Low-Resource Speech Recognition Using Pre-trained Speech Representation Models

by

Huang, Chun Fung Ranzo

A Thesis Submitted to
The Hong Kong University of Science and Technology
in Partial Fulfillment of the Requirements for
the Degree of Master of Philosophy
in Computer Science and Engineering

August 2023, Hong Kong

Authorization

I hereby declare that I am the sole author of the thesis.

I authorize the Hong Kong University of Science and Technology to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the Hong Kong University of Science and Technology to reproduce the thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Huang, Chun Fung Ranzo

14 August 2023

Low-Resource Speech Recognition Using Pre-trained Speech Representation Models

by

Huang, Chun Fung Ranzo

This is to certify that I have examined the above MPhil thesis
and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by
the thesis examination committee have been made.

Thesis Supervisor, Dr. Brian Mak

Department Head, Prof. Xiaofang Zhou

Department of Computer Science and Engineering
14 August 2023

Acknowledgements

I am grateful for the support and guidance provided by my thesis supervisor, Prof. Brian Mak, throughout my study. He taught me that while the technical aspect of a research is important, the way of presenting the research in a clear and consistent manner is also important to make it understandable to a variety of audiences. His attention to detail and feedback have improved the readability of this thesis and been invaluable to me. I am also thankful for his patience as I worked towards completing my thesis.

I would also like to express my appreciation to Prof. Helen Meng at CUHK for creating the theme-based research project on automated NCD detection. The project has brought together wonderful researchers from multiple disciplines at CUHK, HKUST and PolyU, facilitating the share of ideas for creating the complete detection pipeline. It not only provided the data and computing facilities for some of the experiments in this thesis, but also invaluable opportunities to share my research with external guests. I would also like to acknowledge the valuable feedback provided by my colleagues from the project on my presentations on the early investigations and part of this work.

My thanks also go out to my friends, who listened to me and helped me to stay motivated and finally reach a closure.

Contents

List of Figures

List of Tables

Low-Resource Speech Recognition Using Pre-trained Speech Representation Models

by Huang, Chun Fung Ranzo

Department of Computer Science and Engineering

The Hong Kong University of Science and Technology

Abstract

Difficulties in eliciting substantial spoken data from speaker populations of interest and producing the accompanying transcripts result in low-resource scenarios in which the development of robust automatic speech recognition (ASR) systems may be hindered. With the aid of a large volume of unlabeled audio data, self-supervised speech representation learning may address this limitation by learning a model-based feature extractor via a proxy task in advance, thus offering pre-trained representations transferable to the ASR task for fine-tuning. This dissertation reviews current self-supervised speech representation learning methodologies and investigates the application of wav2vec 2.0 ASR on a developing corpus named CU-MARVEL in order to provide automatic transcripts for streamlining its human transcription work. The said corpus involves spontaneous responses from Cantonese-speaking older adults in Hong Kong—a unique setting concerning a language and a population that are both low-resource. We contribute a Cantonese wav2vec 2.0 model that is pre-trained on audio data obtained from the web and segmented using end-to-end neural diarization methods. We evaluate the usefulness of further pre-training on in-domain data and semi-supervised learning by pseudo-labeling for ASR under the pre-training-and-fine-tuning paradigm. Given the availability of cross-lingual wav2vec 2.0 models, we also compare the downstream performance of the monolingual pre-trained model to that resulted from the cross-lingual 300M XLS-R model and justify if a monolingual pre-trained model is necessary. We benchmark our results against those obtained from parallel experiments on the English LibriSpeech corpus. Our best performing model for CU-MARVEL is the 300M XLS-R further pre-trained in two stages: first adapting to the target language and then confining to the target domain. On participants' speech it reduces the character error rate (CER) of the vanilla XLS-R baseline by 23.1% relatively. This dissertation concludes with suggesting directions for future research.

Chapter 1

Introduction

The development of automatic speech recognition (ASR) systems typically relies on *paired* speech (the input) and text (the supervision) for training the acoustic model, which nowadays is often modeled by a neural network. The neural network-based acoustic model maps raw waveform or manually-defined acoustic features (e.g., MFCC and filter-bank coefficients) to sub-word tokens (e.g., phonemes, characters, and byte-pair encoding) and emits frame-level probabilities over the space of sub-word tokens; and given these probabilities, the decoder finds the most probable word sequence that the input utterance conveys via a language model.

Such paired speech and text may be obtained by eliciting speech from recruited speakers in two ways: (i) the speakers are asked to read predefined texts (e.g., Common Voice[1]); (ii) the speakers are asked to freely talk about a predefined topic, and the recorded speech are manually transcribed later (e.g., CALLHOME). Generalization comes into question for models learned from data collected using the first method because predefined texts yield limited phonetic contexts, unless the texts have been carefully designed to be phonetically balanced. Inefficiency is the major problem with the second method since manual transcription is expensive and time-consuming, and hence the method is not scalable. A more vigorous approach to creating paired data is to make use of pre-existing data: LibriSpeech [21] and the later Multilingual LibriSpeech (MLS) [63] are examples of large-scale efforts which create speech corpora by aligning audio and texts from audiobooks in the public domain. However, only a few languages (e.g., English and German) benefit from the alignment approach because many others do not come with a considerable number of public domain books, not to mention audiobooks.

Without a scalable data collection approach that applies, there are languages with only a handful amount of paired data readily available. This lack of supervised data hinders the creation of a robust ASR system for those target languages, and is referred to as a *low-resource*

[1] https://commonvoice.mozilla.org

scenario. Another low-resource scenario is due to age: with the past assumption of the target users of general-purpose ASR systems ranging from teenagers through middle-aged adults, few speech corpora contain representative samples of children or older adult speech. The earliest stage of the development of an in-domain ASR system may also be deemed to be low-resource, in which a few collected audio recordings have their transcriptions done and thoroughly checked to serve as the supervised training data for creating an initial system.

While paired data is hard to obtain, *unpaired* data is not: it is not hard to obtain extensive audio data from the web; and even in the context of in-domain ASR development, audio recordings often arrive faster than manual transcriptions. To improve recognition performance under the supervised setting, an obvious choice of utilizing unlabeled audio data is to create supervisions for the unlabeled data by transcribing them with an initial model, thereby expanding the supervised data set for further training, i.e., semi-supervised learning by pseudo-labeling. This approach lies on the assumption that the initial model's predictions are likely to be correct, at least for those of high confidence [8]; nonetheless, the method is prone to errors contained in the supervised data, low diversity of speakers, and limited phonetic contexts.

The advent of self-supervised speech representation learning challenges the customary supervised setting by suggesting that the properties of a large volume of unlabeled audio data may be exploited through self-supervised learning for improved supervised and semi-supervised learning, especially in low-resource scenarios. In other words, representation learning is done prior to supervised learning. Speech representation learning serves as a *pre-training* task for obtaining a pre-trained model which provides features for a variety of speech-related downstream tasks, including ASR, speaker recognition and spoken language identification, etc. One may directly fine-tune the pre-trained model weights using the objective of a downstream task, or rather extract latent features from the pre-trained model and train a classifier from scratch. The idea of *self-supervised* or *unsupervised pre-training* followed by *supervised fine-tuning* is not completely new in ASR, and the earliest may trace back to the now obsolete greedy layer-wise generative pre-training procedure for DNN-based acoustic models (e.g., see [15]), which was believed to stabilize DNN training but later found unnecessary. It was not until contrastive predictive coding (CPC) [38] that the approach drew new attention. CPC considered representation learning as a pre-training task for ASR by using noise contrastive estimation (NCE) [9], and adopted a more sophisticated design of model architecture, which is

based on CNN and GRU. However, the work of CPC did not examine large-scale pre-training. Later, the wav2vec 2.0 [52] methodology showed a breakthrough in ASR with limited labels that it surpassed the supervised learning-only Librispeech benchmark, and vitalized the line of research on pre-training for speech tasks.

Requiring audio-only data, self-supervised speech representation learning may easily apply to a cross-lingual setting, where data in diverse languages are used to pre-train speech representations. The setting is adopted in a hope that the resulting representations will be generalizable across multiple languages, thereby alleviating the need of language-specific pre-trained models, which are computationally intensive and hence expensive to create: in the case of LibriSpeech, which comprises 960 hours of training data, its wav2vec 2.0 pre-training of a 100M Transformer model using distributed training on 64 NVIDIA Tesla V100 GPUs takes 1.6 days [52]. XLS-R [76], a recent example of pre-trained cross-lingual speech representations available to the public, are a collection of wav2vec 2.0 models in three sizes (namely 300M, 1B, and 2B parameters) pre-trained on 436K hours of speech data in 128 languages, however the majority of data are in European languages. Although the language-universal property of such representations seems attractive, vanilla cross-lingual speech representations may suffer from language interference and underperform the language-specific counterpart.

This thesis investigates the application of ASR using pre-trained wav2vec 2.0 models on the CU-MARVEL corpus, whose data collection and transcription work is still in progress at the time of writing, in order to provide automatic transcripts for streamlining human transcription work. CU-MARVEL comprises spontaneous and conversational responses from recruited Cantonese-speaking older adults in Hong Kong, who are to complete a battery of neurocognitive disorder (NCD) screening tasks on several occasions under the guidance of human assessors, thus involving a language and a population that are both low-resource, at least before the corpus is finalized. We perform parallel experiments on a well-known corpus, the English LibriSpeech, for benchmarking purpose. We have the following contributions:

(i) To the best of our knowledge, we contribute the first publicly available Cantonese wav2vec 2.0 model[2] that is pre-trained on 2.8K hours of spontaneous speech data extracted from podcast and YouTube shows through the use of an end-to-end neural di-

[2] Available at https://huggingface.co/wcfr/wav2vec2-conformer-rel-pos-base-cantonese .

arization model trained on simulated conversations in the matched language.

(ii) We compare the downstream ASR performance of monolingual wav2vec 2.0 models, which provide language-specific representations, to that of the 300M XLS-R, a cross-lingual wav2vec 2.0 model and provides universal speech representations.

(iii) We evaluate the usefulness of further pre-training on monolingual and cross-lingual models using in-domain data for wav2vec 2.0 ASR.

(iv) We examine the effectiveness and necessity of wav2vec 2.0 ASR under settings with varying amounts of pre-training and fine-tuning data.

(v) We evaluate the usefulness of semi-supervised learning by pseudo-labeling with in-domain data for wav2vec 2.0 ASR.

The rest of this thesis is organized as follows: Chapter 2 offers a brief review on the related work on ASR and self-supervised speech representation learning; Chapter 3 describes the methodology and datasets used; Chapter 4 details the experiment configurations and results and discusses their implications; Chapter 5 concludes the thesis and suggests further research directions.

Chapter 2

Related Work

2.1 Automatic Speech Recognition (ASR)

2.1.1 Building Blocks of an ASR system

Automatic speech recognition (ASR) involves the use of a speech recognizer in transcribing audio segments which contain an utterance. More formally, the speech recognizer extracts a sequence of feature vectors $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_T)$ from the input audio samples $a[n]$ and finds the most probable word sequence $\mathbf{w}^* = (w_1^*, ..., w_S^*)$ out of the set of all possible sequences $\mathcal{W}^*$:

$$\mathbf{w}^* = \arg\max_{\mathbf{w} \in \mathcal{W}^*} P(\mathbf{w} \mid \mathbf{X}) \,. \tag{2.1}$$

Modeling the conditional distribution $P(\mathbf{w} \mid \mathbf{X})$ by one model without modularization is generally difficult because we would limit ourselves to using only paired training data. Moreover, we usually model a sequence of sub-word tokens $\mathbf{y}$ (e.g., phonemes, characters, byte-pair encoding) in place of a word sequence $\mathbf{w}$ to better share the units to model and gain the flexibility of representing words unseen during training. Recognizing these facts, we may apply the Bayesian classification rule and rewrite Equation (2.1) to provide a modularized approach, which incorporates the *a priori* knowledge of linguistic structures (e.g., [13, 50]):

$$\begin{aligned}
\mathbf{w}^* &= \arg\max_{\mathbf{w} \in \mathcal{W}^*} P(\mathbf{X} \mid \mathbf{w}) P(\mathbf{w}) / P(\mathbf{X}) && (2.2)\\
&= \arg\max_{\mathbf{w} \in \mathcal{W}^*} P(\mathbf{X} \mid \mathbf{w}) P(\mathbf{w}) && (2.3)\\
&= \arg\max_{\mathbf{w} \in \mathcal{W}^*} \sum_{\mathbf{y} \in \mathcal{Y}^*} P(\mathbf{X} \mid \mathbf{y}) P(\mathbf{y} \mid \mathbf{w}) P(\mathbf{w}) \,, && (2.4)
\end{aligned}$$

where $\mathcal{Y}^*$ is the set of all possible pronunciations (or spellings), and it is assumed that $\mathbf{X}$ and $\mathbf{w}$ are conditionally independent given $\mathbf{y}$. Following the decomposition as in Equation (2.4), we govern the likelihood $P(\mathbf{X} \mid \mathbf{y})$ by an *acoustic model*, the pronunciation probabilities $P(\mathbf{y} \mid \mathbf{w})$

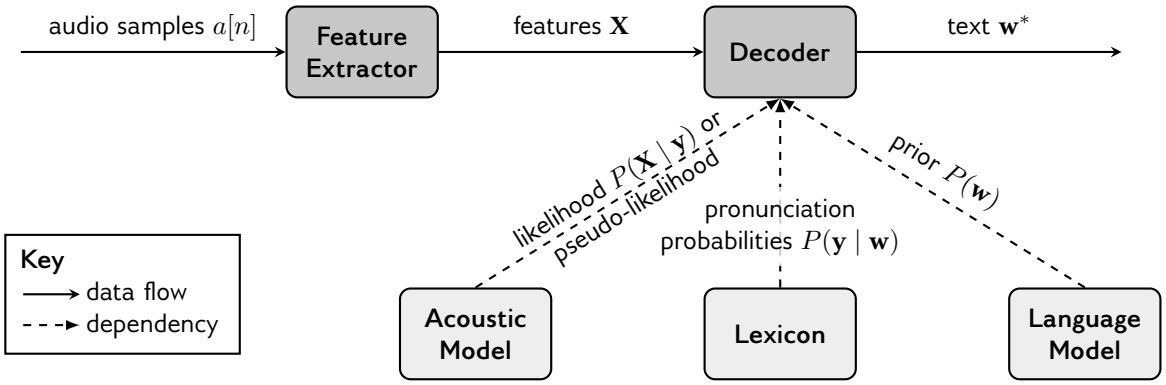


Figure 2.1: Schematic diagram of a typical modularized ASR system

by a *lexicon*, and the prior $P(\mathbf{w})$ by a *language model (LM)*. The formulation allows the components to be learned individually—in particular, we may train the acoustic model using paired data and the language model using text-only data. Figure 2.1 shows a schematic diagram of the resulting ASR system.

2.1.2 Approaches for Acoustic Modeling

The problem of acoustic modeling may be approached by (i) generative modeling, (ii) discriminative modeling, or (iii) a hybrid of the two.

Before the rise of deep learning, generative modeling with GMM-HMM dominated the realm of ASR. In the GMM-HMM framework, first-order HMM is usually assumed and a token (usually phoneme) is sub-divided into a number of states according to a pre-defined topology. GMM-HMM models the likelihood of observing a feature vector sequence $\mathbf{X}$ as

$$P(\mathbf{X} \mid \mathbf{y}) = \prod_{t=1}^{T} P(\mathbf{x}_t \mid s_t) P(s_t \mid s_{t-1}) \,, \tag{2.5}$$

where the observation probabilities $P(\mathbf{x}_t \mid s_t)$ are governed by GMM and the transition probabilities $P(s_t \mid s_{t-1})$ (from state s_{t-1} to s_t) are governed by HMM (see e.g., [13]).

In the 2010s, hybrid ANN-HMM models had received more interest. The state posterior $P(s \mid \mathbf{x})$ is modeled by a neural network which discriminates between different states, while the transition probabilities may be derived from an existing GMM-HMM system or be manually defined and untrained in *end-to-end training* (i.e., without the reliance on a GMM-HMM). Through the use of model architectures that capture a wider temporal context, the conditional

independence assumption in Equation (2.5) may be made less strong. For these models, we adopt the following pseudo-likelihood in place of the state likelihood:

$$P(\mathbf{x} \mid s) \sim P(s \mid \mathbf{x})^{\beta} / P(s)^{\gamma} \,, \tag{2.6}$$

where β and γ are hyper-parameters (see e.g., [34]).

More recently, discriminative acoustic modeling of $P(\mathbf{y} \mid \mathbf{X})$ with deep learning and without the involvement of HMM have become immensely popular. Methodologies in this realm usually feature end-to-end training, often in an attempt to achieve *end-to-end ASR* (i.e., without the modularization as in Equation (2.4) and the use of a pronunciation dictionary). The following are some notable examples:

(i) Connectionist Temporal Classification (CTC) [3] essentially performs frame-level classification of tokens (often characters nowadays) based on an encoder-only architecture. It employs a special 'blank' token $\varnothing$ to separate between characters $y \in \mathcal{Y}$ so that the text predicted may be simply read by taking the argmax of the predicted distributions over $\mathcal{Y} \cup \{\varnothing\}$, collapsing repeating characters and removing the blank tokens, albeit beam search decoding may provide better results.

(ii) RNN-Transducer (RNN-T) [11] defines an encoder-decoder architecture in which an RNN encoder (aka. transcription network) embeds the input features, an RNN decoder (aka. prediction network) embeds the previous output tokens and models the token transitions, and a joiner combines the input and output embeddings and emits token distributions over the time length T and over the output sequence length S for beam search decoding.

(iii) Listen, Attend and Spell (LAS) [26] defines an encoder-decoder architecture in which a Bi-LSTM-based encoder (aka. *Listener*) embeds and downsamples the input features, and an LSTM-based decoder emits a token sequence through the mechanism of encoder-decoder attention (aka. *Attend and Spell*).

(iv) Joint CTC-Attention [32] speeds up the training of attention-based encoder-decoder models by imposing the CTC loss on the encoder.

The upcoming parts of this thesis will focus on encoder-only and non-streaming models.

2.1.3 Training Objectives for Acoustic Modeling

Below we describe two training objectives that may be applied on encoder models.

Connectionist Temporal Classification (CTC)

Since CTC [3] uses a blank symbol $\varnothing$ to separate between tokens $y \in \mathcal{Y}$, it may be thought of as a special case of HMM that adopts a special two-state topology for modeling the tokens, and uses no transition probabilities and no state prior [34]. The first state of that topology corresponds to a character, and the second state refers to the blank symbol and is shared across all the HMMs. In addition, the character state may transit to every other state, and it must go through the blank state to reach itself so as to distinguish between repeating characters.

Let $\mathcal{M}_{\mathbf{w}}$ denote the supervision graph built from the transcription $\mathbf{w}$. The CTC loss for an utterance u is given by

$$\mathcal{L}_{\mathsf{CTC}}^{(u)} = -\log P(\mathcal{M}_{\mathbf{w}} \mid \mathbf{X}) = -\log \sum_{\mathbf{s} \in \mathcal{M}_{\mathbf{w}}} \prod_{t=1}^{T} P(s_t \mid \mathbf{x}_t)\,, \tag{2.7}$$

where the sum is over all possible alignments $\mathbf{s} \in \mathcal{M}_{\mathbf{w}}$ and can be efficiently computed online using the forward-backward algorithm [34].

Maximal Mutual Information (MMI)

According to [29], MMI is a sequence discrminative objective which maximizes the probability of the reference transcript and minimize that of the others. The MMI loss for an utterance u is given by

$$\begin{aligned}
\mathcal{L}_{\mathsf{MMI}}^{(u)} &= -[\log P(\mathbf{X} \mid \mathcal{M}_{\mathbf{w}}) - \log P(\mathbf{X})] && (2.8)\\
&= -\log P(\mathbf{X} \mid \mathcal{M}_{\mathbf{w}}) + \log \sum_{\mathbf{w}' \in \mathcal{W}^*} P(\mathbf{X} \mid \mathbf{w}') P(\mathbf{w}') && (2.9)\\
&\approx -\log P(\mathbf{X} \mid \mathcal{M}_{\mathbf{w}}) + \log P(\mathbf{X} \mid \mathcal{M}_{\mathsf{den}})\,. && (2.10)
\end{aligned}$$

Variants of MMI are characterized by the estimation methods of the denominator graph $\mathcal{M}_{\mathsf{den}}$. Specifically, lattice-free MMI adopts a pruned phone (other tokens are also applicable) n-gram LM estimated from GMM-HMM alignments [29] or random pronunciations of the training transcripts [35] for constructing the denominator graph.

2.1.4 Architectures for Acoustic Models

Below we review some of the commonly used architectures for acoustic models.

Convolutional Neural Network (CNN) and Time-Delay Neural Network (TDNN)

A CNN transforms patches of the input locally along the spatial dimension(s) by applying on each patch a set of learnable kernels or filters of the same size (the *kernel size*). The filters may be *dilated* for an enlarged receptive field without increasing the parameter count. The input may be subsampled by extracting patches every $s > 1$ step (the *stride*). Filter-bank features may be embedded by 2d-CNNs that convolve along the time and frequency axes before 1d-CNNs.

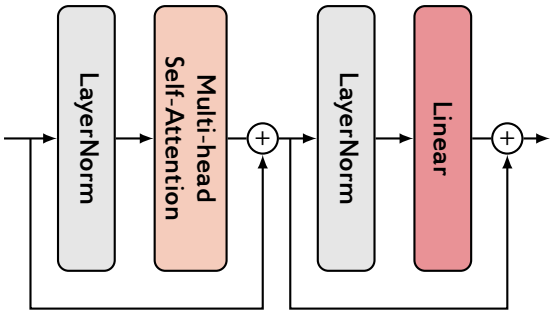
TDNN generalizes dilated CNN in that its kernels may have asymmetric time contexts, which are defined by *splicing* frame offsets of the layer's input. With considerations about online decoding latency and word recognition accuracy (which may not correlate with frame-level accuracy), more contexts are usually taken from the left [22], hence the 'time delay'.

To make TDNN more parameter-efficient, [39] introduces TDNN-F (where 'F' stands for 'factorized'), which factorizes the weight matrix of a TDNN (the kernel and input dimensions are flattened into one) into two smaller and successive factors, with the first enforced to be semi-orthogonal.

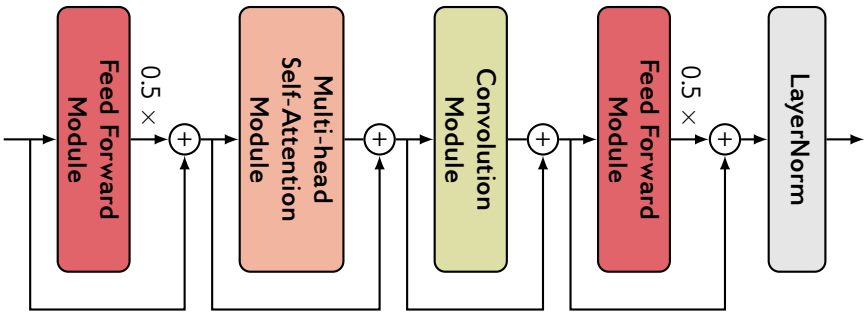
Transformer

Transformer [33] effectively models long-term dependencies. It transforms the embedded input through a global attention mechanism that maps the *query* vectors to an output by computing for each query vector a similarity-based weighted sum of the *value* vectors. Each value vector is weighted by the dot product between its associated *key* vector and the given query vector. In an encoder, *self-attention* may be used in which the query, key and value vectors are projections of the input vectors. For streaming applications, a causal attention mask is used to prevent attending to keys ahead of time during training.

To make the attention mechanism position-aware, the input vectors may be added with sinusoidal positional encoding, relative sinusoidal positional encoding, or learned positional encoding before they are presented to the Transformer layers.



(a) *Pre-norm* Transformer layer



(b) Conformer layer

Figure 2.2: Some building blocks for acoustic models

The pre-norm variant of Transformer, which places layer normalization within the residual blocks and adds layer normalization after a stack of Transformer layers, leads to more stable training and faster convergence [64]. Figure 2.2a shows the pre-norm Transformer layer.

Conformer

Conformer [55], which stands for convolution-augmented Transformer, imposes the notion of local connectivity on the Transformer architecture by incorporating a 1d-convolution module. In a Conformer layer, a pair of feed-forward modules sandwiches the multi-head self-attention and the convolution modules. Every module is a residual block and begins with layer normalization, i.e., the architecture is pre-norm.

The feed-forward module consists of two linear layers, with the first expanding the dimension by a factor of 4 and the second projecting back the the model dimension. The multi-head self-attention module consists of a multi-head self-attention layer and a dropout layer. The convolution module is based on depth-wise separable convolution, and it performs point-wise convolution, followed by depth-wise convolution and again point-wise convolution. Figure 2.2b shows a simple view of the Conformer layer.

2.1.5 Speech Features for ASR

Speech signals exhibit locally quasi-periodic phenomena (which refer to voiced sounds) interjected by transients (which refer to unvoiced sounds) and silences, and are relatively stationary in time intervals of 5 to 25 ms [7]. To ease analysis of the lengthy speech signals, which are usually sampled at 16 kHz for speech applications, feature extraction performs a sliding-window transform on the (real-valued) sampled signal to provide local analyses of the signal while downsampling the signal in the time domain. Another important role of feature extraction is to discard information irrelevant to or not useful for classification, e.g., noise.

Speech features are traditionally extracted by signal processing techniques and presented to the acoustic model, the extraction process is therefore known as *front-end processing*. In recent years, there have been interests in *model-based* feature extractors, which may be learned end-to-end via the optimization of ASR losses, or through a pre-training task of self-supervised representation learning. The latter is particularly interesting because it may exploit the properties of a large volume of unlabeled audio data.

Features based on Signal Processing

Acoustic features extracted through signal processing means are due to local time-frequency analyses, and may be described in a convolution manner. A short-time analysis operation is performed on a window of M samples (which we call the *window length* or *frame size*) every L samples (which we call the *hop size* or *frame shift*) to yield a sequence of feature frames. The t-th segment $a_t[n]$ undergoes a short-time transform which results in a time-localized spectrum $\mathbf{x}_t$. The transform may base on Fourier analysis, Gammatone analysis (e.g., [4]), or rarely, wavelet analysis (e.g., [1]). We may express the transform on a real and discrete signal by a bank of K analysis filters $h_k[n]$ bucketing the frequency range:

$$x_{t,k} = \sum_{n=0}^{M-1} a_t[n] h_k[-n] \,, \tag{2.11}$$

where n is relative to the start of $a_t[n]$, and $K \geq M$ is enforced to ensure the transform is adequately sampled [5, 40].

The rest of this part will assume a context of the short-time Fourier transform (STFT) for its popularity. For discrete time and frequencies, we use the discrete Fourier transform (DFT).

According to [5], a DFT analysis filter is defined by

$$h_k^{\mathsf{STFT}}[n] = w^{\mathsf{A}}[n] e^{-\mathrm{j}\omega_k n}\,, \tag{2.12}$$

where w^{A} is a window function, j is the imaginary unit, and $\omega_k = 2\pi k/K$ is the angular frequency. Then,

$$x_{t,k}^{\mathsf{STFT}} = \sum_{n=0}^{M-1} a[n] w^{\mathsf{A}}[-n] e^{-\mathrm{j}\omega_k n}\,. \tag{2.13}$$

It may be derived from Equation (2.13) that (1) the Fourier spectrum is smeared by the window function in the frequency domain, and (2) there exists a trade-off between time and frequency resolutions in Fourier analysis [7].

Window functions come with different design considerations, but a general agreement is to reduce the size and height of their side lobes in the frequency domain to avoid spectral leakage, i.e., the energy of a frequency leaking to its neighboring frequencies [7]. The Hamming window is often used in ASR, and the Kaldi speech recognition toolkit uses the so-called "Povey window"[1], which is a variant of the Hann window. The Hamming window specifically reduces the size of the first side lobe while Povey and Hann do not, and Povey and Hann decay the sizes of the side lobes much faster than Hamming[2].

According to [72], the configuration of the hop size and window length should be in accord with observations from speech production. The hop size should be short to track the rapid changes of the vocal tract shape, whereas the choice of the window length should be narrow to better localize speech events, and at the same time wide enough to accommodate at least one glottal cycle, smooth out the unvoiced speech signal, and make the analysis invariant to the position of the window. In practice, the hop size usually takes a value of 10 ms, and the window length is usually set to 25 ms (e.g., [7, 13]).

The (linear) spectrogram is a basic representation that displays the power spectrum over time, and is interpretable by human[3], e.g., see [72]. In other words, it does not contain phase infor-

[1] See https://github.com/kaldi-asr/kaldi/blob/master/src/feat/feature-window.cc .

[2] See https://groups.google.com/g/kaldi-help/c/UlxXU8agTaY .

[3] Phonological or phonetic analyses use a window length shorter than ASR applications (e.g., 10 ms) to perform formant analyses and better localize speech events.

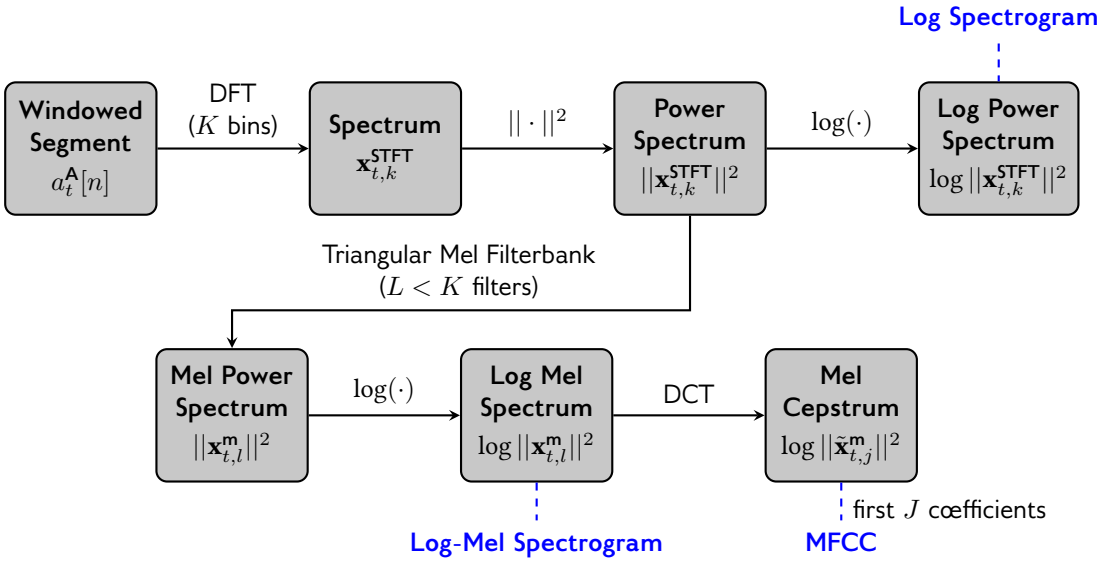
mation. In the past, to avoid the curse of dimensionality in statistical modeling, it is preferred to derive features with reduced dimensions from the spectrogram. With this in mind, together with motivation from the human auditory system, the auditory spectrogram is widely adopted in speech applications. According to [6, 14], the auditory spectrogram is a warped and reduced spectrogram through the imposition of a bank of $L < M$ band-pass filters on the original power spectra. The filters are equally distributed in some logarithmic-based perceptual scales to increase the frequency resolutions in low frequencies and reduce that in high frequencies, mimicking the functionality of auditory filters. In ASR, the mel scale

$$f^{\mathsf{Mel}}(f^{\mathsf{Hz}}) = 2595 \log_{10}\left(1 + \frac{f^{\mathsf{Hz}}}{700}\right) \tag{2.14}$$

is most often used. Possibly inspired by the idea of perceived loudness, the logarithm is applied on the spectrogram to reduce its dynamic range. These end up with the log-mel spectrogram (also mel filterbank features, mel filterbank coefficients). Further, applying the discrete cosine transform (DCT) on each frame of the log-mel spectrogram smooths the spectral estimate and decorrelates the frequency bins, and the end-product is called the mel-frequency cepstral coefficients (MFCC). Moreover, the cesptral representation separates information about the shape of the vocal tract from glottal excitation, and places the two in different regions of the cepstrum, making MFCC well-suited for ASR. Since the lower order cepstral coefficients are more relevant to signifying the vocal tract shape, we usually compute only the first J (typically less than L) DCT coefficients for the log-mel spectrogram. The summarized front-end feature extraction process is depicted in Figure 2.3.

Model-based Features

Although the conventional feature extraction pipeline is on solid ground, and coupling acoustic features with deep learning gives strong baseline results in ASR, the pipeline *per se* does not directly optimize for classification. As a data-driven substitute to the convolution-based transform in conventional front-end processing, convolutional neural networks (CNNs) are indispensable in extracting features from raw waveform. There is no assumption of performing analyses in the Fourier spectrum or cepstrum—the filters are optimized for classification or maximizing the likelihood of the data, depending on the actual objective. In this part, we focus on works that train the model-based extractors end-to-end with an ASR loss, and leave the discussion on self-supervised speech representation learning methods to Section 2.2, since



* The flowchart shows the extraction of one frame of features given a windowed segment, and the blue texts give the terminologies corresponding to a feature sequence.

Figure 2.3: Extraction of acoustic features based on short-time Fourier transform [12]

they come with different motivations.

Given an neural network-based acoustic model which operates on waveform, its layers which extract features from the raw waveform are loosely and collectively referred to as a *waveform encoder* in the following discussion. Such a waveform encoder is often made of '1d-CNN blocks', each of which comprises (1) a 1d-CNN layer that operates on time, (2) a non-linearity, and (3) a pooling layer to offer invariance to minor time shifts or subsample the sequence. In [27, 30], the encoder is simply a 1d-CNN block; in [17, 20, 31, 41, 46], a stack of 1d-CNN blocks; in [24], a 1d-CNN block that operates on time, another on the feature dimension, and a stack of long-short term memory network layers (LSTMs). Comparing waveform-based models to the acoustic feature-based counterpart in recognition rate terms, some find the waveform-based models to perform worse [17, 19, 20], similarly or slightly better [24, 27], or significantly better [27, 41].

With random initialization, the input 1d-CNN layer with a kernel size of at least 10 ms is found to learn bandpass filters with magnitude responses that coarsely resemble some perceptual scales [19, 20, 24, 27]. Varying the kernel size and/or stride of the CNN, which amounts to extracting features at variable rates, the learned filters are found to emphasize on different frequency ranges, deviating much from the perceptual scales [30, 31, 46]. This observation

motivates multi-stream models which jointly learn parallel CNNs operating at different kernel sizes and/or strides that offer complementary features in multiple resolutions for a time step, giving rise to significantly better recognition performance over single-stream models [30, 46].

Some attempt to incorporate prior knowledge about the filters of the lowest 1d-CNN layer into weight initialization or layer design. Seeing the success of auditory-inspired filters, some initialize the trainable CNN weights as the impulse responses of Gammatone filters [19, 24, 41] or scatter filters [41]. [24, 41] find the resultant recognition performances better than random initialization. [47] replaces the input CNN layer with SincNet which enforces the use of Sinc filters. Such filters are essentially bandpass and parameterized only by their low and high cutoff frequencies and therefore very parameter-efficient. The SincNet-based waveform encoder shows better results than the 1d-CNN counterpart in both DNN [47] and joint CTC-attention [62] setups on datasets in small to medium sizes.

Employing more sophisticated components for a waveform encoder may improve recognition performance. For example, [70] uses a waveform encoder which incorporates strided 1d-CNN, local RNNs, global attention and average pooling. The waveform encoder performs a sequence of subsampling operations and provide features at multiple scales at no cost to avail multi-stream learning. The resulting waveform encoder outperforms the 1d-CNN-only counterpart as well as MFCC features on a 21K-hour industrial dataset. Another possible way to improve ASR performance is to create a *tandem* model which makes use of both waveform and acoustic features, e.g., see [24].

2.2 Self-Supervised Speech Representation Learning

The main objective of self-supervised speech representation learning is to learn the properties of unlabeled data (usually in a large volume) that are transferable to the downstream tasks, which are usually supervised. The following text will use the term *pre-training* to refer to the representation learning stage and the term *fine-tuning* to refer to the transfer learning stage.

Speech representations may be learned upon (mel) spectrograms [42, 59–61, 79] or from raw audio [38, 48, 52, 69, 77, 81]. Although the direct use of waveform in pre-training has been commonplace nowadays, pre-training on log-mel spectrograms may provide competitive downstream ASR performance as shown in [79].

Below we take a bottom-up approach to describe the methodologies of self-supervised speech representation learning. Details of the methodologies are summarized in Table 2.1 at the end of this chapter.

2.2.1 Model Architectures

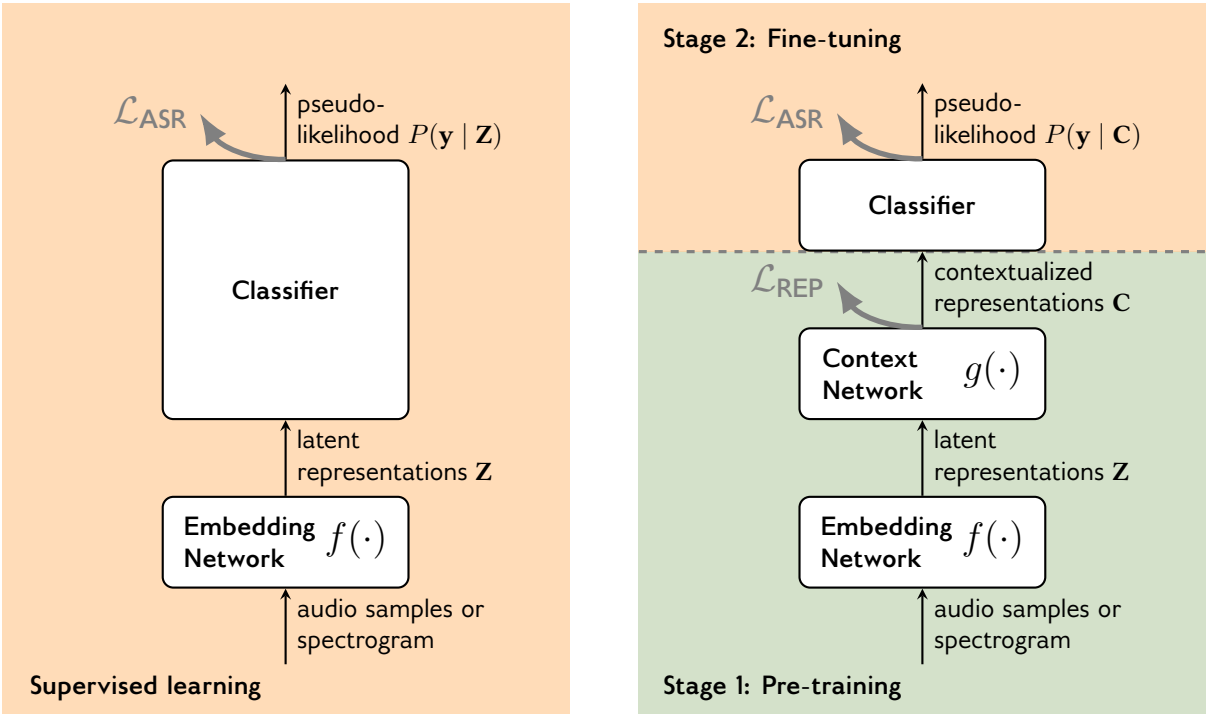
The model architectures for speech representation learning are similar to that for acoustic modeling. Here we define the model components for the two more formally. An encoder network $f : \mathcal{X} \mapsto \mathcal{Z}$, which is often based on CNNs, embeds the input (audio $a[n]$ or spectrogram $\mathbf{X}$) into latent representations $\mathbf{Z}$. For raw audio input, the network amounts to the waveform encoder we described in Section 2.1.5. For acoustic features, it may be optional, depending on the choice of the feature as discussed in Section 2.1.5. A context network $g : \mathcal{Z} \mapsto \mathcal{C}$, which may be based on RNNs or Transformers, contextualizes the latent representations $\mathbf{Z}$ into representations $\mathbf{C}$ that optimize a prediction-based representation learning loss $\mathcal{L}_{\mathsf{REP}}$. A classifier conditions on the latent representations or contextualized representations and performs the ASR classification task. For ASR without speech representation learning in advance (Figure 2.4a), the classifier may be a TDNN or Transformer-based. For ASR preceded by speech representation learning (Figure 2.4b), since the learned representations are already contextualized, the classifier may simply be a linear layer with Softmax activation if we allow the update of the context network.

2.2.2 Contexts for Prediction

The following prediction tasks are characterized by how the representations $\mathbf{C}$ are contextualized given the embedded input $\mathbf{Z}$.

Autoregression Task

An autoregression task essentially means prediction based on past history. In the context of natural language processing (NLP), an autoregression task often refers to next token prediction. For the downstream purpose of speech classification, doing a naive next frame prediction is however not robust because it is easy to exploit the local smoothness of the speech data during prediction. Alternatives include predicting consecutive frames in future [38] or one single frame several steps ahead of time [42]. An unrolled expression for the latter case may



(a) ASR without speech representation learning in advance

(b) ASR preceded by speech representation learning

Figure 2.4: Acoustic modeling with and without speech representation learning

be $\mathbf{c}_t = g^{(t-d)}(\mathbf{Z}_{\leq t-d})$, where $d > 0$. Methodologies such as Autoregressive Predictive Coding (APC) [42], Contrastive Predictive Coding (CPC) [38] and wav2vec [48] fall into this category.

The task may be extended to work in a bi-directional way [58], in which two context networks $\overrightarrow{g}$ operating from left to right and $\overleftarrow{g}$ operating from right to left perform the task independently given the same latent representation embedded by f. The resultant representations are the concatenation of the outputs of $\overrightarrow{g}$ and $\overleftarrow{g}$.

Cloze Task

In a cloze task for speech representation learning, a number of masks, each of which spans consecutive frames, is applied on $\mathbf{Z}$ and the context network recovers the masked frames by conditioning on the unmasked frames. The context network could be autoregressive or non-autoregressive. This is inspired by Bidirectional Encoder Representations from Transformers (BERT) [43], which performs language representation pre-training based on masking tokens. The methodolgies of Deep Contextualized Acoustic Representations (DeCoAR) [59, 60], Mockingjay [61], wav2vec 2.0 [52], Hidden-unit BERT (HuBERT) [69], data2vec [77] and Code BERT (CoBERT) [81] are tasks of this type.

2.2.3 Self-Supervised Learning Objectives

Reconstruction-based Learning

Because of the continuous nature of audio data and hence their latent representations, it is most straightforward to frame the prediction task as a reconstructive one, and minimize the difference between the latent and the contextualized representations over the frames to predict. This is achieved by optimizing the following L1 loss on each frame of an utterance u:

$$\mathcal{L}_{\text{L1}}^{(u,t)} = \left| \mathbf{z}_t^{(u)} - \mathbf{c}_t^{(u)} \right| . \tag{2.15}$$

Methodologies adopting this loss include APC [42], DeCoAR [60], and MockingJay [61].

Contrastive Learning by Noise Contrastive Estimation (NCE)

Noise-contrastive estimation (NCE) [9] allows a model to learn the properties of data through a surrogate classification task which differentiates the data from noise. The classification may be binary [9] or multi-class [28] (also ranking-based [36]). For conditional models estimated by NCE, they are in the following general form:

$$P(y \mid x; \theta) = \frac{\exp\left(s(x, y; \theta)\right)}{Z(x; \theta)} , \tag{2.16}$$

where $s\left(x, y; \theta\right)$ is a scoring function for the prediction y, and $Z(x; \theta) = \sum_{y \in \mathbb{Y}} \exp\left(s(x, y; \theta)\right)$ is the partition function which makes the density ratio a valid probability distribution [36].

We let a model learn to extract frame-level features through the following frame-level classification task. Given the contextualized representation $\mathbf{c}_t^{(u)}$ for the t-th frame of an utterance u, the model is to distinguish the latent representation $\mathbf{z}_t^{(u)}$ for the same frame (i.e., the positive sample) from K distractors $\mathbf{z}'_1, ..., \mathbf{z}'_K$ (i.e., the negative samples) drawn from a noise distribution $P_N(\mathbf{z})$. Let $\mathbb{Z}^{(u,t)}$ include $\mathbf{z}_t^{(u)}$ and the K drawn distractors. For the binary classification case, we may minimize the negative sampling loss [16], a simplified version of NCE-binary[4]:

$$\mathcal{L}_{\text{NEG}}^{(u,t)} = -[\log \sigma \left(s \left(\mathbf{c}_t^{(u)}, \mathbf{z}_t^{(u)} \right) \right) + \sum_{\mathbf{z} \in \mathbb{Z}^{(u,t)} \setminus \{\mathbf{z}_t^{(u)}\}} \log \left(1 - \sigma \left(s \left(\mathbf{c}_t^{(u)}, \mathbf{z} \right) \right) \right)], \tag{2.17}$$

[4] Please refer to [9] and [36] for the formulation of the true NCE-binary.

which is used in wav2vec [48]; and for the ranking case, we minimize the NCE-ranking loss:

$$\mathcal{L}_{\text{NCE\_Ranking}}^{(u,t)} = -\log \frac{\exp\left(s\left(\mathbf{c}_t^{(u)}, \mathbf{z}_t^{(u)}\right)\right)}{\sum_{\mathbf{z}\in\mathbb{Z}^{(u,t)}} \exp\left(s\left(\mathbf{c}_t^{(u)}, \mathbf{z}\right)\right)}\,, \tag{2.18}$$

which is used in CPC [42] and wav2vec 2.0 [52].

Regarding the scoring function, CPC uses bilinear similarity:

$$s\left(\mathbf{c}, \mathbf{z}\right) = \mathbf{c}^\top \mathbf{W} \mathbf{z}\,, \tag{2.19}$$

where $\mathbf{W}$ is a learnable weight matrix; whereas wav2vec and wav2vec 2.0 adopt cosine similarity:

$$s\left(\mathbf{c}, \mathbf{z}\right) = \text{sim}\left(\mathbf{c}, \mathbf{z}\right) = \frac{\mathbf{c}^\top \mathbf{z}}{||\mathbf{c}|| \cdot ||\mathbf{z}||}\,. \tag{2.20}$$

In our application, optimizing NCE-ranking actually maximizes the lower bound on the mutual information between $\mathbf{c}_t^{(u)}$ and $\mathbf{z}_t^{(u)}$, with a larger K resulting in a tighter bound [42]. This stems from the observation that $\exp\left(s\left(\mathbf{c}_t^{(u)}, \mathbf{z}_t^{(u)}\right)\right)$ is positive, and preserves the mutual information between $\mathbf{c}_t^{(u)}$ and $\mathbf{z}_t^{(u)}$:

$$\exp\left(s\left(\mathbf{c}_t^{(u)}, \mathbf{z}_t^{(u)}\right)\right) \propto \frac{P\left(\mathbf{z}_t^{(u)} \mid \mathbf{c}_t^{(u)}\right)}{P\left(\mathbf{z}_t^{(u)}\right)}\,. \tag{2.21}$$

Hence, [42] gives the loss function in Equation (2.18) the name 'InfoNCE'.

The choice of the noise distribution $P_N(\mathbf{z})$ and the number of distractors K affects the downstream ASR performance. Assuming that each utterance regards only one speaker and negative samples are drawn from the noise distribution uniformly, it is found that in general, (1) a noise distribution over the frames confined in the positive sample's utterance works better than one that is over the frames from a minibatch of utterances, which are possibly spoken by various speakers, and the latter may make the positive sample easier to distinguish from the negatives [42, 52], and (2) a larger K may not improve performance [48, 52], e.g., in [52], $K = 100$ works better than $K = 200$. Drawing conclusions from the results of [42, 48, 52], as long as $P_N(\mathbf{z})$ includes frames from the positive sample's utterance, varying K gives only slight performance difference, say, a 1% absolute reduction of word error rate (WER).

Vector Quantization

Considering the discrete nature of speech units, wav2vec 2.0 [52] may use a quantized version of the latent representations $q(\mathbf{Z})$ as the training targets in NCE. This case adopts product quantization with Gumbel Softmax and optimizes the codebook with a diversity loss to maintain the equal utilization of the codebook entries. Its ablation study finds that using quantized targets improves ASR performance compared to continuous targets. Moreover, the discrete latent representations correlate well with phonemes.

Predicting Acoustic Units Discovered Offline: The Case of HuBERT

The training objective of Hidden-Unit BERT (HuBERT) [68, 69] is based on the frame-level classification of discrete acoustic units discovered offline—the 'hidden units'. The frame-level targets are cluster labels automatically obtained from clustering some representations, through e.g., K-Means or GMM. Initially, the targets come from the clustering of MFCCs, which are acoustic features well-justified for ASR (please refer to our earlier discussion in Section 2.1.5). The training process is repeated with targets derived from the learned representations obtained from the previous HuBERT model.

Studied by the original paper, an extension to the single-target learning is to predict labels of different granularity at the same time, which are obtained by specifiying different cluster sizes during clustering. Another extension is Code BERT (CoBERT) [81], which distills the cluster targets for HuBERT using masked tokens prediction or data2vec [77] (see Section 2.2.4).

2.2.4 Extensions

Gaining Environment Robustness

Assuming the pre-training data encompass clean recordings, WavLM [78] extends HuBERT by data augmentation. It adds background noise or an overlapping secondary utterance with lower energy to a primary utterance and lets the model predict the cluster labels for the primary utterance. The model then learns to perform speaker-aware classification and denoising at the same time. The resultant models outperform both wav2vec 2.0 and HuBERT in various speech tasks (including ASR) and overall rank the top places of the Speech Processing

Universal Performance Benchmark (SUPERB)[5].

Pre-training on far-field data and observing that wav2vec 2.0 may produce sub-optimal codebooks, which contain entries that differentiate between (1) speech and non-speech or (2) temporal locations, wav2vec-C [74] ensures the utilization of the codebook by enforcing consistency. It requires the spectrogram input to be reconstructed from its resulting quantized latent representations $q(\mathbf{Z})$. The use of the auxiliary consistency loss ends up with a small relative reduction in WER (best 1.4% relative).

Constructing Contextualized Training Targets from Teacher Model

While we maintained our discussion on training targets that are non-contextualized, i.e., performing contextualized prediction given the non-contextualized representations, data2vec [77] constructs training targets based on contextualized representations. In short, the student model predicts the output of the mean teacher given the same unmasked input, but the student receives a masked version of the input. The downstream ASR performance of this methodology ranks the top of the SUPERB benchmark at the time of writing.

2.2.5 What Do the Learned Representations Encode?

Although self-supervised speech representation models show very competitive performance in diverse speech tasks, some question their interpretability. A complementary line of research therefore attempts to explain the learned representations.

Based on empirical findings observed on pre-trained wav2vec 2.0 models, [80] hypothesizes that the latent representations outputted by the embedding network f may construct an acoustic metric space. It is found that in the latent space $\mathcal{Z}$, (i) different (fundamental) frequencies are spaced evenly by the cosine distance of their corresponding embeddings, and (ii) the vowel space represented by the embeddings exists as a smooth manifold with a grid--like structure.

Using correlation analyses and linear probing methods, it is found that pre-trained speech representation models have the ability to encode acoustic [73, 82, 86], articulatory [84], pho-

[5] SUPERB is established to assess the effectiveness of self-supervised speech representation learning methodologies by evaluating the downstream performance of the released pre-trained models on a battery of speech processing tasks. The up-to-date leaderboard may be found at https://superbbenchmark.org/leaderboard.

netic [66, 71, 73, 82, 86], word [73, 82, 86, 87], and even semantic [73, 86] information to some extent in the context network g. The nature of the training objectives determines where these information are encoded in the network. In general, prediction-based models, which are based on HuBERT's objective, encode acoustic information in the lowest layers and linguistic information in higher layers, whereas the others which form contextualized representations by recovering the latent representations manifest an autoencoder-like trend that the higher layers reverse the acoustic-linguistic hierarchy [73, 82, 86].

Table 2.1: Comparison of speech representation learning methodologies

Methodology	Model Input	Embedding Net f	Context Net g	Input of g	Target	Objective
Autoregression-based						
CPC [38]	raw waveform	strided 1d-CNN	GRU	$f(a[n])$	future $f(a[n])$	InfoNCE
APC [42]	mel spectrogram	n/a	LSTM	$\mathbf{X}$	future $\mathbf{X}$	L1
wav2vec [48]	raw waveform	strided, causal 1d-CNN	causal 1d-CNN	$f(a[n])$	future $f(a[n])$	negative sampling
Cloze-based						
DeCoAR [60]	mel spectrogram	n/a	bi-LSTM	$\mathbf{X}$	$\mathbf{X}$	L1
Mockingjay [61]	linear / mel spectrogram	n/a	Transformer	masked $\mathbf{X}$	$\mathbf{X}$	L1
wav2vec 2.0 [52]	raw waveform	strided 1d-CNN	Transformer	masked $f(a[n])$	$q(f(a[n]))$	InfoNCE + VQ diversity
wav2vec-C [74]	linear spectrogram	LSTM	Transformer	masked $\mathbf{X}$	$q(f(\mathbf{X}))$	InfoNCE + VQ diversity + consistency
HuBERT [69]	raw waveform	strided 1d-CNN	Transformer	masked $f(a[n])$	cluster labels of speech representations	a normalized version of cross-entropy
WavLM [78]	raw waveform, may be mixed with noise	strided 1d-CNN	Transformer	masked $f(a[n])$	HuBERT targets before aug.	same as HuBERT's
data2vec [77]	raw waveform	strided 1d-CNN	Transformer	$f(a[n])$ for mean teacher, masked $f(a[n])$ for student	contextualized output of mean teacher	smooth L1
CoBERT [81]	raw waveform	strided 1d-CNN	Transformer	same as data2vec when self-distilled, otherwise $f(a[n])$	HuBERT targets distilled by code teacher, optionally further self-distilled by mean teacher	L2

Chapter 3

Methodology

3.1 Objectives

In this thesis, we study the application of the self-supervised learning pipeline for ASR using the wav2vec 2.0 methodology on two databases, CU-MARVEL (Cantonese spontaneous speech) and LibriSpeech (English read speech). We cover the following aspects:

(i) **Creating pre-training data with end-to-end neural diarization (EEND)**
This thesis involves the use of unsegmented audio recordings for pre-training a wav2vec 2.0 model from scratch or further pre-training. Assuming the downstream ASR model operates per speaker turn, it is desirable for the pre-training data to be segmented accordingly. In preliminary experiments, we find bad segmentation of the pre-training data could deteriorate the downstream ASR performance. Proper segmentation is therefore a crucial pre-processing step. We consider using end-to-end neural diarization (EEND), which performs voice activity detection for a number of speakers at once (i.e., speaker diarization) with neural networks, for the job. We describe our protocol in training EEND models and performing segmentation using these models.

(ii) **Further pre-training for domain adaptation**
Although applying an off-the-shelf pre-trained wav2vec 2.0 model on in-domain data may save the development cost of training an in-domain model from scratch, it may not be optimal because of the mismatch in the recording environments, recording equipment and speaker demographics between the pre-training data and the in-domain data. We therefore question the gain on ASR performance resulted from the method of further pre-training for domain adaptation.

(iii) **Semi-supervised learning by pseudo-labeling**
Semi-supervised learning provides another way to utilize unlabeled data when matched

labeled data are available. We may use the fine-tuned model to classify the unlabeled data and consider the classification results as ground truths (i.e., pseudo-labels), by assuming that the classifications are likely to be correct [8]. This provides a way to expand the supervised dataset. If self-supervised speech representation learning do really learn from unlabeled data, we may produce better pseudo-labels for the unlabeled data seen during pre-training. We therefore question the gain brought by the combination of self-supervised and semi-supervised approaches.

(iv) **Monolingual vs. cross-lingual pre-trained models**

When a monolingual pre-trained model is not available, the cross-lingual XLS-R [76] pre-trained models are possible alternatives because they should have learned universal speech representations from pre-training data in diverse languages. However, the amount of their pre-training data is not balanced across languages that the representations learned could bias towards the dominant languages, and it is also possible that the representations suffer from multilingual interference. We therefore compare the ASR performance resulted from monolingual and cross-lingual pre-trained models.

3.2 Datasets

The following describes the datasets we mainly use for the creation of ASR systems. A summary of their usages may be found in Appendix A.

3.2.1 Canopy: Cantonese Podcast and YouTube Shows

We collected data from the web to provide training data for creating a Cantonese wav2vec 2.0 model from scratch. To gain better control of audio quality, we pool data from selected podcast shows and a YouTube channel. For brevity, we hereafter refer to this set of data as *Canopy* (**Can**tonese **O**ral language data on **P**odcast and **Y**ouTube shows)[1]. A few of the sources involve scripted and unscripted monologues, while most of them provide conversational content, including casual chats, interviews and discussions. In particular, the YouTube channel we source hosts Skype call-ins. The data also exhibits a mix of near-field and far-field

[1] The word 'canopy' literally means 'shelter'. In ecology, it refers to living organisms that collectively sustain the habitat of a forest.

recording conditions. To the best of our knowledge, most of the speakers are not older adults.

The breakdown of the recordings obtained is given in Table 3.1. The podcast recordings are downloaded via URLs listed in the homepages of the shows on Apple Podcasts in late November 2022, and most of them are in the MP3 format with a sampling rate of 44.1 kHz. The YouTube audio recordings are downloaded via the yt-dlp[2] tool in the same period, and most of them are in the Opus format with a sampling rate of 48 kHz. All recordings are converted into 16 kHz FLAC.

Table 3.1: Breakdown of Canopy

Source	**No. of recordings**	**Recorded hours**	**Estimated no. of speakers***	**(Estimated) no. of main hosts**	
				Male	*Female*
Unlabeled data					
59 podcast shows	3989	2684.4	531†	94†	64†
1 YouTube channel	391	1571.4	882‡	4§	0§

* These include interviewees and callers. An attempt has been made to remove duplicates.
† These figures are obtained by manually checking (with little help from ChatGPT) the episode titles and descriptions, looking up the podcasters' social media pages and listening to some episodes.
‡ This is automatically obtained from clustering speaker embeddings based on the diarization results and will be explained later in Section 4.2.1.
§ These are exact figures.

3.2.2 CU-MARVEL

The CU-MARVEL corpus (**CU**HK-Cognitive Assessment Using **M**achine Le**a**rning Empowe**r**ed **V**oic**e** Ana**l**ysis) [88] is an ongoing effort that targets to collect longitudinal speech data from a thousand Cantonese-speaking older adults (aged over 60 years) in Hong Kong to assist the development of automated tools for screening neurocognitive disorders (NCDs) among the population. Due to the sensitive nature of the dataset, it is not available to the public.

We use the corpus to conduct ASR experiments that involve monolingual and cross-lingual pre-trained models, as well as experiments on further pre-training and semi-supervised learning. We study only the data obtained from the participants' first visits, which we hereafter refer to as the *baseline* data. The baseline data involves in-person conversational sessions in each of which an assessor guides an older adult participant to complete a list of NCD screen-

[2] https://github.com/yt-dlp/yt-dlp

ing tests in a *sound-proof* or *non-sound-proof* room. Every session is audio recorded with a sampling rate of 48 kHz and one channel using a smartphone. For our experiments, all recordings are converted into 16 kHz FLAC. On average, a session recording is more than 1.5 hours long, and a participant speaks for around 30% of the time. With budget and time constraints, manual transcriptions are done for selected screening tasks only. At the time of writing, the transcription work is still in progress.

We use both the labeled and unlabeled training data of the November 2022 release for system development, and use the labeled test data of the February 2023 release for evaluation purposes. The difference between the two releases is due to the amount of labeled data. The breakdown of these data is given in Table 3.2. Note that the unlabeled sessions are unsegmented. For the data studied in this thesis, the majority of participants are aged below 80 years, and the number of female participants is 25% more than that of male participants.

Table 3.2: Breakdown of CU-MARVEL baseline

(a) Breakdown by speaker role and gender

Split	**No. of sessions**	**Recorded hours**	**Manually labeled hours**		**No. of participants***	
			Assessors†	*Participants*	*Male*	*Female*
Partially labeled sessions						
Train (Nov 2022 ver.)	124	196.7	24.3	29.3	44	80
Test (Feb 2023 ver.)	46	72.3	13.0	14.8	20	26
Unlabeled sessions						
Train (Nov 2022 ver.)	288	436.8	n/a	n/a	139	149

* No participant appears in both the training and test sets.
† There are 8 assessors and all of them are female. They all appear in the training set, and only 5 of them appear in the test set.

(b) Breakdown of participants by gender and cognitive condition

Gender	**Condition**	**No. of participants**		
		Labeled train	*Labeled test*	*Unlabeled train*
Male	Healthy	18	9	96
	Minor NCD	20	9	41
	Major NCD	6	2	2
Female	Healthy	34	13	113
	Minor NCD	33	10	30
	Major NCD	13	3	6

3.2.3 LibriSpeech

LibriSpeech [21] is a publicly available ASR corpus of English read speech derived from audiobooks from the LibriVox project. It consists of labeled speech segments in the format of 16 kHz FLAC. The speakers are divided into two pools, *clean* and *other*, according to the recognition difficulty of their speech. Details of the training, development and test sets are provided in Table 3.3.

Table 3.3: Breakdown of LibriSpeech [21]

Split	**Recorded hours**	**No. of speakers**	
		Male	*Female*
Training sets			
train-clean-100	100.6	126	125
train-clean-360	363.6	482	439
train-other-500	496.7	602	564
Development sets			
dev-clean	5.4	20	20
dev-other	5.1	17	16
Test sets			
test-clean	5.4	20	20
test-other	5.3	16	17

This dataset is already studied in the original wav2vec 2.0 work, we mainly use it to benchmark against our in-domain experiments on further pre-training and semi-supervised learning.

3.3 Methods

3.3.1 Segmentation of Pre-training Data

Canopy

We assume the long recordings are cut into shorter segments (e.g., 30 seconds) by a voice activity detection (VAD) front-end before presenting to the speaker diarization module.

Without a diarization model and dataset that match the domain and language of our collected data, we resort to training an EEND model using simulated training data. We source utter-

ances for simulating conversations from the Common Voice project[3], which collects speech data from volunteers worldwide through the use of prompts texts crawled from the web. Specifically, we use the *zh-HK* and *yue* language subsets of Common Voice 11.0 in this work. Although Common Voice offers official training, development and test splits, we wish to obtain a larger pool of training speakers and rather use all data from the language subsets but excluding data from test speakers and reported clips[4,5]. This provides 160.5 hours of source utterances produced by 1871 speakers (of which 30.2% are male, 12.6% are female and the remaining unknown) for conversation simulations.

Conversations are simulated using the mixture simulation algorithm introduced in [83]. Given a set of speakers and the associated source utterances, the algorithm generates a mixture by (i) randomly generating speaker turn transitions of turn-hold, turn-switch, interruption and back-channel, (ii) generating reverberations for each speaker with randomly selected room impulse responses (RIRs), and (iii) eventually mixing the simulated result with background noise according to a signal-to-noise ratio (SNR). On top of these, we (iv) downsample all utterances of a speaker to 8 kHz with a certain probability since we found telephone audio in the real data, (v) apply speed perturbation at speaker level to artificially increase the number of speakers, (vi) apply volume perturbation at utterance level to make the model indifferent to slight volume differences of the same speaker across turns, and (vii) pad silences of random duration at the beginning and the end of the simulated conversation before mixing with noise to cover for imperfect VAD.

To cope with segments containing a variable number of speakers and exploit longer contexts, we adopt an EEND model with self-attention and encoder-decoder attention (SA-EEND + EDA) [44, 56] which embeds the log-mel spectrogram input using self-attention and decides the existence of speakers through encoder-decoder attention. Our implementation adopts icefall's[6] 'Reworked' Conformer as the backbone and is depicted in Figure 3.1. The inner working of the model is further elaborated as follows. After obtaining the sequence of embedded output $\mathbf{E}$ from the Conformer encoder, the LSTM encoder summarizes the entire sequence as two vectors: its final hidden state $\mathbf{h}_T^{\mathsf{enc}}$ and final cell state $\mathbf{c}_T^{\mathsf{enc}}$. To make the summarization process

[3] https://commonvoice.mozilla.org
[4] The 'speakers' are differentiated by their client IDs.
[5] The language subsets altogether come with 294 unique training speakers officially.
[6] https://github.com/k2-fsa/icefall

invariant to the speakers' order of presence, the embeddings $\mathbf{E}$ are chronologically shuffled before presented to the LSTM encoder. The two vectors are then used to initialize the LSTM decoder's hidden state $\mathbf{h}_0^{\mathsf{dec}}$ and cell state $\mathbf{c}_0^{\mathsf{dec}}$. The LSTM decoder outputs a sequence of 'attractors' $\mathbf{A}$, each of which is meant for a speaker, until the attractor vector outputted signifies that no more speaker may be found (the derived speaker existence probability is lower than a pre-defined threshold). The attractors $\mathbf{A}$ are compared to the embeddings $\mathbf{E}$ to determine the voicing probability of each speaker in each frame, which corresponds to an entry in the resultant speaker activity matrix.

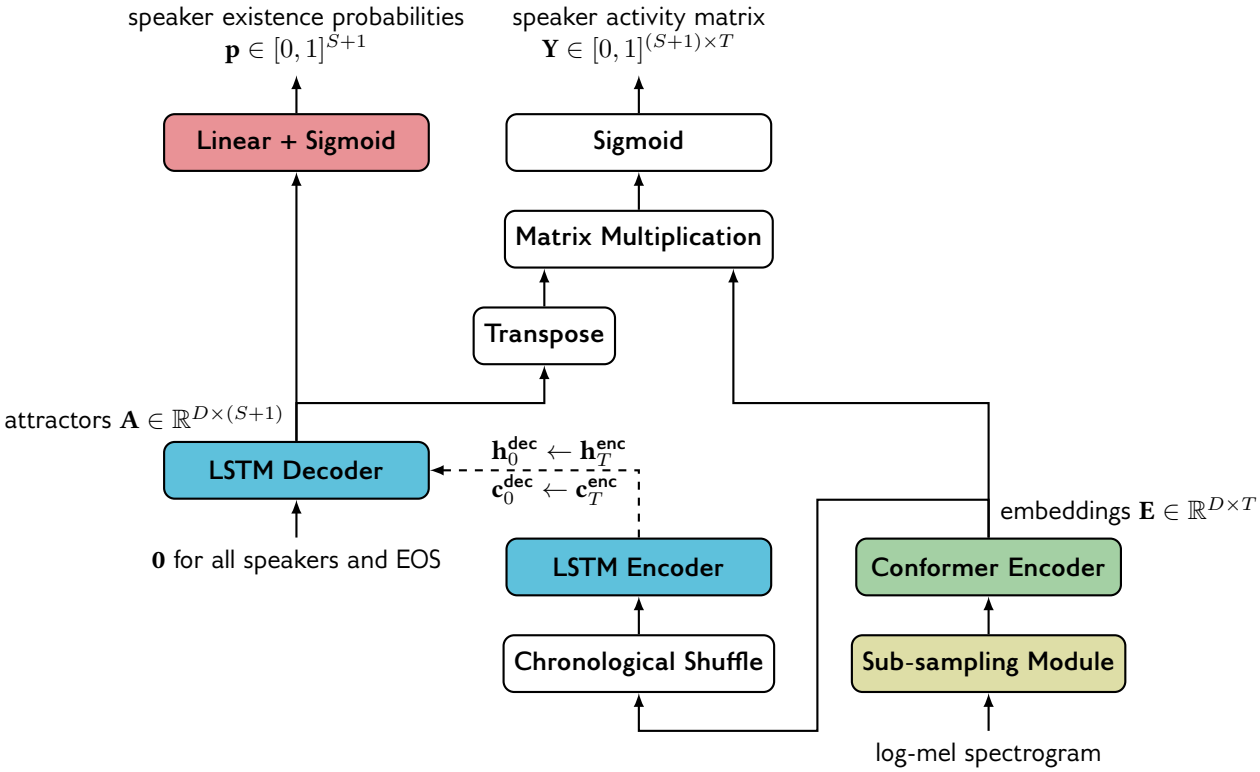
The EEND model is optimized for (i) a permutation-invariant loss for predicting speaker activities that corresponds to the least binary cross-entropy (BCE) loss attained between the model's predicted speaker activity matrix and any speaker-permuted version of the supervision matrix, and (ii) the BCE for predicting the existence of speakers.

CU-MARVEL

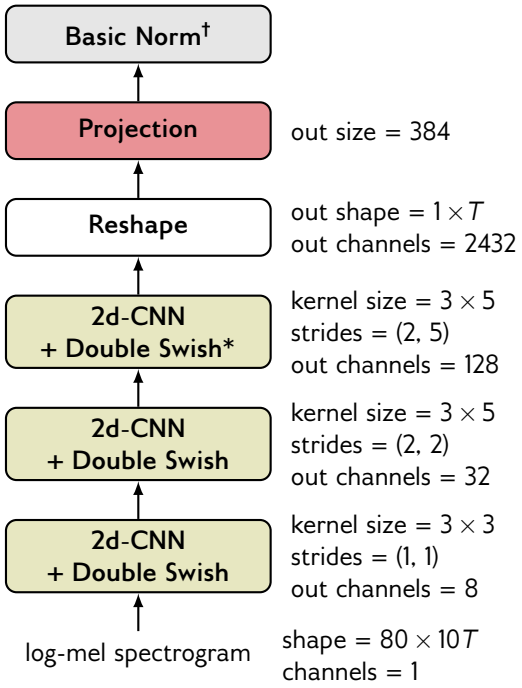
The diarization problem in CU-MARVEL may be deemed simpler in the sense that at most two speakers are involved at any instance of time. Moreover, in-domain labeled data are available. In view of these, we simply further train the pre-trained segmentation pipeline[7] from pyannote.audio[8] [53, 65]. The pipeline (i) performs local diarization of up to four speakers on every 5-second non-overlapping chunk of a given recording using a pre-trained SincNet-LSTM EEND model (which outputs a speaker activity matrix), (ii) extracts speaker embeddings for each locally identified speaker through a pre-trained speaker recognition model (an ECA-PA-TDNN [54] model provided by SpeechBrain[9] which optimized the additive angular margin loss), (iii) identifies speakers across chunks by clustering all obtained speaker embeddings using the agglomerative hierarchical clustering (AHC) algorithm and stitches the local diarization output accordingly.

With the in-domain training data, as per the pipeline defaults we optimize the pre-trained EEND model on 5-second chunks of at most two speakers using the permutation-invariant loss mentioned earlier and optimize the speech activity detection thresholds (voice activity onset, voice activity offset, minimum duration of a speech region, and minimum duration of a

[7] https://huggingface.co/pyannote/segmentation
[8] pyannote.audio is an open-source neural speaker diarization toolkit.
[9] https://huggingface.co/speechbrain/spkrec-ecapa-voxceleb



(a) Overall Structure



(b) Sub-sampling Module

* Applies the Swish function for twice.
† Scales input X by $1/\sqrt{\mathbb{E}[X^2]+\epsilon}$, where the mean is computed over the channel dimension and ϵ is a learnable parameter. See https://github.com/k2-fsa/icefall/blob/master/egs/librispeech/ASR/pruned_transducer_stateless2/scaling.py .

Figure 3.1: Architecture of the SA-EEND + EDA model adopted in this work

non-speech region) against the diarization error rate (DER) using the Tree-structured Parzen Estimator (TPE) algorithm.

3.3.2 wav2vec 2.0 (Further) Pre-training

The wav2vec 2.0 models we consider share a common architecture as depicted in Figure 3.2a. The embedding network is a multi-layer CNN, with a configuration specified in Figure 3.2b fixed across all model sizes. The context network is either a Transformer or Conformer. During pre-training or further pre-training, the NCE targets are quantized by a Gumbel vector quantizer. Assuming that the codebook learned during pre-training is already stable and any domain shift may be compensated by the context network, during further pre-training we freeze the weights of the embedding network and study only the adaptation's effects on the context network.

We will vary the amount of further pre-training data to study the data size requirement for domain adaptation. The following are the two base models mainly involved in this thesis.

XLS-R

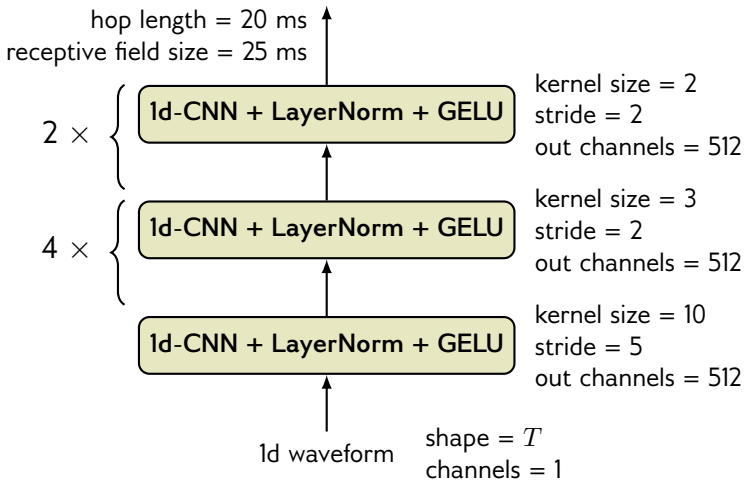
We consider only the 300M XLS-R model for further pre-training. The model is a CNN-Transformer with 24 Transformer layers, each with a dimension of 1024 and 16 attention heads. The model is pre-trained on 436K hours of speech data in 128 languges, which is mix of parliament speech (372K hours), Multilingual LibriSpeech read speech (50K hours), Common Voice read speech (7K hours), YouTube speech (6.6K hours) and phone conversations (1K hours) and mainly in European languages. In particular, 69.4K hours of the data are in English and 181 hours are in Cantonese[10]. We apply the model on both the English and Cantonese data for further pre-training and fine-tuning experiments.

Cantonese wav2vec 2.0 Model

Due to financial and time constraints, we are unable to pre-train from scratch a model that completely matches the configuration of the 300M XLS-R for comparison. To reduce the training time, we opt for a smaller model but with a better inductive bias. We consider a CNN-

[10] The Cantonese figure considers the ISO language codes of *zh-HK* and *yue*, please refer to the discussion on https://github.com/common-voice/common-voice/issues/2926 for their distinctions.

(a) Overall Structure



(b) Embedding Network

* Disabled during evaulation

Figure 3.2: Common architecture of the wav2vec 2.0 models adopted in this work

Conformer model with 12 Conformer layers, each with a dimension of 768 and 12 attention heads. This gives rise to 180M learnable parameters. We apply the resultant pre-trained model on the CU-MARVEL corpus only for further pre-training and fine-tuning experiments.

3.3.3 ASR Fine-tuning

The pre-trained models will serve as the backbone of encoder-only acoustic models for our ASR experiments on end-to-end training of acoustic models. Following the fine-tuning method adopted by the original wav2vec 2.0 paper, we remove the vector quantizer and the output projection layer from the pre-trained models. Afterwards, a newly initialized projection layer is appended to the pre-trained models to serve as the ASR head for fine-tuning. Except for the embedding network, the weights of the rest of the network are fine-tuned to optimize an ASR loss.

CU-MARVEL

We perform ASR training upon the pre-trained models using word-level labels and CTC loss. We adopt a phone lexicon-based system due to the small size of the available labeled data and the large character space of Cantonese Chinese. The transcripts of the labeled data are segmented into words by jieba[11], and our lexicon is based on the pronunciation dictionary from words.hk[12] and Jyutping Table[13]. Our romanization scheme, which is described in Appendix B, is based on the Jyutping scheme. A problem of adopting a phone-based lexicon is that some words come with multiple pronunciations, and we do not know which is actually referred to in an utterance. Moreover, imperfect word segmentation for the training transcripts adds further ambiguities. Therefore, we use k2's[14] implementation of the CTC loss which computes the loss over a finite-state automation (FSA)-based supervision graph which can include alternative pronunciations of a word-level transcription (i.e., a lattice).

During evaluation, we decode the labeled test data with beam search decoding and the use of weighted finite-state transducers (WFSTs). A 3-gram and a 4-gram LM are trained on the CU-MARVEL word-level training transcripts with modified Kneser-Ney smoothing in SRILM

[11] https://github.com/fxsjy/jieba
[12] https://words.hk/faiman/analysis
[13] https://github.com/lshk-org/jyutping-table
[14] https://github.com/k2-fsa/k2

[2]. The decoding graph is an HLG graph, which is a composition of the CTC topology, the lexicon, and the 3-gram LM. The decoded results are re-scored by the 4-gram LM with the method of whole-lattice re-scoring, which removes the LM scores imposed by the 3-gram LM and re-scores all paths in the decoding lattice with the 4-gram LM. During scoring, we remove unintelligible markers, words written in Jyutping and words not spoken in Cantonese or English in the reference and hypothesis transcripts before computing the edit distance. The character error rate (CER) we report in this thesis are *before* mapping filled pauses, which include 誒 *(eh)*, 啊 *(ah)*, 嗯 *(h'm)*, *um* and *em*, to one unit for scoring and *after* removing the 'unknown' words, which represent unintelligible speech, in both the reference and hypothesis transcripts[15].

LibriSpeech

To provide results that can be compared to the original wav2vec 2.0 paper, we perform ASR training upon the further pre-trained XLS-R model using the CTC loss and character-level labels, which include the 26 English alphabets, the apostrophe, and the space character.

During evaluation, decoding is done with lexicon-based decoding in Flashlight Text[16], which performs beam search decoding using a character-level trie and a word-level n-gram LM. The vocabulary is confined to the official lexicon[17] of LibriSpeech, and the LM used is LibriSpeech's official un-pruned 4-gram LM.

3.3.4 Semi-Supervised Learning

After obtaining ASR models from fine-tuning, we decode the unlabeled data with the use of pre-trained language models. The pseudo-labels are the one-best hypotheses found in the decoding lattices. Using the combination of the labeled data and the pseudo-labeled data as the supervised dataset, we train newly fine-tuned models upon the pre-trained models, instead of continue training the previous fine-tuned models.

[15] This is often done when evaluating the output of ASR systems.
[16] https://github.com/flashlight/text
[17] https://www.openslr.org/11

CU-MARVEL

The decoding of the unlabeled training data is done according to the decoding method and using the LM described in Section 3.3.3. The pseudo-labels are the one-best word-level hypotheses. The evaluation procedure is the same as in Section 3.3.3.

LibriSpeech

We deem train-clean-360 and train-other-500 as unlabeled. Psuedo-labels of word-level one-best hypotheses are generated using the decoding method in Section 3.3.3 but a different LM. The official LMs are trained from texts from the LibriSpeech LM corpus[18], which is based on 1478 books from Project Gutenberg. The corpus however involves books seen in the unlabeled data. To obtain a lower bound of the ASR performance achieved by semi-supervised learning, texts that come from the 178 books entailed in the unlabeled data are removed from the LM corpus and a new word-level 4-gram LM (which we later refer to as the 'dev' LM) for the purpose of pseudo-labeling is trained according to the official recipe. The evaluation procedure is the same as in Section 3.3.3.

[18] https://www.openslr.org/11

Chapter 4

Experiments and Discussions

This chapter records the settings and results of experiments conducted on the Cantonese CU-MARVEL and the English LibriSpeech corpora. As a large and fully transcribed corpus, LibriSpeech offers flexibility on the creation of scenarios with different amounts of pre-training and/or fine-tuning data and may therefore shed light on our findings on CU-MARVEL. Therefore, in the following, we will first report our findings on LibriSpeech.

4.1 LibriSpeech

In this section, LibriSpeech serves as our in-domain dataset. Note that we will use **train-all** to refer to its entire training data (*train-clean-100* + *train-clean-360* + *train-other-500*) and **train-clean-both** to denote its entire clean training data (*train-clean-100* + *train-clean-360*).

4.1.1 Further Pre-training Conditions

Experiment 1: Domain of Pre-training Data

In this experiment, we study the gain brought by further pre-training the 300M XLS-R on in-domain data, assuming a fixed set of fine-tuning data.

Using LibriSpeech's *train-all*, we further pre-train the 300M XLS-R model, except for the embedding network, for 80K steps, which amounts to 36 epochs. We apply FP16 training and the AdamW optimizer with a weight decay of 0.01. We use a learning rate of 2e-4 with a linear decay schedule and no warm-up.

We fine-tune (i) the vanilla XLS-R and (ii) the further pre-trained XLS-R on LibriSpeech's *train-clean-100* set. We fine-tune each model, except for the embedding network, for 50K steps, or 163 epochs. We apply FP16 training and the AdamW optimizer without weight decay. We use a learning rate of 3e-5 with a tri-stage schedule as adopted by [52], in which the first

10% of training steps are for warm-up and training the output layer only, the next 40% are for a constant learning rate, and the remaining steps are for linearly decaying the learning rate. We use a mask probability of 0.75, and a layer-drop probability of 0.1. We evaluate the fine-tuned models on LibriSpeech's development and test sets, and use the official word-level 4-gram and a beam size of 500 in decoding.

We trained the models on 6 NVIDIA RTX A6000 GPUs on a rented server. In this computing environment, further pre-training the XLS-R took 2 days and fine-tuning each of the two models took 1 day. The word error rate (WER) results are given in Table 4.1, and we quote results from [52] for their fine-tuning results stemming from a wav2vec 2.0 model pre-trained on the LibriSpeech training data (*train-all*) and another on 60K hours of LibriVox data in the Libri-Light [57] setup (*LibriLight-60K*). Note the XLS-R, *LibriSpeech-train-all* and LibriLight-60K models considered here share an identical CNN-Transformer architecture (a 7-layer CNN embedding network with 4M parameters and a 24-layer Transformer context network with 302M parameters) and a size of 306M[1]. In addition, we show in Table 4.2 that updating the parameters of XLS-R's CNN embedding network during pre-training do not give rise to significant performance difference in the downstream ASR by scaling the gradients back-propagated to the embedding network with different values (with 0.0 meaning no update at all). With a few exceptions in the *test-clean* set, significance tests show that the systems are not significantly different in committing word errors and treating speakers at significance level of $p < 0.01$ (see Appendix C). This suggests that the representations produced by the embedding network remain stable even if we enable it to update.

Compared to the vanilla XLS-R, further pre-training brings 2 to 4% relative improvement on the *clean* sets and 6 to 7% relative improvement on the noisier *other* sets. At significance level of $p < 0.01$, significance tests show that the two systems are significantly different and the further pre-trained XLS-R is the better one (see Appendix C). Even though the pre-training data of XLS-R includes 44.7K hours of English LibriVox data[2], the two XLS-R-based models

[1] This is a more precise figure. We simply refer to this figure as 300M elsewhere in accord to the convention of the original wav2vec 2.0 paper.

[2] Although both XLS-R and LibriLight-60K use audiobook data from LibriVox, the segmentation was done differently. The LibriVox data adopted by XLS-R follows from Multilingual LibriSpeech (MLS), which segmented LibriLight by considering the ASR decoding result by running inference on the data; whereas LibriLight-60K follows from Libri-Light, which simply segmented the data using VAD. Both derivatives of LibriVox have excluded the development and test speakers of LibriSpeech.

Table 4.1: LibriSpeech WER (%) resulted from wav2vec 2.0 models of the same 300M CNN-Transformer architecture and fine-tuned on *train-clean-100* but pre-trained on different datasets

Pre-training data	*dev-clean*	*dev-other*	*test-clean*	*test-other*
Monolingual, In-domain				
LibriSpeech-train-all [52]	2.3	5.7	2.8	6.0
LibriLight-60K [52]	**1.8**	**4.5**	**2.3**	**4.6** *
Cross-lingual, Cross-domain → Monolingual, In-domain				
XLS-R	2.73	6.90	2.95	7.53
+ further pre-train on *LibriSpeech-train-all*	2.67	6.48	2.83	6.98

* All inference results are produced by decoding with the 4-gram LM of the LibriSpeech LM corpus.

Table 4.2: LibriSpeech WER (%) resulted from different gradient multiplier values during further pre-training

Gradient multiplier	**No. of pre-train steps**	*dev-clean*	*dev-other*	*test-clean*	*test-other*
(baseline) 0.0	80K	**2.67**	6.48	2.83	**6.98**
0.1	80K	2.70	**6.46**	2.86	6.98
1.0	80K	2.68	6.49	**2.76**	7.07
1.0	160K	2.69	6.55	2.79	6.99

perform no better than models pre-trained on in-domain data from the very start, suggesting that XLS-R may suffer from domain or language interference and may not provide optimal in-domain performance. Having said that, the further pre-trained XLS-R underperforms LS-960 by less than 15%, and we deem the use of XLS-R and further pre-training for fast prototyping appropriate in scenarios with budget constraints.

Using centered kernel alignment (CKA) with linear kernel [45], we may measure the similarity between the representations outputted at each layer within and across models. 500 random utterances are drawn from the *dev-clean* set for conducting the analysis. As seen from the inter-CKA plot in Figure 4.1, further pre-training imposes the largest changes on middle and last few layers of the XLS-R. According to [73, 86], the canonical correlations between the layer representations and the word labels reaches the maximum in the middle layers and therefore we hypothesize that further pre-training allows in-domain words to be better represented. To confirm this, we apply the word-level analysis method adopted by [73] which computed the word-level average of speech representations and used projection-weighted canonical correlation analysis (PWCCA) [37] to measure the similarity between the averaged representations

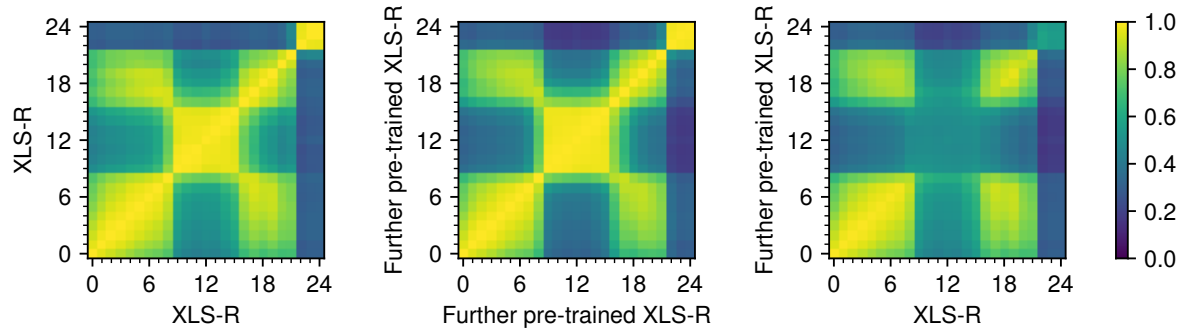


Figure 4.1: Frame-level CKA analysis of further pre-training on LibriSpeech. The axis values represent the layer numbers of the speech representation model specified on the axis labels, and the intensities refer to the CKA values. Layer 0 refers to the output of the CNN embedding network, and the remaining layer numbers refer to that of the Transformer context network. Left panel: the intra-CKA of XLS-R. Middle panel: the intra-CKA of XLS-R further pretrained on LibriSpeech. Right panel: the inter-CKA between the further pre-trained and the vanilla XLS-R.

and the aligned words' corresponding acoustically-grounded word embeddings (AGWE) [49][3] or Global Vectors for Word Representation (GloVe) [18] embeddings[4]. The results are plotted in Figure 4.2. The plots suggest that the similarity of the XLS-R representations and the word embeddings increases at the middle and last few layers after further pre-training, confirming our earlier hypothesis.[5]

Experiment 2: Mixes of In-Domain Data for Further Pre-training

In this experiment, we study the implications of different mixes of in-domain data for further pre-training the 300M XLS-R on the downstream ASR performance. We use the same pre-training and fine-tuning configurations as in Experiment 1, but vary the size and type (*clean* or *other*) of the pre-training data. We study four settings:

(i) **train-clean-both (460 hours)**: this follows from standard practice and biases towards clean data,

[3] https://dl.ttic.edu/librispeech_agwe_map.zip

[4] https://huggingface.co/stanfordnlp/glove/resolve/main/glove.840B.300d.zip

[5] We used CKA in comparing speech representations before and after further pre-training because it is variant to the scale of directions in the activation space [45] and lacks the sensitivity to perturbations affecting functional behavior [67], thus changes in CKA values must by induced by a considerable shift of feature importance. After roughly locating the layers of interest by CKA, we used PWCCA which is 17x slower (measured based on the PWCCA implementation of [37] and the CKA implementation of [45]) but with better sensitivity to uncover the fine-grained changes across layers.

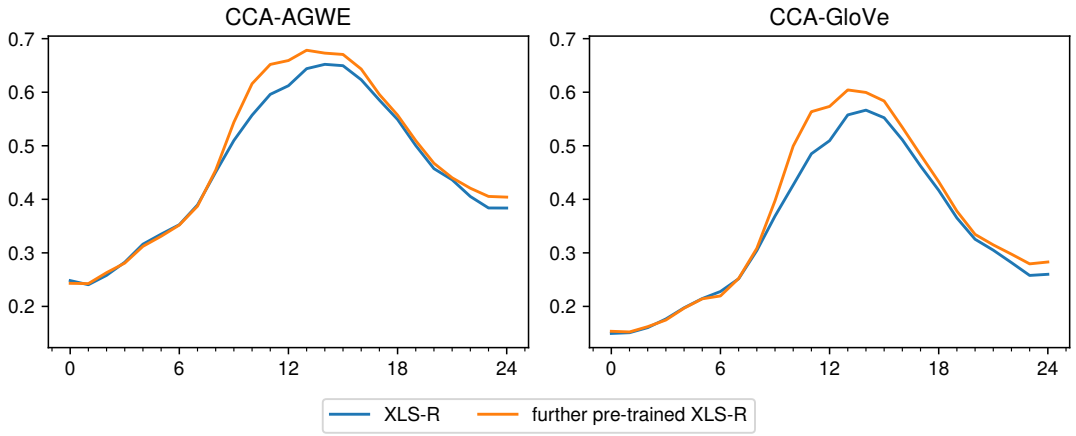


Figure 4.2: Word-level CCA analysis of further pre-training on LibriSpeech. The x-axis represents the layer numbers of a speech representation model, and the y-axis represents the correlation coefficients. Layer 0 refers to the output of the CNN embedding network, and the remaining layer numbers refer to that of the Transformer context network. Left panel: the PWCCA similarity with AGWE. Right panel: the PWCCA similarity with GloVe embeddings.

(ii) **half speakers (480 hours):** a reduced *train-all* that keeps only half of the speakers, which balances between clean and other noiser data but retaining less speakers,

(iii) **half per speaker (480 hours):** a reduced *train-all* that keeps only half of the utterances per speaker, which is more balanced between clean and other noisier data and

(iv) **train-all (960 hours):** the full *train-all*.

We pre-trained the models for 80K steps in the same computing environment of Experiment 1, and pre-training each model took 2 days. The fine-tuning results are given in Table 4.3.

Table 4.3: LibriSpeech WER (%) resulted from different mixes of further pre-training data

Pre-training data		*dev-clean*	*dev-other*	*test-clean*	*test-other*
(i)	train-clean-both (460h)	2.79	7.37	2.92	7.80
(ii)	half speakers (480h)	2.90	7.35	3.05	7.65
(iii)	half per speaker (480h)	2.73	6.57	2.90	7.12
(iv)	train-all (960h)	**2.67**	**6.48**	**2.83**	**6.98**

Comparing the (i) *train-clean-both* and the (iii) *half per speaker* setups, which share a similar size but cleanness, the results suggest that it is more important to include data from the *train-other-500* set during pre-training to improve the performance on the *dev-other* and *test-other* evaluation data. This is confirmed by significance tests at significant level $p < 0.01$ (see Appendix C). Keeping the same data size but different number of speakers, the (iii) *half per speaker* setup clearly outperforms the (ii) *half speakers* setup. On the other hand, maintaining

the same number of speakers, extending the training data from 480 hours in (iii) to 960 hours in (iv) gives no more than 2% relative improvement. This shows that having more distinct voices in the pre-training data is more important than having more data per speaker and is confirmed by significance tests at significant level $p < 0.01$ that (iii) and (iv) are not significantly different in committing word errors (see Appendix C).

4.1.2 ASR Fine-tuning Conditions

Experiment 3: Sizes of Fine-tuning Data

Here we study the implications of the size and type of fine-tuning data on the ASR performance resulted from XLS-R and the further pre-trained XLS-R. We add in the three training sets, *train-clean-100*, *train-clean-360*, and *train-other-500* one at a time to form the supervised training dataset for fine-tuning from the pre-trained model. We follow the other fine-tuning configurations in Experiment 1.

We trained each model for 50K, 160K and 320K steps for the three training data sizes respectively in the same computing environment of Experiment 1 and it took 1 day, 2 days and 4 days respectively. The results are given in Table 4.4.

Table 4.4: LibriSpeech WER (%) resulted from different mixes of fine-tuning data

Fine-tuning data	**Total size**	*dev-clean*	*dev-other*	*test-clean*	*test-other*
XLS-R					
train-clean-100	100 h	2.73	6.90	2.95	7.53
+ train-clean-360	460 h	2.45 (-10.3%)	5.97 (-13.5%)	2.63 (-10.8%)	6.34 (-15.8%)
+ train-other-500	960 h	**2.38** (-2.9%)	**5.20** (-12.9%)	**2.53** (-3.8%)	**5.49** (-13.4%)
Further pre-trained XLS-R (on train-all)					
train-clean-100	100 h	2.67	6.48	2.83	6.98
+ train-clean-360	460 h	2.41 (-9.7%)	5.94 (-8.3%)	2.58 (-8.8%)	6.16 (-11.7%)
+ train-other-500	960 h	**2.34** (-2.9%)	**5.27** (-11.3%)	**2.54** (-1.6%)	**5.52** (-10.4%)

* Figures in brackets refer to relative changes in WER.

It can be seen that the gain brought by an increased amount of supervised data during fine-tuning is smaller for the further pre-trained XLS-R than the vanilla XLS-R while the performance of the former is better than the latter in the low resource setups (100 hours and 460 hours), suggesting that further pre-training does help with mitigating low-resource scenarios. Even in the fully-supervised scenario of using all 960 hours of labeled training data, as

shown in Table 4.5, the wav2vec 2.0 models show, in general, an advantage over models that optimizes the ASR loss from the beginning.

Table 4.5: LibriSpeech WER (%) in the 960h supervised setup

Model	*dev-clean*	*dev-other*	*test-clean*	*test-other*
Non-pre-trained models				
icefall Reworked Conformer, joint BPE CTC-Attention (103M)*			2.59	5.54
CNN-Transformer CTC† [52]	1.8	5.4	2.6	5.8
wav2vec 2.0 models				
LS-960 [52]	**1.7**	**4.6**	**2.3**	**5.0**
XLS-R	2.38	5.20	2.53	5.49
Further pre-trained XLS-R	2.34	5.27	2.54	5.52

* Uses 80-dim log-mel spectrogram as its input. See https://github.com/k2-fsa/icefall/blob/master/egs/librispeech/ASR/RESULTS.md#librispeech-bpe-training-results-conformer-ctc-2 .
† Shares the same architecture as the wav2vec 2.0 models and uses raw waveform as the input.

Experiment 4: Semi-supervised Learning

Finally, we experiment with semi-supervised learning using pre-trained wav2vec 2.0 models. We consider the scenario that we are given *train-clean-100* as the labeled data whereas *train-clean-360* and *train-other-500* as the unlabeled data. We produce pseudo-labels using two pre-trained models fine-tuned on *train-clean-100*: (i) a fine-tuned LS-960 model[6] provided by Facebook (ii) and the further pre-trained XLS-R after fine-tuning obtained in Experiment 3. Afterwards, we restart the fine-tuning process on the pre-trained wav2vec 2.0 model (i.e., instead of the fine-tuned model) using the combination of the labeled *train-clean-100* and the pseudo-labeled *train-clean-360* and *train-other-500*. We perform the semi-supervised learning process for only one round.

Using each model fine-tuned on *train-clean-100*, we decoded the unlabeled data with a beam of 500 and trained the resultant semi-supervised model in the same computing environment of Experiment 1, and it took 4 days to produce a semi-supervised model. The results are given in Table 4.6.

It seems semi-supervised learning deteriorates the performance on the *dev-clean* set. The further pre-trained XLS-R provides significant improvement on the *other* sets in committing

[6] https://dl.fbaipublicfiles.com/fairseq/wav2vec/wav2vec_big_100h.pt

Table 4.6: LibriSpeech WER (%) in the 100h supervised + 860h pseudo-labeled setup

Model	*dev-clean*	*dev-other*	*test-clean*	*test-other*
Pseudo-labeled with the "dev" 4-gram LM, decoded with the official 4-gram LM				
LS-960				
train-clean-100 [52]	**2.3**	5.7	2.8	6.0
+ pseudo-labeling	2.53	**5.46**	**2.79**	**5.91**
Further pre-trained XLS-R				
train-clean-100 (Experiment 3)	2.67	6.48	2.83	6.98
+ pseudo-labeling	2.72	6.10	2.83	6.69

less word errors at significance level $p < 0.01$ (see Appendix C), but does not even outperform the 460-hour supervised setup in Table 4.4. We hypothesize that, although wav2vec 2.0 models provide strong baseline results, a strong LM is still needed to produce better pseudo-labels to assist semi-supervised learning.

4.2 CU-MARVEL

The following details our experiments on the Cantonese CU-MARVEL corpus. Before we set out our further pre-training and fine-tuning experiments, we begin with describing our speech segmentation procedure for preparing the pre-training data and our training configuration for creating our own Cantonese wav2vec 2.0 model.

4.2.1 Speech Segmentation

Canopy

Using the Common Voice dataset to generate training data for training an EEND model which helps in segmenting Canopy, the first step was to determine the time boundaries of speech activities found in each Common Voice utterance. To achieve this, we trained a GMM-HMM system with Speaker-Adaptive Training (SAT) in Kaldi [10] and force-aligned the audio and the transcripts to obtain alignment information at word-level. We proceeded to generating speaker mixtures using the mixture simulation algorithm described in Section 3.3.1 with the following settings:

(i) **Turn transitions:** for 78% of the time, the four turn transitions are generated with equal probabilities; for the remaining 22% of time, only interruption and back-channel

are generated with equal probabilities to simulate chaotic conversations. Transitions are generated until the mixture reaches a duration of truncnorm$(15, 10, 2, 30)$ (before silence padding)[7].

(ii) **Speaker-level reverberation**: no reverberation is generated.

(iii) **Mixture-level background noise:** with a probability of 0.5, the speaker mixture is further mixed with a clip of background noise randomly sampled from the MUSAN [25] and WHAM!48kHz noise [51] datasets according to an SNR uniformly sampled from the range $[10, 20]$.

(iv) **Speaker-level downsampling to 8K:** the downsampling probability is set to 0.2.

(v) **Speaker-level speed perturbation:** a speed perturbation factor is randomly drawn from $\{0.9, 1.0, 1, 1\}$.

(vi) **Speaker- and utterance-level volume perturbation**: A mean volume μ_s (in dB) is drawn from truncnorm$(-25, 5, -40, 20)$ for each speaker s in the mixture, and the volume of an utterance is drawn from truncnorm$(\mu_s, 2, -40, 20)$.

(vii) **Mixture-level silence padding:** two padding durations (in seconds) are sampled from $[0.0, 2.0]$ uniformly for padding to, respectively, the beginning and the end of the mixture.

At the end, a total of 884K mixtures which amount to 4.2K hours were created to simulate 1- to 4-speaker conversations, which respectively account for 18%, 35%, 30% and 18% of all mixtures. We used the mixtures to train an SA-EEND model, which accepts 80-dimensional log mel-filterbank coefficients as the input features. The frame shift is 10 ms and the frame length is 25 ms. The encoder consists of a 3-layer 2D-CNN sub-sampling module, which subsamples the input sequence by a factor of 10, 4 Conformer layers with a dimension of 384 and 6 heads, followed by 1 layer of uni-directional LSTM accepting chronologically shuffled encoded features; the decoder is a 1-layer uni-directional LSTM. The model has 20.7M parameters. Training the model for 20 epochs using 2 NVIDIA RTX A6000 GPUs on a rented server took 1 day.

[7] truncnorm(μ, σ, a, b) denotes a truncated Normal distribution defined over the interval $[a, b]$ and parameterized with mean μ, standard deviation σ, lower bound a and upper bound b.

Segmentation of the raw data was done with SpeechBrain's neural voice activity detection (VAD)[8] front-end and the said SA-EEND model. For the SA-EEND model, we use a voice activity onset of 0.4, a voice activity offset of 0.25 and a speaker existence threshold of 0.5. We merged neighboring segments of the same speaker that are at most 2 seconds apart and kept only the resulting segments that are at least 2 seconds and at most 40 seconds long. The segmentation procedure at the end produced 2.8K hours of speech segments readily for wav2vec 2.0 pre-training.

To estimate the number of speakers for the YouTube data in Canopy (see Table 3.1), we clustered speaker embeddings extracted from a pre-trained ECAPA-TDNN [54] model provided by SpeechBrain[9], which optimizes an additive angular margin loss. Removing duplicated speakers across recordings requires clustering all embeddings at once, which is intractable and we therefore clustered the embeddings in two stages. In the first stage, we operated at the recording level and performed spectral clustering on speaker embeddings pertaining to each recording and computed the average embedding per speaker. In the second stage, we performed agglomerative clustering on the average speaker embeddings collected from all recordings, with single linkage of cosine distance and a linkage distance threshold of 0.3.

CU NCD Screening Data: CU-MARVEL and CUHK-JCCOCC-MoCA

Using CU-MARVEL's baseline labeled training data, further training the pre-trained SincNet-LSTM segmentation model for 10 epochs using 4 NVIDIA Quadro RTX8000 GPUs on a private server took an hour. We repeated the same method on another Cantonese older adult speech corpus named CUHK-JCCOCC-MoCA [75] and segmented its unlabeled data to obtain slightly more data on top of CU-MARVEL's baseline data for pre-training. After combining the labeled data and the automatically segmented data from the two corpora, we obtained 503 hours of speech segments for wav2vec 2.0 pre-training. We hereafter refer to these two corpora collectively as **CU-NCD**.

[8] https://huggingface.co/speechbrain/vad-crdnn-libriparty
[9] https://huggingface.co/speechbrain/spkrec-ecapa-voxceleb

4.2.2 Cantonese wav2vec 2.0

We used the Canopy dataset described in Section 3.2.1 to train a CNN-Conformer (180M parameters, hereafter Cantonese Conformer) for 360K steps (or equivalently 115 epochs) using FP16 training and the AdamW optimizer with weight decay of 0.01. We set the learning rate to 3e-4 following a linear decay schedule with warm-up for 10% of the training steps. The mask probability was set to 0.65 and the mask length was set to 10. Pre-training using 6x NVIDIA RTX A6000 GPUs on a rented server took 9 days.

Experiment 1: Monolingual vs. Cross-lingual Representations

We compare the ASR performance resulted from monolingual and cross-lingual speech representations by fine-tuning them on the CU-MARVEL baseline labeled training data in this experiment.

We consider a setting without pre-trained representations that we have to train an acoustic model from scratch as the baseline. We trained a Kaldi nnet3 'chain' system that employed a 2d-CNN-TDNN model with 9.6M parameters and accepting log mel-spectrogram and online ivectors as inputs and optimized the lattice-free MMI (LF-MMI) objective with alignments obtained from a GMM-HMM system with speaker adpative training (SAT). 3x speed perturbation (0.9x, 1.0x and 1.1x) was applied on the training data to artificially increase the number of speakers. The model was trained for 6 epochs using natural gradient SGD [23] on 1 NVIDIA RTX 8000 GPU, taking half a day.

We fine-tuned the pre-trained (i) Cantonese Conformer and (ii) 300M XLS-R using the CTC loss. We applied the following configuration for fine-tuning: we froze the CNN module and fine-tuned the other parts of the model for 40K steps, or 190 epochs, using FP16 training and the AdamW optimizer without weight decay; the learning rate was set to 3e-5, with a tri-stage schedule in which the first 10% of training steps were for warm-up and training the output layer only, the next 40% were for a constant learning rate, and the remaining steps were for linearly decaying the learning rate; we used a mask probability of 0.75, and a layer-drop probability of 0.1. We trained each model on 2 NVIDIA RTX Quadro RTX 8000 GPUs in a private server, and training each took half a day.

The output of the ASR models are decoded with a search beam of 30. The character error rate (CER) results are given in Table 4.7 with breakdown by speaker role (*assessors* or *participants*)

and recording environment (*sound-proof* or *non-sound-proof*).

Table 4.7: CU-MARVEL CER (%) resulted from monolingual and cross-lingual speech representations

Model	**Venue**				**Overall**	
	Sound-proof		*Non-sound-proof*			
	Assrs.	*Parts.*	*Assrs.*	*Parts.*	*Assrs.*	*Parts.*
No pre-training, log-mel spec.						
Kaldi 'chain' 2d-CNN-TDNN	7.30	22.59	9.06	29.75	8.27	26.09
Monolingual & out-domain						
Cantonese Conformer	**4.67**	**15.07**	**6.46**	**21.24**	**5.65**	**18.08**
Cross-lingual & out-domain						
XLS-R (300M)	5.59	17.21	7.51	22.85	6.64	19.96

* Abbreviations: *Assrs.* – Assessors; *Parts.* – Participants.

Both the pre-trained representations provide performance surpassing Kaldi's. Moreover, the monolingual representations outperform the cross-lingual representations across speaker roles and venues. This suggests that ASR fine-tuning benefits from a pre-trained model which matches the language of the fine-tuning data. Comparing to the cross-lingual baseline, the advantages of the monolingual model, however, may be limited by the following factors. Perhaps attributed to the exclusion of older adult data during pre-training, the assessors enjoyed a greater reduction in overall CER than the older adult participants (a relative reduction of 14.87% vs. 9.43%). Another factor is due to environment robustness: the model may not be optimal for the non-sound-proof venue because the improvement of the Cantonese model in recognizing assessor speech in a non-sound-proof venue is less than that in a sound-proof venue (a relative reduction of 13.98% vs. 16.32%), and it is way worse for the participants (a relative reduction of 7.07% vs. 12.42%).

4.2.3 Further Pre-training Conditions

Experiment 2: Mixes of Further Pre-training Data

In this experiment, we study if (i) monolingual or (ii) cross-lingual speech representations pre-trained on out-domain data would affect the further pre-training performance on in-domain data. We froze the CNN layers, and trained only the context network (either Conformer or Transformer) and the quantization modules.

We first study a straightforward way to adapt the pre-trained speech representations by directly further pre-train the pre-trained model on the in-domain CU-NCD data (503 hours). We did this for 80K steps (which amounts to 96 epochs) using FP16 training and the AdamW optimizer with a weight decay of 0.01. We use a learning rate of 2e-4 with a linear decay schedule and no warm-up. The masking configuration is the same as in Experiment 1. This took us 7 days to complete the pre-training by using 3x NVIDIA Quadro RTX 8000 GPUs on a private server.

At the same time, we consider a two-stage further pre-training method for adapting XLS-R in which we first further pre-train on the mix of Canopy and CU-NCD (altogether 3.3K hours); in the next stage we further pre-train on the in-domain CU-NCD only. In the first stage, we trained for 100K steps (18 epochs) and in the second stage we re-initialized the learning rate schedule and trained for 40K steps (48 epochs). Obviously, this method iterates less on the in-domain data. The two-stage training altogether required 11 days using 3x NVIDIA Quadro RTX 8000 GPUs on a private server.

Table 4.8: CU-MARVEL CER (%) resulted from in-domain further pre-training

Model	**Venue**				**Overall**	
	Sound-proof		*Non-sound-proof*			
	Assrs.	*Parts.*	*Assrs.*	*Parts.*	*Assrs.*	*Parts.*
Baseline: no further pre-training (Table 4.7)						
(a) Cantonese Conformer	4.67	15.07	6.46	21.24	5.65	18.08
(b) XLS-R (300M)	5.59	17.21	7.51	22.85	6.64	19.96
One-stage further pre-training on CU-NCD (out-domain → in-domain)						
(c) Cantonese Conformer	3.75	14.42	4.94	19.36	4.40	16.83
(d) XLS-R (300M)	3.88	14.29	4.97	18.34	4.47	16.27
Two-stage further pre-training (cross-lingual → monolingual & cross-domain → in-domain)						
XLS-R (300M)						
→ (e) *on Canopy + CU-NCD*	4.17	15.30	5.39	19.97	4.84	17.58
→ (f) *on CU-NCD*	**3.61**	**13.35**	**4.80**	**17.42**	**4.26**	**15.34**

The fine-tuning results are given in Table 4.8. All further pre-trained models (c)–(f) superseded the monolingual and cross-lingual baselines we obtained in Experiment 1 and their improvement in committing less word errors is significant for both speaker roles at significance level $p < 0.01$ (see Appendix C). Moreover, the further pre-trained XLS-R models (d) and (f) outperform the further pre-trained monolingual model (c) on participants' speech. This seems

to disagree with our earlier results on LibriSpeech, which showed the monolingual models worked consistently better. However we must stress that the monolingual models in LibriSpeech are in-domain at the same time, while our monolingual here is out-domain. Here, we argue that the outperformance of XLS-R is due its larger model size (300M vs. the monolingual model's 180M). Our two-stage further pre-trained model (f) further improved upon the single stage model (d) and is the best model we can obtain. Therefore, we hypothesize that the two-stage further pre-training method allows the model to first adapt to the target language, then to the target domain in the matched language.

Comparing models (b) and (d), which both stemmed from the XLS-R, with single stage further pre-training the assessor speech shows an overall 32.61% relative improvement, whereas the participants' shows 18.50% relative. One reason why the accuracy of recognizing the assessor speech greatly improves is that the speakers are seen during training (but not the participants), and they speak clearly and use consistent wordings throughout different assessment sessions to give instructions to the participants. We also witness a significant reduction of CER when recognizing the participant speech in a non-sound-proof venue (19.73% relative), which is much more prominent than that for a sound-proof venue (16.94% relative). This suggests the simple method of further pre-training allows the model to gain environmental robustness without the need of sophisticated tricks. However, there still exists a large performance gap in recognizing speech in a non-sound-proof venue when compared to a sound-proof venue: the CER for the former environment is more than 20% higher than the latter. Comparing model (d) (one stage) and model (f) (two-stage), model (f) brings a relative gain of 4.66% and 5.71% on assessors and participants speech respectively. We hypothesize that the participants enjoyed a larger gain because their speech varies more than the assessors'.

Table 4.9: CU-MARVEL CER (%) resulted from different amounts of in-domain further pre-training data

Pre-training data	**Size**	**Train steps**	**Overall**	
			Assrs.	*Parts.*
CU-NCD (Table 4.8 model (d))	503 h	80K	**4.47**	**16.27**
CU-MARVEL-133	133 h	40K	4.86	17.58

We also ask the question of how much pre-training data is needed to provide performance gain for ASR fine-tuning. We consider single stage further pre-training and confine the pre-training data to cover only the utterances (both manually annotated and automatically segmented)

from the partially manually labeled sessions, which give rise to 133 hours of data (hereafter **CU-MARVEL-133**). We compare the resultant further pre-trained model to model (d) in Table 4.8, which is pre-trained on the full amount of the CU-NCD data (503 hours). The results are given in Table 4.9 and they suggest that pre-training this limited amount of data is still beneficial to the downstream ASR performance and outperforms the off-the-shelf pre-trained models significantly in committing less word errors at significance level $p < 0.01$ (see Appendix C).

Which factors are important in determining CER?

Table 4.10: CU-MARVEL CER (%) linear regression report

(a) Regression coefficients

Term	**Coef.**	**SE Coef.**	t**-value**	p**-value**	**VIF**
Constant	9.75	1.24	7.88	0.000	
Education years (standardized)	-1.718	0.716	-2.40	0.021	1.18
Age (standardized)	0.633	0.836	0.76	0.453	1.61
Group					
Minor NCD	1.73	1.42	1.22	0.229	1.15
Major NCD	7.50	2.65	2.83	0.007	1.60
Venue					
Non-sound-proof	3.66	1.36	2.70	0.010	1.09
Gender					
Male	5.35	1.39	3.85	0.000	1.12

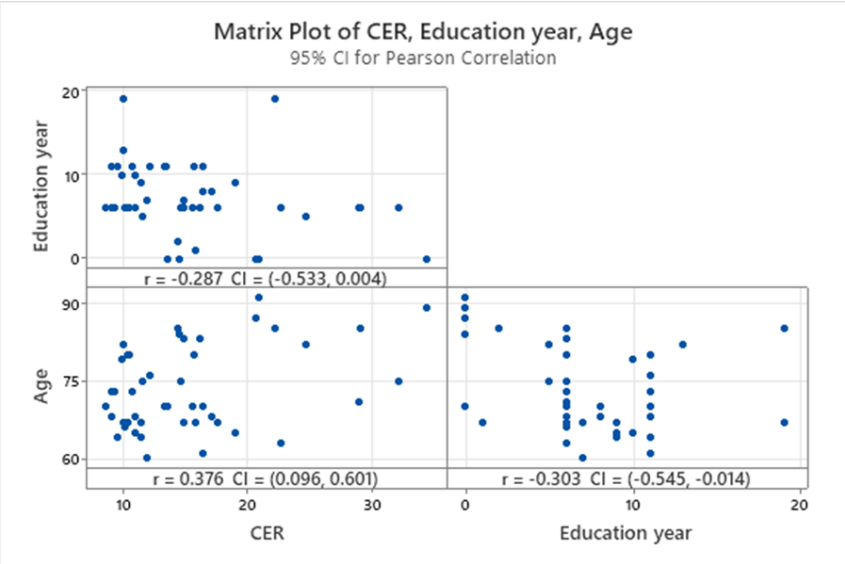
* Abbreviations: *Coef.* – coefficient, *SE Coef.* – standard error of the coefficient, *VIF* – variance inflation factor.

(b) Analysis of Variance (ANOVA)

Source	**DF**	**Adj. SS**	**Adj. MS**	F**-value**	p**-value**
Regression	6	1043.19	173.87	8.92	0.000
Education year	1	112.32	112.32	5.76	0.021
Age	1	11.18	11.18	0.57	0.453
Group	2	158.72	79.36	4.07	0.025
Venue	1	141.71	141.71	7.27	0.010
Gender	1	289.04	289.04	14.83	0.000
Error	39	760.14	19.49		
Total	45	1803.33			

* Abbreviations: *DF* – degree of freedom, *Adj. SS* – adjusted sum of squares, *Adj. MS* – adjusted mean squares.

To understand the relationship between the ASR performance on the participant speech and their demographics, as well as the recording environment, we fit a mutiple linear regression model to predict the participants' CER obtained in our two-stage further pre-trained XLS-R



* Abbreviations: *CI* – confidence interval.

Figure 4.3: CU-MARVEL correlation of CER (%), education and age

(f), from their gender, NCD classification, age, education years, and the recording condition. The regression report is given in Table 4.10. We also provide the correlation figures and plots in Figure 4.3. We set a significance level of $p < 0.02$. F-test of the least-squares fit shows the overall regression model is significant ($p = 0.000$). Although age shows a weak positive correlation with CER ($r = 0.376$), there is a lack of support that an increasing age gives rise to higher CER ($p = 0.453$). On the other hand, while receiving more education years shows a very weak negative correlation ($r = -0.287$) with CER, education is not a significant factor to explain CER ($p = 0.021$). The F-test shows the following factors are significant variables ($p < 0.02$): the participant being a man ($p = 0.000$), the participant having major NCD ($p = 0.007$), and the recording environment being a non-sound-proof one ($p = 0.012$) show a positive relationship with CER. Interestingly, minor NCD is not a significant factor on CER ($p = 0.229$). These preliminary results suggest major NCD (15% of labeled data) and male participant (35% of labeled data) speech are difficult to recognize, possibly because they are under-represented in the labeled data. Another plausible explanation could be that their speech are acoustically and linguistically more variable.

Effects of Overlapped Speech

We binned utterances according to their proportions of overlapping with another speaker's speech, and computed the CER (%) for each of the bins. In Figure 4.4, we plot CER as a

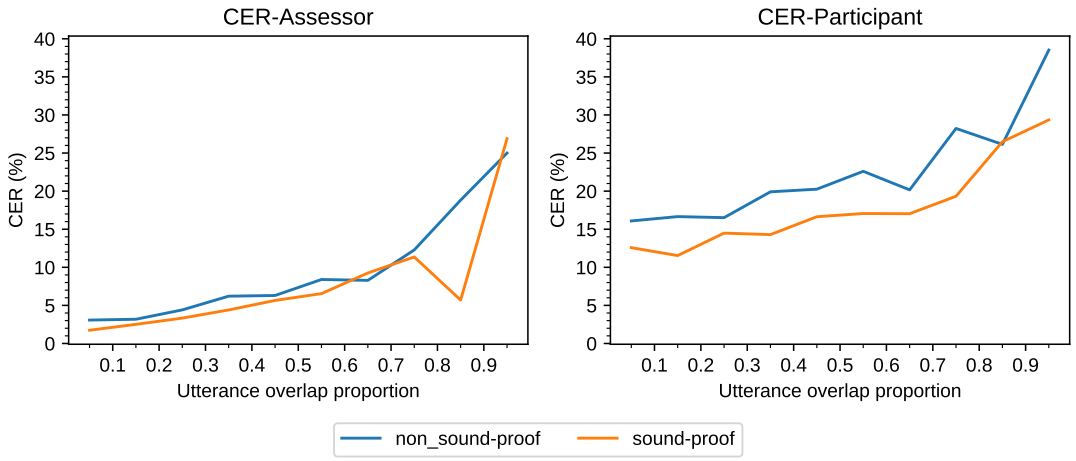


Figure 4.4: CU-MARVEL CER (%) and overlapped speech. Left panel: Assessors' speech CER (%) as a function of utterance overlap proportion. Right panel: Participants' speech CER (%) as a function of utterance overlap proportion.

function of utterance overlap proportion for each of the two speaker roles and for each of the two venue types. It can be seen that CER generally increases with the overlap proportion, suggesting additional measures are needed to recognize overlapped speech correctly.

4.2.4 ASR Fine-tuning Conditions

Experiment 3: Semi-supervised Learning

Pseudo labels based on the one-best hypothesis of whole-lattice rescoring were generated using the one-stage further pre-trained XLS-R (Table 4.8 model (d)) for the CU-NCD data, excluding the labeled training data of CU-MARVEL. Combining the labeled training data of CU-MARVEL and the pseudo-labeled CU-NCD data, we fine-tuned the one-stage further pre-trained XLS-R again using 160K steps (or 81 epochs) on 3x NVIDIA Quadro RTX 8000 GPUs.

Table 4.11: CU-MARVEL CER (%) resulted from semi-supervised learning

Model	**Venue**				**Overall**	
	Sound-proof		*Non-sound-proof*			
	Assrs.	*Parts.*	*Assrs.*	*Parts.*	*Assrs.*	*Parts.*
One-stage further pre-training on CU-NCD (out-domain → in-domain)						
XLS-R (300M) (Table 4.8 model (d))	3.88	14.29	4.97	18.34	4.47	16.27
+ pseudo-labeling	**3.73**	**14.07**	**4.88**	**17.83**	**4.36**	**15.91**

The end results are shown in Table 4.11. The method of semi-supervised learning by pseudo-labeling brings an overall reduction of around 2% in CER, and the improvement in committing

less character errors is significant at significance level $p < 0.01$ (see Appendix C). Since our system is not speaker-adaptive, we hypothesize the improvement is limited by the incorrect decoding result brought by the variability of spontaneous speech and overlapped speech. Another reason may be due to the employment of a language model that is not tailored for modeling disfluencies.

Chapter 5

Conclusions

Oriented towards the need of automatic transcripts for helping manual labor in developing a speech corpus named CU-MARVEL, this thesis explored the use of pre-trained self-supervised speech representation models to leverage in-domain data that are yet to be labeled and successfully improved the ASR performance in this low-resource setup. We collected additional speech data in Cantonese to avail speech representation learning in the language and contributed a Cantonese wav2vec 2.0 model. Adopting further pre-training and semi-supervised learning techniques, we were able to boost the in-domain performance of speech representations models that are pre-trained on out-domain data.

Our experiments on LibriSpeech showed that further pre-training is able to absorb some of the gain brought by fine-tuning a model that is not further pre-trained. This shows the importance of the method in a low-resource setup that is accompanied with a large amount of unlabeled data. In our experiments dedicated to CU-MARVEL, a significant improvement of ASR performance is observed on our adapted XLS-R model that was further pre-trained in two stages. In the first stage, we used a mix of in-domain and out-domain conversational data to adapt the 300M XLS-R to the target language of Cantonese. In the second stage, we limited the further pre-training data to those that are in-domain. The resultant ASR model after fine-tuning is found to improve upon the baseline models that were not further pre-trained and the adapted XLS-R that was further pre-trained on the in-domain data right away in one stage. Comparing to the monolingual and cross-lingual baselines, this model further pre-trained in two stages brought a relative improvement of 24.60% and 35.81% respectively on assessor speech, and 15.16% and 23.16% respectively on participant speech. These results suggest that a large cross-lingual pre-trained model may replace a small monolingual pre-trained model given the availability of abundant data in the target domain for further pre-training.

Possibly because the CU-MARVEL dataset is biased towards female (the male-to-female ratios are 0.8 for labeled and unlabeled training speakers as a whole and 0.55 for labeled training

speakers, and all assessors are female), through linear regression analysis we found our model discriminates against male speech. Considering the difficulties in recruiting participants, a future research direction would be to resolve this discrimination through speaker adaptations techniques or the incorporation of male older adult data obtained elsewhere.

Another research direction is related to the recognition of conversational data. Overlapped speech in the forms of interruption and backchanneling are inevitable in conversations, and is found in around 15% of the duration of CU-MARVEL's labeled data. We did not employ methods tailored for recognizing these kinds of speech. Moreover, although our pre-training data involved speaker mixtures and overlapped speech due to segmentation errors, their effect on the ASR performance is not well-understood. Devising a speech representation learning methodology that is aware of the acoustics and content of conversational speech *per se* (rather than simulated data as in, e.g., [85]) perhaps provides another interesting research direction.

Bibliography

[1] R. Favero and R. King, "Wavelet parameterization for speech recognition: Variations in translation and scale parameters," in *Proceedings of ICSIPNN '94. International Conference on Speech, Image Processing and Neural Networks*, 1994, 694–697 vol.2.

[2] A. Stolcke, "SRILM–an extensible language modeling toolkit," in *Seventh international conference on spoken language processing*, 2002.

[3] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06, Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, 2006, pp. 369–376.

[4] R. Schluter, I. Bezrukov, H. Wagner, and H. Ney, "Gammatone features and feature combination for large vocabulary speech recognition," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, vol. 4, 2007, pp. IV-649-IV–652.

[5] M. M. Goodwin, "The STFT, sinusoidal models, and speech modification," in *Springer Handbook of Speech Processing*, J. Benesty, M. M. Sondhi, and Y. A. Huang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 229–258.

[6] B. Kollmeier, T. Brand, and B. Meyer, "Perception of speech and sound," in *Springer Handbook of Speech Processing*, J. Benesty, M. M. Sondhi, and Y. A. Huang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 61–82.

[7] M. Wèolfel, *Distant speech recognition*, eng. Chichester, U.K: Wiley, 2008.

[8] X. Zhu and A. B. Goldberg, "Overview of semi-supervised learning," in *Introduction to Semi-Supervised Learning*. Cham: Springer International Publishing, 2009, pp. 9–19.

[9] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Y. W. Teh and M. Titterington, Eds., ser. Proceedings of Machine Learning Research, vol. 9, Chia Laguna Resort, Sardinia, Italy: PMLR, May 2010, pp. 297–304.

[10] D. Povey *et al.*, "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*, Hilton Waikoloa Village, Big Island, Hawaii, US: IEEE Signal Processing Society, Dec. 2011.

[11] A. Graves, "Sequence transduction with recurrent neural networks," eng, 2012.

[12] J. R. Hershey, S. J. Rennie, and J. Le Roux, "Factorial models for noise robust speech recognition," in *Techniques for Noise Robustness in Automatic Speech Recognition*. John Wiley & Sons, Ltd, 2012, ch. 12, pp. 311–345.

[13] R. Singh, B. Raj, and T. Virtanen, "The basics of automatic speech recognition," eng, in *Techniques for Noise Robustness in Automatic Speech Recognition*, 1st ed., Chichester, UK: Wiley, 2012, pp. 7–30.

[14] R. M. Stern and N. Morgan, "Features based on auditory physiology and perception," in *Techniques for Noise Robustness in Automatic Speech Recognition*. John Wiley & Sons, Ltd, 2012, ch. 8, pp. 193–227.

[15] P. Swietojanski, A. Ghoshal, and S. Renals, "Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR," in *2012 IEEE Spoken Language Technology Workshop (SLT)*, 2012, pp. 246–251.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds., vol. 26, Curran Associates, Inc., 2013.

[17] D. Palaz, R. Collobert, and M. Magimai-Doss, "Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks," in *Proc. Interspeech 2013*, 2013, pp. 1766–1770.

[18] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543.

[19] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for LVCSR," in *Proc. Interspeech 2014*, 2014, pp. 890–894.

[20] P. Golik, Z. Tüske, R. Schlüter, and H. Ney, "Convolutional neural networks for acoustic modeling of raw time signal in LVCSR," in *Proc. Interspeech 2015*, 2015, pp. 26–30.

[21] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2015, pp. 5206–5210.

[22] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Interspeech 2015*, 2015, pp. 3214–3218.

[23] D. Povey, X. Zhang, and S. Khudanpur, *Parallel training of dnns with natural gradient and parameter averaging*, 2015.

[24] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform CLDNNs," in *Proc. Interspeech 2015*, 2015, pp. 1–5.

[25] D. Snyder, G. Chen, and D. Povey, *Musan: A music, speech, and noise corpus*, 2015.

[26] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," eng, in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2016, pp. 4960–4964.

[27] P. Ghahremani, V. Manohar, D. Povey, and S. Khudanpur, "Acoustic modelling from the signal domain using CNNs," in *Interspeech 2016*, 2016, pp. 3434–3438.

[28] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu, *Exploring the limits of language modeling*, 2016.

[29] D. Povey *et al.*, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Proc. Interspeech 2016*, 2016, pp. 2751–2755.

[30] Z. Zhu, J. H. Engel, and A. Hannun, "Learning multiscale features directly from waveforms," in *Interspeech 2016*, 2016, pp. 1305–1309.

[31] W. Dai, C. Dai, S. Qu, J. Li, and S. Das, "Very deep convolutional neural networks for raw waveforms," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 421–425.

[32] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 4835–4839.

[33] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

[34] A. Zeyer, E. Beck, R. Schlüter, and H. Ney, "CTC in the context of generalized full-sum HMM training," in *Proc. Interspeech 2017*, 2017, pp. 944–948.

[35] H. Hadian, H. Sameti, D. Povey, and S. Khudanpur, "End-to-end speech recognition using lattice-free MMI," in *Proc. Interspeech 2018*, 2018, pp. 12–16.

[36] Z. Ma and M. Collins, "Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 3698–3707.

[37] A. S. Morcos, M. Raghu, and S. Bengio, "Insights on representational similarity in neural networks with canonical correlation," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18, Montréal, Canada: Curran Associates Inc., 2018, pp. 5732–5741.

[38] A. v. d. Oord, Y. Li, and O. Vinyals, *Representation learning with contrastive predictive coding*, 2018.

[39] D. Povey *et al.*, "Semi-orthogonal low-rank matrix factorization for deep neural networks," in *Proc. Interspeech 2018*, 2018, pp. 3743–3747.

[40] Z. Tüske, R. Schlüter, and H. Ney, "Acoustic modeling of speech waveform based on multi-resolution, neural network signal processing," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4859–4863.

[41] N. Zeghidour, N. Usunier, I. Kokkinos, T. Schaiz, G. Synnaeve, and E. Dupoux, "Learning filterbanks from raw speech for phone recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 5509–5513.

[42] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, "An unsupervised autoregressive model for speech representation learning," in *Proc. Interspeech 2019*, 2019, pp. 146–150.

[43] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional Transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[44] Y. Fujita, N. Kanda, S. Horiguchi, Y. Xue, K. Nagamatsu, and S. Watanabe, "End-to-end neural speaker diarization with self-attention," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 296–303.

[45] S. Kornblith, M. Norouzi, H. Lee, and G. Hinton, "Similarity of neural network representations revisited," in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Sep. 2019, pp. 3519–3529.

[46] P. von Platen, C. Zhang, and P. Woodland, "Multi-span acoustic modelling using raw waveform signals," in *Proc. Interspeech 2019*, 2019, pp. 1393–1397.

[47] M. Ravanelli and Y. Bengio, *Speech and speaker recognition from raw waveform with sincnet*, 2019.

[48] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," in *Proc. Interspeech 2019*, 2019, pp. 3465–3469.

[49] S. Settle, K. Audhkhasi, K. Livescu, and M. Picheny, "Acoustically grounded word embeddings for improved acoustics-to-word speech recognition," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 5641–5645.

[50] D. Wang, X. Wang, and S. Lv, "An overview of end-to-end automatic speech recognition," *Symmetry*, vol. 11, no. 8, 2019.

[51] G. Wichern *et al.*, "Wham!: Extending speech separation to noisy environments," in *Proc. Interspeech*, Sep. 2019.

[52] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 12 449–12 460.

[53] H. Bredin *et al.*, "pyannote.audio: Neural building blocks for speaker diarization," in *ICASSP 2020, IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2020.

[54] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification," in *Proc. Interspeech 2020*, 2020, pp. 3830–3834.

[55] A. Gulati *et al.*, "Conformer: Convolution-augmented Transformer for speech recognition," in *Proc. Interspeech 2020*, 2020, pp. 5036–5040.

[56] S. Horiguchi, Y. Fujita, S. Watanabe, Y. Xue, and K. Nagamatsu, "End-to-end speaker diarization for an unknown number of speakers with encoder-decoder based attractors," in *Proc. Interspeech 2020*, 2020, pp. 269–273.

[57] J. Kahn *et al.*, "Libri-light: A benchmark for asr with limited or no supervision," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7669–7673.

[58] K. Kawakami, L. Wang, C. Dyer, P. Blunsom, and A. van den Oord, "Learning robust and multilingual speech representations," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, Online: Association for Computational Linguistics, Nov. 2020, pp. 1182–1192.

[59] S. Ling and Y. Liu, *DeCoAR 2.0: Deep contextualized acoustic representations with vector quantization*, 2020.

[60] S. Ling, Y. Liu, J. Salazar, and K. Kirchhoff, "Deep contextualized acoustic representations for semi-supervised speech recognition," eng, in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Ithaca: IEEE, 2020, pp. 6429–6433.

[61] A. T. Liu, S.-w. Yang, P.-H. Chi, P.-c. Hsu, and H.-y. Lee, "Mockingjay: Unsupervised speech representation learning with deep bidirectional Transformer encoders," eng, in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Ithaca: IEEE, 2020, pp. 6419–6423.

[62] T. Parcollet, M. Morchid, and G. Linarès, "E2E-SincNet: Toward fully end-to-end speech recognition," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7714–7718.

[63] V. Pratap, Q. Xu, A. Sriram, G. Synnaeve, and R. Collobert, "MLS: A large-scale multilingual dataset for speech research," *ArXiv*, vol. abs/2012.03411, 2020.

[64] R. Xiong *et al.*, "On layer normalization in the Transformer architecture," in *Proceedings of the 37th International Conference on Machine Learning*, ser. ICML'20, JMLR.org, 2020.

[65] H. Bredin and A. Laurent, "End-to-end speaker segmentation for overlap-aware resegmentation," in *Proc. Interspeech 2021*, 2021.

[66] Y.-A. Chung, Y. Belinkov, and J. Glass, "Similarity analysis of self-supervised speech representations," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 3040–3044.

[67] F. Ding, J.-S. Denain, and J. Steinhardt, "Grounding representation similarity through statistical testing," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34, Curran Associates, Inc., 2021, pp. 1556–1568.

[68] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021.

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

[70] M. W. Lam, J. Wang, C. Weng, D. Su, and D. Yu, "Raw waveform encoder with multi-scale globally attentive locally recurrent networks for end-to-end speech recognition," in *Proc. Interspeech 2021*, 2021, pp. 316–320.

[71] D. Ma, N. Ryant, and M. Liberman, "Probing acoustic representations for phonetic properties," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 311–315.

[72] P. Martin, *Speech acoustic analysis*, eng. London, England: ISTE Ltd, 2021.

[73] A. Pasad, J.-C. Chou, and K. Livescu, "Layer-wise analysis of a self-supervised speech representation model," in *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2021, pp. 914–921.

[74] S. Sadhu *et al.*, "wav2vec-C: A self-supervised model for speech representation learning," in *Proc. Interspeech 2021*, 2021, pp. 711–715.

[75] S. S. Xu, M.-W. Mak, K. H. Wong, H. Meng, and T. C. Kwok, "Speaker turn aware similarity scoring for diarization of speech-based cognitive assessments," eng, in *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, APSIPA, 2021, pp. 1299–1304.

[76] A. Babu *et al.*, "XLS-R: Self-supervised cross-lingual speech representation learning at scale," in *Proc. Interspeech 2022*, 2022, pp. 2278–2282.

[77] A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli, "data2vec: A general framework for self-supervised learning in speech, vision and language," in *Proceedings of the*

39th International Conference on Machine Learning, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., ser. Proceedings of Machine Learning Research, vol. 162, PMLR, Jul. 2022, pp. 1298–1312.

[78] S. Chen *et al.*, "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.

[79] C.-C. Chiu, J. Qin, Y. Zhang, J. Yu, and Y. Wu, "Self-supervised learning with random-projection quantizer for speech recognition," in *Proceedings of the 39th International Conference on Machine Learning*, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., ser. Proceedings of Machine Learning Research, vol. 162, PMLR, 17–23 Jul 2022, pp. 3915–3924.

[80] K. Choi and E. J. Yeo, *Opening the black box of wav2vec feature encoder*, 2022.

[81] C. Meng, J. Ao, T. Ko, M. Wang, and H. Li, "CoBERT: Self-supervised speech representation learning through code representation learning," 2022.

[82] A. R. Vaidya, S. Jain, and A. Huth, "Self-supervised models of audio effectively explain human cortical responses to speech," in *Proceedings of the 39th International Conference on Machine Learning*, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., ser. Proceedings of Machine Learning Research, vol. 162, PMLR, Jul. 2022, pp. 21927–21944.

[83] N. Yamashita, S. Horiguchi, and T. Homma, "Improving the naturalness of simulated conversations for end-to-end neural diarization," in *Proc. The Speaker and Language Recognition Workshop (Odyssey 2022)*, 2022, pp. 133–140.

[84] C. J. Cho, P. Wu, A. Mohamed, and G. K. Anumanchipalli, "Evidence of vocal tract articulation in self-supervised learning of speech," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[85] M. Fazel-Zarandi and W.-N. Hsu, "Cocktail HuBERT: Generalized self-supervised pre-training for mixture and single-source speech," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[86] A. Pasad, B. Shi, and K. Livescu, "Comparative layer-wise analysis of self-supervised speech models," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[87] R. Sanabria, H. Tang, and S. Goldwater, “Analyzing acoustic word embeddings from pre-trained self-supervised speech models,” in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[88] H. Meng *et al.*, “Integrated and enhanced pipeline system to support spoken language analytics for screening neurocognitive disorders,” in *Interspeech 2023*, forthcoming.

List of Publications

1. **Ranzo C. F. Huang** and Brian Mak, "wav2vec 2.0 ASR for Cantonese-speaking older adults in a clinical setting," in *Interspeech 2023*, forthcoming.

2. Helen Meng, Brian Mak, Man-Wai Mak, Helene Fung, Xianmin Gong, Timothy Kwok, Xunying Liu, Vincent C. T. Mok, Patrick Wong, Jean Woo, Xixin Wu, Ka Ho Wong, Sean Shensheng Xu, Naijun Zheng, **Ranzo C. F. Huang**, Jiawen Kang, Xiaoquan Ke, Junan Li, Jinchao Li, and Yi Wang, "Integrated and enhanced pipeline system to support spoken language analytics for screening neurocognitive disorders," in *Interspeech 2023*, forthcoming.

Appendix A

Dataset Usages

A number of datasets have been used for different purposes in this work. For better clarity, their usages are summarized below.

Table A.1: Summary of dataset usages

Dataset	**Main style**	**Usage**[*]			
		SD	*AP*	*ASR*	*LM*
Noise datasets					
MUSAN		•			
WHAM!48kHz noise		•			
Speech datasets, audio-only					
Canopy (this work)	spontaneous		•		
Speech datasets, with paired audio and texts					
Common Voice 11.0 (*zh-HK & yue*)	read	•			
CU-MARVEL	spontaneous	•	•	•	•
CUHK-JCCOCC-MoCA	spontaneous		•	∘[†]	
LibriSpeech	read		•	•	
Text datasets					
LibriSpeech LM Corpus	written				•

[*] Abbreviations: *SD* – speaker diarization; *AP* – audio pre-training (wav2vec 2.0); *ASR* – automatic speech recognition; *LM* – language modeling

[†] completely pseudo-labeled (i.e., labels are not used, even if there are any)

Appendix B

Cantonese Romanization Scheme

The following tables demonstrate the *Jyutping* Cantonese Romanization Scheme[1] developed by the Linguistic Society of Hong Kong (LSHK). The scheme used in formatting the lexicon of this work is given alongside.

Table B.1: Cantonese romanization scheme

(a) Syllable onsets

Phone		**Example**		
LSHK	*This work*	*Character*	*LSHK*	*This work*
(null initial)		啊	aa3	aa3
b	b	巴	**b**aa1	**b** aa1
p	p	怕	**p**aa3	**p** aa3
m	m	罵	**m**aa6	**m** aa6
f	f	花	**f**aa1	**f** aa1
d	d	打	**d**aa2	**d** aa2
t	t	他	**t**aa1	**t** aa1
n	n	拿	**n**aa4	**n** aa4
l	l	罅	**l**aa3	**l** aa3
g	g	家	**g**aa1	**g** aa1
k	k	卡	**k**aa1	**k** aa1
ng	ng	牙	**ng**aa4	**ng** aa4
h	h	霞	**h**aa4	**h** aa4
gw	gw	瓜	**gw**aa1	**gw** aa1
kw	kw	誇	**kw**aa1	**kw** aa1
w	w	娃	**w**aa1	**w** aa1
z	z	渣	**z**aa1	**z** aa1
c	c	岔	**c**aa3	**c** aa3
s	s	沙	**s**aa1	**s** aa1
j	j	也	**j**aa5	**j** aa5

[1] https://lshk.org/jyutping-scheme

(b) Syllable nuclei

Phone		**Example**		
LSHK	*This work*	*Character*	*LSHK*	*This work*
aa	aa + tone	沙	saa1	s **aa1**
i	i + tone	啲	di1	d **i1**
u	u + tone	風	fung1	f **u1** ng1
e	e + tone	車	ce1	c **e1**
o	o + tone	多	do1	d **o1**
yu	yu + tone	喘	cyun2	c **yu2** n2
oe	oe + tone	鋸	goe3	g **oe3**
a	a + tone	人	jan4	j **a4** n4
eo	eo + tone	水	seoi2	s **eo2** i2

(c) Syllable codas

Phone		**Example**		
LSHK	*This work*	*Character*	*LSHK*	*This work*
p	p	濕	sap1	s a1 **p**
t	t	突	dat6	d a6 **t**
k	k	劈	pek3	p e3 **k**
m	m + tone	減	gaam2	g aa2 **m2**
n	n + tone	陳	can4	c a4 **n4**
ng	ng + tone	生	saang1	s aa1 **ng1**
i	i + tone	隨	ceoi4	c eo4 **i4**
u	u + tone	豆	dau2	d a2 **u2**

(d) Syllabic nasals

Phone		**Example**		
LSHK	*This work*	*Character*	*LSHK*	*This work*
m	m + tone	唔	m4	**m4**
ng	ng + tone	吳	ng4	**ng4**

(e) Tones

Tone			**Example**		
Wong Shik-Ling	*LSHK*	*This work*	*Character*	*LSHK*	*This work*
1	1	1	加	gaa1	g **aa1**
2	2	2	錢	cin2	c **i2 n2**
3	3	3	轉	zyun3	z **yu3 n3**
4	4	4	牛	ngau4	ng **a4 u4**
5	5	5	奶	naai5	n **aa5 i5**
6	6	6	味	mei6	m **e6 i6**
7	1	1	的	dik1	d **i1** k
8	3	3	確	kok3	k **o3** k
9	6	6	值	zik6	z **i6** k

Appendix C

Significance Tests

The following records the statistical test reports of SCTK for comparing the outputs of different ASR systems. The reports are rearranged to avoid redundancy.[1]

Explanation

```
                Composite Report of All Significance Tests
                              For the  Test

                        Test Name                                Abbrev.
    ------------------------------------------------------       -------
          Matched Pair Sentence Segment (Word Error)               MP
    Signed Paired Comparison (Speaker Word Error Rate (%))         SI
      Wilcoxon Signed Rank (Speaker Word Error Rate (%))           WI

These significance tests are all two-tailed tests with the null hypothesis
that there is no performance difference between the two systems.

The first column indicates if the test finds a significant difference
at the level of p=0.05.  It consists of '~' if no difference is
found at this significance level.  If a difference at this level is
found, this column indicates the system with the higher value on the
performance statistic utilized by the particular test.

The second column specifies the minimum value of p for which the test
finds a significant difference at the level of p.

The third column indicates if the test finds a significant difference
at the level of p=0.001 ("***"), at the level of p=0.01, but not
p=0.001 ("**"), or at the level of p=0.05, but not p=0.01 ("*").

A test finds significance at level p if, assuming the null hypothesis,
the probability of the test statistic having a value at least as
extreme as that actually found, is no more than p.
```

[1] The descriptions of the tests may be found at https://web.archive.org/web/20071011230835/http://nist.gov/speech/tests/sigtests/sigtests.htm.

Note: for CU-MARVEL, the phrase ‘word error’ in the above description translates as ‘character error’.

Abbreviations used in naming the ASR systems

```
  fp - further pre-trained
  lr - learning rate
grad - gradient multiplier
 spk - speaker
  pl - pseudo-labeling
```

LibriSpeech Experiment 1

For *dev-clean*:

Test Abbrev.			xlsr_baseline		xlsr_fp				Test Abbrev.
MP	xlsr_baseline			xlsr_fp	<0.001	***		MP
SI				xlsr_fp	<0.001	***		SI
WI				xlsr_fp	<0.001	***		WI
MP		xlsr_fp						MP
SI								SI
WI								WI

For *dev-other*:

Test Abbrev.			xlsr_baseline		xlsr_fp				Test Abbrev.
MP	xlsr_baseline			xlsr_fp	<0.001	***		MP
SI				xlsr_fp	<0.001	***		SI
WI				xlsr_fp	<0.001	***		WI
MP		xlsr_fp						MP
SI								SI
WI								WI

For *test-clean*:

Test Abbrev.			xlsr_baseline		xlsr_fp				Test Abbrev.
MP	xlsr_baseline			xlsr_fp	<0.001	***		MP
SI				xlsr_fp	<0.001	***		SI
WI				xlsr_fp	<0.001	***		WI
MP		xlsr_fp						MP
SI								SI
WI								WI

For *test-other*:

Test Abbrev.			xlsr_baseline		xlsr_fp				Test Abbrev.
MP	xlsr_baseline			xlsr_fp	<0.001	***			MP
SI				xlsr_fp	<0.001	***			SI
WI				xlsr_fp	<0.001	***			WI
MP		xlsr_fp							MP
SI									SI
WI									WI

Reports for ablation study

For *dev-clean*:

Test Abbrev.			xlsr_fp		xlsr_fp_lr5e-5			xlsr_fp_grad_0.1			xlsr_fp_grad_1.0			xlsr_fp_grad_1.0_steps160k			Test Abbrev.
MP				~	0.472			~	0.234		~	0.490		~	0.516		MP
SI		xlsr_fp		xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	SI
WI				~	0.617		~	0.529		~	0.582		~	0.503		WI
MP							~	0.857		~	0.928		~	0.968		MP
SI		xlsr_fp_lr5e-5					xlsr_fp_lr5e-5	<0.001	***	xlsr_fp_lr5e-5	<0.001	***	xlsr_fp_lr5e-5	<0.001	***	SI
WI							~	0.529		~	0.968		~	0.936		WI
MP										~	0.772		~	0.826		MP
SI		xlsr_fp_grad_0.1								xlsr_fp_grad_1.0	<0.001	***	~	1.000		SI
WI										~	0.834		~	0.779		WI
MP													~	0.968		MP
SI		xlsr_fp_grad_1.0											xlsr_fp_grad_1.0_steps160k	<0.001	***	SI
WI													~	0.881		WI
MP																MP
SI		xlsr_fp_grad_1.0_steps160k														SI
WI																WI

Friedman Two-way Analysis of Variance by Ranks

Ho: Testing the hypothesis that all recognizers are the same

Reject if
X2_r > X2 of 5% df4 (9.490)

adjustment = 0.936
X2_r = 1.474

ANALYSIS:

The test statistic X2_r shows that at the 95% confidence
interval, the recognition systems are not significantly
different.

Further, the probablity of there being a difference is
between 10% to 20%.

COMPARISON MATRIX: Comparing All Systems
Using a Multiple Comparison Test

	xlsr_fp	xlsr_fp_lr5e-5	xlsr_fp_grad_1.0_steps160k	xlsr_fp_grad_1.0	xlsr_fp_grad_0.1
xlsr_fp		same	same	same	same
xlsr_fp_lr5e-5			same	same	same
xlsr_fp_grad_1.0_steps160k				same	same
xlsr_fp_grad_1.0					same
xlsr_fp_grad_0.1					

For *dev-other*:

Test Abbrev.			xlsr_fp	xlsr_fp_lr5e-5		xlsr_fp_grad_0.1		xlsr_fp_grad_1.0		xlsr_fp_grad_1.0_steps160k		Test Abbrev.
MP		xlsr_fp		~ 0.960		~ 0.660		~ 0.881		~ 0.347		MP
SI			xlsr_fp	<0.001	***	xlsr_fp_grad_0.1	0.003 **	xlsr_fp	<0.001 ***	xlsr_fp	0.003 **	SI
WI				~ 0.497		~ 0.810		~ 0.704		~ 0.603		WI
MP		xlsr_fp_lr5e-5				~ 0.734		~ 0.920		~ 0.384		MP
SI						xlsr_fp_grad_0.1	<0.001 ***	xlsr_fp_lr5e-5	<0.001 ***	xlsr_fp_lr5e-5	0.003 **	SI
WI						~ 0.741		~ 0.603		~ 0.726		WI
MP		xlsr_fp_grad_0.1						~ 0.646		~ 0.211		MP
SI								xlsr_fp_grad_0.1	0.002 **	xlsr_fp_grad_0.1	0.003 **	SI
WI								~ 0.873		~ 0.441		WI
MP		xlsr_fp_grad_1.0								~ 0.424		MP
SI										xlsr_fp_grad_1.0	0.002 **	SI
WI										~ 0.509		WI
MP		xlsr_fp_grad_1.0_steps160k										MP
SI												SI
WI												WI

Friedman Two-way Analysis of Variance by Ranks

Ho: Testing the hypothesis that all recognizers are the same

Reject if
X2_r > X2 of 5% df4 (9.490)

adjustment = 0.968
X2_r = 1.139

ANALYSIS:

The test statistic X2_r shows that at the 95% confidence interval, the recognition systems are not significantly different.

Further, the probablity of there being a difference is between 10% to 20%.

COMPARISON MATRIX: Comparing All Systems
Using a Multiple Comparison Test

	xlsr_fp_grad_0.1	xlsr_fp	xlsr_fp_lr5e-5	xlsr_fp_grad_1.0	xlsr_fp_grad_1.0_steps160k
xlsr_fp_grad_0.1		same	same	same	same
xlsr_fp			same	same	same
xlsr_fp_lr5e-5				same	same
xlsr_fp_grad_1.0					same
xlsr_fp_grad_1.0_steps160k					

For *test-clean*:

Test Abbrev.			xlsr_fp		xlsr_fp_lr5e-5			xlsr_fp_grad_0.1			xlsr_fp_grad_1.0			xlsr_fp_grad_1.0_steps160k			Test Abbrev.
MP		xlsr_fp		~	0.230		~	0.373		~	0.067		~	0.298			MP
SI			xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	xlsr_fp_grad_1.0	<0.001	***	xlsr_fp_grad_1.0_steps160k	<0.001	***		SI
WI				~	0.280		~	0.194		~	0.078		~	0.289			WI
MP		xlsr_fp_lr5e-5					~	0.522	xlsr_fp_grad_1.0	0.002	**	xlsr_fp_grad_1.0_steps160k	0.032	*		MP
SI							~	1.000	xlsr_fp_grad_1.0	<0.001	***	xlsr_fp_grad_1.0_steps160k	<0.001	***		SI
WI							~	0.682	xlsr_fp_grad_1.0	0.002	**	xlsr_fp_grad_1.0_steps160k	0.032	*		WI
MP		xlsr_fp_grad_0.1							xlsr_fp_grad_1.0	0.018	*		~	0.107			MP
SI									xlsr_fp_grad_1.0	<0.001	***	xlsr_fp_grad_1.0_steps160k	<0.001	***		SI
WI									xlsr_fp_grad_1.0	0.026	*		~	0.114			WI
MP		xlsr_fp_grad_1.0											~	0.497			MP
SI													~	1.000			SI
WI													~	0.603			WI
MP		xlsr_fp_grad_1.0_steps160k															MP
SI																	SI
WI																	WI

Friedman Two-way Analysis of Variance by Ranks

Ho: Testing the hypothesis that all recognizers are the same

Reject if
X2_r > X2 of 5% df4 (9.490)

adjustment = 0.946
X2_r = 12.322

ANALYSIS:

The test statistic X2_r shows, with 95% confidence, that at
least one recognition system is significantly different.

Further, the probablity of there being a difference is
between 98% to 99%.

COMPARISON MATRIX: Comparing All Systems
Using a Multiple Comparison Test

	xlsr_fp_grad_1.0	xlsr_fp_grad_1.0_steps160k	xlsr_fp	xlsr_fp_grad_0.1	xlsr_fp_lr5e-5
xlsr_fp_grad_1.0		same	same	xlsr_fp_grad_1.0	xlsr_fp_grad_1.0
xlsr_fp_grad_1.0_steps160k			same	same	xlsr_fp_grad_1.0_steps160k
xlsr_fp				same	same
xlsr_fp_grad_0.1					same
xlsr_fp_lr5e-5					

For *test-other*:

Test Abbrev.		xlsr_fp	xlsr_fp_lr5e-5	xlsr_fp_grad_0.1	xlsr_fp_grad_1.0	xlsr_fp_grad_1.0_steps160k	Test Abbrev.
MP	xlsr_fp		- 0.171	- 0.976	- 0.219	- 0.841	MP
SI			xlsr_fp <0.001 ***	xlsr_fp_grad_0.1 <0.001 ***	xlsr_fp <0.001 ***	xlsr_fp_grad_1.0_steps160k 0.003 **	SI
WI			- 0.114	- 0.803	- 0.144	- 0.841	WI
MP	xlsr_fp_lr5e-5			- 0.165	- 0.826	- 0.263	MP
SI				xlsr_fp_grad_0.1 <0.001 ***	xlsr_fp_lr5e-5 0.002 **	xlsr_fp_grad_1.0_steps160k <0.001 ***	SI
WI				- 0.162	- 0.741	- 0.180	WI
MP	xlsr_fp_grad_0.1				- 0.222	- 0.865	MP
SI					xlsr_fp_grad_0.1 <0.001 ***	xlsr_fp_grad_0.1 0.003 **	SI
WI					- 0.093	- 0.936	WI
MP	xlsr_fp_grad_1.0					- 0.322	MP
SI						xlsr_fp_grad_1.0_steps160k <0.001 ***	SI
WI						- 0.134	WI
MP	xlsr_fp_grad_1.0_steps160k						MP
SI							SI
WI							WI

Friedman Two-way Analysis of Variance by Ranks

Ho: Testing the hypothesis that all recognizers are the same

Reject if
X2_r > X2 of 5% df4 (9.490)

adjustment = 0.956
X2_r = 9.065

ANALYSIS:

The test statistic X2_r shows that at the 95% confidence
interval, the recognition systems are not significantly
different.

Further, the probablity of there being a difference is
between 90% to 95%.

COMPARISON MATRIX: Comparing All Systems
Using a Multiple Comparison Test

	xlsr_fp_grad_0.1	xlsr_fp_grad_1.0_steps160k	xlsr_fp	xlsr_fp_lr5e-5	xlsr_fp_grad_1.0
xlsr_fp_grad_0.1		same	same	xlsr_fp_grad_0.1	same
xlsr_fp_grad_1.0_steps160k			same	xlsr_fp_grad_1.0_steps160k	same
xlsr_fp				xlsr_fp	same
xlsr_fp_lr5e-5					same
xlsr_fp_grad_1.0					

LibriSpeech Experiment 2

For *dev-clean*:

Test Abbrev.			xlsr_fp	xlsr_fp_half_per_spk		xlsr_fp_half_spks			xlsr_fp_460h			Test Abbrev.
MP		xlsr_fp		~	0.327	xlsr_fp	0.026	*	~	0.246		MP
SI				xlsr_fp	<0.001 ***	xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	SI
WI				~	0.131	xlsr_fp	0.036	*	~	0.177		WI
MP		xlsr_fp_half_per_spk				~	0.180		~	0.779		MP
SI						xlsr_fp_half_per_spk	<0.001	***	xlsr_fp_half_per_spk	<0.001	***	SI
WI						~	0.250		~	0.689		WI
MP		xlsr_fp_half_spks							~	0.363		MP
SI									xlsr_fp_460h	<0.001	***	SI
WI									~	0.211		WI
MP		xlsr_fp_460h										MP
SI												SI
WI												WI

Friedman Two-way Analysis of Variance by Ranks

Ho: Testing the hypothesis that all recognizers are the same

Reject if
X2_r > X2 of 5% df3 (7.820)

adjustment = 0.930
X2_r = 7.234

ANALYSIS:

The test statistic X2_r shows that at the 95% confidence
interval, the recognition systems are not significantly
different.

Further, the probablity of there being a difference is
between 90% to 95%.

COMPARISON MATRIX: Comparing All Systems
Using a Multiple Comparison Test

	xlsr_fp	xlsr_fp_half_per_spk	xlsr_fp_460h	xlsr_fp_half_spks
xlsr_fp		same	same	xlsr_fp
xlsr_fp_half_per_spk			same	same
xlsr_fp_460h				same
xlsr_fp_half_spks				

For *dev-other*:

Test Abbrev.			xlsr_fp	xlsr_fp_half_per_spk		xlsr_fp_half_spks		xlsr_fp_460h		Test Abbrev.
MP		xlsr_fp		~ 0.110		xlsr_fp <0.001 ***		xlsr_fp <0.001 ***		MP
SI				xlsr_fp 0.002 **		xlsr_fp <0.001 ***		xlsr_fp <0.001 ***		SI
WI				~ 0.276		xlsr_fp <0.001 ***		xlsr_fp <0.001 ***		WI
MP		xlsr_fp_half_per_spk				xlsr_fp_half_per_spk <0.001 ***		xlsr_fp_half_per_spk <0.001 ***		MP
SI						xlsr_fp_half_per_spk <0.001 ***		xlsr_fp_half_per_spk <0.001 ***		SI
WI						xlsr_fp_half_per_spk 0.005 **		xlsr_fp_half_per_spk <0.001 ***		WI
MP		xlsr_fp_half_spks						xlsr_fp_half_spks <0.001 ***		MP
SI								xlsr_fp_half_spks <0.001 ***		SI
WI								xlsr_fp_half_spks 0.009 **		WI
MP		xlsr_fp_460h								MP
SI										SI
WI										WI

Friedman Two-way Analysis of Variance by Ranks

Ho: Testing the hypothesis that all recognizers are the same

Reject if
X2_r > X2 of 5% df3 (7.820)

adjustment = 0.930
X2_r = 25.300

ANALYSIS:

The test statistic X2_r shows, with 95% confidence, that at
least one recognition system is significantly different.

Further, the probablity of there being a difference is
greater that 99.9%.

COMPARISON MATRIX: Comparing All Systems
Using a Multiple Comparison Test

	xlsr_fp	xlsr_fp_half_per_spk	xlsr_fp_half_spks	xlsr_fp_460h
xlsr_fp		same	xlsr_fp	xlsr_fp
xlsr_fp_half_per_spk			xlsr_fp_half_per_spk	xlsr_fp_half_per_spk
xlsr_fp_half_spks				same
xlsr_fp_460h				

For *test-clean*:

Test Abbrev.			xlsr_fp	xlsr_fp_half_per_spk		xlsr_fp_half_spks			xlsr_fp_460h			Test Abbrev.
MP		xlsr_fp		~	0.131	xlsr_fp	0.017	*	xlsr_fp	0.014	*	MP
SI				xlsr_fp	<0.001 ***	xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	SI
WI				~	0.401	~	0.051		xlsr_fp	0.015	*	WI
MP		xlsr_fp_half_per_spk				~	0.342		~	0.242		MP
SI						xlsr_fp_half_per_spk	<0.001	***	xlsr_fp_half_per_spk	<0.001	***	SI
WI						~	0.208		~	0.250		WI
MP		xlsr_fp_half_spks							~	0.803		MP
SI									xlsr_fp_half_spks	<0.001	***	SI
WI									~	0.818		WI
MP		xlsr_fp_460h										MP
SI												SI
WI												WI

Friedman Two-way Analysis of Variance by Ranks

Ho: Testing the hypothesis that all recognizers are the same

Reject if
X2_r > X2 of 5% df3 (7.820)

adjustment = 0.943
X2_r = 6.772

ANALYSIS:

The test statistic X2_r shows that at the 95% confidence
interval, the recognition systems are not significantly
different.

Further, the probablity of there being a difference is
between 90% to 95%.

COMPARISON MATRIX: Comparing All Systems
Using a Multiple Comparison Test

	xlsr_fp	xlsr_fp_half_per_spk	xlsr_fp_half_spks	xlsr_fp_460h
xlsr_fp		same	xlsr_fp	xlsr_fp
xlsr_fp_half_per_spk			same	same
xlsr_fp_half_spks				same
xlsr_fp_460h				

For *test-other*:

Test Abbrev.			xlsr_fp	xlsr_fp_half_per_spk		xlsr_fp_half_spks			xlsr_fp_460h			Test Abbrev.
MP		xlsr_fp		~	0.066	xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	MP
SI				xlsr_fp	<0.001 ***	xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	SI
WI				~	0.136	xlsr_fp	<0.001	***	xlsr_fp	<0.001	***	WI
MP		xlsr_fp_half_per_spk				xlsr_fp_half_per_spk	0.007	**	xlsr_fp_half_per_spk	<0.001	***	MP
SI						xlsr_fp_half_per_spk	<0.001	***	xlsr_fp_half_per_spk	<0.001	***	SI
WI						xlsr_fp_half_per_spk	0.010	*	xlsr_fp_half_per_spk	<0.001	***	WI
MP		xlsr_fp_half_spks							xlsr_fp_half_spks	<0.001	***	MP
SI									xlsr_fp_half_spks	<0.001	***	SI
WI									xlsr_fp_half_spks	0.004	**	WI
MP		xlsr_fp_460h										MP
SI												SI
WI												WI

Friedman Two-way Analysis of Variance by Ranks

Ho: Testing the hypothesis that all recognizers are the same

Reject if
X2_r > X2 of 5% df3 (7.820)

adjustment = 0.979
X2_r = 29.034

ANALYSIS:

The test statistic X2_r shows, with 95% confidence, that at
least one recognition system is significantly different.

Further, the probablity of there being a difference is
greater that 99.9%.

COMPARISON MATRIX: Comparing All Systems
Using a Multiple Comparison Test

	xlsr_fp	xlsr_fp_half_per_spk	xlsr_fp_half_spks	xlsr_fp_460h
xlsr_fp		same	xlsr_fp	xlsr_fp
xlsr_fp_half_per_spk			same	xlsr_fp_half_per_spk
xlsr_fp_half_spks				xlsr_fp_half_spks
xlsr_fp_460h				

LibriSpeech Experiment 4

For *dev-clean*:

Test Abbrev.			xlsr_fp		xlsr_fp_pl			libri_fp_pl			Test Abbrev.
MP		xlsr_fp			~	0.472		libri_fp_pl	0.005	**	MP
SI				xlsr_fp	<0.001	***		libri_fp_pl	<0.001	***	SI
WI					~	0.407			~	0.390	WI
MP		xlsr_fp_pl						libri_fp_pl	<0.001	***	MP
SI								libri_fp_pl	<0.001	***	SI
WI									~	0.267	WI
MP		libri_fp_pl									MP
SI											SI
WI											WI

For *dev-other*:

Test Abbrev.			xlsr_fp		xlsr_fp_pl			libri_fp_pl			Test Abbrev.
MP		xlsr_fp			xlsr_fp_pl	<0.001	***	libri_fp_pl	<0.001	***	MP
SI					xlsr_fp_pl	<0.001	***	libri_fp_pl	0.003	**	SI
WI					xlsr_fp_pl	0.002	**		~	0.734	WI
MP		xlsr_fp_pl						libri_fp_pl	<0.001	***	MP
SI								libri_fp_pl	0.003	**	SI
WI									~	0.952	WI
MP		libri_fp_pl									MP
SI											SI
WI											WI

For *test-clean*:

Test Abbrev.			xlsr_fp		xlsr_fp_pl			libri_fp_pl			Test Abbrev.
MP		xlsr_fp			~	0.904		~	0.395		MP
SI				xlsr_fp	<0.001	***		~	1.000		SI
WI					~	0.826		~	0.841		WI
MP		xlsr_fp_pl						~	0.430		MP
SI								~	1.000		SI
WI								~	0.711		WI
MP		libri_fp_pl									MP
SI											SI
WI											WI

For *test-other*:

Test Abbrev.			xlsr_fp	xlsr_fp_pl			libri_fp_pl				Test Abbrev.
MP		xlsr_fp		xlsr_fp_pl	<0.001	***	libri_fp_pl	<0.001	***		MP
SI				xlsr_fp_pl	<0.001	***	libri_fp_pl	<0.001	***		SI
WI				xlsr_fp_pl	0.016	*		~	0.280		WI
MP		xlsr_fp_pl					libri_fp_pl	<0.001	***		MP
SI							libri_fp_pl	<0.001	***		SI
WI								~	0.453		WI
MP		libri_fp_pl									MP
SI											SI
WI											WI

CU-MARVEL Experiment 2

For *assessors*:

Test Abbrev.		canto	canto_fp		Test Abbrev.
MP	canto		canto_fp <0.001	***	MP
SI			~ 0.062		SI
WI			canto_fp 0.043	*	WI
MP	canto_fp				MP
SI					SI
WI					WI

Test Abbrev.		xlsr	xlsr_fp		xlsr_fp2		Test Abbrev.
MP	xlsr		xlsr_fp <0.001	***	xlsr_fp2 <0.001	***	MP
SI			~ 0.062		~ 0.062		SI
WI			xlsr_fp 0.043	*	xlsr_fp2 0.043	*	WI
MP	xlsr_fp				xlsr_fp2 <0.001	***	MP
SI					~ 0.062		SI
WI					xlsr_fp2 0.043	*	WI
MP	xlsr_fp2						MP
SI							SI
WI							WI

Test Abbrev.		xlsr	xlsr_fp		xlsr_fp_133h		Test Abbrev.
MP	xlsr		xlsr_fp <0.001	***	xlsr_fp_133h <0.001	***	MP
SI			~ 0.062		~ 0.062		SI
WI			xlsr_fp 0.043	*	xlsr_fp_133h 0.043	*	WI
MP	xlsr_fp				xlsr_fp <0.001	***	MP
SI					~ 0.062		SI
WI					xlsr_fp 0.043	*	WI
MP	xlsr_fp_133h						MP
SI							SI
WI							WI

For *participants*:

Test Abbrev.		canto	canto_fp		Test Abbrev.
MP	canto		canto_fp	<0.001 ***	MP
SI			canto_fp	<0.001 ***	SI
WI			canto_fp	<0.001 ***	WI
MP	canto_fp				MP
SI					SI
WI					WI

Test Abbrev.		xlsr	xlsr_fp		xlsr_fp2		Test Abbrev.
MP	xlsr		xlsr_fp	<0.001 ***	xlsr_fp2	<0.001 ***	MP
SI			xlsr_fp	<0.001 ***	xlsr_fp2	<0.001 ***	SI
WI			xlsr_fp	<0.001 ***	xlsr_fp2	<0.001 ***	WI
MP	xlsr_fp				xlsr_fp2	<0.001 ***	MP
SI					xlsr_fp2	<0.001 ***	SI
WI					xlsr_fp2	<0.001 ***	WI
MP	xlsr_fp2						MP
SI							SI
WI							WI

Abbreviation: fp2 - further pre-training in two stages

Test Abbrev.		xlsr	xlsr_fp		xlsr_fp_133h		Test Abbrev.
MP	xlsr		xlsr_fp	<0.001 ***	xlsr_fp_133h	<0.001 ***	MP
SI			xlsr_fp	<0.001 ***	xlsr_fp_133h	<0.001 ***	SI
WI			xlsr_fp	<0.001 ***	xlsr_fp_133h	<0.001 ***	WI
MP	xlsr_fp				xlsr_fp	<0.001 ***	MP
SI					xlsr_fp	<0.001 ***	SI
WI					xlsr_fp	<0.001 ***	WI
MP	xlsr_fp_133h						MP
SI							SI
WI							WI

CU-MARVEL Experiment 3

For *assessors*:

Test Abbrev.		xlsr	xlsr_fp			xlsr_fp_pl			Test Abbrev.
MP	xlsr		xlsr_fp	<0.001	***	xlsr_fp_pl	<0.001	***	MP
SI			~	0.062		~	0.062		SI
WI			xlsr_fp	0.043	*	xlsr_fp_pl	0.043	*	WI
MP	xlsr_fp					xlsr_fp_pl	<0.001	***	MP
SI						~	0.375		SI
WI						~	0.139		WI
MP	xlsr_fp_pl								MP
SI									SI
WI									WI

For *participants*:

Test Abbrev.		xlsr	xlsr_fp			xlsr_fp_pl			Test Abbrev.
MP	xlsr		xlsr_fp	<0.001	***	xlsr_fp_pl	<0.001	***	MP
SI			xlsr_fp	<0.001	***	xlsr_fp_pl	<0.001	***	SI
WI			xlsr_fp	<0.001	***	xlsr_fp_pl	<0.001	***	WI
MP	xlsr_fp					xlsr_fp_pl	<0.001	***	MP
SI						xlsr_fp_pl	<0.001	***	SI
WI						xlsr_fp_pl	<0.001	***	WI
MP	xlsr_fp_pl								MP
SI									SI
WI									WI