Dialect Identification Using Resource-Efficient Fine-Tuning Approaches

Zirui Lin*, Haris Gulzar†, Monnika Roslianna Busto†, Akiko Masaki†, Takeharu Eda† and Kazuhiro Nakadai*
* Dept. of Systems and Control Engineering, School of Engineering, Institute of Science Tokyo, Tokyo, Japan
Email: {linzirui, nakadai}@ra.sc.e.titech.ac.jp
† NTT Software Innovation Center, Tokyo, Japan
Email: {haris.gulzar, monikka.busto, takeharu.eda, akiko.masaki}@ntt.com

***Abstract*—Dialect Identification (DI) is a task to recognize different dialects within the same language from a speech signal. DI can help to improve the downstream speech related tasks even when speakers have a strong dialect. However, fine-tuning a speech model for tasks like DI is expensive in terms of computation cost and memory requirement. Recent studies have explored fine-tuning pre-trained speech models for tasks like DI using Parameter-Efficient Fine-Tuning (PEFT) methods, which offer parameter efficiency but limited improvement in memory efficiency and training speed. To address these challenges, we explore Memory-Efficient Fine-Tuning (MEFT) methods, originally proposed for language processing, and apply them to the general-purpose pre-trained speech model. We then comprehensively analyze the GPU memory usage and fine-tuning speed based on various MEFT methods. As a case study, we fine-tune the Whisper model to identify six Mandarin subdialects from the KeSpeech dataset, reducing GPU memory usage by up to 73.25% and accelerating training speed by a factor of 2.1, while maintaining accuracy comparable to vanilla fine-tuning and PEFT methods.**

I. Introduction

Dialect identification (DI) [1], [2] amounts to identifying dialects belonging to the same language branch, which is a specific case of language identification [2], [3] (LID) task. However, DI is more challenging than LID because dialects share similar acoustic and linguistic characteristics compared to different languages [4], [5]. Dialect identification can help improve speech recognition systems and is also expected to enhance human-computer interaction applications and secure remote access communications [6]. Moreover, DI can help drive novel e-health and telemedicine services, which can be especially valuable for elderly and homebound individuals. [4]

Fine-tuning a pre-trained speech model for downstream tasks has been proved to achieve outstanding performance on various tasks as it can leverage the common speech features learned from large corpus during the pre-training [7]–[11]. However, vanilla fine-tuning, which updates patameters of the whole model, demands substantial computational power and memory capacity, making it a costly endeavor, especially in the case that the space complexity of transformer [12] is $O(n^2)$ while n is the sequence length [13], and the sequence lengths of speech models are often longer than that of language models. To alleviate these resource requirements, recent studies [14], [15] have adopted Parameter-Efficient Fine-Tuning (PEFT) methods [16], which update only a small subset of model parameters. PEFT drastically reduces compute and storage overhead while achieving performance on par with full fine-tuning and mitigating catastrophic forgetting [17].

Although Parameter-Efficient Fine-Tuning (PEFT) methods significantly reduce the number of parameters to be updated, their gains in memory efficiency are modest, typically ranging from 10% to 30%. In other words, PEFT methods provide limited advantages for training on low-resource GPUs, as they still require backpropagation through the entire backbone model to compute gradients for parameter updates [18]. This necessitates storing intermediate activations from the backbone model's layers in GPU memory, leading to substantial memory consumption. Furthermore, our observations suggest that PEFT methods have a limited impact on accelerating the training, with improvements of up to about 10% in our evaluations, which offers only marginal benefits for low-resource GPUs.

To better adapt fine-tuning for DI tasks on low-resource GPUs, we introduce Memory-Efficient Fine-Tuning (MEFT) methods [18]–[20] by using DI as an example. MEFT methods have achieved success in Natural Language Processing (NLP) tasks [18]–[20], but its effectiveness on speech related tasks remains to be explored. We deploy MEFT methods on the general-purpose speech recognition architecture, Whisper [21] from OpenAI, which has been pre-trained on a large speech corpus, and fine-tune it for DI tasks. MEFT methods construct a lightweight side network alongside the backbone model, taking the intermediate activations of the backbone model as inputs and updating only the side network. This approach avoids backpropagation through the backbone model, reducing GPU memory usage.

In this work, we evaluate our approaches on subdialects of Mandarin Chinese, as the differences between Mandarin subdialects are relatively subtle compared to dialect-level variations and difficult to be effectively captured by the model [22], making their distinction more challenging. Mutual intelligibility is generally high among Mandarin subdialects despite regional lexical differences. This serves as a rigorous test of the capability.

The main contributions of this work are as follows:

1) We extend the application of MEFT methods to speech processing, systematically addressing the challenges posed by long input sequences and the computational overhead of side-layer operations. Our analysis demonstrates the effectiveness of MEFT in mitigating GPU

979-8-3315-7206-8/25/$31.00

memory bottlenecks in fine-tuning speech models.
2) We report substantial performance gains on the KeSpeech dataset [23], specifically targeting six Mandarin sub-dialects, where our methods consistently outperform baseline systems in subdialect identification accuracy.
3) Our proposed approaches achieve comparable accuracy to full fine-tuning (within 0.67%) while reducing GPU memory consumption by up to 3.7× and improving training throughput by up to 2.1×. Our code is available at *https://github.com/linzr25/Whisper-DI-MEFT*.

II. Related Work

A. Parameter-Efficient Fine-Tuning Methods

Several PEFT methods have been proposed to fine-tune large pre-trained models for downstream tasks. In this section, we introduce several widely used PEFT approaches.

Adapters: Adapters are small, trainable modules typically inserted into pre-trained models, with the original model parameters kept frozen. These modules usually follow a bottleneck architecture that includes a layer normalization (LN), two fully connected (FC) layers, an activation function, and a residual connection. We follow the insertion form consistent with the work [24].

BitFit: Bias-Term Fine-Tuning (BitFit) [27] is a PEFT method that updates only the bias terms and the task-specific classification layer of the model, leaving the other pre-trained parameters unchanged.

LoRA: LoRA adds low-rank matrices to the transformer attention layers. LoRA simulates parameter changes via low-rank decomposition, enabling fine-tuning of large models using a small number of parameters.

AdaLoRA: AdaLoRA [26] refers to Adaptive Low-Rank Adaptation, which addresses the limitation of distributing the parameter budget evenly across all weight matrices/layers, as seen in LoRA and other methods. AdaLoRA adaptively allocates the parameter budget to different weight matrices based on their importance score and parameterizes the updates using singular value decomposition.

These PEFT methods either update a small subset of pre-trained parameters or introduce additional parameters into a pre-trained backbone network. Consequently, they still require back-propagation through the backbone model, because the gradient values are needed even for frozen layers to update the parameters of fine-tuned layers, as shown in Fig. 1 (a), leading to high GPU memory consumption.

B. Existing Work in Dialect Identification

Fine-tuning pre-trained speech models has become a widely adopted approach, as it effectively leverages the knowledge learned from large-scale datasets during pre-training, resulting in improved performance on resource-limited dialect datasets.

Shaik et al. [28] combined self-supervised adaptive pre-training with fine-tuning on the pre-trained wav2vec 2.0 [9] model, achieving high performance in both language and dialect identification. Garg et al. [29] fine-tuned the pre-trained Conformer [30] model for classifying African-American and non-African-American English, improving speech recognition performance. Shen et al. [31] fine-tuned self-supervised pre-trained speech models as feature extractors of the dialect identification system.

Several studies have explored the application of PEFT methods for DI tasks. Radhakrishnan et al. [14] employed Adapters to fine-tune the pre-trained Whisper [21] model for Arabic dialect identification, achieving state-of-the-art accuracy on the ADI-17 [32] dataset. Kamiya et al. [15] applied Adapters to the pre-trained wav2vec 2.0 model for DI and Automatic Speech Recognition (ASR) tasks, leveraging DI to enhance Japanese dialect speech recognition.

Despite the success of these studies, they are all based on vanilla fine-tuning or PEFT methods, therefore, suffer from high resource demands.

III. Method

We explore three MEFT methods for resource-efficient fine-tuning of pre-trained models on DI tasks: LST [18], UniPT [19], and SHERL [20].

A. Ladder Side-Tuning

Ladder Side-Tuning (LST) [18], as illustrated in Fig. 1 (b), constructs a lightweight, trainable side network that runs alongside the backbone model. Each layer in the side network receives input from the intermediate activations via shortcut connections, forming a parallel ladder structure. The i^{th} side layer takes the output from the previous side layer, denoted as h^g_{i-1}, and the downsampled intermediate activations, h^f_i, from the i^{th} backbone layer. The activations are combined using a trainable gating mechanism: $\mu_i * h^f_i + (1 - \mu_i) * h^g_{i-1}$, where $\mu_i = \text{sigmoid}\left(\frac{\alpha_i}{T}\right)$, α_i is a learnable scalar initialized to zero, and $T(= 0.1)$ is a constant temperature.

By introducing direct connections between layers in the side network, LST eliminates the need for backpropagation through the backbone model. Although constructing the side network using transformer blocks outperforms in natural language processing (NLP) and vision-and-language (VL) tasks, we choose bottleneck Adapter blocks to construct the side network in our work. This is because the sequence length of speech features is generally longer than that of text tokens [33], and the space complexity of the standard self-attention mechanism is $O(n^2)$ [13]. We opt not to introduce attention mechanisms in the side network of LST to avoid excessive memory requirements for longer sequences.

B. Universal Parallel Tuning

Universal Parallel Tuning (UniPT) [19], illustrated in Fig. 1 (c), aims to address the potential semantic inconsistencies introduced by the static gating mechanism in LST, which combines the outputs of each backbone layer with its corresponding side layer. UniPT leverages the final layer's superior representation and transfer capabilities in the pre-trained network to overcome this limitation, using its layer feature F_N as a guide. In the interaction layer, the feature from i^{th}

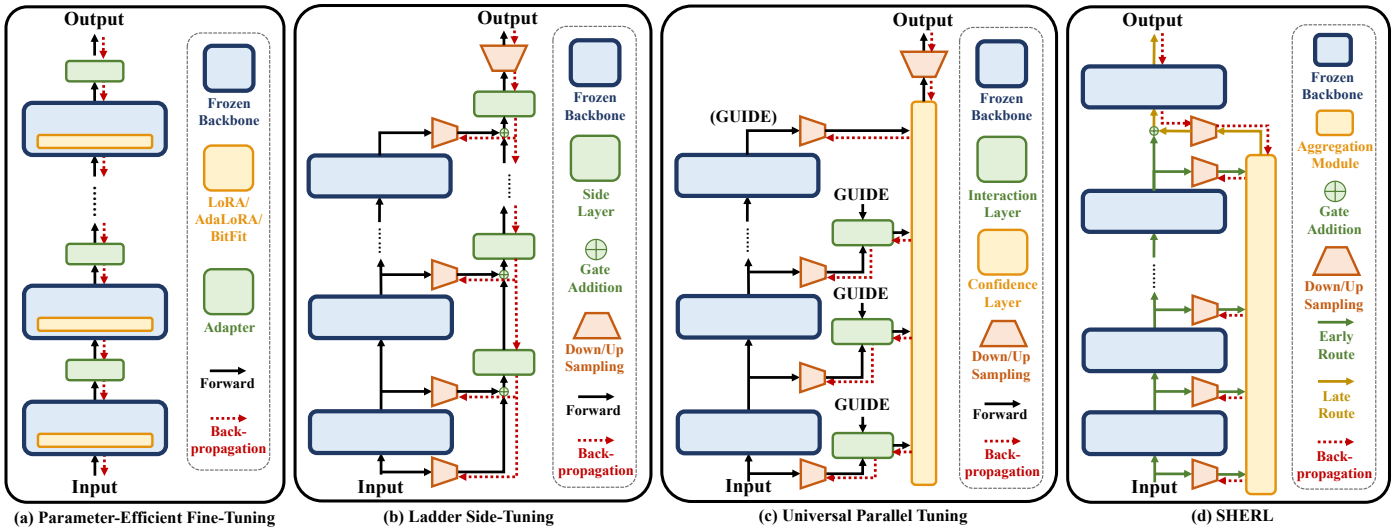

Fig. 1. **(a)** parameter-efficient fine-tuning, including Adapter [24], LoRA [25], AdaLoRA [26] and BitFit [27]; **(b)** Ladder Side-Tuning (LST) [18]; **(c)** Universal Parallel Tuning (UniPT) [19]; **(d)** SHERL [20].

backbone layer F_i serves as the key and value, while the F_N acts as the query. Feature extraction is performed via a cross-attention operation: $F_i' = \left(\|\text{ReLU}(F_N F_i^\top)\|_1 + I\right) F_i$. In the confidence layer, F_N guides the aggregation of features through confidence-based calculations. This interaction ensures semantic coherence between layers.

C. SHERL

SHERL [20], illustrated in Fig. 1 (d), addresses two main issues in MEFT methods: feature redundancy during cross-layer aggregation, which may dilute small but meaningful features, and misalignment between the output projections of the backbone network layers and the original input-output projections. SHERL first computes the cosine similarity between the features of the first $N-2$ layers and sums the values along the embedding dimension to obtain the redundancy rate of each feature. It then uses the output of the $(N-1)^{th}$ layer as guidance, with the feature set of shallow layers as the key and value. Feature extraction is performed through cross-attention, while redundancy is reduced based on the calculated redundancy rate. After the early aggregation, as shown in Fig. 1, SHERL combines the aggregated early features with those from the $(N-1)^{th}$ layer through gate addition, then passes the result through the N^{th} layer to produce the final output.

IV. Experiments

A. Model

Our experiments use the multilingual Whisper-small [21], a transformer-based, general-purpose speech recognition model pre-trained on 680,000 hours of multilingual and multitask supervised data, demonstrating strong speech representation capabilities. The embedding dimension of the model is 768. The input audio is 30 seconds long, converted into a log-Mel spectrogram with a sequence length 1500. If the audio duration is shorter than 30 seconds, it is padded with zeros; if it exceeds 30 seconds, it is truncated to ensure a consistent input length.

For our experiments, we use only the encoder portion of the Whisper model. At the end of the encoder, we append a classification head. The output from the final encoder layer is passed through a projection layer, which maps the feature dimensions to the appropriate space for classification. A pooling operation is then applied to average the features across all time steps of each sample, resulting in a fixed-length pooled representation. Finally, this pooled output is fed into a fully connected classifier to generate the final classification logits. In our configuration, the projection layer has a dimension of 256, and the model has 88 million parameters.

B. Dataset

We evaluate our model on KeSpeech[23], which is a large-scale open-source speech dataset designed to advance sub-dialect identification, featuring 1,542 hours of audio from 27,237 speakers across 34 cities in China. It covers standard Mandarin and 8 subdialects, with rich labeling (transcriptions, speaker identity, subdialect) that enables effective training and evaluation of dialect-related tasks. The inclusion of parallel recordings in both standard Mandarin and regional subdialects supports novel applications like subdialect style conversion, while the extensive speaker diversity and two-phase recordings make it ideal for robust, time-aware dialect modeling.

C. Training Settings

We use the Adam optimizer for all methods in our experiments. We explore learning rates of $\{1 \times 10^{-3}, 5 \times 10^{-4}, 1 \times 10^{-4}\}$ and select the optimal value based on evaluation performance. Any extra parameters introduced into the pre-trained model are randomly initialized. All methods are fine-tuned for 10 epochs in each experimental setup until convergence. Training is performed on a single NVIDIA H100 94GB

TABLE I

COMPARISON OF DIFFERENT METHODS ON SIX MANDARIN SUBDIALECTS FROM THE KESPEECH [23] DATASET. WE REPORT THE OVERALL IDENTIFICATION ACCURACY ON THE TEST SET, GPU MEMORY USAGE, AND TRAINING TIME PER EPOCH IN MINUTES. THE ADAPTER, LORA, AND ADALORA METHODS ARE EVALUATED WITH HIDDEN DIMENSION (DIM), RANK (R), OR INITIAL RANK (INIT_R) OF {64, 128, 256}. THE LST, UNIPT, AND SHERL METHODS ARE EVALUATED WITH REDUCTION FACTORS (RF) OF {2, 4, 8}. THE TRAINING TIME IS AVERAGED OVER ALL TRAINING EPOCHS.

Method	**Trainable Ratio (%)**	**GPU Mem. (GiB) (↓)**	**Time/epoch (min) (↓)**	**Accuracy (%) (↑)**
Kaldi-xvector [23]	——	——	——	56.34
ResNet-34 [23]	——	——	——	61.13
ECAPA-TDNN [23]	——	——	——	60.77
Head Tuning	0.22	7.06	5.68	59.33
Vanilla Fine-Tuning	96.48	76.14	16.58	78.67
Adapter$_{dim=64}$	2.88	64.21	16.21	76.26
Adapter$_{dim=128}$	5.34	65.34	16.33	75.83
Adapter$_{dim=256}$	9.91	67.34	16.50	75.54
LoRA$_{r=64}$	2.82	53.14	15.92	78.06
LoRA$_{r=128}$	5.28	53.76	15.99	78.37
LoRA$_{r=256}$	9.85	54.95	16.09	78.77
AdaLoRA$_{init_r=64}$	2.82	52.87	18.31	76.86
AdaLoRA$_{init_r=128}$	5.29	53.44	18.35	77.71
AdaLoRA$_{init_r=256}$	9.86	56.87	18.49	77.80
BitFit	0.33	47.33	15.05	68.83
LST$_{RF=2}$	7.09	28.37	9.92	77.37
LST$_{RF=4}$	3.80	22.88	9.17	77.18
LST$_{RF=8}$	2.06	20.39	8.58	78.00
UniPT$_{RF=2}$	5.31	43.32	17.40	77.47
UniPT$_{RF=4}$	2.85	36.76	16.25	77.58
UniPT$_{RF=8}$	1.57	34.26	15.51	76.43
SHERL$_{RF=2}$	7.92	45.08	10.78	75.98
SHERL$_{RF=4}$	3.30	31.64	9.03	75.43
SHERL$_{RF=8}$	1.54	21.54	8.02	75.88

GPU with a batch size of 128 and enables mixed precision training [34]. We do not use gradient checkpointing [35] or quantization [36]. No additional regularization techniques are applied.

For Adapter, LoRA, and AdaLoRA, we experiment with hidden dimensions, ranks, or initial ranks from {64, 128, 256}. For LST, UniPT, and SHERL, we test reduction factors (RF) of {2, 4, 8}, which control the embedding dimensions of the downsampling layers, upsampling layers, and the side network. The hidden dimension of the Adapter module in LST is fixed at 256. In vanilla fine-tuning, the two convolutional layers at the bottom of the Whisper model—responsible for processing input representations—and the position embedding layer are frozen. In addition to vanilla fine-tuning, PEFT, and our approaches, we also evaluate head tuning, where only the classification head is updated. For all methods, the classification head is trained during fine-tuning.

D. Experimental Results

Accuracy Performance. The experimental results are shown in Table I. Among PEFT methods, Adapter, LoRA, and AdaLoRA achieve performance comparable to vanilla fine-tuning, significantly outperforming the baseline provided by KeSpeech [23], which trains the entire network. BitFit, however, performed poorly. Our proposed approaches demonstrated similar performance to PEFT methods. Among them, LST with $RF = 8$ achieves an accuracy of 78.00%, outperforming AdaLoRA and Adapter, maintaining accuracy within 0.67% of vanilla fine-tuning and 0.77% of LoRA.

GPU Memory Usage. As shown in Table I, PEFT methods can reduce GPU memory usage compared to vanilla fine-tuning. Among PEFT methods, Adapter reduces memory usage by about 11% to 16%, while LoRA and AdaLoRA reduce memory usage by approximately 28% to 30% compared to vanilla fine-tuning. BitFit achieves the most significant memory reduction at 37.83%, though its performance was suboptimal. MEFT methods demonstrated a further reduction in GPU memory usage compared to PEFT methods. When $RF = 8$, LST, UniPT, and SHERL reduce GPU memory usage by 73.25%, 55.00%, and 71.71%, respectively, compared to vanilla fine-tuning. LST achieves the best performance among MEFT methods, with the lowest memory usage, 2.3× lower than the PEFT method with the lowest memory usage.

Training Speed. The average training time per epoch recorded in Table I shows that PEFT methods offer limited improvements in training speed. In contrast, among MEFT

methods, LST and SHERL significantly enhance training speed. When $RF = 8$, LST and SHERL accelerate training by 1.9× and 2.1×, respectively. Notably, we observe that the convergence speed of MEFT methods is comparable to PEFT methods.

V. Discussion

In our experiments, we found that among three MEFT approaches the training time of UniPT is significantly longer than that of LST and SHERL, with its GPU memory usage notably higher than LST and exceeding SHERL when $RF = \{8, 16\}$. We attribute this to the cross-attention operations introduced at each layer. This operation's time and space complexity is $O(n^2)$, which increases computational complexity and GPU memory usage as the sequence length grows. SHERL introduces this operation only once, while LST, in our setup, does not include attention operations. This reveals that for speech models, it is crucial to minimize the introduction of such operations when fine-tuning with MEFT methods to maintain resource efficiency.

In addition to the MEFT methods implemented in our work, we also tested the memory-efficient zeroth-order optimizer (MeZO) [37], which estimates gradients using only forward passes and parameter perturbations for optimization. However, we found that MeZO failed to converge, possibly due to the higher effective rank of the Hessian matrix of the loss [37] when fine-tuning the pre-trained speech model for the DI task.

Furthermore, for all MEFT methods, besides using the output from the final encoder layer of Whisper model, we experimented with the weighted sum of the outputs from all encoder layers, which performed worse than using only the output from the final encoder layer.

Our future work will focus on further reducing MEFT's resource consumption by integrating advanced memory-saving techniques (e.g., quantization) and conducting ablation studies to identify and minimize the number of layers requiring fine-tuning. We also plan to extend our work to utilize MEFT methods in broader speech-related tasks.

VI. Conclusion

By leveraging a general-purpose pre-trained speech processing model and implementing MEFT approaches, we demonstrated competitive performance on the DI task using the KeSpeech dataset for six Mandarin subdialects. Our proposed approach significantly reduces GPU memory usage by up to 73.25% and accelerates training iteration speed by up to 2.1× compared to the vanilla fine-tuning while maintaining accuracy within 0.67% of vanilla fine-tuning and 0.77% of resource-hungry PEFT approaches like LoRA. These results highlight that MEFT approaches, which have recently been adopted for natural language processing, have the potential to be efficiently used in speech-related tasks.

References

[1] M. Zissman, "Language identification using phoneme recognition and phonotactic language modeling," in *1995 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 5, 1995, 3503–3506 vol.5.

[2] M. A. Zissman, T. P. Gleason, D. M. Rekart, and B. L. Losiewicz, "Automatic dialect identification of extemporaneous conversational, latin american spanish speech," in *1996 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 2, 1996, 777–780 vol. 2.

[3] E. Ambikairajah, H. Li, L. Wang, B. Yin, and V. Sethu, "Language identification: A tutorial," *IEEE Circuits and Systems Magazine*, vol. 11, no. 2, pp. 82–108, 2011.

[4] A. Etman and A. A. L. Beex, "Language and dialect identification: A survey," in *2015 SAI Intelligent Systems Conference (IntelliSys)*, 2015, pp. 220–231. DOI: 10.1109/IntelliSys.2015.7361147.

[5] P. A. Torres-Carrasquillo, T. P. Gleason, and D. A. Reynolds, "Dialect identification using gaussian mixture models," in *The Speaker and Language Recognition Workshop (Odyssey 2004)*, 2004, pp. 297–300.

[6] A. Etman and A. A. L. Beex, "Language and dialect identification: A survey," in *2015 SAI Intelligent Systems Conference (IntelliSys)*, 2015, pp. 220–231.

[7] N. Vaessen and D. A. Van Leeuwen, "Fine-tuning wav2vec2 for speaker recognition," in *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 7967–7971. DOI: 10.1109/ICASSP43922.2022.9746952.

[8] B. Thomas, S. Kessler, and S. Karout, "Efficient adapter transfer of self-supervised speech models for automatic speech recognition," in *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 7102–7106.

[9] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 12 449–12 460.

[10] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, *Unsupervised cross-lingual representation learning for speech recognition*, 2020. arXiv: 2006.13979 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2006.13979.

[11] A. Babu, C. Wang, A. Tjandra, *et al.*, *Xls-r: Self-supervised cross-lingual speech representation learning at scale*, 2021. arXiv: 2111.09296 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2111.09296.

[12] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

[13] M. N. Rabe and C. Staats, *Self-attention does not need $O(n^2)$ memory*, 2022. arXiv: 2112.05682 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2112.05682.

[14] S. Radhakrishnan, C.-H. H. Yang, S. A. Khan, N. A. Kiani, D. Gomez-Cabrero, and J. N. Tegner, "A parameter-efficient learning approach to arabic dialect identification with pre-trained general-purpose speech model," in *Interspeech 2023*, 2023, pp. 1958–1962.

[15] Y. Kamiya, S. Miwa, and A. Kai, "A parameter-efficient multi-step fine-tuning of multilingual and multi-task learning model for japanese dialect speech recognition," in *2024 27th Conference of the Oriental COCOSDA International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (O-COCOSDA)*, 2024, pp. 1–6.

[16] S. Mangrulkar, S. Gugger, L. Debut, Y. Belkada, S. Paul, and B. Bossan, *Peft: State-of-the-art parameter-efficient fine-tuning methods*, https://github.com/huggingface/peft, 2022.

[17] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych, "AdapterFusion: Non-destructive task composition for transfer learning," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021, pp. 487–503.

[18] Y.-L. Sung, J. Cho, and M. Bansal, "Lst: Ladder side-tuning for parameter and memory efficient transfer learning," in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 12 991–13 005.

[19] H. Diao, B. Wan, Y. Zhang, X. Jia, H. Lu, and L. Chen, "Unipt: Universal parallel tuning for transfer learning with efficient parameter and memory," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 28 729–28 740.

[20] H. Diao, B. Wan, X. Jia, *et al.*, "Sherl: Synthesizing high accuracy and efficient memory for resource-limited transfer learning," in *European Conference on Computer Vision*, 2024, pp. 75–95.

[21] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. Mcleavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proceedings of the 40th International Conference on Machine Learning*, vol. 202, 2023, pp. 28 492–28 518.

[22] Q. Li, Q. Mai, M. Wang, and M. Ma, "Chinese dialect speech recognition: A comprehensive survey," *Artificial Intelligence Review*, vol. 57, no. 2, p. 25, 2024.

[23] Z. Tang, D. Wang, Y. Xu, *et al.*, "Kespeech: An open source speech dataset of mandarin and its eight subdialects," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021.

[24] N. Houlsby, A. Giurgiu, S. Jastrzebski, *et al.*, "Parameter-efficient transfer learning for NLP," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97, 2019, pp. 2790–2799.

[25] E. J. Hu, yelong shen, P. Wallis, *et al.*, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022.

[26] Q. Zhang, M. Chen, A. Bukharin, *et al.*, "Adaptive budget allocation for parameter-efficient fine-tuning," in *The Eleventh International Conference on Learning Representations*, 2023.

[27] E. Ben Zaken, Y. Goldberg, and S. Ravfogel, "BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 2022, pp. 1–9.

[28] M. M. Shaik, D. Klakow, and B. M. Abdullah, "Self-supervised adaptive pre-training of multilingual speech models for language and dialect identification," in *2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 11 436–11 440.

[29] S. Garg, Z. Huo, K. C. Sim, *et al.*, "Improving speech recognition for african american english with audio classification," in *2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 12 356–12 360.

[30] A. Gulati, J. Qin, C.-C. Chiu, *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," in *Interspeech 2020*, 2020, pp. 5036–5040.

[31] R. Shen, Y. Li, H. Gu, Y. Wang, J. Huang, and Q. She, "Self-supervised learning representations for dialect identification with sparse transformers," in *Proceedings of the 2024 8th International Conference on Control Engineering and Artificial Intelligence*, Shanghai, China, 2024, pp. 1–6.

[32] S. Shon, A. Ali, Y. Samih, H. Mubarak, and J. Glass, "Adi17: A fine-grained arabic dialect identification dataset," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 8244–8248.

[33] Y. Hono, K. Mitsuda, T. Zhao, K. Mitsui, T. Wakatsuki, and K. Sawada, "Integrating pre-trained speech and language models for end-to-end speech recognition," in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024, pp. 13 289–13 305.

[34] P. Micikevicius, S. Narang, J. Alben, *et al.*, *Mixed precision training*, 2018. arXiv: 1710.03740 [cs.AI].

[35] P. Micikevicius, S. Narang, J. Alben, *et al.*, "Mixed precision training," in *International Conference on Learning Representations*, 2018.

[36] T. Dettmers, M. Lewis, S. Shleifer, and L. Zettlemoyer, "8-bit optimizers via block-wise quantization," in *International Conference on Learning Representations*, 2022.

[37] S. Malladi, T. Gao, E. Nichani, *et al.*, "Fine-tuning language models with just forward passes," in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 53 038–53 075.