INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

AD-TUNING: An Adaptive CHILD-TUNING Approach to Efficient Hyperparameter Optimization of Child Networks for Speech Processing Tasks in the SUPERB Benchmark

Gaobin Yang[1], *Jun Du*[1,*] , *Mao-kui He*[1], *Shutong Niu*[1], , *Baoxiang Li*[3], *Jiakui Li*[3], *Chin-Hui Lee*[2]

[1]University of Science and Technology of China, Hefei, China
[2]Georgia Institute of Technology, Atlanta, GA, USA
[3]Sensetime Research, Shanghai, China

jundu@ustc.edu.cn

Abstract

In this paper, we propose AD-TUNING, an adaptive CHILD-TUNING approach for hyperparameter tuning of child networks. To address the issue of selecting an optimal hyperparameter set P, which often varies for different tasks in CHILD-TUNING, we first analyze the distribution of parameter importance to ascertain the range of P. Next, we propose a simple yet efficient early-stop algorithm to select the appropriate child network from different sizes for various speech tasks. When evaluated on seven speech processing tasks in the SUPERB benchmark, our proposed framework only requires fine-tuning less than $0.1\%\sim10\%$ of pre-trained model parameters for each task to achieve state-of-the-art results in most of the tasks. For instance, the DER of the speaker diarization task is 9.22% relatively lower than the previously reported best results. Other benchmark results are also very competitive. Our code is available at `https://github.com/liyunlongaaa/AD-TUNING`.

Index Terms: efficient tuning, adaptive early-stop algorithm, self-supervised models, CHILD-TUNING, SUPERB

1. Introduction

Self-supervised learning (SSL) has become increasingly popular in recent years due to its ability to learn representations from large quantities of unlabeled data. This approach has been widely used in computer vision (CV) [1][2][3], natural language processing (NLP) [4][5][6], and speech processing [7][8][9] tasks to enhance the performance of deep learning models. The SSL method involves training a shared representation model on unlabeled data using various pretext tasks, such as predicting the missing portion of an image or the masked token in a sentence. This pre-training stage helps the model learn relevant features that can be fine-tuned for various downstream tasks. Although the pre-trained model can be easily fine-tuned to achieve promising results on many downstream tasks, it also has an obvious drawback: conventional approaches finetune all the parameters of the pre-trained model, which becomes prohibitive as the model size and the number of tasks grow. To address this issue, numerous parameter-efficient fine-tuning methods have been proposed in NLP. Among these methods, adapter and its variants [10][11][12] have achieved great success, by inserting a small module into the pre-trained model and only updating the parameters of the small module while freezing the parameters of the pre-trained model. Consequently, we can utilize a shared pre-trained model and only need to store the adapter parameters fine-tuned for each specific task when migrating between different tasks.

Motivated by NLP, there are also studies to introduce adapters to handle speech tasks. Using adapter on Hubert for speaker recognition tasks has been proposed [13]. Adapters have also been employed for Automatic Speech Recognition (ASR) task in [14][15][16]. More recently, [13][17][18] used adapter to solve a variety of speech tasks, and [17] provided a baseline for various parameter-efficient methods in the SUPERB benchmark. However, such adapter methods have the disadvantages of introducing additional inference time [11] and requiring human consideration of the location of adapter insertion [12][19]. These drawbacks can be overcome if we can fine-tune the pre-trained model itself with parametric efficiency without resorting to these small modules [20][21][22], but this aspect of the work has rarely been studied in the speech community.

In this paper, we mainly explore parameter-efficient fine-tuning of the speech pre-trained model itself for various speech tasks. To our knowledge, we are the first to introduce CHILD-TUNING [21] from NLP, a parameter-efficient fine-tuning method without adding any modules. However, the hyperparameter P that determines the size of the child network is very difficult to optimize in CHILD-TUNING, because often the optimal P varies with the task and even the amount of data. To address the above issue, we first analyze the distribution of parameter importance to ascertain the range of P. Secondly, we propose a simple yet effective early-stop algorithm to adaptively select the appropriate child network from different sizes for various speech tasks. Finally, when evaluated seven different tasks in the SUPERB benchmark, our proposed method called Adaptive CHILD-TUNING (AD-TUNING) only needs to fine-tune less than $0.1\%\sim10\%$ of parameters for each task to achieve better performance than full fine-tuning. Benefited from our efficient adaptive early-stop algorithm, we achieved state-of-the-art results in most of the tasks. For instance, the DER of Speaker Diarization (SD) task is 9.22% relatively lower than the previously reported best results. Other benchmark results are also very competitive.

2. Review of CHILD-TUNING

2.1. Overview of CHILD-TUNING

Before introducing CHILD-TUNING, we first present a general formulation of back propagation during vanilla fine-tuning. The model's parameters at the t-th iteration are denoted as w_t and w_0 denotes the pre-trained parameters. During vanilla fine-tuning, all parameters are updated by the gradient descent of the loss function $\mathcal{L}(w_t)$. This process can be expressed as:

$$w_{t+1} = w_t - \eta \frac{\partial \mathcal{L}(w_t)}{\partial w_t} \tag{1}$$

*Corresponding author

10.21437/Interspeech.2023-1167

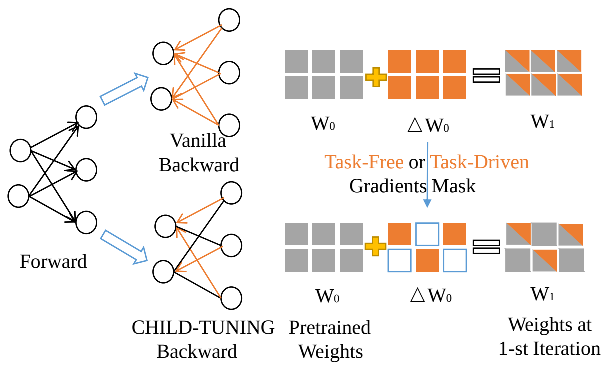

Figure 1: *Illustration of the difference between CHILD-TUNING and Vanilla fine-tuning. CHILD-TUNING forwards on the whole network while backwarding on a subset of network (i.e., child network), To achieve this, a task-free or task-driven mask is performed on the gradients of the non-child network, resetting them to zero.*

where $\frac{\partial \mathcal{L}(w_t)}{\partial w_t}$ are the gradients corresponding to the model parameters w_t, η is the learning rate.

Like vanilla fine-tuning, CHILD-TUNING calculates the gradients for all trainable parameters. However, the key difference is that CHILD-TUNING determines a child network C and only updates this part of parameters. Unlike [21], our child network does not change after determining. To achieve this, a 0-1 mask with the same size as the parameter w is defined as follows:

$$M^{(i)} = \begin{cases} 1, & w^{(i)} \in C \\ 0, & w^{(i)} \notin C \end{cases} \quad (2)$$

where $M^{(i)}$ and $w^{(i)}$ denote the i-th element of the mask M and parameters w, respectively. M will not be changed once it is confirmed. Then, the parameter update at step t of CHILD-TUNING is represented as follows:

$$w_{t+1} = w_t - \eta \frac{\partial \mathcal{L}(w_t)}{\partial w_t} \odot M \quad (3)$$

The comparison of vanilla fine-tuning and CHILD-TUNING can be seen in the description of Fig.1.

2.2. Task-Free variant: CHILD-TUNING$_F$

At the beginning, a task-free child-tuning (CHILD-TUNING$_F$) is adopted which doesn't require any downstream task data. Specifically, CHILD-TUNING$_F$ generates a 0-1 mask M at the 1-st iteration drawn from a Bernoulli distribution with a probability P_F.

$$M \sim \boldsymbol{Bernoulli}(P_F) \quad (4)$$

As P_F increases, the child network grows larger, resulting in updates to more parameters. if P_F equals 1, CHILD-TUNING$_F$ becomes equivalent to vanilla fine-tuning.

2.3. Task-Driven variant: CHILD-TUNING$_D$

CHILD-TUNING$_D$ takes into account the labeled data for the downstream task and identifies the most significant child network. Fisher information estimation [23] is used to find a highly relevant subset of parameters for a specific downstream task.

Formally, the Fisher Information Matrix (FIM) for the model parameters w is defined as follows:

$$\mathbb{F}(\boldsymbol{w}) = \mathbb{E}\left[\left(\frac{\partial \log \mathbb{P}(y|x;w)}{\partial w}\right)\left(\frac{\partial \log \mathbb{P}(y|x;w)}{\partial w}\right)^T\right] \quad (5)$$

where x and y denote the input and the output, respectively. Following [24], given the task-specific training data D, CHILD-TUNING$_D$ uses the diagonal elements of the empirical FIM to point-estimate the task-related importance of the parameters. In form, the Fisher information for the i-th parameter is as follows:

$$\mathbb{F}^{\boldsymbol{i}}(\boldsymbol{w}) = \frac{1}{|D|}\sum_{j=1}^{|D|}\left(\frac{\partial \log \mathbb{P}(y_j|x_j;w)}{\partial w^{(i)}}\right)^2 \quad (6)$$

CHILD-TUNING$_D$ assumes that the more important the parameter toward the target task, the higher Fisher information it conveys. Hence the child network C is comprised of the parameters with the highest information. The child network ratio is denoted as $P_D \in [0,1]$, which determines the size of the child network we select according to the ranking of information. As P_D increases, the size of the child network also grows. And if P_D equals 1, the approach degenerates into vanilla fine-tuning. For simplicity, we use P to denote P_D in subsequent sections.

3. Efficient hyperparameter optimization

3.1. Analysis of the hyperparameter P

In order to determine the appropriate range of P, unlike the arbitrary division in [21], we analyzed the parameter importance distribution. We found similar distribution shapes between different tasks described in Section 4.1, two are shown in Fig.2.

By analyzing the distribution of parameter importance, on the left side of the red line (corresponding to $P >= 0.1$), we observe there are a large number of parameters with low importance gathered. If we arbitrarily increase P, we can only cover more inefficient parameters, but they contribute little to the gradient of the parameters and the update of the model. As we want P not to be too big (for parameter-efficient tuning), and should take the important parameters as much as possible. So the value of P less than 0.1 is more reasonable.

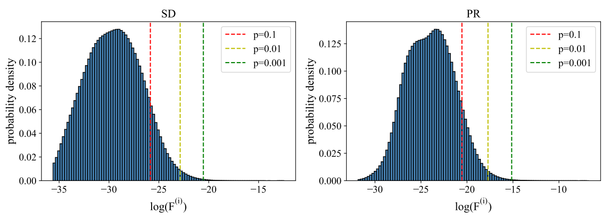

Figure 2: *Fisher information distribution of SD and PR tasks. Fisher information value is scaled by log function. The large importance values are clustered in a small portion on the right.*

3.2. Efficient adaptive early-stop algorithm

Intuitively, the optimal value of P may change corresponding to different tasks and even varies depending on the learning rate. Therefore, within our determined range P, we coarsely selected three different values of P ($P_1 = 0.001, P_2 = 0.01, P_3 = 0.1$, respectively) in order to find the appropriate P value for the diverse tasks as much as possible. In Table 2, our experiments confirmed this view. For example, in the case of Speaker Identification (SID) and ASR tasks, smaller P values outperform larger ones, whereas for the Phoneme Recognition (PR) task, larger P values are more appropriate. In our view, the occurrence of this phenomenon can be attributed not only to the task type but also to the data size, which we discuss further in Section 4.4. Thus there is an intractable problem: without any priori knowledge about the tasks, and the data, it is difficult to directly determine how large a child network is most appropriate.

Fortunately, by observing the training process of different child networks, we found that distinct trends emerge early in the training process. The training curves of different tasks in Fig.3 indicate that continuing to train the child networks with performed poorly early on usually did not yield better results.

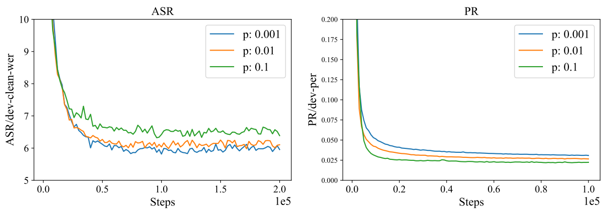

Figure 3: *The promising child network can be distinguished early in training. The plots show training curves for three child networks that different sizes on ASR, PR task respectively.*

Based on the above observations and inspired by the work in AutoML [25][26], we propose a simple yet efficient adaptive early-stop algorithm, to select the most promising child network. The complete pipeline is shown in Fig.4.

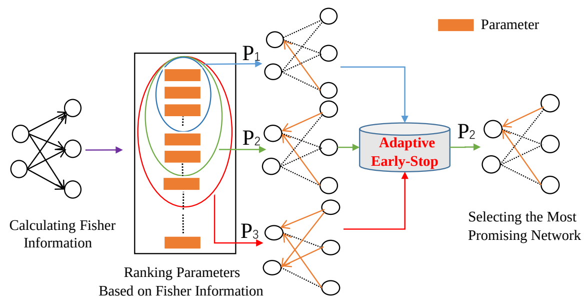
Parameter
Calculating Fisher Information
Ranking Parameters Based on Fisher Information
P1
P2
P3
Adaptive Early-Stop
P2
Selecting the Most Promising Network

Figure 4: *The pipeline of our proposed AD-TUNING.*

Specifically, the algorithm can be determined by four parameters: r, T, n, and c. We can start with n different sizes of child networks (i.e., the number of different P), and train each one in $r \times T$ steps, where T denotes the total number of training steps and r denotes the early training ratio. Then all child networks are evaluated, and only the c most promising ones are fully trained. The evaluation can be based on comparing the performance of the respective task on the development set. So our algorithm is highly efficient since it only runs for $(rn + c(1 - r))T$ steps in total, in contrast to nT steps.

4. Experiment

4.1. Experiment setup

We have verified our proposed AD-TUNING on seven different tasks in the SUPERB benchmark [17], including ASR, PR, SD, SID, Spoken Slot Filling (SF), Spoken Intent Classification (IC), and Keyword Spotting (KS). We developed the experiments based on s3prl[1]. We used Wav2vec2 [27] as an upstream model and kept the feature extraction layer frozen. The weighted sum of multiple hidden states from the upstream model is the final representation. For our adaptive early-stop algorithm, we set n to 3, which corresponds to P equal to $0.001, 0.01, 0.1$, respectively, r to 0.1 or 0.2, and c to 1. It means we select only the most promising child network. Consequently, AD-TUNING only needs to take $1.2T$ steps to complete training instead of $3T$ steps, which train each of the 3 networks separately and then choose the best result. We searched for the optimal learning rate for each downstream task in the range of 10^{-5} to 10^{-3} and used a linear warm-up strategy in our experiments.

4.2. CHILD-TUNING$_F$ vs CHILD-TUNING$_D$

We have done a preliminary experimental comparison of CHILD-TUNING$_F$ and CHILD-TUNING$_D$. The results are shown in Table 1, although CHILD-TUNING$_F$ is not a poor performer, our findings indicate that CHILD-TUNING$_D$ delivers superior results and greater interpretability, so following experiments we mainly work on CHILD-TUNING$_D$.

Method	ASR ↓	PR ↓	SD ↓	SID ↑	IC ↑
CHILD-TUNING$_F$	6.58	2.65	3.7	**79.26**	99.12
CHILD-TUNING$_D$	**6.53**	**2.45**	**3.38**	78.03	**99.34**

Table 1: *Comparison of CHILD-TUNING$_F$ and CHILD-TUNING$_D$ on different tasks when tuning only the wav2vec2 10 percent parameters.*

4.3. Performance in the SUPERB benchmark

To examine the effectiveness of our proposed AD-TUNING, we compared it to different efficient methods in the SUPERB benchmark and the result is shown in Table 2. Note that 'FT' represents fine-tuning, 'Houlsby' is Houlsby adapter with bottleneck 32, 'CNN adapter' only adds CNN adapter to the feature extractor without utilizing Houlsby adapter in the transformer layers, and 'CHAPTER' [13] adds CNN adapter to the feature extractor while inserting Houlsby adapters into the transformer layers. The 'Baseline' here means only tuning the downstream model. For details of other methods, please see [17][28]. According to the experimental results, our adaptive early stopping algorithm successfully identified suitable values of P. Compared to the previously reported best results, AD-TUNING makes the DER of the SD task and the PER of the PR task drop relatively by 9.22% and 3.67%, respectively. As with 'CHAPTER', AD-TUNING is substantially ahead of other approaches on the SID task, achieved an accuracy rate of 91.66%. And on the other tasks, AD-TUNING shows very competitive performance. It is worth noting that there may be some unfairness in the comparison made here. CHILD-TUNING is not necessarily superior to the other methods when the child network remains fixed. The reason is that AD-TUNING allows for the selection of different child networks depending on the task with minimal additional overhead, whereas the other methods lack the ability to make changes once the structure has been fixed, which of course we believe is one of our advantages.

4.4. Low-resource adaptation

In NLP, adapters are shown to have advantages over finetuning when adapting to low-resource datasets [29][30]. To see

[1]`https://github.com/s3prl/s3prl`

Method	Params	ASR ↓	PR ↓	SD ↓	SID ↑	SF ↑	IC ↑	KS ↑
FT	94.7M	6.35	2.45	3.58	66.48	84.87	99.10	95.87
Baseline	0	7.09	7.74	7.05	64.78	86.25	96.39	95.32
Houlsby	0.60M	5.88	3.00	4.00	87.71	85.87	**99.60**	97.17
AdapterBias	0.02M	**5.54**	4.19	5.48	77.38	86.60	99.50	97.30
LoRA	0.29M	6.94	8.74	7.39	62.90	86.25	96.57	96.59
Prefix	0.10M	6.56	4.18	8.17	71.87	85.85	99.31	97.05
CNN adapter	4.07M	6.32	5.42	5.7	78.57	**86.81**	98.39	97.20
CHAPTER	4.67M	6.22	2.95	3.84	91.56	85.94	99.20	95.52
Weighted-sum	12	6.42	5.41	5.88	81.42	86.71	98.34	96.30
CHILD-TUNING (p=0.1)	8.5M	6.53	2.45	3.42	78.03	82.78	99.34	97.34
CHILD-TUNING (p=0.01)	0.85M	6.195	2.76	3.38	**91.88**	85.28	99.47	97.34
CHILD-TUNING (p=0.001)	0.085M	5.96	3.10	4.05	84.35	85.99	99.45	97.11
AD-TUNING (**ours**)	{8.5M, 0.85M, 0.085M}	6.01	**2.36**	**3.25**	91.66	86.10	99.53	**97.37**

Table 2: *Performance of different efficient methods in the SUPERB benchmark. The second column represents trainable parameter used in upstream model. Our AD-TUNING will select the most suitable one from three different child networks. The subtle differences in the results corresponding to the same child networks are caused by the randomness in training.*

if CHILD-TUNING also has such property when applied in speech tasks, we conducted experiments in the low-resource settings. We trained our models on 0.5-hour, 1-hour, 5-hour, and 10-hour datasets generated by Libri-Light [31] and tested on the testing set of LibriSpeech. As illustrated in Fig.5, our experiments suggest that a child network with minimal parameter updates performs best for small datasets. This is likely due to the high risk of overfitting to a small dataset when there are substantial parameter updates. As the number of data increases, the advantage of updating a large number of parameters is gradually shown. We also compared our proposed AD-TUNING with other methods [17], as shown in Fig.6. We found that AD-TUNING can also choose the appropriate child network based on the size of the dataset and perform well.

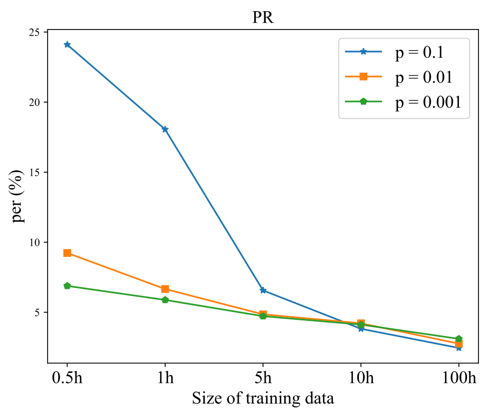

Figure 5: *Performance of CHILD-TUNING in low-resource adaptation. The x-axis represents the size of training data, while the y-axis represents performance of PR. For PR, we report phone error rate (PER).*

4.5. Analysis of parameter updated for each layer

We visualized the ratio of parameters updated for each transformer layer on different tasks, as shown in Fig.7. We found that updated parameters are more intensive at the front and last few layers. One common characteristic is that the parameter updates in the intermediate layer are relatively few, indicating that CHILD-TUNING tends to adjust the model near the input and output parts based on the task. We believe that it is due to the part close to the model input plays a key role in preprocessing the features, while the part close to the model output plays an important connecting role for adaptation to downstream tasks.

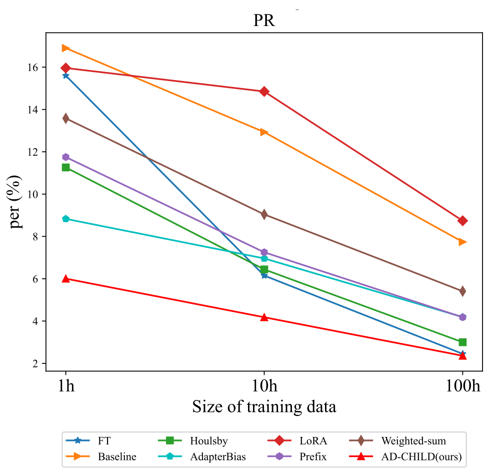

Figure 6: *Performance of different low-resource data in efficient methods. The coordinates are the same as described in Fig.5*

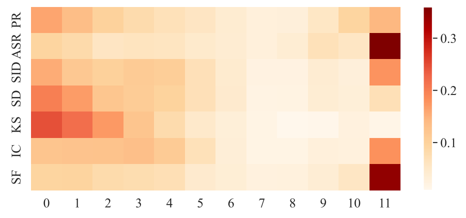

Figure 7: *The ratio of parameter updated in each layer.*

5. Conclusions

In this paper, we introduce CHILD-TUNING for various speech tasks and propose an efficient adaptive early-stop algorithm to select the appropriate child network called AD-TUNING. AD-TUNING is not only efficient in training time compared to training different child networks separately, but also achieves excellent results in the SUPERB benchmark. Furthermore, this work provides a successful paradigm for speech community to explore ways to efficiently fine-tune the speech pre-trained model itself in the future.

6. Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant 62171427.

7. References

[1] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660.

[2] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 000–16 009.

[3] S. Gidaris, A. Bursuc, G. Puy, N. Komodakis, M. Cord, and P. Pérez, "Obow: Online bag-of-visual-words generation for self-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6830–6840.

[4] J. D. M.-W. C. Kenton and L. K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, 2019, pp. 4171–4186.

[5] J. Dodge, G. Ilharco, R. Schwartz, A. Farhadi, H. Hajishirzi, and N. Smith, "Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping," *arXiv preprint arXiv:2002.06305*, 2020.

[6] 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," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[7] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "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.

[8] 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/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021.

[9] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," in *International Conference on Learning Representations*, 2020, pp. 1–12.

[10] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2790–2799.

[11] A. Rücklé, G. Geigle, M. Glockner, T. Beck, J. Pfeiffer, N. Reimers, and I. Gurevych, "Adapterdrop: On the efficiency of adapters in transformers," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 7930–7946.

[12] J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, and G. Neubig, "Towards a unified view of parameter-efficient transfer learning," in *International Conference on Learning Representations*, 2022, pp. 1–15.

[13] Z.-C. Chen, Y.-S. Sung, and H.-y. Lee, "Chapter: Exploiting convolutional neural network adapters for self-supervised speech models," *arXiv preprint arXiv:2212.01282*, 2022.

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

[15] S. Kessler, B. Thomas, and S. Karout, "An adapter based pre-training for efficient and scalable self-supervised speech representation learning," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 3179–3183.

[16] W. Hou, H. Zhu, Y. Wang, J. Wang, T. Qin, R. Xu, and T. Shinozaki, "Exploiting adapters for cross-lingual low-resource speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 317–329, 2021.

[17] Z.-C. Chen, C.-L. Fu, C.-Y. Liu, S.-W. D. Li, and H.-y. Lee, "Exploring efficient-tuning methods in self-supervised speech models," in *2022 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2023, pp. 1120–1127.

[18] S. Otake, R. Kawakami, and N. Inoue, "Parameter efficient transfer learning for various speech processing tasks," *arXiv preprint arXiv:2212.02780*, 2022.

[19] J. Pfeiffer, A. Rücklé, C. Poth, A. Kamath, I. Vulić, S. Ruder, K. Cho, and I. Gurevych, "Adapterhub: A framework for adapting transformers," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 46–54.

[20] D. Lian, Z. Daquan, J. Feng, and X. Wang, "Scaling & shifting your features: A new baseline for efficient model tuning," in *Advances in Neural Information Processing Systems*, 2022, pp. 1–20.

[21] R. Xu, F. Luo, Z. Zhang, C. Tan, B. Chang, S. Huang, and F. Huang, "Raise a child in large language model: Towards effective and generalizable fine-tuning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 9514–9528.

[22] Z. Haojie, G. Li, J. Li, Z. Zhang, Y. Zhu, and Z. Jin, "Fine-tuning pre-trained language models effectively by optimizing subnetworks adaptively," in *Advances in Neural Information Processing Systems*, 2022, pp. 1–14.

[23] M. Tu, V. Berisha, M. Woolf, J.-s. Seo, and Y. Cao, "Ranking the parameters of deep neural networks using the fisher information," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 2647–2651.

[24] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[25] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6765–6816, 2017.

[26] K. Jamieson and A. Talwalkar, "Non-stochastic best arm identification and hyperparameter optimization," in *Artificial intelligence and statistics*. PMLR, 2016, pp. 240–248.

[27] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.

[28] E. J. Hu, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, "Lora: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022, pp. 1–26.

[29] C.-L. Fu, Z.-C. Chen, Y.-R. Lee, and H.-Y. Lee, "Adapterbias: Parameter-efficient token-dependent representation shift for adapters in nlp tasks," in *Findings of the Association for Computational Linguistics: NAACL 2022*, 2022, pp. 2608–2621.

[30] R. He, L. Liu, H. Ye, Q. Tan, B. Ding, L. Cheng, J. Low, L. Bing, and L. Si, "On the effectiveness of adapter-based tuning for pretrained language model adaptation," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2021, pp. 2208–2222.

[31] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P. E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux, "Libri-light: A benchmark for asr with limited or no supervision," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7669–7673.