Noise Robust Speech Recognition for Search and Rescue Domain

by

Sangeet Sagar

Master Thesis

Faculty of Philosophy
Department of Language Science and Technology
Saarland University
Multilinguality and language technology, DFKI

Supervisor
Prof. Dr. Josef van Genabith (UdS, DFKI),
Dipl. Inf. Bernd Kiefer (DFKI)

Advisor
Prof. Dr. Mirco Ravanelli
(Concordia University, Mila-Quebec AI Institute)

Project Domain Consultant
Dr. Ing. Ivana Kruijff-Korbayová (UdS, DFKI)

April 18, 2023

UNIVERSITÄT
DES
SAARLANDES

Declaration of Authorship

Einverständniserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Ich versichere, dass die gedruckte und die elektronische Version der Masterarbeit inhaltlich übereinstimmen.

Declaration of Consent

I hereby confirm that the thesis presented here is my own work, with all assistance acknowledged. I assure that the electronic version is identical in content to the printed version of the Master's thesis.

Ort/Place: Saarbrücken (Saarland)

Datum/Date:

Unterschrift/Signature:

SAARLAND UNIVERSITY

Abstract

Noise Robust Speech Recognition for Search and Rescue Domain

by Sangeet Sagar

Despite recent advancements in speech recognition, there are still difficulties in accurately transcribing conversational and emotional speech in noisy and reverberant acoustic environments. This poses a particular challenge in the Search And Rescue domain, where transcribing conversations among rescue team members is crucial to support real-time decision-making. The scarcity of speech data and associated background noise in SAR scenarios make it difficult to deploy robust speech recognition systems.

This work extends the task of noise robustness in speech recognition to SAR missions. To address this issue, we have created and made publicly available a German speech dataset called *RescueSpeech*. This dataset includes real speech recordings from simulated rescue exercises. Additionally, we have released competitive training recipes and pre-trained models. Our study indicates that the current level of performance achieved by state-of-the-art methods is still far from being acceptable. We hope that the release of RescueSpeech will bring attention to the challenges of speech recognition in SAR scenarios and encourage further research in this field.

Acknowledgements

I am grateful to the Multilinguality and Language Technology Lab at DFKI GmbH Saarbrücken for providing me with financial support throughout the duration of this thesis. My supervisors and advisors have been instrumental in guiding me through the process, and I extend my sincere thanks to them. I am also deeply indebted to the Language Science and Technology department and Ms Christina Deeg for their unwavering support.

I would like to express my gratitude to everyone who contributed to reviewing this work and provided valuable feedback. Their insights and suggestions have greatly enriched my research. In particular, I would like to thank Rishu Kumar (UdS) for his insightful comments and feedback.

Lastly, I would like to extend a special thanks to my friend and colleague Alina Leippert from the A-DRZ project for her assistance in transcribing a portion of the RescueSpeech dataset. Her contributions were invaluable in completing this work, and I deeply appreciate her support.

This work was supported under the project "A-DRZ: Setting up the German Rescue Robotics Center" and funded by the German Ministry of Education and Research (BMBF), grant No. I3N14856.

Content

Declaration of Authorship i

Abstract ii

Acknowledgements iii

1 Introduction 1
- 1.1 Motivation . . . 2
- 1.2 Contribution . . . 4

2 Literature Survey 6
- 2.1 Automatic Speech Recognition . . . 6
- 2.2 Speech Enhancement . . . 9
- 2.3 SpeechBrain . . . 10

3 Technical Background 12
- 3.1 End2End Automatic Speech Recognition . . . 12
  - 3.1.1 End2end Models . . . 12
  - 3.1.2 End2end Architectures . . . 16
  - 3.1.3 Evaluation metrics . . . 20
- 3.2 Single Channel Speech Enhancement . . . 21
  - 3.2.1 *SepFormer*: Transformer-based neural network . . . 21
  - 3.2.2 Evaluation metrics . . . 23

4 Dataset Description 26
- 4.1 Related Corpora . . . 26
- 4.2 General Training Data . . . 26
- 4.3 The RescueSpeech Dataset . . . 27

5 Experimental Setup 30
- 5.1 ASR training . . . 30

5.2 Speech enhancement training 32
5.3 Training strategies 32

6 Results and Discussions **35**
6.1 Pre-training Performance 35
6.2 ASR Performance 36
6.2.1 Combining ASR and Speech Enhancement 39

7 Conclusion **42**
7.1 General findings 42
7.2 Future work 42

List of Figures **44**

List of Tables **46**

Bibliography **47**

55

Chapter 1

Introduction

From keys to touch and touch to voice, technology has grown to make computing more seamless, natural, and simple. Voice assistants, the latest in the technology to hit the consumer market, have taken the interface out of communicating with a machine. Automatic speech recognition (ASR) enables these devices to accurately translate spoken utterances into text. ASR has made communication amongst the diverse population effortless, lucid and today, it has found a wide range of applications in the scientific domain (e.g. voice-based translation software), the commercial world (e.g. live subtitling of conferences), and healthcare (e.g. efficient and expedited diagnosis, voice-assisted appointment booking). It should not come as a surprise that for such systems to reach a human-level accuracy, the amount of speech data needed is significantly large which involves a labour-intensive process of data collection and further manually transcribing them with the help of language experts. Predominant languages like English, German, Spanish, Chinese, etc, have several publicly available speech datasets like CommonVoice [1], LibriSpeech [2], VoxCeleb [3] etc, but for low-resource languages like Kinyarwanda, Zulu, Swahili the available dataset for research is pretty scarce. This poses a problem in training speech recognizers dedicated to such languages. Transcribing speech is challenging when there is a domain mismatch i.e. a model trained on audio collected from public speeches is used to transcribe university mathematics lectures. Such a system is likely to underperform compared to a model trained directly on classroom lectures. A few other obstacles include differences in dialect and accent among speakers. The resulting performance of the ASR model trained on dialogues of native speakers can substantially deteriorate when tested against non-native speakers [4].

Among these challenges, ASR is also known to perform poorly in noisy surroundings since the audio quality is low and is corrupted with unwanted noises like street noise, keyboard noise, crowd noise, babble noise, vehicle noise, etc. This is bound to happen because the training set generally does not constitute any noisy speech data and the model fails to generalize over noisy speech. Type and level of noise are the two main factors that affect ASR performance in noisy conditions. For E.g. transcription of audio recorded in a lunch cafeteria is likely to perform

better than audio recorded in presence of engine noise and emergency vehicle siren. Therefore, a system that is robust to adapt to different noisy conditions as well as clean environments– a noise-robust ASR can be trained to withstand such noisy scenarios. A system may be made to learn to adapt to noises or use a noise-eliminating component to get rid of the noise before the speech recognizer acts on it.

In this work, we focus on speech recognition for the German language that is robust to noises with a special focus on the search and rescue (SAR) domain. Noise in the SAR domain encompasses fire engine noise, vehicle sirens, static radio noise, chopper noise, and heavy breathing during speech. These scenarios often involve making critical decisions in extremely hostile conditions, such as underground rescue operations, nuclear accidents, fire evacuation or collapsed buildings after an earthquake. In such cases, rescue workers must act quickly and accurately to prevent the loss of lives and damage. Transcribing and automatically analyzing the conversations within the rescue team can provide useful support to help the team make the right decisions in a limited amount of time.

1.1 Motivation

This research work is dedicated to the "A-DRZ: Setting up the German Rescue Robotics Center" project. This mega-project aims to provide an efficient response and swift communication in a real disaster scenario by deploying robots enabled with situational understanding.

Rescue response to an emergency involving high-risk scenarios like widespread fire, terrorist bombings, regime change, earthquake, nuclear accidents, etc, often exceeds human capacity to mitigate the damage. It involves making critical decisions in extremely hostile conditions and executing actions with limited resources. Mobile robots and drones are frequently used to assess the accident site and get access to isolated or cut-off locations of the site. This can help with significant damage control and get the rescue team into providing a quicker and better-planned response. It is necessary that the robotic system is aware of the goal of the rescue mission and well adapted to a disaster scenario. So to keep it simple, the idea is to power these robots with spoken language understanding (SLU), allowing them to acquire situational knowledge using the verbal communication carried out among the rescue team.

The current system has been illustrated in Fig: 1.1, but we shall focus primarily on the *speech processing* component as shown in Figure 1.2. All communications among the first-responder team members are captured and transmitted to the speech-processing component. It has three modules– ASR, natural language understanding (NLU), and a post-processing module. Input audio is transcribed using the ASR system, and the text transcriptions are further semantically interpreted using the NLU. It extracts meaning from the text and maintains state of the dialogue.

The post-processing module takes in the NLU hypothesis and some meta-data (like speaker and addressee information), re-evaluates each hypothesis and reranks them.

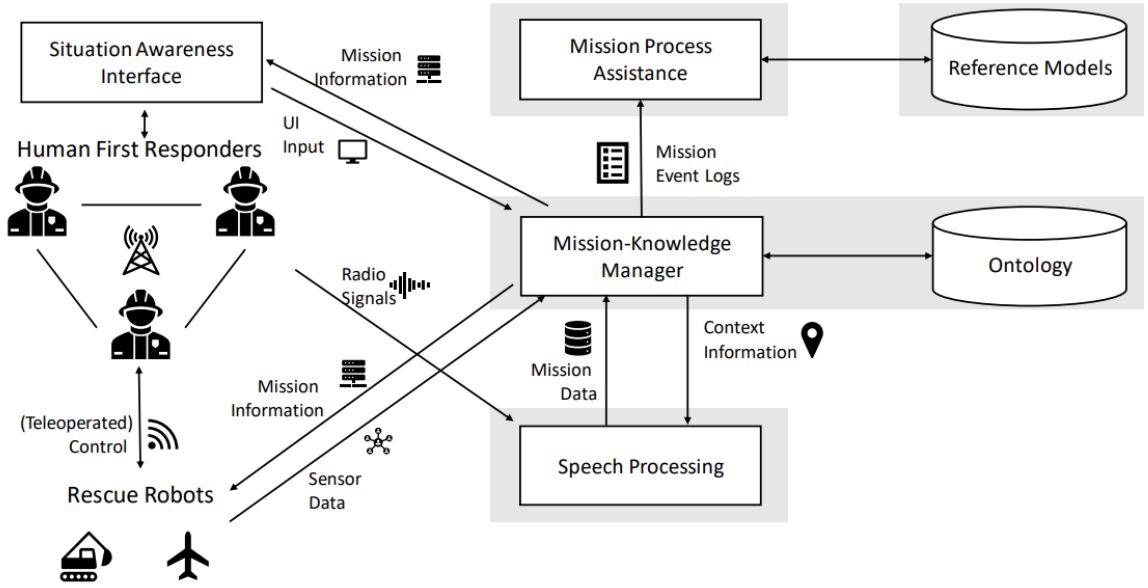

FIGURE 1.1: A-DRZ complete system architecture [5]

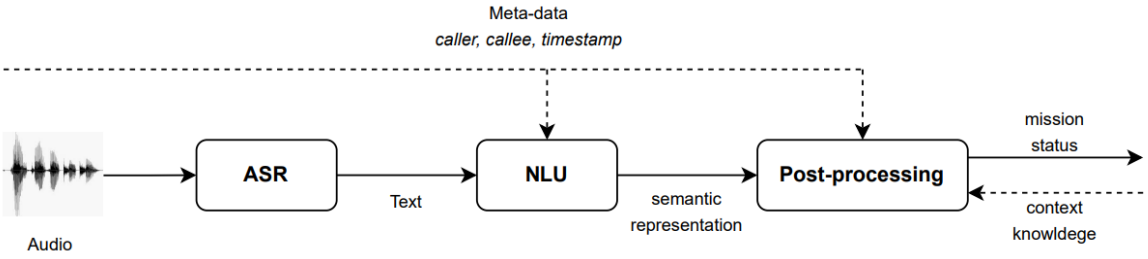

FIGURE 1.2: A-DRZ: speech processing component [6]

ASR in the speech processing component currently relies on a cloud-based commercial service– Cerence Mix ASR, and a locally running ASR– Mozilla Deepspeech [7]. Although commercial ASR services are better in terms of performance, it is not a reliable resort as it needs an internet connection to run its services. Such an internet connection is often absent in an emergency. Also, these commercial or locally installed ASR systems perform poorly with corrupted speech signals or in noise-filled surroundings. This work aims to fill up this void for an noise-robust ASR, specially curated for hostile conditions like the SAR domain, and comparable in performance with commercially available online services.

1.2 Contribution

In this work, we propose the task of noise-robust German speech recognition for SAR domain. It involves developing and training systems capable of performing efficiently during rescue operations in SAR noise-contaminated surroundings. The goal is to enhance communication performance between rescuers with voice-enabled rescue robots. The challenges we address are–

- *Lack of speech data in SAR domain*: Speech data pertaining specifically to the SAR domain is hard to collect and is often barricaded with privacy restrictions, thus limiting their availability to the scientific community. This leaves us with a scarce amount of data, but to train a speech recognizer for SAR application would inarguably require a plethora of such data to reach human-level accuracy. Our work overcomes this challenge by training a speech recognizer on a large amount of openly-available clean speech data (see Section: 4.2), and further fine-tuning on a little in-domain data.
- *Robustness to SAR noises*: To create a speech recognizer that is robust to noise specifically the ones common in the SAR domain like engine noise, vehicle sirens, radio noise, etc would need an ASR adapt to these noises and perform transcription. However, this poses a challenge to the existing ASR frameworks as they are not suited to work in such extreme conditions. We propose multiple approaches to solve this obstacle– (i) perform a multi-condition training (where a system is trained on a uniform mixture of clean and noisy speech signals) (ii) integrate a speech enhancement module (to eliminate the noises of the speech signals before feeding to ASR) with the ASR module (see detailed explanation in Section: 5.3).

To encourage research and development in this field, we have released RescueSpeech, a German dataset for the Search and Rescue Domain Speech. This dataset contains authentic speech recordings between members of a rescue team during several rescue exercises. To the best of our knowledge, we are the first to publicly release an audio dataset in the SAR domain. RescueSpeech contains approximately 2 hours of annotated speech material. Although this amount may seem limited, it is actually quite valuable and can be effectively used to fine-tune large pre-trained models such as wav2vec2.0 [8], WavLM [9], and Whisper [10]. In fact, we demonstrate that this material is also suitable for training models from scratch when combined with proper data augmentation techniques and multi-condition training.

This paper presents a comprehensive collection of experimental evidence for the task at hand– noise-robust German speech recognition. It employs state-of-the-art methods for both speech recognition and speech enhancement, as well as a combination of the two. Despite excelling in simpler scenarios, our results show that even modern ASR systems like Whisper [10], struggle to perform well in the demanding rescue and search domain. We have made our dataset, training

recipes, pre-trained models and training logs available to the community [1]. Additionally, a demo of this work can be found online[2]. With the release of the RescueSpeech dataset, we hope to foster research in this field and establish a common benchmark. We believe that our effort can help raise awareness about the importance of the use of speech technology in SAR missions, and the need for continued research in this domain.

For the desired task the ASR module needs to be online i.e. speech recognizer running locally and transcribing audio in real-time, but for simplicity, we shall focus on building an offline-ASR system.

Thesis structure This thesis is distributed as follows: we perform a detailed literature survey in Chapter 2 where we present how speech recognition and enhancement evolved over the last 20 years as well as an overview of the SpeechBrain toolkit. This is followed by an in-depth discussion on technical background in Chapter 3. Here we dive deep into the concepts driving this thesis. In Chapter 4 we discuss related corpora in this field and training data used to perform experiments. We also describe details on the RescueSpeech dataset. Further, in Chapter 5 we present our complete experiment protocol and training strategies. Next, we present our results and back them with proper analysis and reasoning in Chapter 6. Finally, we conclude our thesis work with closing remarks and suggestions for future work in Chapter 7.

[1] GitHub repository: https://github.com/sangeet2020/speechbrain/tree/develop/recipes/RescueSpeech
[2] Project demo: https://sangeet2020.github.io/

Chapter 2

Literature Survey

In this chapter, we perform a systematic literature review on speech recognition and speech enhancement that will be referenced throughout this thesis. We study classical as well as modern approaches to understand how these technologies have evolved in the past two decades. Based on the title- **Noise Robust Speech Recognition for Search and Rescue Domain**, we break down our survey into two parts- automatic speech recognition and speech enhancement and further explore how their combination has contributed towards a noise-filled robust speech recognition. We believe that these two subjects are an extensive topic of research in itself and are therefore beyond the scope of this thesis to review and analyse all relevant topics. We also take some insights into the SpeechBrain toolkit that has been used to carry out all experiments in this thesis.

2.1 Automatic Speech Recognition

The history of ASR can be dated back to 1952 when Bell Laboratories at IBM created the first ever speech recognition system "Audrey" designed to identify numbers. With a very limited vocabulary in the '50s to a vocabulary ranging to thousands in the '80s, speech recognition saw significant progress using Hidden Markov Models (HMM) [11]. HMMs together with the Baum–Welch algorithm (for parameter estimation) provided the first statistical modelling approach for speech signals and since then it became a popular paradigm to model the probability of sounds being actual words. Although artificial neural networks (ANN) were introduced in the 1940s, it was not until the 1990s, they rose to popularity amongst the speech-processing community [12]. Multi-layered neural networks were used to discriminate among the limited vocabulary of words and tested against speaker-dependent and speaker-independent setups [13]. Another approach for HMM-based speech recognition that gained popularity in the 20s made use of Gaussian mixture models (GMM). In this setup, the state outputs of HMM (or emission probability) are modelled as mixture models by adding discrete latent variables [14], where each speech unit is represented as a GMM distribution in an HMM state [15]. Decoding is performed

using the Viterbi algorithm [16] to find the best sequence of most likely states that generate the given observation. Post-2012 was the era of modern ASR systems that utilized a hybrid approach i.e. combination of HMMs and deep neural networks (DNN). It excelled performance of then most existing systems in terms of accuracy and latency. Here we review this approach thoroughly.

DNN-HMM: a hybrid approach for speech recognition A hybrid ASR system constitutes of HMM and DNN used in combination to perform speech recognition. Unlike a GMM-HMM framework for speech recognition that uses GMMs to estimate observation probabilities of HMM states, DNN-HMM employs DNN to estimate state observation probabilities. It uses multi-layered neural networks to develop strong and complex feature learning ability for acoustic frames [17] and HMM models the sequential time-varying property of speech signals. Amongst these, DNN-HMM based systems have the advantage of efficient decoding using the Viterbi algorithm. Currently, most hybrid ASR systems use ANNs as a label classifier- labelling each speech frame to phoneme. However, DNNs need a large amount of training data means more demanding computing resources. It can have a highly non-convex objective function leading to a suboptimal local minima [18]. [19] shows a word error rate (WER) of 18.5 on SWITCHBOARD (test set 1) using the DNN-HMM approach compared to traditional GMM-HMM with a WER of 27.4%. Another work by [18] used DNN-HMM for speech emotion recognition on eNTERFACE'05 dataset, showed an improvement in emotion recognition accuracy from 42.22% (GMM-HMM) to 53.89% using 6 hidden layers and following discriminative training approach.

Recently end-to-end (E2E) based modeling approaches have been increasingly gaining traction among speech data miners. A non-modularized ASR system comprises of acoustic model, pronunciation model, and language model (LM) and they have to be trained independently each with a different objective function. Presumably, a global optimum is not guaranteed, and an error in one component may not behave well with errors in another component, and this leads to unsatisfactory performance. This motivated researchers to come up with a way that replaces all these components by training a single model. Hence the name- E2E i.e. map a sequence of small acoustic frames directly into a sequence of smallest linguistic units like phonemes. E2E models have made key contributions to ASR and the credit goes to not just one but a family of different approaches (all DNN based). The foundation of our work is based on E2E, and we use this as a base reference throughout the thesis. Here we review this approach briefly while a more detailed study has been conducted in Chapter 3.

E2E approach for speech recognition The most commonly used approaches for E2E ASR are- (i) Connectionist temporal classification (CTC), (ii) Attention-based Encoder-Decoder models, and (iii) Recurrent neural network (RNN)-Transducer. CTC [20] although introduced

in 2006, became a widely used algorithm for ASR after 2016. In a traditional ASR system, fixed alignments are obtained using the forced alignments of acoustic frames with the phones. However, such alignments are absent in E2E systems. CTC addresses this issue by modelling the probability distribution at each time step over all possible phones including a blank label [21]. [22] proposed a system with a CTC-based scoring function to perform a character-level ASR. The system is based on bidirectional LSTM-RNN and trained on a full 81 hrs of Wall Street Journal (WSJ) corpus. Later work focused on training on a significantly larger dataset and incorporating pre-trained language models during decoding and enabling distributed training across multi-GPUs for scaled-up performance [23]. Another work by [24] adopted the same architecture- bidirectional LSTM-RNN network and CTC loss, but trained on about 125K hours of audio data on word level to achieve similar WER as the existing system without using an LM. Attention-based Encoder-Decoder models [25], [26] have become a prominent and most widely used approach in ASR since 2015 (also known as Listen, Attend and Spell (LAS) models). It consists of an encoder (analogous to the acoustic model), an attention mechanism, and a decoder. The encoder computes high-level features from the input acoustic frames, and the attention module (analogous to the alignment model in HMM) computes attention weights to form a context vector (to identify the encoded frames that are relevant to the current output). The decoder (analogous to the pronunciation model and LM) further takes in the context vector and its last output to predict each output label as a function of the previous label. Further study by [27] shows that attention-based models perform closely to other E2E models on dictation tasks and slightly outperform the baseline on the Google VoiceSearch test set. RNN-transducer (RNN-T) [28], is yet another E2E model for ASR proposed in 2012. It consists of an encoder, a prediction network, a joint network, and a softmax. Analogous to an acoustic model in a non-modularized ASR paradigm, the encoder performs a high-level feature representation, and the prediction network takes previous predictions and generates text embeddings. The joint model further combines encoder output and text embeddings and the combined output is followed by a softmax layer. At each time step, the model either predicts a sub-word unit or a blank. Later work [29] focuses on further model optimization in terms of performance and faster training. Another work by [30] jointly models End-Of-Utterance (EOU) with ASR in RNN-T for better latency. These works show that RNN-transducer models are better than CTC or attention-based models but still lack popularity. RNN transducers suffer from high-memory requirements to compute the posteriors of a grid of alignments composed by the encoder and prediction network. Recent works have focused more on optimizing these memory issues with more advanced network structures.

2.2 Speech Enhancement

In real-world conditions, audio data acquisition often involves speech signals getting corrupted due to noises around them which degrade the audio quality and make it difficult for a listener to comprehend its meaning. With speech enhancement, we aim to enhance the quality of spoken audio and extract a cleaner speech signal from a noisy mixture. The enhanced speech signal can be used for further downstream tasks, e.g. ASR. One of the classical approaches for speech enhancement is the classical method that operates in the frequency domain. The signal is passed through a short-time Fourier transform (STFT) to obtain short-time spectral features and phase. These spectral features are used by noise estimators and gain estimators to estimate the magnitude of the noise spectrum. The estimated noisy spectrum is further convoluted with the noisy spectral features to extract clean speech from the noisy speech. The obtained clean spectral features go through inverse- STFT to obtain the signal in the time domain. Below we review the most well-known approaches for speech enhancement– traditional approaches and recent SOTA deep-learning-based methods.

Classical approaches : One of the well-known algorithms for speech enhancement is spectral subtraction [31] which estimates the noise power spectrum by averaging over the input noise spectra from several frames like a moving average filter. Once the noise spectrum is obtained, it is subtracted from the noisy speech spectrum to retrieve the clean signal. The most addressed shortcoming of spectral subtraction is signal distortion i.e. if too much is subtracted, the speech signal is lost and if too less is subtracted, the speech signal is left noisy. Wiener filter [32] is another traditional alternative to spectral subtraction. The fundamental idea of Wiener filtering is that it minimizes the mean-squared error between the reconstructed spectrum and the original spectrum, using some statistical characterization of the original clean spectrum. Speech enhancement becomes challenging when only a noise-corrupted speech signal is present. Addressing this, [33] proposed a Kalman filtering-based method for speech enhancement that leverages the speech production model. Other statistical approaches to speech enhancement are based on Bayesian statistics e.g minimum mean-square short-time spectral amplitude estimator (MMSE STSA) [34]. Unlike spectral subtraction and Wiener filtering which introduce musical noise and residual noise respectively, MMSE-STSA results in enhanced speech without any musical or residual noise. This method focuses on deriving the MMSE STSA estimator and minimizing STSA from the noise spectrum.

Deep learning-based methods : One of the obvious expectations from a speech enhancement is its generalization to unseen conditions. Most classical approaches can not handle unseen noise types and unseen SNR levels. Performance becomes unsatisfactorily low when such a system is exposed to hostile environments with low SNR levels or when there are speaker and

language variances. Therefore systems have to be trained on large datasets that are rich in several noise types and diverse with multiple speakers. Recently proposed enhancement methods effectively address these issues using SOTA deep-neural networks. Proposed in 2019- based on generative adversarial network (GAN), [35] attempts to use metric scores to optimize generators to cheat the discriminators into reaching the desired score(s), while the discriminator tries not to be cheated by learning the true score. It basically tries to learn the surrogate function of the evaluation metric to achieve multi-metric assignments. Evaluated on the TIMIT dataset with 10 noise types and mixed with 5 SNR levels (-10 dB to 10 dB), MetricGAN achieves an average PESQ score of 2.133 (metric assignment- PESQ) and 2.025 (metric assignment- STOI) (these evaluation metrics have been discussed at length in Chapter 3) on the test set. In 2021, two improvements of this approach were proposed- MetricGAN-unsupervised [36] and MetricGAN+ [37]. The former approach uses an unsupervised mechanism for speech enhancement where only noisy (natural or artificial) audio is required. This is done by optimizing speech quality metrics like DNSMOS and SRMR (speech-to-reverberation modulation energy ratio). A standard method would train a supervised model using clean-noisy pairs, but this approach used only noisy audio. In the later improvement- MetricGAN+, rather than learning metrics for clean speech, we learn metrics for noisy speech when training discriminator. The authors also reuse the data generated from previous epochs to train the discriminator so that it does not forget the behavior of the target evaluation metric. [38] proposed a novel method for noise robust speech recognition called as MimicLoss. It employs unique loss functions that help the speech enhancement model to produce output interpretable by the acoustic model by trying to mimic its behavior under clean speech. Further improved in [39] attempts to improve speech intelligibility further. It compares the outputs of the perceptual acoustic model with clean vs denoised speech as input. The perceptual model is used to judge the perceptual quality of the outputs of the enhancement model. More recent work [40] uses a self-attention type Transformer-based network for speech separation (currently SOTA model) (an in-depth description can be found in Chapter 3, Section 3.2.1). It uses a learnable masking-based architecture where it learns a deep masking network based on self-attention, which estimates element-wise masks and these masks are used for separation. The model is trained on the WSJ0-2mix dataset and achieves a scale-invariant signal-to-noise ratio (SI-SNR) of 22.3 dB. We use this approach for speech enhancement i.e. recover clean speech from a noisy speech signal.

2.3 SpeechBrain

SpeechBrain [41] is a general-purpose open-source conversational AI toolkit designed to replicate the functionality of the human brain. It is focused more on speeding research and development of speech and language processing. Its primary consumers are not just speech researchers, but the entire machine learning community who wish to integrate their models into various

speech pipelines and evaluate them against current state-of-the-art systems. The core implementation of SpeechBrain is based on the PyTorch toolkit and its key features are its strength- being flexible, replicable, easy to use, modular, efficient, and well-documented. It can perform tasks ranging from ASR, speaker recognition, diarization tasks, speech enhancement, dialogue processing, etc. – similar to a human brain.

The architecture of SpeechBrain consists of a `Brain` class defined within `core.py` script that contains all necessary steps to train and evaluate a model using inversion of control fashion. This approach offers the advantage of being more explicit, and abstract and eliminating dependencies. The building blocks of `Brain` class is made up of several functions listed below-

1. `compute_forward`: compute forward pass for given batch.
2. `compute_objective`: compute loss for given batch.
3. `on_stage_start`: gets called when a stage (train, test, or valid) starts, typically used to initialize error metrics.
4. `on_stage_end`: gets called when stage ends. It computes the error metrics initialized above, updates the learning rate, updates losses, and prints training logs.
5. `fit()`: iterates over epochs and dataset- basically fitting over the train set and valid set to improve the objective and save the model checkpoint.
6. `evaluate()`: usually called at the end of the training, it iterates over the test set and evaluates the brain performance.

SpeechBrain dataloader uses a basic PyTorch data-loading technique wherein it addresses usual problems in time-series sequences (like speech signals) like variable length and large and complex datasets. Its `DynamicItemDataset` offers a unique and flexible approach to dynamically fetch and transform data before beginning a training loop. The toolkit is easily compatible with datasets annotated in JSON or CSV format, which usually contains text transcripts, phonemes, and path-to-wave files. For all tasks and recipes defined within SpeechBrain, one can train a model simply by calling the training script followed by a human-readable SpeechBrain-developed format hyperparameter file- `python train.py hparams.yaml`. The hyperparameter file is not just a plain text file with a list of hyperparameters, rather we declare variables and objects with their corresponding arguments that we use. These objects control the data loading pipeline, model architecture, decoding, evaluation metric, etc.

In this work, not only do we make extensive use of the SpeechBrain toolkit but also actively contribute our trained models and results for each experiment in a systematic manner. For each result shown in this work, we share full training logs for replicability purposes.

Chapter 3

Technical Background

In this chapter, we dive deep into the technical nuances of speech recognition and speech enhancement. We introduce popular End2End speech recognition models like Connectionist Temporal Classification (CTC), Attention-based Encoder-Decoder models, and End2End architectures that we have used in this thesis. Further, we talk about methods for speech enhancement that have been used in our work.

3.1 End2End Automatic Speech Recognition

End2End modeling approach in speech recognition directly outputs a sequence of tokens using a single network that is trained on a single objective function- unlike traditional hybrid models that consists of multiple components each being optimized separately. It completely eliminates the need to perform multi-stage training and results in a simpler system where the entire focus of the network is to act upon a single objective. It has fewer parameters than traditional models making them less prone to overfitting. This makes End2End models simple, flexible, and outperform traditional hybrid models. In this section, we discuss a few End2End models, popular End2End architectures that have been SOTA at the time of writing, and evaluation metrics adopted in this work to judge the quality of speech recognizers.

3.1.1 End2end Models

The basic building block of an End2End model constitutes an Encoder, an alignment block, and a decoder. The encoder takes in raw audio data to map feature representation vectors into some hidden representations using convolutional neural networks (CNN) or recurrent neural networks (RNN). The aligner or the alignment block aligns the input acoustic frames with the output tokens (character or word sequence) and lastly, the decoder uses encoder output (hidden representations) to predict the final sequence of tokens. It uses the aligner's output to predict

the corresponding text of an acoustic frame. The popular End2End models are (i) CTC (ii) Attention-based Encoder-Decoder model

Connectionist Temporal Classification CTC is designed to map or align input speech frames with text transcriptions (or output labels). It uses dynamic programming to predict the sequence of labels. *But what does an alignment do?* Given an acoustic frame, it tells us the corresponding word (or character) spoken in that frame. This alignment is necessary because the length of the input sequence is much larger than the output labels. Without an alignment, the model would fail to learn the correct mapping between input and output.

CTC models the probability distribution at each time step over all possible alignments. Input acoustic frames have many silent frames and also consist of repeated characters (e.g. *he**ll**o*). It is often the case that the output label is smaller than the input sequence. Therefore, to solve these issues, a `blank` token is introduced which is used to represent a null transition between different output labels. This token does not provide any acoustic or temporal information but allows the model to insert or delete output labels to perform the alignment of input frames with labels.

It is important to note a few important properties of CTC alignment. Once the alignment is done, the length of the aligned sequence is the same as the input sequence which results from merging subsequent tokens and discarding `blank` token. Moreover, the nature of such an alignment is many-to-one – meaning one or more input frames are aligned to a single output label.

Here we describe the step-wise training and decoding process

1. **CTC training**- CTC works by maximizing a conditional probability $p(Y|X)$ to an output label Y for a given input X

$$p(Y|X) = \sum_{A \in \mathcal{A}_{X,Y}} \prod_{t=1}^{T} p_t(a_t|X) \tag{3.1}$$

   The above equation [21] indicates the probability of an alignment a_t at time step t for a given input. The above objective function is marginalized over the set of all alignments $A \in \mathcal{A}_{X,Y}$ between the input acoustic frames and the output labels. During training, the input sequence is fed into an RNN network after which it computes the probability for each output (including `blank` token) $p_t(a_t|X)$ at each time-step. This gives us a matrix of probabilities (the row equals all possible outputs and the column equals the number of inputs). CTC then computes the probability of all possible valid alignments and marginalizes them to get the final distribution i.e. probability of output for a given input.

This is visualized in Figure 3.1. With the ground truth already given, it further employs a loss function where the loss is computed, and error gradients are back-propagated, thereby updating the network weights.

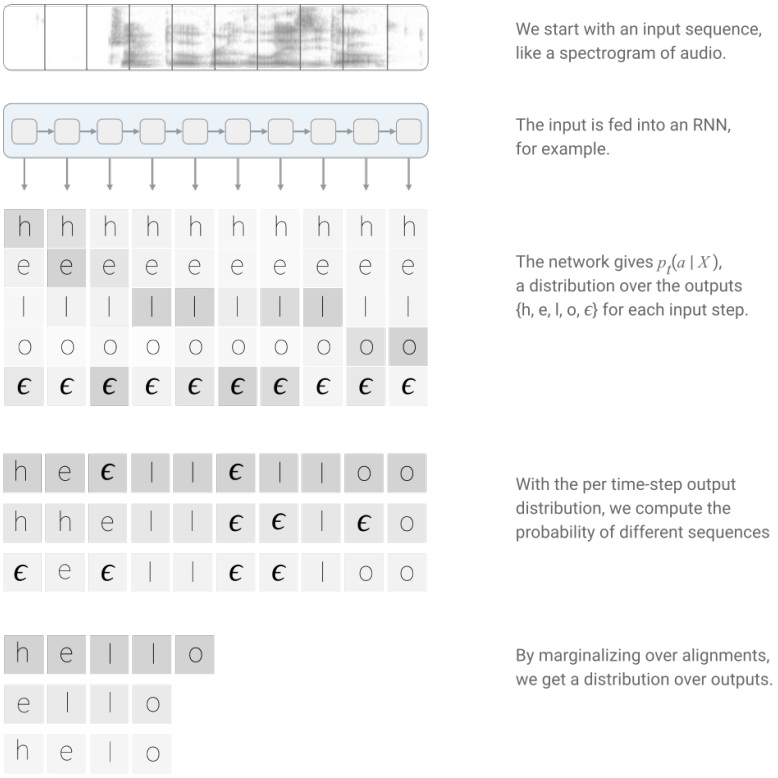

FIGURE 3.1: This illustration [21] explains how CTC determines the conditional probability of an output label for a given input sequence.

2. **CTC decoding**- Once the model is trained, the task is to determine the most probable output sequence for a test input. Basically, we are interested in finding an alignment that maximizes the conditional probability $p(Y|X)$.

$$Y^* = \underset{Y}{\operatorname{argmax}}\, p(Y|X) \tag{3.2}$$

$$Y^* = \underset{Y}{\operatorname{argmax}} \prod_{t=1}^{T} p_t(a_t|X) \tag{3.3}$$

This is done using beam search. It computes a new set of hypotheses at each time step and in the next time step the last hypothesis is extended to obtain a new set of hypotheses, keeping only the top candidate with maximum probabilities. However, keeping all alignments in the beam can be a computationally expensive process. The standard beam search is modified in CTC to handle multiple alignments by merging repeated tokens and getting rid of `blank` tokens whereafter only the output prefix is kept in the beam. In Figure: 3.2, in the third time step, we merge multiple extensions to one hypothesis, and this is further extended to a prefix $[a]$ with two outputs– $[a]$ and $[a, a]$.

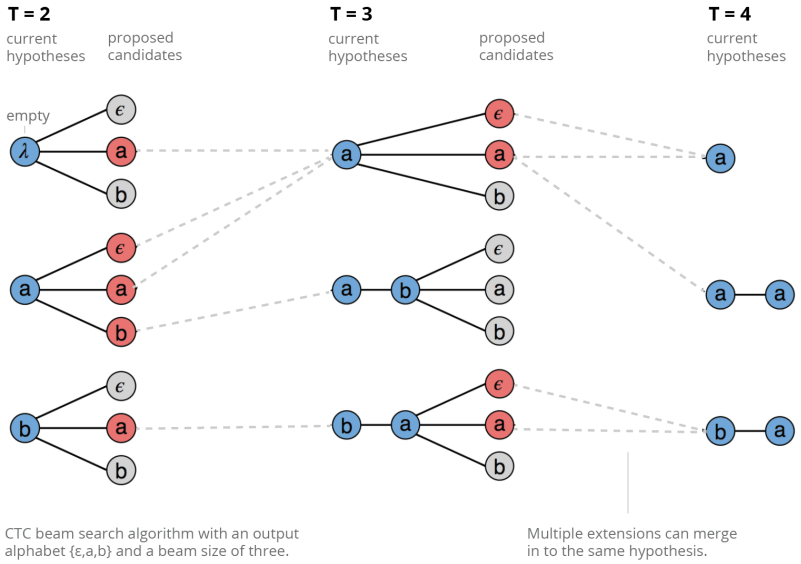

FIGURE 3.2: Illustration from [21]– it explains how CTC decoding uses a modified beam search algorithm to determine the most likely output sequence.

Attention-based Encoder-Decoder model The attention-based encoder-decoder (AED) model is another model used for speech recognition that leverages the attention mechanism to attend to different parts of the input speech and improve the quality of speech recognition. This architecture is frequently used as a sequence-to-sequence (Seq2Seq) model with the idea of transforming a variable-length input sequence into a variable-length output sequence. Although the seq2seq model has been successfully applied to many NLP tasks, it turned out not to be well-suited for longer input sequences. Specifically, the contextualized representation obtained from the encoder would prefer later information in an input sequence while forgetting those from the beginning. The solution for this problem is to use the attention mechanism, which allows the decoder to attend to all input tokens during each decoding step instead of a single contextualized representation. There are various types of this mechanism, but the overall idea is to calculate the input token's attention scores for each decoding step individually. These scores, ideally, tell the decoder which information is relevant for the current step.

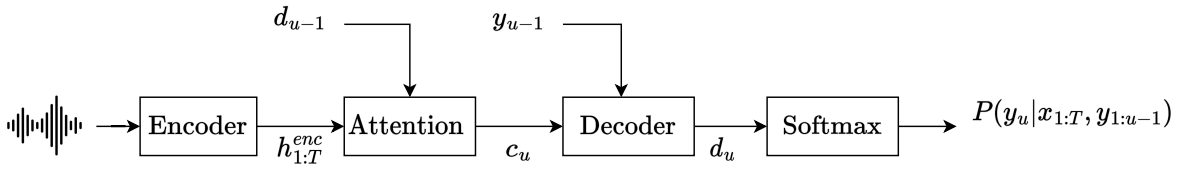

FIGURE 3.3: Block diagram of attention based encoder-decoder model [42]

As shown in Figure 3.3 it consists of an encoder network, an attention mechanism, and a decoder network. Specifically, the steps of the model are as follows:

1. The encoder uses a bidirectional LSTM that processes the input acoustic features into a sequence of high-level hidden vector representation. In Section 3.1.2 we shall discuss our encoder architecture in detail.

$$h_{1:T} = Enc(x_{1:T}) \tag{3.4}$$

$$\boldsymbol{h} = [h_1, h_2, \cdots, h_T] \tag{3.5}$$

2. Next, attention mechanism is applied over encoded representations and previous decoder output to decide which part of the input sequence is relevant for the current decoding step. It generates a weighted sum of encoder's hidden state– context vector c_u.

$$c_u = Attn(\boldsymbol{h}, d_{u-1}) \tag{3.6}$$

3. The decoder then takes the context vector and previously generated output label to compute the probability of an output label $P(y_u|x_{1:T}, y_{1:u-1})$ given previous label outputs.

$$d_u = Dec(c_u, y_{u-1}) \tag{3.7}$$

$$Softmax(d_u) = P(y_u|x_{1:T}, y_{1:u-1}) \tag{3.8}$$

There are different types of attention mechanisms used in speech recognition like location-aware attention, content-based attention, and key-value-based attention. However, in our work, we use a location-based attention mechanism [43] that uses learnable attention parameters capable to track the absolute location of contents in the input sequence that the decoder should focus on.

3.1.2 End2end Architectures

When it comes to building a speech recognition pipeline encoder plays a crucial role in mapping input acoustic frames to high-level feature representation. The way encoding is done has a proportional impact on the accuracy of the desired system. It should form compact representations capturing essential features and information for transcription. In this work, we use the popular CRDNN architecture ([44], [45]) and SOTA wav2vec 2.0 [8] architecture as encoders for our ASR pipeline. However, for comparison purposes, we include the WavLM and SOTA Whisper model but do not discuss them in detail in our work.

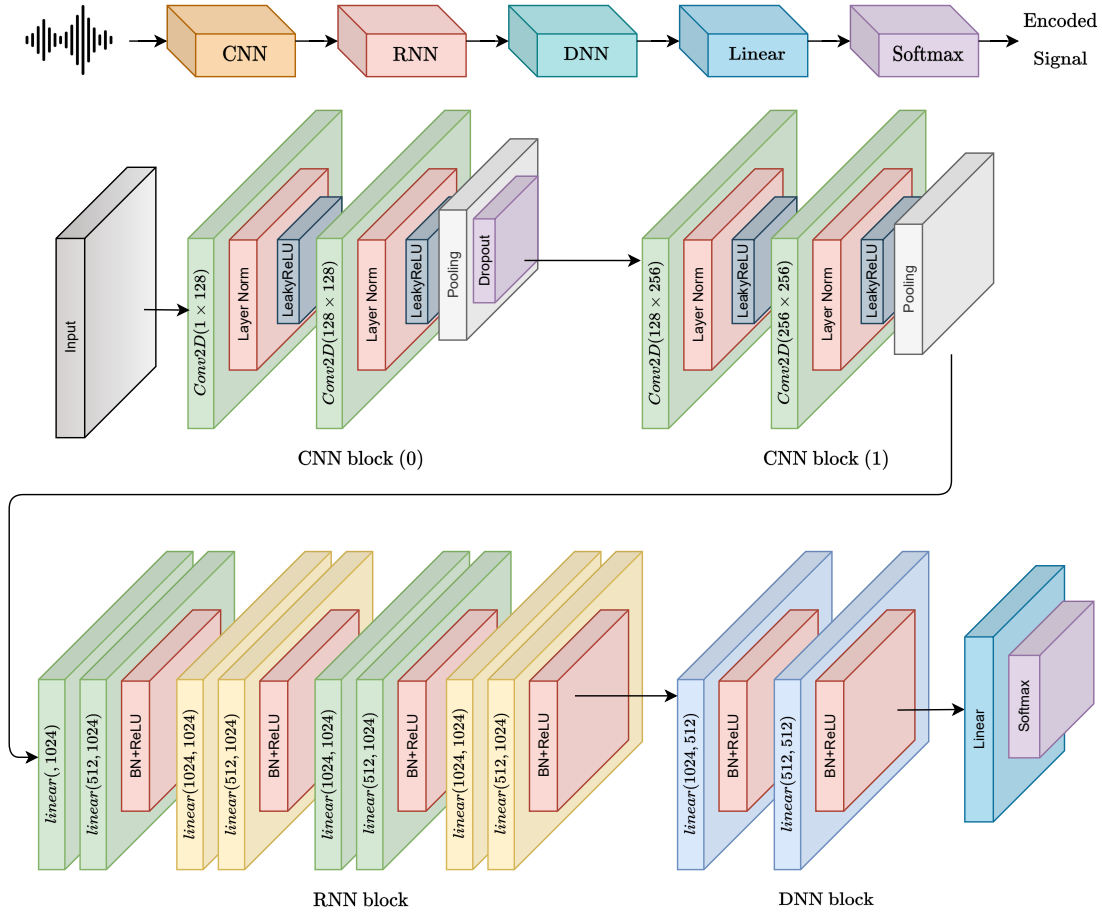

FIGURE 3.4: CRDNN architecture comprising of 2 CNN blocks, 1 RNN block, and 1 dense-neural network (DNN) block, followed by a linear layer and a softmax.

CRDNN CRDNN or convolutional, recurrent, and dense-neural network is a complex neural network often used in cases where the goal is to capture spectral as well as temporal dependencies like speech recognition. As shown in Figure 3.4 the CRDNN architecture is a combination of CNN blocks, RNN, and a DNN block followed by a linear and a softmax layer.

The input to CNN is 40 mel-filterbank coefficients. The CNN layer performs spectral modeling using filters to convolve over the input speech features. In our encoder architecture, we use two such CNN layers with a channel size of (128, 256) to extract high-level speech features. These features are then fed into an RNN block that performs temporal modeling. The RNN block is made of 4 bidirectional-LSTM layers with 1024 neurons in each layer. It performs time-series analysis on the output sequence and extracts key contextual information. It also captures the long-range dependencies in the input signal. RNN output is passed through DNN layers to better capture the non-linear relationship between input speech features and output transcriptions. It transforms the RNN output into a probability distribution over the output transcripts. Since this probability distribution is in a higher dimensional space, we use a linear

layer to project the outputs of DNN into a lower dimension and pass it through softmax to get the probability of the predicted transcript which are sub-word units (size=1000).

For decoding, we use an attentional RNN decoder of type gated recurrent unit (GRU). Beam-search coupled with an RNN-based language model is used on top of the decoding probabilities. The sub-word units estimated with byte-pairwise encoding (BPE) are used as basic recognition tokens. We describe the model parameters and hyper-parameters in Section: 5.1.

wav2vec 2.0 This model used in our work is based on [8] which is an extension of wav2vec [46]. Wav2vec is a self-supervised approach for learning speech representations using unlabelled data. As interesting as it sounds, the functionality is similar to BERT (bidirectional encoder representations from Transformer where a part of the speech representation is masked and the model is pre-trained to predict the small speech unit for the masked parts. Along with this, the model is also trained to identify the predicted speech units. As speech signals are continuous time series with no clear separation between subsequent words, the model learns speech units of 25 ms long. This enables learning high-level contextualized representations. The pre-training is done on a large amount of unlabelled and unannotated audio data. In order to perform downstream tasks, these contextualized representations are then refined using a small labeled data set, reducing the need for large amounts of annotated text in ASR tasks.

Wav2vec2.0 on the other hand, is an improved, more accurate, and robust successor of wav2vec. It uses deeper neural network architecture with more layers and is trained on diverse (rich in speaker and speech variabilities) and increased amounts of data. The framework has been illustrated in Figure: 3.5 - let us try to understand step by step:

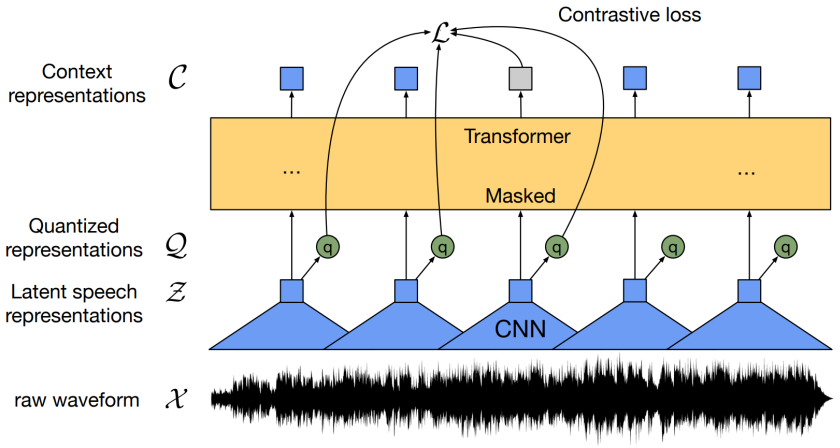

FIGURE 3.5: Wav2vec2.0 framework [8]

1. **Encoder**: The first element of the network is a multi-layer CNN feature encoder, which inputs raw audio and produces intermediate representations. Given an input audio stream $\mathcal{X}$, the encoding function f is determined by $f : \mathcal{X} \rightarrow \mathcal{Z}$.

2. Outputs of CNN are latent speech representations $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_T]$. The representations are 'latent' due to their ability to capture important hidden pieces of information which are not directly observable in the input audio but can be inferred by the model. They are compressed forms of the input, thus reducing the data dimensionality.

3. **Transformer network**: The latent representations are then fed into a Transformer network, defined by $g : \mathcal{Z} \rightarrow \mathcal{C}$, with a self-attention mechanism that captures long-range dependencies. Additionally, it employs relative positional embeddings (different from the usual absolute positional embeddings) to capture the relative positions of speech units in the input audio. It helps the model to learn the relationship between different phonemes or acoustic units. Output to this block is a vector of contextualized representations $[\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_T]$.

4. **Quantization module**: Lastly, the latent speech representations are discretized into fixed-length vectors since text transcripts are also a discreet set of finite elements. This quantization helps generate discriminative speech embeddings which are used for contrastive loss computation.

The first step in pre-training involves covering certain portions of the speech representations and then training the model to predict what has been masked. Basically, the model learns to predict the quantized representations for each masked time step similar to the ones generated in the quantization module. As a training objective, contrastive loss $\mathcal{L}_m$ is used. The idea here is to maximize agreement between similar representations and minimize agreement between dissimilar representations.

$$\mathcal{L}_m = -\log \frac{\exp\left(sim(\mathbf{c}_t, \mathbf{q}_t)/\mathcal{K}\right)}{\sum_{\tilde{\mathbf{q}} \sim \mathbf{Q}_t} \exp\left(sim(\mathbf{c}_t, \tilde{\mathbf{q}_t})/\mathcal{K}\right)} \tag{3.9}$$

Let's take a closer look-

- *Numerator*: given true quantized representation $\mathbf{q}_t$ and context representation $\mathbf{c}_t$, we compute cosine similarity between these: $sim(\mathbf{c}_t, \mathbf{q}_t)$. Then we normalize this with total number of predicted quantized candidates (or distractors) $\mathcal{K}$ to get a probability score.

- *Denominator*: compute a normalized similarity score between the predicted quantized representation $\tilde{\mathbf{q}_t}$ and context representation. This is then marginalized over the set of all distractors: $\sum_{\tilde{\mathbf{q}} \sim \mathbf{Q}_t} \exp\left(sim(\mathbf{c}_t, \tilde{\mathbf{q}_t})\right)$

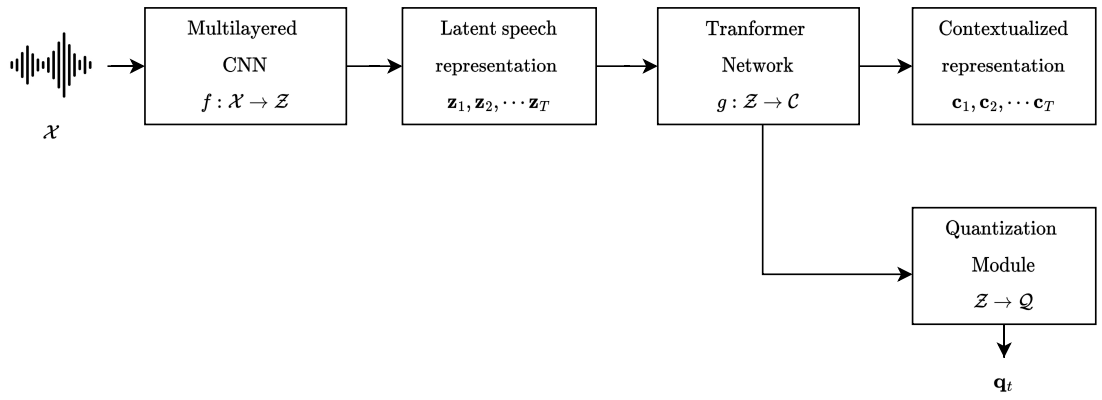

FIGURE 3.6: Wav2vec2.0 architecture

After the model is pre-trained on a large unlabeled dataset, it is then fine-tuned on a task-specific labeled dataset by minimizing the CTC loss function (see CTC in Section: 3.1.1). Fine-tuning helps update the model's weight and guide the model towards a specific learning task as desired.

3.1.3 Evaluation metrics

To assess the quality of transcripts generated by a speech recognizer, we rely primarily on two evaluation metrics- word error rate (WER) and character error rate (CER). These metric scores compare the gold transcript and the predicted transcript and tell us the number of words/chars incorrectly recognized. WER is computed using the formula below:

$$WER = \frac{I + S + D}{N} \tag{3.10}$$

where

- I: total counts of insertions
- S: total counts of substitutions
- D: total counts of deletions
- N: total counts of words in the reference text

To compute CER, the predicted sequence of words is converted into a sequence of characters, and the above formula is used.

3.2 Single Channel Speech Enhancement

Single-channel speech enhancement (SE) involves improving audio intelligibility and the quality of audio recordings done on a single channel. As discussed in Section 2.2 traditional methods for speech enhancement fail when exposed to unknown noise types and low SNR noise levels. Recent years have seen deep-neural networks to be very successful for SE tasks. In this work, we perform SE on mono audio recordings using *SepFormer*– a Transformer-based RNN-free network [40] for speech separation while preserving the speech content and intelligibility to later perform speech recognition.

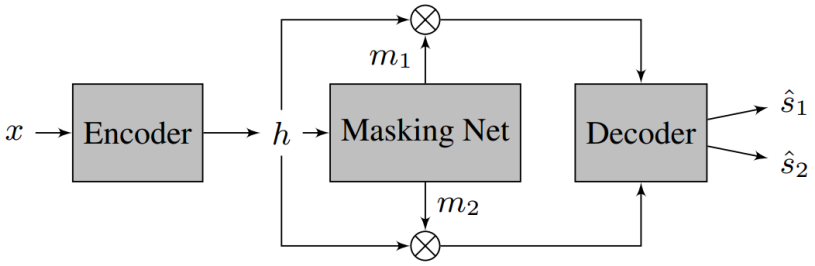

FIGURE 3.7: A high-level block diagram of SepFoprmer architecture.

3.2.1 *SepFormer*: Transformer-based neural network

SepFormer is a multi-head attention Transformer-based source separation architecture. It adopts the dual-path RNN (DPRNN) [47] that models long sequential inputs by splitting them into chunks. DPRNN uses two separate RNNs to model both local dependency within a chunk as well as global dependency across chunks. Inspired by this, SepFormer replaces RNNs with Transformer networks to model both short-term and long-term dependencies.

It uses a fully learnable masking-based architecture composed of an encoder, a masking net, and a decoder. The encoder and decoder blocks are essentially convolutional layers and we learn a deep-masking network based on self-attention which estimates element-wise masks. These masks are used by the decoder the reconstruct the enhanced signal in the time domain. Let's understand each of these blocks in detail.

Encoder The encoder inputs a noisy audio signal in the time domain that learns STFT representations using a single-layered convolutional network.

$$h = ReLU(conv1d(x)) \tag{3.11}$$

Masking Network Figure 3.8 illustrates the masking network. The encoder output h is fed into a normalization layer followed by a linear layer to allow the model to learn complex non-linear representations. Then chunking with overlapping is applied to the activations to split the sequence into chunked representations. In the next step, SepFormer is applied to the sequence of chunked representations (architecture shown in Figure 3.9). As mentioned earlier, SepFormer is based on DPRNN constituting an IntraTransformer (IntraT), a permute, and an InterTransformer (InterT) block. IntraT takes in the chunked representations h' and models the short-term dependencies, and then permute block permutes its last two dimensions. Further, InterT is applied to the permuted output to model the long-term dependencies. The combined operation can be represented as

$$h'' = f_{inter}\left(\mathcal{P}(f_{inter}(h'))\right) \tag{3.12}$$

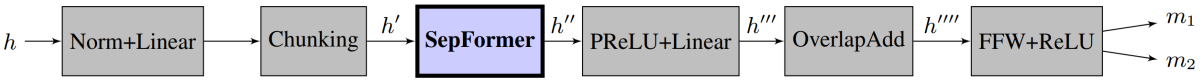

FIGURE 3.8: Masking network

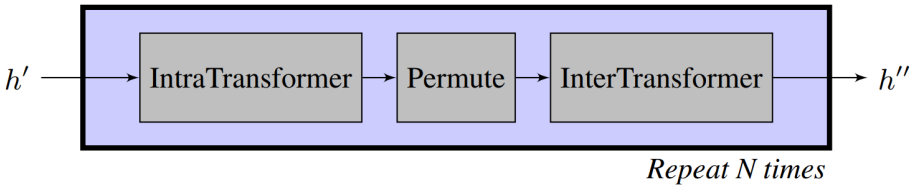

FIGURE 3.9: SepFormer block diagram that combines IntraTransformer and InterTransformer to model short-term and long-term dependencies.

The transformer block is shown in Figure 3.10. Let the input be z. First positional embedding e is added to z to provide additional information about the positions of speech signals in a noisy mixture. Next, we apply layer normalization followed by multi-head attention. This allows attending to different parts of the sequence differently. Lastly, a normalization layer followed by a feed-forward layer is applied to get the output representations. These operations can be described as-

$$z' = z + e \tag{3.13}$$

$$z'' = \text{MHA}(\text{LayerNorm}(z')) \tag{3.14}$$

$$z''' = \text{FF}(\text{LayerNorm}(z'' + z')) + z'' + z' \tag{3.15}$$

The SepFormer block is repeated N times, inside which K Transformer layers are repeated.

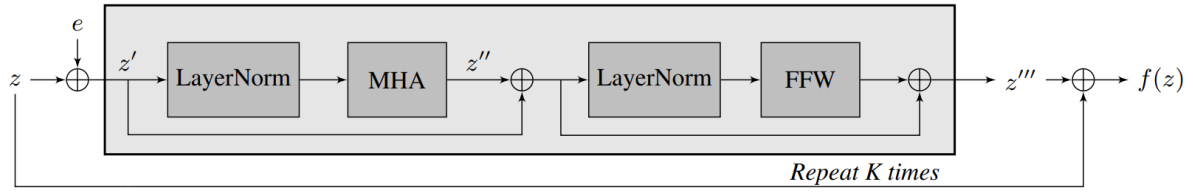

FIGURE 3.10: Transformer network used by both IntraT and InterT.

Decoder The decoder is a transposed conv1d layer that that inputs the masked representations m_k and encoder output h, and performs an element-wise multiplication. This can be represented by

$$\hat{s_k} = \text{Conv1d-transposed}(m_k * h) \tag{3.16}$$

$$\hat{s}_k = (\hat{s}_{\text{clean}}, \hat{s}_{\text{noise}}) \tag{3.17}$$

where k are the number of sources in the noisy mixture- which in our case is 2 (clean speech signal and noisy signal).

The model is trained in a supervised fashion using the permutation invariant SI-SNR [48] (see Section 3.2.2.1 for further details).

$$s_{\text{target}} := \frac{\left(\hat{s}_{\text{clean}}^{\top} s\right) s}{||s||^2} \tag{3.18}$$

$$e_{\text{noise}} := \hat{s}_{\text{clean}} - s_{\text{target}} \tag{3.19}$$

$$\text{SI-SNR} := 10 \log_{10} \left(\frac{||s_{\text{target}}||^2}{||e_{\text{noise}}||^2} \right) \tag{3.20}$$

where s is the target clean speech signal, $\hat{s}_{\text{clean}}$ is the enhanced speech signal, and s_{target} is the scaled target clean speech signal.

3.2.2 Evaluation metrics

Speech enhancement systems are trained with the purpose of reducing noise and distortions, thereby improving the quality of speech signals. Primarily, the performance evaluation of a SE system has two aspects- quality of enhanced speech and its intelligibility. The quality of speech signal refers to how good or bad is the enhanced signal and whether is it still contaminated with unwanted noise. While intelligibility refers to how well a listener understands the enhanced speech signal. Both quality and intelligibility can be measured using subjective and objective methods. Subjective methods involve human listeners evaluating speech intelligibility by performing listening tests and rating it on a scale of 1 to 5. They are by far the most accurate and reliable metric to evaluate SE systems. But, such a task is unscalable, resource expensive (both

in terms of time and money), and requires several native (or fluent) speakers of the language to ensure the accuracy and reliability of the results. On the other hand, objective evaluation methods use metrics like signal-to-noise ratio (SNR), signal-to-distortion ratio (SDR), mean opinion score (MOS), and perceptual evaluation of speech quality (PESQ) to quantify the quality of the enhanced speech. In this work, we use these objective methods to evaluate our SE systems.

3.2.2.1 SI-SNR & SI-SDR

SI-SNR (scale-invariant signal-to-noise ratio) [48] is a commonly used SE system evaluation metric. It is based on the signal-to-noise ratio that computes the ratio of the energy of the target speech signal to the energy difference between the enhanced signal and target signal (see Equation 3.20). Both the enhanced speech signal and original clean speech signal are normalized to have zero mean which ensures that the metric is scale-invariant. To have the best possible enhanced signal, the energy difference $\hat{s}_{\text{clean}} - s_{\text{target}}$ should be minimum. This indicates that the higher the SI-SNR value, the better would be enhancement performance. In our work, we measure the improvement in the SI-SNR value (SI-SNRi) to assess enhanced speech signal quality.

$$\text{SI-SNRi} = \text{SI-SNR}(\text{enhanced}, \text{target}) - \text{SI-SNR}(\text{noisy}, \text{target}) \tag{3.21}$$

However, SI-SNR comes with certain limitations- it assumes that the scale of enhanced and target speech signals are the same and that the noise is stationary in the noisy signal. A better alternative to SI-SNR is- **SI-SDR** (scale-invariant signal to distortion ratio) [49]. It is considered a better metric for evaluating overall how good a speech signal sounds. It accounts for the scaling of the signals.

$$\text{SI-SDR} = 10 \log_{10} \left(\frac{||e_{\text{target}}||^2}{||e_{\text{res}}||^2} \right) \tag{3.22}$$

$$\text{SI-SDRi} = \text{SI-SDR}(\text{enhanced}, \text{target}) - \text{SI-SDR}(\text{noisy}, \text{target}) \tag{3.23}$$

where e_{res} is the residual signal and e_{target} is the scaled target signal.

3.2.2.2 PESQ and STOI

PESQ or Perceptual Evaluation of Speech Quality [50] is a widely accepted standard evaluation metric for enhanced speech signals. Developed by International Telecommunication Union (ITU) in 2000, its initial purpose was to assess audio quality in telecommunications channels. But today, it is very popular in speech enhancement and noise reduction tasks. It is an intrusive metric as it compares reference clean speech signals and enhanced speech signals. Based on the

spectral comparison, it measures the perceptual difference between the clean and the enhanced signal by analyzing the level of degradation in the enhanced signal. It is measured on a scale of -0.5 to 4.5 with 4.5 being the best quality. In this work, we use the PESQ package as published on PyPi [51].

Short-time objective intelligibility (STOI) [52] is another common metric used to estimate the quality of enhanced speech signals. It is an intrusive metric- a function of both clean and enhanced signals. It relies upon spectro-temporal characteristics of short envelopes (300-400 ms) to measure the intelligibility taking into account the effect of masking i.e. reduction in the audibility of softer sound in presence of some louder sound. It is measured on a scale of 0-1 with a high score indicating better intelligibility.

3.2.2.3 DNSMOS P.835

The above-discussed conventional metrics are intrusive as they rely on a "gold-standard" reference signal. However, in real-life scenarios, we often do not have access to clean recordings and this makes the evaluation of speech enhancement tasks difficult. DNSMOS [53, 54], - deep noise suppression (DNS)- mean opinion score (MOS) was proposed in 2021 to evaluate and rank the submissions in the deep-noise suppression challenge. It is a non-intrusive evaluation metric as it does not need the reference clean speech signals and provides results with a high correlation to human evaluation. It uses a single-layered convolutional network to train a model on human ratings obtained from ITU-T P.808 [55]. It computes 3 scores– SIG (speech quality), BAK (background noise quality), and OVRL (overall quality) each on a scale of 1 to 5, with 5 being the best quality.

Chapter 4

Dataset Description

4.1 Related Corpora

To improve the accuracy of speech recognition systems in noisy and reverberant environments, several corpora have been developed, such as CHIME [56–59], DIRHA [60–63], AMI [64], VOiCES [65], and COSINE [66]. Among these, CHIME5 [58] and CHIME6 [59] are especially challenging because it contains conversational speech recorded during a dinner party in a domestic setting, where noise and reverberations are common. RescueSpeech also contains conversational speech recorded in challenging acoustic environments, but the scenario addressed in this corpus is unique and different from a dinner party. The acoustic conditions, emotions, and lexicon used in RescueSpeech are distinct, and thus provide an additional set of challenges for speech recognition systems.

The noisy version of RescueSpeech (see Section: 4.3) can be utilized to train speech enhancement systems that are robust in the acoustic conditions present in the Search and Rescue (SAR) domain. There are numerous datasets that have been released for speech enhancement purposes, including the deep-noise suppression (DNS) dataset [67], VoiceBank-DEMAND corpus [68], and WHAM! and WHAMR! corpora [69], all of which are helpful for training speech enhancement models. However, the key difference with RescueSpeech is that it has been specifically designed for the SAR domain, where characteristic sounds such as sirens, radio signals, helicopters, trucks, and others affect the recordings. This unique characteristic of RescueSpeech makes it an especially valuable resource for training speech enhancement systems that can perform well in SAR environments.

4.2 General Training Data

Speech recognizers and enhancement systems backed with DNN require thousands of hours of speech data to compete against human-level accuracy. Our speech recognizers– CRDNN,

TABLE 4.1: Distribution of sentences and hours in the German CommonVoice10.0 and DNS4 dataset.

	CommonVoice10.0		**DNS4**	
	HRS	#Utts.	HRS	#Utts.
Train	739.17	466189	1317	1186019
Valid	26.97	16067	6.67	5965
Test	27.15	16067	5.17	921

wav2vec2.0 [8], WavLM [9] are pre-trained on full German language Mozilla CommonVoice10.0 corpus [1]. CommonVoice is a massive multilingual speech corpus used primarily for speech technology research and development. The latest version of the dataset consists a total of 27K hours of speech data recorded in 108 languages by more than 50,000 speakers around the world, of which 17K hours are validated. The utterances are recorded in a mono-channel, 16-bit setup and released in MPEG-3 format with 48K Hz sampling rate. The German CommonVoice10.0 version of the dataset used in this work comprises of total 1200h and 498K utterances. Table: 4.1 briefly describes the train/test/valid data statistics.

Our speech enhancement system is trained on DNS4 dataset [67] which was released as a part of ICASSP'22- Deep Noise Suppression Challenge-4. DNS4[7] dataset consists of more than 500h of clean utterances (read speech, French, Spanish, German, Italian, and Russian speech), noisy clips (150 noise types) and real and synthetic room-impulse responses (RIRs). Using provided clean utterances, noisy clips (150 noises types), and RIRs, we generate 1300h of train and 6.7h of the valid set at varying SNR (from -5 dB to 15 dB with a step of 1 dB), and a DNS-2022 baseline dev set is used as the test set. The Sampling rate is set to 16 kHz and only 30% of clean speech is convolved with RIR. Table: 4.1 briefly describes the train/test/valid synthesized data statistics from DNS4 data.

Once the speech recognizers and enhancement models are trained on these datasets, we further fine-tune them on our RescueSpeech dataset.

4.3 The RescueSpeech Dataset

Our dataset is composed of audio recordings by native speakers of the German language made during several simulated SAR exercises. The rescue operation simulated accidents like residential fire, explosions, etc. in presence of a team of rescuers and the conversations were carried out between team members, radio operators, and the team leader. These conversations loosely adopt a typical radio style communication wherein the start/end of a conversation is indicated by the use of certain words, connection quality is relayed, and acceptance or rejection of requests are

[7] https://github.com/microsoft/DNS-Challenge

conveyed. Initially captured at 44.1 kHz sampling rate, these recordings are down-sampled to 16 kHz, and further segmented to obtain a set of mono-speaker single-channel audio recordings. All utterances are also manually transcribed. The total length of the dataset is 1.5h with a total of 1980 sentences with 1269/400/311 sentences in train/test/valid set. We call it the **RescueSpeech clean dataset**. Figure 4.1 shows a histogram plot of the average length of the segmented utterances with an average length of 2.29 sec. We also created a noisy version

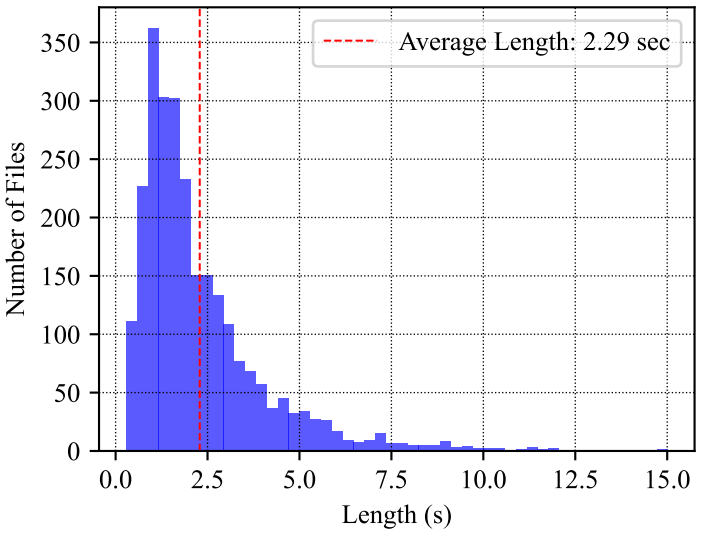

FIGURE 4.1: Histogram plot illustrating the average length of utterances in RescueSpeech in secs.

TABLE 4.2: Distribution of sentences and hours in the RescueSpeech clean and noisy dataset.

	Clean		**Noisy**	
	HRS	#Utts.	HRS	#Utts.
Train	1.02	1543	4.84	3000
Valid	0.26	387	1.43	900
Test	0.32	484	1.40	900

of RescueSpeech by contaminating our dataset with noisy clips from the AudioSet dataset [70] that includes five noise types– *emergency vehicle siren*, *breathing*, *engine*, *chopper*, and *static radio noise*. We utilized both real and synthetic room-impulse responses (RIR) (SLR26, SLR27 [71]) to add reverberation as well. We then added noise sequences to generate noisy utterances with different signal-to-noise ratios (SNR) (from -5 dB to 15 dB with a step of 1 dB). Each clean utterance is randomly corrupted with one of the noise types to generate 3000/900/900 train/valid/test utterances. We also ensure that a noise utterance used in the train set is only in this set. This randomness and exclusivity ensure that each split has an equal proportion for each noise type and that noises in each of the splits are different. This dataset provides a diverse set of noise and reverberation conditions that enable fine-tuning of our speech-enhancement

model for improved accuracy on noisy RescueSpeech. We call this the **RescueSpeech noisy dataset**.

Chapter 5

Experimental Setup

We explored multiple training strategies to perform noise robust speech recognition. Speech recognizers and enhancement models are trained on large corpora and then fine-tuned and evaluated on RescueSpeech data.

5.1 ASR training

We follow two approaches for ASR training- one based on sequence-to-sequence modeling (seq2seq) and the other on the connectionist temporal classification (CTC) method. Both of these involve training a tokenizer on train transcripts of CommonVoice10.0 corpus [1] using SentencePiece [72].

Training scheme : For the seq2seq model, we employ a CRDNN (convolutional, recurrent, and dense-neural network) architecture [44, 45]. The tokenizer generates unigram tokens, and we limit the number of token outputs to 1000 in order to ensure that the model is fed with concise and relevant input. The CRDNN encoder is trained on the full 1200h of the German CommonVoice10.0 corpus. The network is trained on both CTC and negative log-likelihood loss with unigrams as basic recognition units. For decoding, we utilize an attentional-Gated Recurrent Unit (GRU) decoder coupled with a beam search algorithm. Additionally, we incorporate an RNN-based language model (LM) to improve the performance of the model. The LM is trained on total 17M sentences combining Tuda-De[2] [73] (8M sents), Leipzig news corpus [74] (9M sents), and train transcripts of the CommonVoice corpus.

For the CTC-based models, we use wav2vec2.0, and WavLM architecture as encoders for the ASR pipeline. These encoders use a self-supervised approach for learning high-level contextualized speech representation. The model is trained by minimizing CTC loss with characters as basic recognition units. It needs no language model, and decoding is performed

[2]https://www.inf.uni-hamburg.de/en/inst/ab/lt/resources/data/acoustic-models.html

using greedy search. For wav2vec2.0 and WavLM we use pre-trained encoders `facebook/wav2vec2-large-xlsr-53-german`[3] (pre-trained on 56K hours of unlabelled data) and `microsoft/wavlm-large`[4] (pre-trained on 84K hours of unlabelled data) respectively. These models have been pre-trained on large amounts of unlabelled data, making them highly effective at recognizing German speech. To further improve the accuracy of our ASR system, we fine-tuned these pre-trained models on the full German CommonVoice10.0 corpus. Additionally, we also employ a pre-trained Whisper [10] model `openai/whisper-large-v2`[5] (pre-trained on 680K hours of multilingual speech data) to benchmark our systems against a competitive state-of-the-art model. This model uses a Transformer architecture and has achieved impressive results on several speech recognition tasks. The pre-trained Whisper model does not need any training with CommonVoice data. It is only fine-tuned on RescueSpeech dataset.

Model and training parameter, hyperparameters: LM training is based on RNNLM, which is a combination of embedding layer, RNN, and DNN. The output layer of the model has 32 neurons, which generates the probability distribution over the vocabulary. The Embedding size is 128, and LeakyReLU activation is used to introduce non-linearity in the model. The model has 2 layers of RNN, with 2048 neurons in each layer. In addition, the model also has a single fully connected layer with 512 neurons. The model has 52.5M trainable parameters and is trained for 20 epochs on a batch size of 64 with a learning rate (LR) of 1e-4. Each epoch takes approximately 3.3h on a single RTX3090 GPU with 24GB of memory. CRDNN encoder (see Figure 3.4) combines two blocks of CNN (each block with 2 CNN layers with a channel size (128, 256)), an RNN block (4 bidirectional LSTM layers with 1024 neurons in each layer), and 2 blocks of dense-neural network layer, with 512 neurons in each layer. The inputs are 40-dimensional mel-fiterbank features, and the network is trained with an AdaDelta [75] optimizer with a learning rate (LR) of 1 (during fine-tuning, we use LR 0.1). The model has 173M trainable parameters and is trained for 25 epochs with a batch size of 8. During testing, beam search is used with a beam size of 80. Each epoch takes approximately 8h on a single RTXA6000 GPU with 48GB of memory.

For wav2vec2.0 and WavLM CTC, total trainable parameters are 318M, and training is performed for 45 and 20 epochs, respectively with LR 1e-4 on a batch size 8 using an Adam [76] optimizer. Each epoch takes approximately 5.5h on a single RTXA6000 GPU with 48GB of memory.

The Whisper model is fine-tuned for 5 epochs with LR 3e-5 on a batch size 2 using AdamW optimizer, with 1.5G total trainable parameters. Epoch takes approximately 9 mins on a single RTXA6000 GPU with 48GB of memory.

[3] https://huggingface.co/facebook/wav2vec2-large-xlsr-53-german
[4] https://huggingface.co/microsoft/wavlm-large
[5] https://huggingface.co/openai/whisper-large-v2

LR is annealed, and the sampling frequency is set to 16 kHz for all the above approaches. More details on training and model parameters can be found in the GitHub repository.

5.2 Speech enhancement training

In this work, we perform speech enhancement using SepFormer [40]– a multi-head attention Transformer-based source separation architecture. It uses a fully learnable masking-based architecture composed of an encoder, a masking network, and a decoder. This enhancement model is trained on 1300h of clean-noisy pairs synthesized from the DNS4 dataset.

It employs an encoder and decoder with 256 convolution filters with kernel size 16, each with stride 8. The masking network has 2 layers of dual-composition block and a chunk length of 250. With each clean-noisy pair fixed at 4s in length, the model is trained in a supervised fashion using scale-invariant SNR (SI-SNR) loss and Adam optimizer with LR of 1.5e-4. We utilize multi-GPU distributed data-parallel (DDP) training scheme to train the network for 50 epochs with a batch size of 4. Each epoch takes approximately 9h on 8× RTXA6000 GPU.

5.3 Training strategies

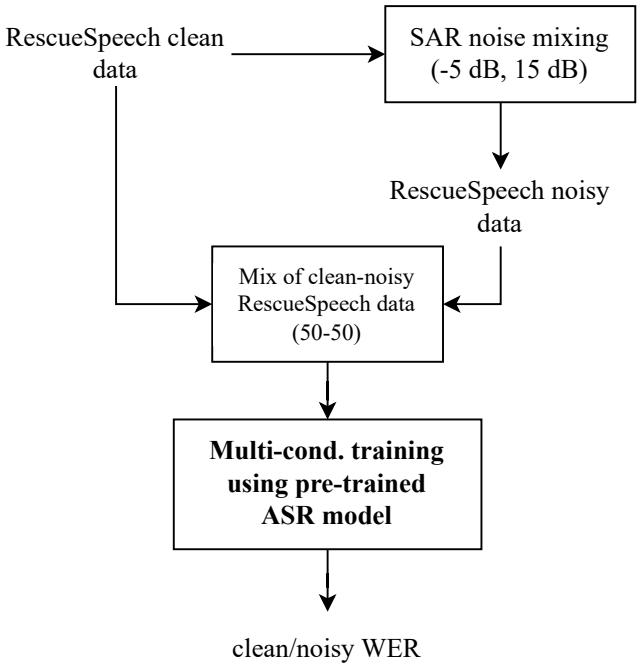

FIGURE 5.1: Training schema for multi-condition training strategy.

We use various training methods to create a robust speech recognition system that operates in the SAR (Search and Rescue) domain. These methods are described below:

1. *Clean training*: After pretraining the ASR and Language Model (LM) models, we fine-tune them on the RescueSpeech clean dataset. This process helps to adapt the models to our target domain. We keep the model and training parameters the same as described in Section 5.1.

2. *Multi-condition training*: Using the same pre-trained model as above, we perform multi-condition training, which involves training the ASR model on an equal mix of clean and noisy audio from the RescueSpeech noisy dataset (see Figure 5.1). By doing this, the model can learn to adapt to different noises present in the utterances, which helps it to perform speech recognition. This method forms the baseline for all our results. We set the learning rate (LR) to 0.1 and keep other parameters the same as above.

3. *Model-combination I: Independent training*: We pre-train a speech enhancement model and then fine-tune it on the RescueSpeech noisy dataset. This model is then integrated with the ASR model trained in the *clean training* stage to perform noise-robust speech recognition. In this stage, we freeze the enhancement model. This has been illustrated in Figure 5.2.

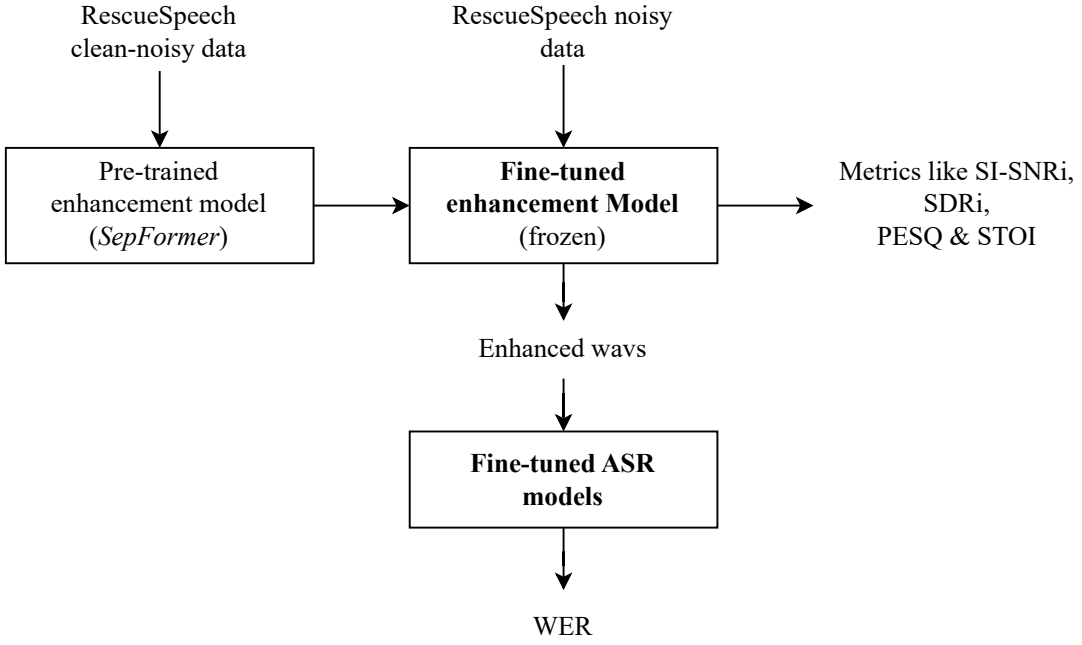

FIGURE 5.2: Training schema for independent model training strategy.

4. *Model-combination II: Joint training*: This is a continuation of the previous stage, where we follow a joint-training approach. We unfreeze the enhancement model and allow gradients from the ASR to propagate back to the speech enhancement model. Updating the

weights of the model in this way enables it to generate output that is as clean as possible, as required by the ASR model. This has been illustrated in Figure 5.3.

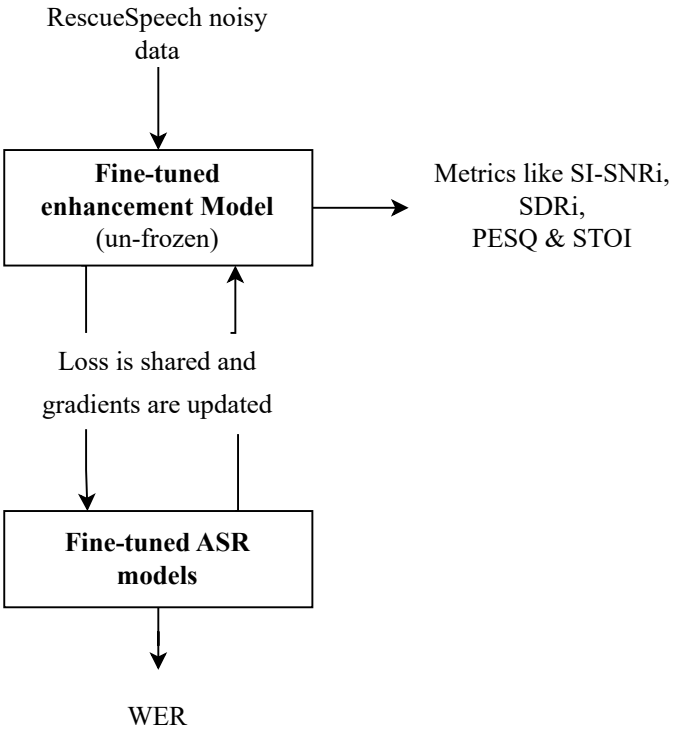

FIGURE 5.3: Training schema for joint model training strategy.

Chapter 6

Results and Discussions

6.1 Pre-training Performance

In the first step, we separately pre-train ASR and speech enhancement models. Our ASR models– CRDNN, wav2vec2.0, and WavLM are pre-trained on full German CommonVoice10.0 corpus. Table 6.1 presents the evaluated performance of these models. CRDNN model performs the best with a WER of 7.92% followed by WavLM with a WER of 8.98%. CRDNN's performance is competitive against WavLM and wav2vec2 even though they have been first pre-trained on thousands of hours of unlabelled data and later fine-tuned on CommonVoice corpus, because CRDNN leverages language model during decoding that captures the statistical patterns and structure of the language and correct errors made by the acoustic model, thus improving the overall accuracy in terms of WER.

Additionally, we pre-train a speech enhancement model, SepFormer, on the DNS4 dataset. Table 6.2 shows evaluation results using DNSMOS on the DNS-4 development set against three metrics– SIG, BAK, and OVRL, with a higher score indicating better quality. In DNS challenge-4, NSNet2 [77] is used as the baseline model. Our model SepFormer performed below the baseline model, failing to improve the perceived quality of the noisy speech signals. This lack of improvement can be attributed to the fact that SepFormer is a large model, and training it with such a large dataset requires a significant amount of computational resources. Despite our initial plan to train it for 150 epochs, we were only able to complete 50 epochs due to resource constraints.

Table 6.1: Comparison of WER on CommonVoice test set for three models: CRDNN, wav2vec2.0-large, WavLM-large at ASR pre-training stage.

ASR Model	**WER**
CRDNN	7.92
Wav2vec2	9.54
WavLM	**8.98**

TABLE 6.2: Evaluation on DNS4 2022 baseline dev set using DNSMOS [53]

Model	**SIG**	**BAK**	**OVRL**
Noisy	2.984	2.560	2.205
NSNet2 [77]	3.014	3.942	2.712
SepFormer	2.999	3.076	2.437

6.2 ASR Performance

As a first attempt to noise robust speech recognition, we created a simple pipeline consisting solely of an ASR model, with no speech enhancement utilized in the front-end. Table 6.3 provides a comparison of different ASR models used on both clean and noisy audio recordings from the RescueSpeech dataset. The models included in the comparison are CRDNN, wav2vec2.0, WavLM, and Whisper. During the pre-training stage, all models (except Whisper) utilized only the CommonVoice dataset. However, during the clean training and multi-condition fine-tuning stage, the RescueSpeech dataset was used.

TABLE 6.3: Comparison of test WERs for CRDNN, wav2vec2.0-large, WavLM-large, and whisper-large-v2 models using different training strategies on clean and noisy speech inputs from the RescueSpeech dataset.

	ASR Model	**clean**	**noisy**
Pre-training	CRDNN	57.05	86.48
	Wav2vec2	50.03	86.45
	WavLM	49.81	83.82
	Whisper	28.41	61.86
Clean training	CRDNN	24.47	59.52
	Wav2vec2	22.16	65.65
	WavLM	**21.67**	61.13
	Whisper	28.39	56.60
Multi-cond. training	CRDNN	27.45	57.95
	Wav2vec2	23.91	60.61
	WavLM	22.48	**55.53**
	Whisper	29.75	62.53

Unsurprisingly, the clean training approach is the most effective when tested on clean audio recordings. The top-performing model in this scenario is WavLM, which achieved a WER of 21.67%. On the other hand, multi-condition training proved to be a superior strategy when

dealing with noisy recordings. In this scenario, the best model is again WavLM, which achieved a WER of 55.53%. The performance gap with clean signals, highlights one more time the significant decline in recognition performance when dealing with challenging acoustic conditions, even for models that were pre-trained using state-of-the-art self-supervised techniques like wav2vec, wavLM, and Whisper (the latter of which is even semi-supervised).

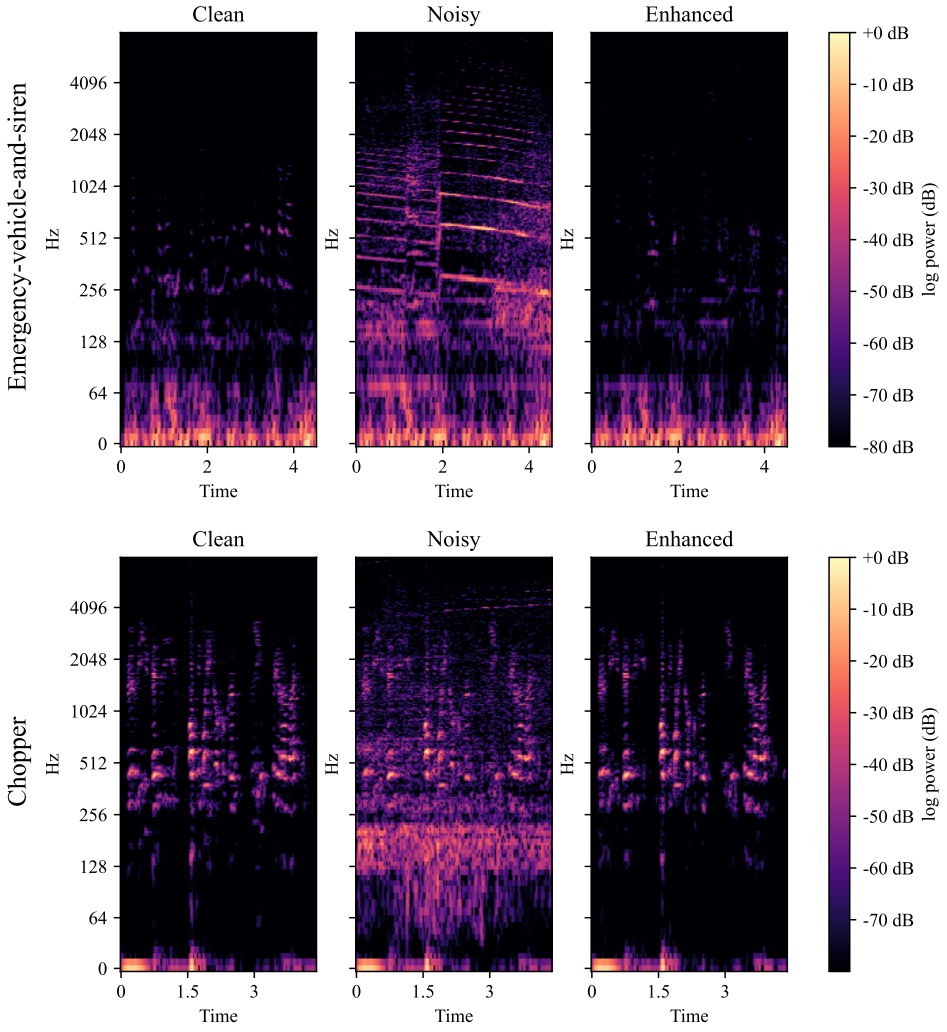

FIGURE 6.1: Log-power spectrogram of clean, noisy, and SepFormer-enhanced utterances for *emergency vehicle siren*, and *chopper* noise types at -5 dB SNR.

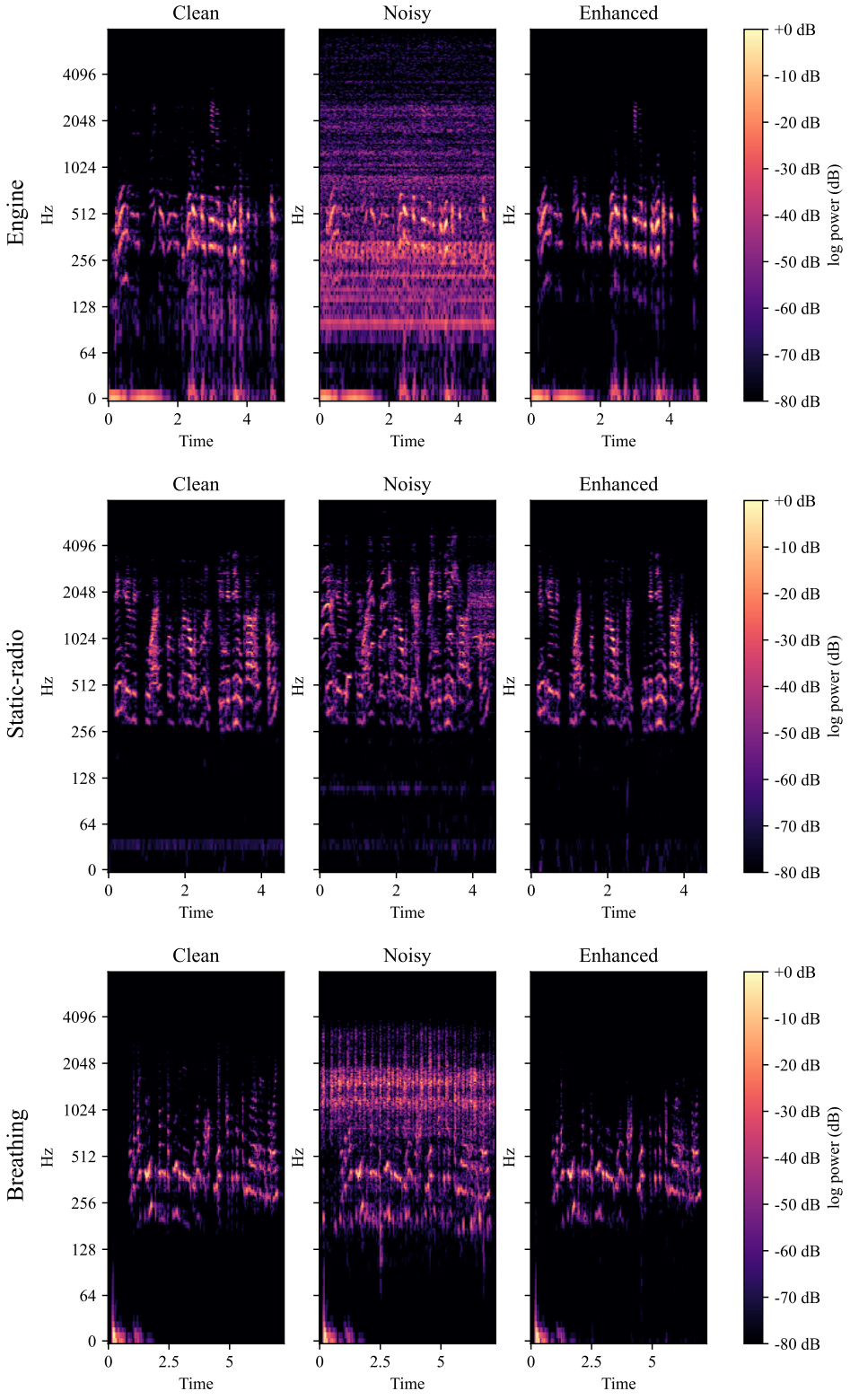

FIGURE 6.2: Log-power spectrogram of clean, noisy, and SepFormer-enhanced utterances for *engine*, *static-radio*, and *breathing* noise types at -5 dB SNR.

6.2.1 Combining ASR and Speech Enhancement

In the further attempts to noise robust speech recognition, our pipeline consisted of a simple combination of a speech enhancement model and a speech recognizer model. Table 6.4 displays the speech enhancement results obtained by incorporating a speech recognizer into the pipeline.

TABLE 6.4: Speech enhancement performance on the RescueSpeech noisy test inputs when combining speech enhancement and speech recognition (Model Comb. I vs Model Comb. II).

	Model Comb. I	**Model Comb. II**			
		CRDNN	wav2vec2	WavLM	Whisper
SI-SNRi	5.624	6.145	5.913	5.959	6.137
SDRi	5.278	5.668	5.465	5.475	5.686
PESQ	2.249	2.304	2.259	2.270	2.296
STOI	0.816	0.823	0.822	0.820	0.822

TABLE 6.5: Word-Error-Rate (WER%) achieved with independent training (Model Comb. I) and joint training (Model Comb. II) of the speech enhancement and ASR modules.

ASR Model	**Model Comb. I**	**Model Comb. II**
CRDNN	56.62	56.02
Wav2vec2	50.39	51.58
WavLM	48.25	50.04
Whisper	**29.97**	33.19

In Section 5.3, we explored two approaches: independent training (Model Comb. I) and joint training (Model Comb. II). The joint training approach resulted in improvements across all considered speech enhancement metrics (SI-SNRi, SDRi, PESQ, STOI) and all ASR modules (CRDNN, Wav2vec2, WavLM, Whisper). Table 6.5 presents the final speech recognition output at the end of the pipeline. Interestingly, a simple combination of the speech enhancement and speech recognition modules performed better than joint training. It is important to note that the speech recognizer is fine-tuned using speech enhanced by the frozen Sepformer. We hypothesize that processing signals from a frozen speech enhancement module makes it easier for the ASR to converge well, given the limited dataset available for fine-tuning. The ASR does not have to continuously adapt to the new speech enhancement output, as in the joint training case. Overall, the best-performing model is the combination of the SepFormer with the Whisper ASR, which achieved a WER of 29.97%.

Figure 6.1, 6.2 shows the log-power spectrogram for noisy audio recordings consisting of *emergency vehicle siren*, *chopper noise*, *engine*, *static-radio*, and *breathing* noise with -5 dB SNR, using the SepFormer model fine-tuned on the RescueSpeech noisy dataset. From a qualitative

standpoint, it appears that SepFormer performs well on noises that impact the SAR domain. Figure 6.3, 6.4 presents PESQ vs SNR and SI-SNRi, SDRi vs SNR for the same noise types. We observed that improvements in SI-SNR and SDR were greater for utterances with an SNR of -5 dB, indicating a more significant enhancement in speech intelligibility and reduction of distortion than for higher SNR utterances. This pattern is consistent across all noise types.

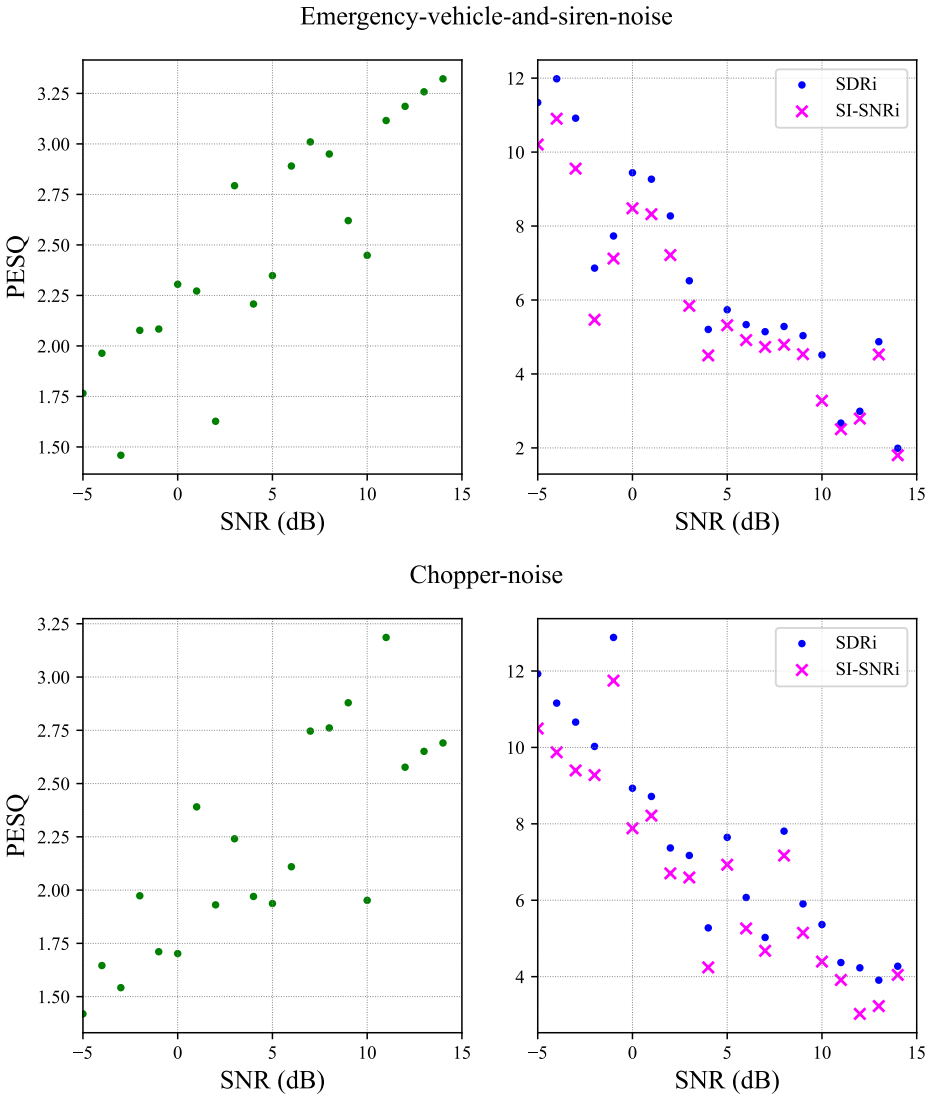

FIGURE 6.3: PESQ, SDRi, SI-SNRi vs SNR of SepFormer enhanced utterances for *emergency vehicle siren*, *chopper*, *engine*, *static-radio*, and *breathing* noise types.

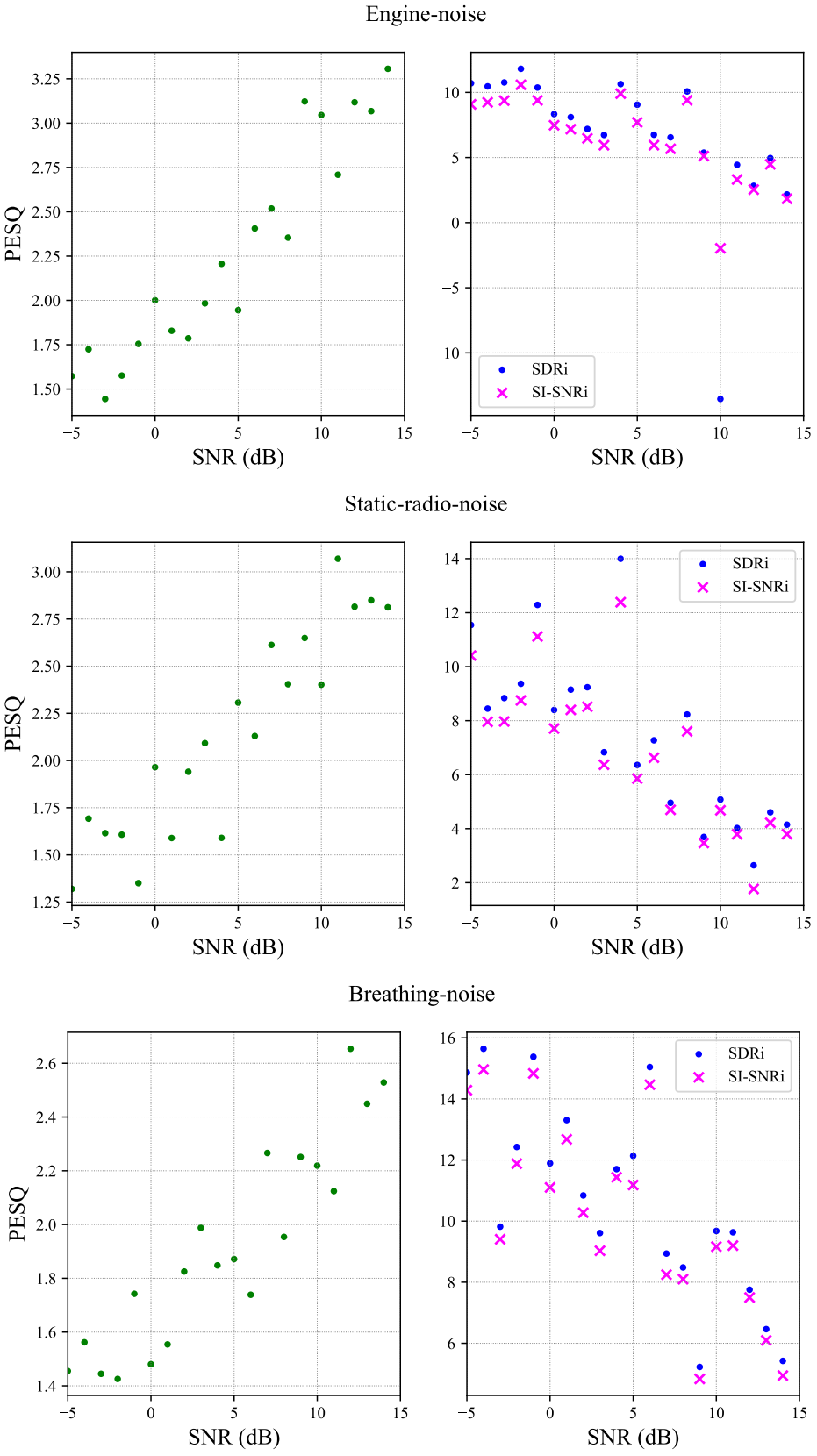

FIGURE 6.4: PESQ, SDRi, SI-SNRi vs SNR of SepFormer enhanced utterances for *engine*, *static-radio*, and *breathing* noise types.

Chapter 7

Conclusion

7.1 General findings

Our work addresses some major challenges that arise in the SAR domain: the lack of speech data, the need for robustness to SAR noises, and conversational speech. To overcome these challenges, we have introduced RescueSpeech, a new dataset of speech data in German that we use to perform robust speech recognition in a hostile noise-filled environment. To achieve this, we proposed multiple training strategies that involve fine-tuning pre-trained models on our in-domain data. We tested different self-supervised models (e.g., Wav2Vec2, WavLM, and Whisper) for speech recognition. Despite leveraging these cutting-edge systems, our best model only achieves a WER of 29.97% on our test set. This result highlights the significant difficulty and the urgent need for further research in this crucial domain.

Overall, our work represents a step forward in addressing the challenges of speech recognition in the SAR domain. By introducing a new dataset, we hope to establish a useful benchmark and foster more studies in this field.

7.2 Future work

In this work, we have attempted to address most of the challenges as discussed. However, in the process of SAR data collection, ASR experiments with the RescueSpeech dataset and difficulties involving speech enhancement with SAR noises, we came across many scopes of future work. We should consider including channel characteristics during ASR training like interference noise, radio noise, and distortion as these affect the quality of the audio signal as it travels from the speaker to the microphone. Speech data pertaining to the SAR domain involves highly emotional speech which our ASR models did not account for, hence, we should include highly emotional speech utterances during our ASR training. Additionally, other than the five SAR noise types discussed in this thesis, there are several noises like foot-stomping, structural

noise due to vibration caused by heavy machinery or chopper blades, and interference noise that need to be addressed when training the speech enhancement model.

We also find that our RescueSpeech dataset is too small in size to make it operational in real SAR scenarios. But we also realise that data collection for such a complex domain comes with various restrictions and difficulties. Therefore, data augmentation techniques can be used to generate more SAR data, and also attempts shall be made to extend the dataset to other languages like English, French, Italian etc.

List of Figures

Figure 1.1 A-DRZ complete system architecture [5] . . . 3
Figure 1.2 A-DRZ: speech processing component [6] . . . 3

Figure 3.1 This illustration [21] explains how CTC determines the conditional probability of an output label for a given input sequence. . . . 14
Figure 3.2 Illustration from [21]– it explains how CTC decoding uses a modified beam search algorithm to determine the most likely output sequence. . . . 15
Figure 3.3 Block diagram of attention based encoder-decoder model [42] . . . 15
Figure 3.4 CRDNN architecture comprising of 2 CNN blocks, 1 RNN block, and 1 dense-neural network (DNN) block, followed by a linear layer and a softmax. . . 17
Figure 3.5 Wav2vec2.0 framework [8] . . . 18
Figure 3.6 Wav2vec2.0 architecture . . . 20
Figure 3.7 A high-level block diagram of SepFoprmer architecture. . . . 21
Figure 3.8 Masking network . . . 22
Figure 3.9 SepFormer block diagram that combines IntraTransformer and InterTransformer to model short-term and long-term dependencies. . . . 22
Figure 3.10 Transformer network used by both IntraT and InterT. . . . 23

Figure 4.1 Histogram plot illustrating the average length of utterances in RescueSpeech in secs. . . . 28

Figure 5.1 Training schema for multi-condition training strategy. . . . 32
Figure 5.2 Training schema for independent model training strategy. . . . 33
Figure 5.3 Training schema for joint model training strategy. . . . 34

Figure 6.1 Log-power spectrogram of clean, noisy, and SepFormer-enhanced utterances for *emergency vehicle siren*, and *chopper* noise types at -5 dB SNR. . . . 37
Figure 6.2 Log-power spectrogram of clean, noisy, and SepFormer-enhanced utterances for *engine*, *static-radio*, and *breathing* noise types at -5 dB SNR. . . . 38
Figure 6.3 PESQ, SDRi, SI-SNRi vs SNR of SepFormer enhanced utterances for *emergency vehicle siren*, *chopper*, *engine*, *static-radio*, and *breathing* noise types. 40

Figure 6.4 PESQ, SDRi, SI-SNRi vs SNR of SepFormer enhanced utterances for *engine*, *static-radio*, and *breathing* noise types. 41

List of Tables

Table 4.1 Distribution of sentences and hours in the German CommonVoice10.0 and DNS4 dataset. 27
Table 4.2 Distribution of sentences and hours in the RescueSpeech clean and noisy dataset. 28

Table 6.1 Comparison of WER on CommonVoice test set for three models: CRDNN, wav2vec2.0-large, WavLM-large at ASR pre-training stage. 35
Table 6.2 Evaluation on DNS4 2022 baseline dev set using DNSMOS [53] 36
Table 6.3 Comparison of test WERs for CRDNN, wav2vec2.0-large, WavLM-large, and whisper-large-v2 models using different training strategies on clean and noisy speech inputs from the RescueSpeech dataset. 36
Table 6.4 Speech enhancement performance on the RescueSpeech noisy test inputs when combining speech enhancement and speech recognition (Model Comb. I vs Model Comb. II). 39
Table 6.5 Word-Error-Rate (WER%) achieved with independent training (Model Comb. I) and joint training (Model Comb. II) of the speech enhancement and ASR modules. 39

Table A.1 Main hyperparameters used in the reported Common Voice experiments. . 55
Table A.2 Main hyperparameters used in the reported DNS4 experiment. 56

Bibliography

[1] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus, 2019. URL https://arxiv.org/abs/1912.06670.

[2] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015. doi: 10.1109/ICASSP.2015.7178964.

[3] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman. Voxceleb: Large-scale speaker verification in the wild. *Computer Science and Language*, 2019.

[4] Dominik Macháček, Jonáš Kratochvíl, Sangeet Sagar, Matúš Žilinec, Ondřej Bojar, Thai-Son Nguyen, Felix Schneider, Philip Williams, and Yuekun Yao. ELITR non-native speech translation at IWSLT 2020. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 200–208, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.iwslt-1.25. URL https://aclanthology.org/2020.iwslt-1.25.

[5] Ivana Kruijff-Korbayová, Robert Grafe, Nils Heidemann, Alexander Berrang, Cai Hussung, Christian Willms, Peter Fettke, Marius Beul, Jan Quenzel, Daniel Schleich, Sven Behnke, Janis Tiemann, Johannes Güldenring, Manuel Patchou, Christian Arendt, Christian Wietfeld, Kevin Daun, Marius Schnaubelt, Oskar von Stryk, Alexander Lel, Alexander Miller, Christof Röhrig, Thomas Straßmann, Thomas Barz, Stefan Soltau, Felix Kremer, Stefan Rilling, Rohan Haseloff, Stefan Grobelny, Artur Leinweber, Gerhard Senkowski, Marc Thurow, Dominik Slomma, and Hartmut Surmann. German rescue robotics center (drz): A holistic approach for robotic systems assisting in emergency response. In *2021 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, page 138–145. IEEE Press, 2021. doi: 10.1109/SSRR53300.2021.9597869. URL https://doi.org/10.1109/SSRR53300.2021.9597869.

[6] Christian Willms, Constantin Houy, Jana-Rebecca Rehse, Peter Fettke, and Ivana Kruijff-Korbayová. Team communication processing and process analytics for supporting robot-assisted emergency response. In *2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, page 216–221. IEEE Press, 2019. doi: 10.1109/SSRR.2019.8848976. URL https://doi.org/10.1109/SSRR.2019.8848976.

[7] Aashish Agarwal and Torsten Zesch. German end-to-end speech recognition based on deepspeech. In *Preliminary proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers*, pages 111–119, Erlangen, Germany, 2019. German Society for Computational Linguistics & Language Technology.

[8] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations, 2020. URL https://arxiv.org/abs/2006.11477.

[9] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei. WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, oct 2022.

[10] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust Speech Recognition via Large-Scale Weak Supervision, 2022. URL https://arxiv.org/abs/2212.04356.

[11] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. doi: 10.1109/5.18626.

[12] E. Ambikairajah and S. Lennon. Neural networks for speech recognition. In Michael F. McTear and Norman Creaney, editors, *AI and Cognitive Science '90*, pages 163–177, London, 1991. Springer London. ISBN 978-1-4471-3542-5.

[13] Richard P. Lippmann. Review of Neural Networks for Speech Recognition. *Neural Computation*, 1(1):1–38, 03 1989. ISSN 0899-7667. doi: 10.1162/neco.1989.1.1.1. URL https://doi.org/10.1162/neco.1989.1.1.1.

[14] Mark Gales and Steve Young. now, 2008. URL https://ieeexplore.ieee.org/document/8187420.

[15] P. Pujol, S. Pol, C. Nadeu, A. Hagen, and H. Bourlard. Comparison and combination of features in a hybrid hmm/mlp and a hmm/gmm speech recognition system. *IEEE Transactions on Speech and Audio Processing*, 13(1):14–22, Jan 2005. ISSN 1558-2353. doi: 10.1109/TSA.2004.834466.

[16] G.D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, March 1973. ISSN 1558-2256. doi: 10.1109/PROC.1973.9030.

[17] Dong Yu and Li Deng. *Automatic Speech Recognition: A Deep Learning Approach*. Springer Publishing Company, Incorporated, 2014. ISBN 1447157788.

[18] Longfei Li, Yong Zhao, Dongmei Jiang, Yanning Zhang, Fengna Wang, Isabel Gonzalez, Enescu Valentin, and Hichem Sahli. Hybrid deep neural network–hidden markov model (dnn-hmm) based speech emotion recognition. In *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pages 312–317, 2013. doi: 10.1109/ACII.2013.58.

[19] Geoffrey Hinton, Li Deng, Dong Yu, George E. Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N. Sainath, and Brian Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. doi: 10.1109/MSP.2012.2205597.

[20] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143891. URL https://doi.org/10.1145/1143844.1143891.

[21] Awni Hannun. Sequence modeling with ctc. *Distill*, 2017. doi: 10.23915/distill.00008. URL https://distill.pub/2017/ctc.

[22] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page II–1764–II–1772. JMLR.org, 2014.

[23] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition, 2014. URL https://arxiv.org/abs/1412.5567.

[24] Hagen Soltau, Hank Liao, and Hasim Sak. Neural speech recognizer: Acoustic-to-word lstm model for large vocabulary speech recognition, 2016. URL https://arxiv.org/abs/1610.09975.

[25] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. End-to-end continuous speech recognition using attention-based recurrent nn: First results, 2014. URL https://arxiv.org/abs/1412.1602.

[26] William Chan, Navdeep Jaitly, Quoc Le, and Oriol 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)*, page 4960–4964.

IEEE Press, 2016. doi: 10.1109/ICASSP.2016.7472621. URL https://doi.org/10.1109/ICASSP.2016.7472621.

[27] Rohit Prabhavalkar, Kanishka Rao, Tara N. Sainath, Bo Li, Leif Johnson, and Navdeep Jaitly. A Comparison of Sequence-to-Sequence Models for Speech Recognition. In *Proc. Interspeech 2017*, pages 939–943, 2017. doi: 10.21437/Interspeech.2017-233.

[28] Alex Graves. Sequence transduction with recurrent neural networks, 2012. URL https://arxiv.org/abs/1211.3711.

[29] Jinyu Li, Rui Zhao, Hu Hu, and Yifan Gong. Improving rnn transducer modeling for end-to-end speech recognition. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 114–121, 2019. doi: 10.1109/ASRU46091.2019.9003906.

[30] Bo Li, Shuo-yiin Chang, Tara N. Sainath, Ruoming Pang, Yanzhang He, Trevor Strohman, and Yonghui Wu. Towards fast and accurate streaming end-to-end asr, 2020. URL https://arxiv.org/abs/2004.11544.

[31] S. Boll. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 27(2):113–120, 1979. doi: 10.1109/TASSP.1979.1163209.

[32] B. Widrow, J.R. Glover, J.M. McCool, J. Kaunitz, C.S. Williams, R.H. Hearn, J.R. Zeidler, Jr. Eugene Dong, and R.C. Goodlin. Adaptive noise cancelling: Principles and applications. *Proceedings of the IEEE*, 63(12):1692–1716, 1975. doi: 10.1109/PROC.1975.10036.

[33] K. Paliwal and A. Basu. A speech enhancement method based on kalman filtering. In *ICASSP '87. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 12, pages 177–180, 1987. doi: 10.1109/ICASSP.1987.1169756.

[34] Y. Ephraim and D. Malah. Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(6):1109–1121, 1984. doi: 10.1109/TASSP.1984.1164453.

[35] Szu-Wei Fu, Chien-Feng Liao, Yu Tsao, and Shou-De Lin. Metricgan: Generative adversarial networks based black-box metric scores optimization for speech enhancement, 2019. URL https://arxiv.org/abs/1905.04874.

[36] Szu-Wei Fu, Cheng Yu, Kuo-Hsuan Hung, Mirco Ravanelli, and Yu Tsao. Metricgan-u: Unsupervised speech enhancement/ dereverberation based only on noisy/ reverberated speech, 2021. URL https://arxiv.org/abs/2110.05866.

[37] Szu-Wei Fu, Cheng Yu, Tsun-An Hsieh, Peter Plantinga, Mirco Ravanelli, Xugang Lu, and Yu Tsao. Metricgan+: An improved version of metricgan for speech enhancement, 2021. URL https://arxiv.org/abs/2104.03538.

[38] Deblin Bagchi, Peter Plantinga, Adam Stiff, and Eric Fosler-Lussier. Spectral feature mapping with mimic loss for robust speech recognition, 2018. URL https://arxiv.org/abs/1803.09816.

[39] Peter Plantinga, Deblin Bagchi, and Eric Fosler-Lussier. Phonetic feedback for speech enhancement with and without parallel speech data, 2020. URL https://arxiv.org/abs/2003.01769.

[40] Cem Subakan, Mirco Ravanelli, Samuele Cornell, Mirko Bronzi, and Jianyuan Zhong. Attention is all you need in speech separation, 2020. URL https://arxiv.org/abs/2010.13154.

[41] Mirco Ravanelli, Titouan Parcollet, Peter Plantinga, Aku Rouhe, Samuele Cornell, Loren Lugosch, Cem Subakan, Nauman Dawalatabad, Abdelwahab Heba, Jianyuan Zhong, Ju-Chieh Chou, Sung-Lin Yeh, Szu-Wei Fu, Chien-Feng Liao, Elena Rastorgueva, François Grondin, William Aris, Hwidong Na, Yan Gao, Renato De Mori, and Yoshua Bengio. Speechbrain: A general-purpose speech toolkit, 2021. URL https://arxiv.org/abs/2106.04624.

[42] Jinyu Li. Recent advances in end-to-end automatic speech recognition, 2021. URL https://arxiv.org/abs/2111.01690.

[43] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition, 2015. URL https://arxiv.org/abs/1506.07503.

[44] Tara N. Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4580–4584, 2015. doi: 10.1109/ICASSP.2015.7178838.

[45] Yusheng Xiang, Tian Tang, Tianqing Su, Christine Brach, Libo Liu, Samuel S. Mao, and Marcus Geimer. Fast crdnn: Towards on site training of mobile construction machines. *IEEE Access*, 9:124253–124267, 2021. doi: 10.1109/ACCESS.2021.3110288.

[46] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. wav2vec: Unsupervised pre-training for speech recognition, 2019. URL https://arxiv.org/abs/1904.05862.

[47] Yi Luo, Zhuo Chen, and Takuya Yoshioka. Dual-path rnn: efficient long sequence modeling for time-domain single-channel speech separation, 2019. URL https://arxiv.org/abs/1910.06379.

[48] Yusuf Isik, Jonathan Le Roux, Zhuo Chen, Shinji Watanabe, and John R. Hershey. Single-channel multi-speaker separation using deep clustering, 2016. URL https://arxiv.org/abs/1607.02173.

[49] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R. Hershey. Sdr - half-baked or well done?, 2018. URL https://arxiv.org/abs/1811.02508.

[50] A.W. Rix, J.G. Beerends, M.P. Hollier, and A.P. Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, volume 2, pages 749–752 vol.2, 2001. doi: 10.1109/ICASSP.2001.941023.

[51] Rafael G. Dantas Miao Wang, Christoph Boeddeker and ananda seelan. Pesq (perceptual evaluation of speech quality) wrapper for python users, May 2022. URL https://doi.org/10.5281/zenodo.6549559.

[52] Cees H. Taal, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen. A short-time objective intelligibility measure for time-frequency weighted noisy speech. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4214–4217, 2010. doi: 10.1109/ICASSP.2010.5495701.

[53] Chandan K A Reddy, Vishak Gopal, and Ross Cutler. Dnsmos: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors, 2020. URL https://arxiv.org/abs/2010.15258.

[54] Chandan K A Reddy, Vishak Gopal, and Ross Cutler. Dnsmos p.835: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors, 2021. URL https://arxiv.org/abs/2110.01763.

[55] Babak Naderi and Ross Cutler. An open source implementation of ITU-t recommendation p.808 with validation. In *Interspeech 2020*. ISCA, oct 2020. doi: 10.21437/interspeech.2020-2665. URL https://doi.org/10.21437%2Finterspeech.2020-2665.

[56] Jon Barker, Ricard Marxer, Emmanuel Vincent, and Shinji Watanabe. The Third CHiME Speech Separation and Recognition Challenge. *Comput. Speech Lang.*, 46(C):605–626, nov 2017. ISSN 0885-2308. doi: 10.1016/j.csl.2016.10.005. URL https://doi.org/10.1016/j.csl.2016.10.005.

[57] E. Vincent, S. Watanabe, A. Nugraha, J. Barker, and R. Marxer. An analysis of environment, microphone and data simulation mismatches in robust speech recognition. *Computer Speech and Language*, 46:535–557, 2017.

[58] Jon Barker, Shinji Watanabe, Emmanuel Vincent, and Jan Trmal. The fifth 'CHiME' Speech Separation and Recognition Challenge: Dataset, task and baselines. In *Proc. of Interspeech*, 2018.

[59] Shinji Watanabe et al. CHiME-6 Challenge: Tackling Multispeaker Speech Recognition for Unsegmented Recordings. In *Proc. 6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020)*, 2020.

[60] Mirco Ravanelli, Luca Cristoforetti, Roberto Gretter, Marco Pellin, Alessandro Sosi, and Maurizio Omologo. The DIRHA-ENGLISH corpus and related tasks for distant-speech recognition in domestic environments. In *Proc. of ASRU*, 2015.

[61] Marco Matassoni, Ramón Fernandez Astudillo, Athanasios Katsamanis, and Mirco Ravanelli. The DIRHA-GRID corpus: baseline and tools for multi-room distant speech recognition using distributed microphones. In *Proc. of Interspeech*, 2014.

[62] Mirco Ravanelli and Maurizio Omologo. On the selection of the impulse responses for distant-speech recognition based on contaminated speech training. In Haizhou Li, Helen M. Meng, Bin Ma, Engsiong Chng, and Lei Xie, editors, *Proc. of Interspeech*, 2014.

[63] Mirco Ravanelli and Maurizio Omologo. Contaminated speech training methods for robust DNN-HMM distant speech recognition. In *Proc. of Interspeech*, 2015.

[64] Steve Renals, Thomas Hain, and Herve Bourlard. Recognition and interpretation of meetings: The AMI and AMIDA projects. In *Proc. of ASRU*, 2007.

[65] Colleen Richey, Maria A. Barrios, Zeb Armstrong, Chris Bartels, Horacio Franco, Martin Graciarena, Aaron Lawson, Mahesh Kumar Nandwana, Allen Stauffer, Julien van Hout, Paul Gamble, Jeff Hetherly, Cory Stephenson, and Karl Ni. Voices Obscured in Complex Environmental Settings (VOICES) corpus, 2018.

[66] Alex Stupakov, Evan Hanusa, Deepak Vijaywargi, Dieter Fox, and Jeff A. Bilmes. The design and collection of COSINE, a multi-microphone in situ speech corpus recorded in noisy environments. *Comput. Speech Lang.*, 26(1):52–66, 2012.

[67] Harishchandra Dubey, Vishak Gopal, Ross Cutler, Ashkan Aazami, Sergiy Matusevych, Sebastian Braun, Sefik Emre Eskimez, Manthan Thakker, Takuya Yoshioka, Hannes Gamper, and Robert Aichner. ICASSP 2022 Deep Noise Suppression Challenge, 2022. URL https://arxiv.org/abs/2202.13288.

[68] Christophe Veaux, Junichi Yamagishi, and Simon King. The voice bank corpus: Design, collection and data analysis of a large regional accent speech database. In *2013 International Conference Oriental COCOSDA held jointly with 2013 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*, pages 1–4, 2013. doi: 10.1109/ICSDA.2013.6709856.

[69] Gordon Wichern, Joe Antognini, Michael Flynn, Licheng Richard Zhu, Emmett McQuinn, Dwight Crow, Ethan Manilow, and Jonathan Le Roux. WHAM!: Extending Speech Separation to Noisy Environments. In *Proc. Interspeech*, September 2019.

[70] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio Set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780, 2017. doi: 10.1109/ICASSP.2017.7952261.

[71] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L. Seltzer, and Sanjeev Khudanpur. A study on data augmentation of reverberant speech for robust speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5220–5224, 2017. doi: 10.1109/ICASSP.2017.7953152.

[72] Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012.

[73] Benjamin Milde and Arne Koehn. Open Source Automatic Speech Recognition for German. In *Speech Communication; 13th ITG-Symposium*, pages 1–5, 2018.

[74] Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 759–765, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/327_Paper.pdf.

[75] Matthew D. Zeiler. ADADELTA: An Adaptive Learning Rate Method, 2012. URL https://arxiv.org/abs/1212.5701.

[76] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, 2014. URL https://arxiv.org/abs/1412.6980.

[77] Harishchandra Dubey, Vishak Gopal, Ross Cutler, Ashkan Aazami, Sergiy Matusevych, Sebastian Braun, Sefik Emre Eskimez, Manthan Thakker, Takuya Yoshioka, Hannes Gamper, and Robert Aichner. ICASSP 2022 Deep Noise Suppression Challenge, 2022. URL https://arxiv.org/abs/2202.13288.

Appendix A

Table A.1: Main hyperparameters used in the reported Common Voice experiments.

Task	**Dataset**	**Technique**	**Experimental Setting**
Speech recognition	Common Voice10.0	CRDNN + seq2seq	Encoder: CRDNN (2 CNNs, 4 Bi-LSTM, 2 DNN layers) Features: 40 fbanks Augmentation: Yes Pretraining: no CTC weight: 0.5 Dropout: 0.15 (for both encoder and decoder) Batchnorm: yes Number of epochs: 25 Batch size: 8 Learning rate: 1.0 LR scheduler: new bob LR annealing factor: 0.8 Optimizer: Adadelta Loss: CTC+NLL Loss Token type: unigrams Number of tokens: 1000 Decoder: Attn-GRU (1 layer) + Beam search (Decoding) Beam size: 80 (Decoding) LM weight: 0.50 (Decoding) CTC weight: 0.0 LM: RNNLM (2 RNN layers, 1 DNN layer) Training Time: 8h/epoch (RTXA6000-48GB)
Speech recognition	Common Voice10.0	Wac2vec2.0/ WavLM + CTC	Encoder: Wav2vec2.0 /WavLM (Transformer) Decoder: Greedy decoder Augmentation: Yes Pretraining (wav2vec): wav2vec2-large-xlsr-53-german Pretraining (wavlm): wavlm-large Dropout: 0.15 (for both encoder and decoder) Batchnorm: yes Number of epochs: 45 Batch size: 8 Learning rate: 1.0 Learning rate wav2vec: 0.0001 LR scheduler: new bob LR annealing factor: 0.9 Optimizer: Adadelta Loss: CTC+NLL Loss Token type: char Number of tokens: 32 Training Time: 5h 35 min/epoch (RTXA6000-48GB)

Table A.2: Main hyperparameters used in the reported DNS4 experiment.

Task	**Dataset**	**Technique**	**Experimental Setting**
Speech enhancement	DNS4	SepFormer	Model: SepFormer (Encoder, MaskNet, Decoder) Sample rate: 16K Encoder: DualPath CNN MaskNet: DualPath Model (2 layers) Decoder: ConvTranspose1d Mixed precision: True Epochs: 150 Batch size: 4 Learning rate: 0.00015 Augmentation: Yes (only speedperturb) Pretraining: no Dropout: 0.0 Normalization: LayerNorm Optimizer: Adam LR scheduler: ReduceLROnPlateau Loss: SI-SNR (with pit wrapper) Training Time: 9h/epoch (8 × RTXA6000-48GB)