Improving Automatic Speech Recognition Performance for Low-Resource Languages With Self-Supervised Models

Jing Zhao and Wei-Qiang Zhang, *Senior Member, IEEE*

***Abstract*—Speech self-supervised learning has attracted much attention due to its promising performance in multiple downstream tasks, and has become a new growth engine for speech recognition in low-resource languages. In this paper, we exploit and analyze a series of wav2vec pre-trained models for speech recognition in 15 low-resource languages in the OpenASR21 Challenge. The investigation covers two important variables during pre-training, three fine-tuning methods, as well as applications in End-to-End and hybrid systems. First, pre-trained models with different pre-training audio data and architectures (wav2vec2.0, HuBERT and WavLM) are explored for their speech recognition performance in low-resource languages. Second, we investigate data utilization, multilingual learning, and the use of a phoneme-level recognition task in fine-tuning. Furthermore, we explore what effect fine-tuning has on the similarity of representations extracted from different transformer layers. The similarity analyses cover different pre-trained architectures and fine-tuning languages. We apply pre-trained representations to End-to-End and hybrid systems to confirm our representation analyses, which have obtained better performances as well.**

***Index Terms*—HuBERT, low-resource language, OpenASR21, self-supervised learning, wav2vec2.0, WavLM.**

I. Introduction

LOW-RESOURCE languages occupy a large proportion of the languages in the world as 94% of languages are spoken by fewer than 1,000,000 people [1]. It is urgent and necessary to pay attention to the research on these languages to conserve the languages as well as the corresponding cultural heritages. Automatic Speech Recognition (ASR) in low-resource languages remains challenging. There are a series of studies focusing on the low-resource problem [2]–[7]. Compared with common languages, it is much more challenging to build an applicable ASR system for low-resource languages due to the lack of transcribed speech data, language scripts and pronunciation lexicons.

Manuscript received 12 January 2022; revised 24 April 2022 and 12 June 2022; accepted 13 June 2022. Date of publication 20 June 2022; date of current version 14 October 2022. This work was supported in part by the National Natural Science Foundation of China under Grant U1836219 and in part by the National Key R&D Program of China. The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Shinji Watanabe. *(Corresponding author: Wei-Qiang Zhang.)*

The authors are with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: zhaoj20@mails.tsinghua.edu.cn; wqzhang@tsinghua.edu.cn).

Digital Object Identifier 10.1109/JSTSP.2022.3184480

Prior supervised training approaches compensate for resource scarcity through either multilingual training or data augmentation. Multilingual learning alleviates the low-resource problem by drawing the support of other languages' information [4], [5], [8], [9]. For example, the multilingual bottleneck feature is a transfer learning strategy that shows generality among different languages [9]. At the same time, transferring from high-resource languages to low-resource languages has also been explored [10]. Multi-task learning is another way which is believed to improve the generalization performance of a single task [3], [11]. Furthermore, the combination of multi-task learning and multilingual learning is proposed as well [8], which optimizes the ASR system with phoneme and grapheme recognition tasks at the same time. Moreover, meta learning is utilized to build MetaASR with the model-agnostic meta learning algorithm (MAML) for low-resource languages [12].

The second kind of technique is based on the target language. Data augmentation is a necessary and efficient method to cooperate with limited data resources [13]–[15]. Articulatory and stacked bottleneck features are designed to make full use of limited data resources [16]. Recently, Text-To-Speech (TTS) has been validated to compensate for the shortage of speech data with the help of large quantities of text data [15], [17], which is easier to obtain. The dual transformation between TTS and ASR can iteratively boost the accuracy of each other [17].

However, the technologies mentioned above still rely on large quantities of labeled data in many other languages, or have difficulties being robust and universal among different languages [8], or have complicated pipelines. By contrast, self-supervised training strategies, leverage much larger amounts of *unlabeled* speech data with promising results [18]–[27]. Self-supervised training objectives are usually based on reconstruction and contrastive learning. The former predicts or reconstructs a part of the real speech or the corresponding acoustic features [18]–[22], while the latter concentrates on distinguishing the positive sample from the distractors [23]–[26]. There are also self-supervised architectures based on other objectives, for example, HuBERT predicts the pseudo labels from clustering [27], PASE [28] and PASE+ [29] optimize the model with a series of regression and binary discrimination tasks at the same time. Recent work has demonstrated that representations extracted from self-supervised models are to some extent universally useful in a variety of downstream speech processing tasks including content, speaker, semantics, and paralinguistics [30]. In this paper, we mainly focus on the task of ASR in low-resource languages. In particular, prior work demonstrated that the one such self-supervised training frame-work known as wav2vec2.0, performed well in cross-lingual, and resource constrained settings. Therefore, we focus specifically on analyzing aspects of the wav2vec2.0 [26] and other similar frameworks [27], [31] in this work.

The OpenASR21 Challenge[1] is created out of the Intelligence Advanced Research Projects Activity (IARPA) Machine Translation for English Retrieval of Information in Any Language (MATERIAL) program[2] to assess the state of the art of ASR technologies under low-resource language constraints. We have achieved outstanding results in the OpenASR21 Challenge with wav2vec2.0 pre-trained models, which obtain evident improvements compared with traditional hybrid ASR systems. The capabilities tested in the open challenges are expected to ultimately support the MATERIAL task of effective triage and analysis of large volumes of data, in a variety of less-studied languages [32].

In the paper, we conduct a series of studies based on self-supervised learning frameworks to further explore the ASR systems of low-resource languages, including data usage, multilingual learning, fine-tuning strategies, comparisons among wav2vec2.0 [26], HuBERT [27] and WavLM [31], as well as visual analyses for the pre-trained model representations with Centered-Kernel-Alignment (CKA) [33]. In addition, our methods used in the OpenASR21 Challenge are included. The main contributions of this paper are summarized as follows:

- For pre-training, we investigate the ASR performance of several existing pre-trained wav2vec2.0 models in 15 languages, which verifies the effectiveness of multilingual training. In addition, three popular pre-trained model frameworks, wav2vec2.0, HuBERT and WavLM, are compared for their cross-lingual performances in low-resource languages. We find that the wav2vec2.0 and HuBERT architectures are not equally well-suited for all languages. The recently proposed WavLM shows the most promising results in the ASR downstream task of low-resource languages.
- For fine-tuning, we explore three techniques, including utilization of speech data, multilingual learning and an auxiliary phoneme recognition task. We find that it is helpful for the pre-trained model to adapt to the target language by continuing training the self-supervised model with the audio data before fine-tuning. Pre-training on multilingual data using self-supervised objectives is helpful, while fine-tuning using labeled multilingual data is not beneficial. Fine-tuning the pre-trained model for phoneme recognition first can improve speech recognition performance, especially for low-resource conditions.
- To better present our analyses, we use CKA [33] to visualize the relations between representations in the pre-trained models. We present the similarities or differences of layer representations in multiple conditions, including different models, different fine-tuning strategies and different languages, which can provide guidance for the application of pre-trained models in encoder-decoder or hybrid systems.

The rest of the paper is organized as follows. First, the self-supervised models, wav2vec2.0, HuBERT and WavLM, as well as the traditional hybrid system are presented briefly in Section II. Then, in Section III, we describe our experimental framework. Next, we compare the ASR performance of pre-trained models with different pre-training datasets and self-supervised frameworks in Section IV. The proposed fine-tuning methods are introduced in Section V. Moreover, we build End-to-End and hybrid systems to utilize representations extracted from different transformer layers in pre-trained models in Section VI, which confirms our similarity analyses for representations in previous sections and achieves better results as well. The main structure of our work is illustrated in Fig. 1. Finally, conclusions are drawn in Section VII.

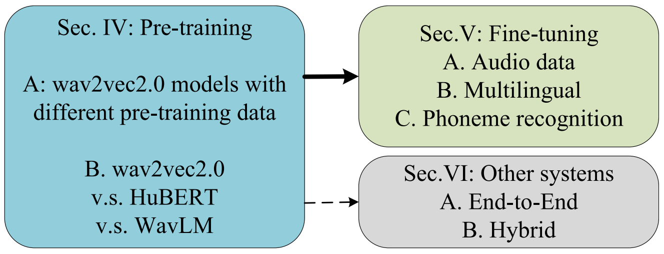


Fig. 1. The overall structure. Our explorations are mainly divided into pre-training and fine-tuning. Some utilizations of pre-trained models in other systems are included as well.

II. Related Works

We incorporate wav2vec2.0 [26], HuBERT [27] or WavLM [31] pre-trained models in our systems. We describe the three models in the following subsections. In addition, we review the traditional hybrid Deep Neural Network / Hidden Markov Model (DNN/HMM) ASR architecture since it is a common choice for ASR of low-resource languages.

A. *wav2vec2.0*

wav2vec2.0 is a framework for self-supervised learning of speech representations, which solves a contrastive task defined over a quantization of the latent representations. The approach relies on distinguishing the positive sample from a set of distractors in the same audio utterance. The positive samples are spans of latent representations masked with a certain proportion, which is similar to masked language modeling [34]. The model is composed of a multi-layer convolutional feature encoder, a context network following the transformer [34] architecture and a quantization module. During pre-training, raw audios are first input to the feature encoder to obtain latent speech representations, which are fed to the transformer [34] encoders next. The outputs are context representations built over continuous speech representations. For contrastive training, the outputs of

[1][Online]. Available: https://sat.nist.gov/openasr21

[2][Online]. Available: https://www.iarpa.gov/index.php/research-programs/material

the feature encoder are discretized to a finite set of speech representations via product quantization. Then, the discriminative process is performed in the discrete representation space [26].

The training objective is to optimize the contrastive loss augmented by a codebook diversity loss. A portion of the feature encoder outputs is masked before being fed into the context network. Then, for each masked time step, the correct quantized representation is distinguished from a set of negative samples, which are uniformly sampled from other masked time steps of the same utterance [26].

B. HuBERT

The Hidden unit BERT (HuBERT) is a BERT-like [34] pre-training approach based on an offline clustering step to generate noisy labels first. The model architecture is nearly the same as wav2vec2.0, while the training task is masked prediction rather than contrastive learning. The acoustic unit discovery model, such as K-means on Mel Frequency Cepstral Coefficients (MFCC) features, is proposed to provide frame-level targets for model training. With the same mask strategies as wav2vec2.0, the cross-entropy loss is computed over the masked time steps after the transformer encoder. Furthermore, the learned representation quality improves dramatically with iteratively refining K-means cluster assignments using learned latent representations for later iterations [27].

C. WavLM

WavLM [31] is a speech self-supervised learning model based on the HuBERT framework, with an emphasis on both spoken content modeling and speaker identity preservation. For the context network, the transformer architecture is equipped with gated relative position bias [35] to improve its capability on recognition tasks. For better speaker discrimination, an utterance mixing training strategy is adopted, where additional overlapped utterances are created in an unsupervised way and incorporated during model training. During pre-training, the model is optimized by mask prediction loss, and trained for two or three iterations like HuBERT. For the first iteration, the targets are obtained by k-means clustering on the MFCC features of the training data. For the second and third iterations, the latent representations generated by the previous iteration model take the place of MFCC features to get new pre-training targets.

D. Hybrid Systems

DNN/HMM based systems usually have more promising performance for low-resource ASR than End-to-End model [36], [37]. A typical workflow is shown in Fig. 2, which consists of pre-processing, data augmentation, feature extraction, training of the acoustic model and the language model, decoding and system fusion. For hybrid systems the pronunciation lexicon is used to map between the predicted acoustic subword units and words, which is essential to low-resource languages because it directly determines the quantity of Out-Of-Vocabulary (OOV) words.

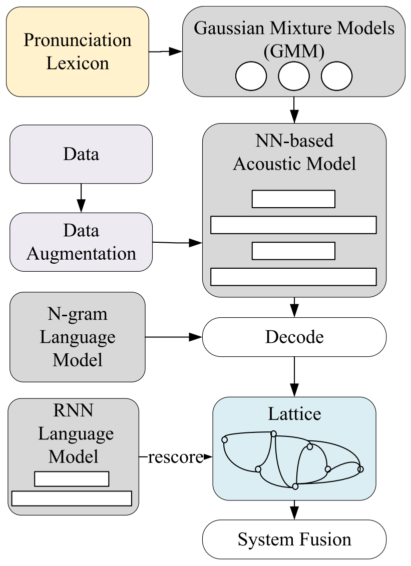


Fig. 2. The workflow of hybrid systems.

For data augmentation, there are some popular methods, such as speed perturbation [13], reverberation and SpecAugment [38]. When training the acoustic model, i-vectors are usually appended to acoustic features to integrate speaker information into the model [39]. For the acoustic model, the proposed TDNN-F [40], CNN-TDNN-F and CNN-TDNN-F-A [41] architectures are used. An N-gram language model is trained with SRILM [42] while a Recurrent Neural Network Language Model (RNNLM) is used to rescore lattices after decoding.

III. Basic Workflow With Self-Supervised Model

A. Pre-Training and Fine-Tuning

The complete speech recognition system we adopt is based on a self-supervised model, and it has two basic steps: pre-training and fine-tuning. Self-supervised pre-training frameworks use varying model architectures and training strategies [18]–[27]. What they have in common is that the pre-training models are hungry for large quantities of audio data, which is extremely time consuming and computationally expensive. It is difficult for most researchers to train a performant self-supervised model from scratch without enough data or computational resources. Therefore, most academic research using self-supervised models has focused instead on fine-tuning these large, pre-trained models on small amounts of data to accomplish specific downstream tasks. Our investigations focus on the the application of existing pre-trained models to the task of low-resource ASR. The wav2vec2.0 [26], HuBERT [27] and WavLM [31] models are selected as the main frameworks for research in this paper for their promising performance in speech recognition.

For speech recognition, a linear classifier maps high dimensional self-supervised representations to the acoustic subwords in a target language. In addition, a word boundary token is usually included in the target units to obtain words from characters or graphemes. The model architecture is similar among wav2vec2.0, HuBERT, and WavLM, so

TABLE I
THE 15 LANGUAGES IN THE OPENASR21 CHALLENGE

Language	Code	Family	Dataset
Amharic	am	Afro-Asiatic	Babel
Cantonese	yue	Sino-Tibetan	Babel
Farsi	fa	Indo-European	MATERIAL
Georgian	ka	Kartvelian	Babel
Guarani	gn	Tupian	Babel
Javanese	jv	Austronesian	Babel
Kazakh	kk	Turkic	Babel
Kurmanji-kurdish	ku	Indo-European	Babel
Mongolian	mn	Mongolic	Babel
Pashto	ps	Indo-European	Babel
Somali	so	Afro-Asiatic	MATERIAL
Swahili	sw	Niger–Congo	Babel
Tagalog	tl	Austronesian	Babel
Tamil	ta	Dravidian	Babel
Vietnamese	vi	Austroasiatic	Babel

the fine-tuning step among the three frameworks is identical. Unlike self-supervised pre-training, fine-tuning only requires a small amount of labeled data. During fine-tuning, the CNN-based feature extractor is frozen with parameters fixed. First, only the newly added projection layer is trained with Connectionist Temporal Classification (CTC) loss [43], after which the transformer-based contextualized encoder is updated together. The optimized objective for fine-tuning is CTC criterion.

B. Datasets

The training datasets in 15 low-resource languages from the OpenASR21 Challenge are used in our following explorations. The OpenASR21 Challenge is implemented as a track of NIST's OpenSAT (Open Speech Analytic Technologies) evaluation series. The task focuses on automatic speech recognition under low-resource language constraints, which consists of 15 languages [32]. The languages are listed in Table I with the corresponding language code and family.

For each language, a 10-hour subset is provided as the training dataset, which is conversational telephone speech in two separate channels with sampling rates of 8 kHz, 44.1 kHz or 48 kHz. The conversations vary in length, up to approximately 10 minutes [32]. In addition, a 10-hour subset (DEV) is provided for system development in the challenge, which is used for both development/validation and evaluation for all the experiments in this paper unless there are special explanations. The datasets for most of the languages stem from the IARPA Babel program [44]. The Somali and Farsi datasets stem from the IARPA MATERIAL program.

IV. OPTIONS FOR PRE-TRAINING

Performance of ASR systems built using self-supervised models has close correlations with pre-trained models [30]. We compare some existing pre-trained models to verify the effects of different pre-training datasets on low-resource speech recognition. Besides, we also investigate the cross-lingual ASR performances of the wav2vec2.0, HuBERT and WavLM models. Major contents of this section are displayed in Fig. 3.

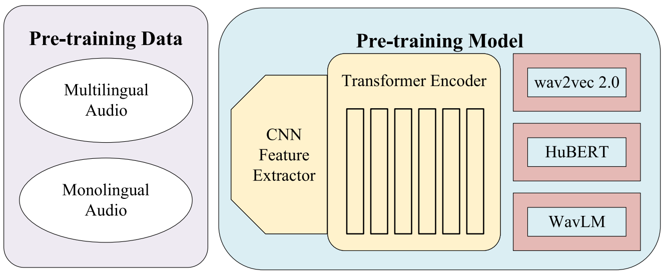


Fig. 3. For the pre-training stage, we focus on the effects of pre-training data, including multilingual and monolingual audio, and model frameworks, including wav2vec2.0, HuBERT and WavLM.

A. Pre-Training Data: Multilingual or Monolingual

For speech recognition of low-resource languages, multilingual training is a common solution [4], [5], [8], [9], [48]. Prior work has shown that representations learned from multilingual self-supervised training tend to correspond to phones, which are often shared across many languages [49]. The large quantities of data from other languages are beneficial to relieve the problem of data sparseness in the target languages. Cross-lingual speech representation learning based on wav2vec2.0 [46], [47] follows a similar idea. In this section, we compare the ASR performances of several publicly available wav2vec2.0 pre-trained model with different training data to investigate the influences of language and speech data quantity.

1) Pre-Trained Models: For self-supervised models, many pre-trained models are trained on English datasets, such as LibriSpeech [50], Libri-light [51] and Common Voice [52]. We choose a LARGE wav2vec2.0 model trained with Libri-light to conduct the following experiments as the only monolingual pre-trained model, which we call w2v-EN-60 k.[3] The architecture contains 24 transformer blocks with model dimensions of 1,024 and inner dimensions of 4,096 for feed-forward network, and 16 attention heads.

For the multilingual pre-trained model, the XLSR-53[4] [46] is trained with 56 k hours of audio data in 53 different languages from 3 datasets (Multilingual LibriSpeech [53], CommonVoice [52], Babel [44]). Most languages in OpenASR21 are included in the pre-training 53 languages. We list the language usage information in Table II for an overall picture. Another multilingual pre-trained model is introduced in [47] with more training data and languages. The XLS-R models are pre-trained on 128 languages and approximately 436 K hours of unlabeled speech data. At the same time, the model parameters can reach 2B. We only experiment on the 300 M model with the same architecture as XLSR-53, which is noted as XLSR-128 in our paper. In addition, there are also related explorations on Indic languages. In [45], CLSRIL-23 is presented as a wav2vec2.0 pre-trained model that learns cross-lingual speech representations from raw audio across 23 Indic languages. Different from

[3][Online]. Available: https://dl.fbaipublicfiles.com/fairseq/wav2vec/wav2vec_vox_new.pt

[4][Online]. Available: https://dl.fbaipublicfiles.com/fairseq/wav2vec/xlsr_53_56k.pt

TABLE II
THE USAGE OF 15 LANGUAGES IN OPENASR21 FOR XLSR-53 PRE-TRAINING

Lang	XLSR-53	Babel Dataset Version	Duration
am	–	IARPA-babel307b-v1.0b-build	43h
yue	Babel	IARPA-babel101b-v0.4c-build	141h
fa	–	–	–
ka	Babel	–	–
gn	–	IARPA-babel305b-v1.0c-build	42h
jv	–	IARPA-babel402b-v1.0b-build	45h
kk	Babel	IARPA-babel302b-v1.0a-build	39h
ku	Babel	IARPA-babel205b-v1.0a-build	41h
mn	CV	IARPA-babel401b-v2.0b-build	46h
ps	Babel	IARPA-babel104b-v0.bY-build	78h
so	–	–	–
sw	Babel	IARPA-babel202b-v1.0d-build	44h
tl	Babel	IARPA-babel106-v0.2g-build	85h
ta	Babel; CV	IARPA-babel204b-v1.1b-build	69h
vi	Babel	IARPA-babel107b-v0.7-build	88h

The second column shows the datasets of the language used for XLSR-53 training. CV stands for CommonVoice. The two columns on the right present information on the extra Babel speech data we use.

TABLE III
THE BASIC INFORMATION OF PRE-TRAINED MODELS INVOLVED

Name	Model	#Params	Pre-Training	
			#langs	Dur
CLSRIL-23 [45]	wav2vec2.0	95M	23	10k
w2v-EN-60k [26]	wav2vec2.0	317M	1	60k
XLSR-53 [46]	wav2vec2.0	317M	53	56k
XLSR-128 [47]	wav2vec2.0	317M	128	436k
HuBERT-EN-60k [27]	HuBERT	317M	1	60k
WavLM-EN-94k [31]	WavLM	317M	1	94k

w2v-EN-60 k and XLSR mentioned above, CLSRIL-23 uses the wav2vec2.0 BASE architecture, which contains 12 transformer blocks with model dimensions of 768, inner dimensions of 3,072 and 8 attention heads. The pre-trained models introduced are listed in Table III.

2) Experimental Settings: We conduct our experiments using the Fairseq toolkit.[5] We keep the same settings for fine-tuning the LARGE models, including w2v-EN-60 k, XLSR-53 and XLSR-128, which follow the configuration in [26]. The learning rate is set to 1×10^{-4}. We optimize with Adam and a tri-state rate schedule where the learning rate is warmed up for the first 10% of updates, held constant for the next 40% and then linearly decayed for the remainder. We fine-tune on a single GPU with a batch of 1.28 M samples. We set the update frequency to 20. For the first 10 k updates, only the output classifier is trained, after which the transformer blocks are also updated. The feature encoder is not trained during fine-tuning. The maximum number of updates is 20 k. Mask probability and mask channel probability are 0.75 and 0.25, respectively. Layer dropout rate is set to 0.1. The 10 h development set is used for both validation and evaluation.

When fine-tuning the CLSRIL-23 BASE model, we set the learning rate to 5×10^{-5} with the same tri-state schedule. Each batch has 3.2 M samples. We use one GPU with update frequency 8. The maximum number of updates is still 20 k with mask probability 0.65 and mask channel probability 0.5. Layer dropout rate is set to 0.05. The settings are the same as these in [26] as well.

For the 15 languages in OpenASR21 datasets, we choose one language from each language family as representatives, except for *yue* to keep the same setting as Section V-B. We fine-tune the pre-trained models with 10 h labeled data of the 10 selected languages respectively. We choose character with diacritics, if any, as the modeling unit for languages with romanized spelling, including Tagalog, Swahili, Javanese, Somali, Guarani, Kurmanji and Vietnamese, while grapheme is utilized for languages without romanized spelling. Specially, Tamil uses Normalization Form D (NFD) for unicode normalization,[6] and the others use Normalization Form C (NFC), such as Pashto and Amharic. The target tokens also include a word boundary token. In particular, the modeling unit of Cantonese is set to characters without a word boundary token since Character Error Rate (CER) is more appropriate than Word Error Rate (WER) to show the performance of Cantonese ASR systems. We adopt case-insensitive scoring in the paper, so we convert all uppercase letters into lowercase letters.

3) Results and Analyses: The ASR results are shown in Table IV. Not surprisingly, the XLSR-128 pre-trained model obtains the lowest WER for every language with the largest pre-training data quantity and variety. Among the three LARGE models, the monolingual model w2v-EN-60 k trained with English audio is obviously in an inferior position when applied to different languages, which also confirms that the multilingual pre-trained models perform better in cross-lingual learning. In the comparison between the BASE and LARGE models, the multilingual BASE model CLSRIL-23 still outperforms the English LARGE model w2v-EN-60 k in 9 of the 10 languages, although the former utilizes less audio data and smaller architecture. The unfair comparison further illustrates the effectiveness of multilingual pre-training.

Since the performances of fine-tuning different pre-trained models vary greatly, we try to examine different model changes before and after fine-tuning. We employ the recently proposed CKA method [33], which is able to measure the relationship between representational similarity matrices, to calculate the correspondence between the transformer encoder layers of wav2vec2.0 pre-trained model. In [33], the CKA similarity has been proved to be invariant to invertible linear transformation, orthogonal transformation and isotropic scaling. For comparing representations of neural networks, it can consistently identify correspondences between layers, not only in the same network trained from different initializations, but across entirely different architectures. For the 5 groups of experiments in Table IV (XLSR-128-miltiFT will be introduced in Section V-B), we obtain the similarity values between representations from transformer layers in the pre-trained model and fine-tuned model. For extracting representations, we randomly choose 100 utterances in total from the DEV sets of the 10 languages and 10 utterances in each language. The results shown are averaged on the 100 utterances. Different numbers of utterances have been tried, including 10, 100 and 500, and we think 100 utterances are enough

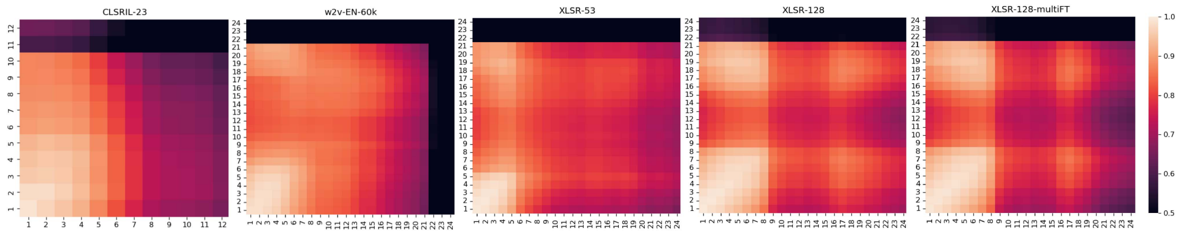
[5][Online]. Available: https://github.com/pytorch/fairseq

[6][Online]. Available: https://unicode.org/reports/tr15/#Norm_Forms

TABLE IV
THE ASR PERFORMANCES (WER) OF 4 DIFFERENT PRE-TRAINED WAV2VEC2.0 MODELS.

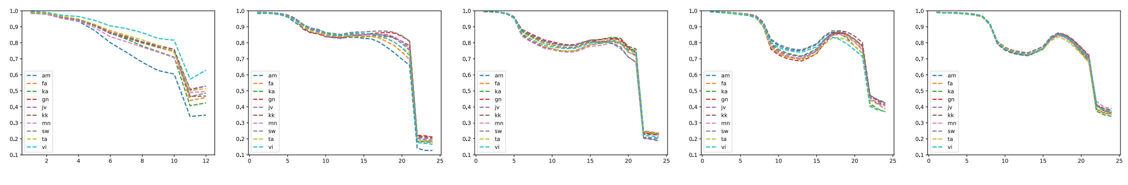
Model	Fine-Tuning		Language										Ave
	#langs	Dur	am	fa	ka	gn	jv	kk	mn	sw	ta	vi	
CLSRIL-23 [45]	1	10h	65.5	60.0	68.7	62.8	68.0	65.5	63.7	53.7	74.8	56.5	63.9
w2v-EN-60k [26]	1	10h	70.5	65.4	65.3	65.0	69.9	65.7	64.3	54.9	79.9	61.2	66.2
XLSR-53 [46]	1	10h	50.4	49.1	50.6	52.3	61.5	53.0	56.5	46.1	72.9	50.7	54.3
XLSR-128 [47]	1	10h	46.0	46.1	45.2	47.8	56.7	49.0	52.4	39.9	70.1	47.8	**50.1**
XLSR-128 [47] (multiFT)	14	140h	62.3	54.9	59.5	63.1	67.0	59.4	62.5	61.2	76.0	56.6	62.2

All the results are obtained on the DEV set inferred without a language model.



(a) CLSRIL-23 (b) w2v-EN-60k (c) XLSR-53 (d) XLSR-128 (e) XLSR-128 (multiFT)

Fig. 4. Similarities between wav2vec2.0 model layers before and after fine-tuning. The vertical and horizontal axes represent the index of transformer blocks in the wav2vec2.0 model before and after fine-tuning respectively. A larger index represents a layer closer to the output.



(a) CLSRIL-23 (b) w2v-EN-60k (c) XLSR-53 (d) XLSR-128 (e) XLSR-128 (multiFT)

Fig. 5. Similarities between the same layer of wav2vec2.0 models before and after fine-tuning for the 10 languages. The horizontal axis represents the index of transformer blocks in the wav2vec2.0 model. A larger number is closer to the output.

since the results do not make evident differences. To keep consistent, we use the same utterances for all the CKA analyses in the paper.

To make the analyses more intuitive, we visualize the values in Fig. 4. The five figures correspond to the five models in Table IV. In each subfigure, the lower left corner corresponds to the first layer, which connects to the CNN feature extractor, and the upper right corner corresponds to the output layer. The horizontal axis represents the index of transformer blocks in the fine-tuned model, while the vertical axis shows that in the pre-trained model without fine-tuning. The darker positions mean less similarity. It is evident that the representations from the last few layers change greatly before and after fine-tuning, regardless of the model architecture or pre-training data, which indicates that there may be more high-level information, such as language characteristics and word meanings, in the layers close to the output end. In contrast, the first few layers near the feature extractor retain more original information after fine-tuning. There is higher similarity among the first few layers, which indicates that some layers in the beginning may be redundant.

What we are more interested in is the diagonals from lower left to upper right. The values of diagonals are CKA similarities between the same transformer layer before and after fine-tuning. We also pay attention to language differences. Therefore, we illustrate the changes in the 5 models for the 10 languages as shown in Fig. 5. For the BASE model CLSRIL-23, the higher layers (larger index) change more after fine-tuning, especially for the top 2 layers, which is also presented in [54]. For the LARGE models, the situations are more complicated in the overall trend. Like the BASE model, the top 3 layers change most, but the middle layers' changes vary in different models. The lines are not simply monotonic at all. Considering different languages, we find that the differences among the 10 languages are not directly affected by the number of languages in the training corpus for the wav2vec2.0 model. By comparing the three figures (b), (c) and (d) in Fig. 5, the variations for different languages are similar. Therefore, monolingual or multilingual pre-trained models have little distinction in language differences. However, in (a), the gaps between languages are more evident, which implies that the smaller model may be more sensitive to language identity. In addition, we present the overall CKA analyses of the 4 models as shown in Fig. 6, which is averaged from the 10 languages in Table IV. The line marked as XLSR-128-multiFT is obtained by multilingual fine-tuning, which is introduced in Section V-B. For the 3 LARGE models, the monolingual model w2v-EN-60 k changes the most in the last 3 layers while the multilingual model XLSR-128 changes least, which is the same order as the ASR performance. Note that comparisons should keep a single variable, which includes the pre-trained model and the fine-tuning method. So the claim is still valid when we take

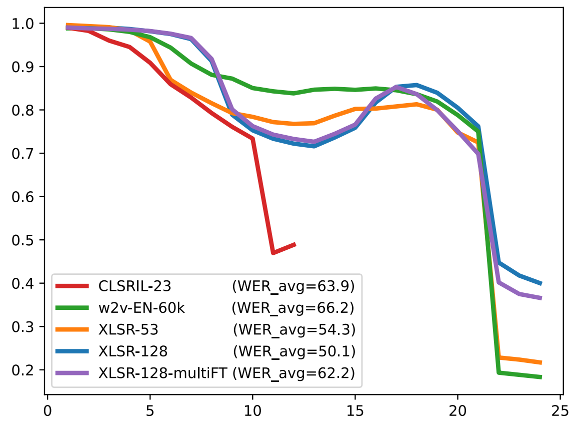


Fig. 6. The CKA similarity analyses averaged by the 10 different languages for the 5 models. The horizontal axis represents the index of transformer blocks in wav2vec2.0.

XLSR-128-multiFT into account. We infer that there may be some connections between the ASR performance and changes of representations during fine-tuning. A direct guess is that the pre-trained model with better performance is likely to change less in the last 3 layers during downstream fine-tuning, since the representations from the original pre-trained model have already been general and effective. The conjecture is also validated in different pre-trained model architectures in Section IV-B2, but we can not deny that there may be a coincidence.

B. Pre-Training Framework: Wav2vec2.0, HuBERT or WavLM

The wav2vec2.0 [26] and HuBERT [27] self-supervised architectures are similar in model structure and have a convolutional waveform encoder, a BERT encoder [34], a projection layer and a code embedding layer. However, they differ a lot in model optimization and training objectives. For WavLM [31], it shares the same training process as HuBERT but modifies the structure of the transformer encoder with gated relative position bias. We have introduced the details in Section II. HuBERT has been validated to perform better than wav2vec2.0 in a series of downstream tasks [27], [30], such as phoneme recognition, automatic speech recognition, speaker identification, intent classification, and emotion recognition. WavLM outperforms HuBERT in the downstream tasks mentioned [31]. However, the tasks and experiments only focus on a single language, English, and have not drawn conclusions on other languages or their cross-lingual performances. Therefore, we try to compare the cross-lingual ASR performances of the three self-supervised architectures in this section.

To ensure fairness, we choose three opensource pre-trained models, w2v-EN-60 k,[7] HuBERT-EN-60 k,[8] and WavLM-EN-94k[9] with similar pre-training corpora and model parameters as the targets. They are trained with 60 k hours of English speech from Libri-light datasets [51]. For WavLM-EN-94 k, the 10 k hours Gigaspeech [55] and 24 k hours VoxPopulil [56] datasets are also applied to training. The model structure is Large with approximately 300 M parameters. The basic information of the models mentioned are listed in Table III. We fine-tune or train our systems with 10 h training data for the 15 languages in the OpenASR21 Challenge to remain consistent with the previous experiments.

1) Experimental Settings: Two groups of experiments with different ASR systems are conducted: one is fine-tuning the pre-trained models, denoted as FT; the other is training enocder-decoder based CTC/Attention ASR systems [57] from scratch using the pre-trained models as fixed feature extractors, denoted as FE.

For fine-tuning, the settings for wav2vec2.0 remain the same as those introduced in Section IV-A2. When fine-tuning the HuBERT model, there are minor adjustments from the wav2vec2.0 settings, with which we obtain better results in our experiments. We use the same learning rate schedule as in [27] where the learning rate is warmed up for the first 10% of updates and then linearly decayed for the remainder. Mask channel probability is 0.5 according to the settings in Fairseq toolkit. The rest are the same as wav2vec2.0. When fine-tuning WavLM model, we utilize the same configurations as wav2vec2.0 model.

For the encoder-decoder based hybrid CTC/Attention ASR system [57], we apply the pre-trained models by extracting representations from them. The pre-trained models are not fine-tuned. The encoder consists of 12 conformer [58] blocks with 2048 hidden states and 8 attention heads. The decoder consists of 6 transformer blocks. We use Weighted Sum (WS) approach [22] to combine representations from all transformer layers in the pre-trained model. The weights of CTC and attention criteria are 0.3 and 0.7 respectively. We use speed perturbation [13] and SpecAugment [38]. The learning rate is set to 0.0002 with Adam optimizer. Maximum training epoch is 50.

2) Results and Analyses: The ASR results of the 15 languages with w2v-EN-60 k, HuBERT-EN-60 k and WavLM-EN-94 k are presented in Table V. The WavLM pre-trained model WavLM-EN-94 k achieves much lower WERs for all the languages in both FT and FE systems. On average, WavLM-EN-94 k pre-trained model has reduced the WER by 18.7% and 12.3% compared with w2v-EN-60 k and HuBERT-EN-60 k when fine-tuning, respectively. Though WavLM-EN-94 k has utilized a larger amount of speech data in English, we do not think the most improvements are attributed to data quantity as the difference in data quantities is not significant. Therefore, the WavLM pre-trained model performs better in cross-lingual low-resource ASR tasks compared with wav2vec2.0 and HuBERT models.

It is interesting to find that there are evident differences among different languages between the wav2vec2.0 and HuBERT models when fine-tuning, while the HuBERT model behaves better for all the 15 languages in FE systems. The 6 languages, gn, jv, ku, so, sw and tl, show lower WERs with the HuBERT pre-trained model, and the remaining 9 languages behave oppositely. Since the differences in the performances of wav2vec2.0 and HuBERT models are evident for the 15 languages except

[7][Online]. Available: https://dl.fbaipublicfiles.com/fairseq/wav2vec/wav2vec_vox_new.pt

[8][Online]. Available: https://dl.fbaipublicfiles.com/hubert/hubert_large_ll60k.pt

[9][Online]. Available: https://github.com/microsoft/unilm/tree/master/wavlm

TABLE V
THE ASR PERFORMANCES (WER) OF WAV2VEC2.0, HUBERT AND WAVLM PRE-TRAINED MODELS IN 15 LOW-RESOURCE LANGUAGES

Sys	**Model**	**Language**															**Ave**
		am	**yue**	**fa**	**ka**	**gn**	**jv**	**kk**	**ku**	**mn**	**ps**	**so**	**sw**	**tl**	**ta**	**vi**	
FT	w2v-EN-60k	**70.5**	**46.0**	**65.4**	**65.3**	65.0	69.9	**65.7**	76.2	**64.3**	**60.7**	70.8	54.9	63.4	**79.9**	**61.2**	**65.3**
	HuBERT-EN-60k	80.7	68.5	72.7	91.3	**60.0**	**61.9**	80.8	**74.4**	85.4	75.2	**64.9**	**44.2**	**53.6**	101.2	61.5	71.7
	WavLM-EN-94k	**48.4**	**37.2**	**50.9**	**47.9**	**52.8**	**58.4**	**52.1**	**69.3**	**54.2**	**51.7**	**61.3**	**42.3**	**50.1**	**70.8**	**48.1**	**53.0**
FE	w2v-EN-60k	62.4	56.2	62.7	61.4	64.2	76.2	70.0	78.8	66.4	62.8	72.4	57.8	67.4	79.2	55.7	66.2
	HuBERT-EN-60k	**61.4**	**46.8**	**59.5**	**60.3**	**62.7**	**70.6**	**65.6**	**76.8**	**63.7**	**61.7**	**70.4**	**55.9**	**63.9**	**77.9**	**53.5**	**63.4**
	WavLM-EN-94k	**54.8**	**44.6**	**53.7**	**54.3**	**57.0**	**65.0**	**58.8**	**72.6**	**57.4**	**56.9**	**65.7**	**48.0**	**58.4**	**74.6**	**49.6**	**58.1**

The system FT means Fine-Tuning the pre-trained models directly. FE means the pre-trained models serve as Front-End in hybrid CTC/Attention ASR systems. The wav2vec2.0 and HuBERT pre-trained models are trained with the same dataset libri-light, while the WavLM pre-trained model scales up the training dataset to 94 k hours with extra Gigaspeech and VoxPopulil datasets. All the results are obtained on the DEV set inferred without a language model. For yue (Cantonese) the results is shown by CER. As WavLM-EN-94 k always obtains the lowest WER, we also bold the lower one in w2v-EN-60 k and HuBERT-EN-60 k.

for Vietnamese, we infer the 2 pre-trained models have preferences for languages or specific characteristics of languages. We investigate the basic information for the 15 languages, including writing system and phonology. We find that all the 6 languages, gn, jv, ku, so, sw and tl, which have lower WER with the HuBERT pre-trained model, use Latin script while the remaining 10 languages except for Vietnamese do not have Latin writing systems. From the perspective of phonology, the 6 languages group owns a smaller number of phonemes on average, which is about 80% of the rest 10 languages group. For example, the Austronesian languages, jv and tl, overall possess smaller phoneme inventories than the world average [59]. Also, gn and so [60], [61] have similar quantities of phonemes as jv and tl, while the quantity of phonemes in ta, with which fine-tuning HuBERT model does not converge, is doubled [62]. It seems that the HuBERT model prefers languages with fewer phonemes or simpler phonology. For the special case, Vietnamese, which uses Latin writing system but has a large phoneme inventory [63], the performances with wav2vec2.0 and HuBERT model are similar. Considering that all the 15 languages obtain lower WERs with the HuBERT model compared with the wav2vec2.0 model in the FE system, which has a more effective structure for the ASR downstream task than the FT system, we conjecture the representations from HuBERT contain more low-level information about acoustics and articulation. So it is helpful and necessary to utilize the representations in a complete ASR system with powerful modeling ability, while wav2vec2.0 is the opposite.

To display the changes of fine-tuning pre-trained models with different languages, we perform CKA similarity analyses between model representations before and after fine-tuning. All the 15 languages for the 3 pre-trained models are shown in Fig. 7. The trends are similar for each self-supervised model in different languages, though the changes are in various degrees. Thus we pay attention to the distinctions among the three self-supervised models. Apparently, the changes to the wav2vec2.0 model are most, especially on the last 3 transformer layers. The curves of WavLM and HuBERT model are closer but the former are higher, which means the parameters of WavLM pre-trained model change less during fine-tuning. Meanwhile, the performances of the ASR task with the WavLM model are the most promising. In the FE system, the results with the wav2vec2.0 pre-trained model are worse than the others for all the 15 languages. Intuitively, the model with better performance may have fewer changes in the top few layers during fine-tuning, which corresponds to our

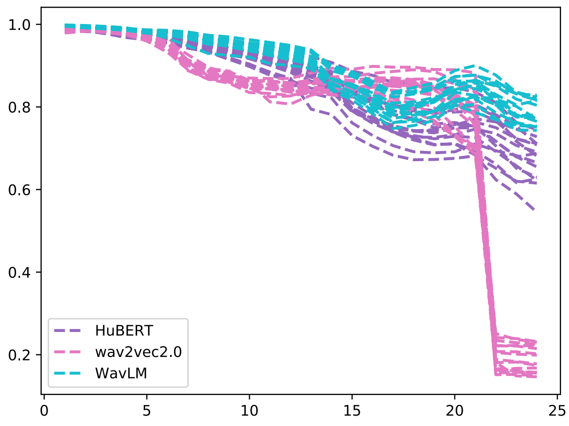


Fig. 7. The CKA similarity of w2v-EN-60 k, HuBERT-EN-60 k and WavLM-EN-94 k before and after fine-tuning for the 15 languages. The horizontal axis represents the index of transformer blocks in the pre-trained models.

inference about the wav2vec2.0 models in Section IV-A3. Consequently, we think the WavLM pre-trained model can provide effective and robust representations for speech recognition of different languages relatively.

We also perform direct similarity analyses between the layers from any two in the three self-supervised models after fine-tuning. As shown in Fig. 8, the three architectures have high similarities except for the top few layers. For different languages, the language with an higher curve in Fig. 8 has a smaller gap in the ASR performance of the two models roughly.

V. FINE-TUNING METHODS

Fine-tuning is an effective method to utilize a pre-trained model for a target downstream task, which is similar to the definition of transfer learning but relatively newer. Therefore, fine-tuning a pre-trained model with limited labeled data is attracting more attention [54], [64]. In this section, we present and analyze 3 fine-tuning methods from different views, including audio data usage, multilingual fine-tuning and auxiliary task, as shown in Fig. 9.

A. *Speech Data Utilization of the Target Language*

In [66], experiments with 6 different English datasets show that using target domain data during pre-training leads to large

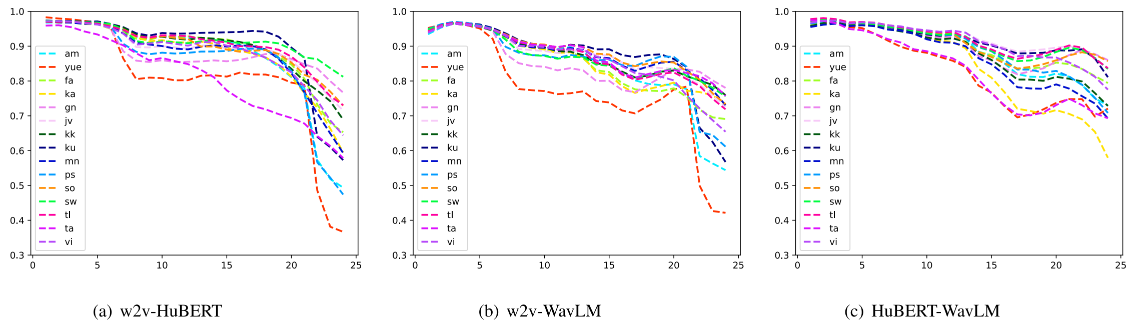


Fig. 8. Similarities between the corresponding layers of pre-trained models (w2v-EN-60 k, HuBERT-EN-60 k and WavLM-EN-94 k) after fine-tuning for 15 languages. The horizontal axis represents the index of transformer blocks in the self-supervised model. A larger number is closer to the output.

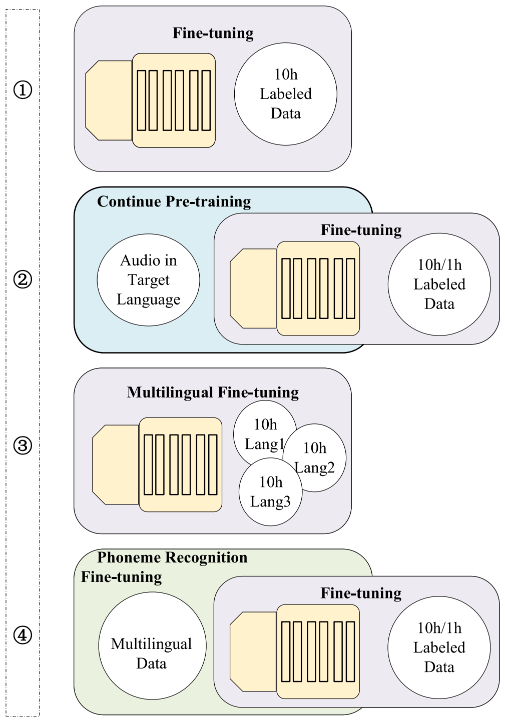


Fig. 9. The four fine-tuning methods. The first one is the basic method, and the remaining three methods are further explored in our paper.

performance improvements across a variety of setups. The domain shift matters when the pre-trained model is applied to downstream tasks, which also implies that the wav2vec2.0 model has learned the information of the speech clips at different levels, including the environment sound, Signal-to-Noise Ratio (SNR), recording equipment, etc. We extend the domain shift research to the multilingual pre-trained model. In addition, it is interesting to verify whether speech data in the target language are still helpful during pre-training in low-resource condition; for example, the total speech data of a target language available is only 10 hours.

Since the pre-trained model is multilingual, to some extent, it has universal characteristics of multiple languages but lacks enough specialty of a specific language. We further train the pre-trained model XLSR-53 with the target language unlabeled speech data from the Build dataset in the Babel program [44] of the corresponding language being processed for the 12 languages, which are listed with the Babel dataset in Table II. In this phase, the whole model is optimized by contrastive loss augmented by a codebook diversity loss [26]. Next, we fine-tune the obtained model with the labeled 10 h data for the language. The two-stage fine-tuning method is helpful to make the pre-trained model adapt to a single language efficiently. We name the strategy FT2, while directly fine-tuning the pre-trained model is denoted as FT. In addition, we deal with the training transcript text in two ways after removing all speech aspects, such as mispronunciations, and non-speech aspects, such as coughs. One is keeping all the words in the transcription of the speech, while the other is filtering out the OOV words by the pronunciation lexicon offered, which is similar to the DNN/HMM hybrid system but there is no symbol to replace the OOV words. According to our experiments, the former method performs better and is marked by (w/ OOV). The results of the three systems introduced above are shown in Table VI.

1) Experimental Settings: For the experimental settings, when continuing the pre-training stage of XLSR-53, most configurations are referred from [26]. There is no layer dropout. We optimize with Adam, warming up the learning rate for the first 32 k updates to a peak of 1×10^{-3}, which is reduced from the settings in [26] for the purpose of fine-tuning rather than training from scratch, and then linearly decaying it. The maximum number of updates is reduced to 100 k considering the smaller data quantity. The Gumbel softmax temperature is annealed from 2 to a minimum of 0.1 by a factor of 0.999995 at every update. The model is trained on a single GPU within 1.2 M tokens a batch. The fine-tuning settings remain the same as those in Section IV-A2.

We decode the systems with a 4-gram language model trained with the text data from Babel as shown in Table II, which are the same as hybrid systems. The Language Model (LM) weight is set to 1 with beam 5 for DEV set. There is little difference between 1 and 2 for the LM weight according to our observations. Experiments show that a larger beam usually obtains a lower WER but also takes a longer time. We set the beam to 500 in the evaluation period of the OpenASR21 Challenge while keeping the beam as 5 for the DEV set to save time.

TABLE VI
THE DEV RESULTS (WER) FROM HYBRID SYSTEMS AND PRE-TRAINED SYSTEMS FOR THE 15 LANGUAGES IN THE OPENASR21 CHALLENGE

Sys	**Language**															**Ave12/15**
	am	**yue**	**fa**	**ka**	**gn**	**jv**	**kk**	**ku**	**mn**	**ps**	**so**	**sw**	**tl**	**ta**	**vi**	
Hybrid_S1	**37.3**	42.1	54.4	42.8	**41.0**	**54.0**	**46.4**	**66.1**	**48.3**	47.0	**55.9**	**34.4**	42.2	**62.0**	47.4	47.4/**48.1**
Hybrid_S2	37.7	42.0	**54.2**	**42.3**	41.8	54.5	47.9	66.3	48.6	47.2	56.6	34.8	42.9	62.9	47.0	47.8/48.4
Hybrid_S3	38.5	**40.7**	55.0	44.3	41.6	54.3	**46.4**	65.6	48.4	**46.3**	56.4	34.5	**41.9**	62.9	**46.4**	**47.3**/48.2
Hybrid_S4	40.9	42.1	58.0	47.2	44.9	57.6	52.9	70.5	52.5	51.2	59.7	37.9	44.1	65.4	50.3	50.9/51.7
FT	44.0	37.0	**46.3**	**44.7**	46.2	53.6	46.5	62.5	47.9	45.2	**53.9**	40.5	45.0	64.3	40.2	47.7/47.9
FT2	40.8	35.8	—	—	42.5	51.0	43.9	60.1	44.7	41.8	—	36.8	41.5	61.8	35.9	44.7/45.4
FT2(w/ OOV)	**38.6**	**35.8**	—	—	**41.3**	**49.8**	**42.1**	**59.5**	**43.9**	**37.9**	—	**34.6**	**39.5**	**60.0**	**36.6**	**43.3/44.3**

There are 4 hybrid systems, S1-S4, among which S1 and S2 employ CNN-TDNN-F and CNN-TDNN-F-A as acoustic models respectively, S3 performs RNNLM rescoring based on the decoding results of S2, and S4 is the augmented system by MUSAN [65] with CNN-TDNN-F or CNN-TDNN-F-A. All 3 fine-tuning systems are based on the wav2vec2.0 pre-trained model XLSR-53. FT means fine-tuning the XLSR-53 pre-trained model with the 10 h labeled data. FT2 continues training the pre-trained model with speech data from Babel in an unsupervised way before fine-tuning. (w/ OOV) means the transcripts used during fine-tuning keep all the words while FT and FT2 eliminate the OOVs in the original text for the train datasets. The results of Cantonese is shown by CER. Ave12 is the average of 12 languages excluding fa, ka and so, and ave15 is the overall average of the 15 languages. For ave15 of FT2 and FT2(w/ OOV), the values of fa, ka and so are from FT.

For the training of hybrid systems, the batch size is set to 128 with 6 epochs in total. The initial learning rate is 0.001 and decays to 0.0001 finally. More details can be found in [41].

2) Results and Analyses: We also present the results of our hybrid systems in the OpenASR21 Challenge for comparison, named as S1-S4 in Table VI. From Table VI, we can find that the simple fine-tuning method with XLSR-53 (FT) performs better than the best single hybrid system on average, although the gap is small between hybrid system S1 and pre-trained system FT. Since the decoding beam is small for quick inference, the pre-trained systems do not show their best performances. For a single language, there is no certain correlation between the system performance and whether the language dataset is used for pre-training. In other words, compared with the best hybrid system, the languages with better performance in the FT system are not always in the pre-training language list for XLSR-53, and some of the languages with worse performance in the FT system have been used for XLSR-53 pre-training as well. We infer that whether the specific language is in the pre-training datasets for the wav2vec2.0 model does not make a big difference to the final ASR results under low-resource conditions. The results also indicate that there are certain common characteristics among different languages and that the multilingual learning method is beneficial to low-resource languages. The FT2 systems achieve evident progress with some extra speech data. On average, the WER decrease from the best single hybrid system to FT2(w/ OOV) is 8.5% and 7.9% for 12 and 15 languages respectively.

3) Data Quantity and Audio Segment: We further explore the influences of the speech data quantity and the audio segment method here. In Table VI, the FT2 system performs better than FT for every single language with a 6.3% relative improvement on average. The unsupervised speech data used for continuing training the XLSR-53 model are only 63 hours on average for the 12 languages. For the low-resource languages, several tens of hours of speech from the target domain can make a difference from the ASR system, although the pre-trained model XLSR-53 has already employed the same part of the data during the original training for 9 of 13 Babel languages.

Since we do not have extra speech data for the languages from the MATERIAL program, we try to perform the FT2 procedure with the same 10 h data. The experimental settings remain the same. The results of Farsi and Somali are shown in Table VII.

TABLE VII
FT2 SYSTEM PERFORMANCES (WER) WITH 10 H DATA FOR TWO LANGUAGES FROM MATERIAL PROGRAM

System	**LM**	**Language**		**Ave**
		fa	**so**	
FT	4-gram	46.3	53.9	50.1
	w/o	50.5	66.0	58.3
FT2	4-gram	45.9	53.1	**49.5**
	w/o	50.0	62.7	**56.4**

TABLE VIII
THE EFFECTS OF SPEECH SEGMENTS DURING TRAINING

Segments	**LM**	**Language**			**Ave**
		am	**jv**	**ku**	
ground truth	4-gram	38.6	49.8	59.5	**49.3**
	w/o	44.9	56.9	66.9	**56.2**
10s	4-gram	39.4	50.0	59.0	49.5
	w/o	47.0	58.0	67.8	57.6

The $ground\ truth$ means the audio is cropped according to transcripts. The $10\,s$ means the audios are cut into 10 s utterances by force. The results are shown by WERs.

The average WER decreases by 1.9% when inferring without a language model, which demonstrates that the wav2vec2.0 model can learn useful and peculiar information about the target language in both pre-training and fine-tuning stages. To make full use of the limited data of the target language, it is acceptable to continue training the pre-trained model with available speech data first.

As for the way of audio segment, it is an inevitable thing in practical situations because we do not have segments for speech audio without transcripts. To determine the effects of audio segments, we re-split the audio data with a fixed segmentation time duration. The segment duration is set to 10 s in the experiments. For example, an 595 s audio would be cut into 59 10s-clips and a 5s-clip. The silent parts are included as well. We randomly pick three languages for the comparison experiments, which are presented in Table VIII. The results are the WER of the DEV set inferred with the 4-gram language model or without language model. It is intuitive that cropping the audio by force hurts the performance to some extent, as shown by the average WER decoding without language model. However, the results with the language model remain nearly the same and are not harmed by direct splitting. We think the LM plays a role in making up for bad portions in the pre-trained model. Therefore,

TABLE IX
THE ASR PERFORMANCES (WER) OF PHONEME FINE-TUNED MODEL XLSR-53-FT40P

Data	Model	Language					Ave
		am	yue	fa	ka	ku	
10h	XLSR-53	**50.4**	45.4	50.5	**50.7**	**71.1**	53.6
	XLSR-53-FT40P	52.0	**38.7**	**48.8**	52.2	**71.1**	**52.6**
1h	XLSR-53	85.4	56.8	**66.7**	70.3	85.7	73.0
	XLSR-53-FT40P	**71.6**	**52.2**	67.9	**69.2**	**81.4**	**68.5**
Train XLSR-53		×	✓	×	✓	✓	—
FT XLSR-53-FT40P		✓	×	×	✓	×	—

All the results are obtained on the DEV set inferred without the language model. For yue (Cantonese) the results is shown by CER. In the last two rows, ✓indicates that the language data are used for pre-training XLSR-53 or fine-tuning XLSR-53-FT40P, while × means not.

it is acceptable to split the speech audio by force without speech activity detection during pre-training if the language model is not too bad.

B. Multilingual Fine-Tuning

Multilingual fine-tuning is a method of multilingual training that is usually applied to low-resource languages [4], [5], [8], [9], [67]. However, for the multilingual pre-trained model XLSR-53, multilingual fine-tuning does not outperform monolingual fine-tuning for speech recognition [46]. We verify this with a pre-trained model XLSR-128, which is trained with more data in more languages.

1) Experimental Settings: We mix up the 10 h training set of 14 languages for fine-tuning except Cantonese, because the best model selection criterion is CER for Cantonese, which is not consistent with WER of other languages. During multilingual fine-tuning, modeling units are de-duplicated after being merged from the 14 languages. We subset one hour from the 10 h DEV set of each language to mix up for validation, which is 14 hours in total. We refer to the experiment setup for fine-tuning with 100 h dataset in [26]. The learning rate is set to 3×10^{-5} with 80 k updates in total. Mask probability and mask channel probability are both 0.5. The other settings remain the same as fine-tuning a LARGE model with 10 h data.

2) Results and Analyses: The results are presented in Table IV, referred to as XLSR-128-multiFT, which performs worse than XLSR-128 and XLSR-53. We also calculate the accuracy of language identification to learn the effect of language recognition. Since all the 14 languages share the same output token set, we obtain the accuracy roughly by counting the number of words in the inference result. If a word in the inference contains any token which does not belong to the language, we treat the word as a mistake of language identification. The average accuracy for the 14 languages are 98.2%, and all the numbers fall in the interval of 95.2%, 99.5%. So the mistakes in language identification are not responsible for the poorer ASR results of most languages directly. Therefore, it is beneficial to concentrate on a single language during fine-tuning so that the model can adapt to the target language better and obtain better performance. The results also indicate that the methods of data utilization for pre-training and fine-tuning are discrepant. Pre-training aims for larger data diversity and more kinds of language to make the pre-trained model more general and robust, while fine-tuning concentrates on a specific domain and a single language to become more compatible and suitable for the target task. In another way, a main difference between the multilingual self-supervised pre-trained model and most traditional multilingual training methods is that the former does not care about language identification information while the latter usually adds expert modules for different languages. Consequently, the downstream task needs to pay more attention to the target language by fine-tuning.

C. Phoneme Fine-Tuning Benefits Low-Resource ASR

In [68], the improvement or hurt relations of different downstream tasks are analyzed, including four aspects of speech: content, speaker, semantics and paralinguistics, which affirms the improvement from phoneme recognition task to automatic speech recognition task. In this paper, we further investigate the effects of phoneme-level task on low-resource ASR under multilingual training conditions. To keep the same baseline as the XLSR-53 pre-trained model, we utilize the multilingually fine-tuned wav2vec2.0 model with phonemes proposed in [69], which is obtained by simultaneously fine-tuning the XLSR-53 pre-trained model to perform phoneme recognition in multiple training languages. The model XLSR-53-FT40P[10] has been fine-tuned with 21 languages in Common Voice [52] (version 6.1) and 19 languages in Babel [44] and is used in our following explorations. During phoneme fine-tuning, International Phonetic Alphabet (IPA) symbols are used, and phonemized transcriptions are obtained by Phonetisaurus phonemizer.[11]

For the low-resource ASR task, we remove the final phoneme output layer in the XLSR-53-FT40P model. Then for a target language, we add a new output layer on the top of the model to keep the same architecture between the XLSR-53 and XLSR-53-FT40P based systems. The detailed experimental settings are the same as Section IV-A2. Since the languages used in XLSR-53 pre-training and XLSR-53-FT40P fine-tuning are not completely consistent, we select 5 languages to cover the four different situations, which is shown in the last two rows of Table IX.

As presented in Table IX, there is no evident difference among the selected languages, which means that the results are not decided by whether the language has already been used in unsupervised pre-training or supervised fine-tuning for phoneme recognition. In addition, we also try to fine-tune the models with a smaller subset, which is one hour of data randomly split from the 10 h training dataset. Most languages achieve lower WERs by fine-tuning the phoneme fine-tuned model XLSR-53-FT40P. On average, the WER decreases by 1.9% when fine-tuning with 10 h labeled data, while a 6.2% decrease is found with 1 h labeled data. To conclude, multilingual fine-tuning for phoneme recognition first does benefit the ASR task, especially for low-resource condition.

[10][Online]. Available: https://dl.fbaipublicfiles.com/fairseq/wav2vec/zero_shot/phonetisaurus_40lang_m10.pt

[11][Online]. Available: https://github.com/AdolfVonKleist/Phonetisaurus

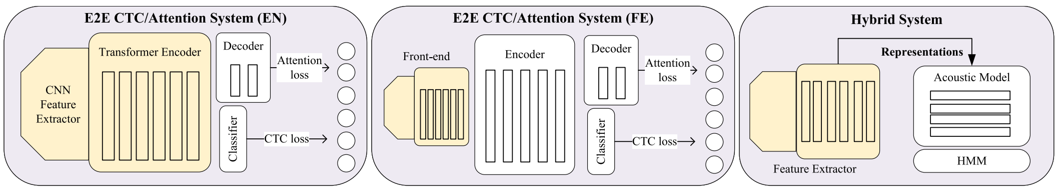


Fig. 10. E2E and hybrid ASR systems based on the wav2vec2.0 pre-trained model. The CTC/Attention architecture is adopted in the E2E ASR system, which can apply the pre-trained model as the encoder (EN) or front-end module (FE).

VI. Other Frameworks With Pre-Trained Models: Choose Suitable Layers for Representations

Representation learning is a popular method in machine learning, as different representations can entangle and hide more or less the different explanatory factors of variation behind the data [70]. Recently, there have been a series of works investigating or assessing speech representations from self-supervised models [30], [71]. At the same time, self-supervised pre-trained models are adopted as backbones or utilized for extracting representations to deal with different downstream tasks [72]–[75].

In this section, we focus on representations from different layers in the pre-trained model for the low-resource ASR task, which is evaluated by End-to-End and hybrid systems.

A. End-to-End System

Recently, the transformer based E2E ASR system has obtained promising performance [76]. However, it does not benefit low-resource languages due to scarce acoustic and text resources. Some works try to relieve the problem with multilingual training [77] by taking advantage of other languages. Similarly, we apply the multilingual pre-trained model to the CTC/Attention-based E2E system [57] to cooperate with low-resource condition and evaluate the pre-trained representations as well. There are two different methods to utilize the pre-trained model: loading the pre-trained model as the encoder and adding a transformer-based decoder to build the system; treating the pre-trained model as a front-end module to extract representations. The overview structures are shown in Fig. 10, and the two methods mentioned above are denoted as EN and FE. We still use the XLSR-53 pre-trained wav2vec2.0 model in this section since it is the first publicly available multilingual pre-trained model and has attracted much attention. The pre-trained models HuBERT-EN-60 k and WavLM-EN-94 k are used as supplements.

1) Experimental Settings: The experiments are conducted with the ESPnet toolkit [78]. In the EN system, which applies the pre-trained model as encoder, we adopt one transformer block with 4 attention heads as decoder, according to our ablation experiments with 1, 2, 3, 4, 6 transformer blocks. The number of linear units is 1024. Only the decoder is trained in the first 100 k updates, after which the top N (we experiment with $N = 0, 6, 12$ and 24) layers in the wav2vec2.0 encoder are trained together. The dropout rate in the decoder is 0.1 and there is no dropout in the encoder. During training, we use the Adam optimizer with a learning rate of 0.001, warmed up in 25000 steps. The maximum training epoch is 50. For the CTC and attention loss, the weights are 0.3 and 0.7, respectively. We keep the best 8 models by the accuracy in the DEV set to finally obtain the average model. The experiments are performed on Farsi with 10 h data for training and 10 h DEV set for validation and evaluation. The output units are obtained by the unigram approach [79] with 500 items in total. Speed perturbation [13] is utilized. No language model is incorporated. The decoding beam is 30 with the CTC weight of 0.4.

TABLE X
The ASR Results (WER) on Farsi DEV With Different Layers of XLSR-53 Frozen in E2E Systems (EN)

#Frozen layers	**0**	**6**	**12**	**18**	**24**
WER	74.5	**73.7**	76.8	82.4	97.6

No LM is incorporated.

In the FE system, we treat the pre-trained model as the feature extractor rather than the encoder. For the encoder, 12 conformer [58] blocks are used with 2 different settings: one adopts a 2048-dimension feed-forward layer and eight-head multi-head attention with 512 dimensions, and kernel size in the Conformer block is set to 31; the other adopts a 2048 dimension feed-forward layer and four-head multi-head attention with 256 dimensions, and Kernel size is set to 15. The former is a common setting while the latter is more suitable for low-resource condition [80]. There are 6 transformer blocks with 2048 dimension feed-forward layers in the decoder. The number of heads is the same as the corresponding encoder. In addition to extracting representations from a single transformer layer in the pre-trained model, we also adopt the Weighted Sum (WS) approach for combining representations from all transformer layers of the pre-trained model. Except for XLSR-53, the fine-tuned model XLSR-53-FT, which is obtained from fine-tuning XLSR-53 model with the same 10 h labeled data, HuBERT-EN-60 k and WavLM-EN-94 k are utilized as well. The parameters of pre-trained models are fixed during system training. Speed perturbation [13] and SpecAugment [38] are utilized. The learning rate is set to 0.0002 with Adam optimizer. The other settings remain the same as in the EN system.

2) Results and Analyses: We show the results of the EN systems in Table X. The frozen layers are counted from the input end. The overall performances of the EN systems are not satisfactory compared with fine-tuning the XLSR-53 model directly. However, we find that it is a better choice to freeze the first 6 transformer layers in the wav2vec2.0 encoder, which corresponds to the similarity analyses in Fig. 5 and Fig. 6. Therefore, when applying a pre-trained model to a target low-resource language, we hope the parameters of the first few transformer layers do not change much, which is beneficial to the final performance and training cost.

For the FE systems, results are shown in Table XI. The overall WERs are lower than these in Table X. The performances with pre-trained models are better than the baselines without

TABLE XI
THE ASR RESULTS (WER) ON FARSI DEV WITH REPRESENTATIONS FROM DIFFERENT LAYERS OF PRE-TRAINED MODELS IN E2E SYSTEMS (FE)

Front-end	**Representation layer**					
	6	**12**	**18**	**21**	**24**	**WS**
None(1)	88.3					
None(2)	68.2					
XLSR-53 (1)	58.3	54.7	**54.4**	54.6	98.3	55.3
XLSR-53 (2)	57.1	53.7	**53.4**	53.6	96.5	53.6
XLSR-53-FT (1)	56.6	48.6	43.8	43.7	44.2	**43.5**
XLSR-53-FT (2)	55.4	48.8	44.3	43.9	44.3	**43.8**
HuBERT-EN-60k (1)	62.9	63.8	63.4	62.5	97.7	**59.5**
WavLM-EN-94k (1)	61.2	54.9	54.8	54.7	55.8	**53.7**

The markers (1)(2) refer to the two settings of conformer block used in the system, successively. No LM is incorporated. WS means Weighted Sum.

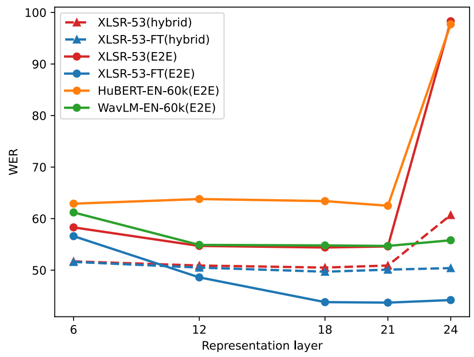


Fig. 11. The ASR results (WER) on Farsi DEV with representations from different layers of wav2vec2.0 model.

pre-trained model as front-end. Compared the two settings of conformer-based encoder, the gaps are narrowed when applying pre-trained model as front-end. Thus, the representations of pre-trained model are able to compensate for the weakness of downstream architecture. For the representation layer selection, it is suitable to extract representations from the 18th to 21st transformer layers with XLSR-53, while the last 6 layers are feasible with XLSR-53-FT. Besides, combining fine-tuned model XLSR-53-FT and conformer-based encoder-decoder system achieves the best performance for a single system without LM in this paper. We illustrate the results of the FE systems with different pre-trained models in Fig. 11 to learn the overall situations more intuitively with the line chart. The solid lines with dots are the FE systems with the first setting for conformer encoder, while dashed lines with triangles represent the hybrid systems, which are introduced next. Focused on the solid lines, we find the pre-trained models are classified into two categories by the performances of the last 3 layers' representations. The XLSR-53 and HuBERT-EN-60 k have a similar trend, which shows the performances with the last few layers' representations are decreasing rapidly. The performances of representations from WavLM-EN-94 k and XLSR-53-FT are less sensitive to the layer index.

B. Hybrid System

In hybrid systems, we regard the wav2vec2.0 pre-trained model as a feature extractor, which is similar to the multilingual bottleneck feature extractor [9], [81]. First, the raw speech

TABLE XII
THE ASR RESULTS (WER) ON FARSI DEV WITH REPRESENTATIONS FROM DIFFERENT LAYERS OF THE WAV2VEC2.0 MODEL IN HYBRID SYSTEMS

wav2vec2.0 model	Representation layer				
	6	12	18	21	24
XLSR-53	51.7	50.9	**50.5**	50.9	60.7
XLSR-53-FT	51.6	50.5	**49.7**	50.1	50.4

audio is input to the wav2vec2.0 pre-trained model, and then we choose a layer to obtain the representations, which serve as features to the acoustic model in the hybrid system. There followed training and decoding of acoustic model. The kaldi toolkit [82] is used for hybrid systems. Note that the frame shift is 20 ms in wav2vec2.0 and 10 ms in the hybrid system, which is inconsistent. We let two adjacent frames of the hybrid system share the representation of one frame in wav2vec2.0 to solve the problem [41]. The XLSR-53 pre-trained model and the fine-tuned model XLSR-53-FT as mentioned in Section VI-A, are utilized.

1) Experimental Settings: We use the TDNN-F architecture [40] as the acoustic model, which is composed of 11 TDNN-F blocks in total with 768 hidden dimensions and 160 bottleneck dimensions. The 100-dim i-vectors [39] are appended to the 1024-dim representations extracted from the wav2vec2.0 model. The settings of training and decoding also remain the same as in Section V-A1. We perform the experiments with Farsi as well.

2) Results and Analyses: The final results are presented in Table XII, which share a similar overall trend with the End-to-End system. Therefore, we put the results of hybrid systems in Fig. 11 as well. We find that the representations extracted from the 18th to the 21st layer perform better in the six ASR systems. The fine-tuned model XLSR-53-FT obtains a lower WER as expected. For the fine-tuned XLSR-53-FT model, there are few differences among the features extracted from the last 12 transformer layers, while the XLSR-53 model reveals the distinctiveness of the last few layers. Besides, the E2E systems are more sensitive to representations from different pre-trained models or different layers in the same pre-trained model. To conclude, the top seven layers (18-24th layers) in the fine-tuned wav2vec2.0 model (XLSR-53-FT) or WavLM model are more suitable for feature extraction. The top few layers in the pre-trained wav2vec2.0 or HuBERT model are not beneficial to be input to the ASR system directly.

VII. CONCLUSION

We have explored speech recognition systems for low-resource languages with pre-trained wav2vec2.0, HuBERT and WavLM self-supervised models in this paper. We focus on how to improve the performance based on the pre-trained models with only 10 h labeled datasets in 15 low-resource languages. Our systems have also achieved excellent results in the OpenASR21 Challenge. The research includes several aspects: data and architecture for pre-training, methods of fine-tuning, analyses of representations extracted from transformer layers, and applications in E2E and hybrid systems.

To highlight, it is important to make full use of the speech data in the target language by continuing training the self-supervised model first to fit the language better. Multilingual pre-trained models have advantages in zero-shot cross-lingual speech recognition, even though the training corpus and model parameters are both less than the monolingual model. However, multilingual fine-tuning is in an inferior position compared with monolingual fine-tuning. We verify by experiments that the phoneme recognition task can improve the low-resource ASR performance. At the same time, we analyze the relation among representations from transformer layers of the pre-trained model to learn the model changes after fine-tuning for different languages, pre-trained models and fine-tuning methods. On the whole, the self-supervised architecture is the major effect factor for the different similarity distributions of the representations. Generally, there are more evident changes in the last few transformer layers of the pre-trained model after fine-tuning. We present the distinctions among representations from different layers of the pre-trained models. It is applicable to extract effective representations from the 18th-21st transformer layers for all the LARGE pre-trained models investigated in the paper. The results further validate our analyses of representation similarity. We would investigate the self-supervised architectures in a broader view in the future.

REFERENCES

[1] D. M. Eberhard, G. F. Simons, and C. D. Fennig, “Ethnologue: Languages of the world,” SIL International: Dallas, TX, USA. 2021. [Online]. Available: http://www.ethnologue.com

[2] S. Thomas, M. L. Seltzer, K. Church, and H. Hermansky, “Deep neural network features and semi-supervised training for low resource speech recognition,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2013, pp. 6704–6708.

[3] D. Chen and B. K.-W. Mak, “Multitask learning of deep neural networks for low-resource speech recognition,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 23, no. 7, pp. 1172–1183, Jul. 2015.

[4] S. Dalmia, R. Sanabria, F. Metze, and A. W. Black, “Sequence-based multi-lingual low resource speech recognition,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 4909–4913.

[5] H. Xu et al., “A comparative study of BNF and DNN multilingual training on cross-lingual low-resource speech recognition,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Dresden, Germany, 2015, pp. 2132–2136.

[6] Y. Miao, F. Metze, and S. Rawat, “Deep maxout networks for low-resource speech recognition,” in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2013, pp. 398–403.

[7] M. Müller, S. Stüker, and A. Waibel, “Towards improving low-resource speech recognition using articulatory and language features,” in *Proc. Int. Workshop Spoken Lang. Transl.*, vol. 45, 2016, pp. 1–7.

[8] K. D. N., “Multilingual speech recognition for low-resource indian languages using multi-task conformer,” 2021, *arXiv:2109.03969*.

[9] E. Chuangsuwanich, “Multilingual techniques for low resource automatic speech recognition,” Ph.D. dissertation, Dept. Elect. Eng., MIT Press, Cambridge, USA, 2016.

[10] S. Khare et al., “Low resource ASR: The surprising effectiveness of high resource transliteration,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 1529–1533.

[11] D. Chen, B. Mak, C. Leung, and S. Sivadas, “Joint acoustic modeling of triphones and trigraphemes by multi-task learning deep neural networks for low-resource speech recognition,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2014, pp. 5592–5596.

[12] J.-Y. Hsu, Y.-J. Chen, and H.-y. Lee, “Meta learning for end-to-end low-resource speech recognition,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 7844–7848.

[13] T. Ko et al., “Audio augmentation for speech recognition,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Dresden, Germany, 2015, pp. 3586–3589.

[14] A. Ragni et al., “Data augmentation for low resource languages,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Singapore, 2014, pp. 810–814.

[15] R. Gokay and H. Yalcin, “Improving low resource Turkish speech recognition with data augmentation and TTS,” in *Proc. Int. Multi-Conf. Syst., Signals Devices*, 2019, pp. 357–360.

[16] V. M. Shetty et al., “Articulatory and stacked bottleneck features for low resource speech recognition,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Hyderabad, India, 2018, pp. 3202–3206.

[17] J. Xu et al., “LRSpeech: Extremely low-resource speech synthesis and recognition,” in *Proc. ACM Conf. Knowl. Discov. Data Mining*, Virtual Event, CA, USA, 2020, pp. 2802–2812.

[18] Y. Chung et al., “An unsupervised autoregressive model for speech representation learning,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 146–150.

[19] Y. Chung, H. Tang, and J. R. Glass, “Vector-quantized autoregressive predictive coding,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 3760–3764.

[20] A. H. Liu, Y.-A. Chung, and J. Glass, “Non-autoregressive predictive coding for learning speech representations from local dependencies,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3730–3734.

[21] A. T. Liu, S. -W. Yang, P. -H. Chi, P. -C. Hsu, and H. -Y. Lee, “Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, Barcelona, Spain, 2020, pp. 6419–6423.

[22] A. T. Liu, S. Li, and H. Lee, “TERA: Self-supervised learning of transformer encoder representation for speech,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 2351–2366. 2021.

[23] M. Riviere, A. Joulin, P.-E. Mazaré, and E. Dupoux, “Unsupervised pretraining transfers well across languages,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 7414–7418.

[24] S. Schneider et al., “wav2vec: Unsupervised pre-training for speech recognition,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 3465–3469.

[25] A. Baevski, S. Schneider, and M. Auli, “vq-wav2vec: Self-supervised learning of discrete speech representations,” in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–12.

[26] A. Baevski et al., “wav2vec 2.0: A framework for self-supervised learning of speech representations,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Vancouver, Canada, 2020, pp. 1–12.

[27] W.-N. Hsu et al., “HuBERT: Self-supervised speech representation learning by masked prediction of hidden units,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 3451–3460, 2021.

[28] S. Pascual et al., “Learning problem-agnostic speech representations from multiple self-supervised tasks,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 161–165.

[29] M. Ravanelli et al., “Multi-task self-supervised learning for robust speech recognition,” in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 6989–6993.

[30] S.-w. Yang et al., “Superb: Speech processing universal performance benchmark,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 1194–1198.

[31] S. Chen et al., “Wavlm: Large-scale self-supervised pre-training for full stack speech processing,” 2021, *arXiv:2110.13900*.

[32] “OpenASR21 challenge (open automatic speech recognition 2021 challenge) evaluation plan,” 2021. [Online]. Available: https://www.nist.gov/document/openasr21-challenge-evaluation-plan

[33] S. Kornblith et al., “Similarity of neural network representations revisited,” in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3519–3529.

[34] J. Devlin et al., “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. Conf. North American Chapter Assoc. Comput. Linguistics: Human Lang. Technol.*, 2019, pp. 4171–4186.

[35] Z. Chi et al., “Xlm-E: Cross-lingual language model pre-training via electra,” in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2021, pp. 6170–6182.

[36] A. Andrusenko, A. Laptev, and I. Medennikov, “Towards a competitive end-to-end speech recognition for chime-6 dinner party transcription,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 319–323.

[37] J. M. Perero-Codosero, F. M. Espinoza-Cuadros, and L. A. Hernández-Gómez, “A comparison of hybrid and end-to-end ASR systems for the iberspeech-rtve 2020 speech-to-text transcription challenge,” *Appl. Sci.*, vol. 12, no. 2, 2022, Art. no. 903.

[38] D. S. Park et al., “SpecAugment: A simple data augmentation method for automatic speech recognition,” in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Graz, Austria, 2019, pp. 2613–2617.

[39] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using I-vectors," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2013, pp. 55–59.
[40] D. Povey et al., "Semi-orthogonal low-rank matrix factorization for deep neural networks," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 3743–3747.
[41] J. Zhao et al., "Automatic speech recognition for low-resource languages: The THUEE systems for the IARPA openasr20 evaluation," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2021, pp. 335–341.
[42] A. Stolcke, "SRILM - An extensible language modeling toolkit," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, Denver, CO, USA, 2002, pp. 1–4.
[43] A. Graves et al., "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2006, pp. 369–376.
[44] M. J. F. Gales et al., "Speech recognition and keyword spotting for low-resource languages: Babel project research at CUED," in *Proc. Workshop Spoken Lang. Technol. Under-resourced Lang.*, 2014, pp. 16–23.
[45] A. Gupta et al., "CLSRIL-23: Cross lingual speech representations for indic languages," 2021, *arXiv:2107.07402*.
[46] A. Conneau et al., "Unsupervised cross-lingual representation learning for speech recognition," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 2426–2430.
[47] A. Babu et al., "XLS-R: Self-supervised cross-lingual speech representation learning at scale," 2021, *arXiv:2111.09296*.
[48] N. T. Vu, F. Kraus, and T. Schultz, "Rapid building of an ASR system for under-resourced languages based on multilingual unsupervised training," in *Proc. 12th Annu. Conf. Int. Speech Commun. Assoc.*, 2011, pp. 3145–3148.
[49] A. Pasad, J. Chou, and K. Livescu, "Layer-wise analysis of a self-supervised speech representation model," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2021, pp. 914–921.
[50] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2015, pp. 5206–5210.
[51] J. Kahn et al., "Libri-light: A benchmark for ASR with limited or no supervision," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 7669–7673.
[52] R. Ardila et al., "Common Voice: A massively-multilingual speech corpus," in *Proc. Lang. Resour. Eval. Conf.*, 2020, pp. 4218–4222.
[53] V. Pratap et al., "MLS: A largescale multilingual dataset for speech research," in *Proc. INTERSPEECH*, 2020, pp. 2757–2761.
[54] A. Pasad, J.-C. Chou, and K. Livescu, "Layer-wise analysis of a self-supervised speech representation model," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2021, pp. 914–921.
[55] G. Chen et al., "GigaSpeech: An evolving, multi-domain ASR corpus with 10, 000 hours of transcribed audio," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3670–3674.
[56] C. Wang et al., "VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation," in *Proc. Annu. Meeting Assoc. Comput. Linguistics Int. Joint Conf. Natural Lang. Process.*, 2021, pp. 993–1003.
[57] S. Watanabe, T. Hori, S. Kim, J. R. Hershey, and T. Hayashi, "Hybrid CTC/attention architecture for end-to-end speech recognition," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 8, pp. 1240–1253, Dec. 2017.
[58] A. Gulati et al., "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 5036–5040.
[59] R. Blust et al., *The Austronesian Languages*. Asia-Pacific Linguistics, School of Culture, History and Language, Canberra, Australia, 2013.
[60] R. Walker, "Guaraní voiceless stops in oral versus nasal contexts: An acoustical study," *J. Int. Phonetic Assoc.*, vol. 29, no. 1, pp. 63–94. 1999.
[61] J. Saeed, *Somali*, vol. 10. Amsterdam, The Netherlands: John Benjamins Publishing, 1999.
[62] S. B. Steever, *The Dravidian Languages*. Milton, MA, USA: Routledge, 2019.
[63] D. Hwa-Froelich, B. W. Hodson, and H. T. Edwards, "Characteristics of vietnamese phonology," *Amer. J. Speech- Lang. Pathol.*, vol. 11, no. 3, pp. 264–273, Aug. 2002.
[64] Y. Wang, A. Boumadane, and A. Heba, "A fine-tuned wav2vec 2.0/HuBERT benchmark for speech emotion recognition, speaker verification and spoken language understanding," 2021, *arXiv:2111.02735*.
[65] D. Snyder, G. Chen, and D. Povey, "MUSAN: A music, speech, and noise corpus," 2015, *arXiv:1510.08484*.
[66] W.-N. Hsu et al., "Robust wav2vec 2.0: Analyzing domain shift in self-supervised pre-training," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 721–725.
[67] M. Wiesner, D. Raj, and S. Khudanpur, "Injecting text and cross-lingual supervision in few-shot learning from self-supervised models," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2022, pp. 8597–8601.
[68] Y.-C. Chen et al., "Speech representation learning through self-supervised pretraining and multi-task finetuning," 2021, *arXiv:2110.09930*.
[69] Q. Xu, A. Baevski, and M. Auli, "Simple and effective zero-shot cross-lingual phoneme recognition," 2021, *arXiv:2109.11680*.
[70] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.
[71] S. Evain et al., "LeBenchmark: A reproducible framework for assessing self-supervised representation learning from speech," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 1439–1443.
[72] C. Yi, S. Zhou, and B. Xu, "Efficiently fusing pretrained acoustic and linguistic encoders for low-resource speech recognition," *IEEE Signal Process. Lett.*, vol. 28, pp. 788–792, 2021.
[73] H. Yu et al., "Language recognition based on unsupervised pretrained models," *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3271–3275.
[74] D. Seo, H.-S. Oh, and Y. Jung, "Wav2KWS: Transfer learning from speech representations for keyword spotting," *IEEE Access*, vol. 9, pp. 80682–80691, 2021.
[75] H. Gao et al., "Zero-shot cross-lingual phonetic recognition with external language embedding," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 1304–1308.
[76] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 5884–5888.
[77] V. M. Shetty and M. S. M. NJ, "Improving the performance of transformer based low resource speech recognition for Indian languages," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 8279–8283.
[78] S. Watanabe et al., "ESPNet: End-to-end speech processing toolkit," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 2207–2211.
[79] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *Proc. Ann. Meeting Assoc. Comput. Linguistics*, 2018, pp. 66–75.
[80] J. Shi et al., "Leveraging end-to-end ASR for endangered language documentation: An empirical study on Yolóxochitl mixtec," in *Proc. Conf. Eur. Chapter Assoc. Comput. Linguistics: Main Volume*, P. Merlo, J. Tiedemann, and R. Tsarfaty, 2021, pp. 1134–1145.
[81] N. T. Vu, F. Metze, and T. Schultz, "Multilingual bottle-neck features and its application for under-resourced languages," in *Proc. Spoken Lang. Technol. Under-Resourced Lang.*, 2012, pp. 90–93.
[82] D. Povey et al., "The Kaldi speech recognition toolkit," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2011, pp. 1–4.

Jing Zhao received the B.S. degree in electronics science and technology from Tsinghua University, Beijing, China, in 2020. She is currently working toward the master's degree with the Speech and Audio Technology Lab, Department of Electronic Engineering, Tsinghua University. Her research interests include automatic speech recognition and keyword search for low-resource languages.

Wei-Qiang Zhang (Senior Member, IEEE) received the B.S. degree in applied physics from the University of Petroleum, Dongying, China, in 2002, the M.S. degree in communication and information systems from the Beijing Institute of Technology, Beijing, China, in 2005, and the Ph.D. degree in information and communication engineering from Tsinghua University, Beijing, in 2009. From 2016 to 2017, he was a Visiting Scholar with the Center for Computer Research in Music and Acoustics, Stanford University, Stanford, CA, USA.

He is currently an Associate Professor with the Department of Electronic Engineering, Tsinghua University and the Head of the Speech and Audio Technology Lab. His research interests include speech and audio recognition and analysis, signal and information processing, and machine learning.