Dual-path Adaptation of Pretrained Feature Extraction Module for Robust Automatic Speech Recognition

Hao Shi[1], *Tatsuya Kawahara*[1]

[1]Graduate School of Informatics, Kyoto University, Kyoto, Japan

shi@sap.ist.i.kyoto-u.ac.jp

Abstract

Self-supervised learning (SSL)-based pretrained models have significantly improved automatic speech recognition (ASR) performance. As the feature extraction (FE) module has also been well-trained with a large amount of training data, freezing the FE during finetuning for downstream ASR tasks is common. When there is a severe mismatch between the simulated noisy data for pretraining and real noisy data, however, finetuning the FE with the real noisy data should be done without losing the effective information of the pretrained FE. In this paper, we propose a dual-path adaptation of the FE to address this problem. It combines the frozen pretrained FE path and the finetuned-adapted FE path with convolutional fusion layers. Moreover, adapters are inserted into the Transformer encoder. The experimental results using the CHiME–4 dataset show that using adapters for the FE or the Transformer encoder is effective, but achieving synergy of these two is challenging. Finetuning of the FE combined with adapters in the encoder realizes effective model adaptation. Moreover, the proposed method utilizes the complementarity between the pretrained and the finetuned FE paths, achieving significant improvements even with noise-robust WavLM models.

Index Terms: Automatic speech recognition, noise robustness, self-supervised learning, adapter

1. Introduction

Self-supervised learning (SSL) [1, 2, 3, 4, 5] is a machine learning approach in which a model discerns patterns from unlabeled data by autonomously creating supervisory signals or labels. Compared to the conventional supervised learning that relies on human-annotated data for training [6], SSL extracts features and encoded representations with massive unlabeled data for subsequent tasks [7, 8]. SSL greatly improves the performance of automatic speech recognition (ASR) [4, 9, 10, 11]. Typical SSL models for ASR [4, 9, 12] contain a feature extraction (FE) module and a Transformer encoder [6]. The unsupervised FE module is trained well and universally with a large amount of data. Thus, it is a common practice to freeze the parameters of the FE [13, 14] and finetune the following Transformer layers only.

Although this finetuing method benefits many speech-oriented tasks [7, 8, 15, 16], two mismatches emerge in the FE when targeting noise-robust ASR. The primary mismatch arises as the pretraining data of the FE is primarily based on clean and simulated noisy speech. In contrast, the main application needs to tackle real noisy speech [4, 9]. The second mismatch is caused by the divergent data distribution between clean and noisy speech [9]. While some pretraining datasets incorporate noisy speech, noise in unsupervised training may lead to erroneous cluster assignments during the quantization process [9, 12]. Finetuning the FE parameters can mitigate the mismatch for noise-robust ASR, but the pretrained information of the FE will be diminished.

Furthermore, SSL-based models typically consist of a huge number of parameters [4, 9, 10, 11]. Consequently, an efficient method to adapt the model to new scenarios is critical. Inserting adapters [17, 18, 19] within the model presents a simple but effective method. Inserting an adapter in the encoder layer or after the layer can achieve good effects at the encoding level [17]. In this process, the parameters of the Transformer encoder are frozen, and only the adapters are finetuned [17, 20]. Addressing the mismatch mentioned above at the feature level is also necessary for noise-robust ASR, in addition to encoder-level adaptation. However, there are limited studies on FE adaptation for SSL-based pre-trained models.

In this paper, we first investigate the adaptation of the FE module based on finetuning and adapters. Based on the observation, we propose a dual-path adaptation of FE for improving the noise-robust ASR. It consiss of a frozen pretrained FE path and a finetuned-adapted FE path. The frozen pretrained FE path keeps the information learned from massive pretraining data, while the finetuned FE path deals with real noise. These two paths are fused with convolutional layers in a masking way similar to speech enhancement [21, 22, 23, 24]. The fusion of the two paths aims to utilize the complementarity between them. The proposed method has a synergy with adapters in the encoder.

The rest of this paper is organized as follows. Section 2 describes related works. Section 3 describes our proposed method. Section 4 gives experimental settings and results. Section 5 gives the conclusion and future work.

2. Preliminaries

2.1. HuBERT and WavLM

HuBERT [9] is a self-supervised model designed for ASR. Its training can be divided into two stages: pretraining and finetuning. During the pretraining, some segments of the features extracted by the FE module are masked. Then, the Transformer is trained to infer the quantized code of the masked (and unmasked) segments using the masked segments. WavLM [12] extends HuBERT by using simulated noise speech in the pretraining process to learn noise robustness capabilities. Cross-entropy loss is defined during the pretraining stage. Once pretrained, the model can be finetuned on a labelled ASR dataset with the connectionist temporal classification (CTC) loss [25]. The FE module is shown in Fig. 1–(a). It contains several 1-D convolutional layers, which extract the feature embedding x' from the time-domain waveform x.

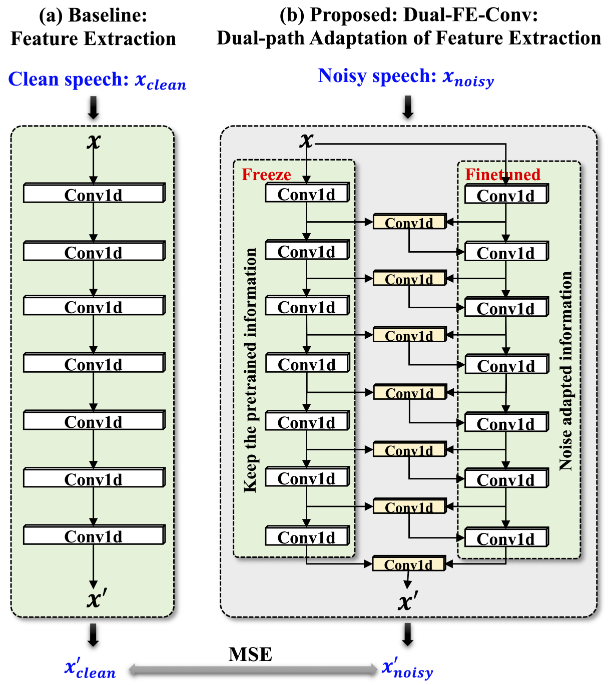
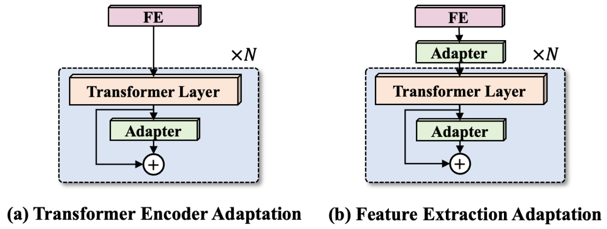

(a) Baseline:
Feature Extraction
(b) Proposed: Dual-FE-Conv:
Dual-path Adaptation of Feature Extraction
Clean speech: x_{clean}
Noisy speech: x_{noisy}
x
Conv1d
x'
Freeze
Finetuned
Keep the pretrained information
Noise adapted information
x'_{clean}
MSE
x'_{noisy}

Figure 1: *Neural network structure of (a) baseline feature extraction module; (b) proposed dual-path adaptation of feature extraction module (Dual-FE-Conv).*



(a) Transformer Encoder Adaptation (b) Feature Extraction Adaptation

Figure 2: *Flowchart of (a) adapter-based adaptation of the Transformer encoder; (b) adapter-based adaptation of both the FE module and the Transformer encoder.*

2.2. Adapter

An adapter was proposed [18] to efficiently adapt a large pretrained model in natural language processing tasks. The adapter structure depends on task requirements and the model architecture [26]. A simple but highly effective adapter contains two dense layers of a bottleneck shape called LoRA [17]. In this work, the adapter's input is the output of the Transformer layer in the pretrained model. The adaptation process is depicted as follows:

$$e' = e + \text{adapter}(e) \tag{1}$$

Here, e represents the original encoder layer output, and e' represents the adapted feature. As shown in Fig. 2–(a), an adapter is inserted after each Transformer layer. As shown in Fig. 2–(b), it can also be inserted after the FE module.

3. Dual-path Adaptation of Feature Extraction

A pretrained FE module has already learned good feature representations. Therefore, the FE module is often frozen during finetuning to maintain the information learned from massive data. However, a mismatch can exist between the simulated speech for pretraining and the real noisy speech for evaluation.

In this paper, we propose a dual-path adaptation of the FE module for noise-robust ASR, which is depicted in Fig. 1–(b). The proposed FE module contains two paths: the pretrained FE path keeps the information learned from the massive pretraining data; the adapted FE path is finetuned with the target noisy data, which is more suitable for noisy ASR but may lose the information learned in the pretraining. These two paths can be combined by simply adding:

$$x_{fused} = x_{frozen} + x_{finetuned} \tag{2}$$

where x_{frozen}, $x_{finetuned}$, and x_{fused} denote the features derived from the frozen FE module, those from the finetuned FE module, and the fused features, respectively. This adding fusion method is denoted as **Dual-FE-Add**. We also propose to use additional 1-D convolutional layers to fuse information from the two paths layer by layer:

$$x_{fused} = Conv_{1d}(Concat(x_{frozen}, x_{finetuned})) \tag{3}$$

$Conv_{1d}$ denotes the convolutional 1-D layer, and $Concat$ denotes the concatenation. In this paper, the kernel size of the $Conv_{1d}$ layers is 1, and $1 \times 1 - conv$ block is also known as pointwise convolution. Thus, the $Conv_{1d}$ layer, which is similar to the masking way in the speech enhancement [27], fuses effective information from the dual-path features. This convolutional fusion method is denoted as **Dual-FE-Conv**.

We also introduce pretraining for dual-path FE. Clean speech is input to the frozen FE module to obtain the target x'_{clean}. Then, it is compared against the adapted noisy feature x'_{noisy} derived from the proposed method to calculate the mean squared error (MSE) loss as shown in Fig. 1:

$$\mathcal{L} = ||x'_{clean} - x'_{noisy}||^2 \tag{4}$$

Furthermore, another adapter is incorporated into the Transformer encoder. It is added after each Transformer encoder layer, shown in Fig. 2–(a). As the number of the adapter parameters is much smaller than that of the Transformer encoder, only finetuning the adapter can efficiently adapt the model to different noise scenarios.

4. Experimental Evaluations

4.1. Dataset

The experiments utilized the CHiME-4 dataset [28]. It includes four different noise conditions: bus (BUS), cafe (CAF), pedestrian area (PED), and street junction (STR). The audio in the dataset was digitized at a sampling rate of 16 kHz. All simulated and real noisy data from channels 1 to 6 were utilized during the model training phase. The Channel 5 noisy data from the development and evaluation sets were used for testing.

4.2. Experimental settings for FE module adaptation

For all SSL pretrained models, the FE module contained the same parameters of 7 Conv_1d layers. Except for the input of

Table 1: *Evaluation with **HuBERT** finetuned with **LibriSpeech–960**: "FE" represents the feature extraction module; "Enc" representes the Transformer encoder; "FT" means finetuning all parameters; "Ada" means the use of adapters.*

	Exp.	FE		Enc	Real Development Sets					Real Evaluation Sets				
		FT	**Ada.**	**Ada.**	**BUS**	**STR**	**PED**	**CAF**	**AVE**	**BUS**	**STR**	**PED**	**CAF**	**AVE**
1	Clean Trained HuBERT				25.1	23.9	15.5	21.9	21.6	42.5	25.3	28.7	33.2	32.4
2				✓	18.4	17.6	10.6	15.4	15.5	30.4	17.7	19.9	23.6	22.9
3			✓		21.1	18.5	11.8	16.8	17.1	35.3	19.9	21.5	25.5	25.5
4			✓	✓	19.2	18.0	10.2	15.1	15.6	31.4	18.5	19.4	23.3	23.2
5		✓			24.4	23.5	14.7	21.3	21.0	42.2	24.9	29.1	33.9	32.5
6		✓		✓	14.7	11.5	8.5	10.8	**11.4**	22.5	11.6	13.8	15.8	**15.9**
7	Dual-FE-Add	✓		✓	12.3	9.1	6.8	8.5	**9.2**$\star$	19.4	9.3	11.1	13.0	**13.2**$\star$
8	Dual-FE-Conv	✓			14.9	11.8	8.7	11.0	11.6	23.1	11.0	13.7	17.0	16.2
9	Dual-FE-Conv	✓		✓	11.9	8.1	6.6	8.3	**8.7**$\star$	17.7	8.6	11.0	12.4	**12.5**$\star$

($\star$: p-value $<$ 0.01 against Exp.–6)

the first Conv_1d layer, the number of the input and output channels was 512. Moreover, the kernel size and stride were (10, 5), (3, 2), (3, 2), (3, 2), (3, 2), (2, 2), (2, 2), respectively. The frozen and finetuned FE modules adopted this setting. The number of the input and output channels for the fusion Conv_1d layers were 1024 and 512, respectively. Their kernel size and stride were all 1. During training, SpecAug [29] was applied to the input features for the adapted FE only. We introduced pretraining for the dual-path FE based on the MSE loss in Eqn. (4). The training data was taken from the CHiME–4 dataset. The training epoch was 2.

4.3. Experimental settings for ASR back-ends

We used various ASR back-ends to evaluate the effectiveness of the proposed method. HuBERT models were employed by following the same configuration as fairseq toolkit[1].

- **HuBERT–extraLarge trained with clean speech**: We used the HuBERT model trained with Librispeech-960 as the baseline[2] in order to make a mismatched scenario between training and testing (Exp.–1). It contained 48 Transformer layers. In each Transformer layer, the embedding dimension was 1280, the inner FFN dimension was 5120, the number of attention heads was 16, and the projection dimension was 1024.
- **HuBERT–Large trained with noisy speech**: We evaluated with the ASR back-end finetuned with noisy data. We finetuned the pretrained checkpoint of HuBERT Large[3] with Librispeech (960 hours) [30] and the MUSAN noise dataset [31] (Exp.–10). The noisy speech was made with randomly selected signal-to-noise ratios (SNRs) within the range of 0 to 20 dB. The FE is based on HuBERT Large. It contained 24 Transformer layers with 1024 embedding dimensions and 4096 inner FFN dimensions, and the number of attention heads was 16.
- **WavLM–Large trained with noisy speech**: We also explore the performance of the proposed method with a noise-robust FE. We finetuned the pretrained checkpoint of WavLM Large[4] with Librispeech (960 hours) [30] and the MUSAN noise dataset [31] (Exp.–17). It contained 24 Transformer encoder layers, 1024-dimensional hidden states, and 12 attention heads. Furthermore, it adopted the gated relative position bias in the self-attention.

The input and output dimenssons for adapters were 1280, and the dimension of the middle bottleneck layer was 16. It should be emphasized that inserting adapters in the Transformer encoder layer does not require parameter pretraining.

4.4. Comparison of adapter-based adaptation

Table 1 (upper rows) shows the performance of different ASR systems for the development and evaluation sets, respectively. By comparing Exp.–1 and Exp.–2, adding an adapter into the Transformer layer significantly improved the performance of ASR. This shows that adapter-based encoder-level adaptation is very effective. By comparing Exp.–1 and Exp.–3, adding an adapter in the FE module also significantly improved the performance of ASR. However, Exp.–4 shows that inserting an adapter into both the FE module and Transformer encoder does not improve ASR performance from Exp.–2. The result suggests that combining these two-module adaptations with the adapters presents a challenge. The Transformer encoder adaptation more readily influences the overall performance of the model.

Then, we tried to finetune only the FE module instead of using adapters. According to the results of Exp.–5, the performance is not improved in the development sets and degraded in the evaluation sets. This result shows that finetuning the FE module does not achieve effective noise reduction or adaptation. On the other hand, as shown in Exp.–6, combining the Transformer adapter with FE finetuning significantly improved the performance. Compared with Exp.–2, which only inserts an adapter to the encoder, it showed 29% and 31% relative improvements in the development and evaluation sets, respectively. It also significantly outperforms Exp.–5. The result shows that FE finetuning is effective only when combined with the encoder adaptation, which addresses the mismatch.

The similar trend is observed with the noise speech-trained ASR systems, which were shown in Table 2 and 3.

[1] https://github.com/facebookresearch/fairseq/blob/main/examples/hubert/config/finetune/base_10h.yaml

[2] https://dl.fbaipublicfiles.com/hubert/hubert_xtralarge_ll60k_finetune_ls960.pt

[3] https://huggingface.co/facebook/hubert-large-ll60k

[4] https://huggingface.co/microsoft/wavlm-large/blob/main/pytorch_model.bin

Table 2: *Evaluation with **HuBERT** finetuned with **LibriSpeech–960** and **MUSAN noises**.*

Exp.		**FE**		**Enc**	**Real Development Sets**					**Real Evaluation Sets**				
		FT	**Ada.**	**Ada.**	**BUS**	**STR**	**PED**	**CAF**	**AVE**	**BUS**	**STR**	**PED**	**CAF**	**AVE**
10	Noise Trained HuBERT				19.8	16.5	13.8	14.9	16.3	26.3	17.2	17.8	20.1	20.3
11				✓	15.1	12.8	9.5	11.5	12.3	21.8	13.4	15.4	16.0	16.6
12		✓			16.9	13.8	11.6	13.3	13.9	22.8	13.8	16.1	16.9	17.4
13		✓		✓	14.0	11.8	9.4	11.2	**11.6**	20.0	11.8	14.3	14.9	**15.3**
14	Dual-FE-Add	✓		✓	12.4	9.6	8.1	9.4	**9.9**$\star$	18.4	10.2	13.6	14.3	**14.1**$\star$
15	Dual-FE-Conv	✓			15.9	13.6	11.2	12.8	13.4	21.5	13.5	15.7	16.9	16.9
16	Dual-FE-Conv	✓		✓	11.9	9.8	8.9	8.2	**9.7**$\star$	17.8	10.1	13.7	13.0	**13.7**$\star$

($\star$: p-value $<$ 0.01 against Exp.–13)

Table 3: *Evaluation with **WavLM** finetuned with **LibriSpeech–960** and **MUSAN noises**.*

Exp.		**FE**		**Enc**	**Real Development Sets**					**Real Evaluation Sets**				
		FT	**Ada.**	**Ada.**	**BUS**	**STR**	**PED**	**CAF**	**AVE**	**BUS**	**STR**	**PED**	**CAF**	**AVE**
17	Noise Trained WavLM				11.0	9.8	7.9	9.1	9.5	14.1	9.0	10.1	10.8	11.0
18				✓	9.5	7.8	6.9	7.9	8.1	12.2	8.1	9.5	9.4	9.8
19		✓			10.9	9.0	7.7	9.0	9.1	13.2	8.9	9.7	10.6	10.6
20		✓		✓	9.1	7.9	6.9	8.0	**8.0**	11.6	8.1	8.8	9.4	**9.5**
21	Dual-FE-Add	✓		✓	8.2	7.0	5.9	6.2	**6.8**$\star$	10.5	6.8	7.9	8.3	**8.4**$\star$
22	Dual-FE-Conv	✓			11.2	9.9	8.1	9.2	9.6	13.1	9.0	9.9	10.8	10.7
23	Dual-FE-Conv	✓		✓	8.7	7.3	6.5	7.2	**7.4**$\diamond$	11.7	7.2	8.6	8.6	**9.0**$\diamond$

($\star$: p-value $<$ 0.01 against Exp.–20; $\diamond$: p-value $<$ 0.05 against Exp.–20)

4.5. Effect of dual-path FE for clean speech-trained HuBERT

Table 1 (lower rows) shows results of the proposed dual-path adaptation of the FE for clean speech-trained HuBERT. Simple adding (Exp.–7) provides performance improvement compared with Exp.–6, but a much larger improvement is gained when using the convolutional layers to fuse the two features layer by layer. Compared with Exp.–6, Exp.–9 showed 24% and 21% relative improvements in real data of the development and evaluation sets, respectively. These results confirm information complementarity between the two features. This complementarity can be effectively utilized with more complex networks like $Conv_{1d}$ layers for the clean model. The performance difference between Exp.–9 and Exp.–7 is statistically significant (p-value $<$ 0.05).

The result without adapters in the encoder (Exp.–8) shows an improvement from the same setting (Exp.–5), but it is much degraded from Exp.–9, showing the importance of the adapter.

We also compared the proposed system with directly finetuning the HuBERT–extraLarge with CHiME–4 dataset. The average WER of the real evaluation sets was 13.5, which is worse than the proposed methods (Exp.–9).

4.6. Evaluations with noisy speech-trained HuBERT

Table 2 shows the results with noisy speech-trained HuBERT. The FE module adaptation was also effective in this model. The improvements by Exp.–14 and Exp.–16 from Exp.–13 are significant (p-value $<$ 0.01) for development and evaluation sets. However, the improvement without adapters (from Exp.–12 and Exp.–15) is not so large. The Dual-FE-Conv (Exp.–16) was better than Dual-FE-Add (Exp.–14), but the difference between them is not significant.

4.7. Evaluations with noisy speech-trained WavLM

Table 3 shows the results with WavLM. The proposed method was also effective for this model. The improvement from Exp.–20 to Exp.–23 is statistically significant (p-value $<$ 0.05) for the development and evaluation sets. In this model, however, the performance of Dual-FE-Add (Exp.–21) was better than Dual-FE-Conv (Exp.–23) (p-value $<$ 0.05). The synergy of the proposed dual-path FE adaptation with adapters within the encoder is confirmed, but the complex fusion mechanism is not needed in the noise-robust model.

5. Conclusions

In this paper, we have proposed a dual-path adaptation of the feature extraction (FE) module to address the data mismatch between pretraining and evaluation. The proposed FE module combines the frozen pretrained and finetuned adapted FE paths. The features extracted by these two paths contain information complementarity. Furthermore, the 1-D convolutional layers are adopted to fuse the information between these two paths layer by layer. Moreover, we used adapters to adapt the Transformer encoder. The experimental results using the CHiME–4 dataset show that the combination of finetuning FE with adapters in the encoder provides synergy, and the proposed method significantly improved the ASR performance.

6. Acknowledge

This work was supported by JST Moonshot R&D Grant Number JPMJMS2011.

7. References

[1] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2019.

[2] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. CVPR*, 2020.

[3] J. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent - a new approach to self-supervised learning," in *Proc. NIPS*, 2020, pp. 21 271–21 284.

[4] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," in *Proc. NIPS*, vol. 33, 2020, pp. 12 449–12 460.

[5] S. Dang, T. Matsumoto, Y. Takeuchi, H. Kudo, T. Tsuboi, Y. Tanaka, and M. Katsuno, "Using self-learning representations for objective assessment of patient voice in dysphonia," in *Proc. APSIPA ASC*. IEEE, 2022, pp. 359–363.

[6] L. Dong, S. Xu, and B. Xu, "Speech-Transformer: A No-Recurrence Sequence-to-Sequence Model for Speech Recognition," in *Proc. ICASSP*, 2018, pp. 5884–5888.

[7] S. w. Yang, P.-H. Chi, Y.-S. Chuang, C.-I. J. Lai, K. Lakhotia, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin, T.-H. Huang, W.-C. Tseng, K. t. Lee, D.-R. Liu, Z. Huang, S. Dong, S.-W. Li, S. Watanabe, A. Mohamed, and H. y. Lee, "SUPERB: Speech processing Universal PERformance Benchmark," *arXiv preprint arXiv:2105.01051*, 2021.

[8] A. Mohamed, H.-y. Lee, L. Borgholt, J. D. Havtorn, J. Edin, C. Igel, K. Kirchhoff, S.-W. Li, K. Livescu, L. Maaløe, T. N. Sainath, and S. Watanabe, "Self-supervised speech representation learning: A review," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1179–1210, 2022.

[9] W. Hsu, B. Bolte, Y. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units," *IEEE/ACM TASLP*, vol. 29, pp. 3451–3460, 2021.

[10] A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli, "data2vec: A General Framework for Self-supervised Learning in Speech, Vision and Language," in *Proc. ICML*, 2022, pp. 1298–1312.

[11] 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. ICASSP*, 2020, pp. 6419–6423.

[12] S. Chen, Z. Wang, C.and Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, J. Wu, M. Zeng, X. Yu, and F. Wei, "WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.

[13] B. Thomas, S. Kessler, and S. Karout, "Efficient Adapter Transfer of Self-Supervised Speech Models for Automatic Speech Recognition," in *Proc. ICASSP*, 2022, pp. 7102–7106.

[14] S. Kessler, B. Thomas, and S. Karout, "An Adapter Based Pre-Training for Efficient and Scalable Self-Supervised Speech Representation Learning," in *Proc. ICASSP*, 2022, pp. 3179–3183.

[15] Y. Gao, L. Wang, J. Liu, J. Dang, and S. Okada, "Adversarial domain generalized transformer for cross-corpus speech emotion recognition," *IEEE Transactions on Affective Computing*, 2023.

[16] H. Shi, M. Mimura, and T. Kawahara, "Waveform-domain speech enhancement using spectrogram encoding for robust speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pp. 1–12, 2024.

[17] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-Rank Adaptation of Large Language Models," *arXiv preprint arXiv:2106.09685*, 2021.

[18] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," *arXiv preprint arXiv:1705.08045*, 2017.

[19] Y. Gao, H. Shi, C. Chu, and T. Kawahara, "Enhancing two-stage finetuning for speech emotion recognition using adapters," in *Proc. ICASSP*. IEEE, 2024, pp. 11 316–11 320.

[20] H. Shi and T. Kawahara, "Exploration of adapter for noise robust automatic speech recognition," *arXiv preprint arXiv:2402.18275*, 2024.

[21] Y. Wang and D. Wang, "A structure-preserving training target for supervised speech separation," in *Proc. ICASSP*, 2014, pp. 6107–6111.

[22] Y. Wang, A. Narayanan, and D. Wang, "On training targets for supervised speech separation," *IEEE/ACM TASLP*, vol. 22, no. 12, pp. 1849–1858, 2014.

[23] S. Dang, T. Matsumoto, Y. Takeuchi, and H. Kudo, "A separation priority pipeline for single-channel speech separation in noisy environments," in *Proc. ICASSP*. IEEE, 2024, pp. 12 511–12 515.

[24] H. Shi, M. Mimura, L. Wang, J. Dang, and T. Kawahara, "Time-domain speech enhancement assisted by multi-resolution frequency encoder and decoder," in *Proc. ICASSP*, 2023, pp. 1–5.

[25] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in *Proc. ICML*, 2006, p. 369–376.

[26] Y. Qian, X. Gong, and H. Huang, "Layer-Wise Fast Adaptation for End-to-End Multi-Accent Speech Recognition," *IEEE/ACM TASLP*, vol. 30, pp. 2842–2853, 2022.

[27] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM TASLP*, vol. 27, no. 8, pp. 1256–1266, 2019.

[28] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. ASRU*, 2015, pp. 504–511.

[29] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," in *Proc. Interspeech*, 2019, pp. 2613–2617.

[30] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.

[31] D. Snyder, G. Chen, and D. Povey, "MUSAN: A Music, Speech, and Noise Corpus," *arXiv preprint arXiv:1510.08484*, 2015.