Dropout Mixture Low-Rank Adaptation for Visual Parameters-Efficient Fine-Tuning

Zhengyi Fang[1*], Yue Wang[1*], Ran Yi[1✉], and Lizhuang Ma[1,2]

[1] Shanghai Jiao Tong University
[2] MoE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University
{oliverfang,imwangyue,ranyi,lzma}@sjtu.edu.cn
https://github.com/Oliver7th/DMLoRA

Abstract. Parameter-efficient fine-tuning methods adjust a small subset of parameters in large models, achieving performance comparable to or even surpassing that of models fine-tuned with the full parameter set, and significantly reducing the time and computational costs associated with the fine-tuning process. Despite the developments of parameter-efficient fine-tuning methods for large models, we observe significant performance disparities across different vision tasks. We attribute this pronounced performance variability to the insufficient robustness of current parameter-efficient fine-tuning methods. In this paper, we propose a robust reparameterization framework for parameter-efficient fine-tuning. This framework has a dynamic training structure and introduces no additional computational overhead during the inference stage. Specifically, we propose Dropout-Mixture Low-Rank Adaptation (DMLoRA), which incorporates multiple up and down branches, to provide the model with a more robust gradient descent path. As training proceeds, DMLoRA gradually drops out branches to achieve a balance between accuracy and regularization. Additionally, we employ a 2-Stage Learning Scalar (LS) strategy to optimize the scale factor for each layer's DMLoRA module. Experimental results demonstrate that our method achieves state-of-the-art performance on the benchmark VTAB-1k and FGVC datasets for parameter-efficient fine-tuning.

Keywords: Paramter-Efficient Fine-Tuning · Dropout-Mixture Low-Rank Adaptation · Gradual Branch Dropout · 2-Stage Learning Scalar

1 Introduction

In the era of large models, training a large model consumes a significant amount of time and computational resources. Fine-tuning based on pretrained large models is the prevailing paradigm for their applications. However, for large models with a large number of parameters, fine-tuning on limited training data often

✉ Corresponding Author.
* Equal Contributions.

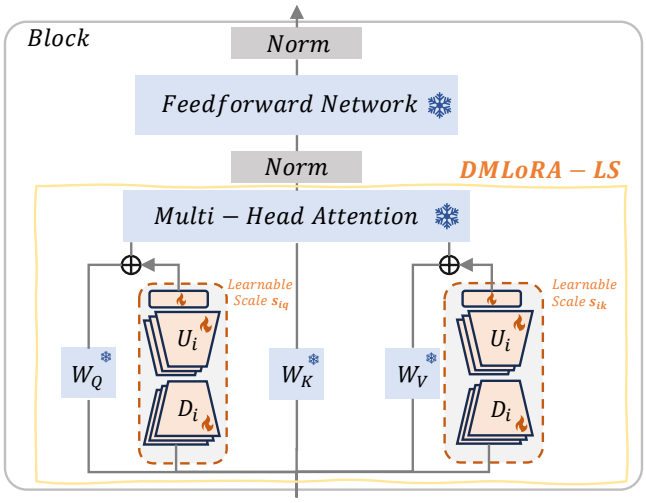

Fig. 1: Our robust PEFT network contains Dropout-MixLoRA (*DMLoRA*), and 2-Stage Learning Scalar (*LS*). The trainable components are highlighted in orange.

leads to severe overfitting, resulting in a notable decline in performance after fine-tuning. To address this issue, fine-tuning a small subset of parameters in large models can effectively prevent overfitting, while also saving training time and computational resources. This type of approach is known as Parameters-Efficient Fine-Tuning (PEFT).

Currently, most large models are based on the Transformer architecture, so the existing PEFT methods mainly revolve around discussions concerning Transformer models. Generally, a Transformer consists of multiple blocks, each containing self-attention layers (or cross-attention) and feedforward network layers. PEFT methods can be categorized into *incremental parameter-based* and *non-incremental parameter-based* approaches. Houlsby [14] proposed an incremental parameter-based method by adding extra Adapter layers within the Transformer block, while some methods opt to select existing parameters or reparameterize newly trained parameters into the original model, constituting non-incremental parameter approaches. Unlike incremental parameter methods, non-incremental parameter methods introduce no additional computational overhead during model inference and preserve the original model structure. In this paper, our proposed method falls into the category of non-incremental parameter methods. Compared to previous non-incremental parameter methods, our trained model demonstrates enhanced robustness.

In the field of natural language processing, PEFT has seen considerable developments. Since vision large models emerge slightly later than their language counterparts, the progress of PEFT methods in the vision domain has lagged behind that of natural language processing. Considering that the Transformer models used in the vision tasks are similar to those in natural language process-

ing, some efficient fine-tuning methods from natural language processing can be applied to vision tasks. However, compared to natural language processing tasks, vision tasks need to deal with more complex and variable data, *e.g.,* different resolutions in training and inference, different scales of objects in images, which hinders the direct applicability of existing PEFT methods from natural language processing to vision. From our experimental observations, we have identified distinct advantages among various PEFT strategies in different vision datasets and tasks. We believe this discrepancy stems from insufficient regularization during the model training process. To address these issues, in this paper, we explore several methods aimed at enhancing model robustness, and achieve state-of-the-art results on validation datasets.

We propose a robust PEFT approach with a dynamic training structure, as shown in Fig. 1. Firstly, observing that the optimization direction during the early stages of parameter-efficient fine-tuning significantly impacts the final results, we propose a *Dropout Mixture Low-Rank Adaptation (DMLoRA)* approach, to better guide the optimization direction and impose stronger regularization during the initial training stages, while gradually relaxing the regularization as training proceeds. At the outset of training, we initialize multiple branches of Up and Down matrices, which are randomly selected during training. As training progresses, we gradually reduce the number of branches with random dropout, achieving a balance between regularization and model accuracy. Secondly, we discover that the scale factor significantly impacts reparameterization methods. To address this issue, we propose a *2-Stage Learning Scale Training (LS)* method. Initially, re-parameterization branches are trained with an appropriate scale. Subsequently, with fixed branch model parameters, we optimize the scale factor to redistribute the influence of each layer's LoRA (Low-Rank Adaptation) module. Extensive experiments demonstrate that our method achieves state-of-the-art performance. Our contributions can be summarized as follows:

- We propose a robust and parameter-efficient fine-tuning method with a dynamic training structure, which surpasses previous state-of-the-art methods in fine-tuning performance, without introducing additional computational overhead during inference.
- We propose Dropout Mixture Low-Rank Adaptation (DMLoRA), which incorporates multiple LoRA branches to impose stronger regularization constraints during the early stages of model training, and gradually reduces the number of branches as training progresses, greatly enhancing the performance of Low-Rank Adaptation.
- We propose 2-Stage Learning Scalar strategy, which uses a 2-Stage Learning scheme to optimize the scale factor for each layer's DMLoRA module, enhancing the performance of the Adapter.

2 Related Work

2.1 Vision Transformer and Pre-traing

The introduction of the Transformer [37] network structure based on Self-Attention marked the onset of the era of large models in both academia and industry. While Transformers were initially applied in natural language processing, subsequent research [5,26,39] has demonstrated that the Transformer architecture also brings significant performance improvements to vision tasks in the fields of detection and generation. Moreover, for Transformer models, whether in the CV or NLP, pre-training is a crucial stage. Existing pre-training methods include supervised methods [5], unsupervised methods (or self-supervised methods) [2,12,13], and cross-modal methods [21,33]. These methods aim to provide a set of initialized parameters for subsequent downstream task training. Current research indicates the importance of pre-training for the performance of Transformer models in downstream tasks, as different pre-training strategies can lead to noticeable differences in model performance metrics. The central focus of this paper is precisely fine-tuning vision model parameters based on a set of pre-trained weights. To ensure fairness, we adopt prior work's model and pre-training methods.

2.2 Parameters-Efficient Fine-Tuning

In the era of large models [1,31,35], model parameters' scale reaches trillions, intensifying time and resource demands for training. To fine-tune pre-trained large models, two intuitively apparent approaches are full-parameter fine-tuning and fine-tuning only the Head network for specific tasks. Generally, full-parameter fine-tuning can achieve better results when there is sufficient data. However, due to the involvement of all parameters in the optimization process, the computational resources and time required for full-parameter fine-tuning are often substantial. On the other hand, fine-tuning only the Head network for a specific task has a smaller computational overhead, but the model's accuracy is noticeably lower compared to full-parameter fine-tuning. Parameters-Efficient Fine-Tuning (PEFT) [4, 23] is proposed to strike a balance between model accuracy and training efficiency. Current PEFT methods have shown performance comparable to or even surpassing full-parameter fine-tuning. This is achieved by adjusting key parts of the Transformer parameters in PEFT methods, effectively avoiding overfitting issues that may arise from optimizing too many parameters.

Existing PEFT methods can be categorized into Adapter methods, Prompt methods, parameter selection methods, reparameterization methods, and composite structure methods. *(1) Adapter methods* [3,14,17,25,32,34,40] primarily involve inserting lightweight networks in the middle of Transformer Block layers to adjust the content of hidden layers. Most Adapters are serially inserted between Transformer Blocks [3, 14, 32, 34, 40], but there are also a few parallel structures [11] with Block sub-modules (e.g., Attention, FFN). *(2) Prompt methods* [16,20,22] modify the hidden layers themselves, as there are multiplication operations in the Attention network, making the hidden layers operators. In

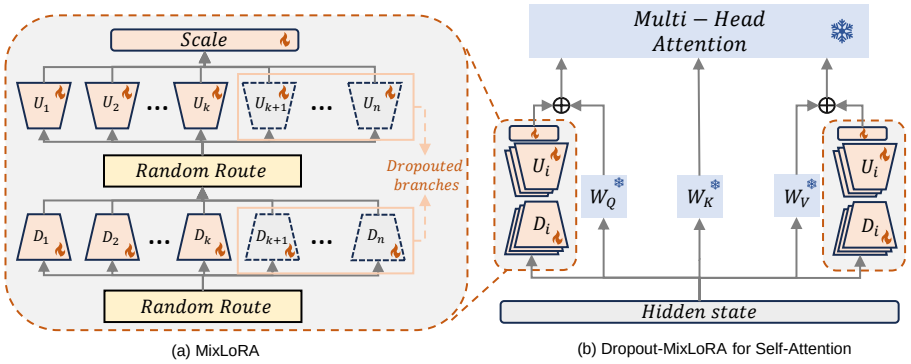

Fig. 2: Our re-parameterization module design. (a) Our *MixLoRA* contains multiple up and down branches, contributing to the enhancement of the model's robustness. (b) During training, we gradually drop out some branches for higher performance, and we call the whole algorithm as *Dropout-MixLoRA (DMLoRA)*. Our DMLoRA is applied to the linear transformation matrices of Query and Value in the Attention module, where the Scale parameter is learned with a *2-Stage Learning Scalar (LS)* strategy.

terms of placement, some methods [16, 20] insert prompts between each Block, while others [22] directly expand the matrices that require multiplication. *(3) Parameter selection methods* [9, 41] involve optimizing only some "important" parameters while keeping others unchanged. These methods do not require introducing additional structures, but selecting these parameters is a challenging task. *(4) Reparameterization methods* [6, 7, 15, 27] add modules that can be merged with the original network through reparameterization algorithms, stemming from researchers' observations of changes in the rank of parameter matrices. *(5) Composite methods* [11, 24, 28, 29] combine the above approaches. Currently, individual PEFT methods show pros/cons under varied data, hinting at multi-strategy integration as the future path. Adapter and Prompt methods introduce additional inference overhead due to the inclusion of parameters that cannot be eliminated. In contrast, parameter selection and reparameterization methods [7, 9, 15, 27, 41] do not modify the inference network structure, thus incurring no standalone computational overhead during inference. This paper primarily focuses on the improvement of reparameterization methods, ensuring that our framework introduces no additional computational overhead during the inference stage.

3 Method

3.1 Overview

Based on our experimental observations, previous Parameters-Efficient Fine-Tuning (PEFT) methods exhibit significant performance disparities across different datasets. We attribute this discrepancy to the insufficient robustness of

these parameter-efficient fine-tuning methods. To address this issue, we propose a dynamic training structure reparameterization approach to enhance robustness during fine-tuning, which consists of two novel designs : **(1) Dropout-Mixture Low-Rank Adaptation (DMLoRA):** In the initial stages of training, we establish multiple Up and Down matrices for Low-Rank Adaptation, randomly matching them to enhance the robustness of training (MixLoRA); and as training proceeds, we gradually reduce the number of branches (Mixture Dropout) to balance regularization and performance (Sec. 3.2); **(2) 2-Stage Learning Scalar (LS):** Observing the substantial impact of Scale parameters on the performance of reparameterization methods, we propose a two-stage Scale learning approach to enhance the performance of the reparameterization method (Sec. 3.3).

3.2 Dropout-Mixture Low-Rank Adaptation

For large model fine-tuning tasks, the amount of training data used for fine-tuning is significantly less than the training data used for pre-training the large models. Consequently, in the early stages of fine-tuning, the training tends to be unstable. Additionally, since the performance of Transformer models is greatly influenced by the optimization direction of initial parameters, finding a robust optimization direction is crucial during the initial stages of training. For a robust model, when we make slight modifications to a subset of its weight parameters, we expect that the performance will not be significantly affected. Inspired by AdaMix [40], we expand the up and down matrices of Low-Rank Adaptation into *multiple randomly selected branches*, aiming to enhance the robustness of the model during training.

Although expanding Low-Rank Adaptation into multiple branches can improve robustness, it increases training burden. Meanwhile, when the training data is limited (less than 1,000 samples), overly emphasizing regularization in model training often leads to a decrease in model performance. We observed that models based on Self-Attention require better control and training strategies in the early stages of training (*e.g.*, a better initial optimization direction and a gradually increasing learning rate strategy). Based on this observation, we propose to employ a stronger regularization strategy during the early stage of training to guide the model with a good initial direction, while gradually relaxing the regularization as training progresses. Specifically, in the initial stages of training, we set a large number of down and up matrices; and we *gradually reduce the number of trainable branches as training proceeds*, increasing the probability of the remaining branches being selected. In this way, we achieve a balance between training regularization and accuracy.

Next, we first introduce Mixture Low-Rank Adaptation, our design of expanding Low-Rank Adaptation into multiple randomly selected branches to improve robustness; and then introduce Mixture Dropout, our design of gradually reducing the number of branches as training proceeds, to balance regularization and performance; and finally summarize the differences to AdaMix.

Mixture Low-Rank Adaptation. Low-Rank Adaptation (LoRA) is based on experimental observations: during fine-tuning of large models, the update ΔW of weight parameters forms a non-full-rank matrix. According to matrix theory, an $N \times N$ matrix with rank r can be decomposed into two matrices, $N \times r$ and $r \times N$. After fine-tuning, the model's weight parameters $W + \Delta W$ can be expressed as $W + BA$ with a low-rank decomposition, where $B \in \mathbb{R}^{N \times r}, A \in \mathbb{R}^{r \times N}$. Since the parameter number of matrices A and B is much smaller than that of W, this substitution significantly reduces the number of parameters during fine-tuning. Following [15], we apply this Low-Rank Adaptation strategy in the linear transformations of Q and V in the Attention module. The adjusted linear layer can be formulated by the following equation:

$$\begin{aligned} f(x) &= (W + \Delta W)x + b \\ &= (W + s \cdot BA)x + b \\ &= W'x + b, \end{aligned} \tag{1}$$

where s is a scale factor. Since matrices A and B can be integrated into W' through mathematical calculations, there is no additional computational cost during the inference stage.

We introduce new branches based on this formula, forming a *Mixture Low-Rank Adaptation (MixLoRA)*. In our design, both matrices A and B are *expanded into multiple branches and randomly selected* during the training process. We denote this random routing as RR, and the branch number as n. Our MixLoRA can be formulated as:

$$f_{multi}(x) = (W + s \cdot \mathcal{RR}(\mathbb{B})\mathcal{RR}(\mathbb{A}))x + b, \tag{2}$$

where $\mathbb{A} = \{A_i | i \leq n, i \in \mathbb{N}\}$, $\mathbb{B} = \{B_i | i \leq n, i \in \mathbb{N}\}$, and $\mathcal{RR}$ is the random routing algorithm to randomly select a matrix A and a matrix B from $\mathbb{A}$ and $\mathbb{B}$ respectively. Fig. 2 shows the structure of Mixture Low-Rank Adaptation.

Mixture Dropout. Mixture is a powerful regularization training method; and using multiple branches in the early stages of training helps the model to find better initial parameters. However, as the model accuracy improves, multiple branches may result in overly strong regularization, hindering accuracy improvement. Therefore, we propose to *gradually reduce the number of branches* as training proceeds, to strike a balance between regularization and model performance.

We initialize n Up branches and n Down branches at the beginning of training. After k training epochs, we reduce the number of branches by half, for example, from n Up branches to $n/2$. We cyclically perform the halving operation for each k epochs until n decreases to a specified number. This process is illustrated in Fig. 3. During each halving, we adopt a random dropout strategy to remove half of the branches. In our experiments, we have compared two strategies: *random dropout halving* (randomly dropping out half branches) and *average aggregating halving* (averaging the parameters of two branches to aggregate into one). The experiments demonstrate that the random dropout halving outperforms the average aggregating halving strategy.

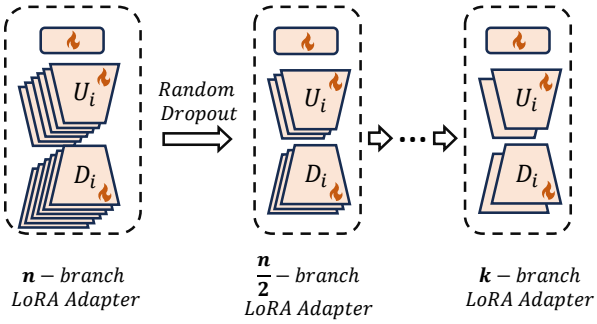

Fig. 3: Mixture Dropout. We initiate training with a higher number of LoRA branches and gradually reduce the number of branches as training progresses until the number decreases to a specified number. This strategy imposes a stronger regularization on the model in the early stages of training, and gradually relaxes the regularization as training proceeds, thereby enhancing fine-tuning performance.

Mixture Evaluation. To evaluate the performance of a model with multiple branches, we form an inference model by averaging the parameters of each branch (averaging the parameters of multiple Up and Down matrices separately). Since the additional parameters will be integrated into the original model parameters, similar to Eq. (1), our fine-tuning method in the scenario of multiple branches does not incur additional computational overhead during the inference stage.

Differences to AdaMix. The multi-branch architecture in our Dropout Mixture LoRA, is similar to the previously mentioned AdaMix structure, which incorporates the concept of multi-branch adapters. However, compared to AdaMix, our Dropout Mixture LoRA has two main differences: (1) AdaMix maintains a constant number of branches (n) throughout the training phase, only averaging the parameters of n branch adapters during inference. This design introduces strong regularization during the early stages of training and enhances model robustness; but, as training progresses, it imposes excessive regularization and significantly restricts the parameter optimization space, leading to decreased training efficiency and affecting final performance. In contrast, our Dropout Mixture LoRA enhances regularization with multiple branches during the early training stages, while continuously prunes branches as the training proceeds, *i.e.*, the number of branches gradually decreases during training, instead of keeping constant as AdaMix. In this way, our approach ensures high efficiency in later training phases and avoids performance decrease caused by excessive regularization. (2) While AdaMix employs naive adapters as the unit structure for branches, Dropout Mixture LoRA utilizes the more powerful LoRA(against naive adapter) as the unit structure, undoubtedly improving model performance. Further experimental comparison between the two strategies will be reported in Sec. 4.4.

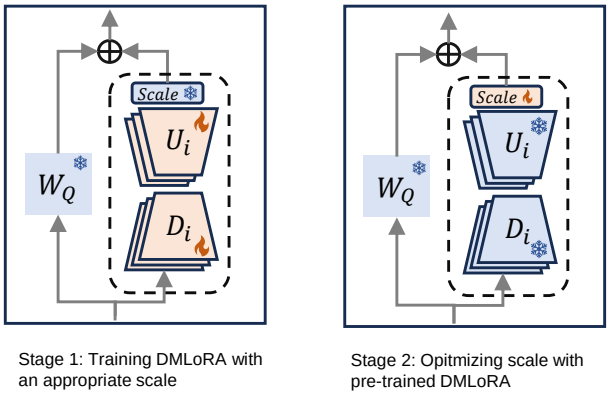

Fig. 4: 2-Stage Learning Scalar. We train the model in two stages: (1) *Stage 1* Appropriate Scaling: we first train the mixture reparameterization module with an appropriate scale selected from a predefined set.(2) *Stage 2* Learning Scale: we then fix the parameters of mixture reparameterization module and optimize the Scale parameters.

3.3 2-Stage Learning Scalar

For reparameterization methods, the choice of the *scale* parameter significantly impacts the final model's performance, as well-demonstrated by previous experimental results. Our experiments reveal that directly training the learnable scale parameter alongside the reparameterization module does not achieve the same effectiveness as treating the scale as a hyperparameter beforehand. We attribute this to the sensitivity of scale compared to other parameters and its variation spanning multiple orders of magnitude.

To address this problem, we propose 2-Stage Learning Scalar, a more robust method for learning the scale parameter. The specific process is illustrated in Fig. 4. During training, we use an appropriate scale for the reparameterization module. Our training procedure consists of two stages. 1) *Appropriate Scaling*: identifying an appropriate scale, and train the mixture reparameterization module with the scale. 2) *Learning Scale*: after completing the training of the reparameterization module, we freeze all Up and Down matrices and proceed to optimize the Scale parameters. This approach ensures that the obtained scale parameters have suitable values for Q and V components in each layer.

4 Experiments

4.1 Experimental Settings

Model Architecture. Our base model is built upon the architecture of ViT-B/16 [5] and is pre-trained on ImageNet21K in a supervised manner. ViT-B/16 consists of a total of 85.8 million trainable parameters and comprises 12 blocks.

Within each block, the hidden dimension of the self-attention layer is 768. For input images, ViT-B/16 reorganizes patches of size $16 \times 16 \times 3$ into 768-dimensional vectors, which are then fed into subsequent Transformer blocks.

Datasets. We validate our proposed method on multiple datasets, which can be classified into two categories:
VTAB-1K [42]: It comprises 19 tasks, with the image data further categorized into three major types: (1) Natural photos captured by standard or general cameras. (2) Special photos captured by specialized sensors such as remote sensing and medical imaging. (3) Structured photos synthesized in simulated environments. These tasks cover diverse content, including distance (or depth) prediction, object counting, and object classification. Each task in VTAB-1K contains only 1,000 images, making it highly challenging.
FGVC [16] (Fine-Grained Visual Classification): This dataset encompasses 4 tasks involving natural photos across 5 datasets, specifically: (1) Bird classification tasks with CUB-200-2011 [38] and NABirds [36] datasets. (2) Flower classification task with Oxford Flowers [30] dataset. (3) Dog classification task with Stanford Dogs [19] dataset. (4) Car classification task with Stanford Cars [8] dataset. Compared to VTAB-1K, the FGVC dataset has more training data but less image diversity.

Comparison Baselines. We compare our method with the previously parameter-efficient fine-tuning (PEFT) approaches. Our main comparison baselines include: 1) BitFit [41]: adjusts only the Bias in each layer during fine-tuning, keeping the Weight layer unchanged. 2) VPT [16]: introduces trainable prompts during fine-tuning, training only the prompt parameters. 3) E^2VPT [10]: on the basis of VPT [16], adds an embedding matrix at the embedding of key and value in the self attention layer. 4) Adapter [14]: inserts a serial Adapter after the FFN. 5) AdapterFormer [3]: embeds parallel Adapter branches within the FFN. 6) NOAH [43]: utilizes neural architecture search to discover optimal composite structures. 7) FacT [18]: implements a tensorization-decomposition framework to store weight increments. 8) SSF [24]: a composite method involving reparameterization and parameter selection. 9) RepAdapter [27]: introduces a serial reparameterization module.

Implementation Details. For the Dropout-Mixture Low-Rank Adaptation module, we downsample the 768-dimensional vectors to 8 dimensions. In Eq. (1), the dimension of B is 768×8, and the dimension of A is 8×768. We initialize the number of branches n between 4 to 32 based on the characteristics of the dataset. For the Scale parameter, we choose either 1 or 10. In the 2-Stage Learning Scale module, we set the optimization duration for scale as 10 epochs, with a learning rate of $1e-2$. We utilize the AdamW optimizer for all modules. For more information, please refer to our supplementary materials. For all tasks, we set the batch size as 64 on a Single GPU for training and set the input image resolution as 224×224.

Table 1: Results on VTAB-1K benchmark with ViT-B/16 models pre-trained on ImageNet-21K. "# params" specifies the number of trainable parameters in backbones. Average accuracy and # params are averaged over group-wise mean values. The bold data is **the best**, and the underlined data is the second best. Our method has the best average results.

			Natural							Specialized				Structured								
	# Param (M)	Inference Cost	Cifar100	Caltech101	DTD	Flower102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele	Average
Traditional Finetuning																						
Full fine-tuning	85.8	-	68.9	87.7	64.3	97.2	86.9	87.4	38.8	79.7	95.7	84.2	73.9	56.3	58.6	41.7	65.5	57.5	46.7	25.7	29.1	68.9
Linear probing	0	-	64.4	85.0	63.2	97.0	86.3	36.6	51.0	78.5	87.5	68.5	74.0	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	57.6
PEFT methods																						
BitFit [41]	0.10	-	72.8	87.0	59.2	97.5	85.3	59.9	51.4	78.7	91.6	72.9	69.8	61.5	55.6	32.4	55.9	66.6	40.0	15.7	25.1	65.2
VPT-Shallow [16]	0.06	↑	77.7	86.9	62.6	97.5	87.3	74.5	51.2	78.2	92.0	75.6	72.9	50.5	58.6	40.5	67.1	68.7	36.1	20.2	34.1	67.8
VPT-Deep [16]	0.53	↑	**78.8**	90.8	65.8	98.0	88.3	78.1	49.6	81.8	96.1	83.4	68.4	68.5	60.0	46.5	72.8	73.6	47.9	32.9	37.8	72.0
E^2VPT [10]	0.25	↑	78.6	89.4	67.8	98.2	88.5	85.3	52.3	82.5	**96.8**	84.8	73.6	71.7	61.2	47.9	75.8	80.8	48.1	31.7	41.9	73.9
Adapter [14]	0.16	↑	69.2	90.1	68.0	98.8	89.9	82.8	54.3	84.0	94.9	81.9	75.5	80.9	65.3	48.6	78.3	74.8	48.5	29.9	41.6	73.9
AdaptFormer [3]	0.16	↑	70.8	91.2	70.5	99.1	90.9	86.6	54.8	83.0	95.8	84.4	**76.3**	81.9	64.3	49.3	80.3	76.3	45.7	31.7	41.1	74.7
LoRA [15]	0.29	-	67.1	91.4	69.4	98.8	90.4	85.3	54.0	84.9	95.3	84.4	73.6	82.9	69.2	49.8	78.5	75.7	47.1	31.0	44.0	74.5
NOAH [43]	0.36	↑	69.6	**92.7**	70.2	99.1	90.4	86.1	53.7	84.4	95.4	83.9	75.8	82.8	68.9	49.9	**81.7**	**81.8**	48.3	32.8	**44.2**	75.5
FacT [18]	0.07	-	70.6	90.6	70.8	99.1	90.7	88.6	54.1	84.8	96.2	84.5	75.7	82.6	68.2	49.8	80.7	80.8	47.4	33.2	43.0	75.6
SSF [24]	0.24	-	69.0	92.6	**75.1**	**99.4**	91.8	90.2	52.9	**87.4**	95.9	**87.4**	75.5	75.9	62.3	**53.3**	80.6	77.3	**54.9**	29.5	37.9	75.7
RepAdapter [27]	0.22	-	72.4	91.6	71.0	99.2	91.4	90.7	55.1	85.3	95.9	84.6	75.9	82.3	68.0	50.4	79.9	80.4	49.2	**38.6**	41.0	76.1
DMLoRA-LS (Ours)	0.29	-	74.0	90.7	73.9	99.3	**92.2**	**91.1**	**56.4**	85.6	96.5	87.0	76.1	**83.5**	**69.9**	52.0	81.6	80.2	50.2	36.1	43.1	**77.0**

4.2 Comparison Experiments on VTAB-1K

In this section, we compare our method with several previous state-of-the-art methods on VTAB, as shown in Tab. 1. We first compare with results from traditional fine-tuning methods. It's evident that fine-tuning all parameters in traditional methods outperforms solely fine-tuning the head. However, with parameter-efficient fine-tuning methods, the advantage of fine-tuning all parameters gradually diminishes, even surpassing it. Compared to previous Parameters-Efficient Fine-Tuning (PEFT) methods, our method demonstrates superior comprehensive performance. For the average on Natural photo datasets, we outperform previous methods by 0.89%. For the average on Structured photo datasets, our method surpasses previous approaches by 0.85%. Overall, our method achieves a higher average score than the best previous method by 0.9%.

4.3 Comparison Experiments on FGVC

In this section, we compare our method with several previous state-of-the-art methods on FGVC, as shown in Tab. 2. Similarly, we initially present results from traditional fine-tuning methods. It's noticeable that traditional fine-tuning methods exhibit better performance here compared to their results on VTAB-1K. Some parameter-efficient fine-tuning methods also outperform full-parameter fine-tuning. Our method demonstrates superior comprehensive performance compared to other methods. On the CUB-200-2011 dataset, we outperform the previous best method by 0.3%; on NABirds, our results surpass previous methods by 0.9%; for Stanford Dogs, our method outperforms previous methods by 0.6%. Overall, our method achieves the best average score as SSF [24]. Additionally,

Table 2: Performance comparisons on five FGVC datasets with ViT-B/16 models pre-trained on ImageNet-21K. "# params" specifies the number of trainable parameters in backbones. Average accuracy and # params are averaged over group-wise mean values. The bold data is **the best**, and the underlined data is the second best. Our method has the best average results.

Method	# Params (M)	CUB-200-2011	NABirds	Oxford Flowers	Stanford Dogs	Stanford Cars	Average
Full fine-tuning	85.8	87.3	82.7	98.8	89.4	84.5	88.5
Linear probing	0	85.3	75.9	97.9	86.2	51.3	79.3
Adapter [14]	0.41	87.1	84.3	98.5	89.8	68.6	85.7
Bitfit [41]	0.28	88.4	84.2	98.8	91.2	79.4	88.4
VPT-Shallow [16]	0.06	86.7	78.8	98.4	90.7	68.7	84.6
VPT-Deep [16]	0.53	88.5	84.2	99.0	90.2	83.6	89.1
E^2VPT [10]	0.25	89.1	84.6	99.1	90.5	82.8	89.2
SSF [24]	0.24	89.5	85.7	**99.6**	89.6	**89.2**	**90.7**
DMLoRA-LS (Ours)	0.29	**89.8**	**86.6**	99.5	**91.8**	85.7	**90.7**

for individual tasks, our method either secures the best or second-best results. The results on FGVC further indicate the improved performance of our method compared to previous approaches.

4.4 Analysis for Dropout-Mixture LoRA

In this section, we conduct ablation studies on the Dropout-Mixture Low-Rank Adaptation approach to independently investigate the efficacy of branch settings and pruning strategies. In pursuit of this goal, we formulated 5 experimental schemes, with different settings of whether the number of branches decreases during training, the pruning strategy for branches, and whether the up/down matrix is set to a single branch. The configurations of the 5 experimental schemes are presented in Tab. 3. (1) Scheme A retains multiple branches throughout the training process, without gradual decreasing of the training branches, is used to investigate the necessity of pruning branches during training in DMLoRA. (2) Scheme B adopts averaging strategy when pruning branches, instead of random dropout, testing another feasible pruning strategy. (3-4) Scheme C (or D) sets the up matrix (or down matrix) to a single branch to verify the role of the multi-branch structure. (5) Scheme E is our strategy with multiple branches and branch pruning via random dropout during training. As shown in Tab. 4, we observe the scheme A that maintains n branches throughout training (like AdaMix) performs worse than gradually pruning branches during training (ours), with a decline of 2.1% in performance. The schemes C or D that shares up or down matrix exhibit a decrease in performance, notably 1.1%. The scheme B that uses the average aggregating to prune branches during training performs worse compared to those with random dropout (ours), with a decline of 1.8%.

4.5 Analysis for 2-Stage Learning Scalar

In this section, we compare two learning strategies for the Scale parameter: (1) 1-Stage learning strategy: Simultaneous learning of parameters for Low-Rank

Table 3: Configurations of 5 schemes in ablation studies on DMLoRA. **Reduce Branch** indicates whether the number of branches decreases during the training process. **DB** (Dropout Branch) refers to pruning via random dropout, while **AB** (Average Branch) denotes pruning through parameter averaging. **SU** signifies Sharing Up, indicating that the up matrix is set to a single branch, whereas **SD** indicates Sharing Down, implying that the down matrix is set to a single branch.

Settings / Scheme	**Reduce Branch**	Branch Pruning Method		Sharing Layer	
		DB	**AB**	**SU**	**SD**
Scheme A	✗	✗	✗	✗	✗
Scheme B	✓	✗	✓	✗	✗
Scheme C	✓	✓	✗	✓	✗
Scheme D	✓	✓	✗	✗	✓
Scheme E (**Ours**)	✓	✓	✗	✗	✗

Table 4: The ablation experiments conducted on the Dropout-Mixture Low-Rank Adaptation approach on VTAB-1K reveal that the random dropout strategy yields superior performance.

Method \ Dataset	Natural	Specialized	Structured	Avg.
Full fine-tuning	75.9	83.4	47.6	68.9
Linear probing	69.1	77.1	26.9	57.6
Scheme A	80.9	85.4	57.3	74.5
Scheme B	82.1	83.4	58.9	74.8
Scheme C	80.6	85.6	59.5	75.2
Scheme D	81.1	85.8	59.5	75.5
Scheme E (**Ours**)	**82.3**	**86.2**	**61.4**	**76.6**

Adaptation and Scale; (2) 2-Stage learning strategy (ours): Sequential learning, where we first learn parameters for Low-Rank Adaptation, and then learn Scale parameters. The experimental results reported in Tab. 5 reveal that the 1-Stage learning strategy performs worse compared to the 2-Stage learning strategy. This suggests that Scale parameters significantly impact Low-Rank Adaptation. Employing the simultaneous learning strategy leads to interference between these two components, whereas our proposed 2-Stage learning approach provides a more effective solution.

4.6 Efficiency Analysis

With the ViT-B/16 model as base, our DMLoRA method is highly efficient during both training and testing phases. Parameter fine-tuning is achievable using a single NVIDIA RTX 4090, where the average training time on VTAB-1K does not exceed 20 minutes, excluding validation time. Despite introducing multiple branches for training, the increase of the GPU memory usage is negligible(less

Table 5: The ablation experiments conducted on the learning strategies for the Scale parameter on VTAB-1K reveal that the 2-Stage learning strategy for scale parameters is more effective.

Method \ Dataset	Natural	Specialized	Structured	Avg.
Full fine-tuning	75.9	83.4	47.6	68.9
Linear probing	69.1	77.1	26.9	57.6
1-Stage	82.0	83.4	60.7	75.4
2-Stage (**Ours**)	**82.5**	**86.3**	**62.1**	**77.0**

Table 6: Efficiency comparison of ours and some existing PETL methods during inference. We compare on ViT-B/16. ΔP is the increment of the model parameters in the Inference Stage. ΔF is the increment of the FLOPs in the Inference Stage.

Methods	ΔP	ΔF	GPU latency (b=128)
Full fine-tuning	0	0	1×
VPT-Deep [16]	0.53M	5.60G	1.37×
Adapter [14]	0.16M	0.03G	1.06×
Adaptformer [3]	0.16M	0.03G	1.04×
Ours	0	0	1×

than 1%). The slight increase in training duration stems from the additional branches requiring sufficient training to converge. During inference, our modules are seamlessly integrated into the original model, incurring no additional inference time overhead. Tab. 6 shows specific details regarding efficiency analysis.

Due to the random selection of one branch for training in a multi-branch setup, each branch has a relatively lower probability of being chosen in the beginning. Consequently, more epochs are required to adequately train each branch. However, the increase in epochs is not linear with the number of branches, as early-stage stability in parameter optimization leads to quicker convergence. Furthermore, as the number of branches decreases during training, the probability of selecting each branch rises, diminishing the need for excessive branches. Our method incurs a training time overhead of less than 1×, a reasonable trade-off considering the model's performance improvement.

5 Conclusion and Discussion

In this paper, we propose a robust parameter-efficient fine-tuning method with a dynamic training structure comprising two key components: (1) Dropout Mixture Low-Rank Adaptation (DMLoRA) and (2) a Two-Stage Learning Scalar strategy. Our model outperforms previous methods in performance while maintaining a reparameterization architecture, thus avoiding additional inference time overhead. However, we also observe certain performance disparities across different tasks. In future research, we aim to further enhance the overall performance of PEFT methods.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 62302297, 72192821, 62272447), Shanghai Sailing Program (22YF1420300), Young Elite Scientists Sponsorship Program by CAST (2022QNRC001), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), Shanghai Science and Technology Commission (21511101200), the Fundamental Research Funds for the Central Universities (YG2023QNB17, YG2024QNA44), Beijing Natural Science Foundation (L222117).

References

1. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020) 4
2. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9650–9660 (2021) 4
3. Chen, S., Ge, C., Tong, Z., Wang, J., Song, Y., Wang, J., Luo, P.: Adaptformer: Adapting vision transformers for scalable visual recognition. Advances in Neural Information Processing Systems **35**, 16664–16678 (2022) 4, 10, 11, 14
4. Ding, N., Qin, Y., Yang, G., Wei, F., Yang, Z., Su, Y., Hu, S., Chen, Y., Chan, C.M., Chen, W., et al.: Parameter-efficient fine-tuning of large-scale pre-trained language models. Nature Machine Intelligence **5**(3), 220–235 (2023) 4
5. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020) 4, 9
6. Dou, S., Zhou, E., Liu, Y., Gao, S., Zhao, J., Shen, W., Zhou, Y., Xi, Z., Wang, X., Fan, X., et al.: The art of balancing: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. arXiv preprint arXiv:2312.09979 (2023) 5
7. Edalati, A., Tahaei, M.S., Kobyzev, I., Nia, V.P., Clark, J.J., Rezagholizadeh, M.: Krona: Parameter efficient tuning with kronecker adapter. CoRR **abs/2212.10650** (2022). https://doi.org/10.48550/ARXIV.2212.10650, https://doi.org/10.48550/arXiv.2212.10650 5
8. Gebru, T., Krause, J., Wang, Y., Chen, D., Deng, J., Fei-Fei, L.: Fine-grained car detection for visual census estimation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 31 (2017) 10
9. Guo, D., Rush, A.M., Kim, Y.: Parameter-efficient transfer learning with diff pruning. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021. pp. 4884–4896. Association for Computational Linguistics (2021). https://doi.org/10.18653/V1/2021.ACL-LONG.378, https://doi.org/10.18653/v1/2021.acl-long.378 5

10. Han, C., Wang, Q., Cui, Y., Cao, Z., Wang, W., Qi, S., Liu, D.: E 2 vpt: An effective and efficient approach for visual prompt tuning. In: 2023 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 17445–17456. IEEE (2023) 10, 11, 12
11. He, J., Zhou, C., Ma, X., Berg-Kirkpatrick, T., Neubig, G.: Towards a unified view of parameter-efficient transfer learning. In: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022. OpenReview.net (2022), https://openreview.net/forum?id=0RDcd5Axok 4, 5
12. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16000–16009 (2022) 4
13. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9729–9738 (2020) 4
14. Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., Gelly, S.: Parameter-efficient transfer learning for nlp. In: International Conference on Machine Learning. pp. 2790–2799. PMLR (2019) 2, 4, 10, 11, 12, 14
15. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W.: Lora: Low-rank adaptation of large language models. In: The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022. OpenReview.net (2022), https://openreview.net/forum?id=nZeVKeeFYf9 5, 7, 11
16. Jia, M., Tang, L., Chen, B., Cardie, C., Belongie, S.J., Hariharan, B., Lim, S.: Visual prompt tuning. In: Avidan, S., Brostow, G.J., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXIII. Lecture Notes in Computer Science, vol. 13693, pp. 709–727. Springer (2022). https://doi.org/10.1007/978-3-031-19827-4_41, https://doi.org/10.1007/978-3-031-19827-4_41 4, 5, 10, 11, 12, 14
17. Jie, S., Deng, Z.H.: Convolutional bypasses are better vision transformer adapters. arXiv preprint arXiv:2207.07039 (2022) 4
18. Jie, S., Deng, Z.H.: Fact: Factor-tuning for lightweight adaptation on vision transformer. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 1060–1068 (2023) 10, 11
19. Khosla, A., Jayadevaprakash, N., Yao, B., Li, F.F.: Novel dataset for fine-grained image categorization: Stanford dogs. In: Proc. CVPR workshop on fine-grained visual categorization (FGVC). vol. 2. Citeseer (2011) 10
20. Lester, B., Al-Rfou, R., Constant, N.: The power of scale for parameter-efficient prompt tuning. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021. pp. 3045–3059. Association for Computational Linguistics (2021). https://doi.org/10.18653/V1/2021.EMNLP-MAIN.243, https://doi.org/10.18653/v1/2021.emnlp-main.243 4, 5
21. Li, J., Li, D., Xiong, C., Hoi, S.: Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In: International Conference on Machine Learning. pp. 12888–12900. PMLR (2022) 4
22. Li, X.L., Liang, P.: Prefix-tuning: Optimizing continuous prompts for generation. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual

Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021. pp. 4582–4597. Association for Computational Linguistics (2021). https://doi.org/10.18653/V1/2021.ACL-LONG.353, https://doi.org/10.18653/v1/2021.acl-long.353 4, 5
23. Lialin, V., Deshpande, V., Rumshisky, A.: Scaling down to scale up: A guide to parameter-efficient fine-tuning. arXiv preprint arXiv:2303.15647 (2023) 4
24. Lian, D., Zhou, D., Feng, J., Wang, X.: Scaling & shifting your features: A new baseline for efficient model tuning. In: NeurIPS (2022), http://papers.nips.cc/paper_files/paper/2022/hash/00bb4e415ef117f2dee2fc3b778d806d-Abstract-Conference.html 5, 10, 11, 12
25. Lin, W., Wu, Z., Chen, J., Yang, W., Huang, M., Huang, J., Jin, L.: Hierarchical side-tuning for vision transformers. arXiv preprint arXiv:2310.05393 (2023) 4
26. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 10012–10022 (2021) 4
27. Luo, G., Huang, M., Zhou, Y., Sun, X., Jiang, G., Wang, Z., Ji, R.: Towards efficient visual adaption via structural re-parameterization. CoRR **abs/2302.08106** (2023). https://doi.org/10.48550/ARXIV.2302.08106, https://doi.org/10.48550/arXiv.2302.08106 5, 10, 11
28. Mao, Y., Mathias, L., Hou, R., Almahairi, A., Ma, H., Han, J., Yih, S., Khabsa, M.: Unipelt: A unified framework for parameter-efficient language model tuning. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022. pp. 6253–6264. Association for Computational Linguistics (2022). https://doi.org/10.18653/V1/2022.ACL-LONG.433, https://doi.org/10.18653/v1/2022.acl-long.433 5
29. Mercea, O.B., Gritsenko, A., Schmid, C., Arnab, A.: Time-memory-and parameter-efficient visual adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5536–5545 (2024) 5
30. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: 2008 Sixth Indian conference on computer vision, graphics & image processing. pp. 722–729. IEEE (2008) 10
31. OpenAI: GPT-4 technical report. CoRR **abs/2303.08774** (2023). https://doi.org/10.48550/ARXIV.2303.08774, https://doi.org/10.48550/arXiv.2303.08774 4
32. Pfeiffer, J., Kamath, A., Rücklé, A., Cho, K., Gurevych, I.: 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. pp. 487–503 (2021) 4
33. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021) 4
34. Rücklé, A., Geigle, G., Glockner, M., Beck, T., Pfeiffer, J., Reimers, N., Gurevych, I.: Adapterdrop: On the efficiency of adapters in transformers. arXiv preprint arXiv:2010.11918 (2020) 4
35. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A.,

Grave, E., Lample, G.: Llama: Open and efficient foundation language models. CoRR **abs/2302.13971** (2023). https://doi.org/10.48550/ARXIV.2302.13971, https://doi.org/10.48550/arXiv.2302.13971 4

36. Van Horn, G., Branson, S., Farrell, R., Haber, S., Barry, J., Ipeirotis, P., Perona, P., Belongie, S.: Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 595–604 (2015) 10
37. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017) 4
38. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011) 10
39. Wang, W., Xie, E., Li, X., Fan, D.P., Song, K., Liang, D., Lu, T., Luo, P., Shao, L.: Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 568–578 (2021) 4
40. Wang, Y., Mukherjee, S., Liu, X., Gao, J., Awadallah, A.H., Gao, J.: Adamix: Mixture-of-adapter for parameter-efficient tuning of large language models 4, 6
41. Zaken, E.B., Ravfogel, S., Goldberg, Y.: Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. arXiv preprint arXiv:2106.10199 (2021) 5, 10, 11, 12
42. Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A.S., Neumann, M., Dosovitskiy, A., et al.: A large-scale study of representation learning with the visual task adaptation benchmark. arXiv preprint arXiv:1910.04867 (2019) 10
43. Zhang, Y., Zhou, K., Liu, Z.: Neural prompt search (2022) 10, 11