishkar**, “Building ASR systems for Resource-Rich and Resource-Poor languages”, 8th Doctoral Consortium, INTERSPEECH 2022, Incheon, South Korea.

3. Aditya Yadavalli*, **Ganesh S Mirishkar***, and Anil Kumar Vuppala, “Multi-Task End-to-End Model for Telugu Dialect and Speech Recognition”, INTERSPEECH 2022, Incheon, South Korea.
4. Aditya Yadavalli, **Ganesh S Mirishkar***, and Anil Kumar Vuppala, “Exploring the Effect of Dialect Mismatched Language Models in Telugu Automatic Speech Recognition,” in Proceedings NAACL:SRW,USA, 10-17th July 2022.
5. Aditya Yadavalli*, **Ganesh S Mirishkar***, Shelly Jain, and Anil Kumar Vuppala, “An Investigation of Subword-Based Bilingual Automatic Speech Recognition for Indian Languages”, IC3 2022, India.
6. Shelly Jain*, Aditya Yadavalli*, **Ganesh S Mirishkar***, Chiranjeevi Yarra, and Anil Kumar Vuppala, “IE-CPS Lexicon: An Automatic Speech Recognition Oriented Indian-English Pronunciation Dictionary”, 18th ICON, December 2021.
7. **Ganesh S Mirishkar**, Aditya Yadavalli, and Anil Kumar Vuppala, “An Investigation of Hybrid architectures for Low Resource Multilingual Speech Recognition system in Indian context”, ICON, December 2021.
8. **Ganesh S Mirishkar**, Vishnu Vidyadhara Raju V, Meher Dinesh Naroju, Sudhamay Maity, Prakash Yalla, and Anil Kumar Vuppala, “CSTD-Telugu Corpus: Crowd-Sourced Approach for Large-Scale Speech data collection”, 13th IEEE APSIPA, December 2021.
9. **Sai Ganesh Mirishkar**, Vandan Mujadia, Dipti Misra Sharma and Anil Kumar Vuppala, “INDICAP: A Multilingual Speech Translation Application”, Show and Tell paper, IEEE SLT 2021, Shenzhen, China, 2021.
10. Bipasha Sen, Aditya Agarwal, **Mirishkar Ganesh**, and Anil Kumar Vuppala, “Reed: An Approach towards quickly bootstrapping Multilingual Acoustic models”, IEEE SLT 2021, Shenzhen, China, 2021.

Other publications

Book Chapter:

1. Anil Kumar Vuppala, **Ganesh S Mirishkar**, Prakash Yalla and Vishnu Vidyadhara Raju V, "Outcomes of Speech to Speech translation for broadcast speeches and crowdsourced speech data collection pilot projects," Springer, December 2021.

Conferences:

1. Kowshik Siva Sai Motepalli, Shivang Gupta, Narasinga Vamshiraghushimha, Ravi Kumar, **Mirishkar Sai Ganesh** and Anil Kumar Vuppala, "Enhancing Language Identification in Indian Context through Exploiting Learned Features with Wav2Vec2.0", SPECOM, Hubli-IIT Dharwad, India, 2023
2. Narasinga Vamshi Raghu Simha, **Mirishkar Sai Ganesh** and Anil Kumar Vuppala, "Enhancing Stutter Detection in Speech using Zero Time Windowing Cepstral Coefficients and Phase Information", SPECOM, Hubli-IIT Dharwad, India, 2023.
3. Shaarada D. Yamini, **Ganesh S. Mirishkar**, Anil Kumar Vuppala, and Suresh Purini, "Hardware Accelerator for Transformer based End-to-End Automatic Speech Recognition System", 30th Reconfigurable Architectures Workshop (RAW), St. Petersburg, Florida USA, May 2023.
4. Nayan Vats, **Ganesh S Mirishkar**, Purva Barche, and Anil Kumar Vuppala, "Exploring High Spectro-Temporal Resolution for Alzheimer's Dementia Detection", IEEE SPCOM 2022, India.
5. Vishnu Vidyadhara Raju Vegesna, Krishna Gurugubelli, **Sai Ganesh Mirishkar** and Anil Kumar Vuppala "Towards Feature-Space Emotional Speech Adaptation for TDNN based Telugu ASR systems", Interspeech 2019.

Bibliography

- [1] S. Furui, “Generalization problem in asr acoustic model training and adaptation,” in *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, 2009, pp. 1–10.
- [2] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L.-L. Lim, *et al.*, “English conversational telephone speech recognition by humans and machines,” *arXiv preprint arXiv:1703.02136*, 2017.
- [3] H. Yadav and S. Sitaram, “A survey of multilingual models for automatic speech recognition,” *arXiv preprint arXiv:2202.12576*, 2022.
- [4] D. Galvez, G. Damos, J. Ciro, J. F. Cerón, K. Achorn, A. Gopi, D. Kanter, M. Lam, M. Mazumder, and V. J. Reddi, “The people’s speech: A large-scale diverse english speech recognition dataset for commercial usage,” *arXiv preprint arXiv:2111.09344*, 2021.
- [5] G. Chen, S. Chai, G. Wang, J. Du, W.-Q. Zhang, C. Weng, D. Su, D. Povey, J. Trmal, J. Zhang, *et al.*, “Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio,” *arXiv preprint arXiv:2106.06909*, 2021.
- [6] J. Baker, “The dragon system—an overview,” *IEEE Transactions on Acoustics, speech, and signal Processing*, vol. 23, no. 1, pp. 24–29, 1975.
- [7] D. R. Reddy, “Speech recognition by machine: A review,” *Proceedings of the IEEE*, vol. 64, no. 4, pp. 501–531, 1976.
- [8] F. Jelinek, “Continuous speech recognition by statistical methods,” *Proceedings of the IEEE*, vol. 64, no. 4, pp. 532–556, 1976.

- [9] S. Young, "A review of large-vocabulary continuous-speech," *IEEE signal processing magazine*, vol. 13, no. 5, p. 45, 1996.
- [10] M.-Y. Hwang and X. Huang, "Shared-distribution hidden markov models for speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 1, no. 4, pp. 414–420, 1993.
- [11] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.
- [12] P. Mermelstein, "Distance measures for speech recognition, psychological and instrumental," *Pattern recognition and artificial intelligence*, vol. 116, pp. 374–388, 1976.
- [13] J. A. Bilmes *et al.*, "A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *International computer science institute*, vol. 4, no. 510, p. 126, 1998.
- [14] A.-r. Mohamed, G. E. Dahl, and G. Hinton, "Acoustic modeling using deep belief networks," *IEEE transactions on audio, speech, and language processing*, vol. 20, no. 1, pp. 14–22, 2011.
- [15] L. R. B.-H. Juang, "Fundamentals of speech recognition prentice hall," *Englewood Clis*, 1993.
- [16] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on audio, speech, and language processing*, vol. 20, no. 1, pp. 30–42, 2011.
- [17] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, 2011, pp. 24–29.

- [18] A. Graves, A.-r. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks,” in *2013 IEEE international conference on acoustics, speech and signal processing*. Ieee, 2013, pp. 6645–6649.
- [19] H. Sak, A. W. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” 2014.
- [20] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, and G. Penn, “Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition,” in *2012 IEEE international conference on Acoustics, speech and signal processing (ICASSP)*. IEEE, 2012, pp. 4277–4280.
- [21] A. H. Liu, T. Tu, H.-y. Lee, and L.-s. Lee, “Towards unsupervised speech recognition and synthesis with quantized speech representation learning,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7259–7263.
- [22] A. Graves and N. Jaitly, “Towards end-to-end speech recognition with recurrent neural networks,” in *International conference on machine learning*. PMLR, 2014, pp. 1764–1772.
- [23] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, *et al.*, “Deep speech: Scaling up end-to-end speech recognition,” *arXiv preprint arXiv:1412.5567*, 2014.
- [24] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, “End-to-end continuous speech recognition using attention-based recurrent nn: First results,” *arXiv preprint arXiv:1412.1602*, 2014.
- [25] Y. Miao, M. Gowayyed, and F. Metze, “Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 167–174.

- [26] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, “End-to-end attention-based large vocabulary speech recognition,” in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 4945–4949.
- [27] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.
- [28] R. Collobert, C. Puhersch, and G. Synnaeve, “Wav2letter: an end-to-end convnet-based speech recognition system,” *arXiv preprint arXiv:1609.03193*, 2016.
- [29] H. Tang, L. Lu, L. Kong, K. Gimpel, K. Livescu, C. Dyer, N. A. Smith, and S. Renals, “End-to-end neural segmental models for speech recognition,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1254–1264, 2017.
- [30] H. Sak, M. Shannon, K. Rao, and F. Beaufays, “Recurrent neural aligner: An encoder-decoder neural network model for sequence to sequence mapping,” in *Interspeech*, vol. 8, 2017, pp. 1298–1302.
- [31] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, “Hybrid ctc/attention architecture for end-to-end speech recognition,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.
- [32] T. N. Sainath, Y. He, B. Li, A. Narayanan, R. Pang, A. Bruguier, S.-y. Chang, W. Li, R. Alvarez, Z. Chen, *et al.*, “A streaming on-device end-to-end model surpassing server-side conventional model quality and latency,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6059–6063.
- [33] J. Li, R. Zhao, Z. Meng, Y. Liu, W. Wei, S. Parthasarathy, V. Mazalov, Z. Wang, L. He, S. Zhao, *et al.*, “Developing rnn-t models surpassing high-performance hybrid models with customization capability,” *arXiv preprint arXiv:2007.15188*, 2020.

- [34] E. Riloff and J. Wiebe, “Learning extraction patterns for subjective expressions,” in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, 2003, pp. 105–112.
- [35] D. Yarowsky, “Unsupervised word sense disambiguation rivaling supervised methods,” in *33rd annual meeting of the association for computational linguistics*, 1995, pp. 189–196.
- [36] H. Scudder, “Probability of error of some adaptive pattern-recognition machines,” *IEEE Transactions on Information Theory*, vol. 11, no. 3, pp. 363–371, 1965.
- [37] J. Kahn, A. Lee, and A. Hannun, “Self-training for end-to-end speech recognition,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7084–7088.
- [38] G. Synnaeve, Q. Xu, J. Kahn, T. Likhomanenko, E. Grave, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, “End-to-end asr: from supervised to semi-supervised learning with modern architectures,” *arXiv preprint arXiv:1911.08460*, 2019.
- [39] B. Li, T. N. Sainath, R. Pang, and Z. Wu, “Semi-supervised training for end-to-end models via weak distillation,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2837–2841.
- [40] S. H. K. Parthasarathi and N. Strom, “Lessons from building acoustic models with a million hours of speech,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6670–6674.
- [41] S. Novotney and R. Schwartz, “Analysis of low-resource acoustic model self-training,” in *Tenth annual conference of the international speech communication association*, 2009.
- [42] G. Zavaliagos and T. Colthurst, “Utilizing untranscribed training data to improve performance.” in *LREC*. Citeseer, 1998, pp. 317–322.
- [43] A. v. d. Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding,” *arXiv preprint arXiv:1807.03748*, 2018.

- [44] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, “Improving language understanding by generative pre-training,” 2018.
- [45] S. Ling, Y. Liu, J. Salazar, and K. Kirchhoff, “Deep contextualized acoustic representations for semi-supervised speech recognition,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6429–6433.
- [46] A. T. Liu, S.-w. Yang, P.-H. Chi, P.-c. Hsu, and H.-y. Lee, “Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6419–6423.
- [47] W. Wang, Q. Tang, and K. Livescu, “Unsupervised pre-training of bidirectional speech encoders via masked reconstruction,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6889–6893.
- [48] S. Ling and Y. Liu, “Decoar 2.0: Deep contextualized acoustic representations with vector quantization,” *arXiv preprint arXiv:2012.06659*, 2020.
- [49] J. Bai, W. Wang, Y. Zhou, and C. Xiong, “Representation learning for sequence data with deep autoencoding predictive components,” *arXiv preprint arXiv:2010.03135*, 2020.
- [50] 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.
- [51] Q. Xu, A. Baevski, T. Likhomanenko, P. Tomasello, A. Conneau, R. Collobert, G. Synnaeve, and M. Auli, “Self-training and pre-training are complementary for speech recognition,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 3030–3034.
- [52]

- [53] A. Graves, “Sequence transduction with recurrent neural networks,” *arXiv preprint arXiv:1211.3711*, 2012.
- [54] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [55] L. Dong, S. Xu, and B. Xu, “Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition,” in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 5884–5888.
- [56] A. Zeyer, P. Bahar, K. Irie, R. Schlüter, and H. Ney, “A comparison of transformer and lstm encoder decoder models for asr,” in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 8–15.
- [57] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, *et al.*, “A comparative study on transformer vs rnn in speech applications,” in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 449–456.
- [58] J. Li, Y. Wu, Y. Gaur, C. Wang, R. Zhao, and S. Liu, “On the comparison of popular end-to-end models for large scale speech recognition,” *arXiv preprint arXiv:2005.14327*, 2020.
- [59] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, “Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7829–7833.
- [60] T. Schultz and A. Waibel, “Multilingual and crosslingual speech recognition,” in *Proc. DARPA Workshop on Broadcast News Transcription and Understanding*, 1998, pp. 259–262.
- [61] K. Veselý, M. Karafiát, F. Grézl, M. Janda, and E. Egorova, “The language-independent bottleneck features,” in *2012 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2012, pp. 336–341.

- [62] S. Thomas, M. L. Seltzer, K. Church, and H. Hermansky, “Deep neural network features and semi-supervised training for low resource speech recognition,” in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6704–6708.
- [63] F. Grézl, M. Karafiát, and K. Veselý, “Adaptation of multilingual stacked bottle-neck neural network structure for new language,” in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 7654–7658.
- [64] A. Baevski, W.-N. Hsu, A. Conneau, and M. Auli, “Unsupervised speech recognition,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 826–27 839, 2021.
- [65] A. Baevski, S. Schneider, and M. Auli, “vq-wav2vec: Self-supervised learning of discrete speech representations,” *arXiv preprint arXiv:1910.05453*, 2019.
- [66] R. J. Weiss, J. Chorowski, N. Jaitly, Y. Wu, and Z. Chen, “Sequence-to-sequence models can directly translate foreign speech,” *arXiv preprint arXiv:1703.08581*, 2017.
- [67] Z.-Q. Zhang, Y. Song, M.-H. Wu, X. Fang, and L.-R. Dai, “Xlst: Cross-lingual self-training to learn multilingual representation for low resource speech recognition,” *arXiv preprint arXiv:2103.08207*, 2021.
- [68] B. Li, Y. Zhang, T. Sainath, Y. Wu, and W. Chan, “Bytes are all you need: End-to-end multilingual speech recognition and synthesis with bytes,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5621–5625.
- [69] P. Swietojanski, A. Ghoshal, and S. Renals, “Hybrid acoustic models for distant and multichannel large vocabulary speech recognition,” in *2013 IEEE workshop on automatic speech recognition and understanding*. IEEE, 2013, pp. 285–290.
- [70] S. Dalmia, R. Sanabria, F. Metze, and A. W. Black, “Sequence-based multi-lingual low resource speech recognition,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4909–4913.

- [71] E. Yılmaz, H. van den Heuvel, and D. Van Leeuwen, “Code-switching detection using multilingual dnns,” in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 610–616.
- [72] S. Liang and W. Yan, “Multilingual speech recognition based on the end-to-end framework,” *Multimedia Tools and Applications*, 2022.
- [73] L. Lu, A. Ghoshal, and S. Renals, “Acoustic data-driven pronunciation lexicon for large vocabulary speech recognition,” in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 2013, pp. 374–379.
- [74] D. B. Paul and J. Baker, “The design for the wall street journal-based csr corpus,” in *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*, 1992.
- [75] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “Librispeech: an asr corpus based on public domain audio books,” in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.
- [76] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, “Darpa timit acoustic-phonetic continuous speech corpus cd-rom. nist speech disc 1-1.1,” *NASA STI/Recon technical report n*, vol. 93, p. 27403, 1993.
- [77] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, “Common voice: A massively-multilingual speech corpus,” *arXiv preprint arXiv:1912.06670*, 2019.
- [78] J. J. Godfrey, E. C. Holliman, and J. McDaniel, “Switchboard: Telephone speech corpus for research and development,” in *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, vol. 1. IEEE Computer Society, 1992, pp. 517–520.
- [79] B. M. L. Srivastava, S. Sitaram, R. K. Mehta, K. D. Mohan, P. Matani, S. Satpal, K. Bali, R. Srikanth, and N. Nayak, “Interspeech 2018 low resource automatic speech recognition challenge for indian languages.” in *SLTU*, 2018, pp. 11–14.

- [80] O. Kjartansson, S. Sarin, K. Pipatsrisawat, M. Jansche, and L. Ha, “Crowd-sourced speech corpora for javanese, sundanese, sinhala, nepali, and bangladeshi bengali,” 2018.
- [81] A. Bhanushali, G. Bridgman, G. Deekshitha, P. Ghosh, P. Kumar, S. Kumar, A.-R. Kolladath, N. Ravi, A. Seth, A. Singh, *et al.*, “Gram vaani asr challenge on spontaneous telephone speech recordings in regional variations of hindi,” in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2022. International Speech Communication Association, 2022, pp. 3548–3552.
- [82] A. Diwan, R. Vaideeswaran, S. Shah, A. Singh, S. Raghavan, S. Khare, V. Unni, S. Vyas, A. Rajpuria, C. Yarra, A. Mittal, P. K. Ghosh, P. Jyothi, K. Bali, V. Seshadri, S. Sitaram, S. Bharadwaj, J. Nanavati, R. Nanavati, K. Sankaranarayanan, T. Seeram, and B. Abraham, “Multilingual and code-switching asr challenges for low resource indian languages,” *Proceedings of Interspeech*, 2021.
- [83] M. A. B. Pilar, and R. A. G., “Subword dictionary learning and segmentation techniques for automatic speech recognition in tamil and kannada,” 2022. [Online]. Available: <https://arxiv.org/abs/2207.13331>
- [84] ———, “Knowledge-driven subword grammar modeling for automatic speech recognition in tamil and kannada,” 2022. [Online]. Available: <https://arxiv.org/abs/2207.13333>
- [85] W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig, “Achieving human parity in conversational speech recognition,” *arXiv preprint arXiv:1610.05256*, 2016.
- [86] J. C. Félix-Brasdefer, “Data collection methods in speech act performance,” *Speech act performance: Theoretical, empirical and methodological issues*, vol. 26, no. 41, pp. 69–82, 2010.
- [87] J. Freitas, A. Calado, D. Braga, P. Silva, and M. Dias, “Crowdsourcing platform for large-scale speech data collection,” *Proc. Fala*, 2010.
- [88] A. Butryna, S.-H. C. Chu, I. Demirsahin, A. Gutkin, L. Ha, F. He, M. Jansche, C. Johny, A. Katanova, O. Kjartansson, *et al.*, “Google crowdsourced speech corpora and related

- open-source resources for low-resource languages and dialects: an overview,” *arXiv preprint arXiv:2010.06778*, 2020.
- [89] K. Prasad, S. Virk, M. Nishioka, and C. Kaushik, “Crowd-sourced technical texts can help revitalise indian languages,” in *Proceedings Of LREC*, 2018, pp. 11–16.
- [90] P. Jonell, C. Oertel, D. Kontogiorgos, J. Beskow, and J. Gustafson, “Crowdsourced multi-modal corpora collection tool,” in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.
- [91] S. Arora, K. K. Arora, M. K. Roy, S. S. Agrawal, and B. Murthy, “Collaborative speech data acquisition for under resourced languages through crowdsourcing,” *Procedia Computer Science*, vol. 81, pp. 37–44, 2016.
- [92] B. Abraham, D. Goel, D. Siddarth, K. Bali, M. Chopra, M. Choudhury, P. Joshi, P. Jyoti, S. Sitaram, and V. Seshadri, “Crowdsourcing speech data for low-resource languages from low-income workers,” in *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 2819–2826.
- [93] P. Jyothi and M. Hasegawa-Johnson, “Acquiring speech transcriptions using mismatched crowdsourcing,” in *Proceedings of the AAAI Conference On Artificial Intelligence*, vol. 29, no. 1, 2015.
- [94] H. K. Vydana, K. Gurugubelli, V. V. R. Vegesna, and A. K. Vuppala, “An exploration towards joint acoustic modeling for indian languages: Iit-h submission for low resource speech recognition challenge for indian languages, interspeech 2018.” in *INTERSPEECH*, 2018, pp. 3192–3196.
- [95] J. Billa, “Isi asr system for the low resource speech recognition challenge for indian languages.” in *INTERSPEECH*, 2018, pp. 3207–3211.
- [96] V. M. Shetty and S. Umesh, “Exploring the use of common label set to improve speech recognition of low resource indian languages,” in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 7228–7232.

- [97] M. G. Kumar, J. Kuriakose, A. Thyagachandran, A. Seth, L. D. Prasad, S. Jaiswal, A. Prakash, H. Murthy, *et al.*, “Dual script e2e framework for multilingual and code-switching asr,” *arXiv preprint arXiv:2106.01400*, 2021.
- [98] T. Javed, S. Doddapaneni, A. Raman, K. Bhogale, G. Ramesh, A. Kunchukuttan, P. Kumar, and M. M. Khapra, “Towards building asr systems for the next billion users,” *ArXiv*, vol. abs/2111.03945, 2021.
- [99] C. Kim and R. M. Stern, “Robust signal-to-noise ratio estimation based on waveform amplitude distribution analysis,” in *Ninth Annual Conference of the International Speech Communication Association*, 2008.
- [100] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, *et al.*, “Espnet: End-to-end speech processing toolkit,” *arXiv preprint arXiv:1804.00015*, 2018.
- [101] 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 *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.
- [102] T. Holter, J. Epps, A. Gopalakrishnan, E. Choi, O. Kenny, D. Grayden, A. Burkitt, B. Kraal, M. Wagner, P. Collings, *et al.*, “Automatic speech recognition,” 1965.
- [103] X. Huang, J. Baker, and R. Reddy, “A historical perspective of speech recognition,” *Communications of the ACM*, vol. 57, no. 1, pp. 94–103, 2014.
- [104] M. Karafiát, L. Burget, P. Matějka, O. Glembek, and J. Černocký, “ivector-based discriminative adaptation for automatic speech recognition,” in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, 2011, pp. 152–157.
- [105] V. Peddinti, G. Chen, V. Manohar, T. Ko, D. Povey, and S. Khudanpur, “Jhu aspire system: Robust lvcsr with tdnns, ivector adaptation and rnn-lms,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 539–546.

- [106] G. S. Mirishkar, V. V. R. V, M. D. Naroju, S. Maity, P. Yalla, and A. K. Vuppala, “Cstd-telugu corpus: Crowd-sourced approach for large-scale speech data collection,” in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2021, pp. 511–517.
- [107] S. Madikeri, S. Dey, P. Motlicek, and M. Ferras, “Implementation of the standard i-vector system for the kaldı speech recognition toolkit,” *Idiap*, Tech. Rep., 2016.
- [108] R. Haeb-Umbach and H. Ney, “Linear discriminant analysis for improved large vocabulary continuous speech recognition.” in *icassp*, vol. 92. Citeseer, 1992, pp. 13–16.
- [109] A. Acero and R. M. Stern, “Environmental robustness in automatic speech recognition,” in *International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 1990, pp. 849–852.
- [110] M. Fujimoto and Y. Riki, “Robust speech recognition in additive and channel noise environments using gmm and em algorithm,” in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1. IEEE, 2004, pp. I–941.
- [111] A. Waibel and K.-F. Lee, *Readings in speech recognition*. Morgan Kaufmann, 1990.
- [112] V. Peddinti, D. Povey, and S. Khudanpur, “A time delay neural network architecture for efficient modeling of long temporal contexts,” in *Sixteenth annual conference of the international speech communication association*, 2015.
- [113] 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*, vol. 33, pp. 12 449–12 460, 2020.
- [114] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, “fairseq: A fast, extensible toolkit for sequence modeling,” *arXiv preprint arXiv:1904.01038*, 2019.
- [115] G. Mirishkar, A. Yadavalli, and A. K. Vuppala, “An investigation of hybrid architectures for low resource multilingual speech recognition system in indian context,” in *Pro-*

- ceedings of the 18th International Conference on Natural Language Processing (ICON)*, 2021, pp. 205–212.
- [116] P. Swietojanski, A. Ghoshal, and S. Renals, “Unsupervised cross-lingual knowledge transfer in dnn-based lvcsr,” in *2012 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2012, pp. 246–251.
- [117] A. Imankulova, R. Dabre, A. Fujita, and K. Imamura, “Exploiting out-of-domain parallel data through multilingual transfer learning for low-resource neural machine translation,” *arXiv preprint arXiv:1907.03060*, 2019.
- [118] J. Cho, M. K. Baskar, R. Li, M. Wiesner, S. H. Mallidi, N. Yalta, M. Karafiat, S. Watanabe, and T. Hori, “Multilingual sequence-to-sequence speech recognition: architecture, transfer learning, and language modeling,” in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 521–527.
- [119] A. Das and M. Hasegawa-Johnson, “Cross-lingual transfer learning during supervised training in low resource scenarios,” in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.
- [120] C. Liu, Q. Zhang, X. Zhang, K. Singh, Y. Saraf, and G. Zweig, “Multilingual graphemic hybrid asr with massive data augmentation,” *arXiv preprint arXiv:1909.06522*, 2019.
- [121] S. Thomas, K. Audhkhasi, and B. Kingsbury, “Transliteration based data augmentation for training multilingual asr acoustic models in low resource settings,” *Proc. Interspeech 2020*, pp. 4736–4740, 2020.
- [122] J. Cui, B. Kingsbury, B. Ramabhadran, A. Sethy, K. Audhkhasi, X. Cui, E. Kislal, L. Mangu, M. Nussbaum-Thom, M. Picheny, *et al.*, “Multilingual representations for low resource speech recognition and keyword search,” in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 259–266.
- [123] H. Xu, V. H. Do, X. Xiao, and E. S. Chng, “A comparative study of bnf and dnn multilingual training on cross-lingual low-resource speech recognition,” in *Sixteenth annual conference of the international speech communication association*, 2015.

- [124] A. Rahimi, Y. Li, and T. Cohn, “Multilingual ner transfer for low-resource languages,” 2019.
- [125] V. M. Shetty and S. Umesh, “Exploring the use of Common Label Set to Improve Speech Recognition of Low Resource Indian Languages,” pp. 7228–7232, 2021.
- [126] K. Prahallad, E. N. Kumar, V. Keri, S. Rajendran, and A. W. Black, “The iit-h indic speech databases,” in *Thirteenth annual conference of the international speech communication association*, 2012.
- [127] A. Baby, N. Nishanthi, A. L. Thomas, and H. A. Murthy, “A unified parser for developing indian language text to speech synthesizers,” in *International Conference on Text, Speech, and Dialogue*. Springer, 2016, pp. 514–521.
- [128] B. Ramani, S. L. Christina, G. A. Rachel, V. S. Solomi, M. K. Nandwana, A. Prakash, S. A. Shanmugam, R. Krishnan, S. K. Prahallad, K. Samudravijaya, *et al.*, “A common attribute based unified hts framework for speech synthesis in indian languages,” in *Eighth ISCA Workshop on Speech Synthesis*, 2013.
- [129] A. Pandey, B. M. L. Srivastava, and S. V. Gangashetty, “Adapting monolingual resources for code-mixed hindi-english speech recognition,” in *2017 International Conference on Asian Language Processing (IALP)*. IEEE, 2017, pp. 218–221.
- [130] S. H. Mallidi and H. Hermansky, “Novel neural network based fusion for multistream asr,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5680–5684.