University of Nottingham
UK | CHINA | MALAYSIA

Self-Supervised Learning for Automatic Speech Recognition In Low-Resource Environments

Thesis submitted to the University of Nottingham for the degree of **Doctor of Philosophy, October 2023.**

Kavan Fatehi

20167617

Supervised by

Dr. Ayse Kucukyilmaz
Professor Andrew French

Signature _____

Date _____ / _____ / _____

Abstract

Supervised deep neural networks trained with substantial amounts of annotated speech data have demonstrated impressive performance across a spectrum of spoken language processing applications, frequently establishing themselves as the leading models in respective competitions. Nonetheless, a significant challenge arises from the heavy reliance on extensive annotated data for training these systems. This reliance poses a significant scalability limitation, hindering the continual enhancement of state-of-the-art performance. Moreover, it presents a more fundamental obstacle for deploying deep neural networks in speech-related domains where acquiring labeled data is inherently arduous, expensive, or time-intensive, which are considered as low-resource ASR problems in this thesis.

Unlike annotated speech data, collecting untranscribed audio is typically more cost-effective. In this thesis, we investigate the application of self-supervised learning in low-resource tasks, a learning approach where the learning objective is derived directly from the input data itself. We employ this method to harness the scalability and affordability of untranscribed audio resources in problems where we do not have enough training data, with the goal of enhancing the performance of spoken language technology. In particular, we propose three self-supervised methodologies. One model is based on the concept of *two-fine-tuning steps*, while the other two

revolve around the notion of *identifying an improved hidden unit*. These approaches are designed to learn contextualized speech representations from speech data lacking annotations. We demonstrate the capacity of our self-supervised techniques to learn representations that convert the higher-level characteristics of speech signals more effectively than conventional acoustic features. Additionally, we present how these representations enhance the performance of deep neural networks on ASR tasks with limited resources. Beyond introducing novel learning algorithms, we conduct in-depth analyses to comprehend the properties of the acquired self-supervised representations and elucidate the distinct design elements that separate one self-supervised model from another.

Acknowledgements

First and foremost, I would like to extend my deepest gratitude to Dr. Ayse Kucukyilmaz and Prof. Andrew French. Their continuous support, invaluable guidance, and insightful feedback have been instrumental in shaping this research. Their dedication and commitment to excellence have always inspired me to push boundaries and strive for excellence.

I am also immensely grateful to my former supervisor, Dr. Mercedes Torres Torres. Her foundational teachings and mentorship during the initial stages of my doctoral journey laid a strong groundwork upon which I built my research.

I would also like to thank my friends, Amir Shirian, Seyed Amir Hosseini and Erfan Loweimi for their support and collaboration. I have learned so much from all of you, and I am grateful for the opportunity to have such a vibrant and supportive friends.

On a personal note, special thanks are reserved for my wife, who has been my constant pillar of strength. Her unwavering support, understanding, and encouragement have been the backbone of this endeavor. Her belief in me, even during challenging times, was a source of inspiration and provided the motivation to persevere.

Lastly, to all who have directly or indirectly contributed to this journey, my sincere appreciation for your influence and for making this achievement possible.

Contents

List of Tables

List of Figures

Abbreviations

AEs Autoencoders.

AM Acoustic Model.

APC Autoregressive Predictive Coding.

ASR Automatic Speech Recognition.

BERT Bidirectional Encoder Representations from Transformers.

CC Contrastive Clustering.

CCA Canonical Correlation Analysis.

CCH Cluster-Level Contrastive Head.

CER Character Error Rate.

cltLSTM contextual layer trajectory LSTM.

CNN Convolutional Neural Network.

CNNs Convolutional Neural Networks.

CTC Connectionist Temporal Classification.

DNNs Deep Neural Networks.

E2E End-to-End.

FNN Feedforward Neural Networks.

FSMN feed-forward sequential memory networks.

GMM Gaussian Mixture Model.

HMM Hidden Markov Model.

HMM/DNN Hidden Markov Model/Deep Neural Network.

HRE High-Resource Environment.

ICH Instance-Level Contrastive Head.

LA Local Aggregation.

LABERT Local Aggregation BERT.

LCBLSTM latency-controlled BLSTM.

LM language model.

LM Recurrent Neural Network LM.

LPC Linear Predictive Coding.

LREs Low-resource Environments.

ltLSTM Layer-Trajectory Long Short-Term Memory.

LVCSR Large Vocabulary Continuous Speech Recognition.

MAML Model-Agnostic Meta Learning.

MFCCs Mel-frequency Cepstral Coefficients.

MLM Masked Language Modeling.

NCE Noise Contrastive Estimation.

NLP natural language processing.

NPC non-autoregressive predictive coding.

OOV out-of-vocabulary.

PCB Pair Construction Backbone.

PER Phone Error Rate.

PNMI Phone-Normalized Mutual Information.

RBM Restricted Boltzmann Machines.

RCCBERT Regularized Contrastive Clustering BERT.

RNNs Recurrent Neural Networks.

SID Speaker Identification.

SSL Self-Supervised Learning.

TDNN Time-delay neural networks.

TDS Time-Depth Separable.

VAE Variational Autoencoders.

VICReg Variance-Invariance-Covariance Regularization.

VQ-VAE Vector-Quantized Variational Autoencoders.

WER Word Error Rate.

WSJ Wall Street Journal.

Chapter 1

Introduction

Speech, being the most common form of human communication [1], has motivated both computer scientists and linguists to develop machines, which can effectively communicate with humans.This has led to the creation of Automatic Speech Recognition (ASR) systems [2]. ASR systems have played an important role in advancing the field of machine learning [3]. Today, personal assistants on smartphones, such as Apple's Siri, or voice-activated services, such Amazon's Echo, are examples of ASR systems already used for everyday tasks [4].

Speech and Spoken Language Technology cover a broad range of functions. These include ASR systems, which convert speech into text, and speech synthesis, which does the opposite by turning text into speech [5, 6]. Additionally, they facilitate speaker identification, emotion classification and affect recognition [2]. For most of these tasks, the ultimate goal remains the same: to have seamless and natural communication between humans and machines based on speech [1]. The past decade has seen a rapid development of deep learning methods in machine learning. The application of deep learning in ASR has resulted in more robust and accurate models, together

with a wide range of different applications [7, 8, 9]. By employing complex neural network architectures, such as Convolutional Neural Networks (CNNs) [10, 1] and Recurrent Neural Networks (RNNs) [11, 12, 13, 14], deep learning algorithms can effectively process and interpret vast amounts of spoken language data [15]. This has enabled ASR systems to better understand and transcribe speech with nuances in accent, tone, and context, significantly reducing error rates [16, 17, 18]. Additionally, deep learning has empowered ASR systems to adapt to new languages and dialects more quickly [19].

State-of-the-art ASR systems achieve Word Error Rate (WER) as low as 9.34% [20] with two-headed cltLSTM, which is trained with transcribed data from a variety of Microsoft products, and 3.6% [21] with Conv-Transformer Transducer on Librispeech clean data, while some other models in the area obtain a WER of 9.92% [22], and 10.92% [23]. In ASR systems research area, the advent of large datasets has undeniably crucial role in the advancement of the state-of-the-art ASR models. However, it is crucial to consider that the achievement of significantly low WERs is not only attributable to the availability of these extensive datasets. Indeed, Transformer-based models, the application of data augmentations techniques and the improvment of the language models have all significantly contribute to the reduction of the WERs in ASR system algorithms. Two-headed cltLSTM approach [20] which uses a training set of 65k hours of speech data and cltLSTM and Conv-Transformer Transducer[21] models that respectively use training sets of more than 30k hours and more than 1k hours of data are examples that link the amount of training data to the reduction of the WERs. The Neural Speech Recognizer model [24] uses 125k hours of YouTube videos to obtain a WER of 13.5%, and Deep Speech 2 [1] obtains a WER of 13.59% by using almost 12k hours of English speech and 9.4k

hours of Mandarin Chinese speech to train. Therefore, in addition to the amount of training data, in-domain data is also very important in model learning. Using in-domain data can help to ensure that the model learns the most relevant and important patterns from the data, which makes the training process more efficient.

The requirement for extensive datasets to effectively train deep learning-based ASR systems is a consistent characteristic across all state-of-the-art methodologies. In fact, the availability of larger datasets generally correlates with enhanced performance in contemporary deep learning ASR systems [4]. For example, the amount of data in the TED-LIUM 3 dataset [25], which was released in 2018, more than doubled in comparison to the amount of data from the previous release, TED-LIUM 2 [26], released in 2014. Consequently, the proposed model trained with TED-LIUM 3 was able to achieve better results [25]: By using a classical 3-gram language model used in a beam search on top of the end-to-end architecture, WER decreases to 13.7% with the TED-LIUM 3 training data, while with the TED-LIUM 2 training data, the same model reached a WER of only 20.3% [25]. An end-to-end ASR architecture is a model that directly transforms input speech sequences into output label sequences, without the need for any intermediate phonetic or linguistic representation [26]. So, in end-to-end ASR models, an increased volume of training data enhances the model's ability to learn diverse linguistic patterns and acoustic variations, thereby improving its accuracy and robustness in real-world speech recognition tasks. Furthermore, in [27], the authors achieved an improvement in ASR performance by training the model over LibriSpeech dataset, which contains over 1k hours of speech. The positive effect of larger training datasets in ASR model's performance is also observable with the VoxCeleb2 dataset [28]. By increasing the number of sentences and participants com-

pared to the previous version of VoxCeleb [29], the models trained with VoxCeleb2 outperformed the same models previously trained with VoxCeleb.

The remarkable performance achieved by these systems, paired with the extensive training data, lead to an intriguing machine learning question: What occurs when the training dataset is restricted? For instance, what if the application environment for these ASR systems is highly specialized, making the usual available corpora like LibriSpeech and WSJ dataset [30] irrelevant? What if collecting substantial amounts of data proves to be challenging?

Low-resource Environments (LREs) refer to situations where are characterized by a scarcity of available training data, that presents various challenges in the development of efficient ASR models. This limitation impacts the ability to train and refine models effectively, due to the constrained data [31]. These may include scenarios with noise like environmental noise, channel noise or speaker-related noise, restricted speaker vocabulary [32], or specialized vocabulary requirements [33]. ASR systems face significant limitations in under-resourced languages and specific domains, as current state-of-the-art models struggle to generate highly accurate output sequences due to the insufficiently large training datasets. The primary hurdle lies in the inadequacy of acoustic and textual data in these under-resourced or low-resource domains [4].

Additional examples of LREs are related to the specific domain of the ASR system. These domains can be characterized as highly specialized ASR tasks, such as ASR systems designed for children [34] or the accented speech recognition task [35]. Researchers have not yet treated domain-specific or domain adaptation in ASR systems in much detail. State-of-

the-art ASR models use popular benchmark datasets, such as Librispeech [27], WSJ [30], Fisher [36], and Switchboard [37] which contain public speech data [6]. The nature of the data has a significant impact on the vocabulary and the form of the conversations. A domain adaptation in the ASR system can be seen in [38], which has shown that the amount of in-domain data has a direct effect on the accuracy of ASR models when working with conversational topics spoken by people with dysarthria, and corroborated the importance of more inclusive systems. Previous state-of-the-art techniques of ASR systems lack focus on non-native language speakers, so acoustics and linguistics are not considered in the evaluation of the systems [39]. Our experiments show that general benchmark datasets are insufficient for specialized LREs.

Current speech recognition systems require massive amounts of labeled data to train, but most real-world applications do not have much. This is the aims of our works to develop new methods to train speech recognition systems without the need for large amounts of labeled data, making them more accessible and practical for a wider range of applications. To develop an ASR system for a specific application, a substantial amount of speech data with transcriptions (known as a spoken corpus) is necessary. However, in many cases, such a corpus suitable for the target application is not available. The primary challenge arises from the fact that the task-related data required for training the LRE ASR is not adequately represented in the existing public training set. Often, researchers and practitioners try to overcome this challenge by collecting as much data as possible from similar conditions. However, this approach does not fully address the problem, as the uniqueness of each LRE's linguistic and acoustic properties means that even a large, but only somewhat relevant dataset may not be sufficient.

Utilizing data specific to the target domain is optimal for enhancing our

models. However, obtaining such data is challenging, as discussed earlier. The usual method for developing a new application involves the creation of a new dataset by recording and transcribing audio. The issue lies in the impracticality of recording a large volume of new audio for every unique use-case, especially considering the data-intensive nature of artificial neural networks. Traditional training approaches for these extensive models encounter a bottleneck in data collection. The process of collecting and generating data resources is not only time-consuming but also expensive. Acquiring acoustic data is particularly challenging, as it entails obtaining the audio and then transcribing it. Unlike text data, that is generally less challenging to collect comparing to audio data, primarily due to the greater availability and accessibility of text-based resources compared to the complexities and resource requirements involved in recording and processing audio data.

The mentioned scenario has spurred significant research in self-supervised representation learning. This approach involves leveraging labels generated from well-designed pretext tasks, to supervise the pre-training of deep neural networks. The parameters obtained from this pre-training are subsequently employed, either wholly or partially, to initialize the parameters of task-specific deep neural networks. This strategy helps in addressing downstream tasks with a reduced need for extensively annotated data compared to traditional supervised learning.

Self-supervision entails training deep neural networks to predict one aspect of the input data based on another part of the input. This stands in contrast to supervised learning, where networks predict a predefined target output, and generative modeling, where networks estimate input data density or learn a generator for it. The key distinction in self-supervised learning algorithms lies in how they define the labels used for prediction, which is

the pretext task. The choice of this pretext task influences the resulting learned representations' (in)variances and, consequently, their effectiveness for various downstream tasks.

Self-supervised learning methods have proven to enhance the efficiency of learning with reduced data samples in various domains such as images, videos, speech, text, and graphs [40, 41, 42, 43, 44]. Some findings indicate that the quality of self-supervised representations improves logarithmically with the increase in the amount of unlabeled pre-training data [45]. This suggests that with advancements in data collection and computational capabilities, the performance achievable through "free" pre-training could see ongoing enhancements, as larger pre-training sets can be utilized without the necessity of manual annotation for new data.

1.1 Research Questions

This thesis seeks to explore several key research questions:

- How can self-supervised learning be effectively utilized to enhance speech recognition technology, particularly in the context of low-resource ASR challenges?
- What are the properties of the speech representations learned through self-supervised methods, and how do these properties contribute to the effectiveness of ASR systems?
- Can new methodologies be developed to train speech recognition models efficiently without relying on extensive labeled data?
- How can these self-supervised learning methods enable ASR models to learn useful speech representations with minimal labels, and what

impact does this have on the performance of ASR systems in low-resource settings?

1.2 Thesis Contributions

This thesis makes the following contributions:

- We provide a thorough series of experiments to assess the effectiveness of the latest High-Resource Environment (HRE) ASR models when tested in scenarios with limited resources. Our findings indicate that simply increasing training data from a different domain does not enhance the accuracy of ASR systems in low-resource environments. The empirical evidence suggests that utilizing deeper model structures is not efficient for LREs, even though they prove potent with abundant training data. Furthermore, we demonstrate that a successful approach involves pre-training with a language resource rich in data and fine-tuning with pertinent in-domain data for effective handling of low-resource ASR tasks.

- We illustrate a method for enhancing the performance of low-resource ASR tasks by leveraging large-scale corpora from unrelated domains. To overcome the challenge of limited training data, we introduce a novel model called **ScoutWav**. This model combines Self-Supervised Learning (SSL) with context-based word boundary information to create a high-performing ASR model for Low-Resource Environments. ScoutWav employs an improved Scout Network with a context vector embedding mechanism, enabling it to capture both local acoustic characteristics and broader contextual attributes, thus yielding high-quality word boundary data for a two-stage fine-tuning process.

Initially, we pre-train a wav2vec 2.0 [46] model using a high-resource dataset and then fine-tune it with LR data to adapt it to the specific task. Recognizing that different layers within a Transformer architecture capture various levels of linguistic information, we employ a wav2vec 2.0 layer analysis to identify subpar layers that fail to adequately capture acoustic-linguistic features. These underperforming layers are subsequently improved through a second fine-tuning step, leveraging context-based word boundary data to embed global context awareness into the ScoutWav model. We showcase the performance of the ScoutWav model on two different LRE datasets.

- We introduced **Local Aggregation BERT (LABERT)**, a new self-supervised speech representation learning model designed to generate speech representations suitable for low-resource ASR tasks. It is inspired by HuBERT [47], but uses a different approach to detect hidden units in the latent feature space. LABERT uses a committee-based active learning model to select more informative speech units for training. This helps to address the data bottleneck in low-resource ASR. To detect hidden units, LABERT uses a non-parametric aggregation method instead of a global clustering algorithm. This makes LABERT more efficient and scalable to large datasets. To select informative speech units, LABERT uses a committee-based active learning model. This model is trained to classify speech units with similar statistical structures into the same clusters. This allows LABERT to select a subset of the data that is both informative and diverse. This approach helps overcome data bottlenecks and enables the modeling of well-suited representations for downstream LRE ASR tasks.

- We proposed a new self-supervised speech representation learning model called **Regularized Contrastive Clustering BERT (RC-**

CBERT). It is proposed to generate speech representations that are well-suited for low-resource ASR tasks. RCCBERT is inspired by HuBERT, but it adopts a contrastive learning based clustering model to identify hidden units within the latent feature space. In this work, we adapt a one-stage online deep clustering method called Contrastive Clustering (CC) for the speech recognition problem to identify hidden units. CC uses a deep model to learn the feature matrix whose rows and columns correspond to the instance and cluster representations, respectively. In other words, CC utilizes the label as a special representation by mapping the input instances into a subspace with a dimensionality of the cluster number. This allows CC to view the matrix's rows as the likelihood of a particular cluster assignment (or soft labels for instances), and the columns of the feature matrix could be interpreted as the cluster distributions over instances (i.e., cluster representations). RCCBERT employs regularizing constraints to impose slow changes in the latent representations and overcome the definition of the negative samples, which helps to overcome data bottlenecks and enables the model to present well-suited representations for downstream LRE ASR tasks.

1.3 Chapter Road Map

The following chapters in this thesis are structured as follows:

- Chapter 2 provides an overview of the key concepts and background materials that are relevant to the research presented in the thesis. The goal of this chapter is to ensure that readers have a sufficient understanding of the field to appreciate the contributions of the thesis.

- Chapter 3 compares a set of deep neural network models with different percentages of the pre-training data to determine the key factors that influence in model selection for low-resource ASR problems.

- Chapter 4 presents self-supervised representation learning that makes use of the idea of two-step fine-tuning for low-resource ASR problems.

- Chapter 5 proposes a novel method that integrates local aggregation function and active learning technique to detect informative hidden speech units in low-resource ASR tasks.

- Chapter 6 presents another approach based on the contrastive oriented clustering to detect more informative speech units in low-resource scenarios.

- Chapter 7 summarizes this thesis and discusses possible future directions.

Chapter 2

Literature Review

In this chapter, we present the foundational information relevant to the thesis. We initiate by presenting an overview of automatic speech recognition in Section 2.1, which is a crucial application in speech processing and a key evaluation task for the systems in this thesis. Following that, in Section 2.2, we conduct a survey of three commonly employed neural network architectures in speech processing: RNN, CNN, and Transformer. Section 2.3 commences with a brief historical review of neural network pretraining and then delves into its recent advancements in self-supervised techniques, notably in the realms of visual, textual, and speech representation learning. Section 2.4 provides a focused review of prior research in neural representation analysis, establishing the methodological foundation for the subsequent analysis of self-supervised speech representations within this thesis. Finally, Section 2.5 surveys various low-resource ASR system approaches, establishing a benchmark for the subsequent evaluation of our proposed method.

2.1 Automatic Speech Recognition

The objective of automatic speech recognition is to empower machines with the ability to transcribe human speech into text automatically. We limit our discussion to the impact of End-to-End (E2E) deep learning techniques that are driving state-of-the-art speech-to-text recognition models. Speech-to-text ASR systems can detect input voice signals and produce the corresponding transcribed text in a computer-readable format [48]. The quality of the speech recognition system affects the difficulty of machine language understanding, which in turn can also influence the efficacy of spoken dialogue systems [49]. Therefore, it is a crucial step to create seamless communication between humans and machines. A brief history of ASR systems and their components are presented in Section 2.1, followed by a discussion of deep learning based state-of-the-art ASR techniques in Section 3. Section 4 provides an overview of applications of ASR on LREs, to present challenges, which motivated this study.

2.1.1 History and Components of ASR Systems

An ASR system converts an acoustic input sequence $X = \{x_1, ..., x_T\}$ of length T into a word sequence $W = \{w_1, ..., w_N\}$ of length N. The goal of the system is to find the most likely label sequence $\hat{W}$ for the speech input vector, X as follows:

$$\hat{W} = \underset{W \in \mathcal{V}^*}{\arg\max} P(W|X) \tag{2.1}$$

where $\mathcal{V}^*$ refers to all the label sequences [3]. Based on Equation 2.2, an ASR system aims to build a model, which can accurately compute the

posterior distribution $P(W|X)$.

The first speech recognition system, which mapped the output of a filter bank to hand-constructed templates to recognize 10 digits, was proposed by Bell Labs in the 1950s [50]. Voice-activated typewriters [51] and speaker-independent 10-vowel recognition systems were examples that used a filter-based approach in speech recognition [52]. These primary systems were able to detect only a single word which are called isolated word recognition systems.

Subsequently, researchers showed increased interest in expanding speech recognition to Large Vocabulary Continuous Speech Recognition (LVCSR) systems [53, 54]. Unlike in isolated word recognition systems, context in speech data and large vocabulary corpus has been major problems in ASR models for LVCSR [3]. Hence, integration of multiple models were developed, where models based on the Hidden Markov Model (HMM) showed impressive results [55], by categorizing the ASR problem into sub-problems (such as language and acoustic aspects being handled separately) and learning multiple models for each sub-problem category. However, In multiple model approaches, neural networks were employed for acoustic modelling, and hybrid Hidden Markov Model/Deep Neural Network (HMM/DNN) systems demonstrated significant effectiveness [56].

Categories of ASR Systems

A large and growing body of literature has investigated ASR systems. Based on fundamental principles and basic innovations, previous research can be classified into classical and end-to-end categories:

Classical ASR Systems A classical ASR approach converts input audio to its associated text representation by using multiple models to solve each sub-task related to the larger problem [4]. This approach was used in ASR for several decades [57, 55, 58]. The minimal configuration of these models includes a combination of three independent modules: the acoustic model, lexical model, and language model, each playing a different role in the whole system.

Figure 2.1 shows a general classical ASR system, which includes acoustic (acoustic–phonetic) modeling, lexical (pronunciation, lexicon/vocabulary) modeling, and language modeling as the three main sub-problem components. An acoustic model captures the relationship between the audio signal and the phonetic units, the lexical model captures probabilistic correlation between latent variables and lexical units, and the language model calculates the likelihood of a sequence of words.

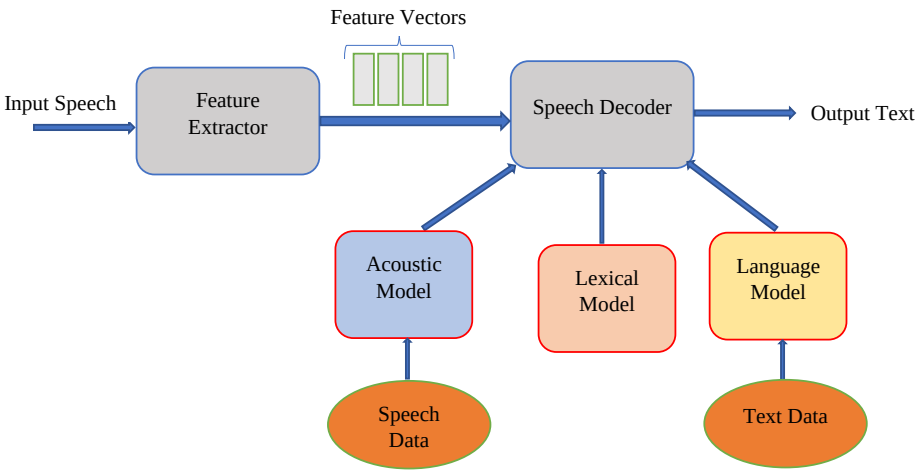


Figure 2.1: Architecture of a generalized classical ASR system consisting of acoustic, language and lexical modelling as well as a speech decoder.

An Acoustic Model (AM) is created by taking an audio recording of speech and then providing a statistical representation of the sound that makes up each phoneme. The main goal of the acoustic model is to map a sequence of

acoustic features to a sequence of phonetic units. For a given text sequence Y, an acoustic model provides the following estimation:

$$p(X|Y) \tag{2.2}$$

in which presents the probability or likelihood of a speech utterance X given Y.

To obtain a statistical model for ASR models, we need a collection of audio recordings along with their parallel text transcripts. This collection is called a corpus. The corpus is typically made up of a large number of audio-transcript pairs, where each pair consists of an audio recording and its corresponding text transcript. The audio recording is represented as a raw waveforms. Feature vectors are extracted from the audio signal using a variety of techniques, such as the log Mel-filterbank features and Mel-frequency Cepstral Coefficients (MFCCs). The sequence length of the feature vectors varies from recording to recording. The text transcript Y is a sequence of textual tokens. The specific tokens used in the transcript depend on the ASR system's final usage. For example, if the ASR system is to be used for transcribing conversations, the tokens might be words or phrases. If the ASR system is to be used for transcribing text, the tokens might be characters or subwords. Once we have the corpus, we can train a statistical model to map from the sequence of feature vectors to the sequence of textual tokens.

A commonly employed approach to represent Y involves utilizing a manually crafted pronunciation model that associates each word in Y with its corresponding pronunciation and associated likelihood. In this model, pronunciation model emits a sequence of phonemes, given a word. The

adoption of such a pronunciation model enhances the efficiency of modeling $P(X|Y)$. Instead of needing to estimate a distinct probability distribution for each unique word in the vocabulary, the acoustic model now only needs to account for a set of fundamental acoustic units, such as phonemes. These phonemes, which are limited in number and shared across words in a language, simplify the modeling process. If we denote the phonemes sequence obtained from Y using the pronunciation model as U, the task of modeling the conditional distribution of an audio sequence X given a text sequence Y, $P(X|Y)$, is transformed into the task of modeling the conditional distribution of X given the phone sequence U, $P(X|U)$.

Hidden Markov Models are popular models for acoustic modeling [59]. Probabilistic or deterministic models can be used in this structure as an acoustic model. A Gaussian Mixture Model (GMM) is a probabilistic model that is used to represent the distribution of feature vectors in a multidimensional space. These feature vectors are extracted from the audio signal and represent various properties of the speech, such as frequency content, energy levels, and spectral dynamics. The GMM is used to model the statistical properties of these feature vectors for different phonemes (basic units of speech sound) or words in the language being recognized. HMMs can be combined with GMM, referred to as GMM-HMM systems [60], to improve the accuracy of ASR systems. In contrast, a deterministic model such as Deep Neural Networks (DNNs) produces the output sequence exactly like the input sequence. GMMs and DNNs can be combined to calculate the hidden states of HMM to produce a final output sequence [61]. Recent developments in the field of deep learning have led to renewed interest in ASR systems to provide more accurate text [62]. As a result, HMMs were also combined with DNNs for acoustic modeling. In the process of parameter estimation for a DNN-based acoustic model, the typical procedure begins

by employing a pre-trained baseline GMM-HMM speech recognizer. This recognizer is employed to determine the target state label for each frame within the audio sequence. Once the target state sequence has been identified, the DNN-based acoustic model can be trained using backpropagation along with standard gradient descent methods.

The lexical model is responsible for mapping acoustic features of speech to corresponding words in the vocabulary. The lexical model is typically based on a pronunciation dictionary that contains the mapping between phonemes or sub-word units and their corresponding word forms. The pronunciation dictionary may also contain information about the stress and intonation patterns of words, which can be important for accurately transcribing spoken language. In some ASR systems, the lexical model may also include a mechanism for handling out-of-vocabulary (OOV) words, which are words that are not included in the vocabulary. This can be done by mapping OOV words to a set of similar words in the vocabulary, or by using a separate module to generate new pronunciations for OOV words based on their spelling or phonetic structure.

The language model (LM) is used to apply constraints on the recognition process to capture the structure and semantics of the target language. Within the ASR systems, pronunciation models play a critical role in bridging the gap between the symbolic realm of text and the acoustic world of spoken language. They operate as a specialized component within the LM, tasked with the crucial function of mapping written words to their corresponding phonetic representations. Unlike LMs, which focus on analyzing word sequences and assessing their likelihood based on grammatical and semantic principles, pronunciation models delve deeper into the linguistic structure. They essentially serve as a phonetic dictionary, decoding individual words identified by the LM and representing them through their

constituent phonemes – the fundamental building blocks of spoken language. Hence, language modeling is a process which used to convert a sequence of phonetic units into meaningful words and sentences. Recurrent Neural Network LM (LM) [63, 64] is one of the most well-known LM models. LM is primarily responsible to model the statistical properties of language, such as word probabilities and word sequences. This allows the ASR system to determine the likelihood of different word sequences occurring in a given context. For example, it helps distinguish between homophones (words that sound the same but have different meanings) by considering the surrounding context.

There are two main types of language models used in ASR:

- **N-gram language models**: These models are based on the n-gram hypothesis, which states that the probability of a word occurring depends on the $n-1$ words that have come before it. For a given a text sequence $W = (w_1, w_2, ..., w_N)$ where N represents the number of tokens (like words) in the sequence. A n-gram model decomposes the likelihood of producing W into the multiplication of the probabilities for each token in the sequence. Each token's probability is dependent on all the tokens that precede it in the sequence.

- **Neural language models**: Neural language models are a type of statistical language model that uses neural networks to predict the next word in a sequence. They are more powerful than n-gram language models because they can learn more complex relationships between words. Neural language models are typically trained on a large corpus of text, such as a book or a news article. The corpus is used to train the neural network to predict the next word in a sequence.

A speech decoder is another component of a classical ASR system that converts audio input data into a sequence of words. The acoustic signal is converted to a vector of features of the speech signal that is used to reduce the dimension [2]. An ASR decoder can produce a representation of the recognition hypotheses, and then, by applying language models, it is able to present the best recognition hypothesis. Therefore, language models can help to decide between interpretations of the same acoustic information, and using it in the ASR model increases the accuracy of the ASR model [1].

A classical ASR system can be simply formalized as follows:

$$\hat{W} = \arg\max_{W} P(W|X) = \arg\max_{W} (P(X|W).P(W)) \tag{2.3}$$

in which $\hat{W}$, $P(W|X)$, $P(X|W)$, and $P(W)$ are the estimated word sequence that the ASR system predicts, the posterior probability of the word sequence W given the observation sequence X, likelihood of observing the sequence X given the word sequence W which is modeled by the acoustic model, and prior probability of the word sequence W which is modeled by the language model, respectively.

Classical structures have some limitations [65]: As different models need different training methods and in-domain data, the training process becomes increasingly complex when trying to achieve global optimization. In addition, conditional independence between the tasks addressed by each model is assumed to simplify the training process. This is not reflective of the reality, in which all parts of speech are interconnected together.

End-to-End ASR Systems End-to-end models are supervised methods of learning in which the input audio feature is directly mapped to an

output sequence [4]. In this sense, end-to-end models are altered classical architectures that just use a deep network to convert audio to text directly. Therefore, there is no need to design many modules with different optimization functions [66].

Figure 2.2 shows the architecture of the end-to-end model, in which the encoder and decoder are the two main parts of the model. The encoder is the part of the model that maps the input sequence into a feature sequence, while the decoder produces the final text representation [66]. This architecture enables the model to learn the acoustic model and language model together within a network.

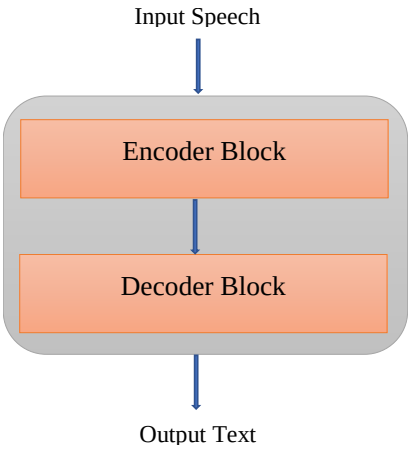


Figure 2.2: Architecture of an End-to-End Automatic Speech Recognition System

A considerable amount of literature has been published on end-to-end ASR systems [67, 68, 69, 70]. End-to-End models can be trained from scratch and can operate on words, sub-words or characters. They can be formally defined as:

$$P(W|X) = NeuralNetwork(X), \tag{2.4}$$

where X is the speech data and $NeuralNetwork(X)$ is a single, unified model that directly maps speech X into word probabilities.

The end-to-end speech recognition models use a single loss function for parameter estimation. This enables the model to use a function which is completely relevant to the final result, hence significantly improves the accuracy of the ASR system. Unlike the classical approach, there is no need for additional processing to obtain true transcription in the end-to-end model.

As the end-to-end model replaces the engineering process, which required to build classical systems, with a learning process, there is no longer a need for domain knowledge and experience to build the ASR model [3]. However, an end-to-end ASR system uses a deep neural network to directly convert the input sequence into an output sequence, it needs a large volume of training data to acquire higher accuracy, which is not feasible for every real application.

2.2 A Survey of the State-of-the-Art in End-to-End ASR Systems

Vast majority of state-of-the-art research in the speech recognition area has focused on large training datasets that use up to hundreds or thousands of hours of audio data in their models. In this section, we will discuss the extensive literature in the area according to the base architecture used in the model, which will be examined in five main branches. Section 2.2.1 will cover CNN- and RNN-based strategies. Section 2.2.2 and 2.2.3 will discuss recent research on Attention-Based models and Transformer-based models, respectively.

2.2.1 Methods Based on Convolutional Neural Networks and Recurrent Neural Networks

Due to the impressive results of DNN models, almost all state-of-the-art ASR systems use a modification of DNNs in their structure [71, 72, 18]. As RNN architectures, especially LSTMs, are a good option for sequence processing, they have been used in state-of-the-art STT systems [12]. CNN-based models are primarily known for their success in image processing and computer vision [73]. However, CNNs can also be applied to sequence learning tasks, including natural language processing (NLP) and, to some extent, ASR, albeit with certain limitations and adaptations. While CNNs are not the first choice for sequence learning due to their lack of explicit mechanisms to handle long-range dependencies in sequences, their ability to capture local patterns and their efficiency in training make them valuable in certain contexts or as part of hybrid models that leverage the strengths of different architectures. The combination of CNN layers before RNN layers are used to help the model to provide more accurate feature extraction [74]. In this section we summarize the CNN and RNN based a Table 2.1 illustrates an overview of these models.

RNN-Based Models for ASR

RNN-based architectures were used as language models in end-to-end models in the literature in [75, 76]. Research has shown the restricted advantages of minor architectural improvements in the original LSTM as a language model [77]. The design of Vanilla-LSTM is based on the use of intuitive multiplicative gates. Highway connections [78] and residual connections [79, 80] are the most prominent changes in the LSTM-based architectures, along with dropout [81].

A typical LSTM layer converts the input vector x_t to the output vector h_t through a gate-cell structure as follows [82]:

$$i_t = \sigma(W_{ix}X_t + W_{ih}h_{t-1} + P_i \odot c_{t-1} + b_i) \tag{2.5}$$

$$f_t = \sigma(W_{fx}X_t + W_{fh}h_{t-1} + P_f \odot c_{t-1} + b_f) \tag{2.6}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \phi(W_{cx}X_t + W_{ch}h_{t-1} + b_c) \tag{2.7}$$

$$o_t = \sigma(W_{ox}X_t + W_{oh}h_{t-1} + P_o \odot c_t + b_o) \tag{2.8}$$

$$h_t = o_t \odot \phi(c_t) \tag{2.9}$$

where X_t is the speech spectrum input at the time step t. The activation of the input, output, forget and memory cells are i_t, o_t, f_t and c_t, respectively. The output of the LSTM cell is h_t. $W_{.x}$ and $W_{.h}$ which are the weight matrices for the input and recurrent inputs, respectively. P_i, P_o, P_f are vectors that are associated with peephole connections. Finally, b_i, b_f, b_c, and b_io are bias vectors.

A large and growing body of literature has investigated the LSTM-RNN model [11, 12, 83, 84] and has shown state-of-the-art results on ASR tasks. LSTM variations, such as multiple LSTM layers stacked, yield better results [12]. However, gradient vanishing errors and the need for a large amount of training data are major problems of such models [85].

The gradient vanishing problem can be partially solved by adding skip connections or gating functions between LSTM layers. Residual LSTMs [79, 80] have connections between layers, reducing the severity of the gradient vanishing problem. Highway-LSTMs [78] are another structure that connects the memory cells in adjacent layers to create a parallel path for data flow. In Grid-LSTMs [86] the network created a multidimensional

grid with memory cells in LSTM cells. The results show better performance compared to Highway-LSTM on several ASR tasks. Layer-Trajectory Long Short-Term Memory (ltLSTM) combines stacked LSTMs, which scans the outputs from time-LSTMs, with a summarized layer to include trajectory information for the final classification [82]. This model decouples the time recurrence and the final classification. Based on this architecture, the forward-propagation of time-LSTM and layer-LSTM can be considered in two separate threads, hence the network computation time is the same as the standard time-LSTM [82].

Future context frames always carry noteworthy information, which helps to predict the target label more accurately. In [22], the authors improve the accuracy of ltLSTM models by utilizing future context frames [82]. They use a fixed-size vector representation of the variable's future, named *look-ahead embedding*. Bi-directional LSTM can reduce the latency of the BLSTM model [78] by using chunk-wise forward LSTM. Time-delay neural networks (TDNN) [87] and feed-forward sequential memory networks (FSMN) [88] use the same architecture: a window of acoustic frames fed into 1-D Convolutional Neural Networks. These models consider future acoustic frames to improve the accuracy of the models. In, [20] the authors improved the contextual layer trajectory LSTM (cltLSTM) [22] by using a two-headed structure in which one head has zero latency while the other head has a small latency.

Furthermore, in end-to-end models, deep-BLSTM neural networks obtain state-of-the-art results [89, 90]. As BLSTM-based ASR systems need the entire speech utterance to compute the output frames, it is necessary to know the entire past and future context of the speech [91]. Therefore, it can be concluded that as such models need large future context, they can not be used for streaming ASR systems. A possible solution to over-

come this problem can be used by overlapping chunks of frames to compute the output of the backward LSTM. Consequently, latency-controlled BLSTM (LCBLSTM) has been proposed [92, 93]. However, an important issue in these models is that frame overlapping increases the computational cost. Deep contextualized acoustic representations (DeCoAR) [94] exploits a large amount of unlabeled data through representation learning and reconstruct a temporal slice of filterbank features from past and future context frames. By using contextual information, such as the words that come before and after a particular sound, DeCoAR can better understand the meaning and context of spoken language. DeCoAR has shown promising results in a variety of speech recognition tasks, including speech-to-text transcription, speaker recognition, and keyword spotting.

Feedforward Neural Networks (FNN) have also been combined with RNN to improve the accuracy of STT systems. In [95], the authors presented different networks in which FNNs were combined with LSTM to show temporal patterns and to summarize the long history of previous inputs. LSTM Recurrent Projection was proposed in [12], in which the authors added a feedforward layer by considering recurrent information based on the output of LSTM. Simultaneously, a LSTM structure was updated by adding FNN before and after LSTM in [96]. However, while clear improvements have been introduced to manage the gradient vanishing problem, no clear improvements or modifications have been done to LSTM-based models to tackle the limited training datasets.

CNNs for ASR

Convolutional Neural Network (CNN) is a specialized form of neural network that uses a supervised deep feature learning model to process data

Architecture	Paper	Data set (duration)	Error rate
RNN/CNN	DLSTM[11]	TIMIT	37.6 - 17.7(PER)
	[12]	Google Voice Search Task (1900 h)	11.8 - 10.7 (WER)
	two-head cltLSTM [20]	English Spoken Utterance (200 h-2000 h)	12.24 - 9.34 (WER)
	Highway LSTM[78]	AMI (100 h)	57.5 - 37.7
	[24]	data from YouTube, Google Videos, and Broadcast News	24.0 (WER)
	ltLSTM[82]	Microsoft Cortana and Conversation Data (30k h)	19.41 - 9.28 (WER)
	[73]	WSJ (81 h), Librispeech (1000 h)	19.7 - 17.3 (WER)
	Residual LSTM [80]	AMI (100 h)	57.5 - 37.7 (WER)
	residual LSTM [79]	TIMIT, HKUST (150 h)	50.8 - 39.3 (PER/CER)
	improved ltLSTM[22]	Microsoft Anonymized Production (65k h)	19.41 - 9.28 (WER)
	[84]	artificially created data (2k h)	15.7 (WER)
	prioritized Grid LSTM (pGLSTM) [85]	AMI (100 h), HKUST (150 h), GALE Mandarin, Arabic MGB	22.54 (WER)
	feedforward sequential memory networks (FSMN)[88]	Switchboard (300 h)	13.2 (WER)
	[89]	TIMIT	29.0 - 18.3 (WER)
	[90]	TIMIT	-
	time-delay LSTM (TDLSTM) [91]	(81 h), HKUST (150 h), LibriSpeech (1000 h)	35.5 - 4.6 (WER)
	DBLSTM-HMM[92]	Switchboard (300 h)	14.7 (WER)
	[95]	CHiME	11.91 (WER)
	CRNN [96]	HKUST	31.43 (WER)
	[97]	Bable (10 h)	83.8 - 67.7 (WER)
	[98]	Bable (10 h)	-
	[99]	TIMIT	-
	Jasper [100]	WSJ, Hub5	16.1 - 2.95 (LER)
	TDS convolution [101]	LibriSpeech (1000 h)	7.25 - 3.01 (WER)
	DeCoAR [94]	WSJ (81 h), LibriSpeech (1000 h)	10.38 - 4.64 (WER)
	PASE [102]	DIRHA	33.5 - 29.8 (WER)
CTC-Based	Quartznet [103]	WSJ (81 h), Librispeech (1k h)	10.98 - 2.96 (LER)
	Contextnet[104]	Librispeech (1k h)	1.9 (LER)
	[105]	TIMIT	30.51 (LER)
	[69]	WSJ (81 h)	14.1 (WER)
	Deep speech 2 [1]	WSJ, LibriSpeech, Vox Forge, CHiME (11940 h)	50.7 - 3.1 (WER)
	[106]	TIMIT	-
	[107]	Switchboard-1(300 h), Fisher (1698 h)	20.8 (WER)

Table 2.1: CNN and RNN based ASR models.

[108]. A specialized kind of linear operation called convolution is utilized by this type of deep neural network. One of the first research on CNN has been done in [109] which this network has been used to identify handwritten characters. Image and video recognition, recommendation systems, image classification, medical image analysis, and natural language processing are different applications of this network. [71]. CNN architecture needs a large amount of data for training to be able to utilize it for applications with high-dimensional input data, such as image processing and speech recognition. Furthermore, by increasing the number of parameters to train in deeper structures, this network requires high performance computing power [110].

CNNs are efficient deep networks that exploit local properties in speech recognition [98]. The frequency variation in speech signals is another application of this network in speech recognition systems[111]. Recent developments in CNNs have led to interest in their use in both high-resource [112] and low-resource [97] environments. CNN is used for acoustic modeling in [99, 113]. In these approaches, to achieve more stable acoustic features from the input audio, convolution layers are applied over the windows of the acoustic frames. Jasper [100] is a deep convolution model in which the convolutional layer $1D$ is stacked with skip connections. While Jasper's fully convolutional design allows for efficient handling of sequential audio data, this approach might not capture the temporal dynamics as effectively as RNN based models or those incorporating attention mechanisms. CNNs are excellent for extracting hierarchical features from data, but the inherently sequential and context-dependent nature of speech might benefit from architectures designed to handle long-term dependencies more explicitly. Jasper's reliance on dense residual connections and a large number of convolutional layers can lead to a substantial increase in computational

resources and memory usage. This could limit its applicability in resource-constrained environments, such as mobile devices or embedded systems, despite its design intentions for efficiency and scalability. Additionally, the training complexity and data requirements for Jasper to achieve optimal performance may be prohibitive for some applications. The need for large labeled datasets and significant computational power for training might not be feasible for all organizations or languages, particularly those with limited resources or less-represented languages. In [101], the authors used depth-wise separable convolution layers [114] to improve the speed and accuracy of CNN networks and introduce an innovative approach to ASR by employing a sequence-to-sequence model that utilizes Time-Depth Separable (TDS) convolutions. This architecture aims to improve both the computational efficiency and the accuracy of ASR systems by integrating TDS convolutions into the model, which separates the convolution operation into time-based and depth-based components. This separation allows for a significant reduction in the number of parameters and computational complexity while maintaining or even enhancing model performance on speech recognition tasks.

QuartzNet [103] is another CNN-based architecture that obtains state-of-the-art results in ASR systems. The authors proposed a very deep network that uses $1D$ time-channel separable convolution layers. ContextNet [104] is a CNN-based model that squeeze-and-excitation layer [115] to enhance the accuracy of the ASR system in terms of WER. ContextNet used an SE layer after the convolution layer to gain global information from the audio input to obtain the output of the system. However, with this growth in the proposed CNN-based ASR, there is an increasing concern about modeling the long-term context dependencies among the spectrum of speech signals. As CNNs access the context of higher layers, this network cannot modu-

late information from the lower layers. Problem-agnostic speech encoder (PASE) [102] is an end-to-end architecture that consists of several layers of convolutional neural networks and long short-term memory networks. It takes raw speech waveforms as input and generates high-level features that capture the salient characteristics of the speech signal. These features can then be used as input to other speech processing models or for downstream tasks such as speech recognition or speaker identification.

While RNNs capable of processing sequential data, often face difficulties in directly mapping the speech input sequence to the textual output sequence. This challenge is primarily due to RNNs' limitations in handling long-term dependencies within sequences [105]. The output phonemes or other small output units require further processing to produce the final transcription [3]. Due to this fact, pre-segmentation should be applied to the training data, and post-processing of the output will be necessary to produce the final label sequence [105]. In addition, end-to-end models faced with data alignment problem when using RNN and CNN to model the time-domain features. The loss functions in RNN and CNN are defined based on each point in the input sequence, therefore, for training purposes, these models should know the alignment relation between the output sequence and the target sequence. Connectionist Temporal Classification (CTC) is a loss function that can be combined with RNN and CNN to overcome this drawback. CTC can solve the problem of the alignment relation while calculating the loss [3]. Hence, it can be concluded that with the advent of the CTC loss, data alignment and producing target transcription problems have been solved, and RNN and CNN can be used in the end-to-end ASR model. Table 2.1 presents CTC-based ASR models.

BLSTM-CTC, proposed in [116], contains a feedforward layer combined with two LSTM layers. The authors showed that by increasing the number

of hidden units in the network structure, the accuracy of the ASR system could be improved. Another combination of deep bidirectional LSTM and CTC objective functions is presented in [67], where audio spectrograms are processed by a deep bidirectional LSTM layer and finally a CTC loss function is used as the output layer. In [106], an end-to-end model is implemented that consists of two different neural networks for phoneme recognition. Convolutional layers are applied for frame-level classification, and RNN with CTC is used to decode frames in the output sequence. The combination of hierarchical CNNs with a CTC layer without using recurrent connections is used in [65], in which the authors showed the ability of the CNN model to capture temporal dependencies. To produce a large context window for each output in this model, a stacked convolutional layer is utilized, which is followed by multiple fully connected and CTC layers.

A deep RNN layer with CTC utilized for large labelled training datasets for two different languages, English and Mandarin, is shown in [117]. The direct acoustics-to-word CTC model [107] demonstrated results on two well-known benchmark datasets, Switchboard and CallHome. In this model, two techniques are proposed to improve the training of the ASR model in these datasets. To speed up the training process in ASR, the authors in [68] used a partition scheme to improve parallelization and mapping their RNN model to GPUs successfully. This model also used a novel combination of collected and synthesized data to develop a robust process to account for realistic variations in noise and speakers. This model is an efficient method for large-scale data training in the ASR task.

An important assumption in the CTC model is that all labels in the output sequence are independent of each other [3], so the CTC cannot model languages. Therefore, the models that are used with CTC have to be combined with external language models to improve the final accuracy of the model.

In [118], the authors combined a recurrent neural network architecture with a language model which includes a large vocabulary and continuous speech recognition. The results demonstrated and emphasized the importance of using language models to achieve high accuracy in ASR systems. Furthermore, other models, such as Deepspeech2 [1], have shown the importance of using a language model in a complex structure. Recently, the CTC loss has been combined with Attention-based and Transformer-based models and has achieved better results on ASR tasks, as we will cover in the next sections.

2.2.2 Methods Based on Attention Models

The Neural Transducer [119] considered the attention method on chunks for input and using end-of-chunk symbol for training. Incremental prediction is one of the major problems that affect the models presented in the previous sections. This problem arises when there are new arriving input data or long input and output sequences are presented to the model. Neural Transducer models are able to overcome them by computing the next-step distribution conditioned on the partially observed input. In [120], an end-to-end model based on hard monotonic attention was presented for online decoding and has linear time complexity. In [121], Monotonic chunk-wise Attention (MoChA) was proposed. In MoChA, soft attention module is calculated over the small chunks of data which obtained from the input sequence. An improved MoChA-based ASR system is presented in [122], where CTC and cross-entropy losses are jointly used to train the MoChA models and the MWER model is adopted to optimize the model. In [123, 124] a hybrid CTC-attention architecture is proposed, which utilizes CTC loss as a regularization process in an attention-based network. In [125] an end-to-end

Architecture	Paper	Data set (duration)	Error rate
Attention-Based	Neural Transducer [119]	TIMIT	33.4 - 18.2 (PER)
	[120]	TIMIT	16.0 (PER)
	MoChA [121]	WSJ	17.4 - (WER)
	Improved MoChA[122]	LibriSpeech (1000 h)	8.82 (WER)
	hybrid CTC/attention[123]	WSJ (81 h), CHiME	43.45 - 11.27 (CER)
	joint CTC-attention [124]	WSJ1 (81 hours), WSJ0 (15 hours), CHiME	44.99 - 7.36 (WER)
	CTC/attention [125]	LibriSpeech (1000 h), HKUST(200 h)	22.5 - 5.3 (WER)
	[127]	WSJ (81 h), TIMIT	25.8 (WER)
	[129]	WSJ (81 h), TIMIT	15.7 (WER)

Table 2.2: Attention based ASR models.

hybrid CTC attention architecture was proposed using a stable monotonic chunk-wise attention(sMoChA) to provide a stream-based global attention and a truncated CTC (T-CTC) to compute prefix scores.

Another common output sequence for attention-based end-to-end ASR systems is a character (i.e., *grapheme*) sequence [126, 127]. In [128], words and sub-word units are used as the language model to be learned in the decoder. The length bias and the corresponding beam problem are the main problems of the attention-based encoder-decoder model, which has been mentioned in [129], and a heuristic-based model is not suitable for it; therefore, a beam search structure based on reinterpreting the posterior sequence was proposed. Attention based ASR models are set out in Table 2.2.

2.2.3 Methods Based on Transformer Networks

Transformer Networks have become one of the most popular and powerful models in natural language processing [130]. The architecture of the Transformer model has made it possible to train a stack of self-attention layers [131] by applying residual connections between layers, [132] followed by a normalization layer [133]. In [134, 135], a transformer decoder was used as a language model and showed impressive results on different benchmarks.

Transformer based ASR models are presented in Table 2.3.

Speech-Transformer [23] is an end-to-end sequence-to-sequence model that has no recurrence and is completely based on attention mechanisms. This model uses the basic structure of a transformer network, but the encoder combines a self-attention layer with convolution layers to present an approximate hidden representation with character length. In [136], the authors mentioned that an encoder based on the self-attention mechanism is unable to present an effective acoustic model, so a combination with LSTM layers was proposed. A transformer-based acoustic model for hybrid ASR systems is presented in [137] that evaluated several architectures to encode the input sequence based on absolute or relative positional information. Furthermore, applying iterated loss enables it to train a deeper model based on transformer networks. Wav2vec 2.0 is a Transformer-based framework for self-supervised learning of representations from raw audio [46]. Wav2vec 2.0 used a multi-layer convolutional neural network to encode the input data and, after masking the spans of the resulting latent speech representations, fed them to a Transformer network to build the contextualized representations. Another self-supervised learning model is HuBERT [47] based on the Transformer encoder that used an offline clustering step to present target labels for a BERT-like prediction loss [138].

The Conformer [139] is a hybrid model in which the Transformer and convolution layers are combined to capture content-based global interactions and relative offset-based local correlations, respectively. In [140], the Lite-Transformer architecture, an efficient mobile NLP, was proposed. In this model, self-attention is combined with convolution layers between a pair of feedforward modules. The Conv-Transformer Transducer [21] is a transducer framework that is suitable for streaming ASR systems. In this model, a unidirectional transformer is combined with interleaved convolution lay-

ers for capturing the future context, to present the audio encoding process. In [141], a Transformer-based architecture is proposed for post-processing ASR tasks. This model receives the output of the acoustic model and presents grammatically and semantically the correct final output sequence. w2v-BERT [142] is a combination of contrastive learning and Masked Language Modeling (MLM), where the first method involves training a model to convert continuous speech signals into a limited number of distinct speech tokens that can be distinguished from each other. The second method trains the model to understand the context of speech by predicting masked tokens in the discretized speech data.

In [143], a Transformer-Transducer architecture was proposed. The authors presented a training technique that defines both streaming and non-streaming models in a single algorithm. A stack of transformer layers is applied to encode the audio. In [144] a speech Transformer is combined with a bidirectional decoder to learn the encoder and decoder. The encoder in STBD is similar to a standard encoder in transformer networks, but it has two different unidirectional decoders which generate two different directional targets. Finally, the use of convolutional layers instead of positional embedding in transformer networks is a strategy used in [145] to acquire relative positional information. Transformer Encoder Representations from Alteration (TERA) [146] proposes a self-supervised learning method for training transformer encoder models to generate high-quality speech representations. The method is designed to address the challenge of training speech recognition systems in low-resource settings, where large amounts of labeled speech data are not available. TERA learns through the reconstruction of acoustic frames from their altered counterpart which are time, frequency, and magnitude. Speech SimCLR [147] is a self-supervised objective for speech representation learning where applies augmentation

on raw speech and its spectrogram. Speech SimCLR objective is the combination of contrastive loss that maximizes agreement between differently augmented samples in the latent space and reconstruction loss of input representation. WavLM [148] learns universal speech representations from large amount of unlabeled speech data and adapts effectively across various speech processing tasks. BERT-based Speech pre-Training with Random-projection Quantizer (BEST-RQ)[149] is a self-supervised learning algorithm for speech recognition which masks speech input and feeds into the encoder. The encoder learns to predict the masked region based on the unmasked speech signals where the learning targets are labels provided by a random-projection quantizer. ScoutWav [150] is a model which integrates context-based word boundaries with self-supervised learning, wav2vec 2.0, to present a low-resource ASR model. ScoutWav pretrains a model on high-resource environment datasets and then fine-tunes with the LRE datasets to obtain context-based word boundaries. The resulting word boundaries are used for fine-tuning with a pre-trained and iteratively refined wav2vec 2.0 to learn appropriate representations for the downstream ASR task. LABERT (Local Aggregation and BERT-based Self-Supervised Learning [151] is a novel approach designed to enhance ASR systems, particularly in low-resource scenarios. LABERT addresses the challenge of detecting informative hidden units within ASR models by combining local aggregation techniques [152] with self-supervised learning strategies derived from the BERT (Bidirectional Encoder Representations from Transformers) framework. This method focuses on efficiently utilizing the limited available data to improve the representation of speech features, making it more effective for ASR tasks. By leveraging local context and self-supervised learning, LABERT aims to identify and emphasize the most relevant features within speech data, thus improving the model's ability to recognize speech accurately with fewer resources. The approach demon-

Architecture	Paper	Data set (duration)	Error rate
Transformer	[135]	TEDLIUM (200 h)	38.6 - 15.82 (WER)
	[136]	LibriSpeech (1000 h)	2.26 (WER)
	[137]	LibriSpeech (1000 h)	11.28 - 4.3 (WER)
	HuBERT [47]	LibriSpeech (1000 h)	6.8 - 1.7 (WER)
	wav2vec 2.0 [46]	LibriSpeech (1000 h), TIMIT	15.6 - 1.9 (WER)
	Conformer [139]	AISHELI-1 (178 h)	10.56 - 6.64 (CER)
	Lite-Transformer [140]	LibriSpeech (1000 h)	14.7 - 5.2 (WER)
	[141]	LibriSpeech (1000 h)	8.3 - 3.4 (WER)
	Speech-transformer [23]	WSJ (81 h)	12.2 - 10.92 (WER)
	Tera [146]	LibriSpeech (1000 h), TIMIT	8.31 - 6.01 (WER)
	w2v-BERT [142]	LibriSpeech (1000 h), Libri-Light(60k h)	5.0 - 1.3 (WER)
	Speech SimCLR [147]	LibriSpeech (1000 h), TIMIT, IEMOCAP	15.1 - 5.89 (WER)
	WavLM [148]	VoxCeleb1, VoxCeleb2	29.2 - 4.0 (WER)
	BEST-RQ [149]	LibriSpeech (1000 h)	2.7 - 1.4 (WER)
	ScoutWav [150]	LibriSpeech (1000 h), WSJ, TEDLIUM, Common Voice	10.14 - 24.55 (WER)

Table 2.3: Transformer based ASR models

strates potential in boosting the performance of low-resource ASR systems by enhancing the quality and informativeness of speech representations, offering a promising direction for future research in speech processing technologies.

2.3 Self-Supervised Learning

2.3.1 Definition of Self-Supervised Learning

Over the course of the past decade, there has been a remarkable revolution in the field of speech processing, driven by the advancements in deep learning techniques. This revolution has opened up new possibilities and unleashed the potential for a variety of real-world applications. At the heart of this transformation lies the use of supervised learning in conjunction with deep neural networks. This approach has proven to be a game-changer, particularly in scenarios where there is an ample supply of labeled data [153].

What makes this development so significant is the substantial leap in performance it has facilitated. Traditional methods in speech processing often struggled to handle the intricacies of spoken language, especially in noisy or complex environments. However, deep learning models have shown exceptional prowess in understanding and interpreting speech patterns [72].

Furthermore, the versatility of deep learning-based speech processing has paved the way for a multitude of applications. These range from voice assistants that understand and respond to natural language commands to automatic transcription services that convert spoken words into written text with remarkable accuracy. Additionally, speech recognition in noisy environments, such as crowded restaurants or busy streets, has improved significantly, making it possible to develop technologies that function effectively in real-world settings [154].

To address the challenge of obtaining labeled data, researchers have explored methods that utilize unpaired audio-only information. These approaches aim to expand the range of industrial applications for speech and address the limitations posed by languages with limited available resources [155]. Taking inspiration from how children acquire their first language by listening and engaging with their family and environment, scientists are investigating the use of raw audio waveforms and spectral signals to develop speech representations. These representations are designed to encompass a wide spectrum of information, including low-level acoustic elements, lexical knowledge, and even syntactic and semantic details. Subsequently, these acquired representations can be employed in various downstream applications that require minimal labeled data [156, 157]. In a formal sense, representation learning refers to algorithms that extract hidden features capable of capturing the fundamental explanatory factors behind the observed input [157].

Representation learning methods are typically categorized under unsupervised learning, a branch of machine learning that identifies inherent patterns in training data without predefined labels or scores [158]. This unsupervised designation distinguishes these methods from supervised techniques, which assign labels to training samples, and semi-supervised methods, which use a small set of labeled samples to guide the learning process with a larger volume of unlabeled data. Unsupervised learning encompasses various techniques, such as k-means clustering [159], mixture models [160], and autoencoders [161].

Within the realm of unsupervised learning, there's a rapidly growing subcategory known as self-supervised learning. Self-supervised learning approaches leverage information extracted from the input data itself to generate labels for training, with the goal of learning representations that prove valuable for subsequent tasks. For instance, it's worth noting that traditional unsupervised k-means clustering doesn't adhere to this self-supervised definition, as it primarily focuses on minimizing within-cluster variance during the learning process.

We can break down self-supervised learning approaches into two stages. In the initial stage, an SSL model is employed to pre-train a representation model. In the subsequent stage, downstream tasks can either utilize the acquired representation from the fixed model or refine the entire pre-trained model during a supervised phase [162]. Examples of these downstream applications include automatic speech recognition and Speaker Identification (SID).

Desirable speech representations should ideally possess three key characteristics: they should be disentangled, invariant, and hierarchical. This means that they should be able to separate and distinguish various factors within

spoken utterances, such as speaker identity, style, emotion, surrounding noise, and communication channel noise, which are typically richer in information compared to the corresponding text transcriptions [163]. Additionally, these representations should remain consistent even when confronted with changes in background noise and communication channels, ensuring their stability across different application scenarios. Lastly, the ability to create hierarchical features at different levels, including acoustic, lexical, and semantic, is essential to accommodate various application requirements. For example, a task like speaker identification benefits from a detailed, low-level acoustic representation, whereas speech translation tasks require a more abstract and semantic representation of the input utterance [163].

2.3.2 Background

Self-supervised representation learning is a paradigm in machine learning that aims to learn useful representations of data without relying on explicit human-provided labels. Instead, SSL techniques generate their own supervisory signals by exploiting the inherent structure of the data. In this section, we will discuss the history of self-supervised representation learning methods and how they have led to the recent surge in interest in these methods.

Clustering Models

Early investigations into acquiring hidden speech and audio representations primarily employed basic models that directly optimized the likelihood of training data or used the expectation maximization (EM) algorithm as an

intermediary. These initial studies also utilized straightforward clustering techniques. For instance, in research like [164], word patterns underwent a semi-automated clustering process using methods like k-means. Subsequently, isolated words were identified by associating them with the training cluster that was the closest match to the test data.

By evolving, in the modeling, progressing to the point where subword units were described using Gaussian mixture models [165]. This advancement allowed for a better representation of the various nuances present in the input data. Initially, GMMs were constructed for context-independent phonemes. Subsequently, state-clustering algorithms [166] were developed to create GMMs for context-dependent phonemes. In these mixture models, each latent component served as a template for a typical speech frame, which posed challenges when dealing with extensive datasets exhibiting diverse characteristics.

Neural Models

Recently, there has been a noticeable shift in emphasis within representation learning towards neural models. These neural models, when compared to GMMs and HMMs, offer distributed representations with a greater capacity to efficiently encode various input signals into latent binary codes. Some early techniques in this domain include Restricted Boltzmann Machines (RBM) [162], denoising autoencoders [167], Noise Contrastive Estimation (NCE) [168], sparse coding [169], and energy-based methods [170]. Many of these techniques have found applications in computer vision (CV) and natural language processing problems, which served as a source of inspiration for their adaptation to speech-related tasks.

Learning By Optimizing Pretext Task

The emerging trend focuses on networks that achieve the mapping of input data to desired representations by addressing a pretext task. A pretext task serves as a means for the model to effectively obtains the knowledge present in unlabeled data. It's crucial for this task to present a sufficient level of difficulty to encourage the model to grasp higher-level abstract representations and not be too simple, which could lead to the exploitation of basic shortcuts. Early advancements in this area encompassed end-to-end training of deep neural architectures using pretext tasks, such as restoring the original colors in black-and-white images [171], simultaneous learning of latent representations and their cluster assignments [172], and predicting the relative positions of image patches [173]. Another well-received approach involves variational autoencoders (VAEs) [[174]. While conventional autoencoders focus on unsupervised learning by reconstructing input data after it passes through an information bottleneck, VAEs take a different route by estimating a neural model of a probability density function. This estimation approximates the unknown true distribution of the observed data, for which we only have access to independently identically distributed samples.

In the self-supervised learning domain, a common pretext task associated with autoencoding involves creating an object based on incomplete information about it. This concept finds widespread application in natural language processing, as seen, for instance, in tasks where the aim is to predict the next token in a sentence using the preceding tokens, as demonstrated in ELMo [175], or in predicting the masked tokens within a sentence, as exemplified by the Bidirectional Encoder Representations from Transformers (BERT) series [176]. Another prevalent pretext task in the third wave of

SSL is contrastive learning [177], wherein a model is trained to distinguish a target instance from a set of negative samples.

2.4 Speech Representation Learning

Speech is a complex and dynamic signal that is characterized by its temporal and spectral variability. This variability makes it challenging to develop self-supervised learning pretext tasks that are directly applicable to speech processing. Pretext tasks that are based on image patches or text sequences may not be effective for speech processing because speech signals do not have a clear spatial or sequential structure.

In contrast to computer vision, where an image typically maintains a fixed-size representation, representing a speech utterance as a sequence of variable length is more appropriate. Consequently, pretext tasks created for CV are generally not directly transferable to speech processing. Text and speech can both be expressed in sequential form. Therefore, it seems intuitive to employ learning techniques initially designed for text directly in the context of speech. However, a significant disparity lies in the fact that speech signals are characterized by sound pressure measurements with thousands of samples per second, leading to notably longer sequences compared to text. Even attempts to shorten the sequence length using spectral representations can still result in hundreds of frames per second. Employing conventional neural network architectures like Transformers to process such extended sequences can present challenges concerning computational speed and memory demands. Furthermore, in NLP, a prevalent practice involves employing a pretext task that emulates a categorical distribution of masked or forthcoming inputs. Given the structure of text where it

can be segmented into distinct tokens like words, subwords, or characters, establishing a finite vocabulary for these tasks is a straightforward process. However, this concept doesn't transfer to speech modeling due to the continuous nature of speech signals. Therefore, adapting self-supervised models from NLP and CV directly to speech processing necessitates innovative approaches tailored to the unique attributes of speech signals.

Self-supervised techniques for acquiring speech representations can be broadly divided into generative, contrastive, and predictive approaches.

2.4.1 Generative Approaches

In generative approaches, the pretext task involves generating or reconstructing the input data using a restricted perspective. This encompasses predicting forthcoming inputs based on prior inputs, distinguishing masked from unmasked elements, or discerning the original input from an altered or corrupted view. Autoencoders (AEs) are one of the main groups of algorithm in the generative approaches which consists of an encoder and decoder and the pretext task is to reconstruct the given input. The Variational Autoencoders (VAE) presents a probabilistic approach to the autoencoder, establishing the latent representation through a posterior distribution involving stochastic latent variables [174]. Another model within this realm is the Vector-Quantized Variational Autoencoders (VQ-VAE) [178], which builds upon the original VAE [179] by introducing a unique parameterization for the posterior distribution related to discrete latent representations.

The concept of Autoregressive Predictive Coding (APC) [180], [181] draws upon the principles of traditional Linear Predictive Coding (LPC) utilized

in speech feature extraction [182] and autoregressive language models applied to text. In this approach, the model is trained to anticipate forthcoming information based on past observations. In [183], the Autoregressive Predictive Coding objective is expanded to encompass multi-target training. This modified objective involves generating frames both from the past and future, considering the preceding context. Additionally, the VQ-APC technique described in [184] incorporates quantization into the APC objective, introducing an information bottleneck that acts as a regularization mechanism. One limitation of Autoregressive Predictive Coding is its focus on encoding information solely from previous timesteps and not considering the complete input. To address this concern, DeCoAR [94] merges the bidirectional capability found in ELMo, a widely used NLP model [175], with APC's reconstruction objective. This hybrid approach aims to overcome the limitation and enable the encoding of information from the entire input.

Masked reconstruction methods are another generative approaches which takes the inspiration from BERT's masked language model task [176]. In the pre-training phase of BERT, some tokens within input sentences are obscured by substituting them with a designated masking token or an alternative input token. The model's objective is to successfully restore these masked tokens based on the non-masked ones. Recent research has delved into comparable preliminary tasks for speech representation learning. Much like the DeCoAR model mentioned earlier, this approach enables a model to acquire contextualized representations that capture information from the complete input. pMPC [185] is a method that chooses speech frames with masking based on the phonetic segmentation present within a spoken expression. However, it is important to note that acquiring this segmentation requires some labeled data. While many studies employ masking

over the temporal dimension of the input, it is also possible to mask speech over the frequency dimension when using spectral input features [186]. In the case of non-autoregressive predictive coding (NPC) [187], time masking is incorporated through masked convolution blocks. Drawing inspiration from XLNet [188], some proposals suggest reconstructing the input from a shuffled version [189] to bridge the gap between the pre-training and fine-tuning phases in masking-based approaches. DeCoAR 2.0 [190] incorporates vector quantization, demonstrating enhanced learned representations. Additionally, the TERA model [146] introduces two dropout regularization methods—attention dropout and layer dropout—which are modifications of the original dropout method [81].

2.4.2 Contrastive Approaches

A speech signal encapsulates richer information compared to text, encompassing elements like speaker identity and prosodic features. This complexity makes generating speech a more challenging task. Hence, it might not be the most effective approach to uncover contextualized hidden factors of variation by focusing solely on reconstructing the unprocessed speech signal. Contrastive models take a different approach to representation learning; they do so by discerning a target sample (positive) from other unrelated samples (negatives) with reference to an anchor representation. The main objective of this pretext task is to maximize the similarity in the latent space between the anchor and positive samples, while minimizing the similarity between the anchor and negative samples.

One notable instance of a contrastive model is Contrastive Predictive Coding (CPC) [177]. CPC employs a convolutional module to create localized representations, followed by a recurrent module that generates contextual-

ized representations. From these contextualized representations, an anchor representation is derived through a linear projection. The wav2vec model, as described in [191], builds upon the CPC method. It employs fully convolutional parameterizations in its representation model, incorporating receptive fields of 30 ms and 210 ms. Unlike the CPC loss, which addresses a 1-of-N classification task per instance—assigning the anchor to the positive class or incorrectly to one of the $N-1$ negative classes—the wav2vec loss focuses on a sequence of N separate binary classifications.

The wav2vec 2.0 model [46] integrates both contrastive learning and masking approaches. Similar to the CPC model, it employs the InfoNCE loss [177] to enhance the similarity between a contextualized representation and a localized representation. Additionally, a quantization module is utilized to derive a discrete representation, which practically means avoiding negative sampling from the same category as the positive. To process input waveforms, the model incorporates a convolutional module followed by a Transformer encoder. Furthermore, the wav2vec-C approach [192] extends wav2vec 2.0 by incorporating a consistency term in the loss. This term aids in reconstructing input features from the acquired quantized representations, resembling the concept in VQ-VAE [193].

Despite the effectiveness of representations learned through contrastive methods in various downstream applications, they encounter several difficulties when applied to speech data. One key challenge lies in how the determination of positive and negative samples can indirectly introduce invariances into the learned representations. Additionally, due to the absence of explicit segmentation in speech input for acoustic units, both negative and positive samples may not correspond to complete language units but instead represent partial or multiple units, depending on the span covered by each sample. Furthermore, the smooth and continuous nature of speech

input makes it challenging to establish a contrastive sampling strategy that consistently provides samples truly indicative of positive and negative relationships with the anchor in a meaningful manner.

2.4.3 Predictive Approaches

Like the contrastive approaches mentioned earlier, predictive approaches define their pretext task based on a learned target. However, in contrast to the contrastive methods, they do not utilize a contrastive loss. Instead, they opt for loss functions such as squared error and cross-entropy. While a contrastive loss helps prevent the model from learning a trivial solution using negative samples, predictive methods handle this challenge differently. For predictive approaches, targets are computed outside the model's computational graph, often with a completely separate model. Consequently, the predictive setup bears similarities to teacher-student training. The initial development of predictive approaches was spurred by the successful BERT-like methods in NLP [194] and the DeepCluster technique in computer vision [172].

Directly applying BERT-type training to speech input faces a challenge due to the continuous nature of speech. An alternative approach, known as Discrete BERT [172], employs a pre-trained vq-wav2vec model to derive a discrete vocabulary [195] by utilizing quantization to learn discrete representations. Discrete BERT showcased the effectiveness of self-supervised speech representation learning by achieving a WER of 25% on the standard test-other subset, setting a valuable precedent for subsequent approaches, despite its reliance on an advanced representation learning model to discretize continuous inputs. On the other hand, the Hidden Unit BERT (HuBERT) approach [196] utilizes quantized MFCC features as targets,

employing classic k-means for learning. Unlike Discrete BERT, HuBERT takes the raw waveform as input, preventing any loss of essential information due to input quantization. It adopts an architecture akin to wav2vec 2.0, involving a convolutional module, a Transformer encoder, and a softmax normalized output layer.

The HuBERT model is able to learn both acoustic and language models because of the way it is trained. First, the model learns to represent unmasked speech frames as continuous values. This is similar to how acoustic models work, where each frame of speech is represented as a discrete unit, such as a phoneme or mel frequency cepstral coefficient (MFCC). Second, the HuBERT model learns to predict masked speech frames by using the context of the surrounding frames. This is similar to how language models work, where the model predicts the next word in a sequence by using the context of the previous words. In other words, the HuBERT model is forced to learn to represent the acoustic features of speech and the long-range temporal dependencies between speech frames. This allows the model to be used for both acoustic and language modeling tasks.

WavLM [148] is a self-supervised pre-trained speech model that is designed to learn spoken content modeling and speaker identity preservation. It is largely similar to HuBERT, but it has two key extensions:

- **Gated relative position bias**: WavLM extends the Transformer self-attention mechanism with a gated relative position bias. This bias is added to the attention weights before they are normalized. The bias is computed based on the input to the Transformer layer at the current time step and also incorporates a relative positional embedding for the difference between the current time step and the time step of the attention target.

- **Utterance mixing**: WavLM uses an utterance mixing strategy to augment the training data. This strategy involves combining signals from different speakers to create new training examples. Specifically, random subsequences from other examples in the same batch are scaled and added to each input example. Only the targets corresponding to the original example are predicted during pre-training. This forces the model to learn to filter out the added overlapping speech.

The iterative process of pre-training for HuBERT and wavLM might pose a logistical challenge, especially when dealing with extensive datasets. Moreover, these models encounter difficulty in ensuring the adequacy of the initial vocabulary derived from MFCC features.

In this chapter, we have explored a variety of ASR models, with a particular emphasis on their application and efficacy in low-resource scenarios (LREs). As we transition to the next chapter, our focus will broaden to include a comprehensive overview and empirical evaluation of state-of-the-art deep learning methods for ASR, methods that are originally designed for high-resource environments (HREs). Our objective is to investigate their utility and adaptability in LREs.

2.5 Low-Resource ASR systems

In this section of the thesis, we provide a comprehensive review of state-of-the-art models addressing low-resource ASR problems. Our focus will focus on the core techniques employed in these models, such as cross-lingual transfer learning, data augmentation, and the use of specialized architectures designed for low-resource settings. In [197], the authors explore the

utilization of self-supervised learning models as a solution to enhance ASR models in languages that lack extensive annotated resources. By leveraging unlabelled audio data, the study demonstrates how self-supervised learning techniques can significantly improve the accuracy and efficiency of ASR systems for low-resource languages. The authors propose a novel framework that combines large-scale pre-trained models with a small amount of labeled data, highlighting the model's ability to generalize from high-resource languages to their low-resource counterparts. The results indicate substantial improvements in speech recognition accuracy, showcasing the potential of self-supervised models in overcoming the challenges associated with linguistic diversity in ASR problems. This approach not only reduces the dependency on extensive labeled datasets but also provides a solution for more inclusive language technologies, enabling better communication and accessibility across different language speakers. In [198] the authors address the critical role of ASR technologies in preserving and documenting endangered languages. ASR system presents a valuable tool for the urgent task of documenting endangered languages. While traditional ASR methods rely heavily on transcribed data and phonetic dictionaries (scarce for low-resource languages), recent advancements in end-to-end ASR systems offer a solution. These systems, powered by self-supervised representation learning, can effectively utilize large corpora of untranscribed speech data. This technical approach significantly reduces the need for manual annotation by linguists, accelerating the documentation process and safeguarding the linguistic and cultural heritage embodied within endangered languages. An innovative approach to enhancing ASR systems for low-resource languages through the use of high-resource language transliteration models proposed in [199]. Instead of directly training an ASR model for a low-resource language, the authors propose transliterating high-resource language text into the script of the target low-resource language. This

allows for pre-training on large datasets from the high-resource language, effectively boosting the low-resource ASR model's performance. Surprisingly, this technique proves effective even when the languages come from unrelated families, and the authors demonstrate significant gains in ASR accuracy, particularly in very low-resource scenarios. By converting large-scale English speech transcriptions into the script of the target language, the authors create a robust initialization for End-to-End models. After pretraining on this transliterated data, models are fine-tuned using limited speech samples from the target language. This approach has proven more effective than standard transfer learning techniques for various languages. The enhanced performance likely stems from the way transliteration forces English transcriptions to share model parameters across encoder and decoder layers, promoting better cross-lingual adaptation.

A novel approach to enhance ASR systems in languages with limited linguistic resources presents in [200]. The work introduces a methodical approach to mine and utilize audio-text pairs from publicly available data sources, including radio broadcasts, podcasts, and internet videos. By mining audio and text pairs from public sources, specifically the archives of All India Radio, the authors create a dataset called Shrutilipi. This dataset contains over 6,400 hours of labeled audio across 12 Indian languages. The solution involves adapting the Needleman-Wunsch algorithm to align sentences with corresponding audio segments, even when dealing with errors due to OCR, extraneous text, and non-transcribed speech. Integrating Shrutilipi into ASR training significantly reduces the word error rate for several languages, including Hindi. In [201], the authors introduce an effective approach to enhance direct speech-to-text translation (ST) for low-resource languages. The method involves pre-training a model on a high-resource ASR task and subsequently fine-tuning its parameters for

ST. By pre-training on 300 hours of English ASR data, the authors significantly improve Spanish-English ST from a BLEU score of 10.8 to 20.2, even when only 20 hours of Spanish-English ST training data are available. Notably, the pre-trained encoder (acoustic model) plays a crucial role in this improvement, despite the shared language being the target text rather than the source audio. Furthermore, the approach remains effective even when the ASR language differs from both the source and target ST languages, as demonstrated by pre-training on French ASR to enhance Spanish-English ST. Finally, the method proves beneficial for a true low-resource scenario, where pre-training on a combination of English and French ASR data improves Mboshi-French ST from 3.5 to 7.1 BLEU, using only 4 hours of available data. Deep maxout networks (DMNs) explores in [202] to improve ASR systems when dealing with low-resource languages (i.e., languages with limited amounts of transcribed speech data). DMNs are a type of neural network that use the maxout activation function, allowing them to learn more complex representations. The authors demonstrate that DMNs are particularly well-suited for low-resource ASR to reduce model size and their compatibility with the dropout regularization technique. The authors extend DMNs to hybrid and bottleneck feature systems, exploring optimal network structures. On the Babel corpus, DMNs significantly improve low-resource speech recognition. Additionally, DMNs introduce sparsity to hidden activations, acting as sparse feature extractors. MetaASR [203] proposes a novel approach to enhance ASR systems in low-resource scenarios. The authors treat ASR tasks for different languages as distinct tasks and apply meta learning to initialize model parameters from a variety of pretraining languages. Specifically, they employ the Model-Agnostic Meta Learning (MAML) algorithm to achieve rapid adaptation on unseen target languages. By evaluating their approach using six languages as pretraining tasks and four languages as target tasks, they demonstrate that

their method significantly outperforms state-of-the-art multitask pretraining approaches across various combinations of pretraining languages. In [204], the authors investigate the application of Wav2vec2.0 model to enhance speech recognition in low-resource scenarios. While Wav2vec2.0 has demonstrated its powerful representation ability on the Librispeech corpus, which belongs to the audiobook domain, this work extends its evaluation to real spoken scenarios and languages beyond English. The authors apply pre-trained models to solve low-resource speech recognition tasks in various spoken languages. Remarkably, they achieve more than 20% relative improvements in six languages compared to previous work, with English showing a gain of 52.4%. Additionally, using coarse-grained modeling units (such as subword or character) yields better results than fine-grained units (like phone or letter).

In [205], the authors explore a novel speaker augmentation approach to improve ASR systems in scenarios where there are limited data resources, especially regarding speaker variability. In the context of low-resource tasks, where the diversity of speakers in the training data is limited, the authors propose the use of a text-to-speech (TTS) system trained with speaker representations from a variational autoencoder (VAE). This system is capable of synthesizing speech data with a wide range of speaker and text diversity, significantly enhancing the robustness and accuracy of ASR systems. By synthesizing data from unseen speakers, the approach enables the generation of diverse training datasets that were previously unachievable. Their experiments, conducted on a Switchboard task with only 50 hours of data, demonstrate a remarkable reduction in the WER by 30% relative to systems without data augmentation, and by 18% relative to systems using only traditional feature augmentation approaches like SpecAugment. Cross-Lingual Self-Training (XLST) [206] is a novel pre-training framework,

aimed at enhancing multilingual representation learning for speech recognition in low-resource settings. Recognizing the challenge posed by data scarcity in training ASR systems, especially for languages with limited annotated data, XLST leverages a small amount of annotated data from a non-target language alongside large volumes of un-annotated multilingual data. The method initially employs a phoneme classification model trained on the non-target language data to generate initial targets. Subsequently, it trains another model on the multilingual un-annotated data, focusing on maximizing frame-level similarity between the output embeddings of the two models. Incorporating mechanisms such as moving average and multi-view data augmentation, the framework significantly reduces phoneme error rates in downstream speech recognition tasks across five low-resource languages when compared to state-of-the-art self-supervised methods. In [207] the authors investigate the potential of leveraging unsupervised speech representation learning from noisy radio broadcasting archives to develop ASR systems for low-resource languages. Targeting the 700 million illiterate individuals worldwide, the study introduces a novel approach to bridging the digital divide by enhancing the availability of speech recognition technology for languages spoken by illiterate populations, who are often the most underserved. The authors release two valuable datasets: the West African Radio Corpus, with 142 hours of audio in over 10 languages, and the West African Virtual Assistant Speech Recognition Corpus, consisting of 10K labeled audio clips in four languages. By training a speech encoder on these datasets, they demonstrate comparable performance to baseline models trained on higher-quality data for multilingual speech recognition tasks and superior performance for language identification tasks. The authors in [208] propose a comprehensive approach that involves optimizing the use of available data through several innovative methods. They introduce strategies for multilingual speech recognition, emphasizing the importance

of leveraging similarities and correlations between languages. The techniques proposed include utilizing the posterior of the target language from a language classifier for data weighting, dynamic curriculum learning for effective data allocation, and length perturbation for data augmentation. These methods collectively form a new strategy aimed at enhancing data usage efficiency for languages with scarce resources. The evaluation of the proposed methods on datasets like CommonVoice and Babel demonstrates significant improvements in ASR performance, showcasing reductions in word error rates and character error rates for various target low-resource languages.

MixSpeech [209] is an innovative adaptation of the mixup technique, which involves creating augmented data by blending pairs of input speech sequences (e.g., mel-spectrograms or MFCC) and their corresponding textual sequences with a certain weight, resulting in a model that is trained to recognize both sequences simultaneously. This approach simplifies the data augmentation process, requiring only a single hyper-parameter for weight combination, unlike the more complex SpecAugment that necessitates careful tuning of multiple parameters. The effectiveness of MixSpeech is demonstrated through experiments on several low-resource ASR datasets including TIMIT, WSJ, and HKUST, where it not only outperforms baseline models without augmentation but also shows superior performance compared to the SpecAugment method, achieving significant improvements in terms of Phone Error Rate (PER) and WER. Task-based Meta Poly-Loss (TMPL) [210] is a novel method designed to optimize multilingual meta-learning for low-resource speech recognition. TMPL addresses the misalignment between the loss functions and the learning paradigms of meta-learning by treating speech recognition tasks as samples and employing PolyLoss as the meta-loss function. This approach enables TMPL

to serve as a linear combination of polynomial functions based on task query loss, thus allowing for tailored attention adjustment across different tasks to accommodate various datasets. The study provides a theoretical analysis demonstrating how TMPL enhances meta-learning capabilities through this adaptive attention mechanism. Experimental validation across multiple datasets shows that TMPL significantly outperforms conventional gradient-based meta-learning methods, effectively mitigating the misalignment issue and improving speech recognition accuracy in low-resource languages. MUST [211] introduces a novel framework designed to enhance ASR systems for low-resource languages through a multilingual student-teacher learning approach. This approach addresses the limitations of conventional knowledge distillation methods, which require the student model's classes to be a subset of the teacher model's classes, a requirement that restricts the utilization of acoustically similar languages with differing character sets. MUST overcomes this by employing a posterior mapping model that translates the posteriors from a teacher language to the student language ASR, thereby allowing the use of these transformed posteriors as soft labels for knowledge distillation learning. The study experiments with various teacher ensemble schemes and demonstrates that the MUST learning approach can significantly reduce the relative CER by up to 9.5% compared to baseline monolingual ASR systems. Hidden Unit Clustering (HUC) framework [212] presents a novel approach to self-supervised representation learning from raw audio. This method focuses on generating semantically rich speech representations by categorizing the outputs of a neural network into a limited number of phoneme-like units. The process involves windowing audio samples, processing them through convolutional and LSTM layers to create contextual vector representations, and then clustering these representations to train the model. The paper demonstrates the effectiveness of this approach through experiments on

low-resource speech applications, specifically within the ZeroSpeech 2021 challenge and on datasets like TIMIT and GramVaani.

Chapter 3

Neural Networks Architecture Analysis for Low-Resource ASR

In recent years, significant strides within the realm of deep learning have reignited interest in integrating the fundamental components of speech recognition systems into a unified end-to-end model. The primary goal of such a model is to directly translate input audio sequences into the corresponding output text sequences [203]. However, it is important to note that the present state-of-the-art methods for training these end-to-end models demand a substantial volume of annotated data to achieve optimal performance [203]. Consequently, it is evident that to construct a highly accurate and resilient automatic speech recognition system for a new domain or application, amassing a large dataset of recorded and transcribed speech, often referred to as a spoken corpus, is imperative [31].

The growing interest in end-to-end models for speech recognition lies in their potential to streamline complex systems by directly mapping audio

to text, promising improved efficiency and easier integration. However, the major challenge is the need for extensive, accurately labeled training data, which is labor-intensive and resource-intensive to obtain. This underscores the critical importance of large recorded and transcribed speech datasets, especially when expanding into new domains, highlighting the need for comprehensive spoken corpora to achieve robust and precise system performance.

This is a challenging prospect for those interested in carrying out research in domains where large amounts of data for training are not available. We define low-resource environments as environments where the lack of sufficient amount of training data diminishes the performance of the ASR system. Examples of this include domains such as new or less wide-spread languages (e.g., the Kyrgyz language) [31], domains in which a highly technical or specific language is required (e.g., a chemical plant, or a surgery theatre), child speech recognition [33], speakers with speech disorders (e.g., dysarthria) [31], or speakers with accents are all examples of low-resource environments. In these environments, to the best of our knowledge, there are very limited suitable public corpora for training purposes.

This indicates the need to understand the various perceptions of low-resource environments that exist in ASR systems. As shown in the previous chapter, most state-of-the-art models need more than 2k hours of transcribed audio as training data [31]. Such requirements are simply unattainable in low-resource environments. And, as our experiments in this thesis show, benchmark corpora and models architecture prove to be insufficient to achieve robust ASR systems using models designed for high resource environments. Consequently, special attention should be considered when developing low-resource ASR systems to account for such limited training data. In this chapter, we analysis different neural network architecture for

low-resource setting to evaluate their performance in such problems.

3.1 Datasets

In the preceding section, different ASR techniques were discussed, which need a large amount of data for training. This section provides an overview of the datasets that can be used to train and evaluate ASR models. Furthermore, we present our own low-resource dataset, I-CUBE, which has been used to test state-of-the-art ASR techniques in low-resource task.

3.1.1 Datasets for HRE ASR Task

In this section, we summarize datasets which provide speech and the corresponding transcripts, speaker labels, or a large amount of speech data but with limited or no labels.

Librispeech [27] is a large-scale corpus (more than 1k hours) of read English speech that has been widely used to train and evaluate ASR tasks. This corpus is created from audio-books that are part of the LibriVox project and contains more than 2,000 hours of speech sampled at 16 kHz [27]. Speakers in Librispeech are divided based on lower-WER speakers and higher-WER speakers, which are cleaned and pre-processed. The training portion of the corpus is divided into three subsets with approximate size of 100, 360, and 500 hours.

The Wall Street Journal (WSJ) corpus [30] consists of speaker-independent (SI) read material, divided into training, development test, and evaluation test sets. WSJ has 90 utterances from each of the 92 speakers that are designated as training material for speech recognition models. A further 48 speakers each read 40 sentence utterances containing only words from a fixed 5,000-word vocabulary of 40 sentences from the 64,000-word vo-

cabulary, which will be used as testing material. Each of the total of 140 speakers also recorded a common set of 18 adaptation sentences. Standard close talking and multiple secondary microphones and equal numbers of male and female speakers are important considerations to support the diversity of voice quality and dialect [30]. All recorded materials were taken from the WSJ text corpus and recorded in a clean environment using close-talking microphones.

Fisher corpus [36] is based on the Fisher telephone conversation collection protocol, which was proposed by the Linguistic Data Consortium (LDC). Fisher data collection asked participants to speak on an assigned topic that was randomly selected from a list that changed periodically. This strategy allowed them to cover a large vocabulary [36]. The main purpose of the data collection protocol in Fisher was to be able to produce over 2k hours of conversational speech data from calls. After 11 months, LDC was able to collect 16,454 calls, with an average of 10 minutes in duration, totalling 2,972 hours of audio [36]. In the Fisher, 53% of calls were made by females, with 38% of subjects aged between 16 and 29, 45% aged between 30 and 49, and 17% aged over 50 [36].

VoxCeleb [29] is a large-scale speaker identification and audio-visual dataset which contains around 100,000 utterances of 1,251 celebrities, short clips of human speech, extracted from interview videos uploaded to YouTube. VoxCeleb has about 2000 hours of speech. VoxCeleb is gender balanced in which 55% of speakers are male and selected from a wide range of different ethnicity, accents, professions, and ages.

TED-LIUM is a corpus that contains audio transcriptions of TED talks. TED-LIUM is presented in 3 different versions which are TED-LIUM Release 1 [213], TED-LIUM Release 2 [26] and TED-LIUM Release 3 [25].

TED-LIUM Release 3 contains 2351 audio talks in NIST sphere format (SPH) and includes talks from TED-LIUM Release 2 and 452 hours of audio.

Common Voice (CV) [214] is an open-source dataset that aims to provide a diverse collection of speech recordings from speakers of different ages, genders, and accents, in order to support the development of more inclusive and accurate speech recognition systems. As of version 7.0, the Common Voice corpus contains approximately 11,000 hours of audio in 76 different languages. The dataset is constantly growing, as new recordings are contributed by volunteers. In addition to the audio recordings, the Common Voice corpus also includes metadata about the recordings, such as the speaker's age, gender, and accent, as well as information about the recording environment and any background noise that may be present. This metadata can be useful for training and evaluating speech recognition models, particularly those that aim to be robust to variations in speaker and acoustic conditions.

3.1.2 Datasets for LRE ASR Task

A low-resource speech recognition dataset refers to a collection of speech data that is limited in size, quality, or diversity, making it challenging to train robust speech recognition models. Low-resource speech recognition datasets are particularly challenging for ASR systems because they may lack sufficient examples of rare or out-of-vocabulary words or may contain significant amounts of noise or speaker variation, which can lead to poor recognition performance.

TORGO [215] is a low-resource dataset which contains approximately three

hours of speech. TORGO consists of aligned acoustic recordings from 15 speakers, including 7 control speakers without any disorder and 8 speakers with different levels of dysarthria. Speakers were asked to read single words or sentences and describe the content of some photos. A total of, 5980 and 2762 utterances were recorded from healthy and dysarthric speakers, respectively.

Nemours [216] database is a low-resource speech collection of 74 short sentences spoken by 11 speakers with varying degrees of dysarthria, resulting in a total number of 814 recordings. Furthermore, Nemours contains two connected speech paragraphs, which are produced by each of the 11 speakers.

UASpeech [217] database is the largest corpus of dysarthric speech in American English. It is a collection of 541 read speech recordings from 19 individuals with cerebral palsy. The prompt words include three repetitions of the first ten digits, three repetitions of 26 radio alphabet letters, three repetitions of 19 computer commands, common words from the 'Grandfather Passage', and uncommon words from phonetically balanced sentences one time each.

3.1.3 I-CUBE: a Human-Robot Interaction Dataset

I-CUBE is a Human-Robot collaboration dataset which collected during the first experimental phase of the I-CUBE project (Industrial Co-Bots Understanding Behavior). In this experimental phase, participants were asked to interact using natural language, such as speech, facial expressions, and gestures, with an actor who posed as a robot. They had to instruct and ultimately teach this robot how to sort different garments into four baskets

as if they were sorting their own laundry. During the experiments, the robot would also respond to the participant's actions with its own actions or speech. Video recordings of each session were collected, resulting in a total of 42 videos, which is more than 300 minutes of video footage, including audio.

3.2 Experiments

In this section, we describe the series of comprehensive experimental evaluations we carried out to sufficiently investigate the performance of state-of-the-art HRE approaches in ASR systems in low-resource environments. Section 3.2.1 describes the methodology used to evaluate the different models. Evaluation metrics are presented in Section 3.2.2.

3.2.1 Evaluation Protocol

Our evaluation protocol was designed to obtain evidence on the two questions we present at the beginning of our study. Namely, given a low-resource environment:

- What is the performance achieved by training models using only high-resource benchmark data and testing on low-resource datasets?
- What is performance benefit achievable by pre-training with high-resource benchmark data and fine-tuning the trained model with low-resource data?

We selected two well-known benchmark datasets for pre-training of different ASR methods: Libripeech [27] and WSJ [30]. These datasets are two com-

mon for the evaluation of the high-resource ASR systems. Both datasets have similar characteristics: multiple speakers, clean read speech (sourced from texts) recorded at a sampling rate of 16 kHz [218]. To analyse different models in low-resource task, we selected I-CUBE and UASpeech which used to test models after pre-training them on Librispeech and WSJ. The UASpeech dataset can be considered a low-resource dataset due to its limited size and specificity. In terms of models, we chose different network architectures that have obtained state-of-the-art results in the last few years. In terms of LSTM-based networks, we trained and tested LSTM, BLSTM, ltLSTM [82], cltLSTM [22] and Residual LSTM [79, 80] networks. For each of these methods, we trained each network with 2, 4, 6, and 8 layers. Furthermore, different structures of the concatenated 2-layer LSTM and fully connected feedforward neural network are examined to show their performance. The composition of convolutional neural networks and 2-layer LSTM and 2-layer GRU has also been examined to present in detail the possibility of such architectures for low-resource ASR systems. Furthermore, We selected the basic model of the Transformer with 6 Encoders and 6 Decoders to examine it in the low-resource environment. Finally, we have selected QuartzNet, wav2vec 2.0 and HuBERT models which showed promising results in ASR tasks.

We consider two training scenarios:

- Train all models from scratch using Librispeech and WSJ separately. Test on the relevant dataset, UASpeech and I-CUBE.
- Pre-train all models from scratch using Librispeech and WSJ separately. Fine-tune with the UASpeech and I-CUBE datasets. Finally, test on UASpeech and I-CUBE.

In both scenarios, we applied 10-fold cross-validation during training, and reported average results with standard deviations. For the pre-training scenario, we split the I-CUBE and UASpeech datasets into ten folds and, in each iteration, nine folds were used as the fine-tuning data for the trained model and the remaining fold as the test set. To ensure that all folds are tested, ten iterations are performed. The output alphabet of the target text consisted of 31 classes and 26 lowercase letters.

Since we also wanted to focus on the role of the amount of data during training and its effects, we trained all models in both scenarios with 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90% and 100% of training data in both datasets (LibriSpeech and WSJ).

Finally, to focus on verifying the models with a fair comparison, for all methods which are based on stacked LSTM layers, we use 1024 hidden units, and the output of each LSTM layer is reduced to 512 using a linear projection layer. Furthermore, to examine the FNN methods, we use *tanh(.)* as the activation function for the hidden layer and the *softmax* function for the output layer. As a pre-processing step, we compute Mel Spectrograms to convert the input raw audio into the ASR model. we use a frame length of 25 ms, a hop length of 10 ms, target a frequency range up to 16 kHz, apply 80 mel filters. We use the $AdamW$ optimizer [219] as a hyperparameter setting with an initial learning rate of 0.001. Different models will be trained to predict the probability distribution of all characters in the alphabet by using CTC loss function.

3.2.2 Metrics

We present the results according to two metrics:

Word Error Rate (WER) : WER is a standard metric for measuring ASR performance. WER is a word-level measure which takes the predicted transcription of the model and the ground truth transcription, and measures the Levenshtein distance. Levenshtein distance is a minimal number of insertions, deletions, and substitutions of words for the conversion of a hypothesis to a reference [220]. This metric is calculated as follows:

$$WER = SubstitutionError + InsertionError + DeletionError$$

$$\begin{aligned} SubstitutionError &= \frac{Numberofsubstitutionerrors}{Numberofgroundtruthwords} \\ InsertionError &= \frac{Numberofinsertionerrors}{Numberofgroundtruthwords} \\ DeletionError &= \frac{Numberofdeletionerrors}{Numberofgroundtruthwords} \end{aligned} \tag{3.1}$$

WER is normally reported as a percentage. [220].

Character Error Rate (CER) : CER is an important metric in ASR system evaluation. CER measures the error of the characters between the predicted transcription of the model and the ground-truth transcription. The calculation of the CER is similar to the WER, but it is a character-level measure.

As we carried out cross-validation during our evaluation, we report on average WER and CER across all folds, along with their associated standard deviation.

3.3 Evaluating Pre-trained ASR Models on Low-Resource Speech Datasets

In this section, we present the results obtained by examining different ASR methods on Librispeech and WSJ. We pre-train different models with different percentage of data and then test with low-resource I-CUBE and UASpeech data. The best WERs obtained from each model trained on Librispeech and WSJ, and tested on I-CUBE are summarized in Table 3.1. Complete results based on different numbers of layers and percentage of data are explicitly listed in Appendix A.

Among LSTM with different numbers of layers (refer to Appendix A for complete comparison), the 6-layer LSTM model performs the best when trained with 100% of the data, achieving WERs of 24.28% and 24.17% on WSJ and Librispeech, respectively. The WER of the model trained with 10% of the WSJ dataset is, 45.79% while it is 45.23 for the Librispeech. When the amount of data increases from 10% to 20%, WERs decrease 5.61% for the WSJ and 4.06% for Librispeech. A significant improvement in WER of 10.15% occurrs when the amount of the data increases from 30% to 40% in WSJ while this improvement is 8.64% for Librispeech.

The 6-layer ResLSTM outperforms other configurations in terms of the number of layers (see Appendix A for complete comparison) achieving WERs of 25.13%, 24.14% when trained with 100% of WSJ and Librispeech datasets, respectively. Increasing the amount of pre-trained WSJ data from 40% to 50% results in a significant improvement, reducing the WER by 10.07%, while the WER for LibriSpeech is improved by 9.84%.

The 6-layer ltLSTM model outperformed other layer configurations in ltLSTM, as well as the previous two models (see Appendix A for the complete

		10%	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
6-layer LSTM	WSJ	45.79	43.22	40.38	36.28	32.61	31.48	29.12	27.86	25.53	24.28
	LibriSpeech	45.23	43.39	41.18	37.62	34.21	32.78	30.67	27.49	25.31	24.17
6-layer ResLSTM	WSJ	45.91	43.87	40.92	37.42	33.65	32.71	29.83	28.29	26.94	25.13
	LibriSpeech	45.68	43.62	41.34	37.57	33.87	32.15	30.18	27.13	24.91	24.14
6-layer ltLSTM	WSJ	45.83	43.51	39.41	35.94	32.35	30.52	28.97	27.18	25.09	24.11
	LibriSpeech	45.39	43.53	40.91	37.29	33.62	31.87	28.73	26.42	24.68	24.08
6-layer cltLSTM	WSJ	45.69	43.19	39.27	35.67	32.14	30.27	28.59	26.76	24.71	23.91
	LibriSpeech	45.28	43.27	39.17	36.73	33.41	31.69	30.42	28.03	26.13	26.18
FNN+2-layer LSTM	WSJ	51.38	49.71	46.13	43.19	41.58	39.75	37.13	35.81	34.69	33.97
	LibriSpeech	50.39	49.13	45.48	42.61	40.13	38.79	35.41	33.29	31.63	29.34
2-layer LSTM+FNN+ 2-layer LSTM	WSJ	51.69	50.37	47.62	44.51	41.85	40.19	38.61	37.61	35.82	34.88
	LibriSpeech	50.73	50.21	46.73	43.67	41.76	40.08	38.19	35.83	33.12	30.39
2-layer LSTM+FNN+ FNN	WSJ	51.53	50.18	47.27	44.23	41.31	39.87	37.63	35.47	34.81	33.92
	LibriSpeech	50.47	49.89	46.91	43.39	41.23	39.72	37.21	34.37	32.59	29.18
2-layer BLSTM	WSJ	44.74	42.21	40.28	36.93	33.81	31.12	29.89	28.31	26.83	26.07
	LibriSpeech	44.58	41.98	40.19	37.28	34.93	33.96	30.38	29.13	28.07	27.18
1-D CNN+2-layer BLSTM	WSJ	44.92	42.53	40.63	37.61	34.28	31.57	30.31	28.91	27.12	26.19
	LibriSpeech	45.12	42.39	40.51	37.59	35.42	34.21	30.82	29.89	28.69	27.63
1-D CNN+2-layer LSTM	WSJ	47.38	45.61	43.12	41.17	38.62	35.58	33.62	31.92	29.71	27.22
	LibriSpeech	48.17	46.21	43.69	41.58	39.27	36.21	34.87	33.47	30.19	28.78
1-D CNN+2-layer GRU	WSJ	47.69	46.13	44.33	41.89	39.58	36.71	33.89	31.29	29.48	27.13
	LibriSpeech	48.81	47.21	45.31	42.43	39.81	36.32	33.51	31.87	29.64	28.63
QuartzNet	WSJ	44.27	42.18	38.85	35.39	31.98	30.11	28.11	25.98	23.89	22.85
	LibriSpeech	44.11	41.53	38.32	35.79	31.72	29.93	27.98	25.65	23.51	22.13
Transformer	WSJ	43.78	41.28	38.31	34.97	31.52	29.92	27.71	25.61	23.46	22.32
	LibriSpeech	42.19	40.87	37.92	35.32	31.49	29.67	27.43	25.29	23.19	21.27
wav2vec 2.0	WSJ	38.15	36.92	34.15	31.18	29.15	27.34	25.83	24.91	22.74	21.65
	LibriSpeech	36.83	35.13	33.98	30.29	28.51	26.12	24.13	23.28	21.29	20.41
HuBERT	WSJ	38.49	36.87	34.11	30.92	29.05	27.10	25.51	24.70	22.62	21.52
	LibriSpeech	36.95	35.05	33.58	30.13	28.17	25.93	24.03	23.12	21.30	20.15

Table 3.1: Best WER results for each model when pre-training on WSJ and Librispeech and testing on I-CUBE. The columns of the table denote the percentage of pre-training data used.

comparison). This model achieved WERs of 24.11% and 24.08% when trained with 100% of the data from both datasets. The WER improved by 9.98% when the data increased from 40% to 50% on WSJ. Although a slight improvement of 2.43% occurred in Librispeech when the model received 10% more data from 90% to 100%. The same layer configuration in cltLSTM outperforms all other stack LSTM structures. This model achieved a WER of 23.91% and 26.18% based on the 6 number of layers in its structure on WSJ and Librispeech datasets, respectively. Interestingly, this improved WER in cltLSTM is related to the use of future context frames.

In our results, we evaluate three different configurations of the 2-layer LSTM and the fully connected feedforward neural network. The first structure, called FNN-LSTM, is created by cascading 2-layer LSTM after FNN.

This model achieved WERs of 33.97% and 29.34% when trained on WSJ and Librispeech. This model achieved a WER of 51.38% when it received 10%5 of the WSJ data and improved by 19.07% when increasing the data amount to 50%. Meanwhile, increasing the amount of data from 60% to 70%, the WER got 8.71%better.

In the second configuration, we insert a FNN layer between two 2-layers of LSTMs, called LSTM-FNN-LSTM, to present another architecture for the combination of the LSTM and FNN. This structure obtained 34.01% and 30.39% WERs on WSJ and Librispeech, respectively. Finally, we create a different model by cascading two FNN layers after one 2-layer LSTM, called LSTM-FNN-FNN, which achieved WERs of 33.92% and 29.18% on WSJ and Librispeech, respectively. The LSTM-FNN-FNN structure achieved 7.63% WER improvement when its data increased from 70% to 80%. In combination of the LSTM and FNN layers, the LSTM-FNN-FNN structure obtained a better WER than the other models.

We evaluated the performance of 2-layer BLSTM on Librispeech and WSJ and achieved 27.18% and 26.07% WERs, respectively. The BLSTM starts with WER 44.74% in the WSJ and 44.65% in the Librispeech datasets. The 2-layer BLSTM is close to the 2-layer LSTM in terms of WERs and gets better performance. By inserting a 1-D CNN layer before the 2-layer BLSTM, WER increased and achieved 26.19% and 27.63% in WSJ and Librispeech, respectively. The WER of the 1-D CNN and 2-layer BLSTM are 44.92% and 45.12%, respectively. Based on 10% of the data, by adding a 1-D CNN layer before 2-layer BLSTM, WER decreased by 0.4%. Furthermore, cascading 1-D CNN before 2-layer LSTM, due to increased WER on this model, we obtained WERs of 27.22% and 28.78% on WSJ and Librispeech. We see that a hybrid model by combining the 1-D CNN and 2-layer LSTM has similar performance. In addition, we added a 2-layer

GRU after 1-D CNN and achieve WERs of 27.13% and 28.63% on those two datasets. What is interested in this type of hybrid model is that the combination of the CNN with BLSTM outperforms of the other CNN combinations.

In addition, to investigate the effects of utilizing a large amount of pre-training data on the Transformer model in a low-resource environment, we examined the base model of the Transformer with 6 Encoders and 6 Decoders. This model achieved WERs of 22.32% and 21.27% when trained on WSJ and Librispeech, respectively. In this model, by increasing the Librispeech data amount from 40% to 50%, WER improved by 8.89%, while this improvement on WSJ was 7.75%. Compared with other previous models, Transformer achieves a large margin improvement.

Finally, we examine the same strategy for pre-training for QuartzNet, wav2vec 2.0 and HuBERT models. wav2vec 2.0 and HuBERT have similar WERs and outperform QuartzNet. HuBERT obtained 21.52% and 20.15% WERs on WSJ and Librispeech, respectively. While wav2vec 2.0 achieved 21.65% WER on WSJ and 20.14% WER on Librispeech.

Similar results are presented in Table 3.2 when the same pre-trained models are tested on the UASpeech dataset.

Figure 3.1, 3.2 shows the obtained CER on after training different models on WSJ and Librispeech datasets and testing with I-CUBE and UASpeech, respectively. On both datasets, Transformer obtained the best results and at each stage the CER is improved by increasing the amount of training data.

These experiments showed that increasing the amount of data can improve the performance of the ASR system in different architectures, however the

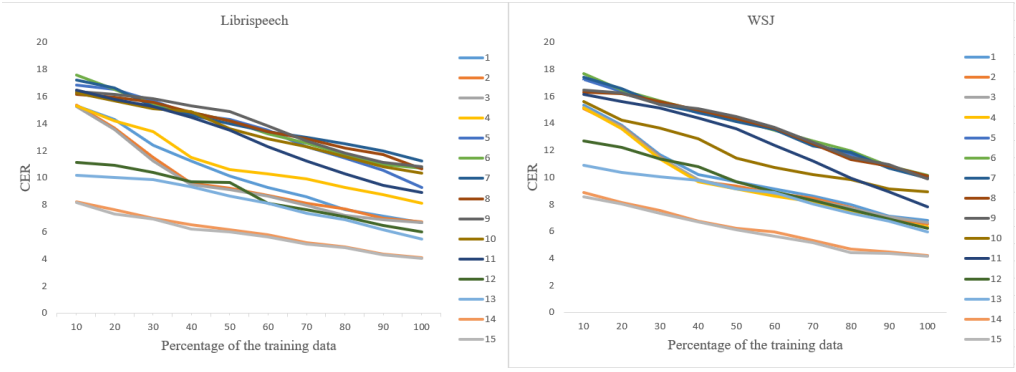


Figure 3.1: CER in percentage for models trained with WSJ and Librispeech and tested with I-CUBE. 1: 6-layer LSTM, 2: 6-layer ResLSTM, 3: 6-layer ltLSTM, 4: 6-layer cltLSTM, 5: FNN+2-layer LSTM, 6: 2-layer LSTM+FNN+2-layer LSTM, 7: 2-layer LSTM+FNN+FNN, 8: BLSTM, 9: CNN+2-layer BLSTM, 10: CNN+2-layer LSTM, 11:CNN+2-layer GRU, 12: Transformer, 13:QuartzNet, 14:wav2vec 2.0, 15:HuBERT.

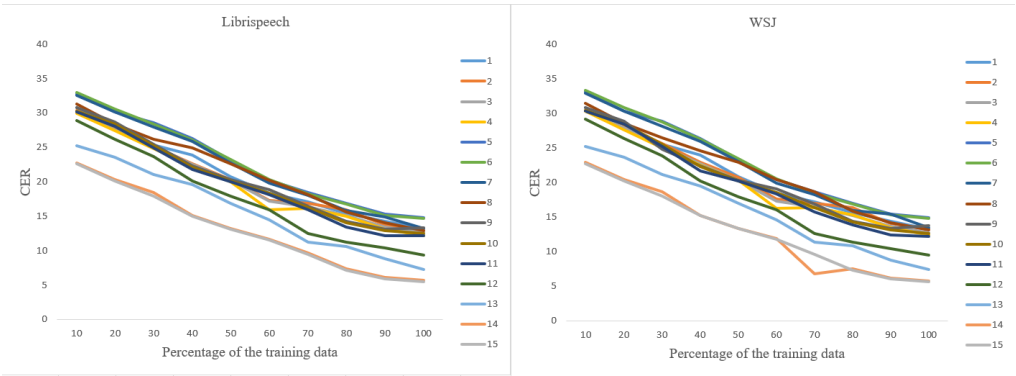


Figure 3.2: CER in percentage for models trained with WSJ and Librispeech and tested with UASpeech. 1: 6-layer LSTM, 2: 6-layer ResLSTM, 3: 6-layer ltLSTM, 4: 6-layer cltLSTM, 5: FNN+2-layer LSTM, 6: 2-layer LSTM+FNN+2-layer LSTM, 7: 2-layer LSTM+FNN+FNN, 8: BLSTM, 9: CNN+2-layer BLSTM, 10: CNN+2-layer LSTM, 11:CNN+2-layer GRU, 12: Transformer, 13:QuartzNet, 14:wav2vec 2.0, 15:HuBERT.

WERs tend to be improved when models are tested on LRE datasets. Our second series of experiments aims demonstrate the performance of the models when the training and testing data are from the same domain. Therefore, we use LibriSpeech and WSJ datasets to train and test the models. The best results based on the number of layers and the percentage of data in each model are presented in Table 3.3. The complete results are listed in the Appendix A.

The performance of the 6-layer LSTM improved in terms of WERs by 13.77% on WSJ and 13.41% on LibriSpeech when the amount of data increased from 10% to 20%. The improvements are 17.65% and 12.37%

		10%	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**	**100%**
6-layer LSTM	WSJ	65.71	59.83	54.48	50.11	47.45	44.51	41.19	39.32	37.12	36.27
	LibriSpeech	63.39	58.72	52.21	49.39	47.35	43.52	39.74	37.89	36.28	35.94
6-layer ResLSTM	WSJ	64.15	59.98	65.51	52.39	48.18	43.37	42.71	40.28	38.51	38.19
	LibriSpeech	63.57	59.38	55.32	51.49	48.61	42.58	41.39	39.15	38.93	37.17
6-layer ltLSTM	WSJ	64.89	59.13	54.23	50.17	47.13	43.78	40.85	38.87	36.57	36.15
	LibriSpeech	63.29	59.10	52.87	49.31	46.93	43.12	38.87	37.12	36.08	35.29
6-layer cltLSTM	WSJ	64.21	59.43	54.21	50.08	47.21	43.27	40.92	38.68	36.19	35.83
	LibriSpeech	63.82	58.39	52.47	48.89	46.39	42.27	38.23	36.89	35.91	35.07
FNN+2-layer LSTM	WSJ	69.77	65.18	62.21	58.63	54.21	50.18	49.51	47.21	45.38	42.39
	LibriSpeech	68.21	64.33	61.79	58.27	53.97	49.87	48.93	47.15	45.31	41.75
2-layer LSTM+FNN+	WSJ	69.31	65.39	62.89	58.71	54.39	50.48	49.75	47.53	45.42	42.61
2-layer LSTM	LibriSpeech	68.57	64.69	61.72	58.33	54.12	50.27	49.11	47.28	45.52	42.25
2-layer LSTM+FNN+	WSJ	69.15	64.98	62.21	58.17	53.97	49.37	48.83	46.93	45.21	41.87
FNN	LibriSpeech	68.78	63.51	60.15	57.34	53.28	48.78	47.51	46.22	44.89	41.53
2-layer BLSTM	WSJ	65.93	59.71	54.68	50.39	47.58	44.12	41.28	39.21	36.83	36.39
	LibriSpeech	64.28	59.41	53.17	49.53	47.12	43.67	39.17	37.65	36.47	35.57
1-D CNN+2-layer BLSTM	WSJ	65.83	59.69	54.83	50.89	48.13	44.28	41.51	39.35	37.05	36.57
	LibriSpeech	63.91	58.98	53.39	49.88	47.78	44.29	39.28	38.12	36.58	36.02
1-D CNN+2-layer LSTM	WSJ	66.78	61.93	55.18	51.12	48.83	44.87	42.18	41.53	37.71	36.98
	LibriSpeech	64.89	59.65	54.12	50.87	47.79	45.17	40.54	38.93	37.62	36.58
1-D CNN+2-layer GRU	WSJ	66.65	61.83	55.28	52.87	48.93	45.91	42.31	41.87	38.12	37.85
	LibriSpeech	64.51	59.95	54.39	51.09	48.08	45.83	40.97	39.51	36.98	36.83
QuartzNet	WSJ	60.12	55.58	50.49	47.21	44.89	40.28	37.39	35.27	33.95	32.83
	LibriSpeech	58.83	54.39	49.87	46.95	43.18	39.28	36.33	34.87	32.28	31.98
Transformer	WSJ	58.39	54.37	49.31	46.98	42.57	39.83	36.41	34.12	32.74	31.29
	LibriSpeech	57.64	52.97	48.51	45.21	41.28	36.95	34.19	33.75	30.83	30.48
wav2vec 2.0	WSJ	51.75	47.28	45.35	43.65	40.81	37.63	35.49	33.71	30.28	28.23
	LibriSpeech	50.29	48.78	44.83	42.92	39.61	36.53	34.74	32.89	29.71	27.65
HuBERT	WSJ	51.78	47.20	45.29	43.60	40.72	37.60	33.45	31.65	30.21	28.18
	LibriSpeech	50.13	48.75	44.70	42.91	39.55	35.15	33.17	32.15	29.58	27.51

Table 3.2: Best WER results for each model when pre-training on WSJ and Librispeech and testing on UASpeech.

respectively in WSJ and Librispeech when the training data amount is increased by 10 percent from 30% to 40%. Almost the same amount of improvement is seen in other models. The most interesting result obtained is when the percentage of the data is increased from 90% to 100%, where the rate of improvement is 3.88%5 and 6.26% on WSJ and Librispeech, respectively. The obtained results by Transformer in terms of WER is 14.21% in WSJ and 13.73% in Librispeech, while QuartzNet, wav2vec 2.0 and HuBERT achieved 14.38%, 6.21, 6.13% WERs on Librispeech, respectively. These results emphasize the importance of the volume of the training data and the relevance of training and test data.

In the third series of our experiments, we aimed to tackle the issue of domain difference in training and testing data for LREs by introducing a fine-tuning step after pre-training the model with HRE datasets. Here, the trained models are fine-tuned by in-domain LRE data to improve the

		10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
6-layer LSTM	WSJ	45.31	39.07	34.72	29.48	25.97	22.98	21.93	20.28	18.79	18.06
	LibriSpeech	40.67	37.08	30.18	27.32	24.89	22.53	21.23	20.28	18.53	17.61
6-layer ResLSTM	WSJ	45.43	39.19	35.19	29.91	26.31	23.30	22.39	20.53	19.21	18.27
	LibriSpeech	40.97	36.41	30.28	26.61	24.13	21.98	20.78	20.23	18.73	17.51
6-layer ltLSTM	WSJ	45.12	38.89	34.51	29.27	25.62	22.43	21.32	20.02	18.53	17.91
	LibriSpeech	40.17	34.89	30.03	35.97	23.51	21.58	20.33	19.61	17.28	16.47
6-layer cltLSTM	WSJ	44.96	38.71	34.39	29.16	25.48	22.29	21.19	19.94	18.28	17.41
	LibriSpeech	39.96	34.32	29.91	25.83	22.42	20.52	19.71	18.63	16.97	15.61
FNN+2-layer LSTM	WSJ	47.21	41.83	34.28	30.42	27.53	25.19	23.61	21.49	20.32	19.18
	LibriSpeech	45.63	40.68	33.78	29.81	26.41	24.79	22.58	20.49	19.27	18.63
2-layer LSTM+FNN+	WSJ	47.93	42.87	36.21	32.49	29.78	27.52	25.18	22.83	21.59	19.97
2-layer LSTM	LibriSpeech	46.27	41.29	34.79	31.19	26.83	25.27	22.91	20.81	19.57	18.87
2-layer LSTM+FNN+	WSJ	47.39	41.58	34.32	30.27	27.21	24.97	23.17	21.28	20.12	18.93
FNN	LibriSpeech	45.59	40.37	33.45	29.37	26.15	24.45	22.31	20.29	19.13	18.42
2-layer BLSTM	WSJ	46.51	40.32	33.79	29.51	26.89	24.62	23.08	20.95	19.89	18.57
	LibriSpeech	44.83	39.65	33.12	28.36	25.62	23.71	21.78	19.83	18.86	18.21
1-D CNN+2-layer BLSTM	WSJ	46.39	40.12	32.87	28.32	25.69	24.17	22.77	20.51	19.32	18.05
	LibriSpeech	44.65	39.58	32.92	28.17	25.43	23.58	21.48	19.49	18.73	17.98
1-D CNN+2-layer LSTM	WSJ	45.83	39.98	32.48	27.89	25.21	23.72	22.13	20.18	18.91	17.65
	LibriSpeech	44.53	39.47	32.71	28.07	25.28	23.36	21.31	19.27	18.61	17.83
1-D CNN+2-layer GRU	WSJ	46.23	40.18	32.69	28.13	25.62	23.92	22.56	20.39	19.11	17.92
	LibriSpeech	44.68	39.55	32.78	28.18	25.48	23.68	21.53	19.39	18.87	18.08
QuartzNet	WSJ	43.79	38.51	32.13	28.04	24.93	21.53	18.39	17.71	16.39	15.19
	LibriSpeech	41.38	35.21	30.82	26.62	23.95	21.83	19.61	17.38	15.51	14.38
Transformer	WSJ	42.15	37.62	31.35	27.17	23.69	21.18	18.72	16.53	15.28	14.21
	LibriSpeech	39.17	34.72	29.48	25.75	22.49	20.23	17.83	16.19	14.69	13.73
wav2vec 2.0	WSJ	29.81	26.65	22.93	19.23	17.65	14.29	12.55	10.48	8.93	7.78
	LibriSpeech	27.39	23.38	21.58	18.78	14.28	13.92	11.39	9.31	8.75	6.21
HuBERT	WSJ	29.75	25.31	22.73	19.10	17.21	13.91	12.15	10.13	8.70	7.53
	LibriSpeech	27.19	23.18	21.27	18.35	16.13	13.51	11.12	9.08	8.49	6.13

Table 3.3: Best WER results for each model when trained and tested on WSJ and Librispeech datasets.

performance of the ASR task. Therefore, we pre-trained the different models on Librispeech and WSJ datasets and then fine-tuned the models using LRE data (I-CUBE or UASpeech) to explore these effects. Table 3.4 presents the results obtained from the pre-training of the different models on WSJ and Librispeech and fine-tuning on I-CUBE. The 6-layer model outperforms all other numbers of layers for LSTM, ResLSTM, ltLSTM and cltLSTM. By applying the fine-tuning over LSTM, the WER achieved by the model improved by 0.13% on WSJ, using 10% of data. In 6-layer LSTM, increasing the pre-training data from 40% to 50% enhanced the WER to 9.06% but after fine-tuning WER improved to 9.11% on WSJ data. The 6-layer ResLSTM on Librispeech improved WER by 0.26% while enhancing the WER by 0.19% on Librispeech. The ResLSTM achieved 3.91% WER improvement on Librispeech after fine-tuning with I-CUBE data. The 2-layer ltLSTM got 1.29% improvement after receiving 10% of Librispeech

dataset, while 6-layer ltLSTM model achieved near one percent improvement on WSJ. Furthermore, 8-layer cltLSTM can improve WER by 3.81% on Librispeech dataset.

All 2-layer LSTM and FNN configurations after fine-tuning got better WERs in different percentages of the train data on both datasets. After fine-tuning with I-CUBE data, LSTM after 2 layers of the FNN outperform the other similar structures. The 2-layer BLSTM model improved WER by 1.72% with Librispeech, which outperforms all the combination of the LSTMs and FNNs. By fine-tuning the models, which are a combination of the 1-D CNN with 2-layer BLSTM, LSTM, and GRU, all them got better WER compared with just pre-training. Transformer achieved a WER of 22.01% and 20.87% on WSJ and Librispeech datasets, respectively, which got 1.25% and 1.88% improvement after fine-tuning. In this scenario, Quartzet obtained 22.78% and 22.08% WERs on WSJ and Librispeech while wav2vec 2.0 achieved 20.21% on Librispeech. HuBERT model obtained 20.03% WER on Librispeech dataset which outperforms all models.

Similar trends can be seen when the models are fine-tuned with UASpeech data, as shown in Table 3.5.

3.4 Discussion

This Thesis is set out with the aim of assessing the importance of data size for pre-training and fine-tuning an ASR model in low-resource environment. The large volume of data in pre-training and fine-tuning phases are the most important parameters for the low-resource ASR. By comparing the obtained results from Table 3.3, we can conclude that increasing the amount of the related training data has a strong relationship with the

		10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
6-layer LSTM	WSJ	45.73	43.17	40.31	36.21	32.55	31.41	28.95	27.74	25.44	24.17
	LibriSpeech	45.16	43.31	41.12	37.54	34.12	32.68	30.55	27.37	25.19	23.94
6-layer ResLSTM	WSJ	45.83	43.75	40.81	37.21	33.42	32.49	29.66	27.81	26.18	24.11
	LibriSpeech	45.59	43.54	41.26	37.48	33.75	31.98	30.05	26.97	24.79	23.98
6-layer ltLSTM	WSJ	45.76	43.46	39.34	35.83	32.27	30.38	28.83	27.03	24.93	23.87
	LibriSpeech	45.31	43.46	40.82	37.18	33.51	31.73	28.62	26.29	24.51	23.63
6-layer cltLSTM	WSJ	45.62	43.11	39.15	35.58	31.97	30.08	28.44	26.59	24.58	23.67
	LibriSpeech	45.18	43.21	39.11	36.62	33.23	31.54	30.29	27.89	25.94	25.83
FNN+2-layer LSTM	WSJ	51.29	49.62	45.97	43.08	41.47	39.58	36.88	35.69	34.57	33.84
	LibriSpeech	50.31	49.03	45.38	42.53	39.94	38.65	35.28	33.18	31.51	29.25
2-layer LSTM+FNN+	WSJ	51.61	50.28	47.53	44.42	41.73	40.08	38.49	37.48	35.71	33.91
2-layer LSTM	LibriSpeech	50.67	50.14	46.64	43.53	41.64	39.91	38.04	35.71	32.97	30.27
2-layer LSTM+FNN+	WSJ	51.46	50.11	47.18	44.11	41.17	39.73	37.51	35.34	34.42	33.74
FNN	LibriSpeech	50.39	49.78	46.79	43.27	41.12	39.61	37.12	34.26	32.48	28.89
2-layer BLSTM	WSJ	44.71	42.17	40.21	36.82	33.69	31.01	29.78	28.15	26.65	25.87
	LibriSpeech	44.51	41.87	40.04	37.07	34.75	33.67	30.23	28.91	27.87	26.71
1-D CNN+2-layer BLSTM	WSJ	44.87	42.48	40.53	37.49	34.17	31.46	30.23	28.83	26.98	26.08
	LibriSpeech	45.06	42.32	40.41	37.47	35.29	34.14	30.73	29.74	28.57	27.51
1-D CNN+2-layer LSTM	WSJ	47.31	45.55	43.05	41.08	38.51	35.47	33.52	31.81	29.57	27.13
	LibriSpeech	48.11	46.15	43.59	41.46	39.19	36.14	34.73	33.32	30.07	28.61
1-D CNN+2-layer GRU	WSJ	47.62	46.06	44.24	41.78	39.48	36.62	33.78	31.19	29.12	26.98
	LibriSpeech	48.75	47.17	45.22	42.31	39.73	36.19	33.38	31.68	29.49	28.52
QuartzNet	WSJ	44.06	42.01	38.79	35.30	31.91	30.04	28.03	25.91	23.81	22.78
	LibriSpeech	43.93	41.39	38.25	35.69	31.64	29.86	27.91	25.57	23.47	22.08
Transformer	WSJ	43.69	41.15	38.17	34.77	31.44	29.81	27.57	25.38	23.21	22.01
	LibriSpeech	42.11	40.63	37.78	35.15	31.38	29.53	27.21	25.03	22.91	20.87
wav2vec 2.0	WSJ	38.03	36.71	33.98	31.03	29.02	27.21	25.68	24.71	22.50	21.48
	LibriSpeech	36.71	34.99	33.71	30.19	28.42	26.03	23.97	23.12	21.07	20.21
HuBERT	WSJ	38.30	36.75	33.92	30.89	28.89	26.91	25.30	24.49	22.42	21.32
	LibriSpeech	36.81	34.91	33.39	30.01	28.05	25.78	23.88	22.91	21.17	20.03

Table 3.4: Best WER results for each model when pre-training on WSJ and Librispeech, and Fine-Tuning and test on I-CUBE.

performance of the ASR system. As training and test data are from the same domain, in each step of increasing the percentage of the training data, the WER of the system increased. Therefore, these significant differences in the improvement rate indicate a relatively good correlation between the WER and data from the related domain for training the ASR system.

However, by examining the results in Tables 3.1 and 3.2, which were obtained by testing different models on I-CUBE and UASpeech, only a slight improvement was observed in each step by increasing the percentage of the training data for all models. These results provide further support for the claim that data from an irrelevant domain is unable to improve significantly the performance of the ASR systems. Accordingly, a significant increase in the amount of training data from another domain can not significantly improve the ASR system performance.

		10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
6-layer LSTM	WSJ	65.39	59.70	54.21	49.83	47.21	44.30	40.93	39.11	37.01	36.15
	LibriSpeech	63.21	58.50	51.97	49.21	47.15	43.35	39.61	37.62	36.11	35.73
6-layer ResLSTM	WSJ	63.97	59.73	56.21	52.13	47.93	43.17	42.61	40.07	38.23	37.97
	LibriSpeech	63.38	59.17	55.08	51.13	48.33	42.27	41.65	38.92	38.65	37.01
6-layer ltLSTM	WSJ	64.60	58.87	53.98	50.01	46.87	43.61	40.68	38.57	36.29	35.91
	LibriSpeech	63.06	58.81	52.57	49.10	46.61	42.91	38.58	36.98	35.88	35.01
6-layer cltLSTM	WSJ	63.98	59.17	53.93	49.81	46.93	43.01	40.63	38.39	35.97	35.61
	LibriSpeech	63.51	58.11	52.21	48.59	46.11	42.03	37.93	36.57	35.69	34.83
FNN+2-layer LSTM	WSJ	69.28	64.87	62.17	58.41	53.91	49.89	49.23	46.98	45.09	42.04
	LibriSpeech	68.02	64.07	61.48	58.03	53.62	49.68	48.69	46.83	45.17	41.31
2-layer LSTM+FNN+ 2-layer LSTM	WSJ	69.05	65.07	62.51	58.32	54.12	50.28	49.33	47.21	45.19	42.33
	LibriSpeech	68.28	64.31	61.48	58.08	53.83	50.03	48.33	47.05	45.31	41.92
2-layer LSTM+FNN+ FNN	WSJ	68.87	64.52	61.97	57.83	53.62	49.08	48.61	46.62	45.02	41.55
	LibriSpeech	68.33	63.28	59.93	56.91	53.01	48.52	47.21	65.93	44.62	41.18
2-layer BLSTM	WSJ	65.71	59.38	54.37	50.11	47.21	43.79	40.98	38.85	36.57	36.09
	LibriSpeech	64.03	59.20	52.88	49.27	46.82	43.42	38.33	37.49	36.15	35.17
1-D CNN+2-layer BLSTM	WSJ	65.53	59.31	54.57	50.51	47.83	43.98	41.17	39.04	36.83	36.21
	LibriSpeech	63.75	58.62	53.11	49.47	47.45	44.08	39.05	37.79	36.17	35.78
1-D CNN+2-layer LSTM	WSJ	66.43	61.65	54.93	50.83	48.52	44.51	41.89	41.21	37.31	36.67
	LibriSpeech	64.57	59.36	53.71	50.51	47.31	46.83	40.17	38.65	37.28	36.23
1-D CNN+2-layer GRU	WSJ	66.28	61.57	55.03	52.46	48.69	46.62	42.05	41.51	37.73	37.51
	LibriSpeech	64.31	59.68	54.11	50.83	47.79	45.49	40.63	39.13	36.61	36.41
QuartzNet	WSJ	59.81	55.29	50.17	46.93	44.53	39.97	37.02	34.92	33.61	32.51
	LibriSpeech	58.62	54.07	49.51	46.63	42.78	38.88	36.02	34.57	31.89	31.63
Transformer	WSJ	58.07	54.08	48.97	46.61	42.31	39.51	36.11	33.78	32.51	30.97
	LibriSpeech	57.31	52.67	48.28	44.83	40.93	36.64	33.83	33.41	30.51	30.13
wav2vec 2.0	WSJ	51.63	47.13	45.17	43.48	40.65	37.49	35.20	33.58	30.15	28.03
	LibriSpeech	50.11	48.51	44.67	42.71	39.45	36.21	34.45	32.60	29.51	27.39
HuBERT	WSJ	51.61	47.03	45.09	43.39	40.51	36.39	33.28	31.39	30.12	28.05
	LibriSpeech	49.98	48.61	44.58	42.69	39.28	35.01	32.98	32.51	29.31	27.32

Table 3.5: Best WER results for each model when pre-training on WSJ and Librispeech, and Fine-Tuning and Testing on UASpeech

Increasing the amount of pre-train data improves the performance of the stack LSTM models in different number of layers. By increasing the training data on WSJ from 10% to 100%, the performance of the 2-layer LSTM model has improved by 41.68%, while this increase for the Librispeech dataset is 38.92%. The 4, 6, and 8-layer LSTM models have 43.03%, 46.97%, and 39.21% WER improvements on the WSJ dataset, respectively. This is the same model with the same number of layers on the Librispeech dataset, which has improved performance by 40.47%, 46.56% and 35.87%, respectively. When increasing the amount of data from 10% to 100%, we observed a considerable accuracy improvement, while increasing the number of model layers has led to WERs degradation. The ResLSTM, ltLSTM, and cltLSTM models have consistent improvements by increasing the amount of data to 100% in both the WSJ and Librispeech datasets. The cltLSTM got better WERs on both WSJ and Librispeech and this is

because of the future context frames, which provide more valuable information for this model. The stack LSTM layer got better results in terms of WER by receiving more amount of data and in each percentage. Thus, stacked LSTM structures are powerful when we access a sufficient amount of relevant data to train the ASR system.

All stack LSTM layer models got better WERs by applying fine-tuning on low-resource data. The results make it clear that domain-related data does play a strong role in training a model for a particular ASR task. The 6-layer ResLSTM got 4.05% improvement when fine-tuned with I-CUBE on WSJ in terms of WER which is the greatest among the other models, while ltLSTM and cltLSTM is close to one percent enhancement. The same number of layers for cltLSTM got a better WER improvement when fine-tuned with I-CUBE on Librispeech, which is 1.33%. These results make it clear that there was a significant positive correlation between the amount of domain-related data and the performance of the low-resource ASR system. Another important finding was that pre-training on a high resource data and then fine-tuning on a relevant domain data is the best structure to deal with low-resource environments.

The results of this study indicate a positive correlation between the size of the data and model structure. FNN-LSTM is a model in which FNN helps the model detect the factors of variation through inputs; therefore, LSTM can learn temporal correlations [95]. The most obvious finding from the results is that, such functionality is possible in a high resource data environment, and this topology does not enhance WER of the ASR in the low-resource environment. Similar architectures (combination of LSTMs and FNNs) have had the same performance in such environments. With successive increases in the intensity of the pre-training data in these architectures, the WER improved. Following the addition of domain-related

data, a significant increase in the WER was recorded.

The Transformer model achieved 40.01% and 49.58% WER improvements over WSJ and Librispeech, respectively. In each step, increasing the amount of data improved the WER to show that neural networks are extremely data hungry. After fine-tuning the Transformer, WER raised by 1.25% and 1.88% for WSJ and Librispeech, respectively. The most striking result from the fine-tuning results is that pre-training followed by fine-tuning on the domain-specific data can develop ASR results on the specific domain.

In this Chapter, we explored a detailed examination of various neural network architectures, analyzed their performance and adaptability in the context of low-resource environments. As we move forward to Chapter 4, we will introduce ScoutWav, a novel approach that merges context-based word boundary with the advanced self-supervised learning mechanism of wav2vec 2.0 to enhance wav2vec 2.0 for low-resource tasks. Our methodology involves an initial phase of pre-training on high-resource datasets to obtain a rich representations which are used for two-step of fine-tuning phase within low-resource scenarios. This fine-tuning process, focused on leveraging contextually informed word boundaries, aims to improve the model's ability to accurately capture and interpret spoken language.

Chapter 4

Two-Step Fine-Tuning on Self-Supervised Learning

Recent ASR advances have achieved remarkable results, but challenges persist in low-resource environments where training data is limited or unrepresentative. In this chapter, we present ScoutWav, a novel low-resource ASR model that integrates context-based word boundaries with self-supervised learning (wav2vec 2.0). Our approach involves pre-training on high-resource datasets to derive context-based word boundaries. These boundaries are then used to fine-tune a pre-trained and iteratively refined wav2vec 2.0 model for the downstream LRE task. To optimize wav2vec 2.0 for the LRE, we employ Canonical Correlation Analysis (CCA) to dynamically identify the layers requiring the next step of the fine-tuning. This targeted refinement enables wav2vec 2.0 to learn more descriptive LRE-specific representations. Finally, the representations learned through this two-step fine-tuning process are applied to downstream LRE tasks. Experiments on I-CUBE and UASpeech datasets demonstrate that ScoutWav, leveraging target domain word boundaries and automatic layer analysis, achieves up

to a 12% WER reduction in LRE settings.

4.1 Introduction

Recent advancements in the end-to-end (E2E) ASR systems have demonstrated significant improvements. These systems require large amounts of labeled speech data to achieve high performance, a requirement that may not be feasible across all applications [31]. A low-resource environment (LRE) refers to scenarios where training data and associated labels are scarce and challenging to obtain. Examples of LREs include newly emerging languages (e.g., Kyrgyz) [221] or specific speaker groups with diverse accents [31]. Given the impressive accuracy achieved by E2E ASR models trained on abundant labeled data, there arises a compelling need to leverage unlabeled data during the development of ASR models for LREs.

Recent advancements in self-supervised learning (SSL) have demonstrated its potential to extract meaningful representations from unlabeled data, leading to improved performance in ASR across both low and high resource settings [222, 223]. The core principle of SSL involves leveraging large volumes of unlabeled data to learn generalizable representations, which are subsequently fine-tuned for specific downstream tasks using smaller amounts of labeled data [222]. Recently, wav2vec 2.0 [46] has emerged as a powerful layer-based SSL model built upon the Transformer architecture [47]. While SSL models show promise in achieving high-quality representations for ASR tasks and improved performance through fine-tuning on in-domain data, the current paradigm often relies on a single fine-tuning step, potentially limiting the model's ability to fully adapt to the specific demands of the downstream task. This limitation contributes to a remaining performance gap, necessitating further exploration of more effective fine-tuning approaches.

Recently, Wang et al. [224] have introduced a new low-latency E2E model,

called the scout network (SN), which showed state-of-the-art results in HRE ASR systems. Their model is based on the prioritizing word-specific contextual information for output token prediction. The model employs two distinct components: the SN to detect the word boundary word boundaries and a separate recognition network (RN) that leverages context from preceding frames for sub-word detection. While this method performs well for HRE, the lack of global context information within the SN architecture could potentially limit its performance. We propose an enhanced approach that builds upon the strengths of the SN while addressing its potential shortcomings. By integrating global context information alongside word-specific boundaries, we aim to improve by integrating global context information alongside word-specific boundaries, we aim to improve the performance of the model.

In this chapter, we demonstrate the use of out-of-domain large-scale corpora to boost the performance of low-resource (LR) ASR tasks. To address the training data bottleneck, our model, ScoutWav integrates an SSL model with context-based word boundaries to obtain a high-performance ASR model for LREs. ScoutWav incorporates an enhanced Scout Network (SN) equipped with a context vector embedding mechanism. This mechanism captures both local acoustic features and global context attributes, leading to the generation of high-quality word boundary data for a two-stage fine-tuning process. Firstly, we pre-train a wav2vec 2.0 model on a high-resource (HR) dataset. This pre-trained model is subsequently fine-tuned on the LR data to adapt it to the target domain. Since different layers in Transformer architectures can capture different linguistic information [222], we employ a wav2vec 2.0 layer analysis. This analysis identifies layers poorly capturing acoustic-linguistic features. To enhance these layers, we implement a second fine-tuning step utilizing context-based word

boundary data, effectively embedding global context into ScoutWav.

4.2 Proposed Approach

ScoutWav is an end-to-end ASR model which integrates context-based word boundary with a layer analysis module to efficiently adapt a wav2vec 2.0 pre-trained model to a target downstream ASR task in a low-resource environment. The overall ScoutWav training procedure is shown in Figure 4.1. Obtaining context-based representations is the main aim of the ScoutWav approach to increase the performance of the high-resource ASR model in low-resource environments. Context-based representations enhance the robustness of ScoutWav by providing a deeper understanding of semantic and syntactic language structures, leading to more accurate speech interpretation and transcription. The proposed model consists of two modules: a) building context-based word boundaries and b) layer analysis-based fine-tuning. In the first module, we pre-train an SN on high-resource data and then fine-tune the model with low-resource (LR) data to achieve context-based word boundaries for the target task. In the second module, we pre-train wav2vec 2.0 with the high-resource (HR) dataset and fine-tune with the LR dataset to adapt the model for the target LR task. After fine-tuning wav2vec 2.0, we apply a layer analysis to detect the poor layers. These poor layers are then improved by a second stage of fine-tuning using the context-based word boundary data to enhance and adapt those layers to the low-resource target ASR task.

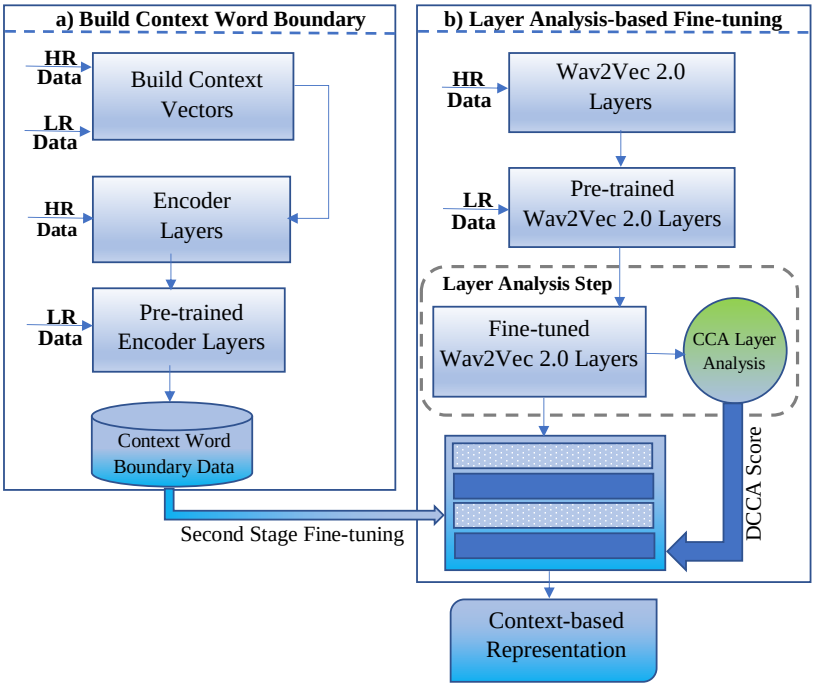


Figure 4.1: ScoutWav structure and training procedure

4.2.1 Build Context-Based Word Boundary

In this section, we summarize how a high-resource ASR model can be adapted for an LR task by capturing the most valuable global and local contextual information. The most valuable contextual information for preparing the annotated output text can be obtained from the speech segment that is related to the target word [224]. Therefore, a look-ahead-based SN model is used to detect the word boundary in the speech segment to identify where a word starts and ends. The SN is focused on identifying boundaries corresponding to the specific word being generated. Therefore, this is more targeted than general embeddings that provide broader linguistic relationships. Look-ahead approaches in data processing, particularly within speech and audio analysis, offer significant advantages over chunk-based methods. By incorporating immediate future information, they enable seamless real-time processing, enhanced contextual awareness, and more accurate predictions or recognitions. This dynamic integration

of past and imminent data improves the handling of overlapping in speech input and provides better adaptability to task target task requirements. SN consists of CNN layers for pre-processing of the input sequence, followed by N_s self-attention layers. Then, a combination of the linear layer and a sigmoid layer is used to detect the probability of the boundary p_i. The output of the current frame depends on the previous one. To train the model, the following cross-entropy loss is minimized to optimize the model for the word-boundary structure:

$$Loss = \sum b_i log(p_i) = \sum b_i log(Sigmoid(W h_i^s)) \quad ,$$

where $b_i \in 0, 1$, h_i, and W are the ground truth of the word boundary, the output of the hidden sequence, and the trainable matrices, respectively.

The SN is a neural component that learns to detect the most important contextual information. This adaptation is different from the static nature of most embeddings. In LREs, the context information at each boundary should be adapted to the LR task to have reasonable performance in the target environment. An SN does not capture global contextual information when detecting the word boundary, reducing the overall performance of the ASR model in both high- and low-resource settings. In contrast, ScoutWav utilizes two sets of context vectors in each self-attention layer, that are calculated through all previous frames to capture not only local acoustic information, but also global context features. This allows ScoutWav to adapt reasonably well to the LR downstream task. The first set of the context vectors are calculated in each layer of each block and fed into the upper layer of the current layers. The second vector is obtained by concatenating all vectors in the current layer to share the global characteristics, speaker, and linguistic between the layers to enhance the adaptation proce-

dure of the model into the LRE. We calculate the multihead self-attention as follows:

$$MHD(Q^n, K^n, V^n) = Concat(head_1, \cdots, head_m)W_O^n$$

$$head_i = Attention(Q^n W_{Q,i}^n, K^n W_{K,i}^n, V^n W_{V,i}^n) \quad ,$$

where W represents trainable matrices. In the first layer, Q^1, K^1, and V^1 are represented as a feature matrix which include block input and context vector. This context vector is initialized using rearranged positional encoding for each layer, utilizing only the output of each encoder layer. In subsequent layers, we enhance Q, K, and V with two additional context vectors: one from each encoder of the previous layer, and a summarized context vector from all encoders in the current layer. Integrating this contextual information into our Scout Network (SN) yields an improved model capable of more accurate word boundary detection. To adapt the improved SN to low-resource tasks, we first pre-train it on high-resource data and then fine-tune it with the LR dataset.

4.2.2 Layer Analysis-Based Fine-tuning

In this section, we summarise how we adapt the wav2vec 2.0 approach by integrating layer analysis of the model and two-step fine-tuning mechanism to achieve a higher performance for LREs. The wav2vec 2.0 framework maps the raw audio sequence into a high-level contextual representation through a set of convolutional layers followed by self-attention layers, which are trained with a contrastive objective. Investigating the Transformer layers of the BERT model in natural language processing indicated that different blocks behave differently and capture different levels of information;

the earlier blocks represent syntactic information, while the high-level ones present high-level semantic information [221]. Therefore, such a layer analysis over wav2vec 2.0 helps to have a better insight of layers behavior to enhance and fit the model for the low-resource ASR setting. To get a better understanding of layer behavior, we use Canonical Correlation Analysis (CCA) [225] inspired by [226] over different layers of wav2vec 2.0 and detect poor layers, which may not be well suited for the LR target ASR task. Then the context-based word boundaries obtained from the previous section are used for the second stage of the model fine-tuning to improve the performance of the poor layers.

We use Canonical Correlation Analysis (CCA) [225] as a measure to detect which layer of the wav2vec 2.0 model may not be well suited for the target low-resource ASR task. CCA is a statistical approach for finding maximum correlations between linear combinations of two continuous-value vectors. It can be used to calculate the similarity between layer representations and the acoustic feature vector, evaluating how well different model layers adapt to the downstream task. CCA takes n pairs of vectors $(x_1, y_1), \cdots, (x_n, y_n)$ as input and return a correlation score as a similarity measure between two vectors. In ScoutWav, we use Deep CCA (DCCA) [227] to explore the complex relationship between data by passing it through a deep network. The output of the network is then fed into CCA to measure the similarity. The DCCA solution can be defined as follows:

$$\begin{aligned} &\arg\max_{W_1,W_2} \rho = tr(W_1' f_1(X^1) f_2(X^2)' W_2) \\ s.t. &\begin{cases} W_1'(f_1(X^1)f_1(X^1)' + r_1 I)W_1 = I \\ W_2'(f_2(X^2)f_2(X^2)' + r_2 I)W_2 = I \end{cases} \end{aligned} \tag{4.1}$$

where f_1 and f_2 are two DNN networks, $f_1(X^1)$ and $f_2(X^2)$ are DNN outputs which are interpreted by CCA to calculate the similarity score. The tr calculates the total correlation; W_1 and W_2 are corresponding weight matrix embedded; r_1 and r_2 are regularization constants. The similarity score is between 0 and 1, where 1 is the maximum similarity.

This stage involves three key steps. First, we pre-train the wav2vec 2.0 model on high-resource data. Then, we fine-tune the model specifically on the low-resource target data. Our layer analysis procedure then systematically examines each layer using a word embedding vector to detect the poor layers. Finally, we employ context-based word boundaries for a second stage of fine-tuning. This targeted refinement focuses on the poor layers, adapting them for the target task and ultimately enhancing the model's overall performance in the low-resource setting.

4.3 Experiments

4.3.1 Datasets

We examine the performance of ScoutWav on two low-resource datasets, I-CUBE and UASpeech, to demonstrate its effectiveness in low-resource environments.

4.3.2 Experiment Setup

For the WSJ, the models were trained on the SI-284 set and evaluated on the eval92 set. We trained the models with LibriSpeech, by using 960 hours of training data, and evaluated with data from both clean and contaminated

testsets. Finally, for TED and CV datasets, we used 10-fold cross-validation and reported average and standard deviation WER across all folds. The input acoustic features were extracted by employing 80-dim log Mel-filter bank features with 3-dim pitch features and with a hop size of 10 ms and a window size of 25 ms, which were normalized with the mean and variance. For the WSJ setup, the number of output classes was 52, including the 26 letters of the alphabet, space, noise, symbols such as period, an unknown marker. To predict the probability distribution of all characters in the alphabet, we use the CTC loss function and use AdamW optimizer [228] as a hyperparameter setting with an initial learning rate of 0.001. The text is tokenized using SentencePiece [229] and we set the vocabulary size to 5000. We run the second-stage fine-tuning stage for 20 epochs. We also use beam width $K = 10$, boundary decision threshold $\sigma = 0.0005$, language model weight $\alpha = 0.5$ and length penalty $\beta = 2.0$. We use Montreal forced aligner [230] to define phone and word segment. Finally, we pre-train and fine-tune wav2vec 2.0 in two different settings; Base setting and Large setting. For the Base setting, we replicated the architecture from [231] with the following parameters: $d_{model} = 512$, $d_{ff} = 2048$, $d_h = 4$, $N_e = 12$ and $N_d = 6$ for a fair comparison to previous works. For the Large setting, we used the architecture from [232] with $N_e = 24$ and $N_d = 12$ and for both settings, the down-sampling rate r is 4.

4.3.3 Results

We carried out a WER comparison on different datasets to evaluate our proposed context-based word boundary detection model in ScoutWav with SN and a chunk-based model. In ScoutWav, we pre-train each model with HR data and then fine-tune with the target LR in-domain data. The

results are summarized in Table 4.1, which shows that ScoutWav outperforms other models for both the I-CUBE and UASpeech datasets. The best performance is achieved after pre-training with LibriSpeech. This indicates a correlation between the model performance and the amount of pre-training data. As the amount of pre-training data increases, model performance tends to improve due to a more comprehensive representation of linguistic and acoustic variability, enabling the model to learn more robust and generalizable features. In summary, ScoutWav's integration of contextual information to extract local and global features significantly enhances its word boundary detection capabilities compared to Scout Network and chunk-based methods. This contextual awareness allows ScoutWav to discern subtle linguistic and acoustic cues that simpler models overlook. The result is more precise identification of word boundaries within continuous speech, minimizing transcription errors and demonstrating the value of comprehensive contextual analysis in word boundary detection for speech recognition systems.

		High-Resource Data			
Model	**LR Data**	**Libri**	**WSJ**	**TED**	**CV**
SN	**I-CUBE**	16.41	18.83	17.39	20.17
	UASpeech	28.87	30.12	29.73	33.48
Chunk-Based	**I-CUBE**	19.81	21.35	20.93	22.87
	UASpeech	31.18	33.98	32.35	34.11
ScoutWav	**I-CUBE**	**14.29**	16.37	17.28	19.87
	UASpeech	**25.93**	28.17	26.53	30.13

Table 4.1: WER for detecting context-based word boundary on different datasets with different models.

In the second stage of our experiments, we investigated how different layers within the pre-trained and fine-tuned wav2vec 2.0 model process various acoustic attributes of the input. Figure 4.2 compares results after pre-training wav2vec 2.0 on four high-resource datasets, followed by fine-tuning with I-CUBE data and a second stage of fine-tuning using context-based

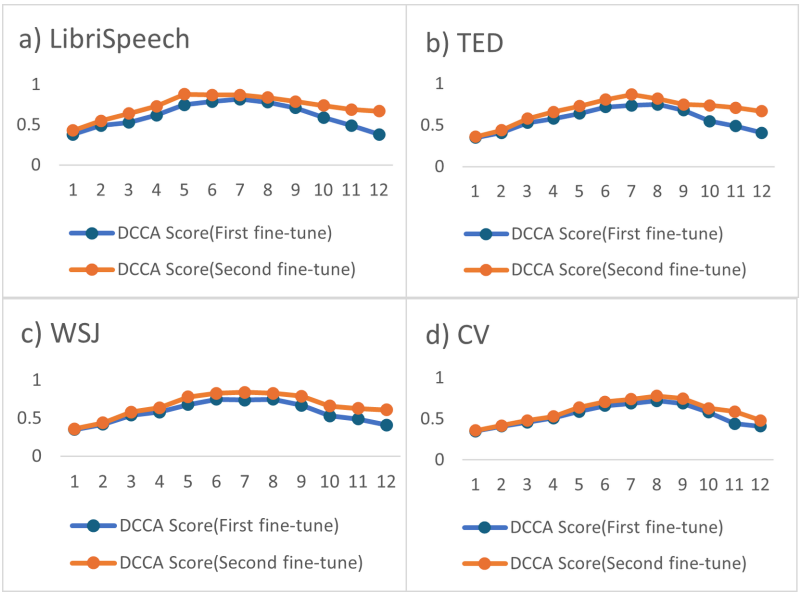


Figure 4.2: Layer analysis of wav2vec 2.0 with different pre-training and fine-tuning with I-CUBE data.

word boundaries. To measure layer-to-input similarity, we used DCCA scores for the Base setting (12 layers). Similar DCCA trends were observed for the Large setting. Our analysis reveals that the first (1-3) and last (9-12) layers diverge from the input, suggesting they learn less directly relevant representations. Conversely, middle layers (5-8) demonstrate greater similarity to the input data, indicating their suitability for the final target task. Importantly, after the second fine-tuning stage focused on poor layers using word boundaries, we observed improvement in the final layers. This highlights the effectiveness of our approach in refining wav2vec 2.0's internal representations. Interestingly, we found a correlation between the accuracy of context-based word boundaries and the degree of layer improvement: the CV dataset, where word boundary accuracy was lowest, showed a correspondingly less significant improvement rate. This finding underscores the importance of accurate word boundary detection for maximizing the benefits of our layer-targeted fine-tuning strategy.

We further analyzed the large setting of the model (24 layers), measuring DCCA scores as presented in Figures 4.3 (LibriSpeech dataset) and 4.4

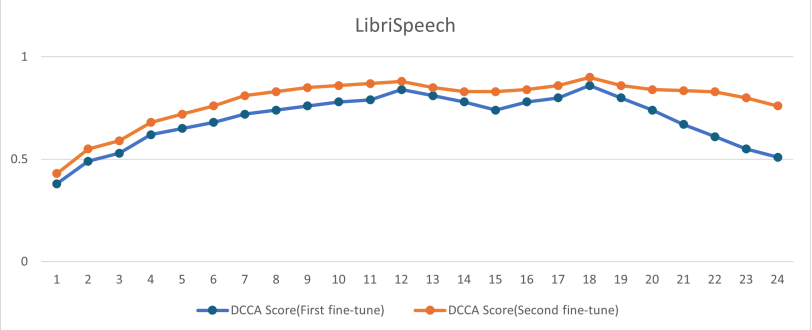


Figure 4.3: Layer analysis of wav2vec 2.0 for large setting after pre-training on LibriSpeech and fine-tuning with I-CUBE data.

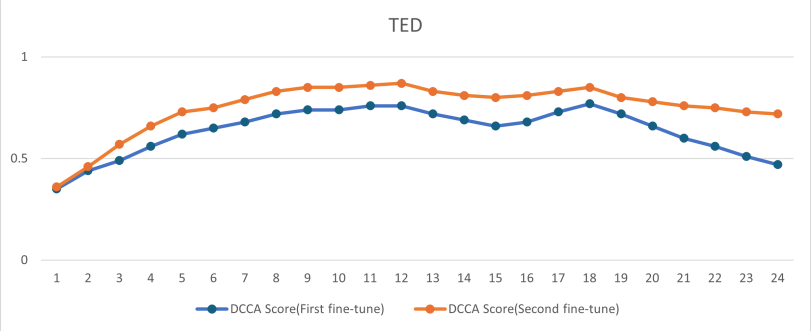


Figure 4.4: Layer analysis of wav2vec 2.0 for large setting after pre-training on TED and fine-tuning with I-CUBE data.

(TED dataset). Consistent with our previous findings, the first, and last layers of the model exhibited divergence from the input speech, indicating less suitable representations. After applying our secondary fine-tuning strategy targeting these poor layers using word boundaries, we observed substantial enhancements, particularly within the final layers. This result shows the effectiveness of our approach in refining wav2vec 2.0 for better alignment with the target task, even within a larger model architecture.

The analysis of the wav2vec 2.0 layers with the UASpeech dataset are shown in Figure 4.5. Similar to the results achieved with I-CUBE, the second step fine-tuning with obtained word boundaries helps the model to extract more contextual information from the first and last layers of the model that lead to improve the performance of the ASR model in the LRE.

Figure 4.2 and Figure 4.5 indicate that the last layers of the model have

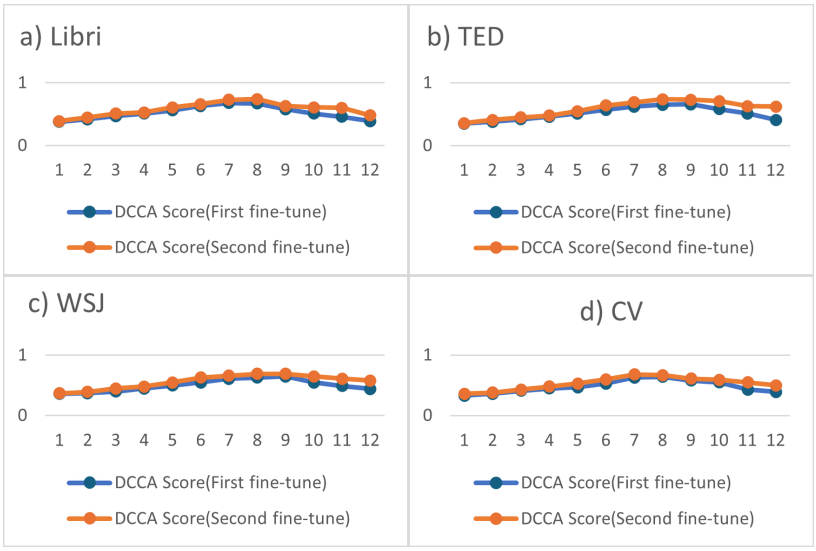


Figure 4.5: Layer analysis of wav2vec 2.0 with different pre-training and fine-tuning with UASpeech data.

the largest improvement after the second fine-tuning step, which indicates that that the pre-trained and fine-tuned model is significantly improved by the context-based word boundary fine-tuning to embed task-specific information.

We further analyzed the representational capabilities of the large model (24 layers) by measuring DCCA scores after pre-training on LibriSpeech and TED datasets (Figures 4.6 and 4.7, respectively. Similar to fine-tuning the model on I-CUBE data, the first, and last Transformer layers exhibited divergence from the input speech, suggesting less suitable representations. However, upon applying our secondary fine-tuning strategy using word boundaries, we observed significant improvement, particularly within the final layers. This finding demonstrates the robustness of our approach: even across diverse pre-training datasets and a larger model architecture, targeted fine-tuning with word boundaries effectively enhances wav2vec 2.0's representations for the target task.

Table 4.2 demonstrates the superiority of ScoutWav in low-resource scenarios. After the second fine-tuning step on the pre-trained and fine-

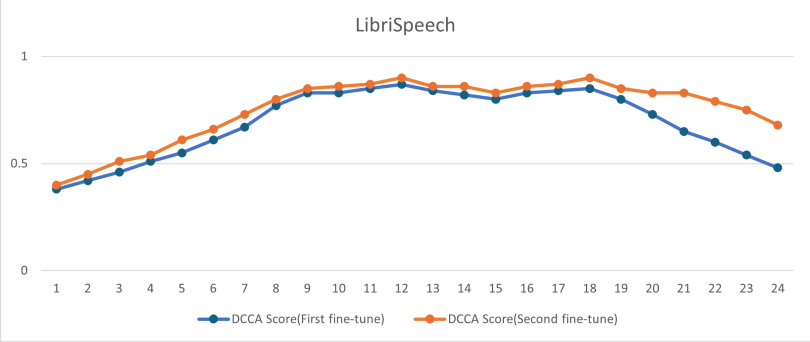


Figure 4.6: Layer analysis of wav2vec 2.0 for large setting after pre-training on LibriSpeech and fine-tuning with UASpeech data.

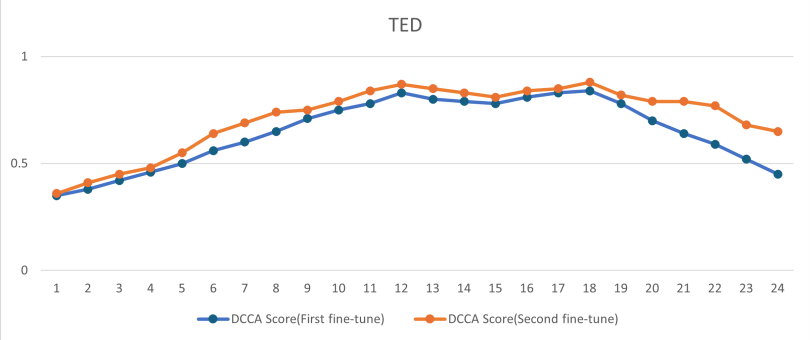


Figure 4.7: Layer analysis of wav2vec 2.0 for large setting after pre-training on TED and fine-tuning with UASpeech data.

tuned model, ScoutWav significantly outperforms both wav2vec 2.0 and QuartzNet [103]. Specifically, the Large ScoutWav model achieves impressive WERs of 10.14% on I-CUBE and 13.32% on UASpeech, representing improvements of 12% and 6.7%, respectively, over the next-best scores from Large wav2vec 2.0. This advantage extends across different datasets and settings: Base ScoutWav outperforms Base wav2vec 2.0 by 0.7% on I-CUBE and a substantial 7.7% on UASpeech. Furthermore, ScoutWav consistently surpasses QuartzNet by a considerable margin. These results highlight the effectiveness of our approach in refining speech representations for low-resource ASR tasks.

LRE	Method	Libri	WSJ	TED	CV
I-CUBE	**ScoutWav Base**	15.32	16.73	15.21	17.89
	ScoutWav Large	**10.14**	**13.98**	**12.57**	17.78
	wav2vec 2.0 Base	17.38	16.61	15.45	18.42
	wav2vec 2.0 Large	11.61	14.73	13.64	**17.22**
	QuartzNet	26.51	29.75	28.39	31.53
UASpeech	**ScoutWav Base**	18.46	22.21	19.38	24.55
	ScoutWav Large	**13.32**	**15.29**	**14.93**	**18.35**
	wav2vec 2.0 Base	19.07	23.94	21.31	25.18
	wav2vec 2.0 Large	14.28	16.23	15.19	18.87
	QuartzNet	29.15	34.93	31.79	36.79

Table 4.2: WER results for different methods in two LREs. Best performing models are highlighted.

4.4 Chapter Summary

In this chapter, we introduce ScoutWav, an end-to-end ASR model specifically designed for low-resource environments. It employs a two-step fine-tuning process to adapt a high-resource ASR model to the LR target domain. Our novel context-based word boundary mechanism, capturing both global and local acoustic patterns, enables ScoutWav to accurately detect word boundaries within LRE data. Additionally, a layer analysis module identifies underperforming model layers. By targeting the second fine-tuning step on these layers using context-based word boundaries, ScoutWav achieves significant performance improvements over established ASR models. In the next chapter, we'll explore how combining data selection criteria with layer analysis can further mitigate the training data bottleneck in LRE ASR.

Chapter 5

Combination of Local Aggregation and Self-Supervised Learning for Detecting Speech Hidden Units

Advances in deep learning have led to impressive results in ASR systems. However, ASR performance in Low-Resource Environments remains a challenge due to the limited training data available for specific domains. We propose that careful data sampling criteria, focused on selecting the most informative speech samples, can be crucial for overcoming this training data bottleneck. Our proposed method, Local Aggregation BERT (LABERT), combines an active learning model with an adapted local aggregation metric for self-supervised speech representation learning. Active learning identifies the most informative speech units, while the aggregation metric en-

courages the model to cluster similar data points in the latent space and separate dissimilar ones. This approach assists in uncovering hidden patterns within LRE tasks. We evaluate LABERT's performance on two LRE datasets, I-CUBE and UASpeech, to explore its effectiveness within LRE ASR problems.

5.1 Introduction

End-to-end (E2E) automatic speech recognition (ASR) systems have made significant progress by the availability of vast amounts of labeled speech data. However, these systems struggle in low-resource environments (LREs), where representative training data is limited – such as for lesser-known languages like Kyrgyz [221] or among speakers with diverse accents [31]. This scarcity of data and labels poses a significant challenge for LREs [31]. One key bottleneck in LREs is selecting the most informative training samples. To address this, **LABERT** combines active learning, which prioritizes informative speech samples, with self-supervised learning. By incorporating the Local Aggregation function, LABERT effectively identifies and groups similar speech patterns within hidden layers, aiming to improve speech recognition performance in these challenging scenarios.

Self-Supervised Learning (SSL) models have emerged as a pivotal approach in deriving data representations from unlabeled samples, which are then fine-tuned on labeled data [46]. wav2vec is an SSL model [191], which employs the Contrastive Predictive Coding (CPC) methodology. This method emphasizes pre-training speech representations by predicting subsequent acoustic frames within a sequence. A noteworthy evolution in this domain is the vq-wav2vec model [195], a synthesis of the foundational wav2vec with the BERT model. This integration is designed to extract BERT-style speech representations through a two-stage training. Building upon the aforementioned architecture, the DiscreteBERT model [233] enhances the capabilities of vq-wav2vec by integrating a pre-trained BERT model and subsequently fine-tuning it for downstream ASR applications. The W2v-BERT [142] advances self-supervised speech pre-training by introducing a hybrid framework that merges the strengths of contrastive learning and

masked language modeling. Inspired by wav2vec 2.0 [46], it utilizes contrastive learning to distinguish true future speech segments from distractors, enhancing the understanding of temporal dependencies. Simultaneously, it employs a BERT-style masked prediction task on quantized representations to foster learning of contextual representations at a fine-grained level. The paper demonstrates that this dual approach of contrastive learning and masked prediction outperforms models that rely on either technique alone, resulting in more robust speech representations that excel in downstream speech recognition tasks. In contrast, the BEST-RQ [149] presents a novel self-supervised learning approach for speech recognition leveraging a random-projection quantizer. The method projects speech input features onto a randomly initialized matrix and uses a randomly-initialized codebook for discretization. This quantizer simplifies computation and preserves the original structure of speech data. The model is trained to predict masked parts of the speech signal, learning from the context of unmasked segments. Experiments on the LibriSpeech dataset demonstrate the approach's effectiveness, achieving reduced word error rates compared to other SSL baselines.

DeLoRes model [234] introduces a novel self-supervised learning framework for audio representation learning, specifically designed for low-resource scenarios. The core idea focused on decorrelating latent spaces to promote the learning of diverse and non-redundant information within audio samples. To achieve this, DeLoRes measures the cross-correlation matrix between outputs of two identical networks fed with distorted versions of the same audio. Inspired by the Barlow Twins objective, it aims to produce embeddings that are invariant to distortions while maximizing informational richness. The authors demonstrate DeLoRes's effectiveness on downstream tasks, showcasing its ability to learn robust representations even with lim-

ited training data and computational resources. TERA [146] introduces a self-supervised speech pre-training approach using Transformer Encoders. Unlike prior approaches that rely on a single auxiliary objective, TERA utilizes alterations along three axes (time, frequency, and magnitude) to train a Transformer Encoder on unlabeled speech data. This allows the model to learn robust representations through reconstructing acoustic frames from their altered versions. TERA can be employed for either speech representation extraction or fine-tuning with downstream models, achieving strong performance on various tasks like phoneme classification, keyword spotting, speaker recognition, and speech recognition. The study also explores the influence of different alteration techniques, the amount of pre-training data, and the type of features used for pre-training.

In the ASR literature, clustering approaches are also employed as a method to obtain pseudo-labels for SSL. Deep Cluster [172] uses k-means algorithm to group similar instances and optimizing an encoder network through a classification loss. Hidden unit BERT (HuBERT) [47] introduces a novel self-supervised framework for learning robust speech representations. The model addresses issues common in prior self-supervised speech representation techniques by using k-means clustering to generate more reliable pseudo-labels. HuBERT masks portions of the input audio and then trains a transformer-based model to predict the masked targets. This masked prediction approach, inspired by BERT's success in NLP, forces the model to extract contextually relevant speech representations. SwAV [41] introduces a novel unsupervised learning technique for training convolutional networks used in computer vision. SwAV addresses shortcomings of conventional contrastive learning methods, which often rely on computationally expensive pairwise feature comparisons. Instead, SwAV compares 'codes' (cluster assignments) generated from multiple augmented views of the same image.

SwAV enforces consistency between these cluster assignments, leading to robust feature representations. The method's efficiency and scalability, along with its performance on downstream tasks, demonstrate its effectiveness in learning meaningful visual representations without the need for explicit labels. However, many clustering algorithms suffer from the seed selection problem, resulting with noisy clustering results, which would negatively affect the learning process in the LRE ASR task. Our approach, Local Aggregation BERT (LABERT), draws inspiration from Local Aggregation (LA) [152], and applies a local non-parametric aggregation in a latent feature space instead of within the global clustering algorithm. LABERT selects more informative speech units and feeds them into the LA function, which enables it to address the noisy and arbitrary clustering process and to model the interrelation similarity more accurately in the latent spaces for the LRE ASR system.

LABERT (**L**ocal **A**ggregation with **BERT**) is a novel self-supervised representation learning model for learning speech representations, especially for low-resource ASR tasks. Drawing inspiration from HuBERT, LABERT uses an offline hidden unit detection module to give noisy labels to a BERT-like pre-training model. Uniquely, LABERT employs non-parametric aggregation in a latent space for visual embedding [152], rather than relying on a global clustering technique to detect hidden units to learn speech representations. To tackle the limited training data issue in LREs, we incorporate a committee-driven active learning approach combined with an LA function to detect more valuable speech samples in the latent space. This works by enhancing the LA's ability to spot close neighbours within the latent space around given speech samples. By continuously improving and fine-tuning the active learning model during training, LABERT effectively identifies speech units with similar characteristics in the latent space

and allowing them to be grouped together into the same clusters. As a result, this procedure allows LABERT to detect an informative and diverse subset of the data to train a model, and obtain more accurate speech units to achieve performance comparable to the full dataset to address the data bottleneck and model a well-suited representation for the downstream LRE ASR task.

5.2 Proposed Approach

LABERT is an end-to-end ASR model which explores how to effectively use speech-only data to improve the performance of the speech recognition system in a low-resource environment. As illustrated in Figure 5.1, LABERT comprises two core components, which consists of: a) hidden unit discovery with local aggregation function and b) masked target unit prediction. To extract meaningful representations from raw audio in the first module, LABERT employs the Local Aggregation function. This function moves similar audio units together in the embedding space, while enabling dissimilar units to separate from each other. To mitigate noisy clustering issues, LABERT incorporates a committee-based active learning approach for selecting more informative initial unit seeds [152]. Crucially, the clustering process leverages an iterative strategy: the first iteration uses MFCCs features, while subsequent iterations utilize carefully selected representations generated by a Canonical Correlation Analysis (CCA) module [225]. In the second module, inspired by the success of BERT [235], LABERT employs a masked language modeling objective to predict hidden units. The model calculates cosine similarity scores between context vectors and every hidden unit embedding from all available hidden units. Finally, cross-entropy loss is used for the prediction process.

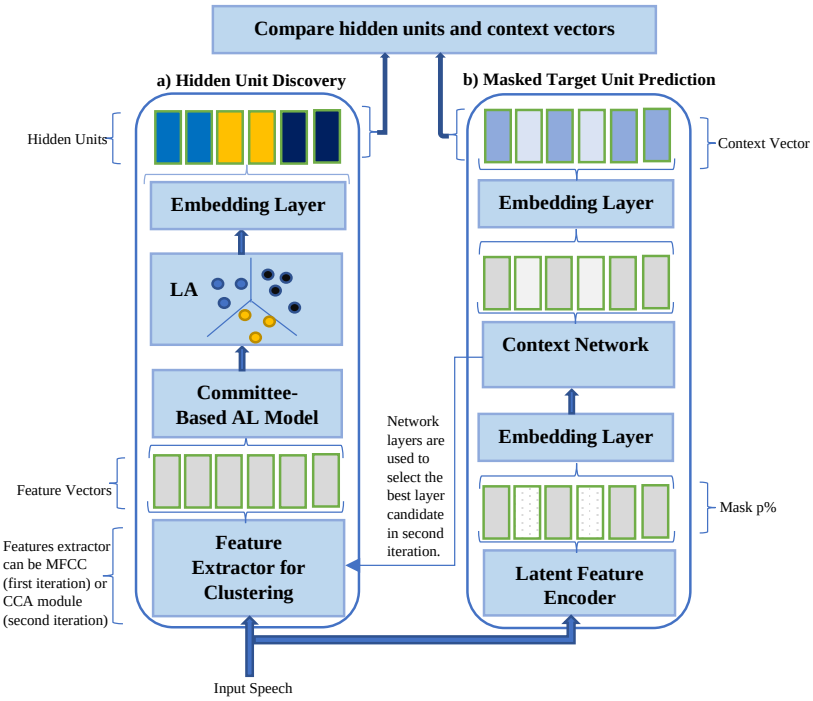


Figure 5.1: The structure of LABERT model.

5.2.1 Hidden Unit Discovery through LA function

In LABERT, LA function is adapted to extract hidden speech units from raw audio data. Our main objective is to train an embedding function $Y = f(X)$, which can effectively map the input speech $X = [X_1, ..., X_T]$ to the corresponding features, $Y = [Y_1, ..., Y_T]$, where similar units are grouped together, and dissimilar ones are separated. To do this, we identify two sets of neighbors, close neighbors (C_i) and background neighbors (B_i), dynamically during the training of the embedding function for X_i and its embedding Y_i [152]. Close neighbors are embeddings that are similar to Y_i, while background neighbors are employed to establish the distance scale with respect to which the assessment of closeness is made. Within the context of LREs, local aggregation metric enables the LABERT model to scale the measurement for the target downstream task to obtain a better performance in such environments. After detecting the close and background neighbors, LABERT forces the current embedding toward close neighbors

but far from background ones.

At each optimization step, the background neighbors for a given embedded point Y_i are defined as the k closest embedded points (denoted as $g_k(Y_i)$) within the embedding space Y. The distance between these neighbors is evaluated by employing the cosine distance. The value of k, representing the number of background neighbors is a hyperparameter for the algorithm. To find the close neighbors, k-means clustering algorithm is applied to cluster all embedding spaces in Y to cluster the representations into P groups $\{1, ..., P\}$. To mitigate the impact of noisy clustering results, LABERT employs a committee-based active learning approach. This technique selects more informative embeddings as initial clustering seeds for the k-means algorithm. By providing higher quality seeds, LABERT achieves more accurate hidden unit classification, which is crucial for downstream speech recognition tasks. The number k of background neighbors and number P of clusters are hyperparameters of the model.

Taking into account the definition of close and background neighbors, an LA level is defined as $L(C_i, B_i|\Theta, X_i)$ for each speech unit X_i. Θ parameters are tuned during the training to maximize the level of local aggregation. In [152], the probability that a feature Y to be considered as the i-th unit is defined as:

$$P(i|Y) = \frac{exp({Y_i}^T Y/\tau)}{\sum_{j=1}^{T} exp({Y_j}^T Y/\tau)} \tag{5.1}$$

where $\tau \in [0, 1]$ is a fixed hyperparameter.

The probability that a feature Y is classified as a unit in a speech frame T is computed as:

$$P(T|Y) = \sum_{i \in T} P(i|Y) \tag{5.2}$$

The level of local aggregation is defined as the negative log-likelihood of Y_i

being a close neighbour (is in C_i), given that Y_i is recognized as a background neighbour (is in B_i):

$$L(C_i, B_i|\Theta, X_i) = -log\frac{P(C_i \bigcap B_i|Y_i)}{P(B_i|Y_i)} \tag{5.3}$$

Finally, the loss to be minimized is:

$$\text{Loss} = L(C_i, B_i|\Theta, X_i) + \lambda \left\|\Theta\right\|_2^2 \tag{5.4}$$

where λ is a regularization hyperparameter.

As explained earlier, LABERT employs a committee-based active learning approach to obtain more informative parts of speech data, considering them as seeds for the local aggregation function. In contrast to traditional committee-based active learning approaches, which relies on multiple structurally distinct models to highlight model differences [236], LABERT leverages a streamlined approach inspired by SMCA [236]. Instead of building variant committee models through dropout, LABERT utilizes the R-Drop model [237]. This approach addresses SMCA's inconsistency between training and inference, which can hinder performance in both high- and low-resource ASR scenarios. In R-Drop model, each speech input X_i passes through the model twice during training, generating two outputs: $\rho_1^\omega(l_i|x_i)$ and $\rho_2^\omega(l_i|x_i)$, where l_i is the transcribed text of x_i. R-Drop then minimizes the bidirectional Kullback-Leibler (KL) divergence between these distributions, promoting regularization:

$$L_{KL}^{i} = \frac{1}{2}\Bigg(D_{KL}\Big(\rho_1^{\omega}(l_i|x_i)||\rho_2^{\omega}(l_i|x_i)\Big) + D_{KL}(\rho_2^{\omega}(l_i|x_i)||\rho_1^{\omega}(l_i|x_i)\Big)\Bigg) \tag{5.5}$$

By applying the R-Drop method during the training step, the dropout hypotheses of the seed model may diverge significantly from the standard hypotheses within the model. Specifically, we focus on the frame-level differences between these hypotheses. This divergence considered as a crucial data selection metric, allowing us to extract more informative speech units from utterances. These selected units then utilize for the clustering process within the local aggregation function. LABERT tackles the challenge of noisy clustering by incorporating two essential criteria: informativeness and diversity. LABERT prioritizes informative speech segments. By identifying and retaining the most relevant units, we enhance the overall quality of the clustering process. This step is critical for accurate downstream tasks. To avoid redundancy and robustness, LABERT dynamically computes diversity during the pre-training phase. We employ the B_i set, which adapts to the specific context, that prevents the local aggregation function from selecting data that is too similar to each other. This diversity-driven approach ensures that LABERT produces more precise clusters of speech units.

5.2.2 Masked Target Unit Prediction

In this section, we summarize the utilization of the BERT model and explore strategies for selecting the high quality representations from the learned layers to enhance the second iteration in the local aggregation

function. BERT, a powerful language model, operates through a masked prediction mechanism on extensive text data. Its pre-trained representations can effectively compensate for the scarcity of text data in low-resource ASR problems. Inspired by approaches like HuBERT and wav2vec 2.0, our LABERT method employs a similar mask generation strategy. However, we selectively mask only a fraction ($p\%$) of the chosen timesteps. This deliberate choice ensures that the model receives real input, addressing any inconsistencies between training and testing phases. Building upon insights from the previous chapter (Chapter 4), we introduce a layer analysis module to identify the most suitable layer within the model for our target low-resource ASR task. We use CCA as our measuring tool. CCA quantifies the maximum correlations between linear combinations of continuous value vectors. By assessing the similarity between layer representations and the acoustic feature vector, we detect how effectively different layers adapt to the downstream LRE task. Our approach compels the LABERT to learn task-specific representations from the downstream ASR context.

5.3 Experiments

5.3.1 Datasets

For unsupervised pre-training, we use the full 960 hours of LibriSpeech (Libri) [27], full 81 hours of WSJ [30], 1k hours of Common Voice (CV) [238] and 450 hours of TED-LIUM 3 (TED3) as our high-resource environment datasets. We examine the performance of LABERT on two low-resource datasets, ICUBE and UASpeech, to demonstrate its effectiveness in low-resource environments.

5.3.2 Experiment Setup and Metrics

Our pre-trained models follow the wav2vec 2.0 architecture [46], comprising a convolutional waveform encoder, a BERT encoder [176], a projection layer, and a code embedding layer. We utilize two LABERT configurations: BASE and LARGE. The first two closely follow the architectures of wav2vec 2.0 BASE and LARGE, respectively. To demonstrate the efficiency of our proposed method in utilizing low-quality cluster assignments, we employ the k-means algorithm [239] for acoustic unit discovery. This algorithm, known for its simplicity, models isotropic Gaussians with equal scalar variances for each acoustic unit. For generating labels to initialize LABERT training on the HRE datasets, we perform k-means clustering with 50 and 100 clusters using 39-dimensional MFCC features. These features consist of 13 base coefficients along with their first- and second-order derivatives. In order to improve the quality of targets for subsequent iterations, we apply k-means clustering to the latent features extracted from the LABERT model. These latent features are obtained from the LABERT model pre-trained in the previous iteration, specifically at an intermediate transformer layer. Clustering is performed using the MiniBatchKMeans algorithm from scikit-learn [240], which iteratively fits mini-batches of samples. For all LABERT configurations, we employ a mask span of $l = 10$. Unless specified otherwise, a random $p = 8\%$ of waveform encoder output frames are selected as mask start points. Optimization is performed using the AdamW optimizer [219] to update the model with an initial learning rate of 0.001. The learning rate undergoes a linear warmup for the initial $p = 8\%$ of training steps, reaching a peak value before decaying linearly to zero. Peak learning rates are set at $5e{-}4$ for BASE and $1.5e{-}3$ for LARGE models. We set $k = 4096$ to compute B_i using the nearest neighbors procedure. In computing C_i, we run the k-means clustering algorithm with

50 and 100 clusters on 39-dimensional MFCC features, to obtain labels for LABERT pre-training over the HRE data sets. We considered 50 and 100 clusters on 100h, 300h and 500h of speech samples from LibriSpeech and fine-tuned the model with I-CUBE for cluster quality analysis. The input acoustic features are 80-dimensional filterbanks, extracted with a hop size of 10 ms and a window size of 25 ms, which are normalized with the mean and variance. For the WSJ setup, the number of output classes is 52, including the 26 letters of the alphabet, space, noise, symbols such as period and an unknown marker. The text is tokenized using SentencePiece [241] and we set the vocabulary size to 5000.

Benchmarking results are presented for the pre-trained and fine-tuned wav2vec 2.0 and HuBERT models in Base and Large settings, as well as for QuartzNet and DiscreteBert. The primary evaluation metric we used is the WER. We also compute the Phone Purity and Phone-Normalized Mutual Information (PNMI) to evaluate the quality of the obtained cluster assignments from LA function in different layers: We obtain phonetic transcripts that are aligned at the frame level to quantify the correlation between the LA assignments and the underlying phonetic units.

Let $[y_1, ..., y_t]$ and $[f_1, ..., f_t]$ be frame-level and LA function labels, respectively. The joint distribution of y and f is the normalized number of occurrences of the labels:

$$p_{y,f}(i,j) = \frac{\sum_{t=1}^{T}[y_t = i \wedge f_t = j]}{T} \tag{5.6}$$

where i and j demonstrate the i^{th} phoneme class and j^{th} LA function class label [47]. Phone Purity measures the frame-level phone accuracy if we transcribe each LA function class with the most likely phone label. It is defined as $\mathbb{E}_{p_f(j)}\Big[p_{y|f}(y^*(j)|j)\Big]$, where $p_{y|f}(y^*(j)|j)$ is the conditional

probability of phone given a class label j and $y^*(j)$ is the most likely phone label for the j-th class. Higher purity indicates greater quality.

PNMI is an information-theoretic metric used to measure the similarity between two clusterings of data. It measures the percentage of uncertainty about the phone label y that is reduced after observing the class label f and is defined as follows, where H(.) is the entropy:

$$\frac{I(y,f)}{H(y)} = 1 - \frac{H(y|f)}{H(y)} \tag{5.7}$$

A higher value of PNMI in our analysis indicates that the quality of LA clustering is better.

5.3.3 Results

Table 5.1 presents the ASR performance of LABERT in terms of the WER when tested on I-CUBE and UASpeech LRE datasets, after being pre-trained and fine-tuned on I-CUBE and UASpeech, respectively. Comparisons are reported for wav2vec 2.0 and HuBERT in Base and Large settings, as well as DiscreteBERT [233] and QuartzNet[103]. We show that the performance of LABERT is improved by increasing the amount of unlabeled data during pre-training (see Section 6.3.1) which indicates the scalability of the proposed model. In the Base setup, after fine-tuning on I-CUBE, LABERT achieved WERs of 13.39%, 14.78%, 14.93% and 17.35% when pre-trained on LibriSpeech, TED, WSJ and CV corpora, respectively, which outperformed the other algorithms in the Base setting. LABERT achieved even better results in the Large setting when tested on I-CUBE dataset, with WER of 9.53%, 10.24%, 12.21% and 16.63% after pre-training over LibriSpeech, TED, WSJ and CV, respectively. LABERT significantly

LRE	Method	Libri	TED	WSJ	CV
I-CUBE	**LABERT – Base**	**13.39**	**14.78**	**14.93**	**17.35**
	LABERT – Large	**9.53**	**10.24**	**12.21**	**16.63**
	wav2vec 2.0 – Base	17.38	15.45	16.61	18.42
	wav2vec 2.0 – Large	11.61	13.64	14.73	17.22
	HuBERT – Base	16.81	16.43	15.98	18.13
	HuBERT – Large	11.28	12.71	14.39	16.81
	QuartzNet	26.51	29.75	28.39	31.53
	DiscreteBERT	27.93	31.35	29.48	33.38
UASpecch	**LABERT – Base**	**17.28**	**18.65**	**21.13**	**23.91**
	LABERT – Large	**11.27**	**12.28**	**15.11**	**17.93**
	wav2vec 2.0 – Base	19.07	21.31	23.94	25.18
	wav2vec 2.0 – Large	14.28	15.91	16.23	18.87
	HuBERT – Base	19.31	21.18	24.49	25.93
	HuBERT – Large	14.93	15.58	16.39	18.98
	QuartzNet	29.15	34.93	31.79	36.75
	DiscreteBERT	31.48	36.75	32.19	37.21

Table 5.1: Word error rate (WER) results obtained with different methods pretrained in HRE datasets (Libri, TED, WSJ and CV) and fine-tuned in two LREs (I-CUBE and UASpeech). The best performing models in corresponding settings are highlighted.

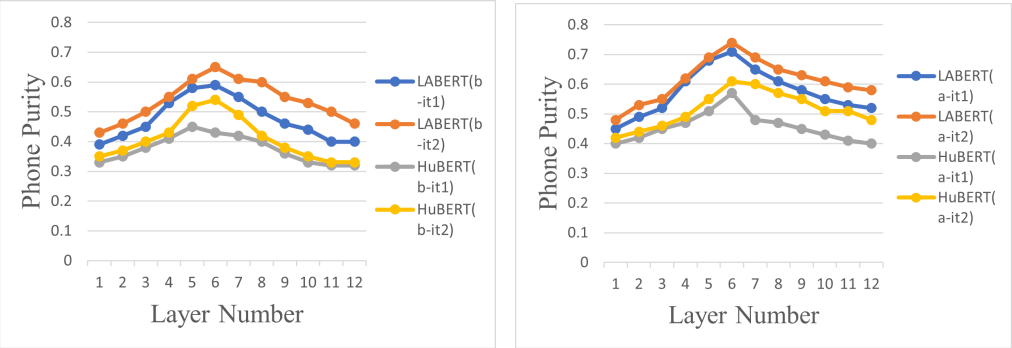
outperformed QuartzNet and DiscreteBERT as well. Similarly, after fine-tuning over UASpeech on Base and Large settings, LABERT achieved best results across all benchmark algorithms.

The PNMI results are shown in Table 5.2. These results demonstrate that the PNMI increases with the amount of pre-training speech data, which enhance the quality of the cluster results. A possible explanation for this might be that by increasing the pre-training data, the committee-based active learning approach can select more informative speech units for seed initialization of the LA function, therefore LABERT can improve the quality of the clusters in LRE tasks.

Finally, we evaluate the quality of the local aggregation function for detecting hidden units in each layer of LABERT. In this analysis, we considered the first two iterations of the LABERT after pre-training the model on LibriSpeech dataset and fine-tuning it with I-CUBE and UASpeech. The

Feature	Number of Clusters	PNMI		
		100h	300h	500h
MFCC	50	0.384	0.387	0.338
	100	0.432	0.435	0.435
Selected Layer From CCA	50	0.631	0.633	0.633
	100	0.785	0.787	0.787

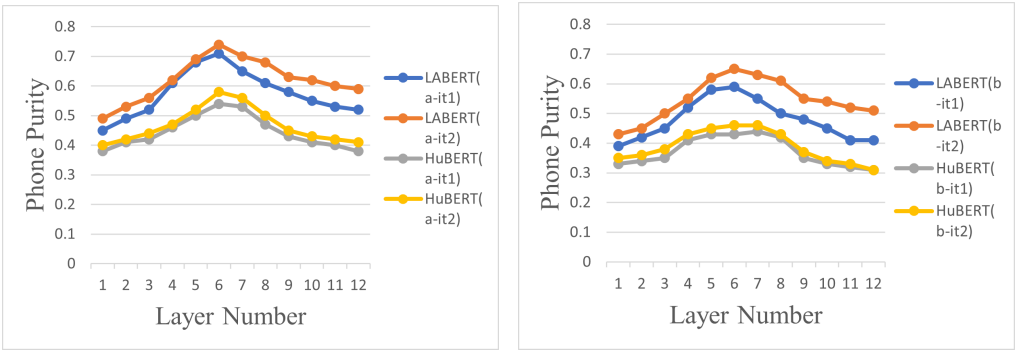
Table 5.2: PNMI values for different cluster numbers and pre-training data size. Fine-tuning is done using I-CUBE.



(a) I-CUBE (b) UASpeech

Figure 5.2: Phone purity of LABERT and HuBERT in Base configuration after pre-training on LibriSpeech and fine-tuning over I-CUBE and UASpeech for the first and second iteration.

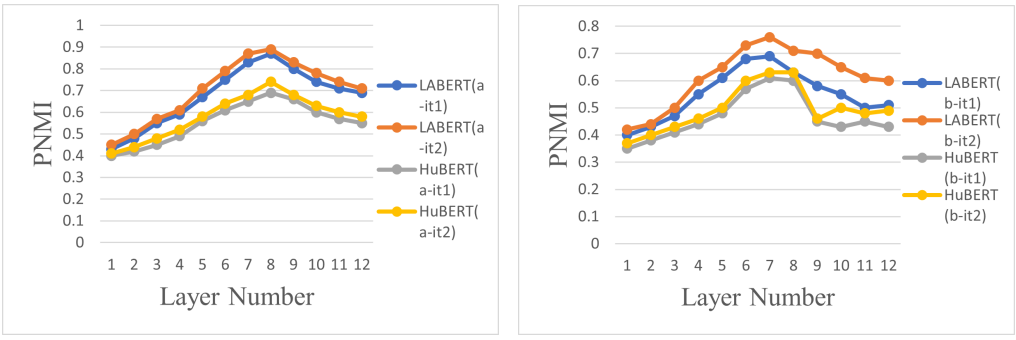
results are compared with HuBERT since it achieved the next best WER results in the earlier analysis. Phone Purity and PNMI are shown in Figures 5.2 and 5.4, for each layer of the model after pre-taining on LibriSpeech dataset, respectively. We observe that the phone purity gradually increased in the first layers after pre-training and fine-tuning with both I-CUBE and UASpeech. The interesting finding is that in the last layers of both models, phone purity decreased. The same trend is observed in the PNMI after fine-tuning the models with both I-CUBE and UASpeech. The middle layers (7-9) of the LABERT, which were selected by the CCA module to feed into the AL model for selecting initial seeds for hidden units detection process, exhibited the highest PNMI. In both LRE settings, fine-tuning on I-CUBE and UASpeech, significant phone purity and PNMI results were observed at these middle layers, suggesting that they are well-suited for the downstream LRE ASR task. In addition, Phone Purity and PNMI of the



(a) I-CUBE (b) UASpeech

Figure 5.3: Phone purity of LABERT and HuBERT in Base configuration after pre-training on TED and fine-tuning over I-CUBE and UASpeech for the first and second iteration.

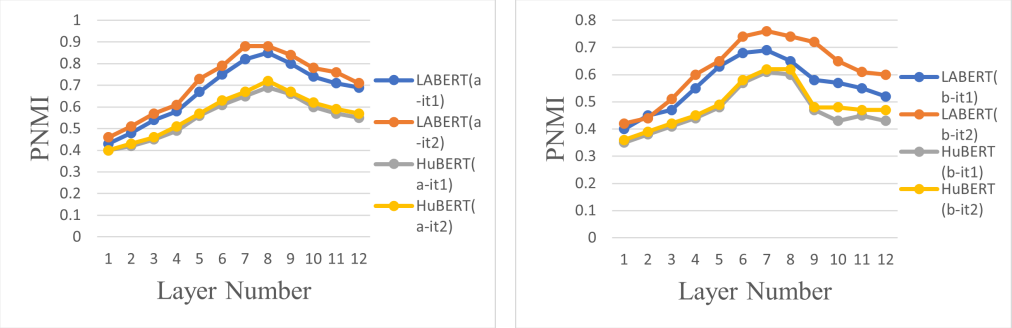
LABERT after pre-taning on TED dataset and fine-tuning on I-CUBE and UASpeech are presented in Figures 5.3 and 5.5, respectively. It can be seen the same trend in phone-purity and PNMI in LABERT after pre-taning on TED and these results reflect the performance of the layer analysis in LABERT to detect better layers in the model for second iteration of the training. Notably, the phone purity and PNMI analysis revealed that the LABERT model demonstrates more stable clusters, indicating that it performs reasonably well for low-resource ASR tasks.



(a) I-CUBE (b) UASpeech

Figure 5.4: PNMI of LABERT and HuBERT in Base configuration after pre-training on LibriSpeech and fine-tuning over I-CUBE and UASpeech for the first and second iteration.

To better understand the properties of the discrete units learned by LABERT, we focused on the fifth layer of the model and computed cluster purity,



(a) I-CUBE (b) UASpeech

Figure 5.5: PNMI of LABERT and HuBERT in Base configuration after pre-training on TED and fine-tuning over I-CUBE and UASpeech for the first and second iteration.

Method	Cluster purity	Phone purity	PNMI
MFCC	0.06	0.30	0.28
HuBERT-iter1	0.15	0.60	0.60
HuBERT-iter2	0.15	0.61	0.61
VQ-APC	0.08	0.24	0.19
LABERT-iter1	**0.17**	**0.70**	**0.70**
LABERT-iter2	**0.17**	**0.72**	**0.71**

Table 5.3: Discrete Unit Quality on LibriSpeech Dev Set. Fine-tuning is done using I-CUBE.

phone purity, and phone-normalized mutual information (PNMI) and the results are presented in Table 5.3. To calculate these metrics, we use forced alignment to obtain the ground truth phone for each feature frame in the LibriSpeech development clean and development other sets. We used the MFCC clusters, which are used to train the first iteration of HuBERT, as a baseline for purity and PNMI. The first and second iterations of HuBERT, which served as the teacher in HuBERT's iterative pre-training procedure, showed a significant improvement over MFCCs. However, performing LA function clustering on LABERT produced even better quality clusters. Overall, LABERT achieved comparable phone purity and PNMI to other methods, while being more efficient.

5.4 Chapter Summary

In this chapter, we propose LABERT, a self-supervised speech representation learning model for ASR in low-resource environments. LABERT combines a committee-based active learning model for informative speech unit selection with a local aggregation function for hidden unit detection. The local aggregation function learns feature embeddings that cluster similar speech units while separating dissimilar ones. Pre-trained on four high-resource datasets and fine-tuned on two LRE datasets, our model demonstrates up to a 16.63% WER reduction on LR data, surpassing the performance of state-of-the-art ASR models. This demonstrates that LABERT generates representations highly effective for speech recognition tasks in low-resource settings. In the following chapter, we explore the potential benefits of using regularization terms to help the ASR model towards more informative speech units and investigate the use of contrastive clustering to enhance the quality of clusters derived from speech data.

Chapter 6

Regularized Contrastive Clustering for Detecting Speech Hidden Units

To develop a reliable automatic speech recognition system, a substantial amount of high-quality labeled data is needed. This becomes a challenging problem when dealing with speech recognition tasks that are resource-limited. Nowadays, self-supervised contrastive learning is making significant strides in such low-resource contexts. In this chapter, we introduce **RCCBERT** (**R**egularized **C**ontrastive **C**lustering **BERT**) method for self-supervised speech representation learning. This method leverages a contrastive clustering step to generate aligned target labels for a BERT-like prediction loss. RCCBERT relies primarily on the consistency of the contrastive clustering step rather than general purpose clustering model to enhance the model to minimize the inter-cluster similarities to separate different clusters. Given that speech variations are gradual and subtle over a short span, this nuances the definition of negative samples in contrastive

clustering. The approach of the definition of negative samples can ignore the fact that samples from different parts of the same speech, may also have the same pronunciation as the positive samples. To address this, we incorporated regularization terms into the contrastive clustering process to impose constraints on the features, aiming to achieve a more refined representation in low-resource situations. This integration forces RCCBERT to learn better speech representations and achieve superior results in low-resource speech recognition tasks. We evaluate RCCBERT with two LRE datasets: I-CUBE and UASpeech to explore the performance of our model in the LRE ASR problems.

6.1 Introduction

ASR systems facilitate the transcription of spoken language into text. While traditional ASR approaches demonstrate efficacy, recent advancements indicate superior performance from end-to-end frameworks [242]. Unlike traditional models, which necessitate distinct components for acoustic, pronunciation, and language modeling, end-to-end systems offer a more integrated and streamlined training process [243]. However, a key challenge with end-to-end models is their reliance on vast amounts of well-labeled data. This poses a problem when dealing with low-resource tasks where labeled speech data is limited. Therefore, finding ways to improve ASR performance in these low-resource scenarios is a crucial research area.

Self-supervised learning methods have emerged as a popular solution for training ASR systems in low-resource scenarios. Contrastive learning, a prominent SSL technique which trains a network to discern similarities and dissimilarities between data and enables the model to learn more intricate representations, which allows the model to understand more complex features [244]. The quality of representations learned through contrastive learning relays on the quality of negative samples – data point pairs definitively classified as dissimilar. While image transformations frequently generate negative samples in computer vision [245], strategies vary. These include carefully selecting positive samples [246], filtering out ineffective negative samples [247], increasing negative sample volume [248], and constructing more challenging negative examples [249]. Such techniques have demonstrably enhanced the performance of contrastive learning models across various tasks.

Many contrastive learning models have been proposed for ASR tasks. Contrastive Predictive Coding (CPC) [177] is a popular self-supervised method

which uses autoregressive model to predict future speech frames from the current speech information. It utilizes an autoregressive approach combined with noise-contrastive estimation, which helps the model to filter out less relevant details and noise. wav2vec model [191] incorporates the concepts from CPC to introduce a noise-contrast learning approach for binary classification. This enables wav2vec to utilize vast amounts of unlabeled data and subsequently enhance its performance in extracting features for speech recognition tasks. vq-wav2vec [195] introduces a quantization module to convert the feature space from being continuously infinite to discretely finite. By merging with BERT, it enables the model to replace the traditional acoustic features with speech representation. Wav2vec 2.0 model [46] introduces a novel self-supervised structure using contrastive learning that integrates the Gumbel softmax quantization module with BERT from vq-wav2vec into a single model. JUST [250] enhances the wav2vec 2.0 framework by incorporating self-supervised techniques that utilizes both contrastive loss and MLM loss approaches. It also employs supervised RNN-T loss [251] for combined training, aiming for improved accuracy in multilingual settings with limited resources. wav2vec-S [252] is a semi-supervised pre-training approach based on wav2vec 2.0 that optimizes the pre-training of the models in the low-resource speech recognition scenarios.

Clustering techniques are utilized to generate pseudo-labels for SSL. Deep Cluster model [172] employs the k-means algorithm to group similar samples and optimizing an encoder network through a classification loss. HuBERT [47] incorporates an offline clustering phase to present noisy labels for a BERT-like prediction loss. SwAV [41] applies an online clustering process to create pseudo-labels within a mini-batch format, while JULE [253] adopts an iterative approach, progressively combining data points

and utilizing the resulting clusters to guide the learning of distinctive representations. However, this learning strategy that transitions between representation learning and clustering stages can lead to degrading clustering quality. To address this challenge, Contrastive Clustering (CC) model [254] is proposed, which is a one-stage online deep clustering method. CC employs a deep network to learn a feature matrix, in where the rows represent instance representations and the columns present cluster representations. Essentially, this method views the label as a special representation by mapping input instances into a specific space determined by the number of clusters. The matrix's rows can be seen as the likelihood of a particular cluster assignment (or soft labels for instances), while the columns indicate cluster distributions over instances (i.e., cluster representations).

While numerous contrastive learning models have emerged for low-resource ASR tasks, the process of choosing negative speech samples has not been addressed adequately. Since variations in speech are often subtle and continuous over short spans, designating negative samples from different segments of the same speech overlooks the possibility of them having identical pronunciations to the positive samples. Recognizing these differences is crucial in low-resource speech recognition. In this chapter, we proposed **RCCBERT** (**R**egularized **C**ontrastive **C**lustering **BERT**) model to leverage a contrastive clustering step to generate aligned target labels for a BERT-like prediction loss. In this chapter, we utilized regularizing constraints to control the slow changes in the latent representations and optimize the corresponding loss function to eliminate the impact of negative samples in contrastive clustering, which allows the model to learn better speech representations in the low-resource ASR problems.

6.2 Proposed Approach

RCCBERT is a speech representation model which proposed to enhance speech recognition performance in LRE ASR scenarios, by making efficient use of speech-only data. RCCBERT includes two main components: a) the extraction of hidden units from raw audio using the Contrastive Clustering (CC) module, and b) predicting masked target units. The CC function clusters similar audio units closely in an embedding space and keeps dissimilar ones apart. To overcome the issue of slow changes in speech data to have a better negative sample selection within this function, we have used regularization terms to force the model to minimize the inter-cluster similarities to separate different clusters. Like LABERT model [151] in the previous chapter, MFCCs are utilized for clustering. Later stages employ chosen representations from the CCA module [225]. The second module employs an MLM objective, similar to the approach in BERT [235], to predict masked hidden units. It achieves this by calculating the cosine similarity between context vectors and all hidden unit embeddings, with predictions evaluated using cross-entropy loss.

6.2.1 Regularized Contrastive Clustering

In RCCBERT, Contrastive Clustering [254] method is adapted to speech data to extract hidden speech units for low-resource scenario by adding regularization constrains inspired from [255]. The CC module is composed of three jointly learned components: Pair Construction Backbone (PCB), an Instance-Level Contrastive Head (ICH), and a Cluster-Level Contrastive Head (CCH). Essentially, PCB creates data pairings using data augmentations and derives features from these augmented samples. Following this,

ICH and CCH apply contrastive learning to the rows and columns of the feature matrix, respectively. Upon completion of the training, cluster can be derived from the soft labels predicted by the CCH part. CC employs data augmentations to form pairs of data. For a given data, x_i, two random transformations, T^a and T^b, from a consistent augmentation set are used. This process produces two samples, represented as $x_i^a = T^a(x_i)$ and $x_i^b = T^b(x_i)$. Previous research indicates the importance of selecting the appropriate augmentation approach for optimal performance in subsequent tasks and in CC model, five data enhancement techniques are utilized: ResizedCrop, ColorJitter, Grayscale, HorizontalFlip, and GaussianBlur. One shared deep neural network is used to extracted features from the augmented samples via $h_i^a = f(x_i^a)$ and $h_i^b = f(x_i^b)$.

Contrastive learning seeks to detect the similarities between positive pairs and reduce them between negative pairs. In CC, due to the absence of pre-existing labels for clustering, pairs, both positive and negative, are formed at the instance level based on pseudo-labels created by data augmentations. Specifically, positive pairs are formed from samples augmented from a single instance, while negative pairs are formed from different instances. To counteract the data loss caused by the contrastive loss, CC does not immediately apply contrastive learning to the feature matrix. Instead, a two-layer nonlinear MLP is used to project the feature matrix into a subspace, represented as $z_i^a = MLP(h_i^a)$, where the instance-level contrastive loss is used. The similarity between pairs is determined using the cosine distance.

$$S(z_i^{k_1}, z_j^{k_2}) = \frac{(z_i^{k_1})(z_j^{k_2})^T}{\left\|z_i^{k_1}\right\| \left\|z_j^{k_2}\right\|} \tag{6.1}$$

in which $k_1, k_2 \in a, b$ and $i, j \in [1, N]$. In CC, the loss for a given data

sample x_i^a is:

$$l_i^a = -log \frac{exp(s(z_i^a, z_i^b))/\tau_I)}{\sum_{j=1}^{N}[exp(s(z_i^a, z_j^a))/\tau_I) + exp(s(z_i^a, z_j^b))/\tau_I)]} \tag{6.2}$$

where τ_I is the instance-level temperature parameter. So, the instance-level contrastive loss is computed as follows:

$$\iota_{instance} = \frac{1}{2N} \sum_{i=1}^{N} (l_i^a + l_i^b). \tag{6.3}$$

Based on the concept of label as the representation, when a data sample is mapped into a space with dimensions equal to the number of clusters, the i-th component of its feature can be seen as its likelihood of being part of the i-th cluster. Consequently, the feature vector signifies its soft label. Therefore, consider Y^a as the output from CCH for a mini-batch using the first augmentation (with Y^b being the result of the second augmentation). The value $Y_{n,m}^a$ represents the probability of the n-th sample being assigned to the m-th cluster, given that N is the batch size and M is the number of clusters. Like the instance-level contrastive head, a two-layer MLP is used to map the feature matrix into a M-dimensional space y^a.

$$S(y_i^{k_1}, y_j^{k_2}) = \frac{(y_i^{k_1})(y_j^{k_2})^T}{\left\|y_i^{k_1}\right\| \left\|y_j^{k_2}\right\|} \tag{6.4}$$

where $k_1, k_2 \in a, b$ and $i, j \in [1, M]$ and the loss function is calculated as follows:

$$l_i^a = -log \frac{exp(s(y_i^a, y_i^b))/\tau_C)}{\sum_{j=1}^{M}[exp(s(y_i^a, y_j^a))/\tau_C) + exp(s(y_i^a, y_j^b))/\tau_C)]} \tag{6.5}$$

where τ_C is the cluster-level temperature parameter. The cluster-level con-

trastive loss is computed as follows:

$$\iota_{cluster} = \frac{1}{2M}\sum_{i=1}^{M}(l_i^a + l_i^b) - H(Y) \tag{6.6}$$

where $H(Y)$ is the entropy of cluster assignment probabilities and computed as follows:

$$H(Y) = \sum_{i=1}^{M}[P(y_i^a)logP(y_i^a) + P(y_i^b)logP(y_i^b)]. \tag{6.7}$$

The optimization of ICH and CCH occurs in a single, unified step. Both heads are optimized concurrently, and the overall goal combines both the instance-level and cluster-level contrastive losses.

The multi-dimensional nature of audio data, encompassing spectral and temporal characteristics, necessitates speech recognition models capable of discriminating between subtle auditory variations. Contrastive learning plays a crucial role in this domain, leverages both positive samples, which exhibit similar speech patterns, and negative samples, characterized by their dissimilarity, to train models in the discernment of fine-grained auditory distinctions. The quality of the negative samples is a critical element on contrastive learning to obtain an efficient model, where homogeneous or redundant examples can constrain the performance of the model to capture the broad spectrum of auditory differences in the speech data.

We implemented the **Regularization for Negative Sample Diversity** approach to optimize negative sampling within our contrastive learning framework, aiming to improve the model's ability to discriminate between speech patterns. This regularization encourages a diverse and varied set of negative samples, ensuring that the model remains consistently exposed to

a broad spectrum of contrasting auditory examples. The negative diversity regularization is defined as follows:

$$\iota_{negative\_diversity} = -Var(d_{i,j}). \tag{6.8}$$

where $d_{i,j}$ denotes the distance between the embeddings of a pair of negative speech samples x_i and x_j. By maximizing the variance, $Var(d_{ij})$, we ensure that the distances between various pairs of negative samples cover a wide range, thereby enhancing the diversity within the negative sample space. Speech data is characterized by its temporal dynamics, nuances, and variations across different speakers, accents, and phonemes. By ensuring diversity in the negative samples, we improve the discriminative capabilities of the RCCBERT model in LRE ASR scenarios. In our framework, the regularization terms considered as an approach to prevent the overfitting of the model to a limited set of contrasting examples. This is particularly critical in low-resource speech recognition, where audio data may significantly differ from the training set.

Variance-Invariance-Covariance Regularization (VICReg) [255] is a self-supervised method for training joint embedding architectures that emphasizes the preservation of information within embeddings. Its core principle lies in a loss function comprising three terms:

- **Variance Term**: Maintains the variance of each embedding dimension above a threshold, preventing representational collapse.
- **Invariance Term**: Encourages similarity between embeddings derived from different augmentations of the same input data.
- **Covariance Term**: Regularizes the covariance between embedding dimensions, promoting decorrelation and ensuring the extraction of

distinct features.

The covariance criterion in VICReg model aims to prevent informational collapse caused by redundancy among embedding variables. It achieves this by promoting decorrelation between embedding dimensions, ensuring that each dimension captures distinct and non-overlapping aspects of the input data. Inspired by VICreg, we define the covariance regularization and the covariance matrix is defined as follows:

$$C(Z) = \frac{1}{n-1}\sum_{i=1}^{n}(z_i - \bar{z})(z_i - \bar{z})^T \tag{6.9}$$

where $\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i$.

Then, the covariance regularization term c is defined as:

$$c(Z) = \frac{1}{d}\sum_{i \neq j}[C(Z)]_{i,j}^2 \tag{6.10}$$

where $\frac{1}{d}$ is a factor that scales the criterion as a function of the dimension.

Finally, the overall loss function is a weighted average of the instance-level and cluster-level contrastive loss, negative sample diversity and covariance terms:

$$l = \alpha\iota_{instance} + \beta\iota_{cluster} + \gamma\iota_{negative\_diversity} + \delta\iota covariance \tag{6.11}$$

where α, β, γ and δ are hyper-parameters controlling the importance of each term in the loss.

6.2.2 Masked Target Unit Prediction

In this section, we provide an overview of utilizing the BERT model [194] and discuss how to select an optimal representation from BERT for the second iteration of the CC module. Similar to the previous chapter, RCCBERT model adopts the masked prediction model for pretraining on a large text corpus, mitigating the lack of text data in low-resource ASR tasks. However, to address train-test inconsistencies, only a percentage $p\%$ of selected timesteps are masked. Inspired by [150], RCCBERT incorporates a layer analysis module to identify the most suitable BERT layer for the target LRE ASR task in the CC function's second iteration. CCA [225] measures the maximum correlations between linear combinations of two continuous-value vectors. We apply CCA to assess the similarity between layer representations and acoustic feature vectors, determining the degree of each layer's adaptation to the downstream task. This strategy forces the model to learn task-specific representations, potentially enhancing performance in the target domain.

6.3 Experiments

6.3.1 Datasets

For unsupervised pre-training, we employ a high-resource environment dataset consisting of: 960 hours of LibriSpeech, 81 hours of Wall Street Journal (WSJ), 1,000 hours of Common Voice (CV), and 450 hours of TED-LIUM 3 (TED3). We evaluate RCCBERT's performance in low-resource environments using UASpeech and I-CUBE datasets.

6.3.2 Experiment Setup and Metrics

During pre-training, RCCBERT is trained using the chosen HRE datasets in two distinct settings: Base Setting: Utilizing a 12-layer encoder and Large Setting: Employing a 24-layer encoder. For model configuration, we set the mask span to $l = 10$ and $p = 8\%$ of the waveform encoder output frames are randomly selected as the initial mask. We considered 50 and 100 clusters on 100h, 300h and 500h of speech samples from LibriSpeech and fine-tuned the model with I-CUBE for cluster quality analysis. The input acoustic features are 80-dimensional filterbanks, extracted with a hop size of 10 ms and a window size of 25 ms, which are normalized with the mean and variance. For the WSJ setup, the number of output classes is 52, including the 26 letters of the alphabet, space, noise, symbols such as period and an unknown marker. To predict the probability distribution of all characters in the alphabet, we use the CTC loss function and use AdamW optimizer [219] to update the model with an initial learning rate of 0.001. The text is tokenized using SentencePiece [241] and we set the vocabulary size to 500. Benchmarking results are presented for the pre-trained and fine-tuned wav2vec 2.0 and HuBERT models in Base and Large settings, as well as for QuartzNet and DiscreteBert. The primary evaluation metric we used is the WER. We also compute the Phone Purity and Phone-Normalized Mutual Information (PNMI) to evaluate the quality of the obtained cluster assignments from LA function in different layers: We obtain phonetic transcripts that are aligned at the frame level to quantify the correlation between the LA assignments and the underlying phonetic units. Similar to the previous chapter, we employ Phone Purity and PNMI metrics to measure the similarity between two clusterings of speech data.

6.3.3 Results

Table 6.1 demonstrates the ASR performance of RCCBERT after pre-training and fine-tuning on the I-CUBE and UASpeech LRE datasets, respectively. Results are presented in terms of WER and compared to baselines including wav2vec 2.0, HuBERT (Base and Large settings), DiscreteBERT [1], and QuartzNet [2]. Notably, RCCBERT's performance scales with increased unlabeled data during pre-training. In the Base setting, after I-CUBE fine-tuning, RCCBERT achieved WERs of 12.58%, 14.32%, 14.21% and 16.48% when pre-trained on LibriSpeech, TED, WSJ, and CV, respectively – outperforming all other Base models. Enhanced results were observed in the Large setting on the I-CUBE dataset, with WERs of 8.91%, 9.93%, 11.77% and 15.28%. RCCBERT also significantly outperformed QuartzNet and DiscreteBERT. Similar superiority was observed after fine-tuning on UASpeech, with RCCBERT achieving the lowest WERs on both Base and Large settings across all benchmarks.

Table 6.2 demonstrates a positive correlation between the volume of pre-training speech data and the PNMI metric, suggesting improved clustering performance. This trend likely stems from the regularized contrastive learning method, which, with increased data, encourages the model to minimize inter-cluster similarities, leading to better cluster separation. Consequently, RCCBERT demonstrates enhanced clustering quality in LRE tasks.

To evaluate the ability of the RCCBERT to detect hidden units within different layer, we analyzed the first two iterations after pre-training on LibriSpeech and fine-tuning on I-CUBE and UASpeech datasets. Results were compared with HuBERT due to its strong performance in the previous analysis. Figures 6.1 and 6.2 illustrate Phone Purity and PNMI metrics

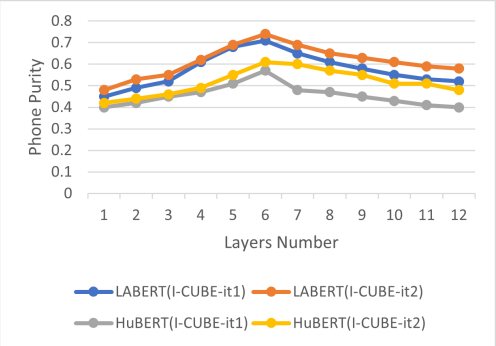
LRE	Method	Libri	TED	WSJ	CV
I-CUBE	**RCCBERT – Base**	**12.58**	**14.32**	**14.21**	**16.48**
	RCCBERT – Large	**8.91**	**9.93**	**11.77**	**15.28**
	wav2vec 2.0 – Base	17.38	15.45	16.61	18.42
	wav2vec 2.0 – Large	11.61	13.64	14.73	17.22
	HuBERT – Base	16.81	16.43	15.98	18.13
	HuBERT – Large	11.28	12.71	14.39	16.81
	QuartzNet	26.51	29.75	28.39	31.53
	DiscreteBERT	27.93	31.35	29.48	33.38
UASpecch	**RCCBERT – Base**	**16.63**	**17.21**	**20.88**	**23.12**
	RCCBERT – Large	**10.81**	**11.66**	**14.68**	**17.09**
	wav2vec 2.0 – Base	19.07	21.31	23.94	25.18
	wav2vec 2.0 – Large	14.28	15.91	16.23	18.87
	HuBERT – Base	19.31	21.18	24.49	25.93
	HuBERT – Large	14.93	15.58	16.39	18.98
	QuartzNet	29.15	34.93	31.79	36.75
	DiscreteBERT	31.48	36.75	32.19	37.21

Table 6.1: Word error rate (WER) results obtained with different methods pretrained in HRE datasets (Libri, TED, WSJ and CV) and fine-tuned in two LREs (I-CUBE and UASpeech). The best performing models in corresponding settings are highlighted.

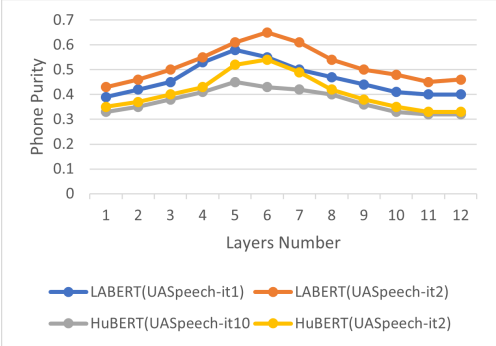
Feature	Number of Clusters	PNMI		
		100h	300h	500h
MFCC	**50**	0.386	0.393	0.41
	100	0.446	0.449	0.452
Selected Layer From CCA	**50**	0.667	0.683	0.695
	100	0.816	0.823	0.834

Table 6.2: PNMI values for different cluster numbers and pre-training data size. Fine-tuning is done using I-CUBE.

for each model layers, respectively. Phone purity showed an initial increase within the first layers after pre-training and fine-tuning on both datasets. Interestingly, this metric decreased in the final layers of both models. Fine-tuning on I-CUBE and UASpeech also revealed a similar trend in PNMI. RCCBERT's middle layers (7-9), selected by the CCA module, consistently demonstrated the highest PNMI. Significant phone purity and PNMI within these middle layers in both LRE settings suggest their suitability for the downstream ASR task. Overall, this analysis indicates that RCCBERT maintains stable clusters, demonstrating its effectiveness in low-resource



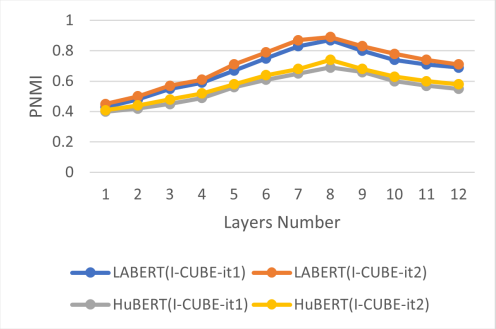
(a) I-CUBE



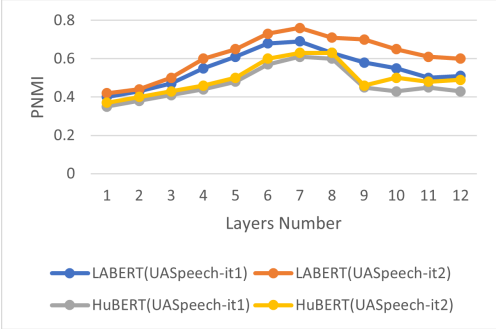
(b) UASpeech

Figure 6.1: Phone purity of RCCBERT and HuBERT in Base configuration after pre-training on LibriSpeech and fine-tuning over I-CUBE and UASpeech for the first and second iteration.

ASR scenarios.



(a) I-CUBE



(b) UASpeech

Figure 6.2: PNMI of RCCBERT and HuBERT in Base configuration after pre-training on LibriSpeech and fine-tuning over I-CUBE and UASpeech for the first and second iteration.

6.4 Chapter Summary

In this chapter, we introduced RCCBERT, a self-supervised speech representation model for ASR within low-resource environments. RCCBERT combines regularization constrains with the Contrastive Clustering (CC) approach for detecting hidden units. It employs a clustering-oriented regularized contrastive learning approach to force the model to minimize the inter-cluster similarities to separate different speech units into different

clusters. We initially train RCCBERT on four well-known high-resource datasets, and then fine-tune it using two datasets from low-resource environments. Ultimately, RCCBERT learns representations, which are useful to a variety of speech recognition tasks in low-resource settings.

Chapter 7

Conclusions

7.1 The Problem

This dissertation aims to answer the question: How can we effectively use limited datasets to develop Acoustic Models for Automatic Speech Recognition using self-supervised? We chose to focus on self-supervised learning approaches because they do not require the collection of more in-domain data. Consequently, the techniques explored in this study can be employed on any speech recognition dataset.

7.2 Thesis Summary

This thesis explored the use of self-supervised learning to enhance automatic speech recognition performance in low-resource environments. Chapter 3 presents a rigorous analysis of deep learning architectures specifically designed for this challenge. Understanding the importance of data in low-resource settings, we systematically varied the available pre-training data

from 10% to 100%. Evaluating model performance across these data segments revealed the impact of data availability on each architecture. This approach allowed us to identify models demonstrating superior adaptability and resilience, while also establishing performance thresholds linked to varying data volumes within low-resource ASR tasks.

In Chapter 4, we explore how out-of-domain datasets can boost performance within low-resource ASR. Our novel ScoutWav model integrates self-supervised learning, context-sensitive word boundary detection, and a two-stage fine-tuning process which proposed for low-resource environments ASR problems. Initially, we pre-train a wav2vec 2.0 model on a high-resource dataset. Then, an SN network leverages context vector embedding to extract both local acoustic and global context information for high-quality word boundary data. This data drives the first fine-tuning stage, adapting the model to the LR task. To further enhance performance, we perform layer analysis on the stacked layers of the Transformer, identifying layers with poor acoustic-linguistic representation. These layers considered for the second fine-tuning step using the context-driven word boundary data, thereby reinforcing ScoutWav's ability to understand global context of the speech data, especially in low-resource scenarios.

In Chapter 5, we present LABERT, an innovative self-supervised learning model to obtain speech representations in LRE tasks. While drawing inspiration from HuBERT, LABERT incorporates an offline hidden unit detection module that provides noisy labels to a pre-training model similar to BERT. Instead of relying on global clustering methods to detect hidden units, LABERT employs a non-parametric approach within a latent space for visual embedding. To tackle the scarcity of training data in low-resource environments, we integrate a committee-centric active learning strategy alongside an LA function. This strategy enhances the LA's

ability to identify neighboring speech samples within the latent space. By continuously improving and fine-tuning the active learning model during training, LABERT effectively identifies speech units with similar characteristics, allowing them to be grouped together into clusters.

Chapter 6 introduces Regularized Contrastive Clustering BERT (RCCBERT), a model that combines contrastive clustering for target label generation with a BERT-style predictive loss. To further enhance latent representations, we apply regularization techniques to modulate subtle variations. This optimization of the loss function reduces the impact of negative samples during contrastive clustering. As a result, RCCBERT learns enhanced speech representations, especially in the context of low-resource ASR challenges.

This thesis explores the use of self-supervised learning to improve ASR performance in low-resource problems. Chapter 3 provides a crucial foundation by systematically analyzing the impact of data volume on different deep learning architectures designed for LREs. This analysis shoes how pretraining data availability influences model adaptability and performance, establishing benchmarks for subsequent chapters. Building on Chapter 3, Chapter 4 introduces ScoutWav, which strategically leverages out-of-domain datasets to boost ASR performance in low-resource settings. ScoutWav integrates self-supervised learning with context-aware word boundary detection and a two-stage fine-tuning process, demonstrating how high-resource data can be effectively adapted to low-resource tasks. This chapter advances the thesis by offering a novel, context-sensitive approach to model refinement for low-resource ASR. In chapter 5, LABERT model uniquely employs an offline hidden unit detection module and committee-based active learning approach, which significantly improving upon prior methods for LRE tasks. This chapter demonstrates a continued refinement of ASR

strategies, building upon the foundational understanding established in Chapter 3 and the model innovation showcased in Chapter 4. Chapter 6 presents Regularized Contrastive Clustering BERT (RCCBERT) which synthesizes the insights gleaned from earlier chapters, focusing on refined latent representations and the mitigation of negative sample impact. This chapter showcases a comprehensive approach to low-resource ASR, demonstrating the power of deep learning architectures, self-supervised learning strategies, and innovative fine-tuning methods developed throughout this work.

7.3 Thesis Contributions

The main contributions of this thesis are as follows:

- We conducted an extensive series of experiments to assess the performance of ASR models, which are typically optimized for high-resource environments, in resource-constrained scenarios. Our analysis reveals that simply augmenting training data from diverse domains does not necessarily improve the precision of ASR systems in low-resource settings. Contrary to expectations, complex model architectures do not yield significant improvement for low-resource environments, even if they perform well with abundant training data. Instead, we emphasize a practical strategy: starting with pre-training on data-rich language resources and subsequently fine-tuning using relevant in-domain data. This approach ensures optimal performance in low-resource ASR challenges.

- To address the challenges of ASR in low-resource settings, we present ScoutWav, a novel model integrating self-supervised learning and

context-aware word boundaries. ScoutWav leverages large, out-of-domain datasets to overcome limited training data within low-resource environments. Initially, a wav2vec 2.0 model is pre-trained on a high-resource dataset. An enhanced SN network with context vector embedding then extracts both local acoustic features and global context information for high-quality word boundary data. Our approach involves a two-stage fine-tuning process: First, we start by pre-training a wav2vec 2.0 model on a high-resource dataset. Then, we fine-tune the model using low-resource data, adapting it to the specific ASR task. Additionally, we recognize that different layers within a Transformer architecture capture varying levels of linguistic information. To address this, we perform a wav2vec 2.0 layer analysis to identify underperforming layers that inadequately capture acoustic-linguistic features. Subsequently, we enhance these layers through a second fine-tuning step, incorporating context-based word boundary data to imbue global context awareness into the ScoutWav model.

- We introduced LABERT, an innovative self-supervised speech representation model specifically designed for low-resource ASR applications. Inspired by HuBERT, LABERT leverages a unique offline hidden unit detection module. To address data scarcity, LABERT integrates committee-based active learning, ensuring the selection of informative speech units. LABERT utilizes non-parametric consolidation within the latent space for greater adaptability and scalability. We train LABERT to categorize speech units with shared statistical structures into clusters. This facilitates the selection of a diverse and representative data subset, optimizing the model's ability to learn robust representations for downstream LRE ASR tasks.

- We introduce RCCBERT, a novel low-resource ASR model that inte-

grates Contrastive Clustering (CC) for informative hidden unit identification. CC is an online deep clustering method that operates in a single stage. It employs a deep model to learn a feature matrix, where each row corresponds to instance representations, and each column corresponds to cluster representations. Essentially, CC interprets the rows as the likelihood of specific cluster assignments (or soft labels for instances) and the columns as the distributions of clusters over instances (i.e., cluster representations). RCCBERT builds upon CC by introducing regularizing constraints. These constraints enforce gradual changes in the latent representations, enhancing stability. Additionally, RCCBERT addresses the challenge of negative samples by overcoming their definition. By incorporating these improvements, RCCBERT effectively tackles data limitations in low-resource ASR scenarios. It learns well-suited representations for downstream tasks, ensuring optimal performance in low-resource environments.

7.4 Future Directions

The findings of this doctoral research illuminate multiple avenues for future exploration. A brief elaboration on some of these promising directions is provided as follows:

- The scarcity of labeled data poses a significant challenge within low-resource speech recognition. To address this limitation, constraint-oriented clustering models offer a compelling solution. By incorporating domain-specific constraints, these models facilitate accurate and meaningful clustering of speech data, even with limited datasets. This allows for more effective relationship discovery be-

tween speech segments, maximizing the utility of sparse data. Consequently, constraint-oriented clustering enhances speech recognition and understanding in low-resource scenarios. This approach holds significant promise for future research, potentially overcoming the challenges posed by data limitations in low-resource speech recognition.

- Modern speech generation systems, including text-to-speech synthesis and speech-to-speech translation, often utilize a single deep neural network. TTS converts text to speech waveforms, while S2S translates spoken utterances between languages. Generating high-quality, natural-sounding speech in both tasks requires robust speech understanding and linguistic insights. This thesis demonstrates that our self-supervised pre-training frameworks effectively initialize deep neural networks with strong acoustic representations. This initialization leads to superior performance when fine-tuned on both ASR and speech-to-text translation tasks.

- Recently, neural network architectures and self-supervised pre-training objectives have converged across different modalities, such as text, speech, and vision. In particular, training large Transformer models with masked language modeling-like objectives has become the dominant pre-training paradigm for all three modalities. This convergence makes it a natural and promising next step to build a single model that can learn cross-modal speech representations. Previous work on multi-modal pre-training of speech and text, as well as speech and vision, has relied heavily on the use of parallel data for supervised learning of cross-modal alignments. However, parallel data is more difficult to scale up than unpaired data. The fact that different modalities are now sharing similar neural architectures and

self-supervised pre-training objectives could potentially alleviate the models' reliance on parallel data. While some initial efforts have been made, there remains an extensive scope for further advancement and refinement, especially in low-resource ASR tasks.

Bibliography

[1] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, Jie Chen, Jingdong Chen, Zhijie Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Ke Ding, Niandong Du, Erich Elsen, Jesse Engel, Weiwei Fang, Linxi Fan, Christopher Fougner, Liang Gao, Caixia Gong, Awni Hannun, Tony Han, Lappi Johannes, Bing Jiang, Cai Ju, Billy Jun, Patrick LeGresley, Libby Lin, Junjie Liu, Yang Liu, Weigao Li, Xiangang Li, Dongpeng Ma, Sharan Narang, Andrew Ng, Sherjil Ozair, Yiping Peng, Ryan Prenger, Sheng Qian, Zongfeng Quan, Jonathan Raiman, Vinay Rao, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Kavya Srinet, Anuroop Sriram, Haiyuan Tang, Liliang Tang, Chong Wang, Jidong Wang, Kaifu Wang, Yi Wang, Zhijian Wang, Zhiqian Wang, Shuang Wu, Likai Wei, Bo Xiao, Wen Xie, Yan Xie, Dani Yogatama, Bin Yuan, Jun Zhan, and Zhenyao Zhu. Deep speech 2 : End-to-end speech recognition in english and mandarin. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 173–182, New York, New York, USA, 20–22 Jun 2016. PMLR.

[2] Laurent Besacier, Etienne Barnard, Alexey Karpov, and Tanja Schultz. Au-

tomatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56:85–100, 2014.

[3] Dong Wang, Xiaodong Wang, and Shaohe Lv. An overview of end-to-end automatic speech recognition. *Symmetry*, 11(8):1018, 2019.

[4] Vincent Roger, Jérôme Farinas, and Julien Pinquier. Deep neural networks for automatic speech processing: a survey from large corpora to limited data. *EURASIP Journal on Audio, Speech, and Music Processing*, 2022(1):19, 2022.

[5] Ramón López-Cózar, Zoraida Callejas, David Griol, and José F Quesada. Review of spoken dialogue systems. *Loquens*, 1(2):012, 2014.

[6] Piotr Szymański, Piotr Żelasko, Mikolaj Morzy, Adrian Szymczak, Marzena Żyła-Hoppe, Joanna Banaszczak, Lukasz Augustyniak, Jan Mizgajski, and Yishay Carmiel. WER we are and WER we think we are. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3290–3295, Online, November 2020. Association for Computational Linguistics.

[7] Ali Bou Nassif, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. Speech recognition using deep neural networks: A systematic review. *IEEE Access*, 7:19143–19165, 2019.

[8] 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)*, pages 4960–4964. IEEE, 2016.

[9] Mahbubul Alam, Manar D Samad, Lasitha Vidyaratne, Alexander Glandon, and Khan M Iftekharuddin. Survey on deep neural networks in speech and vision systems. *Neurocomputing*, 417:302–321, 2020.

[10] Tara N Sainath, Abdel-rahman Mohamed, Brian Kingsbury, and Bhuvana Ramabhadran. Deep convolutional neural networks for lvcsr. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8614–8618. IEEE, 2013.

[11] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.

[12] Hasim Sak, Andrew W. Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Interspeech*, 2014.

[13] Haşim Sak, Oriol Vinyals, Georg Heigold, Andrew Senior, Erik McDermott, Rajat Monga, and Mark Mao. Sequence discriminative distributed training of long short-term memory recurrent neural networks. In *Fifteenth annual conference of the international speech communication association*, 2014.

[14] Yajie Miao, Jinyu Li, Yongqiang Wang, Shi-Xiong Zhang, and Yifan Gong. Simplifying long short-term memory acoustic models for fast training and decoding. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2284–2288. IEEE, 2016.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[16] Dong Yu, Li Deng, and George Dahl. Roles of pre-training and fine-tuning in context-dependent dbn-hmms for real-world speech recognition. In *Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2010.

[17] Tara N Sainath, Brian Kingsbury, Bhuvana Ramabhadran, Petr Fousek, Petr Novak, and Abdel-rahman Mohamed. Making deep belief networks effective for large vocabulary continuous speech recognition. In *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 30–35. IEEE, 2011.

[18] Navdeep Jaitly, Patrick Nguyen, Andrew W. Senior, and Vincent Vanhoucke. Application of pretrained deep neural networks to large vocabulary speech recognition. In *Interspeech*, 2012.

[19] Dong Yu and Jinyu Li. Recent progresses in deep learning based acoustic models. *IEEE/CAA Journal of automatica sinica*, 4(3):396–409, 2017.

[20] Jinyu Li, Rui Zhao, Eric Sun, Jeremy HM Wong, Amit Das, Zhong Meng, and Yifan Gong. High-accuracy and low-latency speech recognition with two-head contextual layer trajectory lstm model. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7699–7703. IEEE, 2020.

[21] Wenyong Huang, Wenchao Hu, Yu Ting Yeung, and Xiao Chen. Conv-Transformer Transducer: Low Latency, Low Frame Rate, Streamable End-to-End Speech Recognition. In *Proc. Interspeech 2020*, pages 5001–5005, 2020.

[22] Jinyu Li, Liang Lu, Changliang Liu, and Yifan Gong. Improving layer trajectory lstm with future context frames. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6550–6554. IEEE, 2019.

[23] Linhao Dong, Shuang Xu, and Bo Xu. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888. IEEE, 2018.

[24] Hagen Soltau, Hank Liao, and Haşim Sak. Neural Speech Recognizer: Acoustic-to-Word LSTM Model for Large Vocabulary Speech Recognition. In *Proc. Interspeech 2017*, pages 3707–3711, 2017.

[25] François Hernandez, Vincent Nguyen, Sahar Ghannay, Natalia Tomashenko, and Yannick Estève. Ted-lium 3: twice as much data and corpus repartition for experiments on speaker adaptation. In *International Conference on Speech and Computer*, pages 198–208. Springer, 2018.

[26] Anthony Rousseau, Paul Deléglise, Yannick Esteve, et al. Enhancing the ted-lium corpus with selected data for language modeling and more ted talks. In *LREC*, pages 3935–3939, 2014.

[27] 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. IEEE, 2015.

[28] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. VoxCeleb2: Deep Speaker Recognition. In *Proc. Interspeech 2018*, pages 1086–1090, 2018.

[29] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. VoxCeleb: A Large-Scale Speaker Identification Dataset. In *Proc. Interspeech 2017*, pages 2616–2620, 2017.

[30] Douglas B Paul and Janet Baker. The design for the wall street journal-based csr corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*, 1992.

[31] Josh Meyer. *Multi-task and transfer learning in low-resource speech recognition*. PhD thesis, The University of Arizona, 2019.

[32] Samson Juan and Sarah Flora. *Exploiting resources from closely-related languages for automatic speech recognition in low-resource languages from Malaysia*. PhD thesis, Université Grenoble Alpes (ComUE), 2015.

[33] Fei Wu et al. *Child Speech Recognition as Low Resource Automatic Speech Recognition*. PhD thesis, Johns Hopkins University, 2020.

[34] Rui Yan. " chitty-chitty-chat bot": Deep learning for conversational ai. In *IJCAI*, volume 18, pages 5520–5526, 2018.

[35] Sining Sun, Ching-Feng Yeh, Mei-Yuh Hwang, Mari Ostendorf, and Lei Xie. Domain adversarial training for accented speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4854–4858. IEEE, 2018.

[36] Christopher Cieri, David Miller, and Kevin Walker. The fisher corpus: a resource for the next generations of speech-to-text. In *LREC*, volume 4, pages 69–71, 2004.

[37] John J Godfrey, Edward C Holliman, and Jane McDaniel. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on*, volume 1, pages 517–520. IEEE Computer Society, 1992.

[38] Meredith Moore, Hemanth Venkateswara, and Sethuraman Panchanathan. Whistle-blowing asrs: evaluating the need for more inclusive automatic speech recognition systems. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, volume 2018, pages 466–470, 2018.

[39] Allison Koenecke, Andrew Nam, Emily Lake, Joe Nudell, Minnie Quartey, Zion Mengesha, Connor Toups, John R Rickford, Dan Jurafsky, and Sharad Goel. Racial disparities in automated speech recognition.

Proceedings of the National Academy of Sciences, 117(14):7684–7689, 2020.

[40] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020.

[41] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

[42] Yuan Gong, Cheng-I Lai, Yu-An Chung, and James Glass. Ssast: Self-supervised audio spectrogram transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10699–10709, 2022.

[43] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[44] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *International Conference on Learning Representations*, 2019.

[45] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *Proceedings of the ieee/cvf International Conference on computer vision*, pages 6391–6400, 2019.

[46] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli.

wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460, 2020.

[47] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[48] Karl Wilmer Scholz, James S Irwin, and Samir Tamri. Dialogue flow interpreter development tool, April 4 2006. US Patent 7,024,348.

[49] Asli Celikyilmaz, Li Deng, and Dilek Hakkani-Tür. Deep learning in spoken and text-based dialog systems. In *Deep Learning in Natural Language Processing*, pages 49–78. Springer, 2018.

[50] Ken H Davis, R Biddulph, and Stephen Balashek. Automatic recognition of spoken digits. *The Journal of the Acoustical Society of America*, 24(6):637–642, 1952.

[51] Harry F Olson and Herbert Belar. Phonetic typewriter. *The Journal of the Acoustical Society of America*, 28(6):1072–1081, 1956.

[52] James W Forgie and Carma D Forgie. Results obtained from a vowel recognition computer program. *The Journal of the Acoustical Society of America*, 31(11):1480–1489, 1959.

[53] Taras K Vintsyuk. Speech discrimination by dynamic programming. *Cybernetics*, 4(1):52–57, 1968.

[54] Bishnu S Atal and Suzanne L Hanauer. Speech analysis and synthesis by linear prediction of the speech wave. *The journal of the acoustical society of America*, 50(2B):637–655, 1971.

[55] Lalit R Bahl, Frederick Jelinek, and Robert L Mercer. A maximum likelihood approach to continuous speech recognition. *IEEE transactions on pattern analysis and machine intelligence*, (2):179–190, 1983.

[56] Rohit Prabhavalkar, Takaaki Hori, Tara N. Sainath, Ralf Schlüter, and Shinji Watanabe. End-to-end speech recognition: A survey. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:325–351, 2024.

[57] James Baker. The dragon system–an overview. *IEEE Transactions on Acoustics, speech, and signal Processing*, 23(1):24–29, 1975.

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

[59] Stephen E Levinson, Lawrence R Rabiner, and M Mohan Sondhi. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. *Bell System Technical Journal*, 62(4):1035–1074, 1983.

[60] Matthew Nicholas Stuttle. *A Gaussian mixture model spectral representation for speech recognition*. PhD thesis, University of Cambridge, 2003.

[61] Yajie Miao, Mohammad Gowayyed, and Florian Metze. Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 167–174. IEEE, 2015.

[62] Jinyu Li, Li Deng, Reinhold Haeb-Umbach, and Yifan Gong. *Robust automatic speech recognition: a bridge to practical applications*. Academic Press, 2015.

[63] Stefan Kombrink, Tomas Mikolov, Martin Karafiát, and Lukás Burget. Recurrent neural network based language modeling in meeting recognition. In *Interspeech*, volume 11, pages 2877–2880, 2011.

[64] Joshua T Goodman. A bit of progress in language modeling. *Computer Speech & Language*, 15(4):403–434, 2001.

[65] Ying Zhang, Mohammad Pezeshki, Philémon Brakel, Saizheng Zhang, César Laurent, Yoshua Bengio, and Aaron Courville. Towards End-to-End Speech Recognition with Deep Convolutional Neural Networks. In *Proc. Interspeech 2016*, pages 410–414, 2016.

[66] Takaaki Hori, Shinji Watanabe, Yu Zhang, and William Chan. Advances in Joint CTC-Attention Based End-to-End Speech Recognition with a Deep CNN Encoder and RNN-LM. In *Proc. Interspeech 2017*, pages 949–953, 2017.

[67] Alex Graves and Navdeep Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International conference on machine learning*, pages 1764–1772, 2014.

[68] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[69] Andrew L. Maas, Awni Y. Hannun, Daniel Jurafsky, and Andrew Y. Ng. First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns. *CoRR*, abs/1408.2873, 2014.

[70] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. End-to-end continuous speech recognition using attention-based

recurrent nn: First results. In *NIPS 2014 Workshop on Deep Learning, December 2014*, 2014.

[71] George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on audio, speech, and language processing*, 20(1):30–42, 2011.

[72] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. 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.

[73] Vitaliy Liptchinsky, Gabriel Synnaeve, and Ronan Collobert. Letter-based speech recognition with gated convnets. *CoRR, vol. abs/1712.09444*, 1, 2017.

[74] 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. IEEE, 2015.

[75] Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.

[76] Volkmar Frinken, Francisco Zamora-Martinez, Salvador Espana-Boquera, Maria José Castro-Bleda, Andreas Fischer, and Horst Bunke. Long-short term memory neural networks language modeling for handwriting recognition. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 701–704. IEEE, 2012.

[77] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In *International conference on machine learning*, pages 2342–2350, 2015.

[78] Yu Zhang, Guoguo Chen, Dong Yu, Kaisheng Yaco, Sanjeev Khudanpur, and James Glass. Highway long short-term memory rnns for distant speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5755–5759. IEEE, 2016.

[79] Yuanyuan Zhao, Shuang Xu, and Bo Xu. Multidimensional residual learning based on recurrent neural networks for acoustic modeling. In *Interspeech*, pages 3419–3423, 2016.

[80] Jaeyoung Kim, Mostafa El-Khamy, and Jungwon Lee. Residual LSTM: Design of a Deep Recurrent Architecture for Distant Speech Recognition. In *Proc. Interspeech 2017*, pages 1591–1595, 2017.

[81] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[82] Jinyu Li, Changliang Liu, and Yifan Gong. Layer Trajectory LSTM. In *Proc. Interspeech 2018*, pages 1768–1772, 2018.

[83] Yajie Miao and Florian Metze. On speaker adaptation of long short-term memory recurrent neural networks. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[84] Tara N. Sainath and Bo Li. Modeling time-frequency patterns with lstm vs. convolutional architectures for lvcsr tasks. In *Interspeech*, 2016.

[85] Wei-Ning Hsu, Yu Zhang, and James Glass. A prioritized grid long short-term memory rnn for speech recognition. In *2016 IEEE Spoken Language Technology Workshop (SLT)*, pages 467–473. IEEE, 2016.

[86] Nal Kalchbrenner, Ivo Danihelka, and Alex Graves. Grid long short-term memory. *arXiv preprint arXiv:1507.01526*, 2015.

[87] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[88] Shiliang Zhang, Cong Liu, Hui Jiang, Si Wei, Lirong Dai, and Yu Hu. Nonrecurrent neural structure for long-term dependence. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(4):871–884, 2017.

[89] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.

[90] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. Bidirectional lstm networks for improved phoneme classification and recognition. In *International Conference on Artificial Neural Networks*, pages 799–804. Springer, 2005.

[91] Niko Moritz, Takaaki Hori, and Jonathan Le Roux. Unidirectional neural network architectures for end-to-end automatic speech recognition. In *INTERSPEECH*, pages 76–80, 2019.

[92] Kai Chen and Qiang Huo. Training deep bidirectional lstm acoustic model for lvcsr by a context-sensitive-chunk bptt approach. *IEEE/ACM*

Transactions on Audio, Speech, and Language Processing, 24(7):1185–1193, 2016.

[93] Albert Zeyer, Ralf Schlüter, and Hermann Ney. Towards online-recognition with deep bidirectional lstm acoustic models. In *Interspeech*, pages 3424–3428, 2016.

[94] Shaoshi Ling, Yuzong Liu, Julian Salazar, and Katrin Kirchhoff. Deep contextualized acoustic representations for semi-supervised speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6429–6433. IEEE, 2020.

[95] Jen-Tzung Chien and Alim Misbullah. Deep long short-term memory networks for speech recognition. In *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5. IEEE, 2016.

[96] Xiangang Li and Xihong Wu. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4520–4524. IEEE, 2015.

[97] William Chan and Ian Lane. Deep convolutional neural networks for acoustic modeling in low resource languages. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2056–2060. IEEE, 2015.

[98] Meng Cai and Jia Liu. Maxout neurons for deep convolutional and lstm neural networks in speech recognition. *Speech Communication*, 77:53–64, 2016.

[99] Honglak Lee, Peter Pham, Yan Largman, and Andrew Y Ng. Unsupervised

feature learning for audio classification using convolutional deep belief networks. In *Advances in neural information processing systems*, pages 1096–1104, 2009.

[100] Jason Li, Vitaly Lavrukhin, Boris Ginsburg, Ryan Leary, Oleksii Kuchaiev, Jonathan M. Cohen, Huyen Nguyen, and Ravi Teja Gadde. Jasper: An End-to-End Convolutional Neural Acoustic Model. In *Proc. Interspeech 2019*, pages 71–75, 2019.

[101] Awni Hannun, Ann Lee, Qiantong Xu, and Ronan Collobert. Sequence-to-Sequence Speech Recognition with Time-Depth Separable Convolutions. In *Proc. Interspeech 2019*, pages 3785–3789, 2019.

[102] Santiago Pascual, Mirco Ravanelli, Joan Serrà, Antonio Bonafonte, and Yoshua Bengio. Learning Problem-Agnostic Speech Representations from Multiple Self-Supervised Tasks. In *Proc. Interspeech 2019*, pages 161–165, 2019.

[103] Samuel Kriman, Stanislav Beliaev, Boris Ginsburg, Jocelyn Huang, Oleksii Kuchaiev, Vitaly Lavrukhin, Ryan Leary, Jason Li, and Yang Zhang. Quartznet: Deep automatic speech recognition with 1d time-channel separable convolutions. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6124–6128. IEEE, 2020.

[104] Wei Han, Zhengdong Zhang, Yu Zhang, Jiahui Yu, Chung-Cheng Chiu, James Qin, Anmol Gulati, Ruoming Pang, and Yonghui Wu. ContextNet: Improving Convolutional Neural Networks for Automatic Speech Recognition with Global Context. In *Proc. Interspeech 2020*, pages 3610–3614, 2020.

[105] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented se-

quence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006.

[106] William Song and Jim Cai. End-to-end deep neural network for automatic speech recognition. *Standford CS224D Reports*, 2015.

[107] Kartik Audhkhasi, Bhuvana Ramabhadran, George Saon, Michael Picheny, and David Nahamoo. Direct Acoustics-to-Word Models for English Conversational Speech Recognition. In *Proc. Interspeech 2017*, pages 959–963, 2017.

[108] I Goodfelow, Yoshua Bengio, and Aaron Courville. Deep learning (adaptive computation and machine learning series), 2016.

[109] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[110] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[111] Tessfu Geteye Fantaye, Junqing Yu, and Tulu Tilahun Hailu. Advanced convolutional neural network-based hybrid acoustic models for low-resource speech recognition. *Computers*, 9(2):36, 2020.

[112] Tara N Sainath, Brian Kingsbury, George Saon, Hagen Soltau, Abdelrahman Mohamed, George Dahl, and Bhuvana Ramabhadran. Deep convolutional neural networks for large-scale speech tasks. *Neural networks*, 64:39–48, 2015.

[113] D. Hau and K. Chen. Exploring hierarchical speech representations with a deep convolutional neural network. In *Proceedings of UKCI'11*, September 2011. United Kingdom Annual Workshop on Computational Intelligence (UKCI'11) ; Conference date: 07-09-2011 Through 09-09-2011.

[114] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[115] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[116] Florian Eyben, Martin Wöllmer, Björn Schuller, and Alex Graves. From speech to letters-using a novel neural network architecture for grapheme based asr. In *2009 IEEE Workshop on Automatic Speech Recognition & Understanding*, pages 376–380. IEEE, 2009.

[117] Dario Amodei, Rishita Anubhai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, et al. End to end speech recognition in english and mandarin. 2016.

[118] Awni Y Hannun, Andrew L Maas, Daniel Jurafsky, and Andrew Y Ng. First-pass large vocabulary continuous speech recognition using bi-directional recurrent dnns. abs/1408.2873, 2014.

[119] Navdeep Jaitly, Quoc V Le, Oriol Vinyals, Ilya Sutskever, David Sussillo, and Samy Bengio. An online sequence-to-sequence model using partial conditioning. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[120] Colin Raffel, Minh-Thang Luong, Peter J Liu, Ron J Weiss, and Douglas Eck. Online and linear-time attention by enforcing monotonic alignments. In *International conference on machine learning*, pages 2837–2846. PMLR, 2017.

[121] Chung-Cheng Chiu* and Colin Raffel*. Monotonic chunkwise attention. In *International Conference on Learning Representations*, 2018.

[122] Kwangyoun Kim, Kyungmin Lee, Dhananjaya Gowda, Junmo Park, Sungsoo Kim, Sichen Jin, Young-Yoon Lee, Jinsu Yeo, Daehyun Kim, Seokyeong Jung, et al. Attention based on-device streaming speech recognition with large speech corpus. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 956–963. IEEE, 2019.

[123] Shinji Watanabe, Takaaki Hori, Suyoun Kim, John R Hershey, and Tomoki Hayashi. Hybrid ctc/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253, 2017.

[124] Suyoun Kim, Takaaki Hori, and Shinji Watanabe. Joint ctc-attention based end-to-end speech recognition using multi-task learning. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4835–4839. IEEE, 2017.

[125] Haoran Miao, Gaofeng Cheng, Pengyuan Zhang, and Yonghong Yan. Online hybrid ctc/attention end-to-end automatic speech recognition architecture. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1452–1465, 2020.

[126] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio. End-to-end attention-based large vocabulary

speech recognition. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 4945–4949. IEEE, 2016.

[127] Liang Lu, Xingxing Zhang, and Steve Renais. On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5060–5064. IEEE, 2016.

[128] Anjuli Kannan, Yonghui Wu, Patrick Nguyen, Tara N Sainath, Zhijeng Chen, and Rohit Prabhavalkar. An analysis of incorporating an external language model into a sequence-to-sequence model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5828. IEEE, 2018.

[129] Wei Zhou, Ralf Schlüter, and Hermann Ney. Robust Beam Search for Encoder-Decoder Attention Based Speech Recognition Without Length Bias. In *Proc. Interspeech 2020*, pages 1768–1772, 2020.

[130] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[131] Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A STRUCTURED SELF-ATTENTIVE SENTENCE EMBEDDING. In *International Conference on Learning Representations*, 2017.

[132] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[133] Kazuki Irie. *Advancing Neural Language Modeling in Automatic Speech Recognition*. PhD thesis, RWTH Aachen University, 2020.

[134] Peter J. Liu*, Mohammad Saleh*, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. In *International Conference on Learning Representations*, 2018.

[135] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3159–3166, 2019.

[136] Matthias Sperber, Jan Niehues, Graham Neubig, Sebastian Stüker, and Alex Waibel. Self-Attentional Acoustic Models. In *Proc. Interspeech 2018*, pages 3723–3727, 2018.

[137] Yongqiang Wang, Abdelrahman Mohamed, Due Le, Chunxi Liu, Alex Xiao, Jay Mahadeokar, Hongzhao Huang, Andros Tjandra, Xiaohui Zhang, Frank Zhang, et al. Transformer-based acoustic modeling for hybrid speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6874–6878. IEEE, 2020.

[138] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[139] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech 2020*, pages 5036–5040, 2020.

[140] Zhanghao Wu*, Zhijian Liu*, Ji Lin, Yujun Lin, and Song Han. Lite transformer with long-short range attention. In *International Conference on Learning Representations*, 2020.

[141] Oleksii Hrinchuk, Mariya Popova, and Boris Ginsburg. Correction of automatic speech recognition with transformer sequence-to-sequence model. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7074–7078. IEEE, 2020.

[142] Yu-An Chung, Yu Zhang, Wei Han, Chung-Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu. W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 244–250. IEEE, 2021.

[143] Anshuman Tripathi, Hasim Sak, Han Lu, Qian Zhang, and JaeYoung Kim. Transformer transducer: One model unifying streaming and non-streaming speech recognition, April 7 2022. US Patent App. 17/210,465.

[144] Xi Chen, Songyang Zhang, Dandan Song, Peng Ouyang, and Shouyi Yin. Transformer with Bidirectional Decoder for Speech Recognition. In *Proc. Interspeech 2020*, pages 1773–1777, 2020.

[145] Abdelrahman Mohamed, Dmytro Okhonko, and Luke Zettlemoyer. Trans-

formers with convolutional context for ASR. *CoRR*, abs/1904.11660, 2019.

[146] Andy T Liu, Shang-Wen Li, and Hung-yi Lee. Tera: Self-supervised learning of transformer encoder representation for speech. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2351–2366, 2021.

[147] Dongwei Jiang, Wubo Li, Miao Cao, Wei Zou, and Xiangang Li. Speech SimCLR: Combining Contrastive and Reconstruction Objective for Self-Supervised Speech Representation Learning. In *Proc. Interspeech 2021*, pages 1544–1548, 2021.

[148] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, et al. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

[149] Chung-Cheng Chiu, James Qin, Yu Zhang, Jiahui Yu, and Yonghui Wu. Self-supervised learning with random-projection quantizer for speech recognition. In *International Conference on Machine Learning*, pages 3915–3924. PMLR, 2022.

[150] Kavan Fatehi, Mercedes Torres Torres, and Ayse Kucukyilmaz. ScoutWav: Two-Step Fine-Tuning on Self-Supervised Automatic Speech Recognition for Low-Resource Environments. In *Proc. Interspeech 2022*, pages 3523–3527, 2022.

[151] Kavan Fatehi and Ayse Kucukyilmaz. LABERT: A Combination of Local Aggregation and Self-Supervised Speech Representation Learning for Detecting Informative Hidden Units in Low-Resource ASR Systems. In *Proc. INTERSPEECH 2023*, pages 211–215, 2023.

[152] Chengxu Zhuang, Alex Lin Zhai, and Daniel Yamins. Local aggregation for unsupervised learning of visual embeddings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6002–6012, 2019.

[153] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[154] Herve A Bourlard and Nelson Morgan. *Connectionist speech recognition: a hybrid approach*, volume 247. Springer Science & Business Media, 1994.

[155] Thomas Kemp and Alex Waibel. Unsupervised training of a speech recognizer: Recent experiments. In *in Proc. EUROSPEECH*, 1999.

[156] Geoffrey E Hinton. Learning multiple layers of representation. *Trends in cognitive sciences*, 11(10):428–434, 2007.

[157] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[158] Michael I Jordan and Tom M Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[159] Robert Gray. Vector quantization. *IEEE Assp Magazine*, 1(2):4–29, 1984.

[160] Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[161] Geoffrey E Hinton and Richard Zemel. Autoencoders, minimum description length and helmholtz free energy. *Advances in neural information processing systems*, 6, 1993.

[162] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[163] Abdelrahman Mohamed, Hung-yi Lee, Lasse Borgholt, Jakob D Havtorn, Joakim Edin, Christian Igel, Katrin Kirchhoff, Shang-Wen Li, Karen Livescu, Lars Maaløe, et al. Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 2022.

[164] J Wilpon and L Rabiner. A modified k-means clustering algorithm for use in isolated work recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 33(3):587–594, 1985.

[165] J-L Gauvain and Chin-Hui Lee. Maximum a posteriori estimation for multivariate gaussian mixture observations of markov chains. *IEEE transactions on speech and audio processing*, 2(2):291–298, 1994.

[166] Steve J Young and Philip C Woodland. State clustering in hidden markov model-based continuous speech recognition. *Computer Speech & Language*, 8(4):369–383, 1994.

[167] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

[168] Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of machine learning research*, 13(2), 2012.

[169] Garimella SVS Sivaram, Sridhar Krishna Nemala, Mounya Elhilali, Trac D Tran, and Hynek Hermansky. Sparse coding for speech recognition. In

2010 IEEE International Conference on Acoustics, Speech and Signal Processing, pages 4346–4349. IEEE, 2010.

[170] Marc'Aurelio Ranzato, Y-Lan Boureau, Sumit Chopra, and Yann LeCun. A unified energy-based framework for unsupervised learning. In *Artificial Intelligence and Statistics*, pages 371–379. PMLR, 2007.

[171] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 649–666. Springer, 2016.

[172] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European conference on computer vision (ECCV)*, pages 132–149, 2018.

[173] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015.

[174] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[175] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*,

pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[176] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

[177] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[178] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[179] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[180] Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass. An unsupervised autoregressive model for speech representation learning. *arXiv preprint arXiv:1904.03240*, 2019.

[181] Yu-An Chung and James Glass. Generative pre-training for speech with autoregressive predictive coding. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3497–3501. IEEE, 2020.

[182] Douglas O'Shaughnessy. Linear predictive coding. *IEEE potentials*, 7(1):29–32, 1988.

[183] Yu-An Chung and James R. Glass. Improved speech representations with multi-target autoregressive predictive coding. *ArXiv*, abs/2004.05274, 2020.

[184] Yu-An Chung, Hao Tang, and James Glass. Vector-Quantized Autoregressive Predictive Coding. In *Proc. Interspeech 2020*, pages 3760–3764, 2020.

[185] Xianghu Yue and Haizhou Li. Phonetically Motivated Self-Supervised Speech Representation Learning. In *Proc. Interspeech 2021*, pages 746–750, 2021.

[186] Weiran Wang, Qingming Tang, and Karen Livescu. Unsupervised pre-training of bidirectional speech encoders via masked reconstruction. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6889–6893. IEEE, 2020.

[187] Alexander H. Liu, Yu-An Chung, and James R. Glass. Non-autoregressive predictive coding for learning speech representations from local dependencies. In *Interspeech*, 2020.

[188] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[189] Xingchen Song, Guangsen Wang, Yiheng Huang, Zhiyong Wu, Dan Su, and Helen Meng. Speech-XLNet: Unsupervised Acoustic Model Pretraining for Self-Attention Networks. In *Proc. Interspeech 2020*, pages 3765–3769, 2020.

[190] Shaoshi Ling and Yuzong Liu. Decoar 2.0: Deep contextualized acoustic representations with vector quantization. *ArXiv*, abs/2012.06659, 2020.

[191] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli.

wav2vec: Unsupervised Pre-Training for Speech Recognition. In *Proc. Interspeech 2019*, pages 3465–3469, 2019.

[192] Samik Sadhu, Di He, Che-Wei Huang, Sri Harish Mallidi, Minhua Wu, Ariya Rastrow, Andreas Stolcke, Jasha Droppo, and Roland Maas. wav2vec-C: A Self-Supervised Model for Speech Representation Learning. In *Proc. Interspeech 2021*, pages 711–715, 2021.

[193] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

[194] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[195] Alexei Baevski, Steffen Schneider, and Michael Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations*, 2020.

[196] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdel rahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[197] Jing Zhao and Wei-Qiang Zhang. Improving automatic speech recognition performance for low-resource languages with self-supervised mod-

els. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1227–1241, 2022.

[198] Emily Prud'hommeaux, Robbie Jimerson, Richard Hatcher, and Karin Michelson. Automatic speech recognition for supporting endangered language documentation. 2021.

[199] Shreya Khare, Ashish R Mittal, Anuj Diwan, Sunita Sarawagi, Preethi Jyothi, and Samarth Bharadwaj. Low resource asr: The surprising effectiveness of high resource transliteration. In *Interspeech*, pages 1529–1533, 2021.

[200] Kaushal Bhogale, Abhigyan Raman, Tahir Javed, Sumanth Doddapaneni, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh M Khapra. Effectiveness of mining audio and text pairs from public data for improving asr systems for low-resource languages. In *Icassp 2023-2023 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 1–5. IEEE, 2023.

[201] Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. *arXiv preprint arXiv:1809.01431*, 2018.

[202] Yajie Miao, Florian Metze, and Shourabh Rawat. Deep maxout networks for low-resource speech recognition. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 398–403. IEEE, 2013.

[203] Jui-Yang Hsu, Yuan-Jui Chen, and Hung-yi Lee. Meta learning for end-to-end low-resource speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7844–7848. IEEE, 2020.

[204] Cheng Yi, Jianzhong Wang, Ning Cheng, Shiyu Zhou, and Bo Xu. Applying wav2vec2. 0 to speech recognition in various low-resource languages. *arXiv preprint arXiv:2012.12121*, 2020.

[205] Chenpeng Du and Kai Yu. Speaker augmentation for low resource speech recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7719–7723, 2020.

[206] Zi-Qiang Zhang, Yan Song, Ming-Hui Wu, Xin Fang, Ian McLoughlin, and Li-Rong Dai. Cross-lingual self-training to learn multilingual representation for low-resource speech recognition. *Circuits, Systems, and Signal Processing*, 41(12):6827–6843, 2022.

[207] Moussa Doumbouya, Lisa Einstein, and Chris Piech. Using radio archives for low-resource speech recognition: towards an intelligent virtual assistant for illiterate users. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14757–14765, 2021.

[208] Yanmin Qian and Zhikai Zhou. Optimizing data usage for low-resource speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:394–403, 2022.

[209] Linghui Meng, Jin Xu, Xu Tan, Jindong Wang, Tao Qin, and Bo Xu. Mixspeech: Data augmentation for low-resource automatic speech recognition. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7008–7012. IEEE, 2021.

[210] Yaqi Chen, Hao Zhang, Xukui Yang, Wenlin Zhang, and Dan Qu. Improving cross-lingual low-resource speech recognition by task-based meta polyloss. *Computer Speech & Language*, page 101648, 2024.

[211] Muhammad Umar Farooq, Rehan Ahmad, and Thomas Hain. Must: A multilingual student-teacher learning approach for low-resource speech recognition. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–6. IEEE, 2023.

[212] Varun Krishna, Tarun Sai, and Sriram Ganapathy. Representation learning with hidden unit clustering for low resource speech applications. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:1036–1047, 2024.

[213] Anthony Rousseau, Paul Deléglise, and Yannick Esteve. Ted-lium: an automatic speech recognition dedicated corpus. In *LREC*, pages 125–129, 2012.

[214] 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. In *International Conference on Language Resources and Evaluation*, 2019.

[215] Frank Rudzicz, Aravind Kumar Namasivayam, and Talya Wolff. The torgo database of acoustic and articulatory speech from speakers with dysarthria. *Language Resources and Evaluation*, 46(4):523–541, 2012.

[216] Xavier Menendez-Pidal, James B Polikoff, Shirley M Peters, Jennie E Leonzio, and H Timothy Bunnell. The nemours database of dysarthric speech. In *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*, volume 3, pages 1962–1965. IEEE, 1996.

[217] Heejin Kim, Mark Hasegawa-Johnson, Adrienne Perlman, Jon Gunderson, Thomas S Huang, Kenneth Watkin, and Simone Frame. Dysarthric

speech database for universal access research. In *Ninth Annual Conference of the International Speech Communication Association*, 2008.

[218] Jan Chorowski, Ron J Weiss, Samy Bengio, and Aäron van den Oord. Unsupervised speech representation learning using wavenet autoencoders. *IEEE/ACM transactions on audio, speech, and language processing*, 27(12):2041–2053, 2019.

[219] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[220] Ekapol Chuangsuwanich. Multilingual techniques for low resource automatic speech recognition. Technical report, Massachusetts Institute of Technology Cambridge United States, 2016.

[221] Alexei Baevski, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. Unsupervised speech recognition. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[222] Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. Layer-wise analysis of a self-supervised speech representation model. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 914–921, 2021.

[223] Wei-Ning Hsu, Anuroop Sriram, Alexei Baevski, Tatiana Likhomanenko, Qiantong Xu, Vineel Pratap, Jacob Kahn, Ann Lee, Ronan Collobert, Gabriel Synnaeve, et al. Robust wav2vec 2.0: Analyzing domain shift in self-supervised pre-training. *arXiv preprint arXiv:2104.01027*, 2021.

[224] Chengyi Wang, Yu Wu, Shujie Liu, Jinyu Li, Liang Lu, Guoli Ye, and Ming Zhou. Low latency end-to-end streaming speech recognition with a scout network. *arXiv preprint arXiv:2003.10369*, 2020.

[225] Harold Hotelling. Relations between two sets of variates. In *Breakthroughs in statistics*, pages 162–190. Springer, 1992.

[226] Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. Layer-wise analysis of a self-supervised speech representation model. *arXiv preprint arXiv:2107.04734*, 2021.

[227] Xinghao Yang, Weifeng Liu, Wei Liu, and Dacheng Tao. A survey on canonical correlation analysis. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2349–2368, 2019.

[228] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[229] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*, 2018.

[230] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech*, volume 2017, pages 498–502, 2017.

[231] Niko Moritz, Takaaki Hori, and Jonathan Le. Streaming automatic speech recognition with the transformer model. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6074–6078. IEEE, 2020.

[232] Chengyi Wang, Yu Wu, Yujiao Du, Jinyu Li, Shujie Liu, Liang Lu, Shuo Ren, Guoli Ye, Sheng Zhao, and Ming Zhou. Semantic mask for transformer based end-to-end speech recognition. *arXiv preprint arXiv:1912.03010*, 2019.

[233] Alexei Baevski, Michael Auli, and Abdelrahman Mohamed. Effectiveness of self-supervised pre-training for speech recognition. *arXiv preprint arXiv:1911.03912*, 2019.

[234] Sreyan Ghosh, Ashish Seth, and Srinivasan Umesh. Delores: Decorrelating latent spaces for low-resource audio representation learning. *arXiv preprint arXiv:2203.13628*, 2022.

[235] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[236] Xiusong Sun, Bo Wang, Shaohan Liu, Tingxiang Lu, Xin Shan, and Qun Yang. Lmc-smca: A new active learning method in asr. *IEEE Access*, 9:37011–37021, 2021.

[237] Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, Tie-Yan Liu, et al. R-drop: Regularized dropout for neural networks. *Advances in Neural Information Processing Systems*, 34:10890–10905, 2021.

[238] Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4218–4222, Marseille, France, May 2020. European Language Resources Association.

[239] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[240] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[241] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 66–75, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[242] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number CONF. IEEE Signal Processing Society, 2011.

[243] Chanwoo Kim, Dhananjaya Gowda, Dongsoo Lee, Jiyeon Kim, Ankur Kumar, Sungsoo Kim, Abhinav Garg, and Changwoo Han. A review of on-device fully neural end-to-end automatic speech recognition algorithms. In *2020 54th Asilomar Conference on Signals, Systems, and Computers*, pages 277–283. IEEE, 2020.

[244] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[245] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere.

In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

[246] David T Hoffmann, Nadine Behrmann, Juergen Gall, Thomas Brox, and Mehdi Noroozi. Ranking info noise contrastive estimation: Boosting contrastive learning via ranked positives. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 897–905, 2022.

[247] Tri Huynh, Simon Kornblith, Matthew R Walter, Michael Maire, and Maryam Khademi. Boosting contrastive self-supervised learning with false negative cancellation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 2785–2795, 2022.

[248] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. *Advances in Neural Information Processing Systems*, 33:21798–21809, 2020.

[249] Yanzhao Zhang, Richong Zhang, Samuel Mensah, Xudong Liu, and Yongyi Mao. Unsupervised sentence representation via contrastive learning with mixing negatives. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11730–11738, 2022.

[250] Junwen Bai, Bo Li, Yu Zhang, Ankur Bapna, Nikhil Siddhartha, Khe Chai Sim, and Tara N Sainath. Joint unsupervised and supervised training for multilingual asr. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6402–6406. IEEE, 2022.

[251] Kanishka Rao, Haşim Sak, and Rohit Prabhavalkar. Exploring architectures, data and units for streaming end-to-end speech recognition with rnn-transducer. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 193–199. IEEE, 2017.

[252] Han Zhu, Li Wang, Gaofeng Cheng, Jindong Wang, Pengyuan Zhang, and Yonghong Yan. Wav2vec-S: Semi-Supervised Pre-Training for Low-Resource ASR. In *Proc. Interspeech 2022*, pages 4870–4874, 2022.

[253] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5147–5156, 2016.

[254] Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. Contrastive clustering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 8547–8555, 2021.

[255] Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022.

Appendix A

Complete Results

Figure A.1 and Figure A.2 show the results obtained from the pre-training different models based on the different number of layers and percentage of the data on WSJ and Librispeech datasets, respectively. Furthermore, the results of the pre-training and fine-tuning on ICUBE data are shown in Figure A.3 for WSJ and Figure A.4 for Librispeech. In addition, the full results after training and testing on WSJ and Librispeech are presented in Figure A.5 and Figure A.6, respectively.

	10%				20%				30%				40%				50%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	44.68	44.89	45.23	45.58	42.57	42.69	43.39	43.72	40.18	40.32	41.18	42.38	37.61	37.45	37.62	38.69	35.18	34.89	34.21	36.11
ResLSTM	44.84	44.97	45.68	45.73	42.93	43.18	43.62	43.92	40.78	40.91	41.34	42.57	38.12	38.19	37.57	39.12	35.63	35.21	33.87	37.91
ltLSTM	44.73	44.93	45.39	45.69	42.68	42.51	43.53	43.83	39.91	39.87	40.91	42.21	37.22	37.15	37.29	38.52	34.81	34.51	33.62	35.87
cltLSTM	44.65	44.76	45.28	45.48	41.93	41.86	43.27	43.68	39.68	39.37	39.17	41.88	37.06	36.92	36.73	37.86	34.27	34.19	33.41	34.92
FNN+LSTM	50.39	-	-	-	49.13	-	-	-	45.48	-	-	-	42.61	-	-	-	40.13	-	-	-
LSTM+FNN+LSTM	50.73	-	-	-	50.21	-	-	-	46.73	-	-	-	43.67	-	-	-	41.76	-	-	-
LSTM+FNN+FNN	50.47	-	-	-	49.89	-	-	-	46.91	-	-	-	43.39	-	-	-	41.23	-	-	-
BLSTM	44.58	-	-	-	41.98	-	-	-	40.19	-	-	-	37.28	-	-	-	34.93	-	-	-
1-D CNN+BLSTM	45.12	-	-	-	42.39	-	-	-	40.51	-	-	-	37.59	-	-	-	35.42	-	-	-
1-D CNN+LSTM	48.17	-	-	-	46.21	-	-	-	43.69	-	-	-	41.58	-	-	-	39.27	-	-	-
1-D CNN+GRU	48.81	-	-	-	47.21	-	-	-	45.31	-	-	-	42.43	-	-	-	39.81	-	-	-
QuartzNet		44.11			41.53				38.32				35.79				31.72			
Transformer		42.19			40.87				37.92				35.32				31.49			

	60%				70%				80%				90%				100%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	34.12	33.21	32.78	34.28	32.48	32.12	30.67	33.65	29.75	28.73	27.49	31.19	28.43	27.13	25.31	30.27	27.29	26.72	24.17	29.23
ResLSTM	34.68	33.89	32.15	33.92	32.78	32.29	30.18	31.71	29.97	29.27	27.13	30.39	28.81	27.43	24.91	29.52	27.58	26.98	24.14	28.86
ltLSTM	33.47	32.95	31.87	33.17	31.62	31.47	28.73	31.22	29.12	28.47	26.42	29.98	28.07	26.81	24.68	28.75	27.18	25.13	24.08	28.14
cltLSTM	33.12	32.78	31.69	32.98	31.46	31.22	30.42	30.98	28.69	28.19	28.03	29.74	27.83	26.53	26.13	29.13	26.93	26.23	26.18	28.86
FNN+LSTM	38.79	-	-	-	35.41	-	-	-	33.29	-	-	-	31.63	-	-	-	29.34	-	-	-
LSTM+FNN+LSTM	40.08	-	-	-	38.19	-	-	-	35.83	-	-	-	33.12	-	-	-	30.39	-	-	-
LSTM+FNN+FNN	39.72	-	-	-	37.21	-	-	-	34.37	-	-	-	32.59	-	-	-	29.18	-	-	-
BLSTM	33.96	-	-	-	30.38	-	-	-	29.13	-	-	-	28.07	-	-	-	27.18	-	-	-
1-D CNN+BLSTM	34.21	-	-	-	30.82	-	-	-	29.89	-	-	-	28.69	-	-	-	27.63	-	-	-
1-D CNN+LSTM	36.21	-	-	-	34.87	-	-	-	33.47	-	-	-	30.19	-	-	-	28.78	-	-	-
1-D CNN+GRU	36.32	-	-	-	33.51	-	-	-	31.87	-	-	-	29.64	-	-	-	28.63	-	-	-
QuartzNet	29.93				27.98				25.65				23.51				22.13			
Transformer	29.67				27.43				25.29				23.19				21.27			

Figure A.1: Pre-train different models on WSJ and test on ICUBE

	10%				20%				30%				40%				50%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	44.81	45.31	45.79	45.93	42.37	42.83	43.22	44.87	40.59	40.19	40.38	44.21	37.17	36.91	36.28	38.16	34.12	33.86	32.61	36.43
ResLSTM	44.93	45.59	45.91	46.11	42.61	43.48	43.87	44.93	41.18	41.29	40.92	41.59	37.89	37.51	37.42	37.75	34.58	34.12	33.65	36.18
ltLSTM	44.87	45.43	45.83	45.97	42.21	42.62	43.51	44.93	40.13	39.87	39.41	41.89	36.82	36.47	35.94	37.79	33.67	33.27	32.35	36.55
cltLSTM	44.79	44.83	45.69	45.91	42.18	42.38	43.19	44.71	39.95	39.71	39.27	41.28	36.29	36.28	35.67	37.22	33.48	32.98	32.14	36.19
FNN+LSTM	51.38	-	-	-	49.71	-	-	-	46.13	-	-	-	43.19	-	-	-	41.58	-	-	-
LSTM+FNN+LSTM	51.69	-	-	-	50.37	-	-	-	47.62	-	-	-	44.51	-	-	-	41.85	-	-	-
LSTM+FNN+FNN	51.53	-	-	-	50.18	-	-	-	47.27	-	-	-	44.23	-	-	-	41.31	-	-	-
BLSTM	44.74	-	-	-	42.21	-	-	-	40.28	-	-	-	36.93	-	-	-	33.81	-	-	-
1-D CNN+BLSTM	44.92	-	-	-	42.53	-	-	-	40.63	-	-	-	37.61	-	-	-	34.28	-	-	-
1-D CNN+LSTM	47.38	-	-	-	45.61	-	-	-	43.12	-	-	-	41.17	-	-	-	38.62	-	-	-
1-D CNN+GRU	47.69	-	-	-	46.13	-	-	-	44.33	-	-	-	41.89	-	-	-	39.58	-	-	-
QuartzNet		44.27			42.18				38.85				35.39				31.98			
Transformer		43.78			41.28				38.31				34.97				31.52			

	60%				70%				80%				90%				100%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	31.63	31.29	31.48	34.62	30.22	29.81	29.12	32.51	28.67	27.69	27.86	30.71	27.43	26.11	25.53	28.17	26.13	25.81	24.28	27.92
ResLSTM	32.41	32.28	32.71	33.61	30.45	30.12	29.83	31.48	29.15	28.61	28.29	29.31	27.68	27.31	26.94	28.42	26.28	26.12	25.13	27.88
ltLSTM	31.28	30.92	30.52	34.61	29.72	29.67	28.97	33.83	28.43	27.41	27.18	31.22	27.19	25.29	25.09	29.48	26.11	24.91	24.11	27.63
cltLSTM	30.97	30.51	30.27	33.86	29.48	29.18	28.59	32.46	28.21	27.29	26.76	30.92	26.73	25.07	24.71	28.75	25.98	24.73	23.91	26.33
FNN+LSTM	39.75	-	-	-	37.13	-	-	-	35.81	-	-	-	34.69	-	-	-	33.97	-	-	-
LSTM+FNN+LSTM	40.19	-	-	-	38.61	-	-	-	37.61	-	-	-	35.82	-	-	-	34.88	-	-	-
LSTM+FNN+FNN	39.87	-	-	-	37.63	-	-	-	35.47	-	-	-	34.81	-	-	-	33.92	-	-	-
BLSTM	31.12	-	-	-	29.89	-	-	-	28.31	-	-	-	26.83	-	-	-	26.07	-	-	-
1-D CNN+BLSTM	31.57	-	-	-	30.31	-	-	-	28.91	-	-	-	27.12	-	-	-	26.19	-	-	-
1-D CNN+LSTM	35.58	-	-	-	33.62	-	-	-	31.92	-	-	-	29.71	-	-	-	27.22	-	-	-
1-D CNN+GRU	36.71	-	-	-	33.89	-	-	-	31.29	-	-	-	29.48	-	-	-	27.13	-	-	-
QuartzNet	30.11				28.11				25.98				23.89				22.85			
Transformer	29.92				27.71				25.61				23.46				22.32			

Figure A.2: pre-train different models on Librispeech and test on ICUBE

	10%				20%				30%				40%				50%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	44.77	45.27	45.73	45.88	42.31	42.78	43.17	44.81	40.53	40.13	40.31	44.17	37.13	36.86	36.21	38.12	34.06	33.79	32.55	36.35
ResLSTM	44.89	45.51	45.83	45.94	42.57	43.42	43.75	44.81	41.12	41.19	40.81	41.47	37.81	37.42	37.21	37.63	34.49	34.04	33.42	36.07
ltLSTM	44.81	45.37	45.76	45.89	42.18	42.56	43.46	44.83	40.07	39.76	39.34	41.78	36.73	36.38	35.83	37.71	33.59	33.18	32.27	36.47
cltLSTM	44.71	44.76	45.62	45.83	42.11	42.32	43.11	44.62	39.87	39.64	39.15	41.21	36.18	36.21	35.58	37.19	33.39	32.87	31.97	36.11
FNN+LSTM	51.29	-	-	-	49.62	-	-	-	45.97	-	-	-	43.08	-	-	-	41.47	-	-	-
LSTM+FNN+LSTM	51.61	-	-	-	50.28	-	-	-	47.53	-	-	-	44.42	-	-	-	41.73	-	-	-
LSTM+FNN+FNN	51.46	-	-	-	50.11	-	-	-	47.18	-	-	-	44.11	-	-	-	41.17	-	-	-
BLSTM	44.71	-	-	-	42.17	-	-	-	40.21	-	-	-	36.82	-	-	-	33.69	-	-	-
1-D CNN+BLSTM	44.87	-	-	-	42.48	-	-	-	40.53	-	-	-	37.49	-	-	-	34.17	-	-	-
1-D CNN+LSTM	47.31	-	-	-	45.55	-	-	-	43.05	-	-	-	41.08	-	-	-	38.51	-	-	-
1-D CNN+GRU	47.62	-	-	-	46.06	-	-	-	44.24	-	-	-	41.78	-	-	-	39.48	-	-	-
QuartzNet		44.06			42.01				38.79				35.3				31.91			
Transformer		43.69			41.15				38.17				34.77				31.44			

	60%				70%				80%				90%				100%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	31.56	31.21	31.41	34.51	30.17	29.75	28.95	32.42	28.61	27.62	27.74	30.63	27.38	26.03	25.44	28.04	26.08	25.73	24.17	27.81
ResLSTM	32.33	32.18	32.49	33.47	34.36	29.98	29.66	31.24	29.06	28.49	27.81	28.79	27.59	27.19	26.18	27.91	26.19	25.98	24.11	27.29
ltLSTM	31.19	30.87	30.38	34.48	29.61	29.59	28.83	33.71	28.31	27.32	27.03	31.12	27.11	25.17	24.93	29.36	26.03	24.71	23.87	27.32
cltLSTM	30.86	30.43	30.08	33.54	29.37	29.08	28.44	32.29	28.19	27.16	26.59	30.74	26.64	24.91	24.58	28.62	25.91	24.61	23.67	26.24
FNN+LSTM	39.58	-	-	-	36.88	-	-	-	35.69	-	-	-	34.57	-	-	-	33.84	-	-	-
LSTM+FNN+LSTM	40.08	-	-	-	38.49	-	-	-	37.48	-	-	-	35.71	-	-	-	33.91	-	-	-
LSTM+FNN+FNN	39.73	-	-	-	37.51	-	-	-	35.34	-	-	-	34.42	-	-	-	33.74	-	-	-
BLSTM	31.01	-	-	-	29.78	-	-	-	28.15	-	-	-	26.65	-	-	-	25.87	-	-	-
1-D CNN+BLSTM	31.46	-	-	-	30.23	-	-	-	28.83	-	-	-	26.98	-	-	-	26.08	-	-	-
1-D CNN+LSTM	35.47	-	-	-	33.52	-	-	-	31.81	-	-	-	29.57	-	-	-	27.13	-	-	-
1-D CNN+GRU	36.62	-	-	-	33.78	-	-	-	31.19	-	-	-	29.12	-	-	-	26.98	-	-	-
QuartzNet	30.04				28.03				25.91				23.81				22.78			
Transformer	29.81				27.57				25.38				23.21				22.01			

Figure A.3: Pre-train different models on WSJ, Testing and Fine-tuning on ICUBE

	10%				20%				30%				40%				50%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	44.62	44.81	45.16	45.48	42.51	42.61	43.31	43.64	40.11	40.24	41.12	42.29	37.54	37.36	37.54	38.59	35.08	34.81	34.12	36.02
ResLSTM	44.78	44.89	45.59	45.65	42.86	43.11	43.54	43.82	40.71	40.82	41.26	42.48	37.98	38.11	37.48	38.99	35.52	35.09	33.75	37.79
ltLSTM	44.68	44.84	45.31	45.61	42.58	42.43	43.46	43.75	39.78	39.81	40.82	42.13	37.16	37.07	37.18	38.43	34.69	34.39	33.51	35.72
cltLSTM	44.57	44.68	45.18	45.41	41.84	41.74	43.21	43.62	39.57	39.26	39.11	41.79	36.96	36.81	36.62	37.76	34.17	34.09	32.33	34.79
FNN+LSTM	50.31	-	-	-	49.03	-	-	-	45.38	-	-	-	42.53	-	-	-	39.94	-	-	-
LSTM+FNN+LSTM	50.67	-	-	-	50.14	-	-	-	46.64	-	-	-	43.53	-	-	-	41.64	-	-	-
LSTM+FNN+FNN	50.39	-	-	-	49.78	-	-	-	46.79	-	-	-	43.27	-	-	-	41.12	-	-	-
BLSTM	44.51	-	-	-	41.87	-	-	-	40.04	-	-	-	37.07	-	-	-	34.75	-	-	-
1-D CNN+BLSTM	45.06	-	-	-	42.32	-	-	-	40.41	-	-	-	37.47	-	-	-	35.29	-	-	-
1-D CNN+LSTM	48.11	-	-	-	46.15	-	-	-	43.59	-	-	-	41.46	-	-	-	39.19	-	-	-
1-D CNN+GRU	48.75	-	-	-	47.17	-	-	-	45.22	-	-	-	42.31	-	-	-	39.73	-	-	-
QuartzNet		43.93			41.39				38.25				35.69				31.64			
Transformer		42.11			40.63				37.78				35.15				31.38			

	60%				70%				80%				90%				100%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	34.03	33.12	32.68	34.21	32.31	32.02	30.55	33.54	29.65	28.61	27.37	31.11	28.33	26.98	25.19	30.18	27.18	26.58	23.94	29.14
ResLSTM	34.58	33.78	31.98	33.81	32.69	32.17	30.05	31.59	29.86	29.08	26.97	30.22	28.68	27.31	24.79	29.41	27.42	26.75	23.98	27.73
ltLSTM	33.36	32.82	31.73	33.02	31.51	31.34	28.62	31.11	29.03	28.38	26.29	29.72	27.98	26.69	24.51	28.62	27.12	24.89	23.63	27.51
cltLSTM	32.98	32.66	31.54	32.83	31.32	31.14	30.29	30.81	28.54	28.06	27.89	29.65	27.71	26.39	25.49	28.88	26.83	26.15	25.83	27.76
FNN+LSTM	38.65	-	-	-	35.28	-	-	-	33.18	-	-	-	31.51	-	-	-	29.25	-	-	-
LSTM+FNN+LSTM	39.91	-	-	-	38.04	-	-	-	35.71	-	-	-	32.97	-	-	-	30.27	-	-	-
LSTM+FNN+FNN	39.61	-	-	-	37.12	-	-	-	34.26	-	-	-	32.48	-	-	-	28.89	-	-	-
BLSTM	33.67	-	-	-	30.23	-	-	-	28.91	-	-	-	27.87	-	-	-	26.71	-	-	-
1-D CNN+BLSTM	34.14	-	-	-	30.73	-	-	-	29.74	-	-	-	28.57	-	-	-	27.51	-	-	-
1-D CNN+LSTM	36.14	-	-	-	34.73	-	-	-	33.32	-	-	-	30.07	-	-	-	28.61	-	-	-
1-D CNN+GRU	36.19	-	-	-	33.38	-	-	-	31.68	-	-	-	29.49	-	-	-	28.52	-	-	-
QuartzNet	29.86				27.91				25.57				23.47				22.08			
Transformer	29.53				27.21				25.03				22.91				20.87			

Figure A.4: Pre-train different models on Librispeech, Testing and Fine-tuning on ICUBE

	10%				20%				30%				40%				50%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	45.83	45.62	45.31	45.49	39.73	39.42	39.07	39.28	35.49	35.21	34.72	34.98	30.21	30.08	29.48	29.73	26.72	26.38	25.97	26.19
ResLSTM	45.91	45.83	45.43	45.52	39.89	39.68	39.19	39.21	35.61	35.43	35.19	34.48	30.62	30.41	29.91	29.58	26.93	26.68	26.31	26.09
ltLSTM	45.71	45.32	45.12	45.2	39.43	39.28	38.89	39.17	35.21	34.83	34.51	34.72	30.07	29.91	29.27	29.48	26.42	26.01	25.62	25.89
cltLSTM	45.62	45.19	44.96	45.27	39.28	39.15	38.71	39.29	35.12	34.65	34.39	34.43	29.91	29.68	29.16	29.57	26.28	25.87	25.48	25.71
FNN+LSTM	47.21	-	-	-	41.83	-	-	-	34.28	-	-	-	30.42	-	-	-	27.53	-	-	-
LSTM+FNN+LSTM	47.93	-	-	-	42.87	-	-	-	36.21	-	-	-	32.49	-	-	-	29.78	-	-	-
LSTM+FNN+FNN	47.39	-	-	-	41.58	-	-	-	34.32	-	-	-	30.27	-	-	-	27.21	-	-	-
BLSTM	46.51	-	-	-	40.32	-	-	-	33.79	-	-	-	29.51	-	-	-	26.89	-	-	-
1-D CNN+BLSTM	46.39	-	-	-	40.12	-	-	-	32.87	-	-	-	28.32	-	-	-	25.69	-	-	-
1-D CNN+LSTM	45.83	-	-	-	39.98	-	-	-	32.48	-	-	-	27.89	-	-	-	25.21	-	-	-
1-D CNN+GRU	45.23	-	-	-	40.18	-	-	-	32.69	-	-	-	28.13	-	-	-	25.62	-	-	-
QuartzNet		43.79			38.51				32.13				28.04				24.93			
Transformer		42.15			37.62				31.35				27.17				23.69			

	60%				70%				80%				90%				100%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	23.47	23.26	22.89	23.08	22.68	22.41	21.93	22.13	20.93	20.69	20.28	20.51	19.61	19.28	18.79	18.93	18.53	18.27	18.06	18.13
ResLSTM	23.78	23.52	23.3	22.98	22.89	22.68	22.39	22.31	21.18	20.93	20.53	20.18	19.93	19.57	19.21	18.75	18.92	18.65	18.27	18.39
ltLSTM	23.26	22.78	22.43	22.61	22.51	21.62	21.32	21.49	20.62	20.13	20.02	20.12	19.47	18.89	18.53	18.72	18.17	18.21	17.91	18.13
cltLSTM	23.07	22.45	22.29	22.53	22.18	21.42	21.19	21.34	20.41	20.24	19.94	20.15	19.32	18.69	18.28	18.52	18.08	18.03	17.41	17.79
FNN+LSTM	25.19	-	-	-	23.61	-	-	-	21.49	-	-	-	20.32	-	-	-	19.18	-	-	-
LSTM+FNN+LSTM	27.52	-	-	-	25.18	-	-	-	22.83	-	-	-	21.59	-	-	-	19.97	-	-	-
LSTM+FNN+FNN	24.97	-	-	-	23.17	-	-	-	21.28	-	-	-	20.12	-	-	-	18.93	-	-	-
BLSTM	24.62	-	-	-	23.08	-	-	-	20.95	-	-	-	19.89	-	-	-	18.57	-	-	-
1-D CNN+BLSTM	24.17	-	-	-	22.77	-	-	-	20.51	-	-	-	19.32	-	-	-	18.05	-	-	-
1-D CNN+LSTM	23.72	-	-	-	22.13	-	-	-	20.18	-	-	-	18.91	-	-	-	17.65	-	-	-
1-D CNN+GRU	23.92	-	-	-	22.56	-	-	-	20.39	-	-	-	19.11	-	-	-	17.92	-	-	-
QuartzNet	21.53				18.39				17.71				16.39				15.19			
Transformer	21.18				18.72				16.53				15.28				14.21			

Figure A.5: Train and Test different models on WSJ

	10%				20%				30%				40%				50%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	41.81	41.07	40.67	41.28	38.73	37.87	37.08	37.61	31.58	30.62	30.18	30.48	28.41	27.85	27.32	27.72	25.79	25.03	24.89	25.38
ResLSTM	42.19	41.39	40.97	41.21	38.08	37.21	36.41	36.92	31.92	30.97	30.28	30.35	28.19	27.83	26.61	26.78	25.09	24.38	24.13	24.51
ltLSTM	41.97	40.53	40.17	40.37	36.65	35.71	34.89	35.32	31.47	30.53	30.03	30.28	27.27	26.62	25.97	26.53	24.61	23.91	23.51	23.47
cltLSTM	41.75	40.21	39.96	40.11	36.58	35.64	34.32	34.71	31.18	30.41	29.91	30.35	26.19	26.59	25.83	26.29	23.47	22.73	22.42	22.83
FNN+LSTM	45.63	-	-	-	40.68	-	-	-	33.78	-	-	-	29.81	-	-	-	26.41	-	-	-
LSTM+FNN+LSTM	46.27	-	-	-	41.29	-	-	-	34.79	-	-	-	31.19	-	-	-	26.83	-	-	-
LSTM+FNN+FNN	45.59	-	-	-	40.37	-	-	-	33.45	-	-	-	29.37	-	-	-	26.15	-	-	-
BLSTM	44.83	-	-	-	39.65	-	-	-	33.12	-	-	-	28.36	-	-	-	25.62	-	-	-
1-D CNN+BLSTM	44.65	-	-	-	39.58	-	-	-	32.92	-	-	-	28.17	-	-	-	25.43	-	-	-
1-D CNN+LSTM	44.53	-	-	-	39.47	-	-	-	32.17	-	-	-	28.07	-	-	-	25.28	-	-	-
1-D CNN+GRU	44.68	-	-	-	39.55	-	-	-	32.78	-	-	-	28.18	-	-	-	25.48	-	-	-
QuartzNet		41.38			35.21				30.82				26.62				23.95			
Transformer		39.17			34.72				29.48				25.75				22.49			

	60%				70%				80%				90%				100%			
	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8	2	4	6	8
LSTM	23.68	22.91	22.53	22.79	21.75	21.49	21.23	21.38	20.61	20.43	20.28	20.35	19.23	19.05	18.53	19.08	18.17	18.83	17.61	18.04
ResLSTM	22.89	22.17	21.98	22.28	21.97	21.21	20.78	21.93	20.89	20.45	20.23	20.37	19.48	19.32	18.73	19.98	18.49	18.18	17.51	18.01
ltLSTM	22.43	21.72	21.58	21.97	21.28	20.98	20.33	20.87	20.53	20.36	19.61	20.03	18.13	17.73	17.28	17.83	17.03	16.61	16.47	16.52
cltLSTM	21.28	20.68	20.52	20.61	20.51	20.18	19.71	20.02	19.18	19.03	18.63	18.95	17.93	17.39	16.97	17.21	16.39	15.97	15.61	15.78
FNN+LSTM	24.79	-	-	-	22.58	-	-	-	20.49	-	-	-	19.27	-	-	-	18.63	-	-	-
LSTM+FNN+LSTM	25.27	-	-	-	22.91	-	-	-	20.83	-	-	-	19.57	-	-	-	18.87	-	-	-
LSTM+FNN+FNN	24.45	-	-	-	22.31	-	-	-	20.29	-	-	-	19.13	-	-	-	18.42	-	-	-
BLSTM	23.17	-	-	-	21.78	-	-	-	19.83	-	-	-	18.86	-	-	-	18.21	-	-	-
1-D CNN+BLSTM	23.58	-	-	-	21.48	-	-	-	19.49	-	-	-	18.73	-	-	-	17.98	-	-	-
1-D CNN+LSTM	23.36	-	-	-	21.31	-	-	-	19.27	-	-	-	18.61	-	-	-	17.83	-	-	-
1-D CNN+GRU	23.68	-	-	-	21.53	-	-	-	19.39	-	-	-	18.87	-	-	-	18.08	-	-	-
QuartzNet	21.83				19.61				17.38				15.51				14.38			
Transformer	20.23				17.83				16.19				14.69				13.73			

Figure A.6: Train and Test different models on LibriSpeech