Tokyo Institute of Technology
School of Engineering
Information and Communications Engineering

Master Thesis

Cross-Lingual Low-Resource Speech Recognition with Large-Scale Multilingual Pre-training

Wenxin Hou
19M18820

Academic Supervisor
Associate Professor
Takahiro Shinozaki

2021

Abstract

Cross-lingual speech adaptation aims to solve the problem of leveraging multiple rich-resource languages to build models for a low-resource target language. Since the low-resource language has limited training data, speech recognition models can easily overfit. In this thesis, I introduce a novel transfer learning framework based on large-scale multilingual pre-training and adapter-based cross-lingual adaptation. Under the introduced framework, I propose a super multilingual model named LID-42, which is trained on up to 5,000-hour training data mixing 42 languages based on hybrid CTC-Attention Transformer. I also propose two novel algorithms: MetaAdapter and SimAdapter. The proposed algorithms leverage adapters which can be easily integrated into the Transformer structure. The MetaAdapter leverages meta-learning to transfer the general knowledge from training data to the test language. SimAdapter aims to learn the similarities between the source and target languages during fine-tuning using the adapters. I conduct extensive experiments on five low-resource languages in Common Voice dataset. Results demonstrate that our MetaAdapter and SimAdapter methods can reduce WER by 2.98% and 2.55% with only 2.5% and 15.5% of trainable parameters compared to the strong full-model fine-tuning baseline. Moreover, I also show that these two novel algorithms can be integrated for better performance with up to 3.55% relative WER reduction.

Acknowledgement

My deepest gratitude goes first and foremost to my supervisor Prof. Takahiro Shinozaki for giving me this precious opportunity to spend my Master of Engineering program under his constant encouragement and guidance at Tokyo Institute of Technology. I have learnt from our meeting and other interactions with him and experienced how optimistic and friendly he was. Without his consistent instruction, much progress in research and completion of my degree program would not have been made.

Second, I would like to express my heartfelt gratitude Dr. Jindong Wang for offering selfless help during my internship at Microsoft Research Asia.

My thanks would also go to my parents and grandparents for their loving considerations and constant support during my study.

Abbreviations

ASR	Automatic Speech Recognition
BERT	Bidirectional Encoder Representations from Transformers
CER	Character Error Rate
CTC	Connectionist Temporal Classification
E2E	End-to-End
LI	Language Identification
LID-42	Language-Independent Transformer for 42-Lingual Speech Recognition
MAML	Model-Agnostic Meta-Learning
NMT	Neural Machine Translation
TER	Token Error Rate
WER	Word Error Rate
XLM	Cross-Lingual Language Model

List of Figures

List of Tables

Contents

Chapter 1

Introduction

1.1 Background

Automatic speech recognition (ASR) based on end-to-end (E2E) models has made remarkable progress by training on large-scale data [1,2]. We can use a single E2E ASR system for a large number of languages [3,4] without complicated language-specific processing. Nevertheless, a well-known limitation of E2E ASR methods is that they require considerable amount of training data to achieve superior performances. Therefore, it remains a challenge for E2E ASR models to achieve reasonable recognition performance for most of the low-resource languages among about 7,000 languages in the world.

Some research has indicated that the performances of low-resource languages benefit by transferring the common knowledge from rich-resource languages in ASR [5]. For instance, as shown in Figure 1.1.1, given Romanian as a low-resource target language, cross-lingual ASR aims to build models by leveraging the available rich-resource languages such as Italian, Welsh, and Russian as source languages. Then the following question naturally arises: given these rich-resource languages as source languages, how can we learn the transferable knowledge from them to build cross-lingual ASR models for the target language Romanian?

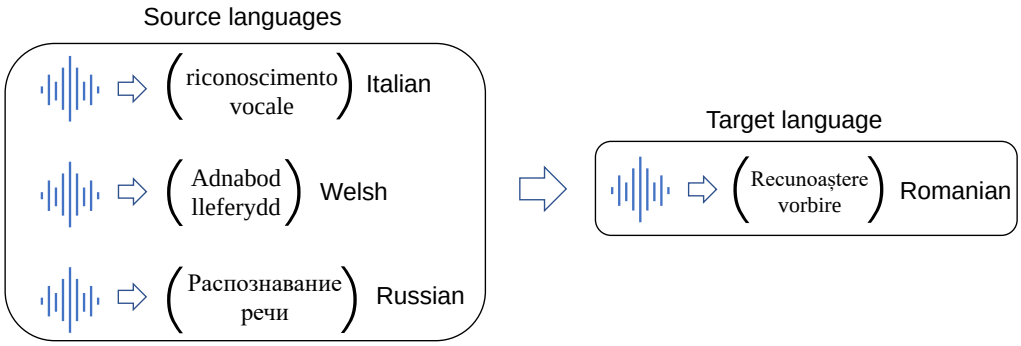


Figure 1.1.1: Illustration of the cross-lingual speech recognition task.

Generally speaking, knowledge transfer can be achieved in three avenues: (1) pre-training on the rich-resource languages and then fine-tuning on the low-resource tasks [3, 6]; (2) performing multi-task training using all languages [7]; and (3) learning the general common knowledge and then rapidly adapting to the low-resource languages using meta-learning [8]. A possible explanation is that different languages intrinsically share some beneficial information like speaker, environment and part of linguistic and phonetic knowledge.

Unfortunately, the existing three avenues all have various limitations. For method (1), due to the limited training data in low-resource languages, direct fine-tuning by re-training the parameters makes the model easily overfit and results in catastrophic forgetting problem. As a consequence, the parameter efficiency for method (1) is low. On the other hand, for method (2), the performance of the target low-resource language could be biased by other high-resource languages with much more training data; Lastly, (3) solves the overfitting problem of (1), but meta-learning pre-training requires heavy computation and it still faces the catastrophic forgetting problem. These problems make current transfer-based methods inefficient [9, 10].

Recently, the adapter module was proposed for parameter-efficient fine-tuning in multilingual settings [10, 11] to mitigate the data imbalance problem among languages. Adapter is an add-on module to the encoder and decoder layers in Transformer that mainly composed of layer normalization and fully-connected

layers. During fine-tuning, we can freeze the backbones of the pre-trained models and only train the adapters which has a small number of task-specific parameters. Pfeiffer et al. [12] studied the fusion of adapters in natural language processing where they linearly combine the outputs of multiple adapters for target classification task adaptation. However, it remains unexplored to investigate the performance of adapters on cross-lingual ASR tasks.

To solve the problems mentioned above, in this thesis, I introduce a novel transfer learning framework based on large-scale multilingual pre-training and adapter-based cross-lingual adaptation. I firstly pre-train a super multilingual Transformer model for 42-lingual ASR using up to around 5,000 hours labelled speech data, which is language-independent and named as LID-42. It serves as the backbone for the following cross-lingual transfer. Then I propose to inject adapters into the backbone to perform cross-lingual adaptation, which promises the parameter efficiency and could avoid the overfitting problem due to their fewer trainable parameters.

Under the introduced framework, I propose two novel algorithms in this thesis: *MetaAdapter* [9] and *SimAdapter* [13]. I propose MetaAdapter to implicitly learn general and transferable speech representations using model-agnostic meta-learning (MAML) [14] and achieved promising results on extremely low-resource languages. I propose *SimAdapter* to explicitly learn the similarity between the source and target languages using the attention mechanism. My key motivation of these methods is that different language in the world are sharing similarities based on their similar geological characteristics or evolution [15–17]. Therefore, it is feasible to implicitly or explicitly model such similarities in the ASR models.

It is worth noting that both of the two algorithms I present in this thesis are parameter-efficient and thus can prevent the overfitting problem. To my best knowledge, there is no existing research that tries to model the cross-lingual ASR tasks by studying their relationship using meta-learning and transfer learning-based adapters. In addition, the MetaAdapter and SimAdapter are compatible,

thus can be integrated for better performance.

1.2 Thesis Organization

The structure of this paper is as follows. In Chapter 2, I review the fundamentals of E2E ASR including CTC, Transformer model with attention mechanism and hybrid CTC-Attention architecture.

Chapter 3 introduces the and presents my super LID-42 model based on hybrid CTC-attention Transformer and 42-lingual language-independent modeling. This chapter is based on my previous work [4].

Chapter 4 introduces the details of the introduced framework including MetaAdapter and SimAdapter algorithms and their integration SimAdapter +. I also present extensive experiments and analysis on the proposed methods and compare them with the conventional fine-tuning or direct training approaches. This chapter is based my previous works [9, 13].

Finally, in Chapter 5, I conclude this thesis and present some possible directions of my future work.

1.3 Contributions

My contributions in this thesis can be summarized as follows:

- I propose a transfer learning framework consisting of 2 novel algorithms based on large-scale pre-training and adapter-based adaptation for cross-lingual and low-resource ASR.
- I present LID-42, a large-scale Transformer-based super multilingual model, which is a strong backbone and has shown significant improvements of the cross-lingual ASR performance on low-resource languages.

- I comprehensively analyze the proposed MetaAdapter and SimAdapter algorithms for cross-lingual low-resource ASR in terms of recognition accuracy, interpretability, and training / inference efficiency via extensive experiments.
- Experiments on five low-resource languages demonstrate a relative WER improvement of 2.98% with MetaAdapter and 2.55% with SimAdapter using only 2.5% and 15.5% trainable parameters compared with the strong full-model fine-tuning , respectively.
- Finally, I show that the two algorithms can be integrated as SimAdapter + to achieve better performance with up to 3.55% relative improvement.

Chapter 2

Foundations of End-to-End Speech Recognition

2.1 Overview

The most widely used end-to-end ASR systems is composed of 2 functional modules named encoder and decoder. The encoder consumes N-frame acoustic features $X = \{x_1, x_2, ..., x_N\}$ (i.e., MFCC, log-Mel filterbank, etc.) of the raw speech as inputs and generates intermediate representations X_{enc}:

$$X_{\text{enc}} = \text{Encoder}(X), \tag{2.1}$$

The decoder is then required to predict the posterior distribution based on the encoder outputs X_{enc}:

$$P(Y|X_{\text{enc}}) = \text{Decoder}(X_{\text{enc}}), \tag{2.2}$$

where n denotes the number of tokens in the predicted text.

In the following subsections, we will firstly introduce the widely-adopted attention-based Transformer model including encoder and decoder for sequence-to-sequence modeling. This is followed by the introduction to Connectionist

Temporal Classification (CTC), another effective non-autoregressive method for E2E ASR. In the third subsection, we will introduce the hybrid CTC-Attention architecture combining the advantages of both CTC and vanilla attention models. Lastly, we introduce the commonly adopted metrics to evaluate the performance of E2E ASR models.

2.2 Speech Transformer Model

Transformer is an attention-based model introduced by Vaswani et al. [18] to solve the sequential modeling tasks. It has shown promising results on many natural language processing (NLP) tasks [19–21]. Dong et al. [22] introduced the Speech-Transformer as a variant of Transformer in the speech processing field. It has been widely adopted for various speech applications. In the following subsections, I will explain the overall model structure as well as the details of the core components of the Speech-Transformer.

2.2.1 Overall Structure

The overall architecture is shown in Figure 2.2.1. As model inputs, the Transformer model the acoustic features X (83-dimensional filter banks with pitch in this thesis). The acoustic features are firstly subsampled by a factor of 4 by 2 convolution layers with kernel size 3 and stride 2. Then the following encoder layers process the subsampled features (or embeddings) by applying self-attention and the feed-forward networks.

$$X_{\text{enc}} = \text{TransformerEncoder}(X_{\text{enc}}) \tag{2.3}$$

Before sent to the Transformer decoder layers, the input subwords or characters are firstly mapped to the corresponding embeddings. The Transformer decoder layers also accept the encoder outputs to perform cross-attention apart

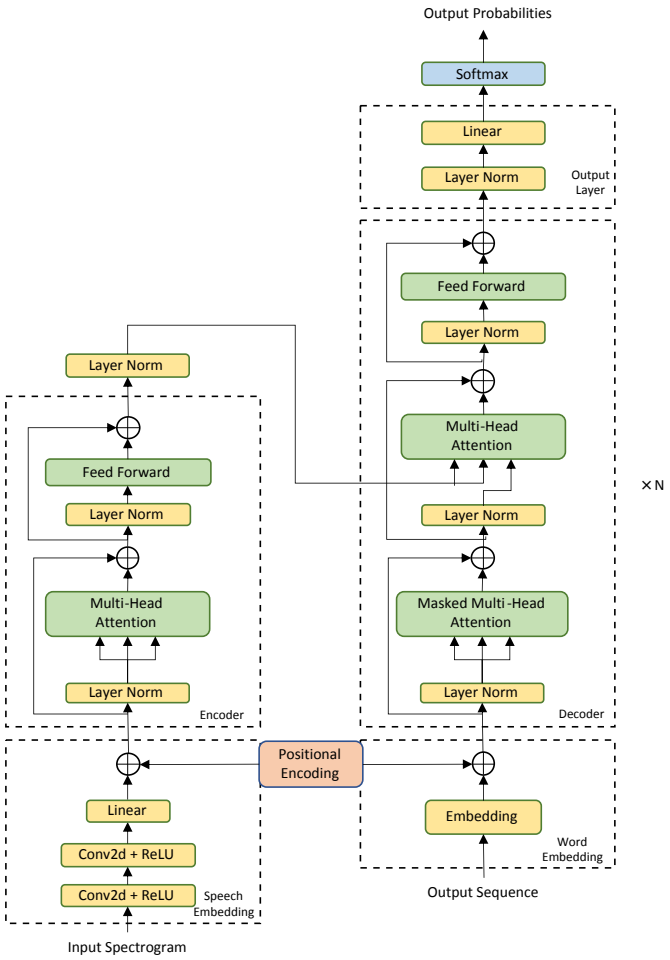


Figure 2.2.1: Illustration of the Speech-Transformer architecture.

from the self-attention and feed-forward. Note that the decoding process of the Transformer decoder is autoregressive, which means that t-th prediction Y_t is conditioned on the previous $t-1$ predictions $Y_{1:t-1}$:

$$P(Y_t|X_{\text{enc}}) = \text{TransformerDecoder}(X_{\text{enc}}, Y_{1:t-1}). \tag{2.4}$$

It is worth noting that unlike the vanilla Transformer [18] that injects layer normalization (LN) layers after the multi-head attention and feed-forward layers (named as Post-LN), Speech-Transformer adopt the Pre-LN structure where the LNs are before other layers. The Pre-LN structure has been demonstrated being able to stabilize the training and accelerate the convergence [23].

2.2.2 Multi-Head Attention

The core component of the Transformer is the multi-head attention. The attention operation can be formulated as below:

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{d_k}})V, \tag{2.5}$$

where d_k is the feature dimension of the key matrix.

$$\text{MultiHead}(Q, K, \mathbf{V}) = \text{Concat}(\text{head}_1, \text{head}_2, ..., \text{head}_n)W^O, \tag{2.6}$$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

Specifically, for the self-attention, the query Q, key K, and value V are the same features from the previous layer; while for the encoder-decoder cross-attention, only Q is from the previous decoder layer while K and V are the encoder outputs.

2.2.3 Feed-Forward Networks

The feed-forward networks (FFN) consist of 2 linear transformation layers and a ReLU activation function. It can be formulated as:

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2, \tag{2.7}$$

where x is the input feature, W_1 and W_2 are trainable weights, b_1 and b_2 are trainable biases.

2.2.4 Positional Encoding

Position information is essential in sequential tasks, for example, change of word orders could drastically change the sentence meaning. Since the attention layers do not contain position information, position-dependent sine- and cosine-

based positional encoding functions are introduced to each element of both the encoder and decoder embeddings as explicit position information.

$$\begin{cases} \text{PE}_{pos,2i} = \sin\left(\text{pos}/10,000^{2i/d_{\text{model}}}\right) \\ \text{PE}_{pos,2i+1} = \cos\left(\text{pos}/10,000^{2i/d_{\text{model}}}\right) \end{cases}, \tag{2.8}$$

where pos and i denote the position and dimension of elements, respectively. d_{model} is the model dimension.

2.2.5 Optimization

We can optimize the Transformer model following the loss function below:

$$\mathcal{L}_{\text{ATT}} = -\log P(Y|X_{\text{enc}}) = -\log \prod_t P(Y_t|X_{\text{enc}}, Y_{1:t-1}), \tag{2.9}$$

where X_{enc} represent the Transformer encoder outputs, $Y_{1:t-1}$ denotes the tokens of all the previous time steps before time step t.

2.3 Connectionist Temporal Classification

Connectionist Temporal Classification (CTC) [24] is an efficient objective for end-to-end ASR. By assuming conditional independence and monotonic alignment, CTC is able to model the target sequence via an alignment variable Z. We can define a many-to-one mapping function $B(\cdot)$ so that

$$Y = B(Z), \tag{2.10}$$

where $B(\cdot)$ removes the blank and repeated symbols. For example, for the alignment $Z =$ "happ<blank>py", the target sequence is $Y = B(Z) =$ "happy".

In practice, the CTC module often consists of a linear transformation and a softmax layer to obtain the prediction score matrix $C \in \mathbb{R}^{T \times V}$, where T and V

denote the number of time steps and vocabulary size, respectively.

Then we can compute the posterior possibility of the alignment Z which maps to the target sequence Y as below:

$$p_{\text{CTC}}(B(Z) = Y|X_{\text{enc}}) = \prod_{t=1}^{T} C_{t,Z_t}, \tag{2.11}$$

where X_{enc} denotes the encoder outputs of T frames, C_{t,Z_t} denotes the confidence score of the Z_t's token at time step t.

The training objective of CTC is to maximize the posterior possibilities of all the possible alignments that could map to the target sequence Y:

$$\mathcal{L}_{\text{CTC}} = -\log \sum_{Z \in B^{-1}(Y)} p_{\text{CTC}}(B(Z) = Y|X_{\text{enc}}), \tag{2.12}$$

where $B^{-1}(Y)$ is a one-to-many mapping that maps Y to all the possible alignments $B^{-1}(Y) = \{Z|B(Z) = Y\}$, which is the inverse of $B(\cdot)$.

2.4 Hybrid CTC-Attention Architecture

The hybrid CTC-attention architecture is proposed by Watanabe et al. [25] to effectively utilize benefits of both CTC and attention decoders during the training and decoding steps in ASR. To be specific, the aforementioned CTC task [24] is introduced as an auxiliary objective for the encoder outputs in order to encourage the monotonic alignment and accelerate the convergence speed [25]. During training, a weighted sum of the sequence-to-sequence attention loss $\mathcal{L}_{\text{ATT}}$ and the CTC loss $\mathcal{L}_{\text{CTC}}$ is employed:

$$\mathcal{L}_{\text{ASR}} = (1 - \lambda)\mathcal{L}_{\text{ATT}} + \lambda\mathcal{L}_{\text{CTC}}, \tag{2.13}$$

where λ denotes the weight of the CTC module.

Similarly, during decoding, the CTC module outputs are used to re-score the

beam search results of the Transformer decoder on-the-fly:

$$\hat{Y} = \arg\max_{Y \in \mathcal{Y}} (1-\lambda) \log P_{\mathrm{ATT}}(Y|X_{\mathrm{enc}}) + \lambda \log P_{\mathrm{CTC}}(Y|X_{\mathrm{enc}}), \tag{2.14}$$

where X_{enc} are the encoded features, $\mathcal{Y}$ denotes the set of the decoding hypotheses.

2.5 Evaluation Metrics

2.5.1 Character / Word Error Rate

In this thesis, I use character error rate (CER) or word error rate (WER) as the evaluation metric. Both CER and WER are commonly used edit-distance-based metrics to evaluate the performance of ASR systems. Their difference is that CER takes characters as the basic units while WER takes words as the basic units.

Given the reference sequence of length N and the ASR predicted sequence, we can align the two sequences by dynamic programming and count the number of insertions I, the number of deletions D and the number of substitutions S, the WER or CER can be calculated as follows:

$$\text{WER (CER)} = \frac{I + D + S}{N}. \tag{2.15}$$

Lower error rates indicate higher accuracy and better ASR performance. Note that the WER or CER could be larger than 100%.

2.5.2 Real-Time Factor

The Real-Time Factor (RTF) metric is often used to evaluate the decoding speed of the ASR systems in terms of the decoding time cost. It is measured by computing the ratio of the model decoding time T_{dec} to the total utterance duration

T_{utt} on the speech data.

$$\text{RTF} = \frac{T_{\text{dec}}}{T_{\text{utt}}}, \tag{2.16}$$

Lower RTF represents smaller decoding time cost, and thus faster decoding speed.

Chapter 3

Multilingual Modeling for End-to-End Speech Recognition

3.1 Related Works

Multilingual E2E ASR is getting increasing attention over the years that handles multiple languages with a single model. Watanabe et al. [26] proposed a language-independent architecture based on hybrid CTC-attention structure [25] with augmented vocabulary for character-based E2E ASR and joint language identification. Toshniwal et al. [27] found that multilingual training leads to significant relative improvement of recognition performance and the results can be further boosted by conditioning the model on a language identifier. Some attempts take a step towards realizing language-universal ASR. Li et al. [28] proposed to replace the characters with the Unicode bytes as the output. Datta et al. [29] unified different writing systems through a many-to-one transliteration transducer. Recently, large-scale multilingual ASR systems have been investigated [3, 4, 6, 10, 30]. Pratap et al. [3] proposed jointly training on 16,000 hours speech data of 51 languages with up to 1 billion parameters. Inspired by [26], I presented LID-42 in [4], a large-scale multilingual acoustic Transformer model trained on 11 mixed corpora of 42 languages.

3.2 Proposed Method: LID-42

LID-42 is a super multilingual ASR model based on 12 Transformer encoders and 6 Transformer decoders with hybrid CTC-Attention architecture. It is trained on around 5000-hour labeled speech data mixing 11 corpora covering 42 language to obtain strong speech and language modeling abilities. I also make two modifications following [26] to adapt it to the multilingual ASR task. The overall system architecture can be found in Figure 3.2.1.

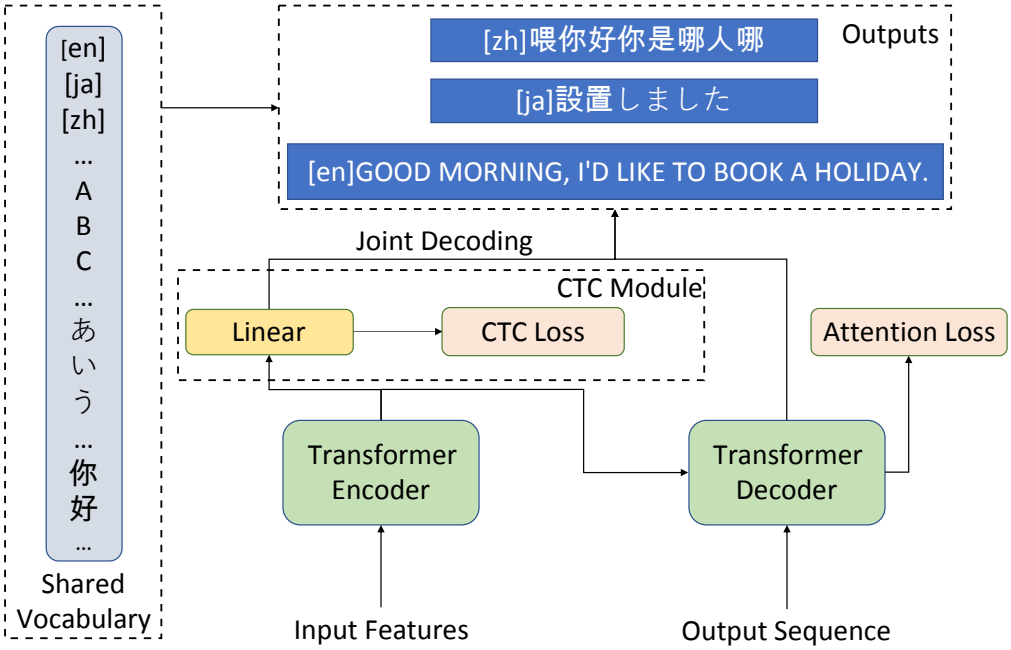


Figure 3.2.1: LID-42 system architecture

3.2.1 Shared Vocabulary of Modeling Units

To enable language-independent training and recognizing, I employ an augmented shared vocabulary of modeling units and language tokens (e.g., `<en>`, `<fr>`) of all the 42 languages. In this thesis, I compare two kinds of modeling units: characters and subwords.

3.2.2 Joint Language Identification Task

To reduce the possibility that predictions switch between languages, the language tokens mentioned above are inserted to the beginning of training labels Y as an auxiliary language identification objective. Consequently, the model learns to first identify the language before predicting the speech contents. This can be regarded as an auxiliary language identification (LID) task. It is worth noting that the language identification objective is only used for LID-42 pre-training and is dropped during transfer learning.

3.3 Evaluation

3.3.1 Mixed Corpora of 42 Languages

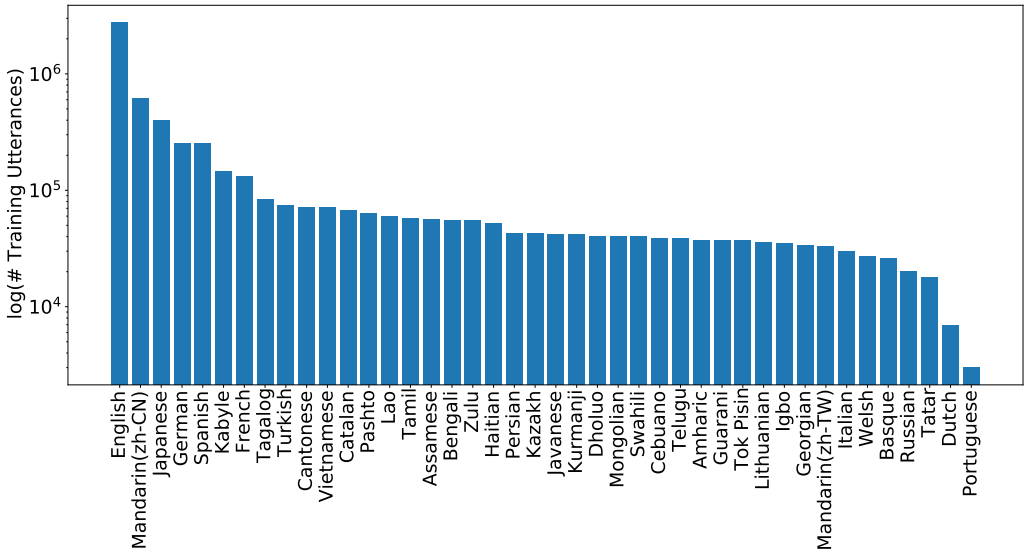


Figure 3.3.1: Number of training utterances for LID-42 (at log scale)

Figure 3.3.1 shows the data I used for training the LID-42 model. It is mixed from 11 corpora including: AISHELL [31], Aurora4, Babel, CHiME4, Common Voice [32], Corpus of Spontaneous Japanese (CSJ) [33], Fisher Switchboard, Fisher Callhome Spanish, HKUST [34], WSJ and Voxforge.

3.3.2 Implementation Details

For all the experiments, 83-dimensional input features are extracted from the raw speech composed of 80-dimensional filter banks and 3-dimensional pitch features computed every 10 ms over a 25 ms window. The detailed Transformer configuration follows the same setting as the *big model* described in [22]. The models are trained using Adam optimizer with a varying learning rate lr strategy:

$$lr = k \cdot d_{\text{model}}^{-0.5} \cdot \min(\text{step}^{-0.5}, \text{step} \cdot \text{warmup\_step}^{-1.5}), \tag{3.1}$$

where hyperparameter k is the learning rate factor, lr linearly warms up before step reaching warmup_step and decreases proportionally to the inverse square root of step afterward.

I employ two kinds of modeling units: characters and subwords. The character vocabulary includes 7,381 characters, and the subword vocabulary includes 15,943 subwords. The subwords are obtained using the SentencePiece library [35]. Apart from characters/subwords, 60 non-language symbols such as language IDs and other symbols (e.g., `<unk>`) are also included in the vocabularies.

The ESPnet toolkit [36] is used to conduct the experiments [1]. I follow the same training and testing data splitting rules of corresponding ESPnet [36] recipes. The model training and evaluation were performed using the TSUBAME 3.0 supercomputer [2]. To accelerate the training process, I applied PyTorch distributed communication package [3] to train the model on 10 computing nodes with 40 NVIDIA TESLA P100 GPUs. This results in a total batch size of 1,280. In total, the character-based model takes around 163 hours while the subword-based model takes 222 hours to complete training.

[1] Recipe for LID-42 is available as part of ESPnet: `https://github.com/espnet/espnet/tree/master/egs/li42/asr1`

[2] https://www.gsic.titech.ac.jp/en/tsubame

[3] https://pytorch.org/docs/stable/distributed.html

3.3.3 Final Results

I present the language-specific and weighted average results in Figure 3.3.2. From left to right, the language-specific results are sorted in the decreasing order by the amount of training data. Firstly, we can observe that the CER results vary a lot among the 42 languages. For example, for languages like English, Japanese, and German, the CERs can be below 10%. On the other hand, for languages like Kazakh, Mongolian, and Tatar the CERs can be larger than 80%. This could be mainly due to their different amount training data and the language-specific task difficulty. The language similarity could also affect the model performance, for example, although the Italian, Welsh, and Dutch do not have much training data, they are similar to other European languages like French, English, and German.

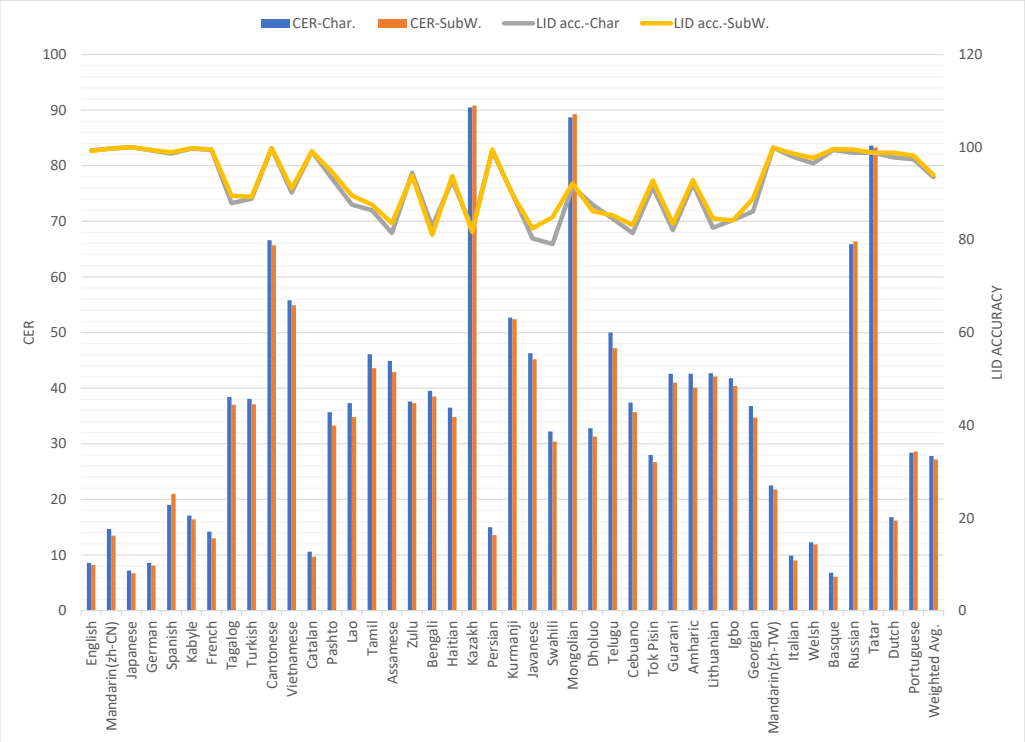


Figure 3.3.2: Character error rates (CER) and language identification (LID) accuracies of character-based model (Char.) and subword-baesd model (SubW.). From left to right, the language-specific results are sorted in the decreasing order by the amount of training data.

Secondly, it is found that the LID accuracy is not totally dependent on the

recognition accuracy, some languages have worse CERs but higher LID accuracies, e.g. Cantonese and Tatar. The reason could be that their unique characteristics make it hard to borrow knowledge from other languages but also make it easier to distinguish them from other languages. The LID accuracies are all beyond 79%, indicating the strong multilingual modeling ability of LID-42.

By comparing the subword- and character-based variants, we can see that the subword-based vocabulary improves the model's overall performance in terms of CER and LID accuracy for 38 and 29 languages, respectively. The weighted average of CER is reduced from 27.8% to 27.2%. Meanwhile, the weighted average of LID accuracy is increased from 93.5% to 94.0%. These results demonstrate the superiority of introducing subwords.

Chapter 4

Exploiting Adapters for Cross-Lingual Speech Recognition

4.1 Problem Definition

The goal of cross-lingual speech recognition is to transfer the knowledge from the existing languages to the new language. Formally speaking, given N rich-resource languages $\{S_1, S_2, \cdots S_N\}$, cross-lingual ASR aims at adapting the pre-trained model to an unseen target low-resource language L_T. Each language S_i is composed of the speech-text pairs and I typically use X and y to denote them in this chapter, respectively, i.e., $S_i = \{X_j, y_j\}_{j=1}^{N_i}$, where N_i is the total number of training data. Also note that the target language is low-resource compared to the training languages, i.e., $N_T \ll N_i, \forall 1 \leq i \leq N$.

4.2 Related Works

4.2.1 Cross-Lingual Speech Recognition

Cho et al. [37] validated the effectiveness of cross-lingual transfer learning for improving ASR performance. And this advantage can be further revealed by

large-scale pre-training [3, 6]. For example, LID-42 can achieve a relative WER reduction of up to 28.1% on low-resource languages by cross-lingual transfer [4]. Yi et al. [38] introduced an adversarial learning objective to learn language-agnostic features. They appended a language discriminator after the shared encoder to distinguish which language the bottleneck features belong to. The objective of the discriminator is to correctly identify the language while the adversarial objective of the encoder is to fool the discriminator. The adversarial training process is realized with the use of the gradient reversal layer (GRL) [39]. Adams et al. [6] performed experiments to analyze the impacts of language similarity, context-independent phoneme CTC objective and the aforementioned language-adversarial classification objective during multilingual pre-training to encourage language-agnostic features for better cross-lingual adaptation.

Besides learning the language-agnostic features, the optimization-based meta-learning approaches [14, 40] that aim to find a proper initialization for rapid adaptation have also been explored for cross-lingual ASR [9]. Hsu et al. [8] proposed to apply the model-agnostic meta-learning (MAML) [14] as the pre-training method and achieved significant improvement over the conventional multilingual pre-training baseline. Xiao et al. [41] proposed the Adversarial Meta Sampling framework by introducing a policy network (task sampler) to dynamically sample languages based on their task difficulty. The ASR model is trained to minimize the loss while the task sampler learns to choose the most difficult languages that can maximize the loss. As a consequence, the learned initialization has a more balanced distance to all languages and shows good generalization capacity in low-resource speech tasks.

4.2.2 Adapters

Due to the large quantity of parameters contained in the Transformer-based models [4, 19, 42, 43], recent literature proposed the *Adapter* structure [44, 45] for parameter-efficient adaptation of pre-trained Transformers [18, 19] on var-

ious downstream tasks including language understanding [46] and neural machine translation (NMT) [18], etc. Adapter is a versatile module that can be plugged into the Transformer blocks. The general philosophy for adapter-based fine-tuning is to freeze the parameters θ_b of the Transformer backbone and only tune the parameters θ_a of the adapter. Compared to fine-tuning the whole Transformer model, fine-tuning the adapters is significantly efficient with acceptable performance loss [44]. Therefore, adapters have been adopted as a fine-tuning technique in few-shot domain adaptation for NMT [47] and unsupervised cross-lingual transfer [48] or domain adaptation [49] of large-scale pre-trained language models like BERT [19] and XLM [50]. Li et al. [51] proposed a hypernetwork that could generate parameters of task-specific adapters from task descriptions to enable zero-shot learning [52]. More recently, Pfeiffer et al. [12] introduced the AdapterFusion module to fuse adapters trained on different tasks to share the knowledge. The difference between my work and theirs are that I focus on the cross-lingual sequence-to-sequence ASR task while they experiment on text classification tasks based on the pre-trained BERT [19].

Some researchers have proposed to apply the Adapters to the E2E ASR tasks. In [10], Kannan et al. proposes to use the adapters to handle the data imbalance problem for large-scale multilingual ASR. After obtaining the model trained on the union of data from all languages, they trained the language-dependent adapters on each of the languages again so that the multilingual backbone shares information across languages while the adapters could allow for per-language specialization. Winata et al. [11] extends this idea by further introducing a common adapter for all languages to learn language-agnostic information in the multilingual data.

4.3 Proposed Method: MetaAdapter

In this section, I introduce the MetaAdapter in detail. MetaAdapter is inspired by the idea of meta-learning [53] for fast adaptation to the new target tasks. In my previous work [9], I investigated two meta-learning algorithms: Model-Agnostic Meta-Learning (MAML) [14] and Reptile [40]. I found that MAML is more robust to the overfitting problem brought by the variance of adaptation data size and pre-training epochs. Therefore, I adopt the MAML as the meta-training algorithm in this thesis.

However, it is expensive to perform meta-learning directly on the full Speech-Transformer model since the model has heavy parameters that could easily overfit on the low-resource target data. To resolve this issue, MetaAdapter utilizes the adapter modules to significantly reduce the adaptation parameters while aiming to find a proper initialization for faster adaptation.

4.3.1 Model-Agnostic Meta-Learning for Adapters

The process of MetaAdapter is illustrated in Figure 4.3.1. Given a pre-trained backbone speech-Transformer ASR model, MetaAdapter is composed of two phases: (i) meta-train the MetaAdapter on a bunch of source tasks; (ii) fine-tune the pre-trained adapter on unseen target tasks.

To use meta-learning, I view different languages as different tasks. I split the parameters of MetaAdapter into two types: the backbone parameters θ_b (i.e., vanilla Transformer) and the parameters of all adapters θ_a. Thus, given N different source languages $\{S_1, S_2, \cdots, S_N\}$, I pre-train the MetaAdapter module f_{θ_a} to obtain good initialization parameters θ_a that could generalize for fast adaptation given any unseen target language. Meanwhile, parameters of the pre-trained backbone θ_b are frozen during both the pre-training and the fine-tuning.

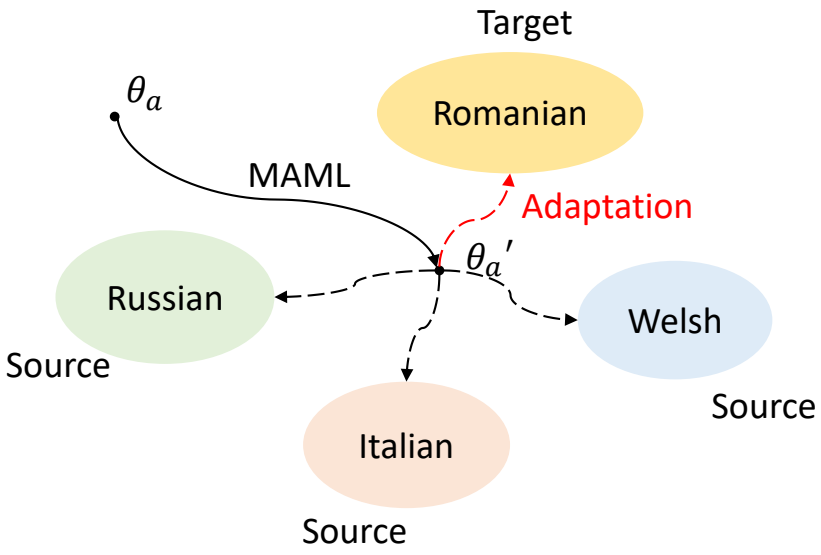


Figure 4.3.1: Illustration of MetaAdapter.

Algorithm 1 Learning algorithm of the MetaAdapter

Input: Rich-resource languages $\{S_1, \cdots, S_N\}$, low-resource task L_T.

1: Train language-specific heads on source languages S_i.
2: Initialize the MetaAdapter.
3: **while** meta-learning not done **do**
4: Optimizing the MetaAdapter using Eq. (4.3).
5: **end while**
6: Train the target head on target language L_T.
7: Fine-tune the MetaAdapter using ASR loss Eq. (2.13).
8: **return** Cross-lingual ASR model.

4.3.2 Training MetaAdapter

In each pre-training episode, two subsets are randomly sampled from each source training language S_i, namely meta-training set S_i^{tr} and meta-validation set S_i^{val}, i.e., $S_i^{tr} \cap S_i^{val} = \emptyset$. One episode is composed of two iterations: an inner iteration and an outer iteration. In the inner iteration, MAML updates the adapter parameters θ_a by performing one or more gradient descent on S_i^{tr}. For notation simplicity, the updated parameter for language S_i using the inner gradient descent iteration is:

$$\theta'_{a,i} = \theta_a - \epsilon \nabla \mathcal{L}_{S_i^{tr}} \left(f_{\theta_a} \right), \tag{4.1}$$

where $\mathcal{L}$ is the ASR loss function as introduced in section **??** and ϵ is the fast adaptation learning rate. In the outer iteration, the adapter parameters are then optimized to improve the performance of $f_{\theta'_{a,i}}$ with respect to θ_a across all the meta-validation sets S_i^{val}. The meta-optimization objective of the outer iteration is:

$$\mathcal{L}_{S_i^{val}}(f_{\theta'_{a,i}}) = \mathcal{L}_{S_i^{val}}\left(f_{\theta_a - \epsilon\nabla_{\theta_a}\mathcal{L}_{S_i^{tr}}(f_{\theta_a})}\right). \tag{4.2}$$

I optimize the meta-optimization objective through gradient descent as:

$$\theta_a = \theta_a - \mu\sum_{i=1}^{N}\nabla_{\theta_a}\mathcal{L}_{S_i^{val}}\left(f_{\theta'_{a,i}}\right), \tag{4.3}$$

where μ is the meta step size.

After pre-training, the MetaAdapter should obtain a proper initialization for any unseen target language(s). The complete training procedure of the MetaAdapter is presented in Algo. 1.

4.4 Proposed Method: SimAdapter

Our motivation of SimAdapter is to improve the adapter-based cross-lingual adaptation as well as the model interpretability by explicitly leveraging the knowledge of the source languages stored in the adapter modules. Here, 'Sim' refers to similarity.

SimAdapter is inspired by existing research on language and speech origins [15–17], which imply that different languages in the world are are sharing similarities based on their similar geological characteristics or cultural developments. Thus, it is feasible to leverage the knowledge of multilingual adapters for new target languages.

4.4.1 Architecture

SimAdapter is a parameter-efficient algorithm that learns the similarities between existing language-specific adapters and the target low-resource language based on the attention mechanism [18]. Similar to the adapters, SimAdapter can also be easily integrated with existing pre-trained models and adapters.

The detailed composition of the SimAdapter is shown in Figure 4.4.1. By taking the language-agnostic representations from the backbone model as the query, and the language-specific outputs from multiple adapter as the keys and values, the final output for SimAdapter over attention are computed as (For notation simplicity, I omit the layer index l below):

$$\mathrm{SimAdapter}(\mathbf{z}, \mathbf{a}_{\{S_1,S_2,...,S_N\}}) = \sum_{i=1}^{N} \mathrm{Attn}(\mathbf{z}, \mathbf{a}_{S_i}) \cdot (\mathbf{a}_{S_i}\mathbf{W}_V)\,, \tag{4.4}$$

where $\mathrm{SimAdapter}(\cdot)$ and $\mathrm{Attn}(\cdot)$ denotes the SimAdapter and attention operations, respectively. Specifically, the attention operation is computed as:

$$\mathrm{Attn}(\mathbf{z}, \mathbf{a}) = \mathrm{Softmax}\left(\frac{(\mathbf{z}\mathbf{W}_Q)(\mathbf{a}\mathbf{W}_K)^\top}{\tau}\right), \tag{4.5}$$

where τ is the temperature coefficient, $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V$ are attention matrices. Note that while $\mathbf{W}_Q, \mathbf{W}_K$ are initialized randomly, $\mathbf{W}_V$ is initialized with a diagonal of ones and the rest of the matrix with small weights ($1e-6$) to retain the adapter representations. Furthermore, a regularization term is introduced to avoid drastic feature changes:

$$\mathcal{L}_{\text{reg}} = \sum_{i,j} \left((\mathbf{I}_V)_{i,j} - (\mathbf{W}_V)_{i,j}\right)^2, \tag{4.6}$$

where $\mathbf{I}_V$ is the identity matrix with the same size as $\mathbf{W}_V$.

In our cross-lingual setting, the SimAdapter module is expected to utilize language-specific knowledge from existing language adapters.

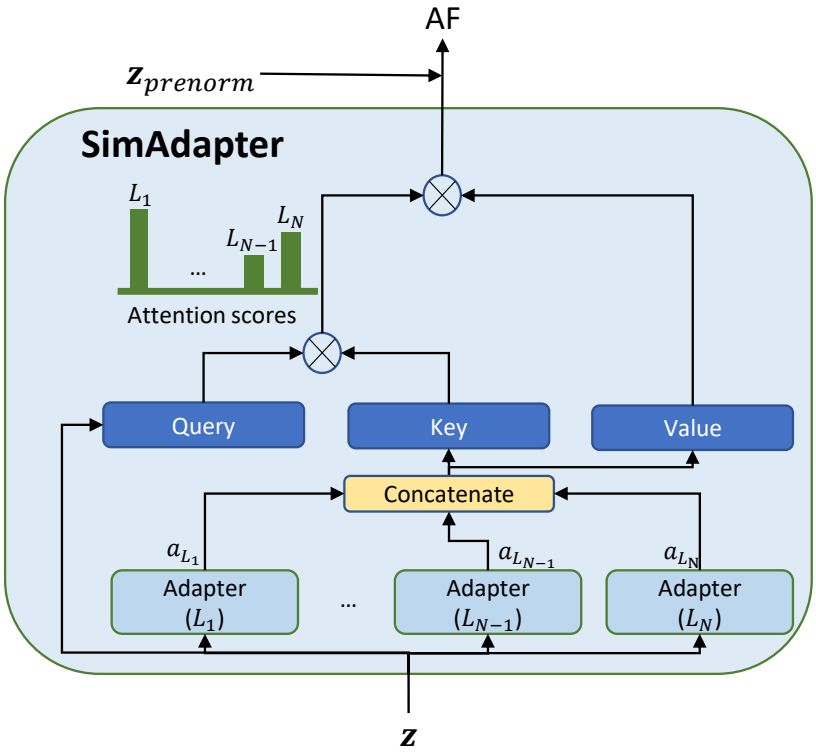


Figure 4.4.1: Illustration of the SimAdapter module.

4.4.2 Fusion Guide Loss

Although SimAdapter aims to benefit from the similar knowledge of other languages, I believe that the most crucial information is stored in the adapter of the target language. However, since the weights of source and target adapters are initialized equally, SimAdapter often distracts its attention significantly from the target language during adaptation and generally does not perform well in our experiments. To alleviate this problem, I propose a fusion guide loss to encourage the model to focus on the corresponding adapters for the similarity learning. Specifically, for each language fusion layer f, I average the cross entropy of adapter attention scores among all K time steps and S samples. The layer-wise guide losses are added up as:

$$\mathcal{L}^{f}_{\text{guide}} = -\frac{1}{K \times S} \sum_{s=1}^{S} \sum_{k=1}^{K} \log \alpha^{s}_{f,k}, \tag{4.7}$$

Algorithm 2 Learning algorithm of SimAdapter

Input: Rich-resource languages $\{S_1, \cdots, S_N\}$, low-resource task L_T.

1: Train language-specific heads on the source languages S_i and the target language.
2: Train Adapters A_t on top of language-specific heads.
3: Initialize SimAdapter layers.
4: **while** not done **do**
5: Optimizing SimAdapter layers using Eq. (4.9).
6: **end while**
7: **return** Target ASR model.

$$\mathcal{L}_{\text{guide}} = \sum_{f} \mathcal{L}^{f}_{\text{guide}}. \tag{4.8}$$

Note that K represents the number of frames in the encoder and the number of tokens in the decoder side, $\alpha^{s}_{f,k}$ denotes the attention score of target language's Adapter.

4.4.3 Training SimAdapter

A difference between the previous application of AdapterFusion [12] and our SimAdapter for cross-lingual ASR is that a language-specific language head is required to be trained for the unseen target language. However, training the Adapters together with the language heads may result in the insufficient learning of semantic information in the adapters. Therefore, in this work, I introduce a three-stage training strategy for SimAdapter-based ASR cross-lingual adaptation.

In the first stage, different from the previous work [9], SimAdapter trains the language-specific heads for each source language S_i as well as the target language separately. This step aligns the language heads to the same latent semantic space of the backbone model. In the second stage, adapters are trained based on the pre-trained heads to learn the information. In the third stage, SimAdapter leverages the fusion of source languages for better adaptation to the target language. Only the parameters of the SimAdapter are trained in this stage.

By adding the $\mathbf{W}_V$ regularization term weighted by η and the fusion guided

loss weighted by γ, the final adaptation objective is given by:

$$\mathcal{L} = \mathcal{L}_{\text{ASR}} + \eta\mathcal{L}_{\text{reg}} + \gamma\mathcal{L}_{\text{guide}}. \tag{4.9}$$

The complete training procedure of SimAdapter is presented in Algorithm 2.

4.4.4 Integration of MetaAdapter and SimAdapter

Although MetaAdapter and SimAdapter can both benefit cross-lingual adaptation by leveraging knowledge of source languages, they are designed from different perspectives. MetaAdapter aims to obtain a proper initialization for fast adaptation by learning from the source languages, which can be regarded as a type of latent transfer. On the other hand, SimAdapter explicitly models the similarities between source and target languages with attention mechanism. Therefore, MetaAdapter is good at handling extremely low-resource languages, while with more training data SimAdapter can capture the language similarities more precisely.

Moreover, note that MetaAdapter and SimAdapter are not independent algorithms. They can be integrated into one algorithm, which I denote as *SimAdapter +*. We can simply fuse the source adapters with the target adapter learned by the MetaAdapter using SimAdapter. This can be seen as a two-stage knowledge transfer process where we aim to learn general and transferable knowledge from meta-learning in the first stage; then, we perform adaptation using the SimAdapter algorithm for fine-grained knowledge transfer to achieve better performance.

4.5 Experiments

4.5.1 Data Set

I adopt the Common Voice 5.1 [32] corpus for our experiments. I follow the official data splits for training, validation and testing. For the SimAdapter, I select five rich-resource languages as source languages and five low-resource languages as targets. Note that the source and target languages are all from European and some of them are spoken in geographically close regions to empirically analyze if the language similarities can be revealed by SimAdapter. The detailed data statistics are shown in Table 4.1.

Table 4.1: Training / validation / testing hours of source and target languages

	Language	Train	Valid	Test
Source	Russian (ru)	80.61	11.78	12.61
	Welsh (cy)	74.84	4.35	4.25
	Italian (it)	88.74	19.74	20.85
	Basque (eu)	73.26	7.46	7.85
	Portuguese (pt)	37.40	5.40	5.85
Target	Romanian (ro)	3.04	0.42	1.66
	Czech (cs)	20.66	2.84	3.13
	Breton (br)	2.84	1.51	1.75
	Arabic (ar)	7.87	2.01	2.09
	Ukrainian (uk)	17.35	2.30	2.36

4.5.2 Compared Approaches

I consider the following fine-tuning-based approaches as well as both end-to-end and conventional hybrid models and trained from random initialization for comparison in this work. To evaluate the efficiency of different methods, I also list numbers of trainable parameters in Table 4.2. It is shown that our MetaAdapter and SimAdapter (and SimAdapter +) only use $\mathbf{2.5}\%$ and $\mathbf{15}\%$ of the training parameters from the full model, respectively, which are significantly parameter-efficient.

Table 4.2: Comparison of number of trainable parameters.

Method	# Trainable Parameters
Hybrid DNN/HMM	14,247K
Full Model	27,235K
Head	77K
Head+(Meta-)Adapter	676K
Head+(Meta-)Adapter+SimAdapter	4,224K

- Hybrid DNN/HMM: Standard hybrid DNN/HMM models are trained with lattice-free MMI [54] criterion using Kaldi [55]. Specifically, I use 9 layers TDNN [56] the acoustic model. The acoustic features are 100-dimensional i-vector [57] and 40-dimensional MFCC. 3-gram language model is applied for decoding. [1]

- Transformer: I train a randomly-initialized Transformer model from scratch.

- Head: I keep the backbone model (LID-42) frozen and train the language-specific head on top of it.

- Full-FT: I fine-tune the full model on every target language individually, resulting in separate 5 models.

- Adapter: I inject and train the vanilla adapters while keeping the backbone model frozen.

- MetaAdapter: I inject the pre-trained MetaAdapter and train it as the vanilla adapters do.

- SimAdapter: I fuse the Adapters of the source languages with the target language to improve the adaptation performance.

- SimAdapter +: Specifically, I combine the MetaAdapter and the SimAdapter (namely SimAdapter +) to evaluate its performance and verify whether MetaAdapter and SimAdapter are compatible.

[1] I did not find proper pronunciation dictionary for Breton. Therefore, only results of the other 4 languages are presented.

4.5.3 Implementation Details

I implement the E2E methods based on the ESPnet [36] codebase [2]. The subword-based LID-42 model is used as the backbone model for adaptation. The acoustic features are extracted following ESPnet. Numbers of SentencePiece [35] subwords are set to 150 and 100 for high- and low-resource languages, respectively.

I use Adam [58] as the optimizer. For the full-model fine-tuning, I follow the same learning rate scheduling strategy as [18] and warmup the learning rate to 0.2 in the first 10 epochs. The total number of training epochs is 200 for full-model fine-tuning and SimAdapter, and 100 for the other methods. Early stopping with patience 10 is adopted except for the training of source heads and adapters. The source languages heads and adapters are trained using a batch size of 1024 and learning rate 0.028. The target heads and adapters are trained using a batch size of 512 and learning rate 0.02. For the SimAdapter, I used a batch size of 128, learning rate $2e-5$ and regularization loss 0.01. I adopt $\eta = 0.01$ for the regularization loss and 1.0 as the guide loss weight γ. The temperature coefficient τ is simply set to 1.0. I train the MetaAdapter for 30 epochs using Adam [58] with $\beta_1 = 0$ in the inner training loop and vanilla stochastic gradient descent (SGD) in the outer loop. The inner adaptation learning rate and initial meta step size μ are 0.028 and 1.0, respectively. The meta step size linearly annealed to 0 over the course of training. The weight of the CTC module λ is set to 0.3 throughout the experiments following ESPnet [36]. Beam size 10 is employed for joint decoding.

I average the word error rate (WER) results on 5 languages to evaluate the overall performance of different methods by default. To reflect the performance on target languages according to their imbalanced test data duration (more test data often represents more training data), I also compute the weighted average

[2]The source code is available at `https://github.com/jindongwang/transferlearning/tree/master/code/ASR/Adapter`

WERs, which is more friendly to the methods that require relatively more training data.

4.5.4 Results and Analysis

Cross-lingual speech recognition

Table 4.3 shows the main results on cross-lingual ASR. The first three columns show the non-fine-tuning-based baselines. First, it can be found that the hybrid DNN/HMM model outperforms Transformer on 2 out of 4 languages (Romanian (ro), Arabic (ar)), and these 2 languages are with least training data. The results indicate that the overfitting issue occurs to the Transformer model. It could further be inferred that even hybrid DNN/HMM has the overfitting problem on the extremely low-resource Romanian language, since lower WER is obtained with the linear head simply trained on top of the frozen but powerful LID-42 backbone.

Table 4.3: Word error rates (WER) on the cross-lingual ASR tasks

Target Language	Hybrid DNN/HMM	Transformer	Head	Full-FT	Adapter	Sim-Adapter	Meta-Adapter	Sim-Adapter+
Romanian (ro)	70.14	97.25	63.98	53.90	48.34	47.37	**44.59**	47.29
Czech ((cs)	63.15	48.87	75.12	34.75	37.93	35.86	37.13	**34.72**
Breton ((br)	-	97.88	82.80	61.71	58.77	**58.19**	58.47	59.14
Arabic ((ar)	69.31	75.32	81.70	47.63	47.31	47.23	46.82	**46.39**
Ukrainian (uk)	77.76	64.09	82.71	**45.62**	50.84	48.73	49.36	47.41
Average	-	76.68	77.26	48.72	48.64	47.48	47.27	**46.99**
+Weighted	-	72.28	77.54	46.72	47.38	46.08	46.12	**45.45**

On the other hand, from the fine-tuning-based approaches presented on the right-hand side, we can observe that the adapters successfully avoid the overfitting problem and outperform the full-model fine-tuning method on 3 very low-resource languages (Romanian, Czech, Arabic). It can be also found that the proposed MetaAdapter and SimAdapter approaches can achieve similar and competitive results on the 5 target languages. Furthermore, I notice that both the MetaAdapter and SimAdapter consistently improve the performance over the adapters and nar-

row the gap with Full-FT on the languages with relatively more training data (Czech and Ukrainian). Meanwhile, the MetaAdapter method performs better on the extremely low-resource languages (ar, ro) and has lower average WER, while SimAdapter shows better results on moderate low-resource languages (br, cs) and obtains lower weighted average WER. Finally, by combining the MetaAdapter with SimAdapter, the SimAdapter + method surpasses all the other approaches and obtains the best average performance on the 5 languages, indicating that the two methods are compatible since they leverage the source information in different ways. Combining the results from Table 4.2 where SimAdapter + only uses 15.5% trainable parameters of the full model, we see that SimAdapter + is both effective and parameter-efficient.

Ablation Study

Impact of different training strategies. I compare the impact brought by different adapter-training strategies, i.e., jointly training the adapter with head and the first two stages of the training strategy proposed in Section **??**. The results are presented in Table 4.4. It is clear that the proposed two-stage training strategy can consistently reduce the WERs of both the adapters and the SimAdapter.

Table 4.4: Comparison of different Adapter training strategies.

Target	Joint	+SimAdapter	Two-stage	+SimAdapter
ro	52.92	53.88	48.34	47.37
cs	39.16	36.79	37.93	35.86
br	65.10	63.37	58.77	58.19
ar	50.53	49.31	47.31	47.23
uk	52.27	48.84	50.84	48.73
Average	52.00	50.44	48.64	47.48
+Weighted	50.35	48.57	47.38	46.08

Impact of pre-training epochs for MetaAdapter

To validate the meta-training effects for the MetaAdapter, I select checkpoints of 5 pre-trained epochs $\{10, 15, 20, 25, 30\}$ and fine-tune them following the same setting as explained in Section 4.5.3. I present the results in Figure 4.5.1. It could be found that the resulted WERs get improved with more pre-training epochs, indicating the effectiveness of meta-learning.

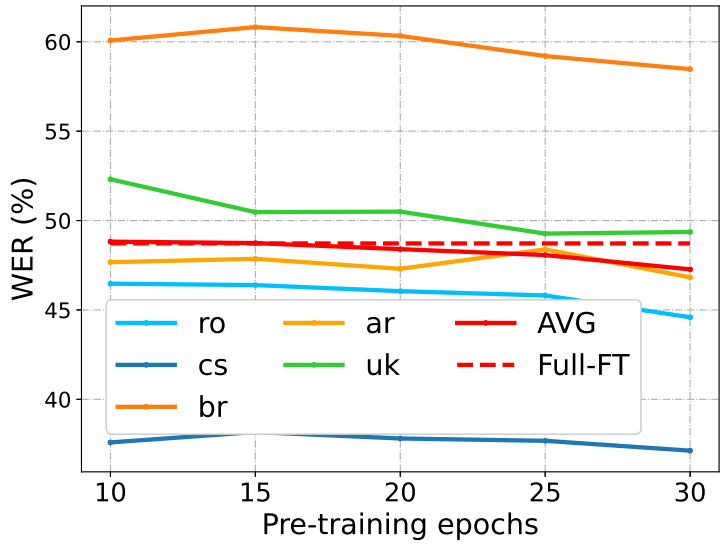


Figure 4.5.1: Pre-training of MetaAdapter

For comparison, I also conduct the same experiment on another adapter pre-trained on source languages using conventional multi-objective learning (MOL) method and visualize the average WERs in Figure 4.5.2. It is clear that with the more pre-training epochs, the MOL-trained adapter tends to overfit on the source data and performs worse on the target languages.

Analyzing the weight of guide loss for SimAdapter

I then analyze the impacts of the weight γ of the proposed guide loss within $\{0, 0.001, 0.01, 0.1, 0.5, 1.0\}$ for the SimAdapter. As shown in Figure 4.5.3, increasing γ generally benefits the improvement of average performance, indicat-

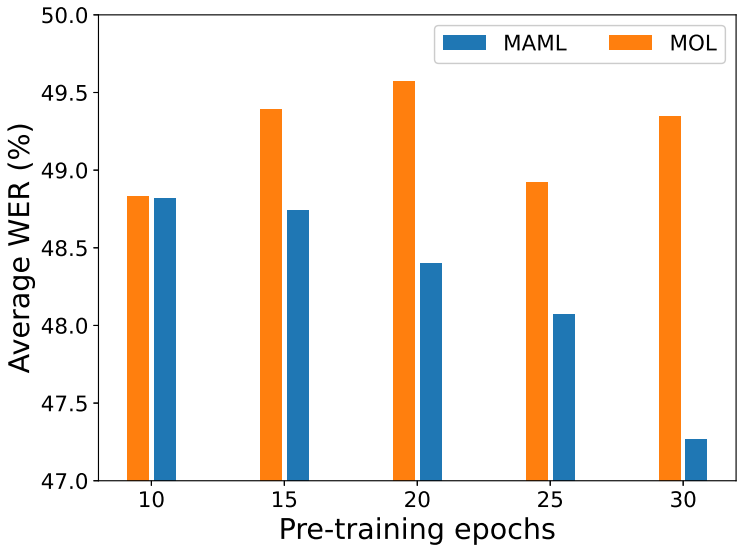


Figure 4.5.2: Comparison between MAML and conventional multi-objective learning (MOL) approach for Adapter pre-training.

ing the correctness of our assumption that SimAdapter layers can learn better under the guidance. We can also observe that when with the increasing of γ, SimAdapter achieves comparable results to full-model fine-tuning at the weight of 0.1 and surpasses it consistently at 0.5 and 1.0.

How much information can be shared across adapters

Although SimAdapter improves the WER results, we do not know whether and how much the other languages can contribute due to the existence of the target language's adapter. Therefore, I fuse the adapters without using the adapters from target languages to see whether additional gains can be obtained with only source adapters. Table 4.5 shows the results. It can be found that even without the target adapter, SimAdapter can still improve the performance for most of the languages except for Romanian, indicating the effectiveness of learning language information from source adapters.

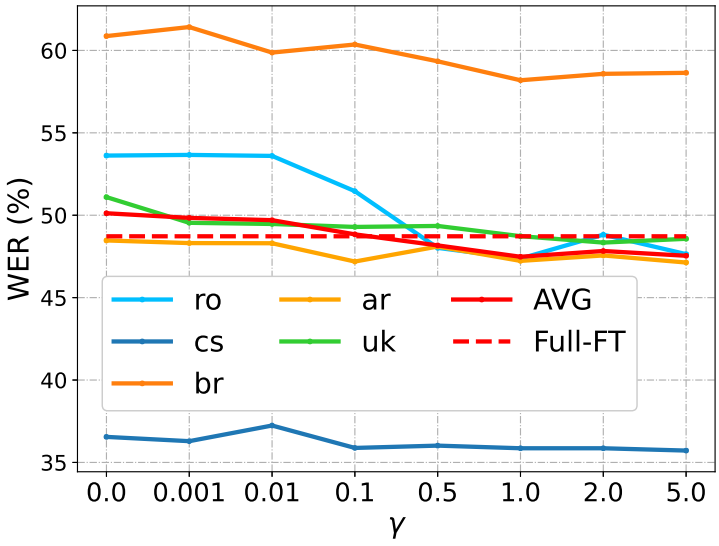


Figure 4.5.3: Guide loss of SimAdapter

Table 4.5: WER results of SimAdapter with or without Adapter L_T. Fusion guide loss is set to 0 for SimAdapter with Adapter L_T.

Target	Head	w/o Adapter L_T	w/ Adapter L_T
ro	63.98	67.83	53.62
cs	75.12	55.06	36.55
br	82.80	77.04	60.87
ar	81.70	64.68	48.47
uk	82.71	69.09	51.10
Average	77.26	66.74	50.12
+Weighted	77.54	65.33	48.39

4.5.5 Attention Visualization

To further show the relationship between source and target languages, I visualize the attention maps for each target language. The attention value reflects their similarities. Figure 4.5.4 shows the results of three different types of languages: (1) without target adapter, (2) with target adapter but no guide loss ($\gamma = 0$), (3) with target adapter and guide loss, and (4) with target MetaAdapter and guide loss.

I take the Ukrainian (uk) as an example. Firstly, from the figure on the left,

we can observe a trend that SimAdapter layers tend to pay more attention to the Russian (ru)'s adapter, which could be because of the linguistic similarity between Ukrainian and Russian. However, after introducing the target adapter, SimAdapter layers obviously turn to focusing more on the target adapter, but there are still diverse attentions across other languages. By introducing the guide loss, the SimAdapter layers is forced to pay more attention to the target adapter and fusing fewer information from other languages.

I also notice that in the first encoder layer, the attention distribution seems to be uniform across the source languages. By analyzing the outputs, I found that the adapters in the first layer tend to keep the backbone representation unchanged via the residual connection. The same phenomenon can also be observed on the Czech (cs) target language. A possible reason could be that the first layer is to extract general acoustic features which is language-independent. Since I observe a similar trend in the first decoder layer (layer 12) that the attention distributions tend to be more distracted, I thus assume that adapters in the bottom layers in both the encoder and decoder are less important for cross-lingual adaptation, which I conduct experiments in next subsection to analyze the performance of fusing different adapters.

4.5.6 Do all Adapter layers need to be fused?

By observing the attention maps, I notice that for some layers, the attention seems to focus solely on the target adapter with a 100% attention score. This phenomenon occurs more frequently in the higher decoder layers, i.e., 12th to 17th layers in Fig. 4.5.4. In such cases, the fusion seems not to be necessary. I doubt whether we can achieve comparable performance while fusing adapters in part of the layers only. Therefore, I conduct the ablation experiments by only fusing part of the layers. The results are presented in Table 4.6. Although some languages (e.g., Breton) can retain the performance by only fusing 2 bottom layers, fusing more layers generally lead to better performance.

Table 4.6: Ablation study of the encoder and decoders

Target	Enc1-Dec1	Enc12-Dec1	Enc12-Dec6
ro	48.39	48.25	**47.37**
cs	37.31	36.30	**35.86**
br	**57.85**	59.08	58.19
ar	47.48	47.34	**47.23**
uk	50.58	48.98	**48.73**
Average	48.32	47.99	**47.48**
+Weighted	47.04	46.55	**46.08**

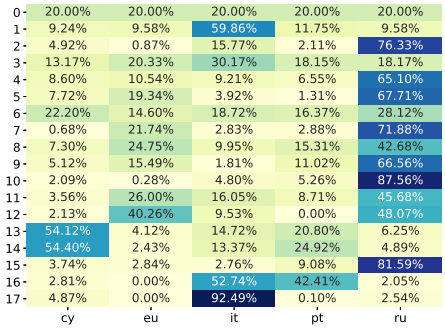
4.5.7 Training and inference time

Finally, I compare the average training time of full-model fine-tuning, MetaAdapter and SimAdapter methods per iteration as well as their inference real-time factor (RTF) on the 5 target languages. The training and decoding are conducted on 1 GeForce RTX 2080 Ti GPU with batch size 64. The results are shown in Table 4.7.

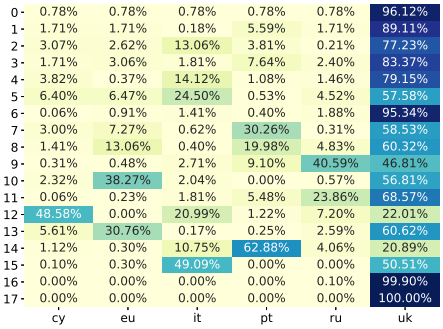
It could be found that the MetaAdapter module significantly accelerates the training process while the SimAdapter introduces minor additional time cost compared with full-model fine-tuning. The RTFs of Full-FT and MetaAdapter are at the same level. The reason that MetaAdapter has slightly lower RTF could be due to its shorter average prediction lengths. On the other hand, the relative RTF increasing of 22.12% brought by SimAdapter is also acceptable.

Table 4.7: Average Training / inference time.

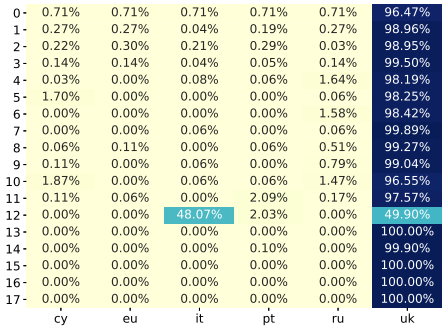
	Training Time (sec.)	RTF
Full-FT	0.253 (-)	0.045 (-)
MetaAdapter	0.143 (43.48%↓)	0.043 (4.06%↓)
SimAdapter	0.263 (3.95%↑)	0.055 (22.12%↑)



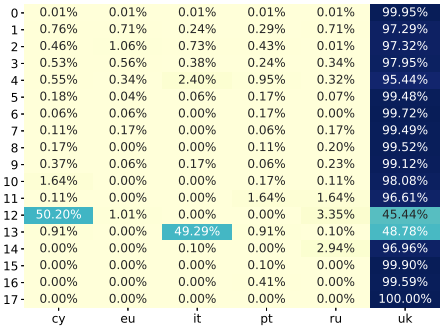
(a) w/o target adapter (uk)



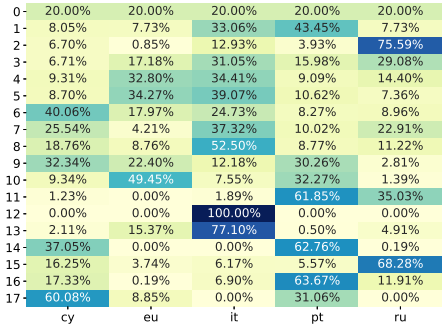
(b) w/o guide loss (uk)



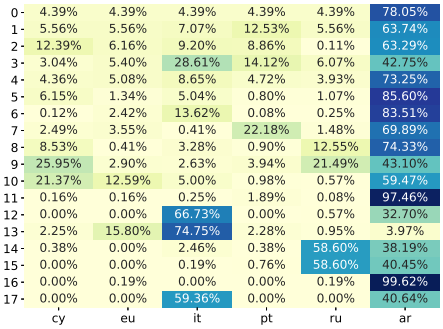
(c) target adapter + guide loss (uk)



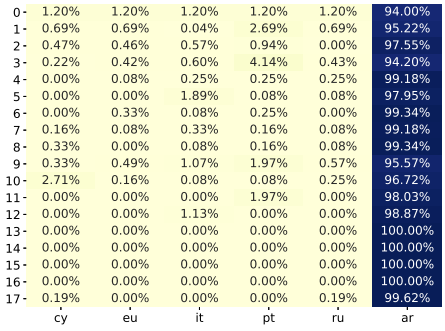
(d) meta-adapter + guide loss (uk)



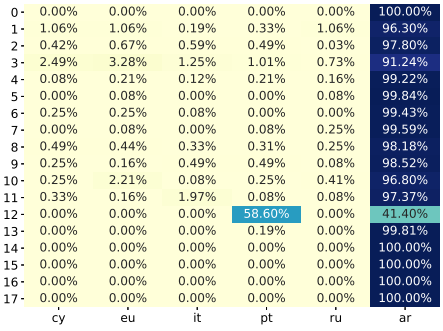
(e) w/o target adapter (ar)



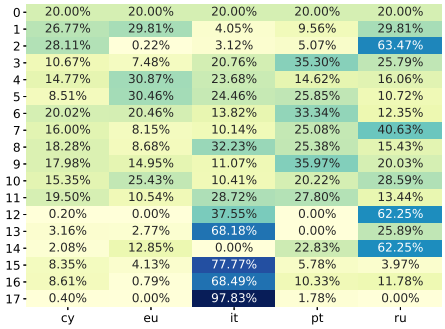
(f) w/o guide loss (ar)



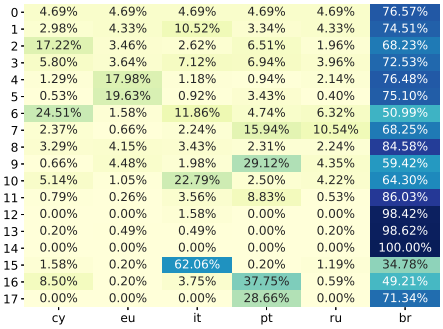
(g) target adapter + guide loss (ar)



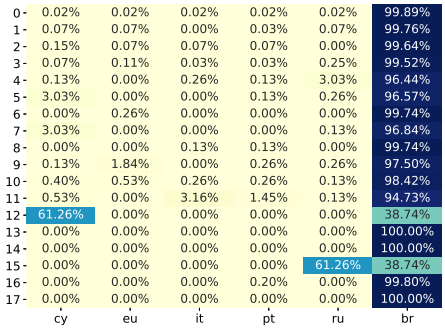
(h) meta-adapter + guide loss (ar)



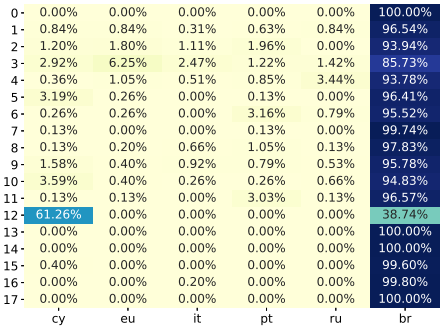
(i) w/o target adapter (br)



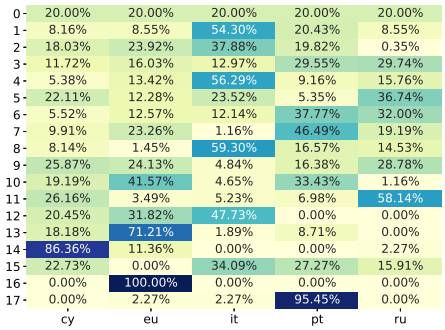
(j) w/o guide loss (br)



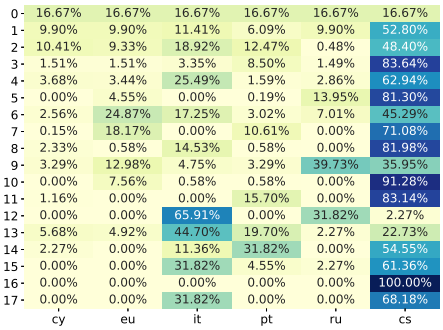
(k) target adapter + guide loss (br)



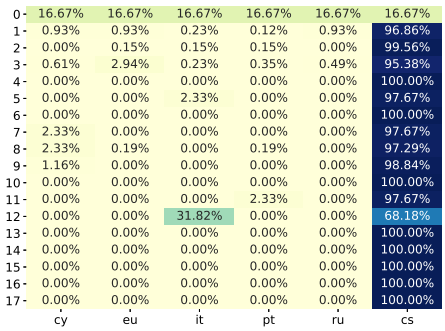
(l) meta-adapter + guide loss (br)



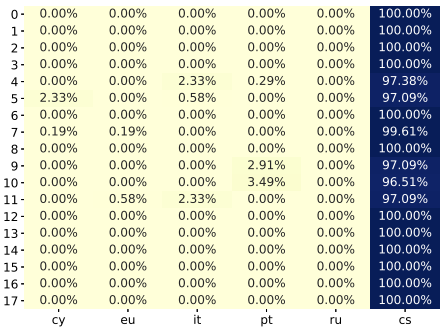
(m) w/o target adapter (cs)



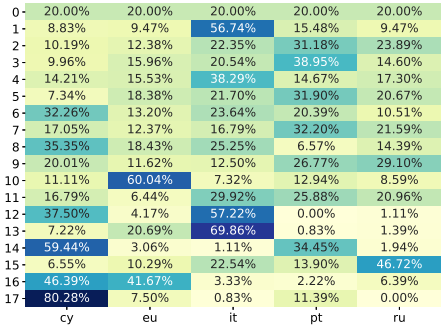
(n) w/o guide loss (cs)



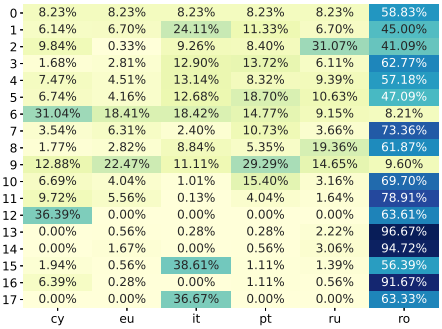
(o) target adapter + guide loss (cs)



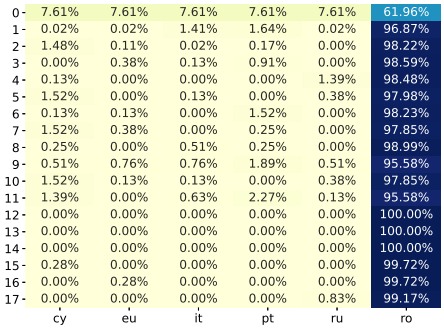
(p) meta-adapter + guide loss (cs)



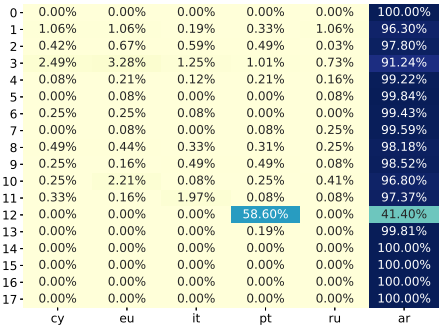
(q) w/o target adapter (ro)



(r) w/o guide loss (ro)



(s) target adapter + guide loss (ro)



(t) meta-adapter + guide loss (ro)

Figure 4.5.4: Attention matrices of five low-resource target languages. A row in the figure denotes a language, whose four settings are: (1) without target adapter, (2) with target adapter but no guide loss ($\gamma = 0$), (3) with target adapter and guide loss, and (4) SimAdapter +. Column index indicates the Transformer layer number, where 0th to 11th layers are encoders, 12th to 17th are decoders. *Best viewed in color and zoomed in.*

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, I propose a new framework, a strong multilingual backbone LID-42, and two novel algorithms MetaAdapter and SimAdapter for cross-lingual low-resource speech recognition. LID-42 is a super multilingual ASR model based on CTC-attention Transformer trained on up to 5,000-hour labeled speech data. MetaAdapter and SimAdapter are adapter-based transfer learning methods for parameter-efficient adaptation. The proposed MetaAdapter implicitly learn from source languages to obtain a proper initialization for any unseen languages. On the other hand, the SimAdapter method explicitly leverages attention mechanism to learn the similarities between the source and target languages during fine-tuning using the adapters. I also show that the two algorithms can be integrated for better performance. Experiments on five low-resource languages from Common Voice dataset demonstrate the superiority of LID-42 and the two algorithms.

5.2 Future Work

In the future, I plan to improve the framework and extend these algorithms to other language families and further improve the training and inference speed of the adaptation methods. I will also investigate larger datasets [59, 60], more effective pre-training approaches [43, 61, 62] and adaptation methods [63, 64] for multilingual pre-training and cross-lingual adaptation to improve the ASR performance on low-resource languages.

Publications

International Conferences

- Wenxin Hou, Jindong Wang, Xu Tan, Tao Qin, Takahiro Shinozaki, ”Cross-domain Speech Recognition with Unsupervised Character-level Distribution Matching”, in Proc. INTERSPEECH, Brno, Czech, August 2021.
- Wenxin Hou, Yidong Wang, Shengzhou Gao and Takahiro Shinozaki, ”Meta-Adapter: Efficient Cross-Lingual Adaptation with Meta-Learning”, in Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Toronto, Ontario, Canada, June 2021.
- Wenxin Hou, Yue Dong, Bairong Zhuang, Longfei Yang, Jiatong Shi and Takahiro Shinozaki, ”Large-Scale End-to-End Multilingual Speech Recognition and Language Identification with Multi-Task Learning”, in Proc. INTERSPEECH, Shanghai, China, October 2020.
- Mingxin Zhang, Tomohiro Tanaka, Wenxin Hou, Shengzhou Gao and Takahiro Shinozaki, ”Sound-Image Grounding Based Focusing Mechanism for Efficient Automatic Spoken Language Acquisition”, in Proc. INTERSPEECH, Shanghai, China, October 2020.
- Shengzhou Gao, Wenxin Hou, Tomohiro Tanaka and Takahiro Shinozaki, ”Spoken Language Acquisition Based on Reinforcement Learning and Word Unit Segmentation”, in Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Barcelona, Spain, May 2020.

Domestic Conferences

- 森滉介, Wenxin Hou, 篠崎隆宏, ”多言語モデルの転移学習による日

本人英語音声認識," 言語処理学会第27回年次大会, E2-3, 2021-3, オンライン開催.

- 篠崎 隆宏, 高 聖洲, ZHANG Mingxin, Hou Wenxin, 田中 智宏, " 言語獲得能力を備えた音声対話エージェントの検討," 人工知能学会，AIチャレンジ研究会，SIG-Challenge-057-13，2020-11, オンライン開催.

- 篠崎 隆宏, 高 聖洲, Hou Wenxin, 田中 智宏, "連続音声からの教師なし単語辞書学習に基づく言語獲得エージェント," 計測自動制御学会 システム・情報部門 学術講演会 2020, GS2-4-4, 2020-11, オンライン開催.

- 篠崎 隆宏, ZHANG Mingxin, 田中 智宏, Wenxin Hou, 高 聖洲, "音声画像グラウンディングに基づいた注意機構による効率的な音声言語獲得," 計測自動制御学会 システム・情報部門 学術講演会 2020, GS2-4-5, 2020-11, オンライン開催.

- 篠崎 隆宏, 高 聖洲, ZHANG Mingxin, Hou Wenxin, 田中 智宏, "相乗的複合学習による効率的な音声言語獲得機構," 第89回 言語・音声理解と対話処理研究会, No11, 2020-9, オンライン開催.

- Shengzhou Gao, Wenxin Hou, Tomohiro Tanaka, Takahiro Shinozaki, " SPOKEN LANGUAGE ACQUISITION BASED ON REINFORCEMENT LEARNING AND WORD UNIT SEGMENTATION," 日本音響学会2020年秋季研究発表会講演論文集, 3-2-8, 2020-9, オンライン開催.

- Hou Wenxin, Dong Yue, ZHUANG BAIRONG, 楊 龍飛, Shi Jiatong, 篠崎隆宏, "超多言語事前学習による低資源音声認識の検討," 日本音響学会2020年秋季研究発表会講演論文集, 2-P1-7, 2020-9, オンライン開催.

Open-Source Projects

- LID-42 Demo, https://github.com/Porridge144/sup-mlt-demo
- EasyEspnet, https://github.com/jindongwang/EasyEspnet
- DeepDA: Deep Domain Adaptation Toolkit, https://github.com/jindongwang/transferlearning/tree/master/code/DeepDA

Bibliography

[1] D. Wang, X. Wang, and S. Lv, "An overview of end-to-end automatic speech recognition," *Symmetry*, vol. 11, no. 8, p. 1018, 2019.

[2] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang *et al.*, "A comparative study on transformer vs rnn in speech applications," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 449–456.

[3] V. Pratap, A. Sriram, P. Tomasello, A. Hannun, V. Liptchinsky, G. Synnaeve, and R. Collobert, "Massively multilingual asr: 50 languages, 1 model, 1 billion parameters," *Proc. Interspeech 2020*, pp. 4751–4755, 2020.

[4] W. Hou, Y. Dong, B. Zhuang, L. Yang, J. Shi, and T. Shinozaki, "Large-Scale End-to-End Multilingual Speech Recognition and Language Identification with Multi-Task Learning," in *Proc. Interspeech 2020*, 2020, pp. 1037–1041. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-2164

[5] E. Chuangsuwanich, "Multilingual techniques for low resource automatic speech recognition," Massachusetts Institute of Technology Cambridge United States, Tech. Rep., 2016.

[6] O. Adams, M. Wiesner, S. Watanabe, and D. Yarowsky, "Massively multilingual adversarial speech recognition," in *Proceedings of the 2019 Confer-*

ence of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 96–108.

[7] S. Zhou, S. Xu, and B. Xu, "Multilingual end-to-end speech recognition with a single transformer on low-resource languages," *arXiv preprint arXiv:1806.05059*, 2018.

[8] J.-Y. Hsu, Y.-J. Chen, and H.-y. Lee, "Meta learning for end-to-end low-resource speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7844–7848.

[9] W. Hou, Y. Wang, S. Gao, and T. Shinozaki, "Meta-adapter: Efficient cross-lingual adaptation with meta-learning," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[10] A. Kannan, A. Datta, T. N. Sainath, E. Weinstein, B. Ramabhadran, Y. Wu, A. Bapna, Z. Chen, and S. Lee, "Large-scale multilingual speech recognition with a streaming end-to-end model," *arXiv preprint arXiv:1909.05330*, 2019.

[11] G. I. Winata, G. Wang, C. Xiong, and S. Hoi, "Adapt-and-adjust: Overcoming the long-tail problem of multilingual speech recognition," *arXiv preprint arXiv:2012.01687*, 2020.

[12] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych, "Adapterfusion: Non-destructive task composition for transfer learning," *arXiv preprint arXiv:2005.00247*, 2020.

[13] W. Hou, H. Zhu, Y. Wang, J. Wang, T. Qin, R. Xu, and T. Shinozaki, "Exploiting adapters for cross-lingual low-resource speech recognition," *arXiv preprint arXiv:2105.11905*, 2021.

[14] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1126–1135.

[15] J. Ke and J. H. Holland, "Language origin from an emergentist perspective," *Applied Linguistics*, vol. 27, no. 4, pp. 691–716, 2006.

[16] P. F. MacNeilage, *The origin of speech*. Oxford University Press, 2010, no. 10.

[17] D. W. Frayer and C. Nicolay, "14 fossil evidence for the origin of speech sounds," 2000.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

[19] 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)*, 2019, pp. 4171–4186.

[20] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2019.

[21] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 7871–7880.

[22] L. Dong, S. Xu, and B. Xu, "Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5884–5888.

[23] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu, "On layer normalization in the transformer architecture," in *International Conference on Machine Learning*. PMLR, 2020, pp. 10 524–10 533.

[24] 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*, 2006, pp. 369–376.

[25] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid ctc/attention architecture for end-to-end speech recognition," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1240–1253, 2017.

[26] S. Watanabe, T. Hori, and J. R. Hershey, "Language independent end-to-end architecture for joint language identification and speech recognition," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 265–271.

[27] S. Toshniwal, T. N. Sainath, R. J. Weiss, B. Li, P. Moreno, E. Weinstein, and K. Rao, "Multilingual speech recognition with a single end-to-end model," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2018, pp. 4904–4908.

[28] B. Li, Y. Zhang, T. Sainath, Y. Wu, and W. Chan, "Bytes are all you need: End-to-end multilingual speech recognition and synthesis with bytes," in

ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019, pp. 5621–5625.

[29] A. Datta, B. Ramabhadran, J. Emond, A. Kannan, and B. Roark, "Language-agnostic multilingual modeling," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8239–8243.

[30] B. Li, R. Pang, T. N. Sainath, A. Gulati, Y. Zhang, J. Qin, P. Haghani, W. R. Huang, and M. Ma, "Scaling end-to-end models for large-scale multilingual asr," *arXiv preprint arXiv:2104.14830*, 2021.

[31] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, "Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline," in *2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*. IEEE, 2017, pp. 1–5.

[32] R. Ardila, M. Branson, K. Davis, M. Kohler, J. Meyer, M. Henretty, R. Morais, L. Saunders, F. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 4218–4222.

[33] K. Maekawa, "Corpus of spontaneous japanese: Its design and evaluation," in *ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003.

[34] Y. Liu, P. Fung, Y. Yang, C. Cieri, S. Huang, and D. Graff, "Hkust/mts: A very large scale mandarin telephone speech corpus," in *International Symposium on Chinese Spoken Language Processing*. Springer, 2006, pp. 724–735.

[35] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *EMNLP 2018*, p. 66, 2018.

[36] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N.-E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen *et al.*, "Espnet: End-to-end speech processing toolkit," *Proc. Interspeech 2018*, pp. 2207–2211, 2018.

[37] J. Cho, M. K. Baskar, R. Li, M. Wiesner, S. H. Mallidi, N. Yalta, M. Karafiat, S. Watanabe, and T. Hori, "Multilingual sequence-to-sequence speech recognition: architecture, transfer learning, and language modeling," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 521–527.

[38] J. Yi, J. Tao, Z. Wen, and Y. Bai, "Adversarial multilingual training for low-resource speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4899–4903.

[39] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[40] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[41] Y. Xiao, K. Gong, P. Zhou, G. Zheng, X. Liang, and L. Lin, "Adversarial meta sampling for multilingual low-resource speech recognition," in *Association for the Advancement of Artificial Intelligence*, 2021.

[42] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," in *International Conference on Learning Representations*, 2019.

[43] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[44] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2790–2799.

[45] A. Bapna and O. Firat, "Simple, scalable adaptation for neural machine translation," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 1538–1548.

[46] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," *EMNLP 2018*, p. 353, 2018.

[47] A. Sharaf, H. Hassan, and H. Daumé III, "Meta-learning for few-shot nmt adaptation," in *Proceedings of the Fourth Workshop on Neural Generation and Translation*, 2020, pp. 43–53.

[48] M. Artetxe, S. Ruder, and D. Yogatama, "On the cross-lingual transferability of monolingual representations," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 4623–4637.

[49] J. Li, R. He, H. Ye, H. T. Ng, L. Bing, and R. Yan, "Unsupervised domain adaptation of a pretrained cross-lingual language model," in *IJCAI*, 2020.

[50] G. Lample and A. Conneau, "Cross-lingual language model pretraining," *arXiv preprint arXiv:1901.07291*, 2019.

[51] Q. Ye and X. Ren, "Zero-shot learning by generating task-specific adapters," *arXiv preprint arXiv:2101.00420*.

[52] O. Weller, N. Lourie, M. Gardner, and M. Peters, "Learning from task descriptions," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 1361–1375.

[53] J. Vanschoren, "Meta-learning: A survey," *arXiv preprint arXiv:1810.03548*, 2018.

[54] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi." in *Interspeech*, 2016, pp. 2751–2755.

[55] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[56] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[57] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[58] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[59] V. Pratap, Q. Xu, A. Sriram, G. Synnaeve, and R. Collobert, "MLS: A Large-Scale Multilingual Dataset for Speech Research," in

Proc. Interspeech 2020, 2020, pp. 2757–2761. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-2826

[60] C. Wang, M. Rivière, A. Lee, A. Wu, C. Talnikar, D. Haziza, M. Williamson, J. Pino, and E. Dupoux, "Voxpopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation," *arXiv preprint arXiv:2101.00390*, 2021.

[61] C. Wang, Y. Wu, Y. Qian, K. Kumatani, S. Liu, F. Wei, M. Zeng, and X. Huang, "Unispeech: Unified speech representation learning with labeled and unlabeled data," in *2021 International Conference on Machine Learning*, July 2021.

[62] 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," *arXiv preprint arXiv:2106.07447*, 2021.

[63] S. Khurana, N. Moritz, T. Hori, and J. Le Roux, "Unsupervised domain adaptation for speech recognition via uncertainty driven self-training," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6553–6557.

[64] W. Hou, J. Wang, X. Tan, T. Qin, and T. Shinozaki, "Cross-domain speech recognition with unsupervised character-level distribution matching," in *Proc. Interspeech*, 2021.