Interspeech 2022
18-22 September 2022, Incheon, Korea

Distilling a Pretrained Language Model to a Multilingual ASR Model

Kwanghee Choi, Hyung-Min Park

Sogang University, Seoul, Republic of Korea
juice500@sogang.ac.kr,hpark@sogang.ac.kr

Abstract

Multilingual speech data often suffer from long-tailed language distribution, resulting in performance degradation. However, multilingual text data is much easier to obtain, yielding a more useful general language model. Hence, we are motivated to distill the rich knowledge embedded inside a well-trained teacher text model to the student speech model. We propose a novel method called the Distilling a Language model to a Speech model (Distill-L2S), which aligns the latent representations of two different modalities. The subtle differences are handled by the shrinking mechanism, nearest-neighbor interpolation, and a learnable linear projection layer. We demonstrate the effectiveness of our distillation method by applying it to the multilingual automatic speech recognition (ASR) task. We distill the transformer-based cross-lingual language model (InfoXLM) while fine-tuning the large-scale multilingual ASR model (XLSR-wav2vec 2.0) for each language. We show the superiority of our method on 20 low-resource languages of the CommonVoice dataset with less than 100 hours of speech data.[1]
Index Terms: automatic speech recognition, knowledge distillation, cross-modal, multilingual

1. Introduction

Compared to other machine learning domains, obtaining speech data is expensive, ending up in unavoidable performance degradation of the automatic speech recognition (ASR) model. Further, collecting non-English speech data is exceptionally difficult as there are not so many speakers available [1, 2, 3]. In contrast, text models have been successfully developed and widely utilized by improving the performance based on the vast web-crawled corpus [4, 5, 6]. Especially, the rise of end-to-end (E2E) transformer-based models showed numerous successes [7, 8, 9, 10], such as cross-lingual language models [6, 11]. Motivated by this, transformer architecture is becoming increasingly common for E2E ASR models [2, 12, 13, 14, 15, 16, 17], where its generalization performance is still limited compared to that of text models. Furthermore, many approaches have been introduced for multilingual ASR, such as adapter modules [2, 18, 19], multi-head architecture [3, 20, 21], logit adjustment [2], language-dependent batching [18, 20], multi-task training [22, 23], language embeddings [22, 23], and cross-lingual training [12]. However, ASR performance has been limited for minor languages, e.g., languages with less than 100 hours of speech. Hence, it naturally raises the following intuition: *Can we utilize the well-trained cross-lingual language models to increase the performance of multilingual ASR models on low-resource languages?*

One of the main approaches to using one model to help the other is knowledge distillation (KD) [24, 25, 26, 27, 28, 29, 30]. KD transfers the information embedded inside the latent representations of the teacher model to the student model, which improves the predictive performance of the student model without any architectural modifications [24]. Hence, it can enjoy the better of two worlds, easily adding the KD loss in a plug-and-play manner to the existing model training, yet enjoying the better performance with the aid of the teacher model. For the classification models, it is common to distill by minimizing the distance between two output logits of the teacher and student networks [24, 25, 28]. Also, many transformer-specific KD methods have been introduced to distill the information more effectively [27, 28, 29, 30]. [27] minimizes the distance between the attention maps or feature outputs of two transformer models with different sizes. [29] distills the attention maps of a transformer-based audio model to smaller architectures such as convolutional or recurrent neural networks.

Although the above methods concentrate on distilling between the same modalities, some methods bridge the differences between speech and text. [25] distills between text and speech models by minimizing the distance between the classification probabilities, whereas [30] minimizes the distances between the first hidden representations or the output logits. Both works sidestep the discrepancy problem between speech and text modalities by disregarding the sequence-level features. [2] handles the multilingual ASR by directly copying the weights of the cross-lingual language model to the ASR model's transformer decoder. [12] uses k-means clustering to segment the audio into phonemic units, where text input is also phonemicized for adversarial training to distinguish the features between two different modalities. [31] aligns the text and speech embeddings via a learnable linear transform layer in an unsupervised manner. Finally, a simple n-gram language model is commonly applied to the final predictions to improve the ASR performance [12, 16, 32]. We did not utilize the method in our work, but we emphasize that it can be used in parallel with the KD methods. KD methods try to yield better predictions in the first place, where using the n-gram language model concentrates on refining the output predictions further.

However, aligning the sequence-level temporal features between text and speech data remains challenging. The shrink mechanism [33] is commonly used to align temporal information between speech and text. It clusters the neighboring outputs into speech segments while discarding the non-speech segments. [34] propose a model that contains speech and text models as its components, where it concentrates on solving the speech-to-text translation task. [35] introduces an encoder-decoder architecture with the losses that accommodates each module to solve the ASR task. [36] inputs speech and text to the transformer encoder and decoder, respectively, improving the performance of speech translation task.

In this paper, we propose a novel method, ***Distill**ing a **L**anguage model **to** a **S**peech model* (Distill-L2S), to leverage the rich knowledge from a well-trained general text model to increase the multilingual ASR performance, especially for low-resource languages. Our method distills the knowledge by

[1] https://github.com/juice500ml/xlm_to_xlsr

10.21437/Interspeech.2022-716

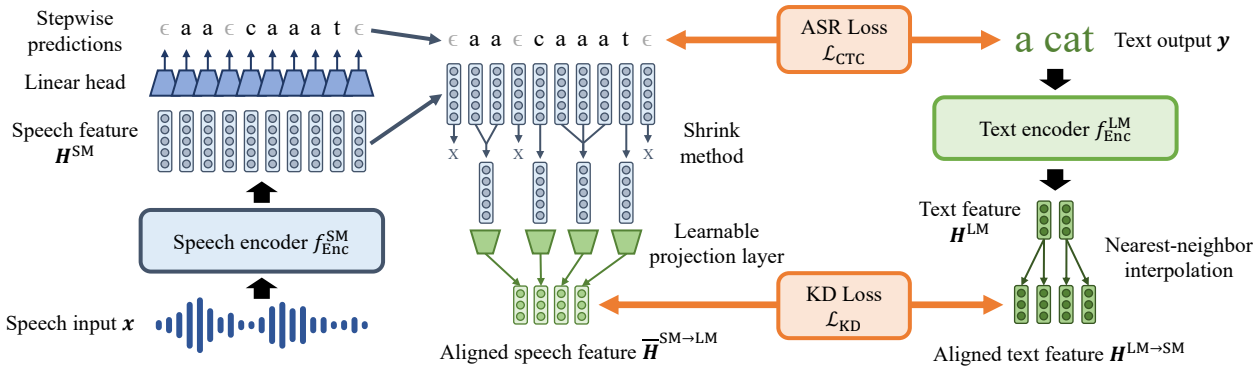


Figure 1: *The concept of our proposed method, Distill-L2S.*

minimizing the differences between text and speech features. We use various methods to mitigate the inherent discrepancy between the two modalities, such as the shrink method [33], nearest-neighbor interpolation [37], and a learnable linear transformation layer [27, 31]. Unlike previous methods, our method avoids additional modification of the pre-existing ASR architectures, only requiring the speech features to be sequence-based. We distill the knowledge of the transformer-based cross-lingual text model, InfoXLM [11], while fine-tuning the transformer-based large-scale ASR model, XLSR-wav2vec 2.0 [32]. We experiment on 20 low-resource languages of the CommonVoice dataset [1] to demonstrate the effectiveness of our method.

2. Proposed Method: Distill-L2S

In this section, we explain our novel method, **Distill**ing a **L**anguage model **to** a **S**peech model (Distill-L2S). We first skim through how ASR models are trained using the CTC loss [38]. Then, we describe the speech and text features used for KD. Finally, we explain our novel loss function that minimizes the distance between features of different modalities. To assuage the different nature of audio and text modalities, we utilize the shrink mechanism [33], nearest-neighbor interpolation [37], and a learnable linear projection layer [27, 31]. Our proposed method is illustrated in Figure 1.

2.1. Training ASR models via the CTC Loss

In this paper, we concentrate on the ASR models that contain hidden states that maintain the sequence-level temporal information. While numerous architectures can satisfy this condition (e.g., CNNs and RNNs) [29], we concentrate on the variants of the wav2vec speech model [16], which utilizes the transformer architecture to achieve state-of-the-art performances [12, 13, 14, 15, 17]. We pass raw audio $\mathbf{x}$ to the speech model f^{SM} to yield the tokenized output text $\mathbf{y}$:

$$\text{Input } \mathbf{x}, \qquad (D_T \times 1) \qquad (1)$$

$$\text{Logits } \mathbf{f}_{ij} = f^{\text{SM}}(\mathbf{x}), \qquad (D^{\text{SM}}_T \times D^{\text{SM}}_V) \qquad (2)$$

$$\text{Probabilities } \mathbf{p}_{ij} = \texttt{softmax}_j(\mathbf{f}_{ij}), \qquad (D^{\text{SM}}_T \times D^{\text{SM}}_V) \qquad (3)$$

$$\text{Predictions } \hat{\mathbf{y}} = \texttt{decode}(\mathbf{p}), \qquad (D^{\text{CTC}}_T \times 1) \qquad (4)$$

where D_T is the length of raw input audio, D^{SM}_T is the output token length after passing through the speech model f^{SM}, D^{SM}_V is the vocabulary size of the speech model's tokenizer, and the $\texttt{softmax}_j$ function normalizes the logits $\mathbf{f}$ in the vocabulary axis D^{SM}_V. We train the model f^{SM} using the CTC loss $\mathcal{L}_{\text{CTC}}$, where `decode` function denotes the decoding procedure of the CTC loss. We defer the detailed description to [38].

2.2. Speech Feature Extraction

The ASR model f^{SM} often yields sequence-level hidden states:

$$\text{Last hidden state } \mathbf{H}^{\text{SM}} = f^{\text{SM}}_{\text{Enc}}(\mathbf{x}), \qquad (D^{\text{SM}}_T \times D^{\text{SM}}_F) \qquad (5)$$

$$\text{Logits } \mathbf{f}_{ij} = f^{\text{SM}}(\mathbf{x}) = \mathbf{H}^{\text{SM}}\mathbf{W}_h + \mathbf{b}_h, \qquad (D^{\text{SM}}_T \times D^{\text{SM}}_V) \qquad (6)$$

where D^{SM}_F is the output feature size of the encoder $f^{\text{SM}}_{\text{Enc}}$. $\mathbf{W}_h$ and $\mathbf{b}_h$ are the learnable weights and biases of the final linear head with shape $D^{\text{SM}}_F \times D^{\text{SM}}_V$ and $1 \times D^{\text{SM}}_V$, respectively. For example, wav2vec [32] feeds the raw audio into multiple 1D convolutional layers, a single linear projection layer, and a transformer encoder to yield the final hidden state. We employ the last hidden state $\mathbf{H}^{\text{SM}}$ that contains the sequence-level information for KD. For brevity, we only describe the linear projection for modeling the logits $\mathbf{f}_{ij}$. However, we emphasize that our KD loss is not limited to the above case as it only requires the hidden state $\mathbf{H}^{\text{SM}}$ and the logits $\mathbf{f}_{\text{ij}}$ to have the same temporal dimension D^{SM}_T.

2.3. Text Feature Extraction

Similar to the above, we extract the features for the target text $\mathbf{y}$ using the transformer encoder of the language model $f^{\text{LM}}_{\text{Enc}}$:

$$\text{Tokenized text } \mathbf{y}_{\text{tok}} = \texttt{tokenize}(\mathbf{y}), \qquad (D^{\text{LM}}_T \times 1) \qquad (7)$$

$$\text{Last hidden state } \mathbf{H}^{\text{LM}} = f^{\text{LM}}_{\text{Enc}}(\mathbf{y}_{\text{tok}}), \qquad (D^{\text{LM}}_T \times D^{\text{LM}}_F) \qquad (8)$$

where we tokenize the raw text $\mathbf{y}$ using the language model's tokenizer. D^{LM}_T is the input text length after tokenization, and D^{LM}_F is the feature size of the hidden states of the transformer encoder $f^{\text{LM}}_{\text{Enc}}$. We defer further details to [10, 11].

2.4. Feature Alignment

To distill the language model $f^{\text{LM}}_{\text{Enc}}$ to the speech model f^{SM}, we freeze the language model's output features $\mathbf{H}^{\text{LM}}$ in eq. 8 and minimize the difference to the speech model's output features $\mathbf{H}^{\text{SM}}$ in eq. 5. However, both features have largely different temporal and feature dimensions. We describe the alignment strategies for both perspectives to yield the final KD loss.

2.4.1. Temporal Dimension Alignment

The inherent difference between speech and text makes temporal alignment difficult. Models trained with the CTC loss $\mathcal{L}_{\text{CTC}}$ are known to yield sparse predictions [38], making the speech feature have lots of "empty" features. However, tokenized text inputs are fed into the language model; hence there are no "empty" text features. We employ the shrink mechanism [33] to mitigate this issue, commonly used to align speech and text features [34, 35, 36]. The mechanism has two valuable properties: it removes the "empty" information and averages within the same neighboring predictions, i.e., a single segment.

$$\bar{\mathbf{H}}^{\text{SM}} = \texttt{shrink}(\mathbf{H}^{\text{SM}}, \mathbf{p}_{ij}), \qquad (\bar{D}_T^{\text{SM}} \times D_F^{\text{SM}}) \quad (9)$$

where $\bar{D}_T^{\text{SM}}$ is the temporal dimension after shrinking.

However, the shrunk speech feature $\bar{\mathbf{H}}^{\text{SM}}$ will still have different temporal dimensions with the corresponding text feature $\mathbf{H}^{\text{LM}}$. To fix the issue, we employ a simple nearest-neighbor interpolation method [37], which is commonly used in image processing. The method naively expands the temporal dimension by duplicating the features multiple times. As the text tokenizer often has a broader vocabulary (i.e., $D_V^{\text{LM}} > D_V^{\text{SM}}$, hence $D_T^{\text{LM}} < D_T^{\text{SM}}$), we expand the text feature's temporal dimension to match that of speech:

$$\mathbf{H}^{\text{LM}\to\text{SM}} = \texttt{interpolate}(\mathbf{H}^{\text{LM}}). \quad (\bar{D}_T^{\text{SM}} \times D_F^{\text{LM}}) \quad (10)$$

By applying the two methods to the speech and the text features, both result in the same temporal dimensionx $\bar{D}_T^{\text{SM}}$.

2.4.2. Feature Dimension Alignment

However, the feature dimensions D_F^{SM} of the speech feature $\bar{\mathbf{H}}^{\text{SM}}$ are still different from D_F^{LM} of the text feature $\mathbf{H}^{\text{LM}\to\text{SM}}$. To fix this, we follow [27], which minimizes the distance between two intermediate representations of the teacher and the student transformer models. It uses the learnable linear projection layer on one of the representations to translate it to the other. The distance between the two is minimized via the mean squared error loss $\texttt{MSE}$. We project the speech feature $\bar{\mathbf{H}}^{\text{SM}}$ to the text feature $\mathbf{H}^{\text{LM}\to\text{SM}}$ using the linear layer:

$$\bar{\mathbf{H}}^{\text{SM}\to\text{LM}} = \bar{\mathbf{H}}^{\text{SM}}\mathbf{W}_f + \mathbf{b}_f, \qquad (\bar{D}_T^{\text{SM}} \times D_F^{\text{LM}}) \quad (11)$$

where $\mathbf{W}_f$ and $\mathbf{b}_f$ is the learnable weights and biases of the projection layer with dimensions $D_F^{\text{SM}} \times D_F^{\text{LM}}$ and $1 \times D_F^{\text{LM}}$, respectively. Finally, the dimensions of eq. 10 and 11 are successfully matched to minimize the distances between them directly.

2.5. Knowledge Distillation Loss

After the temporal and feature dimension alignment, the text feature $\mathbf{H}^{\text{LM}}$ and the speech feature $\mathbf{H}^{\text{SM}}$ are reshaped into $\mathbf{H}^{\text{LM}\to\text{SM}}$ and $\bar{\mathbf{H}}^{\text{SM}\to\text{LM}}$, respectively, ending up in the same shape. We freeze the teacher language model $f_{\text{Enc}}^{\text{LM}}$ and train only the student speech model f^{SM} using the loss that jointly optimizes the ASR loss $\mathcal{L}_{\text{CTC}}$ and the KD loss as follows:

$$\text{ASR loss } \mathcal{L}_{\text{CTC}}, \quad (12)$$

$$\text{KD loss } \mathcal{L}_{\text{KD}} = \texttt{MSE}(\bar{\mathbf{H}}^{\text{SM}\to\text{LM}}, \mathbf{H}^{\text{LM}\to\text{SM}}), \quad (13)$$

$$\text{Final loss } \mathcal{L} = \mathcal{L}_{\text{CTC}} + \lambda\mathcal{L}_{\text{KD}}, \quad (14)$$

where λ controls the strength of the KD, acting as a trade-off hyperparameter.

3. Experiments

In this section, we verify the effectiveness of our novel method, Distill-L2S. First, we briefly describe the multilingual ASR task and the CommonVoice dataset [1]. Then, we cover the student speech model and the teacher text language model, utilizing the already pretrained models for both. We also illustrate the training details for fine-tuning the student speech model on separate languages, where we closely follow the works of [32]. Finally, similar to [2], we evaluated the results using the character error rate (CER). We conduct further analysis of the results.

3.1. Multilingual ASR

Multilingual speech datasets often encounter the issue of long-tailed language distribution; datasets for minor languages are hard to obtain due to the limited number of speakers [1, 2, 3]. This is also the case for the CommonVoice dataset [1], a crowd-sourced multilingual speech dataset. Specifically, we used version 6.1, which contains 60 different languages. 43 languages contain 1 to 100 hours of speech data. Our method is ideal for applying to the multilingual ASR for low-resource languages as it can leverage the extensive knowledge of the broader text corpus via the cross-lingual language model.

3.2. Teacher Model: InfoXLM

For the teacher model, we employed InfoXLM [11], which is a cross-lingual transformer-based general language model. In our experiment, we used the base model that contains 12 layers and 768 hidden states. InfoXLM is pretrained using the recreated CommonCrawl-100 [4], which contains 94 languages. To avoid the performance degradation in the student model-side, we filtered out the languages with less than 1GB of data within CommonCrawl-100, yielding 20 languages for our experiment.

3.3. Student Model: XLSR-wav2vec 2.0

Similar to [32], we fine-tuned XLSR-wav2vec 2.0, a large-scale transformer-based multilingual ASR model. The model architecture is based on the transformer of 24 layers and 1024 hidden states, a total of 317M parameters. We chose the pretrained model that is trained on 53 languages [32]. The 53-language dataset is constructed by merging multiple datasets, including the CommonVoice dataset. We evaluated our models using the test dataset that was not included in the merged training dataset.

3.4. Training Details

We differed the strength of KD by changing $\lambda \in \{0.0, 0.25, 0.5, 1.0\}$ to monitor the effectiveness of our method. $\lambda = 0.0$ becomes equivalent to using the CTC loss alone, being the baseline for the other cases. The transcriptions were tokenized via the unigram algorithm. We borrowed the settings of [32] with some minor differences. We evaluated the models per every 2000 steps on the validation dataset to keep the best model within the total 20000 training steps. For the learning rate, we only used the peak learning rate of 2×10^{-5}. We chose the cosine learning rate schedule with a warm-up of 2000 steps, slightly different from [32]. Our experiments were implemented on the `transformers` library [39]. Consult the aforementioned code for further detail.

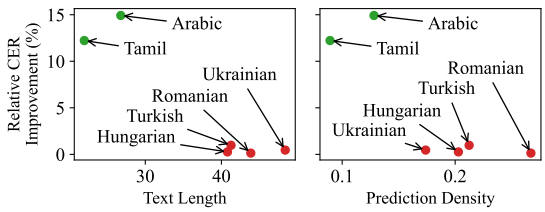
3.5. Experimental Results

Table 1 compares our method on 20 languages with the two baselines, training the model from scratch (weak baseline) and

Table 1: *CER performance (%) comparison on the multiple languages when trained on each langauge of the CommonVoice dataset. The best CER within the same language is written in bold.*

Hours of Speech		GBs of Text	Language	From scratch $\lambda = 0.0$	Fine-tuning $\lambda = 0.0$	Fine-tuning + Distill-L2S (Ours) $\lambda = 0.25$	$\lambda = 0.5$	$\lambda = 1.0$
< 5h	1	14.9	Czech	53.07	7.40	7.43	**7.13**	**7.13**
	1	5.9	Finnish	73.97	11.24	11.46	12.20	**10.87**
	1	74.5	Vietnamese	88.60	32.92	33.18	33.70	**31.99**
	3	1.0	Georgian	65.15	8.83	8.87	8.92	**8.72**
	4	2.3	Lithuanian	69.15	10.99	**10.44**	10.70	10.82
< 10h	5	86.8	Japanese	100.00	100.00	100.00	100.00	100.00
	7	6.2	Slovenian	68.23	7.57	7.50	**7.48**	7.60
	7	1.3	Latvian	69.63	9.25	**8.97**	9.26	9.21
	8	9.5	Hungarian	67.61	11.49	11.99	11.59	**11.46**
	9	11.0	Romanian	80.17	6.70	6.75	6.81	**6.69**
< 20h	12	33.0	Thai	85.56	13.56	13.30	**13.27**	13.80
	13	13.1	Greek	69.19	10.63	**10.33**	10.42	10.62
	17	17.2	Indonesian	65.37	8.40	8.56	**8.30**	8.40
< 50h	22	56.4	Turkish	90.27	7.59	**7.52**	7.58	7.55
	24	7.9	Tamil	79.39	17.26	15.30	**15.15**	15.18
	27	1.4	Estonian	51.37	8.92	**8.34**	8.71	8.48
	43	13.4	Ukrainian	59.88	8.48	8.46	**8.44**	8.50
< 100h	63	39.4	Portuguese	59.64	5.07	**4.98**	5.07	5.09
	63	25.9	Dutch	56.45	8.33	8.72	8.62	**8.05**
	77	16.1	Arabic	66.37	17.36	17.03	15.73	**14.77**

Figure 2: *Comparison of the languages with more than* 10% *(Green) and less than* 1% *(Red) CER improvements over the corresponding fine-tuning baseline.*



fine-tuning without using our KD method (strong baseline). All the settings have the same architecture and training details. Similar to [2], we did not use the n-gram model to remove its influence over the performance and used the CER to evaluate the predictions directly. Our method wins over the strong baseline on all the language settings except for Japanese, verifying the effectiveness of our novel KD method, Distill-L2S. We suspect that the number of Japanese tokens (1249) is too large for the model to handle. Using the pretrained cross-lingual model significantly upgrades the weak baseline that utilizes monolingual data only, where our method manages to improve the performance even further. Especially, Tamil ($12\% \downarrow$) and Arabic ($15\% \downarrow$) show notable decreases in the CER relative to the strong baseline. However, four languages show less than 1% CER decrease: Romanian, Hungarian, Ukrainian, and Turkish.

To analyze why some languages show limited improvement, we compared the two cases above, languages with $> 10\%$ or $< 1\%$ relative CER improvement in Figure 2. We can observe that both the length of the ground truth text and the ratio of the non-blank predictions (Prediction Density) negatively impact our method's performance. We suspect that as both text and speech features get lengthy, it is more likely for temporal misalignment to occur, showing the limitation of the shrink method and the nearest-neighbor interpolation. Further work on this issue needs to be done to improve the performance further.

We additionally compare our method with other multilingual ASR methods in Table 2. The methods are as follows: conditioning with one-hot language ID vectors (LID) [23], language-specific adapters (LAN) [18], Adapt-and-Adjust (A2) [2], and the strong baseline with $\lambda = 0.0$ (XLSR) [32]. Notably, A2 also uses a cross-lingual language model, adds language-specific and language-agnostic adapters, and adjusts the logits during inference. We can see that our method shows superior performance over the compared methods.

Table 2: *CER performance (%) comparison on Dutch and Turkish. Columns with † are the results directly borrowed from [2].*

Language	LID† [23]	LAN† [18]	A2† [2]	XLSR [32]	Distill-L2S (Ours)
Turkish	12.0	11.8	11.3	7.59	**7.52**
Dutch	16.4	16.9	15.8	8.33	**8.05**

4. Conclusion

Given that multilingual speech data often suffer from the long-tailed distribution of languages and text data are much easier to obtain, we are motivated to utilize the well-trained cross-lingual language model for the speech task. Hence, this paper proposes a novel knowledge distillation method, the Distilling a Language model to a Speech model (Distill-L2S), which utilizes the knowledge embedded inside the teacher text model to enhance the predictive performance of the student speech model. Our method mitigates the discrepancy between features of different modalities by utilizing a set of tricks such as the shrink method, nearest-neighbor interpolation, and the learnable linear projection layer. We show our method's effectiveness by evaluating in a large number of languages, where our method consistently shows superiority over the compared methods.

5. Acknowledgements

This work was partly supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (NRF-2020R1A2B5B01002398) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No. 2021-0-02068, Artificial Intelligence Innovation Hub).

6. References

[1] R. Ardila, M. Branson, K. Davis, M. Kohler, J. Meyer, M. Henretty, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *LREC*, 2020.

[2] G. I. Winata, G. Wang, C. Xiong, and S. Hoi, "Adapt-and-adjust: Overcoming the long-tail problem of multilingual speech recognition," *Interspeech*, 2021.

[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," *Interspeech*, 2020.

[4] G. Wenzek, M.-A. Lachaux, A. Conneau, V. Chaudhary, F. Guzmán, A. Joulin, and É. Grave, "Ccnet: Extracting high quality monolingual datasets from web crawl data," in *LREC*, 2020.

[5] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson, "One billion word benchmark for measuring progress in statistical language modeling," *Interspeech*, 2014.

[6] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," *ACL*, 2020.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, vol. 30, 2017.

[8] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *JMLR*, 2020.

[9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *NeurIPS*, 2020.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *NAACL-HLT*, 2018.

[11] Z. Chi, L. Dong, F. Wei, N. Yang, S. Singhal, W. Wang, X. Song, X.-L. Mao, H. Huang, and M. Zhou, "Infoxlm: An information-theoretic framework for cross-lingual language model pre-training," in *NAACL-HLT*, 2021.

[12] A. Baevski, W.-N. Hsu, A. Conneau, and M. Auli, "Unsupervised speech recognition," *NeurIPS*, 2021.

[13] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE TASLP*, 2021.

[14] S. Sadhu, D. He, C.-W. Huang, S. H. Mallidi, M. Wu, A. Rastrow, A. Stolcke, J. Droppo, and R. Maas, "Wav2vec-c: A self-supervised model for speech representation learning," *Interspeech*, 2021.

[15] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *NeurIPS*, 2020.

[16] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," *Interspeech*, 2019.

[17] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," *ICLR*, 2019.

[18] 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," *Interspeech*, 2019.

[19] W. Hou, Y. Wang, S. Gao, and T. Shinozaki, "Meta-adapter: Efficient cross-lingual adaptation with meta-learning," in *ICASSP*, 2021.

[20] T. Sercu, C. Puhrsch, B. Kingsbury, and Y. LeCun, "Very deep multilingual convolutional neural networks for lvcsr," in *ICASSP*, 2016.

[21] S. Dalmia, R. Sanabria, F. Metze, and A. W. Black, "Sequence-based multi-lingual low resource speech recognition," in *ICASSP*, 2018.

[22] 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 *ICASSP*, 2018.

[23] B. Li, T. N. Sainath, K. C. Sim, M. Bacchiani, E. Weinstein, P. Nguyen, Z. Chen, Y. Wu, and K. Rao, "Multi-dialect speech recognition with a single sequence-to-sequence model," in *ICASSP*, 2018.

[24] G. Hinton, O. Vinyals, J. Dean *et al.*, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[25] W. I. Cho, D. Kwak, J. W. Yoon, and N. S. Kim, "Speech to text adaptation: Towards an efficient cross-modal distillation," *Interspeech*, pp. 896–900, 2020.

[26] Y. Liu, H. Xiong, Z. He, J. Zhang, H. Wu, H. Wang, and C. Zong, "End-to-end speech translation with knowledge distillation," *Interspeech*, 2019.

[27] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "Tinybert: Distilling bert for natural language understanding," *EMNLP (Findings)*, 2020.

[28] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[29] K. Choi, M. Kersner, J. Morton, and B. Chang, "Temporal knowledge distillation for on-device audio classification," *arXiv preprint arXiv:2110.14131*, 2021.

[30] S. Kim, G. Kim, S. Shin, and S. Lee, "Two-stage textual knowledge distillation for end-to-end spoken language understanding," in *ICASSP*, 2021.

[31] Y.-A. Chung, W.-H. Weng, S. Tong, and J. Glass, "Unsupervised cross-modal alignment of speech and text embedding spaces," *NeurIPS*, 2018.

[32] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, "Unsupervised cross-lingual representation learning for speech recognition," *Interspeech*, 2021.

[33] Z. Chen, Y. Zhuang, Y. Qian, and K. Yu, "Phone synchronous speech recognition with ctc lattices," *IEEE TASLP*, 2016.

[34] Y. Liu, J. Zhu, J. Zhang, and C. Zong, "Bridging the modality gap for speech-to-text translation," *arXiv preprint arXiv:2010.14920*, 2020.

[35] C. Yi, F. Wang, and B. Xu, "Ectc-docd: An end-to-end structure with ctc encoder and ocd decoder for speech recognition." in *Interspeech*, 2019.

[36] M. A. Di Gangi, M. Negri, and M. Turchi, "One-to-many multilingual end-to-end speech translation," in *IEEE ASRU*, 2019.

[37] *The OpenCV Reference Manual*, 2nd ed., Itseez, 2014.

[38] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[39] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz *et al.*, "Transformers: State-of-the-art natural language processing," in *EMNLP (Demos)*, 2020.