

PARAMETER-EFFICIENT TRANSFER LEARNING OF AUDIO SPECTROGRAM TRANSFORMERS

Umberto Cappellazzo[♡]

Daniele Falavigna♦

Alessio Brutti♦

Mirco Ravanelli♦

[♡] University of Trento, Italy ♦ Fondazione Bruno Kessler, Trento ♦ Concordia University, Canada

ABSTRACT

Parameter-efficient transfer learning (PETL) methods have emerged as a solid alternative to the standard full fine-tuning approach. They only train a few extra parameters for each downstream task, without sacrificing performance and dispensing with the issue of storing a copy of the pre-trained model for each task. For audio classification tasks, the Audio Spectrogram Transformer (AST) model shows impressive results. However, surprisingly, how to efficiently adapt it to several downstream tasks has not been tackled before. In this paper, we bridge this gap and present a detailed investigation of common PETL methods for the adaptation of the AST model to audio/speech tasks. Furthermore, we propose a new adapter design that exploits the convolution module of the Conformer model, leading to superior performance over the standard PETL approaches and surpassing or achieving performance parity with full fine-tuning by updating only 0.29% of the parameters. Finally, we provide ablation studies revealing that our proposed adapter: 1) proves to be effective in few-shot efficient transfer learning, 2) attains optimal results regardless of the amount of the allocated parameters, and 3) can be applied to other pre-trained models. Our code is available at https://github.com/umbertocappellazzo/PETL_AST.

Index Terms— Parameter-Efficient Transfer Learning, Audio Spectrogram Transformer, LoRA, Adapters, Depthwise Convolution

1. INTRODUCTION

Leveraging large pre-trained models for downstream tasks has become a cornerstone of several machine learning domains like natural language processing (NLP) and audio/speech processing. The typical paradigm involves adapting the whole model to each downstream task [1, 2] (i.e., full fine-tuning). Despite achieving remarkable results, this approach leads to a specialized model for each task, which is unfeasible when fine-tuning a model on numerous downstream tasks.

To alleviate this issue, the research community is increasingly focusing on parameter-efficient transfer-learning (PETL) methods, whereby only a small amount of extra pa-

rameters is learned for each task while keeping the pre-trained model frozen [3, 4, 5]. In doing so, the risk of catastrophic forgetting the pre-trained model’s knowledge is also highly reduced, a common problem in continual learning scenarios [6, 7]. For example, **prompt-tuning** methods insert trainable continuous vectors in the input or hidden state of the model, known as prompts [8, 9]. Alternatively, low-rank modules called **adapters**, which follow a bottleneck architecture with a very small intermediate dimension, are introduced into each layer. Another popular method, **LoRA** (Low-Rank Adaptation), leverages low-rank matrix decomposition of pre-trained weight matrices [10]. Several variants of LoRA have been recently proposed to enhance the original implementation leading to better performance and stability [11, 12].

Recently, PETL methods have garnered much attention also in the audio and speech fields. For example, [13, 14, 15] provide extensive experiments on the use of PETL approaches and their combination for self-supervised learning speech models. Also for automatic speech recognition adapters have proven to be an effective solution [16, 17]. For audio classification, the Audio Spectrogram Transformer (AST) [18] obtains superb results, standing out as the state-of-the-art model for several downstream tasks. The problem of how to efficiently transfer the knowledge of the AST is of crucial importance, especially given the typical computational and storage constraints of audio devices. *Surprisingly, this topic has received minimal attention* [19]. Therefore, driven by 1) the absence of previous works, 2) the excellent results obtained by PETL methods in different domains for transformer models, and 3) the need to efficiently adapt the AST model to several downstream tasks, we ask the following question:

(Q) *Can we exploit state-of-the-art PETL methods for the efficient fine-tuning of AST to audio/speech downstream tasks?*

We methodically investigate this research question (**Q**), and to do so we provide a framework whereby we can study the performance attained by several PETL methods on five audio/speech benchmarks. Furthermore, from our experiments, we notice that the bottleneck adapter struggles to achieve on-par performance with respect to the full fine-tuning approach

for speech tasks. We conjecture that this is attributable to the overly simplistic design of the bottleneck adapter, where only linear layers are adopted, which hinders a complete learning of the task at hand. As a consequence, we propose a new adapter design that hinges upon the convolution module of the Conformer model. Our proposed *Conformer adapter* highly benefits from the introduction of the depthwise convolution layer, which allows not only to capture local spatial correlations but also trim down the number of parameters, thus bridging the gap with the full fine-tuning method.

We carry out extensive experiments leading to multiple findings: ① among the standard PETL methods, *LoRA* and *Houlsby bottleneck adapter* achieve the best performance overall, with *LoRA* using fewer parameters; ② our proposed **conformer adapter** provides considerable improvements over the bottleneck adapter, surpassing or attaining performance parity with respect to the full fine-tuning approach while using only 0.29/0.59% parameters compared to it for the Pfeiffer/Houlsby configuration; ③ we study the PETL methods under few-shot settings and their scalability with respect to the number of trainable parameters, validating the efficacy of our proposed adapter; ④ we show empirically that the kernel size of the depthwise convolution is a key parameter to attain the best performance; ⑤ we finally show that the conformer adapter can be also harnessed for the efficient fine-tuning of another pre-trained model like Wav2Vec 2.0.

2. METHODOLOGY

2.1. AST Model Recap

The Audio Spectrogram Transformer (AST) is an attention-based model that achieves state-of-the-art results on various audio and speech tasks [18, 20]. The AST model receives as input audio spectrograms that are patchified and then a linear projection is applied to each patch. This results in a sequence of N tokens of size $d = 768$, which we refer to as $\mathbf{X}_{in} \in \mathbb{R}^{N \times d}$. AST comprises 12 attention layers, each of which is composed of two sub-layers: a *multi-head self-attention* (MHSA) sub-layer and a fully-connected feed-forward (FF) sub-layer. The output of each transformer layer, $\mathbf{X}_{out} \in \mathbb{R}^{N \times d}$ (we omit for simplicity the index of the layer), is computed as follows:

$$\mathbf{X}_{out} = \hat{\mathbf{X}} + \text{FF}(\text{LN}(\hat{\mathbf{X}})), \quad \hat{\mathbf{X}} = \mathbf{X}_{in} + \text{MHSA}(\text{LN}(\mathbf{X}_{in})). \quad (1)$$

Both blocks, MHSA and FFN, include residual connections and layer normalizations (LN) [21], with the LN applied within the residual branch (i.e., Pre-LN).

2.2. Overview of Parameter-efficient Transfer Learning Methods

We now introduce the PETL techniques we used in our experiments: LoRA, prompt/prefix-tuning, and adapter-tuning.

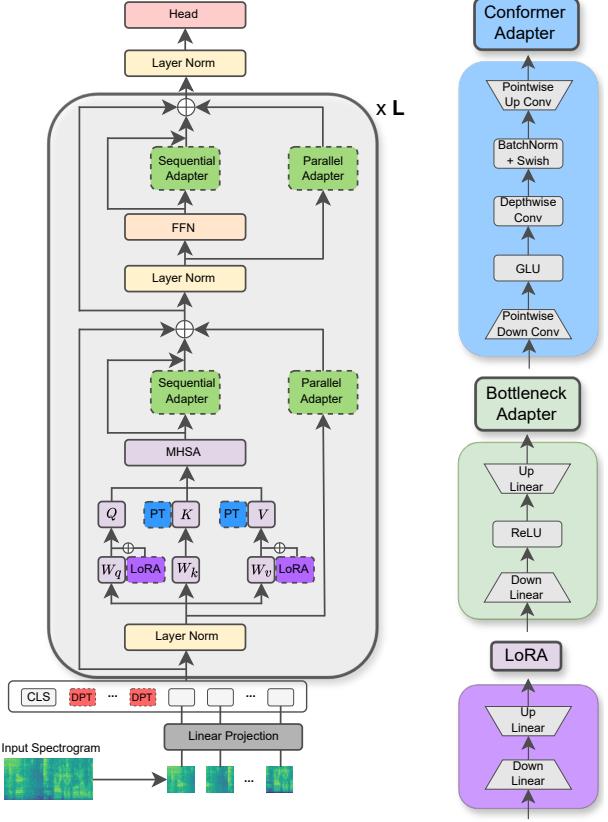


Fig. 1: Left: illustration of the AST model and the integration of PETL methods into it. We use blocks with dashed outlines to characterize the added modules by those methods. **Right:** the inner structure of LoRA, Bottleneck adapter and our proposed Conformer adapter.

LoRA [10]. LoRA introduces trainable low-rank matrices into transformer layers to approximate the weight updates. For a pre-trained weight matrix $\mathbf{W} \in \mathbb{R}^{d \times d_k}$, LoRA represents its update with a low-rank decomposition $\mathbf{W} + \Delta\mathbf{W} = \mathbf{W} + \mathbf{AB}$, where $\mathbf{A} \in \mathbb{R}^{d \times r}$, $\mathbf{B} \in \mathbb{R}^{r \times d}$ are learnable and $r \ll d$. LoRA typically applies this update to the query and value projection matrices, \mathbf{W}_q and \mathbf{W}_v , in the MHSA sub-layer. LoRA computes the query and value matrices like this:

$$\mathbf{Q}/\mathbf{V} = \mathbf{X}_{in} \mathbf{W}_{q/v} + s \cdot \mathbf{X}_{in} \mathbf{A}_{q/v} \mathbf{B}_{q/v}, \quad (2)$$

where s is a tunable scalar hyperparameter.

Prefix-tuning/Prompt-tuning [22, 8]. Prefix-tuning [22] inserts p learnable continuous embeddings of dimension d (i.e., *prompts*) to the keys and values of the MHSA block at every layer. Prompt-tuning [8, 9], instead, prepends the prompts in the input space after the projection layer. Following [9], we consider the “shallow” prompt-tuning version (SPT) where all the prompts are prepended to the first transformer layer, and the “deep” version (DPT) by prepending the prompts uniformly to each transformer layer.

Bottleneck Adapter [23, 24]. Adapters are light subnetworks that are inserted into every transformer layer. To keep the number of parameters limited, adapters exploit a *bottleneck* architecture. The input sequence of hidden dimension d is first down-projected (parametrized by $\mathbf{W}_{down} \in \mathbb{R}^{d \times r}$) into a low-dimensional space with size r (the bottleneck dimension), followed by a non-linear activation function $f(\cdot)$ (e.g., ReLU), and then up-projected back to the original dimension d ($\mathbf{W}_{up} \in \mathbb{R}^{r \times d}$). We refer to this design as **bottleneck** adapter and it is the established choice in the NLP domain [4, 23]. Adapter-tuning is a flexible approach in that we can identify multiple ways in which an adapter can be included in a transformer layer, resulting in different configurations. For example, the adapter can be inserted only after the FF block, (*Pfeiffer* [24]), or after both the MHSA and FF blocks (*Houlsby* [23]). Furthermore, the adapter can be included *sequentially*, either after the FF block [23] (sequential *Pfeiffer*) or after both FF and MHSA blocks [25] (sequential *Houlsby*), or *parallel* to only the FFN block [4, 26], or parallel to both FFN and MHSA blocks [27]. Mathematically, if we consider, as an example, the *Pfeiffer* configuration in which the Bottleneck adapter is placed *sequentially* after the FF block and we let $\mathbf{X}_{FF} = FF(LN(\hat{\mathbf{X}}))$, following the notation in Eq. 1, the output is:

$$\mathbf{X}_{out} = \hat{\mathbf{X}} + \mathbf{X}_{FF} + f(\hat{\mathbf{X}}\mathbf{W}_{down})\mathbf{W}_{up}. \quad (3)$$

Conformer Adapter (Ours). As we will show in Section 3.2, the bottleneck adapter attains competitive results for audio classification tasks, whereas for speech tasks the gap with the full fine-tuning approach is sizeable. We speculate that this happens because the linear design of the bottleneck adapter is not sufficient to disentangle the task at hand. For this reason, we propose to leverage the key block of the Conformer [28], a bleeding-edge model for several speech processing tasks: the *convolution module*. This module highly relies on the depthwise convolution, which is appealing for our PETL setting for two main reasons: 1) it is used for capturing spatial correlations, a crucial aspect for speech downstream tasks, which the bottleneck adapter fails to accomplish, and 2) compared to a standard convolution, it requires fewer parameters, thus making it suitable for parameter-efficient adapters. Therefore, we propose a new adapter that uses the convolution module as the building block and we call it *conformer adapter* (see Fig. 1, right). Specifically, the first pointwise convolution down-projects the input sequence to a dimension of $2r$. Then, the Gated Linear Unit (GLU) halves the hidden dimension to the bottleneck one, r . At this point, the intermediate sequence undergoes the depthwise convolution layer with kernel size equal to k (refer to Section 3.3 for the analysis on this hyper-parameter), as well as the Batch Normalization and Swish activation. Finally, the dimension of the sequence is up-projected to the original d through a pointwise convolution. We show the effectiveness of our conformer adapter in Section 3.

3. EXPERIMENTS

3.1. Implementation Details

Datasets. We evaluate the PETL methods on four audio/speech downstream classification tasks. (1) **Audio classification:** we use the ESC-50 and UrbanSound8K (US8K) datasets. ESC-50 (ESC) [29] consists of 2,000 5-second-long environmental audio recordings of 50 classes. US8K [30] includes 8,732 labeled sound excerpts of urban sounds from 10 classes. (2) **Keyword spotting:** Speech Commands V2 (GSC) [31] has 105,829 1-sec recordings of 35 speech commands. (3) **Intent classification:** Fluent Speech Commands (FSC) [32] includes 30,043 English utterances spanning 31 classes. (4) **Emotion Recognition:** IEMOCAP (IEM) [33] comprises 10,039 utterances from 10 distinct speakers with 4 emotional classes: *neutral, happy, sad, angry*.

Baselines. We include the *full fine-tuning* method (FFT), which finetunes the full pre-trained AST model; and *linear probing*, which only fine-tunes the classification head. We then study various PETL methods: shallow prompt-tuning (SPT), deep prompt-tuning (DPT), prefix-tuning (Pref-T), and BitFit [34], which is a common baseline that fine-tunes only the bias terms of the pre-trained backbone. SPT adds all the 300 prompts to the input of the first transformer layer, whereas DPT adds 25 prompts to each transformer layer. We then include LoRA and bottleneck and conformer (ours) adapters. The dimension of the intermediate space for adapters and LoRA is $r = d/RR$, where $d = 768$ is the hidden dimension of the AST model and RR is the reduction rate. Unless otherwise stated, r is set to 12, 8, and 6 for bottleneck adapter, conformer, and LoRA, respectively. In this way, the resulting number of parameters is roughly the same. For LoRA, following [10], the scaling factor is set to $s = \alpha/RR$, where $\alpha = 16$ leads to the best results (i.e., $s = 8$). We also note that each adapter module is added in parallel to only the MHSA layer (*Pfeiffer*) or both the MHSA and FF layers (*Houlsby*). For this reason, Houlsby adapters require twice as many parameters as *Pfeiffer*. Inserting the adapters sequentially leads to slightly worse results, yet we do not include these results for lack of space. Finally, for the speech tasks we set the kernel size of the depthwise convolution layer to 31, which is the original value proposed in [28], while for audio tasks we found that $k = 8$ gives the best results (we refer the reader to Section 3.3 for a detailed analysis).

Training Details. For all experiments we use the AST model pre-trained on ImageNet-21K and AudioSet provided by the Huggingface Transformers library. The model has around 85.5 million parameters, 12 layers, and the hidden size is 768. For the ablation studies, we also use Wav2Vec 2.0, a well-established pre-trained model for speech tasks [35]. It has around 94M parameters and the same number of layers and hidden size as AST. For all datasets, we use AdamW optimizer with cosine annealing scheduler and weight decay set to 0.1. The initial learning rate is 0.005 for adapters and LoRA,

Table 1: Performance evaluations of the PETL methods on 4 datasets for AST. Best and second-best performances for each dataset are coloured in **Green** and **Red**, respectively.

Method	Par	ESC	US8K	GSC	FSC	Avg
FFT	85M	87.48	84.31	97.31	93.29	90.07
Linear	9/40K	75.85	77.93	41.78	27.52	55.77
BitFit	102K	86.05	82.17	85.51	63.85	79.40
SPT-300	230K	84.30	79.73	75.28	40.85	70.04
DPT-25	230K	86.52	83.67	89.18	68.60	81.99
Pref-T 24	221K	82.93	81.39	83.46	55.75	75.88
LoRA	221K	86.45	83.83	93.61	76.00	84.97
Bottleneck Adapter						
Pfeiffer	249K	88.38	83.44	91.33	73.19	84.09
Houlsby	498K	88.00	82.80	91.75	78.71	85.32
Conformer Adapter						
Pfeiffer	271K	88.30	84.57	96.28	95.48	91.16
Houlsby	542K	85.97	83.59	96.16	96.34	90.51

while for the three prompt-tuning methods is 0.01. Except for US8K that does not provide a validation set by default, for the others we set the hyper-parameters using the validation set. For the ESC and US8K datasets, we run 5-fold and 10-fold cross-validation as suggested in the original papers. Each experiment is carried out using a single A40/V100 GPU. The code and the complete list of hyper-parameters will be released upon acceptance.

3.2. Main Results and Discussion

Table 1 presents the performance comparisons among the various PETL methods. The following observations can be drawn: ① our conformer adapter attains the best performance on average, bringing remarkable improvements over the bottleneck adapter, with the best configurations leading to up to $\{4.9\%, 22.4\%\}$ extra performance improvement on $\{\text{GSC}, \text{FSC}\}$, the two datasets that exhibit the biggest mismatch between the downstream tasks and the data used for pre-training the AST model. Furthermore, our adapter approaches the FFT baseline for GSC, whereas for FSC it is capable of exceeding it by more than 3 points. **Yet, our conformer Pfeiffer/Houlsby adapter only uses 0.29/0.59% parameters compared to the FFT baseline.** ② If we focus on the audio classification tasks, we note good improvements with respect to US8K (it also manages to outstrip FFT by 0.26 points), while for ESC-50 our adapter performs on par with the bottleneck adapter. We point out that the bottleneck adapter outperforms the FFT baseline and it can be already considered a strong approach, so using a more complex design like ours does not improve the performance accuracy. ③ For the bottleneck and conformer adapters, the Houlsby configuration leads to better results for speech classification

Table 2: Few-shot analysis for the ESC-50 and GSC datasets.

Method	ESC				GSC			
	Examples per class				1	2	4	8
DPT-25	32.7	44.3	57.0	71.9	9.4	18.7	43.1	57.1
LoRA	31.8	42.2	58.8	70.7	6.8	15.2	41.8	59.8
Bottleneck	33.0	45.5	60.2	72.8	7.2	16.0	47.9	66.6
Conformer	30.7	41.0	56.2	71.1	5.9	15.5	58.7	77.5

tasks, where having more parameters is beneficial (Houlsby configuration uses twice as many parameters as Pfeiffer), while for audio tasks Pfeiffer achieves better performance accuracy. ④ Among the other PETL methods, we point out that LoRA achieves good results on average, beating the bottleneck Pfeiffer adapter on 3 out of 4 benchmarks.

3.3. Ablation Studies

In this section, we study the efficacy of our proposed adapter under different settings such as few-shot learning and different pre-trained models (e.g., Wav2Vec 2.0). For the bottleneck/conformer adapters, we use the Pfeiffer configuration.

Few-shot Analysis. We evaluate our proposed adapters for few-shot parameter-efficient transfer learning. This scenario is challenging because, in addition to the constraint on the number of trainable parameters, only a few samples are labeled per class. We report the accuracy results for ESC (4 samples) and GSC (32 samples) in Table 2. We see that, whereas for ESC the bottleneck adapter attains the best results, for GSC the gap between this and the conformer adapter is more than 10 points. This again confirms that our proposed adapter is the best choice for speech tasks.

Scaling Abilities. We now want to verify whether our proposed adapter also performs better when fewer parameters (e.g., 50K) or more parameters (up to 1M) are allocated. We restrict our analysis to the Pfeiffer configuration and GSC/FSC datasets. In Fig. 2 we observe that the conformer adapter, regardless of the number of parameters, outstrips the other PETL approaches. In turn, LoRA turns out to be the second best method, and it exhibits strong scaling properties when more parameters are used, bringing better results than bottleneck adapter and DPT. We point out that for FSC, the best result obtained with LoRA requires roughly 900K parameters, whereas the conformer adapter only requires around 100K to achieve the same accuracy.

On the Kernel Size of the Conformer Adapter. We study the impact of the kernel size k of the conformer adapter on the ESC and GSC datasets both for the few-shot (4/32 samples) and full (i.e., no few-shot) settings. We let k vary from 1 to 31, and we point out that setting $k = 31$ only adds roughly 2.8K parameters with respect to $k = 1$. In Fig. 3 we

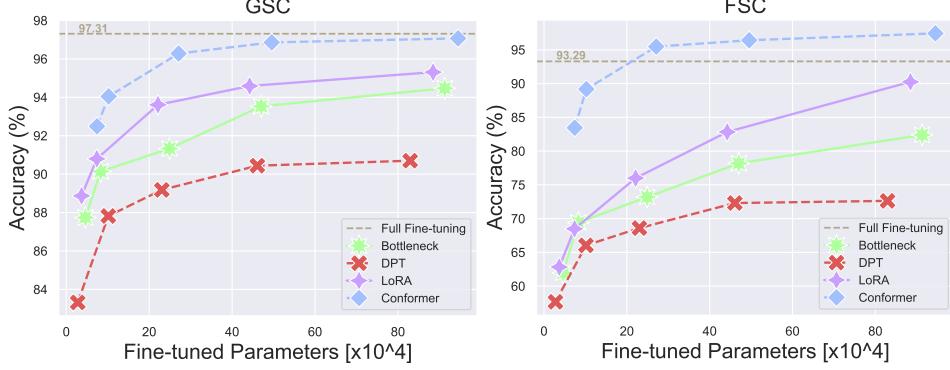


Fig. 2: Scaling trend as more trainable parameters for each PETL method are used for GSC (**Left**) and FSC (**Right**) datasets.

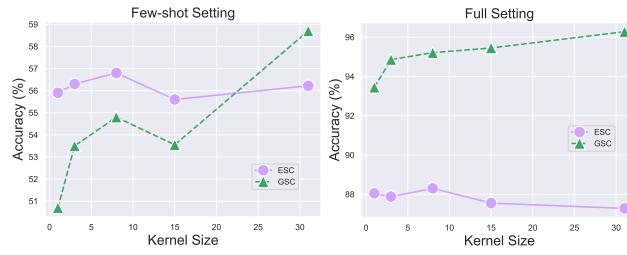


Fig. 3: Impact of the kernel size for the ESC and GSC datasets on the few-shot (**Left**) and full (**Right**) settings.

observe that for ESC, the results are not much influenced by k , and setting $k = 8$ provides the best results. On the contrary, for a more challenging dataset like GSC we see that increasing k leads to better results for both the few-shot and full settings, with the former being more sensitive to it. We speculate that this happens because a larger kernel allows learning more global features, and this is more beneficial when the number of features is small (for the conformer adapter $r = 8$). On the contrary, since ESC is an easier dataset, a smaller kernel is sufficient to achieve optimal results.

Additional Results for Wav2Vec 2.0. Finally, we want to verify whether the proposed conformer adapter can be efficiently harnessed for another pre-trained model. In this direction, we consider Wav2Vec 2.0 [35]. We test the bottleneck and conformer adapters on FSC and GSC, as well as IEMOCAP [33], a benchmark for emotion recognition. For IEMOCAP, we increase the number of parameters to roughly 900K as it is a more challenging dataset. As we can see from Table 3, the performance gap between the adapter approaches and the FFT is large for IEMOCAP. Nonetheless, we can observe that the conformer adapter reaches accuracy results of 55.81, with an improvement of more than 6 points over the bottleneck adapter. For the other two datasets, the conformer adapter turns out again to surpass the bottleneck adapter.

Table 3: Results of bottleneck and conformer adapters for Wav2Vec 2.0 on GSC, FSC and IEMOCAP (IEM) datasets.

Method	Par	GSC	FSC	Par	IEM	Avg
FFT	90M	98.16	99.58	90M	70.05	89.26
Linear	24K	84.93	30.95	4K	36.82	50.90
Bottle	250K	94.96	96.41	895K	48.32	79.90
Conf	272K	95.24	98.33	927K	55.81	83.13

4. CONCLUSION

In this work, we study the problem of parameter-efficient transfer learning for the AST model. To do so, we establish a framework that allows us to examine the performance achieved by the most common PETL methods across five audio/speech benchmarks. We also propose a new adapter module that relies on the conformer convolution module, making effective use of the depthwise convolution. We show that our proposed adapter turns out to be competitive with the full fine-tuning approach and outperforms the established bottleneck adapter, as well as LoRA and prompt-tuning methods. The conformer adapter also provides strong results under few-shot settings, when we vary the number of parameters and if applied to another pre-trained model like Wav2Vec 2.0. Finally, we study the role of kernel size, underscoring its pivotal role in achieving peak performance.

5. REFERENCES

- [1] K. Lv et al., “Full parameter fine-tuning for large language models with limited resources,” *arXiv preprint arXiv:2306.09782*, 2023.
- [2] Y. Wang et al., “A fine-tuned wav2vec 2.0/hubert benchmark for speech emotion recognition, speaker verification and spoken language understanding,” *arXiv preprint arXiv:2111.02735*, 2021.

- [3] L. Xu et al., “Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment,” *arXiv preprint arXiv:2312.12148*, 2023.
- [4] J. He et al., “Towards a unified view of parameter-efficient transfer learning,” in *ICLR*, 2022.
- [5] V. Lalin, V. Deshpande, and A. Rumshisky, “Scaling down to scale up: A guide to parameter-efficient fine-tuning,” *arXiv preprint arXiv:2303.15647*, 2023.
- [6] U. Cappellazzo et al., “Sequence-level knowledge distillation for class-incremental end-to-end spoken language understanding,” *Interspeech*, 2023.
- [7] U. Cappellazzo et al., “Continual contrastive spoken language understanding,” in *ACL*, 2024.
- [8] B. Lester et al., “The power of scale for parameter-efficient prompt tuning,” in *EMNLP*, 2021.
- [9] M. Jia et al., “Visual prompt tuning,” in *ECCV*, 2022.
- [10] E. Hu et al., “Lora: Low-rank adaptation of large language models,” in *ICLR*, 2022.
- [11] Q. Zhang et al., “Adaptive budget allocation for parameter-efficient fine-tuning,” in *ICLR*, 2023.
- [12] S. Liu et al., “Dora: Weight-decomposed low-rank adaptation,” *arXiv preprint arXiv:2402.09353*, 2024.
- [13] T. Lin et al., “Peft for speech: Unveiling optimal placement, merging strategies, and ensemble techniques,” *arXiv preprint arXiv:2401.02122*, 2024.
- [14] Z. Chen et al., “Exploring efficient-tuning methods in self-supervised speech models,” in *SLT*, 2023.
- [15] Y. Li et al., “Evaluating parameter-efficient transfer learning approaches on sure benchmark for speech understanding,” in *ICASSP*, 2023.
- [16] S. Kessler, B. Thomas, and S. Karout, “An adapter based pre-training for efficient and scalable self-supervised speech representation learning,” in *ICASSP*, 2022.
- [17] K. Tomanek et al., “Residual adapters for parameter-efficient ASR adaptation to atypical and accented speech,” in *EMNLP*, 2021.
- [18] Y. Gong, Y. Chung, and J. Glass, “Ast: Audio spectrogram transformer,” in *Interspeech*, 2021.
- [19] N. Selvaraj et al., “Adapter incremental continual learning of efficient audio spectrogram transformers,” *arXiv preprint arXiv:2302.14314*, 2023.
- [20] Y. Gong, C. Lai, Y. Chung, and J. Glass, “Ssast: Self-supervised audio spectrogram transformer,” in *AAAI*, 2022.
- [21] J. Ba et al., “Layer normalization,” *arXiv preprint arXiv:1607.06450*, 2016.
- [22] X. Li and P. Liang, “Prefix-tuning: Optimizing continuous prompts for generation,” *arXiv preprint arXiv:2101.00190*, 2021.
- [23] N. Houlsby et al., “Parameter-efficient transfer learning for nlp,” in *ICML*, 2019.
- [24] J. Pfeiffer et al., “Adapterfusion: Non-destructive task composition for transfer learning,” in *EACL*, 2021.
- [25] R. Mahabadi et al., “Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks,” *arXiv preprint arXiv:2106.04489*, 2021.
- [26] S. Chen et al., “Adaptformer: Adapting vision transformers for scalable visual recognition,” *NeurIPS*, 2022.
- [27] S. Jie and Z. Deng, “Convolutional bypasses are better vision transformer adapters,” *arXiv preprint arXiv:2207.07039*, 2022.
- [28] A. Gulati et al., “Conformer: Convolution-augmented transformer for speech recognition,” *arXiv preprint arXiv:2005.08100*, 2020.
- [29] K. Piczak, “Esc: Dataset for environmental sound classification,” in *ACM Multimedia*, 2015, pp. 1015–1018.
- [30] J. Salamon et al., “A dataset and taxonomy for urban sound research,” in *ACM Multimedia*, 2014.
- [31] P. Warden, “Speech commands: A dataset for limited-vocabulary speech recognition,” *arXiv preprint arXiv:1804.03209*, 2018.
- [32] L. Lugosch, M. Ravanelli, P. Ignoto, V. Tomar, and Y. Bengio, “Speech model pre-training for end-to-end spoken language understanding,” *Interspeech*, 2019.
- [33] C. Busso et al., “Iemocap: Interactive emotional dyadic motion capture database,” *Language resources and evaluation*, vol. 42, pp. 335–359, 2008.
- [34] E. Zaken, S. Ravfogel, and Y. Goldberg, “Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models,” *arXiv preprint arXiv:2106.10199*, 2021.
- [35] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” *NeurIPS*, 2020.