Gradient-based Parameter Selection for Efficient Fine-Tuning

Zhi Zhang[1,2*], Qizhe Zhang[2*], Zijun Gao[2], Renrui Zhang[3], Ekaterina Shutova[1], Shiji Zhou[4], Shanghang Zhang[2†]
[1]ILLC, University of Amsterdam,
[2]National Key Laboratory for Multimedia Information Processing,
School of Computer Science, Peking University,
[3]MMLAB, The Chinese University of Hong Kong,
[4]Department of Automation, Tsinghua University,
{z.zhang, e.shutova}@uva.nl, {theia, shanghang}@pku.edu.cn

Abstract

*With the growing size of pre-trained models, full fine-tuning and storing all the parameters for various downstream tasks is costly and infeasible. In this paper, we propose a new parameter-efficient fine-tuning method, **G**radient-based **P**arameter **S**election (GPS), demonstrating that only tuning a few selected parameters from the pre-trained model while keeping the remainder of the model frozen can generate similar or better performance compared with the full model fine-tuning method. Different from the existing popular and state-of-the-art parameter-efficient fine-tuning approaches, our method does not introduce any additional parameters and computational costs during both the training and inference stages. Another advantage is the model-agnostic and non-destructive property, which eliminates the need for any other design specific to a particular model. Compared with the full fine-tuning, GPS achieves 3.33% (91.78% vs. 88.45%, FGVC) and 9.61% (73.1% vs. 65.57%, VTAB) improvement of the accuracy with tuning only 0.36% parameters of the pre-trained model on average over 24 image classification tasks; it also demonstrates a significant improvement of 17% and 16.8% in mDice and mIoU, respectively, on medical image segmentation task. Moreover, GPS achieves state-of-the-art performance compared with existing PEFT methods.*

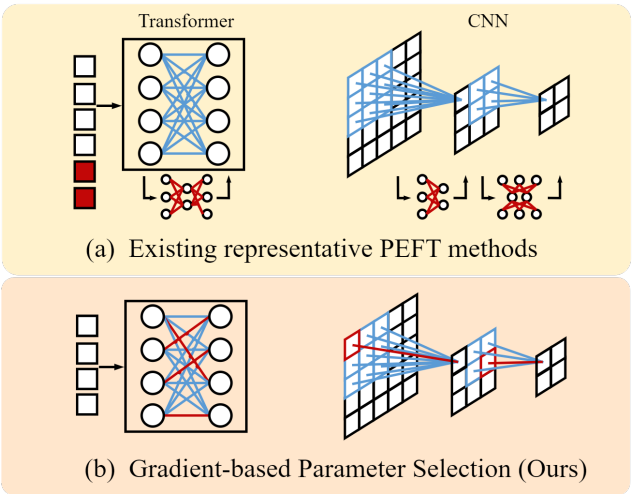
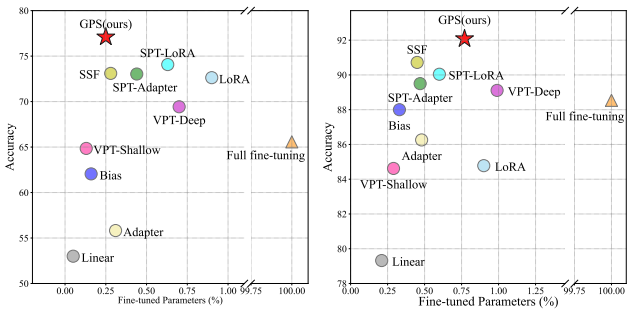


Figure 1. Comparison between our GPS and other PEFT methods. (a) Exiting popular methods introduce extra parameters for tuning downstream tasks, which might need a special design for diverse architectures, such as appending prompt into the input token in Transformer or inserting different modules into different layers (b) Our approach avoids the introduction of additional parameters and solely fine-tunes the selected parameters from the model, employing a unified gradient-based parameter selection method across diverse architectural variations, e.g. Transformer and CNN.

1. Introduction

The pre-training and fine-tuning pipeline has become a common paradigm for adapting large models pre-trained on substantial amounts of data to downstream tasks with fewer training samples. However, fine-tuning all the parameters in the model is memory-intensive and data-inefficient, which is costly and infeasible for multiple downstream tasks given a large-scale model [34, 41, 56]. To tackle this issue, parameter-efficient fine-tuning (PEFT) methods have been proposed with the aim of tuning a minimal number of parameters to fit downstream tasks while keeping most of the parameters frozen. In addition to mitigating memory issues, another benefit of PEFT is that tuning a smaller set of parameters reduces the complexity of optimization and alleviates overfitting concerns when adapting large pre-trained models to downstream tasks with limited labeled data, re-

*Contributed equally.
†Corresponding author.



(a) VTAB-1K (b) FGVC

Figure 2. Performance comparisons of 11 fine-tuning methods with a pre-trained ViT-B/16 model on the VTAB-1k (a) and FGVC (b) benchmarks. Our GPS (red stars) achieves state-of-the-art performance on both benchmarks with only 0.25% and 0.77% average trainable parameters respectively.

sulting in comparable or even superior performance compared to full fine-tuning [41]. Inspired by the success of PEFT in NLP [13, 29, 36, 38, 55, 77, 82], several analogous approaches have been introduced to vision tasks, such as Adapter [34] and Visual Prompt Tuning (VPT) [41], which introduce extra learnable parameters into the backbone and the input space of the pre-trained model, respectively [14]. In addition, SSF, another representative method, transforms the features across layers of the pre-trained model using extra learnable layers [56].

However, these methods introduce additional parameters into the pre-trained model and disrupt its original architecture, leading to increased computational costs during training and/or inference stages. Furthermore, these approaches lack generalizability across various model architectures. Specifically, different models are equipped with distinct components (layers), such as MLPs, activation functions, and self-attention layers. These methods need to determine the optimal locations for inserting extra parameters between different layers; moreover, certain transformer-based techniques cannot be directly applied to convolution-based methods like VPT. Therefore, these methods exhibit limited compatibility with diverse architectures.

To tackle the aforementioned issues, we propose a non-destructive network architecture and model-agnostic PEFT approach, which introduces no extra parameters during both training and test stages and provides a unified solution for various architectures. We select a small number of essential parameters from the pre-trained model and only fine-tune these parameters for the downstream tasks. To select these essential parameters, we propose a fine-grained Gradient-based Parameter Selection (GPS) method. For each neuron in the network, we choose top-K of its input connections (weights or parameters) with the highest gradient value, resulting in a small proportion of the original model's parameters being selected.

Method	Mean Acc.	Params. (%)	Model Agnostic	No extra Train param.	No extra Infer params.	Task Adaptive
Full [41]	70.36	100	✓	✓	✓	✗
Linear [41]	58.48	0.08	✓	✓	✓	✗
Bias [91]	67.54	0.20	✓	✓	✓	✗
Adapter [34]	60.04	0.35	✗	✗	✗	✗
VPT [41]	73.53	0.76	✗	✗	✗	✗
LoRA [36]	75.16	0.90	✗	✗	✓	✗
SSF [56]	76.77	0.32	✗	✗	✓	✗
GPS (ours)	78.64	0.36	✓	✓	✓	✓

Table 1. Comparison between different fine-tuning methods. The ViT-B/16 model accuracy over all 24 tasks in FGVA and VTAB fine-tuned on ViT-B/16 model and the number of tunable parameters are shown in columns Acc. and Params. (%).

Such design offers five-fold benefits: i) The pre-trained model can efficiently tackle downstream tasks because the gradient direction indicates the fastest loss function changes and highest change rate, facilitating efficient gradient descent during model fine-tuning. We also provide a sparse regularized equivalent form for GPS, which indicates better generalization than full fine-tuning; ii) Each neuron within the network possesses the potential to adjust its activation state by fine-tuning selected input connections. Consequently, the pre-trained model exhibits flexibility in modifying features of varying granularities to suit diverse downstream tasks. For instance, when adapting a model pre-trained on ImageNet [10] for CIFAR-100 [73], it is necessary to refine high-level features; whereas for ImageNet-Sketch [84] adaptation, more detailed feature fine-tuning is required. iii) Our approach avoids introducing extra parameters and computational costs and keeps the architecture of the model intact; iv) The selection procedure enables its application across diverse models by adopting a neuron-based rather than a layer-based method, thereby eliminating the necessity for distinct designs for different layers in various models. v) Different from other methods using a pre-defined and consistent strategy for different tasks, our method adaptively selects parameters for each task by our proposed gradient strategy to better fit the domain-specific semantics of different downstream tasks. Please see the difference between our method with others in Tab. 1.

We evaluate our approach on a total of 27 visual tasks (including image classification and semantic segmentation) over 4 different model architectures. Our GPS achieves state-of-the-art performance compared to other PEFT methods and has a good balance between performance and the number of trainable parameters, as illustrated in Fig. 2. Compared with the full fine-tuning, GPS achieves 3.33% (FGVC) and 9.61% (VTAB) improvement of the accuracy while tuning only 0.36% parameters of the pre-trained

model on average over 24 tasks; it also demonstrates a significant improvement of 17% and 16.8% in mDice and mIoU, respectively, on medical image segmentation task. Moreover, we verify the effectiveness of our approach on different network architectures, such as Transformer and Convolutional Neural Networks. Furthermore, we compare GPS with various parameter selection methods and demonstrate its superior properties. GPS provides a new paradigm for PEFT and inspires deeper insights into this field.

2. Related work

Visual parameter efficient fine-tuning In general, there are typically two primary categories of PEFT. Addition-based methods introduce additional parameters to the pre-trained backbone. Adapters [1, 34, 67, 68, 73, 74, 76, 78, 85, 93] adopt a residual pathway and learn down and up projection with a nonlinear activation. Others [61] propose a hyper-network to generate model weights or decompose the dense weighted matrix into the low-rank matrix [44]. Prompt methods [12, 20, 37, 43, 55, 57, 58] wrap the input with context. VPT [41] prepend learnable prompts to the input tokens. SSF [56] achieves promising results by scaling and shifting the feature between layers. Selection-based methods select a subset of the parameters for tuning, such as only fine-tuning bias [91], last K layers [34, 41]. While traditionally considered less effective than addition-based methods, our approach of adaptively selecting parameters for each task yielded surprisingly strong results.

Sub-network training Pruning technique [19, 26, 27, 50, 54, 87] uncovers the importance of subnetworks. The lottery ticket hypothesis [17] articulates that subnetworks can reach the accuracy of the original model. Fine-tuning subnetworks are widely studied. SpotTune [25] designs a policy network to make routing decisions for subset networks. Child-tuning [89] iteratively updates a subset of parameters by masking out some gradients during the backward process. However, these methods are not aligned with the PEFT setting. In this paper, we fix a small number of parameters and only tune them for fitting downstream to achieve PEFT.

3. Approach

3.1. Overview

Different from the currently popular methods introducing additional parameters to fine-tune the pre-trained model for downstream tasks [34, 41, 56], we select only a small number of parameters from the pre-trained model and then only update these parameters during the fine-tuning stage. Specifically, our method has two stages: parameter selection and masked fine-tuning. For each downstream task t, we first select a small portion of important parameters (task-specific parameters) from the original pre-trained model using a gradient-based method. We then fine-tune the pre-trained model for the task t, keeping all other unimportant parameters frozen and updating only selected parameters using a sparse binary mask to set the gradient of unimportant parameters to zero (see Fig. 3).

3.2. Gradient-based parameter selection

Relevant studies have indicated that the pre-trained backbone exhibits diverse feature patterns at distinct parameter positions, and the same positions make varying contributions to fine-tuning various tasks [4, 51, 63, 72, 90]. Therefore, we posit that there exists an optimal subset of parameters for fine-tuning a pre-trained model to a downstream task. This subset is essential and necessary for fine-tuning the task, and the different tasks require a distinct subset. Formally, given a downstream task t with the dataset $\mathcal{D}_t$ and a pre-trained model $\Theta = \{w_1, w_2, \ldots, w_N\}$, we aim to find a subset of $\boldsymbol{w}$, i.e. $\boldsymbol{w} = \{w_1, w_2, \ldots, w_n\}$ ($n \ll N$). we select parameters following two principles: 1) Important for downstream tasks; 2) Distributed over the whole network.

Importance for downstream tasks We identify the importance of parameters in a pre-trained model for a specific task by selecting those with the highest gradient value, which is obtained by calculating the gradient of a loss function with respect to its parameters. The intuition behind this is that the parameters with the largest gradient value indicate the loss function changes fastest along the gradient direction and has the greatest change rate, which facilitates efficient gradient descent during fine-tuning. Specifically, the gradient of the parameters is calculated by

$$\nabla \mathcal{L}_{\mathcal{D}_t}(\Theta) = \left[\frac{\partial \mathcal{L}}{\partial w_1} \cdots \frac{\partial \mathcal{L}}{\partial w_N} \right]^\top \tag{1}$$

where $\mathcal{L}(\boldsymbol{w})$ is the loss function. Normally, when we fine-tune a pre-trained model on a downstream task, we need a new classification head ($i.e.$ MLP) with random initialization. In order to avoid the adverse effects of these randomly initialized parameters on gradient calculation using the cross-entropy loss function, we use Supervised Contrastive Loss (SCL) [46] as the loss function for calculating the gradient during parameter selection, since it does not need to involve the head (We still use cross-entropy loss during fine-tuning stage). SCL is a variant of Contrastive Loss (CL) that aims to bring different augmented samples of the same image closer together in embedding space. In contrast, SCL tries to cluster samples from the same class together, which coincides with our target of the downstream classification tasks. Specifically, given a task

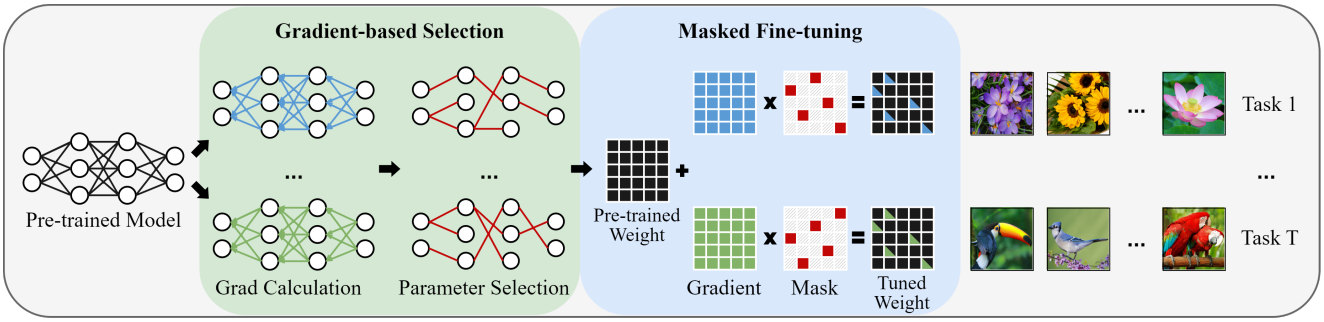


Figure 3. The overall pipeline of GPS. We first select a small portion of important parameters (sub-network) for each task from the original pre-trained model using a gradient-based method. Then only fine-tune the sub-network while keeping other parameters frozen.

with the dataset $\mathcal{D}_t = \{\boldsymbol{x}_i, y_i\}_{i=1...K}$, SCL is calculated by

$$\begin{aligned}\mathcal{L}^{\text{scl}} &= \sum_{i\in\mathcal{D}_t} \mathcal{L}_i^{\text{scl}} \\ &= \sum_{i\in\mathcal{D}_t} \frac{-1}{|P(i)|} \sum_{p\in P(i)} \log \frac{\exp\left(\boldsymbol{z}_i \odot \boldsymbol{z}_p/\tau\right)}{\sum_{a\in A(i)} \exp\left(\boldsymbol{z}_i \odot \boldsymbol{z}_a/\tau\right)}\end{aligned} \tag{2}$$

where i represents i^{th} sample in $\mathcal{D}_t$; $P(i) \equiv \{p \in A(i) : \tilde{y}_p = \tilde{y}_i\}$ is subset of $\mathcal{D}_t$, in which all samples have the same class with i; $A(i) \equiv \mathcal{D}_t \backslash \{i\}$; $\boldsymbol{z}$ is the feature extacted from the pre-trained encoder and $\tau \in \mathcal{R}^+$ is a scalar temperature parameter.

Equivalent with sparse regularization In the above, we implicitly assume that the order of $\left(\left\|\frac{\partial \mathcal{L}}{\partial w_1}\right\| \cdots \left\|\frac{\partial \mathcal{L}}{\partial w_N}\right\|\right)$ is the same as $(\|w_1' - w_1\| \cdots \|w_N' - w_N\|)$, which means selecting parameters with top-n gradient norm is the same as selecting top-n of the fine-tuning changes. Therefore, GPS captures the top-n important parameters for downstream tasks. The optimization objective can be rewritten as

$$\Theta' = \min \mathcal{L}(\Theta') \qquad s.t. \quad \|\Theta' - \Theta\|_0 \leq n \tag{3}$$

where $\|\|_0$ is the l_0 norm and Θ' is the fine-tuned model. By Lagrangian duality, solving the above problem is equivalent to solving the following problem:

$$\mathcal{L}(\Theta') + \lambda \|\Theta' - \Theta\|_0 \tag{4}$$

with appropriate λ. Hence, GPS can be reviewed as a sparse regularized fine-tuning, which may lead to better generalization. Fu et al. [18] demonstrate that Eq. (4) has smaller generalization bound than pure optimization toward $\mathcal{L}$ with full fine-tuning, resulting in better performance.

Distribution over the whole network A simple idea for parameter selection is to select a certain percentage of parameters with the highest gradient from the entire network. Our experiments have shown that with this idea, the majority of the selected parameters are located in the top layers of the network (see Supplementary for details), which is consistent with the findings reported in [34, 35]. However, solely fine-tuning these top-layer parameters is insufficient to mitigate the impact of the pre-trained model's own inductive bias, particularly when there exist substantial disparities in data distributions between upstream and downstream tasks, which need to fine-tune more detailed features from shallower layers. Motivated by various studies indicating the distinct roles played by different components of neural networks [3, 15, 70, 75, 86], we posit that when fine-tuning a pre-trained model for downstream tasks, the adjusted parameters should be distributed throughout the entire network. The reason behind this lies in the ability of the model to adapt the information stored in parameters at different levels of granularity to fit downstream tasks. Therefore, our strategy is that for each neuron in the network, we select top-K (at least one) connections (weights) among all the input connections of the neuron, as shown in Fig. 3. By doing so, every neuron within the network possesses the potential to fine-tune its activation state rather than solely adjust high-level information in the top layers. In other words, our approach fine-tunes the detailed information stored in each neuron of the network, which better fits the downstream task during the fine-tuning stage. Our exploratory experiment further substantiates this assertion, as shown in Tab. 6(a).

Combining the two points above, we first calculate the gradient of the loss with respect to all the weights in the models for a specific task. Then for each neuron in the network, we select top-K connections with the highest gradient value (the modulus of gradient) among all input connections to the neuron. Doing so can not only ensure that important parameters for downstream tasks are chosen and allow the model to tune the activation state of all neurons for better fitting of downstream tasks. Another benefit of this selection procedure is its ease of application across various model architectures, such as Transformer and CNN, avoiding any model-specific design. Our experiments also demonstrate

the effectiveness of our approach across diverse architectures, as shown in Tab. 2 and Tab. 4.

3.3. Masked fine-tuning

After parameter selection for a specific task, we fine-tune the pre-trained model on the task. During fine-tuning, we only update the selected parameters while keeping the remaining parameters of the pre-trained model frozen. As our selected parameters are distributed across all neurons in every layer, only a few parameters within a specific weight matrix of the network are chosen, resulting in the updated matrix being sparse. Therefore, we utilize a mask to help with the sparse training. Specifically, for j^{th} weight matrix $\boldsymbol{W}_j \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$ in the network, we build a same size of binary mask $\boldsymbol{M}_j \in \mathbb{R}^{d_{\text{in}} \times d_{\text{out}}}$:

$$\boldsymbol{M}_j = \begin{cases} 1, & w_j^k \in \boldsymbol{w} \\ 0, & w_j^k \notin \boldsymbol{w} \end{cases} \tag{5}$$

where w_j^k represents k^{th} element in j^{th} weight matrix. For each element in $\boldsymbol{M}_j$, its value is set to 1 if the corresponding parameter in $\boldsymbol{W}_j$ is selected, and 0 otherwise. Then the weight matrix is updated by

$$\boldsymbol{W}_j \leftarrow \boldsymbol{W}_j - \epsilon \nabla \mathcal{L}(\boldsymbol{W}_j) \odot \boldsymbol{M}_j \tag{6}$$

where $\nabla \mathcal{L}(\boldsymbol{W}_j)$ is the gradient of the cross-entropy loss with respect to $\boldsymbol{W}_j$. As a result, the gradients of unselected parameters are zeroed out and excluded from updates, while only a small number of our selected parameters are updated during fine-tuning for downstream tasks. Please see Fig. 3 for a visualization of our method.

4. Experiments

We evaluate GPS on various downstream tasks, including image classification tasks and semantic segmentation tasks with different architectures. First, we briefly introduce our experimental settings, including datasets, backbones, and baselines. Then we demonstrate the effectiveness and universality of GPS. Moreover, we systematically study the impacts of different selection schemes and conduct comprehensive ablation experiments.

4.1. Experimental settings

Datasets Following VPT [41] and SSF [56], we evaluate our GPS method on a series of datasets categorized into three groups: i) ***FGVC***: Fine-Grained Visual Classification (FGVC) benchmark includes 5 downstream tasks, which are CUB-200-2011 [83], NABirds [79], Oxford Flowers [65], Stanford Dogs [45] and Stanford Cars [21]. ii) ***VTAB-1k***: Visual Task Adaptation Benchmark [92] (VTAB) contains 19 visual classification tasks which are grouped into three sets: Natural, Specialized, and Structured. iii) ***CIFAR-100*** [49] and ***ImageNet-1k*** [10]: widely use for general image classification task.

Dataset	CUB -2011	NA-Brids	Oxford Flowers	Stan. Dogs	Stan. Cars	Mean Acc.	Params. (%)
Full [41]	87.3	82.7	98.8	89.4	84.5	88.54	100.00
Linear [41]	85.3	75.9	97.9	86.2	51.3	79.32	0.21
Bias [91]	88.4	84.2	98.8	91.2	79.4	88.40	0.33
Adapter [34]	87.1	84.3	98.5	89.8	68.6	85.66	0.48
LoRA [36]	85.6	79.8	98.9	87.6	72.0	84.78	0.90
VPT-Shallow [41]	86.7	78.8	98.4	90.7	68.7	84.62	0.29
VPT-Deep [41]	88.5	84.2	99.0	90.2	83.6	89.11	0.99
SSF [56]	89.5	85.7	99.6	89.6	89.2	90.72	0.45
SPT-Adapter [28]	89.1	83.3	99.2	91.1	86.2	89.78	0.47
SPT-LoRA [28]	88.6	83.4	99.5	91.4	87.3	90.04	0.60
GPS (Ours)	**89.9**	**86.7**	**99.7**	**92.2**	**90.4**	**91.78**	0.77

Table 2. Performance comparisons on FGVC with ViT-B/16 models pre-trained on ImageNet-21K.

Backbones For a fair comparison, we follow VPT and SSF by using ViT-B/16 [14] pre-trained on ImageNet-21K [10] for the main image classification experiments. Moreover, to demonstrate the universality of our GPS, we also explore other backbones, including Swin Transformer (Swin-B) [59] and ConvNeXt-B [60] for another variant of Transformer-based and CNN-based architecture, respectively. In addition, we conduct experiments on semantic segmentation tasks using SAM [48], a strong segmentation foundation model.

Baselines We compare our GPS with a variety of fine-tuning protocols that can be mainly categorized into three types: i) ***Full***: Full fine-tuning is the most commonly used protocol updating all parameters of the whole model during tuning. ii) ***Selection-based***: This kind of method selects a subset of parameters in the original model for fine-tuning, including linear probing and Bias [91]. Such methods are easy to implement and require no extra computations but have not worked well. Our method belongs to this group and achieves the best performance while ensuring convenience and universality. iii) ***Addition-based***: This kind of method adds new trainable parameters to the backbone, including Adapter [34], VPT [41] and SPT-Adapter [28]. Such methods require extra computations in both the training and inference stages. Other methods like LoRA [36], SSF [56], and SPT-LoRA [28] also add new tunable parameters during the training stage, but these parameters can be reparameterized into the backbone during testing.

Implementation details We follow SSF to process the images in all the FGVC, VTAB-1k and CIFAR-100 datasets. We employ the Adam [47] optimizer with cosine learning rate decay to fine-tune models for 100 epochs, and the linear warm-up is used in the first 10 epochs. All experiments are conducted on the NVIDIA A100 GPU.

Method \ Dataset	Natural							Specialized				Structured								VTAB	
	CIFAR-100	Caltech101	DTD	Flowers102	Pets	SVHN	Sun397	Patch Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr/count	Clevr/distance	DMLab	KITTI/distance	dSprites/loc	dSprites/ori	SmallNORB/azi	SmallNORB/ele	Mean Acc.	Mean Params. (%)
Full [41]	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	65.57	100.00
Linear [41]	63.4	85.0	64.3	97.0	86.3	36.6	51.0	78.5	87.5	68.6	74.0	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	53.00	0.05
Bias [91]	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	62.05	0.16
Adapter [34]	74.1	86.1	63.2	97.7	87.0	34.6	50.8	76.3	88.0	73.1	70.5	45.7	37.4	31.2	53.2	30.3	25.4	13.8	22.1	55.82	0.31
LoRA [36]	68.1	91.4	69.8	99.0	90.5	86.4	53.1	85.1	95.8	84.7	74.2	83.0	66.9	50.4	81.4	80.2	46.6	32.2	41.1	72.63	0.90
VPT-Shallow [41]	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	64.85	0.13
VPT-Deep [41]	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	69.43	0.70
SSF [56]	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	73.10	0.28
SPT-ADAPTER [28]	72.9	93.2	72.5	99.3	91.4	88.8	**55.8**	86.2	96.1	85.5	75.5	83.0	**68.0**	51.9	81.2	51.9	31.7	**41.2**	**61.4**	73.03	0.44
SPT-LoRA [28]	73.5	93.3	72.5	99.3	91.5	87.9	55.5	85.7	**96.2**	85.9	75.9	**84.4**	67.6	52.5	82.0	81.0	51.1	30.2	41.3	74.07	0.63
GPS (Ours)	**81.1**	**94.2**	**75.8**	**99.4**	91.7	**91.6**	52.4	**87.9**	**96.2**	86.5	**76.5**	79.9	62.6	**55.0**	**82.4**	**84.0**	**55.4**	29.7	46.1	**75.18**	0.25

Table 3. Performance comparisons on VTAB-1k with ViT-B/16 models pre-trained on ImageNet-21K.

4.2. Performance on image classification

We present a comprehensive evaluation of the effectiveness of our GPS by comparing it against multiple baselines on 3 benchmarks, comprising a total of 26 datasets. In addition to common benchmarks (FGVC and VTAB-1k), we also compare our method with others on different architectures. We evaluate the performance and effectiveness by Top-1 accuracy (%) and the number of fine-tuned parameters.

Image classification performance As shown in Tab. 2 and Tab. 3, our GPS outperforms all other fine-tuning methods by a large margin on both FGVC and VTAB benchmarks, sufficiently demonstrating that our method of parameter selection is a simple yet effective way for model tuning. On FGVC, GPS outperforms all other fine-tuning methods, including full fine-tuning, on all 5 tasks. It obtains 1.02% and 3.24% accuracy improvement of the mean accuracy compared to the previous SOAT method SSF [56] and full fine-tuning, while it only uses 0.77% of trainable parameters. On VTAB, GPS also outperforms all other fine-tuning methods. Specifically, it obtains 1.11% and 9.61% improvement of the mean accuracy on 19 VTAB tasks compared to the previous SOAT method SPT-LoRA [28] and full fine-tuning. GPS beats the previous SOTA by 1.75%, 0.23%, and 0.63% in the Natural, Specialized and Structured subsets, respectively. Meanwhile, GPS also uses fewer trainable parameters compared to VPT-Deep, SSF, and SPT-LoRA (0.25% vs. 0.70%, 0.28% and 0.63%), which further illustrates the high efficiency of our approach. For most tasks, we exclusively select the top 1 input connection for each neuron; however, for more challenging tasks, multiple connections are chosen (see Supplement for details). The percentage of learnable parameters in our GPS can be explicitly controlled by adjusting the number of connections selected, allowing for a balance between parameter count and performance in tasks.

Architecture	Swin-B		ConvNeXt-B	
	Ave. Acc.	Params.(%)	Ave. Acc.	Params.(%)
Full [41]	92.42	100.00	93.04	100.00
Linear [41]	87.90	0.28	88.00	0.28
SSF [56]	91.54	0.56	92.48	0.56
GPS (Ours)	**92.56**	0.95	**93.32**	0.90

Table 4. Performance comparisons on FGVC benchmark (Average accuracy over 5 tasks) with different model architectures.

Generalization on different architectures Since our method only selects a subset of parameters from the pre-trained model for fine-tuning, it is naturally model-agnostic. We compare GPS with other representative methods across ViT-B/16 (Tab. 2), Swin-B and ConvNeXt-B architectures on the FGVC dataset (Tab. 4), CIFAR-100 and ImageNet-1k (Please see full results in Supplementary). Among all three model architectures, GPS consistently outperforms all other baselines, demonstrating its model-agnostic advantage. The Swin and Convnext have more complex designs than ViT, enabling them to acquire comprehensive and high-quality features during pre-training. Consequently, even the simplest linear probing method yields commendable results on these two architectures, reducing the effectiveness of the PEFT method and causing the previous SOTA SSF method to underperform full fine-tuning. However, in this scenario, our GPS still maintains a lead over full fine-tuning with gains of 0.12% and 0.28%, respectively, further demonstrating the effectiveness of our design and the ability to explore the potential of model backbones.

Computational cost In Fig. 4, we compare the computational cost of GPS with other fine-tuning methods to demonstrate the efficiency of our approach. Following SSF [56],

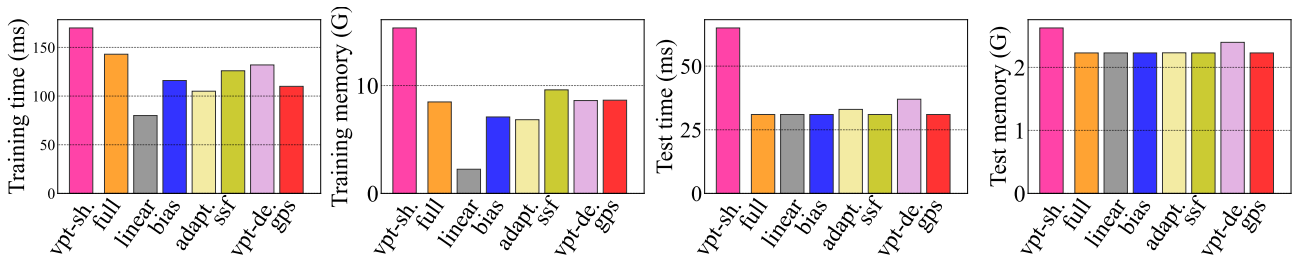


Figure 4. Computational cost of different tuning methods. From left to right: training time, training memory, test time, and test memory. Training/Test time is the time consumed by a mini-batch.

Method	mDice (↑)	mIoU (↑)	Params. (M)
Full [41]	71.1	55.7	93.8
Linear [41]	71.6	46.6	4.06
Bias [91]	86.5	69.1	4.16
Adapter [6]	84.8	66.7	4.12
SSF [56]	87.3	71.7	4.26
GPS (Ours)	**88.1**	**72.5**	4.22

Table 5. Quantitative Result for Polyp Segmentation

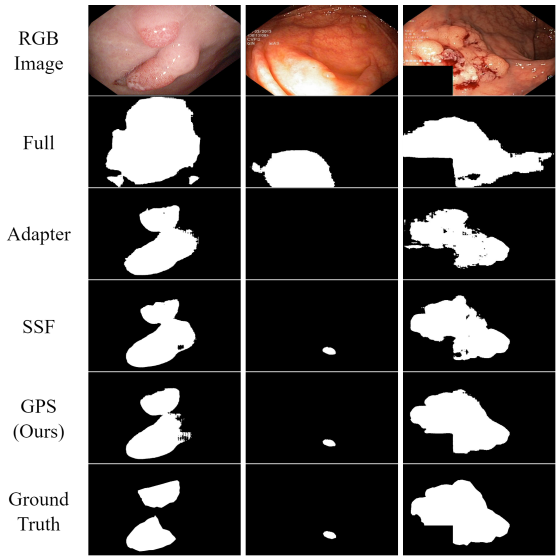


Figure 5. The Visualization of Polyp segmentation task. Our GPS can still handle difficult segmentation cases compared with others.

we reimplement VPT [41] with 200 and 50 prompts for the shallow and deep versions, respectively. A batch size of 32 is used in both the training and inference stages. For a fair comparison, for all experiments, we do not use mixed precision training, which was used in SSF. All metrics are measured on a single NVIDIA A100 GPU. In the training stage, GPS has less time and memory consumption than both VPT and SSF. Compared with full fine-tuning, GPS has a much lower time overhead and a similar memory overhead, but it leads to an increased performance by a large margin. Since GPS is a selection-based method, it does not introduce any additional parameters, so it can achieve the same minimum time and memory overhead as full fine-tuning during inference without any reparameterization operation, which is much lower than the addition-based Adapter and VPT.

4.3. Semantic segmentation

In addition to visual classification tasks, we also explore our method for the task of semantic segmentation. Segment Anything Model (SAM) [48] is a strong foundation model for segmentation. It is pre-trained on a large-scale dataset enabling powerful generalization. However, several studies, e.g. [6], have reported poor performance of SAM on medical segmentation tasks such as polyp segmentation [39]. To address this limitation, they proposed employing Adapter to effectively fine-tune SAM for downstream medical segmentation tasks. Following their experimental setup, we applied our method to SAM and conducted a comparative analysis with other PEFT approaches. Our GPS yielded exceptional results, as shown in Tab. 5 and visually depicted in Fig. 5 (See Supplementary for more case visualization).

4.4. Impacts of different selection schemes

Different selection levels Our GPS selects trainable parameters at the neuron level, i.e. selecting top-k input connection per neuron. We also investigate parameter selection methods at different levels. As shown in Tab. 6 (a), *Net* and *Layer* represent selecting a certain proportion of the parameters with the highest gradient based on the entire network and each layer, respectively. For a fair comparison, we keep the same number of parameters selected over these levels. We can see that the finer the granularity of selection, the better the performance. For example, the accuracy on CUB increases by 0.44% and 0.77% when selection level changes from network to layer, and from layer to neuron.

Different selection criteria We further study the effectiveness of our gradient-based selection method by com-

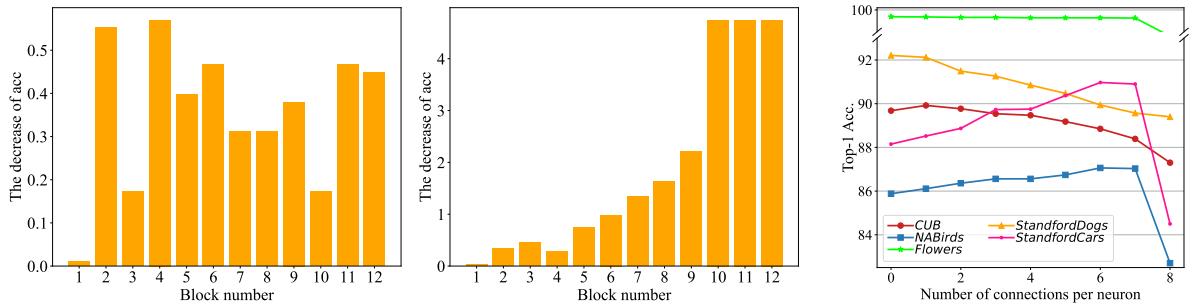


Figure 6. Impacts of different selection locations and quantities. From left to right: (a) Performance drop caused by not selecting parameters from k-th blocks. (b) And by not selecting from the top k blocks. (c) Impacts of different numbers of selected connections on performance.

	Dataset	CUB	NAbirds	Flowers	Cars	Dogs
(a)	Net	86.86	86.55	99.62	89.65	91.32
	Layer	87.30	86.79	99.64	90.03	91.90
(b)	Net Random	86.60	85.98	99.61	89.10	91.34
	Neuron Random	87.17	86.02	99.62	89.52	91.82
	Magnitude	87.29	85.99	99.62	89.29	91.30
(c)	Head+CE	87.05	86.20	99.64	89.25	91.29
	GPS	**88.07** ±0.11	**86.64** ±0.03	**99.69** ±0.01	**90.10** ±0.10	**92.30** ±0.10

Table 6. The result on FGVC for investigating impacts of different selection schemes and ablations. (a) Different selection levels. (b) Different selection criteria. (c) Gradient calculating method.

paring different selection criteria. As shown in Tab. 6 (b), *Net Random* and *Neuron Random* means randomly selecting top-K the input connection for each neuron and selecting the same number of parameters based on the whole network respectively. *Magnitude* means selecting top-K input connections with the largest weight per neuron. As we can see, the increase in the randomness of parameter selection causes a decrease in performance (*Net Random*<*Neuron Random*). The result of *Magnitude* is similar to *Neuron Random*, demonstrating neuron-level selection is crucial.

Different selection location To investigate the impact of selected parameters located at different layers within the network, we conducted experiments using the ViT-B/16 model fine-tuning on CUB and evaluated accuracy degradation when applying our GPS method to select parameters from the entire network except for a specific transformer block or previous several transformer blocks. As shown in Fig. 6(a), it is surprising to note that when we do not select parameters from a specific block, the biggest drop in the accuracy comes from the shallow layers (block 2 and block 4). This finding supports our GPS approach that selects parameters from the entire network rather than just the last few layers. When we do not select the parameters from the first specific number of blocks, it is observed that the accuracy drop increases with more blocks removed (Fig. 6(b)).

4.5. Ablation study

Head-free contrastive loss To obtain more accurate gradients for selecting parameters, inspired by the representation learning pre-training methods, Our GPS adopts the supervised contrastive loss to calculate gradient (without random initialization of the classification head). As shown in Tab. 6 (c), when we use the cross-entropy loss (with the head) to calculate the gradient, the average accuracy on FGVC is dropped by 0.67%, illustrating the importance of obtaining accurate gradients.

Selected connection number As shown in Fig. 6(c) we select top-K input connections per neuron as trainable parameters, ranging from 1 to 15, and conduct experiments on the 5 tasks. We can observe that more trainable parameters do not necessarily lead to better performance, but each data set has a performance peak. In addition, on the dataset with sufficient training data, the addition of trainable parameters can greatly improve the accuracy. Our GPS can easily control the number of trainable parameters and achieve optimal results on each dataset.

Robustness to seeds Addition-based fine-tuning methods like VPT are sensitive to the initialization of additional parameters as well as random seeds, whereas select-based methods are not. All results in Tab. 6 are the average accuracy of three seeds on FGVC datasets (Only shows the std of GPS here. Please see details in supplementary). The results show random seeds have little influence on our method.

5. Conclusion

In this paper, we propose a new paradigm for PEFT, *i.e.* Gradient-based Parameter Selection (GPS). Our approach

does not introduce any additional parameters and only fine-tunes a small subset of the pre-trained model's parameters for downstream tasks, resulting in robust generalization across diverse models and adaptively selecting a subset of parameters for each task. Remarkably, GPS achieves significant improvement on a range of tasks (including image classification and semantic segmentation), compared to the full fine-tuning method. GPS also attains SOTA performance compared to other PEFT methods.

References

[1] Ankur Bapna, Naveen Arivazhagan, and Orhan Firat. Simple, scalable adaptation for neural machine translation. *arXiv preprint arXiv:1909.08478*, 2019. 3, 8

[2] Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016. 7

[3] Jiezhang Cao, Jincheng Li, Xiping Hu, Xiangmiao Wu, and Mingkui Tan. Towards interpreting deep neural networks via layer behavior understanding. *Machine Learning*, 111:1159–1179, 2022. 4

[4] Niladri S Chatterji, Behnam Neyshabur, and Hanie Sedghi. The intriguing role of module criticality in the generalization of deep networks. *arXiv preprint arXiv:1912.00528*, 2019. 3

[5] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. The lottery ticket hypothesis for pre-trained bert networks. *ArXiv*, abs/2007.12223, 2020. 8

[6] Tianrun Chen, Lanyun Zhu, Chaotao Ding, Runlong Cao, Shangzhan Zhang, Yan Wang, Zejian Li, Lingyun Sun, Papa Mao, and Ying Zang. Sam fails to segment anything? – sam-adapter: Adapting sam in underperformed scenes: Camouflage, shadow, and more, 2023. 7

[7] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017. 7

[8] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014. 7

[9] Y. L. Cun, J. S. Denker, and S. A. Solla. *Optimal brain damage*. Advances in neural information processing systems 2, 1990. 8

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 5, 7

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 8

[12] Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Hai-Tao Zheng, and Maosong Sun. Openprompt: An open-source framework for prompt-learning. *arXiv preprint arXiv:2111.01998*, 2021. 3, 8

[13] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi Min Chan, and Weize and Chen. Parameter-efficient fine-tuning of large-scale pre-trained language models. *Nature Machine Intelligence*, 5(3):220–235, 2023. 2

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 5, 1, 8

[15] Fenglei Fan, Jinjun Xiong, Mengzhou Li, and Ge Wang. On interpretability of artificial neural networks: A survey. *IEEE Transactions on Radiation and Plasma Medical Sciences*, 5: 741–760, 2020. 4

[16] Li Fei-Fei, Robert Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006. 7

[17] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018. 3, 9

[18] Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 12799–12807, 2023. 4

[19] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019. 3, 9

[20] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*, 2020. 3, 8

[21] Timnit Gebru, Jonathan Krause, Yilun Wang, Duyun Chen, Jia Deng, and Li Fei-Fei. Fine-grained car detection for visual census estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017. 5, 6, 7

[22] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 7

[23] Ben Graham. Kaggle diabetic retinopathy detection competition report. *University of Warwick*, pages 24–26, 2015. 7

[24] Demi Guo, Alexander M Rush, and Yoon Kim. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*, 2020. 9

[25] Yunhui Guo, Honghui Shi, Abhishek Kumar, Kristen Grauman, Tajana Rosing, and Rogerio Feris. Spottune: transfer learning through adaptive fine-tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4805–4814, 2019. 3, 9

[26] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*, 2015. 3, 9

[27] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015. 3, 9

[28] Haoyu He, Jianfei Cai, Jing Zhang, Dacheng Tao, and Bohan Zhuang. Sensitivity-aware visual parameter-efficient tuning. *arXiv preprint arXiv:2303.08566*, 2023. 5, 6, 2

[29] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *ArXiv*, abs/2110.04366, 2021. 2

[30] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019. 7

[31] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. 7, 8

[32] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021. 7, 8

[33] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262–15271, 2021. 7, 8

[34] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019. 1, 2, 3, 4, 5, 6, 8, 9

[35] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Annual Meeting of the Association for Computational Linguistics*, 2018. 4

[36] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 2, 5, 6

[37] Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Jingang Wang, Juanzi Li, Wei Wu, and Maosong Sun. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. *arXiv preprint arXiv:2108.02035*, 2021. 3, 8

[38] Shengding Hu, Zhen Zhang, Ning Ding, Yadao Wang, Yasheng Wang, Zhiyuan Liu, and Maosong Sun. Sparse structure search for parameter-efficient tuning, 2022. 2

[39] Debesh Jha, Steven A. Hicks, Krister Emanuelsen, Håvard Johansen, Dag Johansen, Thomas de Lange, Michael A. Riegler, and Pål Halvorsen. Medico multimedia task at mediaeval 2020: Automatic polyp segmentation, 2020. 7, 4, 8

[40] Debesh Jha, Pia H Smedsrud, Michael A Riegler, Pål Halvorsen, Thomas de Lange, Dag Johansen, and Håvard D Johansen. Kvasir-seg: A segmented polyp dataset. In *International Conference on Multimedia Modeling*, pages 451–462. Springer, 2020. 8

[41] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, pages 709–727. Springer, 2022. 1, 2, 3, 5, 6, 7, 4, 9

[42] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017. 7

[43] Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. Prompting visual-language models for efficient video understanding. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXV*, pages 105–124. Springer, 2022. 3, 8

[44] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035, 2021. 3, 6, 8

[45] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR workshop on fine-grained visual categorization (FGVC)*. Citeseer, 2011. 5, 6, 7

[46] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *ArXiv*, abs/2004.11362, 2020. 3

[47] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[48] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 5, 7, 4

[49] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 5, 2, 7

[50] John K Kruschke and Javier R Movellan. Benefits of gain: Speeded learning and minimal hidden layers in backpropagation networks. *IEEE Transactions on systems, Man, and Cybernetics*, 21(1):273–280, 1991. 3, 9

[51] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054*, 2022. 3

[52] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, pages II–104. IEEE, 2004. 7

[53] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *ArXiv*, abs/1608.08710, 2016. 8

[54] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. *arXiv preprint arXiv:1608.08710*, 2016. 3, 9

[55] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021. 2, 3, 8

[56] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. *arXiv preprint arXiv:2210.08823*, 2022. 1, 2, 3, 5, 6, 7, 4, 9

[57] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9): 1–35, 2023. 3, 8

[58] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, 2022. 3, 8

[59] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 5, 8

[60] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022. 5

[61] Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. *arXiv preprint arXiv:2106.04489*, 2021. 3, 8

[62] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset, 2017. 7

[63] Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems*, 34: 23296–23308, 2021. 3

[64] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011. 7

[65] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 5, 6, 7

[66] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012. 7

[67] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. *arXiv preprint arXiv:2005.00247*, 2020. 3, 6, 8

[68] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterhub: A framework for adapting transformers. *arXiv preprint arXiv:2007.07779*, 2020. 3, 8

[69] Sai Prasanna, Anna Rogers, and Anna Rumshisky. When bert plays the lottery, all tickets are winning. In *Conference on Empirical Methods in Natural Language Processing*, 2020. 8

[70] John G. Proakis and Dimitris G. Manolakis. Digital signal processing: Principles, algorithms, and applications. 1992. 4

[71] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 8

[72] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128, 2021. 3

[73] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30, 2017. 2, 3, 8

[74] Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. Adapterdrop: On the efficiency of adapters in transformers. *arXiv preprint arXiv:2010.11918*, 2020. 3, 8

[75] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, 2017. 4

[76] Asa Cooper Stickland and Iain Murray. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995. PMLR, 2019. 3, 8

[77] Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, et al. On transferability of prompt tuning for natural language understanding. *arXiv preprint arXiv:2111.06719*, 2021. 2

[78] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5227–5237, 2022. 3, 6, 8

[79] Grant Van Horn, Steve Branson, Ryan Farrell, Scott Haber, Jessie Barry, Panos Ipeirotis, Pietro Perona, and Serge Belongie. 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*, pages 595–604, 2015. 5, 6, 7

[80] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia

Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 1

[81] Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation equivariant cnns for digital pathology. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II 11*, pages 210–218. Springer, 2018. 7

[82] Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. Spot: Better frozen model adaptation through soft prompt transfer. *arXiv preprint arXiv:2110.07904*, 2021. 2

[83] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. *California Institute of Technology*. 5, 6, 7

[84] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[85] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Guihong Cao, Daxin Jiang, Ming Zhou, et al. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*, 2020. 3, 8

[86] Zifeng Wang, Shao-Lun Huang, Ercan Engin Kuruoglu, Jimeng Sun, Xi Chen, and Yefeng Zheng. Pac-bayes information bottleneck. *ArXiv*, abs/2109.14509, 2021. 4

[87] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29, 2016. 3, 9

[88] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010. 7

[89] Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. Raise a child in large language model: Towards effective and generalizable fine-tuning. *arXiv preprint arXiv:2109.05687*, 2021. 3, 9

[90] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014. 3

[91] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021. 2, 3, 5, 6, 7, 4, 9

[92] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019. 5, 6, 7

[93] Renrui Zhang, Rongyao Fang, Wei Zhang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. *arXiv preprint arXiv:2111.03930*, 2021. 3, 8

[94] Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. Masking as an efficient alternative to finetuning for pretrained language models. *arXiv preprint arXiv:2004.12406*, 2020. 9

[95] Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Zihang Jiang, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*, 2021. 8

[96] Daquan Zhou, Zhiding Yu, Enze Xie, Chaowei Xiao, Animashree Anandkumar, Jiashi Feng, and Jose M Alvarez. Understanding the robustness in vision transformers. In *International Conference on Machine Learning*, pages 27378–27394. PMLR, 2022. 8

Gradient-based Parameter Selection for Efficient Fine-Tuning

Supplementary Material

6. Details of experiments

6.1. Baseline description

Vision Transformer (ViT) As a transformer-based visual model, ViT [14] has been widely adopted in various visual tasks. Most of the experiments are conducted on pre-trained ViT architecture in this paper. Given an input image $I \in \mathbb{R}^{H\times W\times 3}$, before feeding the image into the Transformer, the image is partitioned into M patches and appended a [CLS] token for classification purposes, resulting in final input $x \in \mathbb{R}^{(M+1)\times d}$ where d is the dimension of the features. The Transformer typically consists of multiple blocks and each block contains a Multi-head Attention layer (MHA) and two MLP layers[80].

Adapter Work in [34] proposed the Adapter method, which inserts multiple trainable layers (termed as Adapter) into the pre-trained Transformer encoder. Only the Adapter is updated during the fine-tuning stage. These layers can be inserted after either the Multi-head Attention layer or the MLP layer. Adapter comprises two projection matrices, one W^{down} for dimension reduction and the other W^{up} for feature reconstruction to the original dimension. Specifically, given the input $x \in \mathbb{R}^{(M+1)\times d}$, the output of the Adapter is

$$y = \left[W^{\text{up}}\,\phi\left(W^{\text{down}}\,x^{T}\right)\right]^{T} \tag{7}$$

where $W^{\text{up}} \in \mathbb{R}^{d'\times d}$, $W^{\text{down}} \in \mathbb{R}^{d\times d'}$ (where $d' \ll d$), and ϕ is a nonlinear activation function.

Prompt Visual prompt tuning (VPT) introduces learnable parameters ($i.e.$, prompts) into the input space [41]. When fine-tuning downstream tasks, the backbone is fixed, and just tuning these prompts. Formally, the given input $x \in \mathbb{R}^{(M+1)\times d}$ is concatenated with m introduced prompts $p \in \mathbb{R}^{m\times d}$. The final combined input is

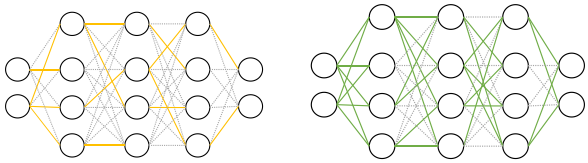
$$x' = [x; p] \tag{8}$$

where $x' \in \mathbb{R}^{(M+1+m)\times d}$ will be feed into the Transformer. There are two versions of VPT, namely VPT-shallow and VPT-deep. The former introduces learnable prompts solely into the input space of the first layer, whereas the latter integrates them into each layer's input space.

Scale and shift feature SSF attempts to scale and shift the features between the layers of the pre-trained model by adding a linear transform layer [56]. During fine-tuning the downstream tasks, only the linear transform layers are updated while the backbone remains frozen. The transform layer consists of two components, scale factor $\gamma \in \mathbb{R}^{d}$ and shift factor $\beta \in \mathbb{R}^{d}$, for feature transformation. To be specific, given the input $x \in \mathbb{R}^{(M+1)\times d}$, the output is calculated by

$$y = \gamma \odot x + \beta \tag{9}$$

where $y \in \mathbb{R}^{(M+1)\times d}$, $\odot$ is the dot product.



(a) One input connection (b) Two input connections

Figure 7. Different number of connections with highest gradient value among all input connections per neuron, such as (a) selecting only one input connection per neuron. (b) two input connections are selected per neuron.

6.2. The number of parameters on different tasks

For each neuron in the network, our GPS method selected at least one of the connections (weight or parameter) with the highest gradient value, among the input connections of the neuron, as shown in Fig. 7(a). For downstream tasks that need more learnable parameters to better fit the data, such as those tasks with dissimilar data distributions from the upstream dataset (such as NABirds) or larger amounts of data (such as CIFAR-100), our method can be easily extended by introducing more learnable parameters. Specifically, for each neuron, we can select multiple input connections with the highest gradient values instead of limiting them to just one, as shown in Fig. 7(b). Tab. 7 show the detailed statics on the number of parameters that are selected in our paper. For most of the tasks in this paper, we just select one of the connections. We also explore the relationship between the number of connections and the number of learnable parameters. As shown in Fig. 8, with the increase in the number of selected connections with the highest gradient value among the input connections per neuron, the number of learnable parameters linear ascent.

6.3. Parameters distribution of *Net* selection

In contrast to our approach, a simple approach is to select the parameters for a specific task by selecting a certain percentage of parameters with the highest gradient from the en-

Dataset	Params. (M)	Conne.s	Dataset	Params. (M)	Conne.s	Dataset	Params. (M)	Conne.s
CUB-200-2011	0.47	2	Pets	0.23	1	DMLab	0.20	1
NABirds	1.35	10	SVHN	0.20	1	KITTI/distance	0.20	1
Oxford Flowers	0.29	1	Sun397	0.55	1	dSprites/loc	0.21	1
Stanford Dogs	0.30	1	Patch Camelyon	0.30	2	dSprites/ori	0.21	1
Stanford Cars	1.07	10	EuroSAT	0.20	1	SmallNORB/azi	0.21	1
CIFAR-100*	0.29	1	Resisc45	0.24	1	SmallNORB/ele	0.20	1
Caltech101	0.29	1	Retinopathy	0.20	1	CIFAR-100	0.58	5
DTD	0.24	1	Clevr/count	0.30	1	CIFAR-100 (Swin)	0.82	5
Flowers102	0.29	1	Clevr/distance	0.20	1	CIFAR-100 (ConvNeXt)	0.78	5

Table 7. The number of learnable parameters and connections across all tasks. CIFAR-100* is a subset of CIFAR-100 in VTAB benchmark. In bracket is the model architecture, without bracket represents the one fine-tuned on ViT-B/16. Params. means the learnable parameters and the Conne. represents the number of selected input connections for each neuron in the network.

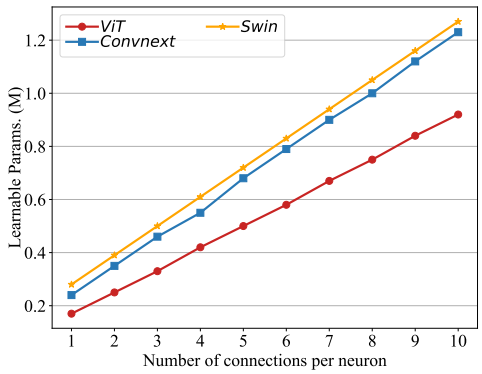


Figure 8. The number of learnable parameters with the different number of connections on VIT-B/16, Swin and Convnext archite. The learnable parameters do not contain the task-specific head.

Method \ Dataset	ImageNet -1K (↑)	ImageNet -A (↑)	ImageNet -R (↑)	ImageNet -C (↓)
Full [41]	83.58	34.49	51.29	46.47
Linear [41]	82.04	33.91	52.87	46.91
Bias [91]	82.74	42.12	55.94	41.90
Adapter [34]	82.72	42.21	54.13	42.65
VPT-Shallow [41]	82.08	30.93	53.72	46.88
VPT-Deep [41]	82.45	39.10	53.54	43.10
SSF [56]	83.10	45.88	56.77	**41.47**
GPS	**83.91**	**46.11**	**57.00**	42.04

Table 8. Performance comparisons on the ImageNet with different model architectures.

tire network [28]. However, as shown in Figs. 9(a) to 9(e), most of the selected parameters are located in the upper layers, specifically block 12 and block 11. As a result, the network is primarily focused on fine-tuning abstract features while lacking the ability to fine-tune detailed information from shallower layers. Our approach addresses this challenge by carefully selecting the input connections for each individual neuron, resulting in our selected parameters being evenly distributed on the whole network, as shown in Fig. 9(f).

7. Additional experiments

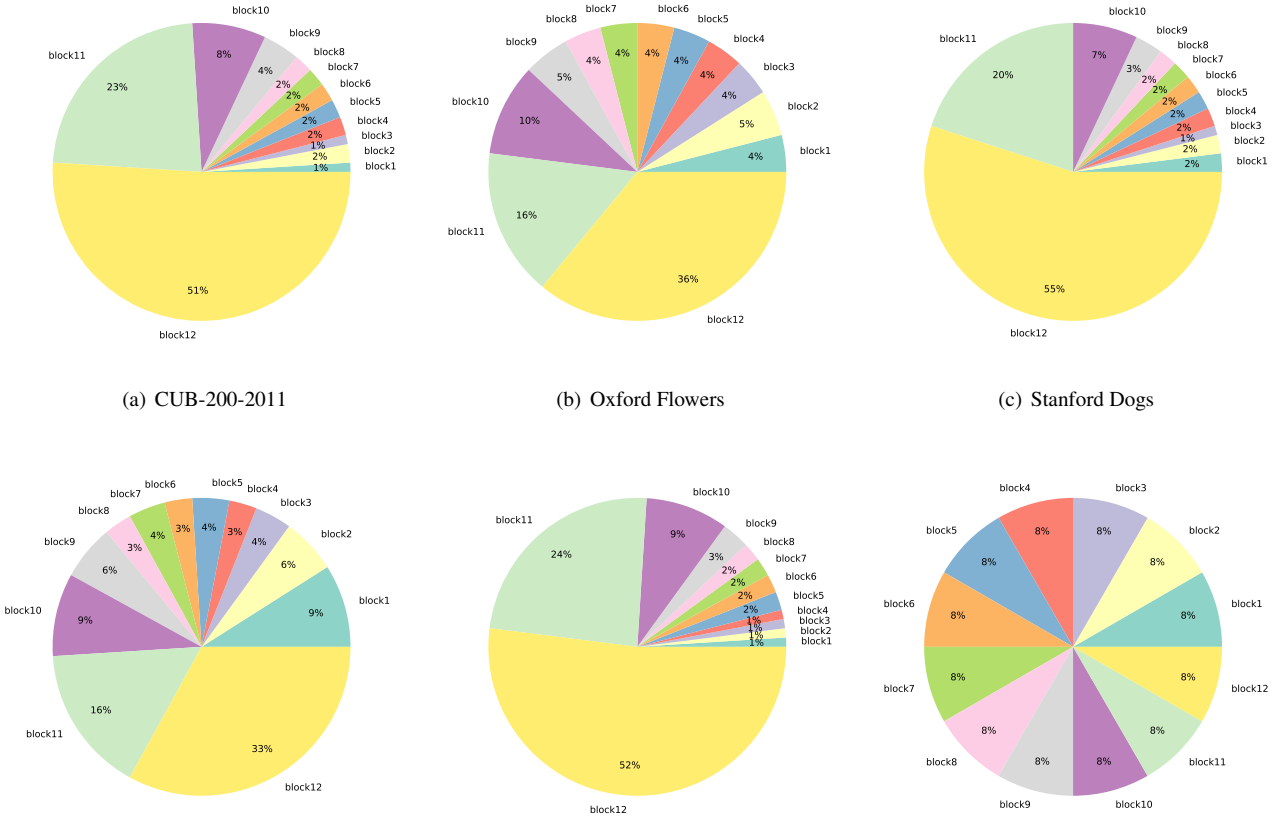
7.1. Robustness and OOD datasets

In addition to standard classification tasks, we further analyze the robustness and OOD generalization ability of GPS. Based on the Imagenet-A, ImageNet-R, and ImageNet-C datasets, we first fine-tune the model on ImageNet-1K, and then test the fine-tuned model on the three datasets respectively. The results are shown in Tab. 8. GPS not only achieves the best performance on the standard ImageNet-1K classification task but also achieves good performance in robustness and generalization tests. Among them, GPS achieves the best results on ImageNet-A and ImageNet-R, outperforms the previous optimal SSF by 0.23%, reflecting the strong stability and generalization ability of our method. On ImageNet-C, GPS performs slightly worse, lagging behind SSF, but still higher than addition-based Adapter and VPT. This result indicates that our method can quickly adapt to the data distribution of downstream tasks, but it needs to be improved in anti-interference.

7.2. More experiments on different architecture

As mentioned in the main body of our paper, our method is model-agnostic, we further compare GPS with other fine-tuning methods across ViT-B/16, Swin-B, and ConvNeXt-B architectures on the ImageNet-1k [10] and CIFAR-100 [49] datasets.

CIFAR-100 As shown in Tab. 10, unlike FGVC and VTAB, GPS and other efficient tuning methods have difficulty in achieving competitive performance as full tuning on CIFAR-100. This may be due to that CIFAR-100 contains more training data, allowing all parameters of the entire model to be adequately trained, which seriously reduces the advantages of efficient fine-tuning methods. However,



(a) CUB-200-2011 (b) Oxford Flowers (c) Stanford Dogs

(d) Stanford Cars (e) NABirds (f) GPS

Figure 9. Distribution of the parameter over the whole network ViT-B/16 on 6 FGVC dataset (CUB-200-2011, Oxford Flowers, Stanford Dogs, Stanford Cars and NABirds). Selecting the 1% parameters with the highest gradient value from the whole network, instead of our method selecting at least one of the connections among all input connections per neuron. In contrast to this method, our GPS has the same distribution over different downstream tasks (f).

Method \ Dataset	CUB-200 -2011	NABrids	Oxford Flowers	Stanford Dogs	Stanford Cars	Mean Acc.	Mean Params. (M)	Mean Params. (%)
ViT-B/16 + Full	87.3	82.7	98.8	89.4	84.5	88.54	85.98	100.00
ViT-B/16 + Linear	85.3	75.9	97.9	86.2	51.3	79.32	0.18	0.21
ViT-B/16 + SSF	89.5	85.7	99.6	89.6	89.2	90.72	0.39	0.45
ViT-B/16 + GPS (Ours)	**89.9**	**86.7**	**99.7**	**92.2**	**90.4**	**91.78**	0.66	0.77
Swin-B + Full	90.7	**89.8**	99.5	88.9	**93.2**	92.42	86.98	100.00
Swin-B + Linear	90.6	86.8	99.2	88.3	74.6	87.90	0.24	0.28
Swin-B + SSF	90.5	88.4	**99.7**	88.7	90.4	91.54	0.49	0.56
Swin-B + GPS (Ours)	**90.8**	88.9	**99.7**	**92.7**	90.7	**92.56**	0.83	0.95
ConvNeXt-B + Full	**91.2**	**90.4**	99.6	89.9	**94.1**	93.04	87.81	100.00
ConvNeXt-B + Linear	90.6	86.9	99.3	89.7	73.5	88.00	0.24	0.28
ConvNeXt-B + SSF	90.8	89.0	**99.7**	90.4	92.5	92.48	0.50	0.56
ConvNeXt-B + GPS (Ours)	91.0	89.6	**99.7**	**93.7**	92.6	**93.32**	0.79	0.90

Table 9. Performance comparisons on FGVC benchmark with different model architectures.

Architecture	ViT-B/16		Swin-B		ConvNeXt-B	
	Acc.	Params.(%)	Acc.	Params.(%)	Acc.	Params.(%)
Full [41]	93.82	100.00	**93.85**	100.00	**94.14**	100.00
Linear [41]	88.70	0.09	89.27	0.12	89.20	0.12
Bias [91]	93.39	0.21	92.19	0.28	92.80	0.27
Adapter [34]	93.34	0.36	92.49	0.38	92.86	0.52
VPT-Shallow [41]	90.38	1.07	90.02	0.15	-	-
VPT-Deep [41]	93.17	1.43	92.62	0.81	-	-
SSF [56]	93.99	0.33	93.06	0.43	93.45	0.42
GPS (Ours)	**94.02**	0.68	93.55	0.96	93.58	0.90

Table 10. Performance comparisons on the CIFAR-100 with different model architectures.

Architecture	ViT-B/16		Swin-B		ConvNeXt-B	
	Acc.	Params.(%)	Acc.	Params.(%)	Acc.	Params.(%)
Full [41]	83.58	100.00	**85.20**	100.00	**85.80**	100.00
Linear [41]	82.04	0.89	83.25	1.17	84.05	1.16
Bias [91]	82.74	1.00	83.92	1.32	84.63	1.31
Adapter [34]	82.72	1.16	83.82	1.43	84.49	1.54
VPT-Shallow [41]	82.08	1.06	83.29	1.19	-	-
VPT-Deep [41]	82.45	1.42	83.44	1.85	-	-
SSF [56]	83.10	1.12	84.40	1.47	84.85	1.44
GPS (ours)	**83.91**	1.37	84.43	1.96	84.87	1.90

Table 11. Performance comparisons on the ImageNet-1k with different model architectures.

GPS still outperforms all previous parameter-efficient tuning methods (Bias, Adapter, VPT, and SSF) and reduces the gap with full fine-tuning to less than 0.5% on all architectures, which further demonstrates the adaptability of our approach to different models.

ImageNet-1k Similar to the results on CIFAR-100, ImageNet-1K contains more training data, which makes it harder for parameter-efficient fine-tuning algorithms to achieve the same accuracy as full fine-tuning, as shown in Tab. 11. However, GPS still outperforms full fine-tuning by 0.33% on ViT structure, and outperforms the previous SOTA method SSF on Swin and ConvNeXt structures respectively, which further shows the generalization of GPS to different model structures.

FGVC As mentioned in the main body of our paper, our method achieves the best result on FGVC benchmark. Tab. 9 shows the full results of Tab. 4 in the main body. Among all three model architectures, GPS consistently outperforms all other baselines, demonstrating its model-agnostic advantage.

7.3. Data-efficient tuning

Recent advances in large foundation model fine-tuning have shown considerable promise in reaching state-of-the-art performance on various tasks. However, in order to reach high accuracy, these methods often need significant volumes of training data, which may be time-consuming and costly to obtain. Here we demonstrate that our method is data efficient, that is, with such a few-shot setting, our method requires only a small amount of training data for tuning to achieve outstanding results that other approaches do not. Specifically, we fine-tune the ViT-B/16 by selecting only k samples for each class in the ImageNet dataset to form a few-shot training set. The value of k and the accuracy of predicted results are illustrated in Fig. 10, which demonstrates the excellent data efficiency of our method especially in extreme cases like k=1.

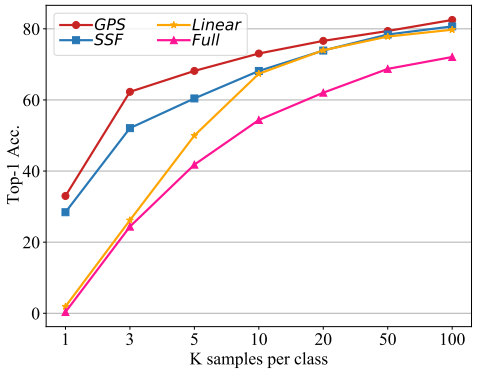


Figure 10. The performance comparison of different fine-tuning methods under k values for each class on ImageNet dataset. Our method is always above the curve of the others. The advantage of our approach is particularly evident in extreme cases where data is extremely scarce (e.g., k=1).

7.4. Random seed for impacts of different selection schemes and ablations

We conduct experiments with three random seeds to investigate the robustness of our method. As shown in Tab. 12, Our parameter selection method significantly outperforms the other methods with small randomness. Tab. 12 is a supplementary of Tab. 6 in the main body of our paper.

8. Visualizations

8.1. Semantic segmentation

As mentioned in the main body of our paper, our approach demonstrates highly promising results in the field of semantic segmentation. We apply our method on the pre-trained strong segmentation model (SAM) [48] and fine-tune on a medical segmentation task – polyp segmentation [39]. Here, we present more case visualizations, which could directly show the effectiveness of our method, as shown in Fig. 11.

		CUB	NAbirds	Flowers	Cars	Dogs
(a)	Net	86.86 ± 0.21	86.55 ± 0.03	99.62 ± 0.01	89.65 ± 0.12	91.32 ± 0.07
	Layer	87.30 ± 0.13	86.79 ± 0.08	99.64 ± 0.01	90.03 ± 0.13	91.90 ± 0.11
(b)	Net Random	86.60 ± 0.10	85.98 ± 0.07	99.61 ± 0.01	89.10 ± 0.12	91.34 ± 0.12
	Neuron Random	87.17 ± 0.15	86.02 ± 0.10	99.62 ± 0.01	89.52 ± 0.23	91.82 ± 0.23
	Magnitude	87.29 ± 0.12	85.99 ± 0.08	99.62 ± 0.00	89.29 ± 0.02	91.30 ± 0.02
(c)	Head+CE	87.05 ± 0.19	86.20 ± 0.14	99.64 ± 0.01	89.25 ± 0.09	91.29 ± 0.01
	GPS	**88.07** ± 0.11	**86.64** ± 0.03	**99.69** ± 0.01	**90.10** ± 0.10	**92.30** ± 0.10

Table 12. Impacts of different selection schemes and ablations. (a) Different selection levels. (b) Different selection criteria. (c) Different ways to calculate gradients.

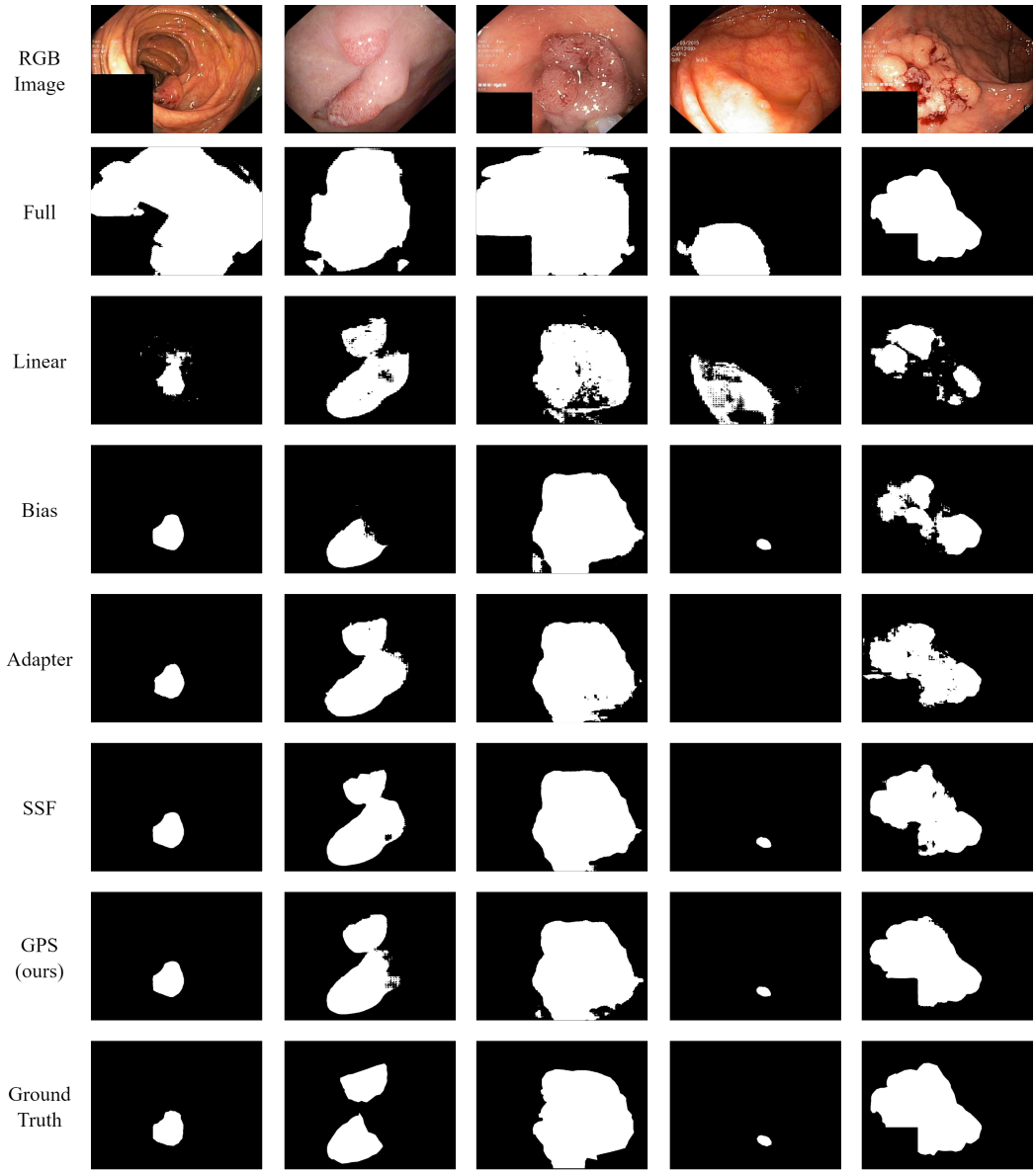
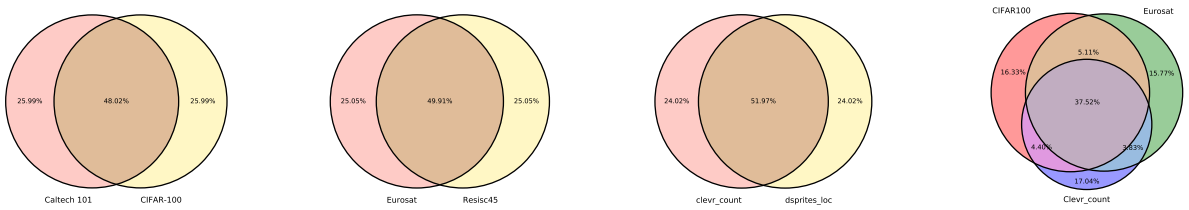
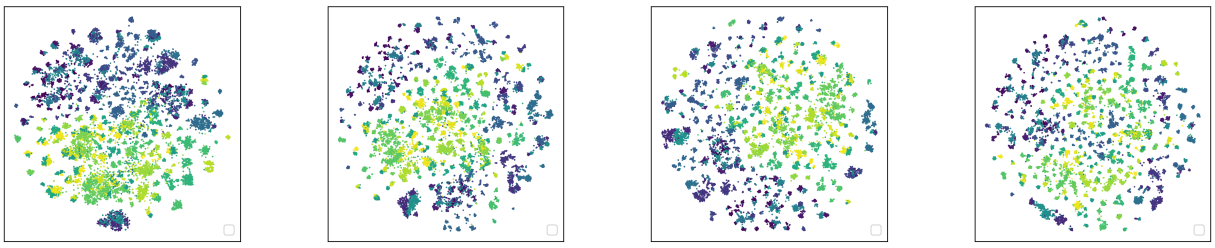


Figure 11. **The Visualization Result of Polyp Segmentation.** Here GT means ground truth. As illustrated in this figure, although SAM and other methods can identify some polyp structures in the image, the result is not accurate. By using GPS, our approach elevates the performance with SAM.



(a) (b) (c) (d)

Figure 12. The overlap of the selected parameters across different tasks. Overlap is determined based on parameter position. If selected parameters share the same position in the network, they are considered to have overlap. We test on the ViT-B/16 with following tasks: (a) Cifar100 and Caltech101; (b) Eurosat and Resisc45; (c) Clevr/count and Dsprites/loc; (d) Cifar100, Eurosat and Clevr/count.



(a) Linear Probing (b) SSF (c) GPS (ours) (d) Full fine-tuning

Figure 13. t-SNE visualization of different fine-tuning methods, including linear probing, SSF, GPS, full fine-tuning.

8.2. Distribution of selected parameters across various tasks

As we select different subsets of parameters from the original model for different downstream tasks, a normal question is how different the distribution of the selected parameters is across different tasks. we test on Vit-B/16 with 6 downstream tasks from VTAB (two from Natural, two from Specialized and the other two from Structured). As shown in Fig. 12, the chosen parameters exhibit a tendency towards 2/3 shared parameters and 1/3 task-specific parameters, despite the dissimilar data distribution of downstream tasks. This is due to our selection scheme, which makes the parameters evenly distribute on the whole network and thus the parameters from shallow layers tend to share parameters as similar findings from the field of multi-task learning [44, 67, 78].

8.3. Feature distribution

On the NABirds datasets, we use t-SNE to visualize the feature distribution of different fine-tuning methods. The results of all comparison methods are obtained based on the ViT-B/16 pre-trained on ImageNet-21k. The visualization results are illustrated in Fig. 13. Feature clustering results using our GPS are superior to those with linear probing, SSF, and full fine-tuning.

9. Details of the evaluation datasets

The statistic of all datasets used in this paper is shown in Tab. 13.

9.1. Image classification

FGVC Fine-Grained Visual Classification (FGVC) benchmark includes 5 downstream tasks, which are CUB-200-2011 [83], NABirds [79], Oxford Flowers [65], Stanford Dogs [45] and Stanford Cars [21]. Each one contains more than 100 classes and a few thousand images. We directly use the public splits if one contains, otherwise, we follow the splits in [41].

VTAB-1k Visual Task Adaptation Benchmark [92] contains 19 visual classification tasks which are grouped into 3 sets: (1) Natural – tasks with natural images captured by standard cameras; (2) Specialized – tasks with images captured via specialized equipment, e.g., medical camera or satellite sensor; (3) Structured – tasks with images synthesized from simulated environments, which require geometric comprehension like object counting and depth estimation. Each one contains only 1000 training examples while

Dataset	Description	#Classes	Train	Val	Test
Fine-Grained Visual Classification (FGVC)					
CUB-200-2011 [83]	Bird recognition	200	5,394	600	5,794
NABirds [79]	Bird recognition	555	21,536	2,393	24,633
Oxford Flowers [65]	Flower recognition	102	1,020	1,020	6,149
Stanford Dogs [45]	Dog recognition	120	10,800	1,200	8,580
Stanford Cars [21]	Car classification	196	7,329	815	8,041
Visual Task Adaptation Benchmark (VTAB-1k) [92]					
CIFAR-100 [49]	Natural	100	800/1000	200	10,000
Caltech101 [16]		102			6,084
DTD [8]		47			1,880
Flowers102 [65]		102			6,149
Pets [66]		37			3,669
SVHN [64]		10			26,032
Sun397 [88]		397			21,750
Patch Camelyon [81]	Specialized	2	800/1000	200	32,768
EuroSAT [30]		10			5,400
Resisc45 [7]		45			6,300
Retinopathy [23]		5			42,670
Clevr/count [42]	Structured	8	800/1000	200	15,000
Clevr/distance [42]		6			15,000
DMLab [2]		6			22,735
KITTI/distance [22]		4			711
dSprites/location [62]		16			73,728
dSprites/orientation [62]		16			73,728
SmallNORB/azimuth [52]		18			12,150
SmallNORB/elevation [52]		9			12,150
General Image Classification Datasets					
CIFAR-100 [49]	General images	100	50,000	-	10,000
ImageNet-1K [10]		1,000	1,281,167	50,000	150,000
Robustness and Out-of-Distribution Datasets					
ImageNet-A [33]	Robustness & OOD	200	-	-	7,500
ImageNet-R [32]		200	-	-	30,000
ImageNet-C [31]		1,000	-	-	75×50,000
Cross-domain Semantic Segmentation Dataset					
Kvasir-SEG [39]	Polyp Segmentation	2	880	-	120

Table 13. Detailed statistics of the datasets evaluated on our work. We follow the VPT [41] for train/val split. This table is partially borrowed from VPT[41] and SSF [56].

a large number of test images (i.e. over 20,000 on average).

CIFAR-100 CIFAR-100 [49] is a widely used general image classification task. It contains 50,000 training and 10,000 test images with 100 categories.

ImageNet-1K ImageNet-1K [10] is the most commonly utilized subset of ImageNet for object classification, encompassing 1000 classes and featuring a training set of 1,281,167 images, a validation set of 50,000 images, and a test set of 100,000 images.

9.2. Semantic segmentation

Polyp segmentation We select kvasir-SEG [40] for polyp segmentation task. We follow the settings in Medico automatic polyp segmentation· task at mediaeval 2020 [39] with a train-valid ratio of 880:120.

9.3. Robustness and OOD

ImageNet-A ImageNet-A [33] contains 200 classes, which is selected from ImageNet-1K (1000 classes). All samples are real-world adversarial samples that caused the ResNet model to produce erroneous classifications.

ImageNet-R ImageNet-R [32] contains art, graffiti, sculptures, tattoos, toys, cartoons, paintings, embroidery, deviantart, graphics, patterns, plastic objects, origami, plush objects, sketches, and video game renditions from ImageNet classes.

ImageNet-C ImageNet-C [31] is an open-source collection of algorithmically generated corruptions, such as blur and noise, that have been applied to the ImageNet test set.

10. Disscussion

Why sub-network? There is a lot of research in the field of neural network pruning, where researchers aim to identify the importance of the parameters in a network and eliminate some unnecessary parameters without performance deterioration (about 90% parameters of the model are pruned) [5, 9, 53, 69]. Motivated by this, we posit the existence of a sub-network containing crucial parameters that can be fine-tuned for optimal performance on downstream tasks.

Magnitude or gradient? In contrast to the approaches of identifying the importance of the parameters in [5, 9, 53, 69], which rely on weight magnitude to determine parameter importance, our method identifies parameter importance based on gradient values. An important difference between gradient and magnitude is that the gradient-based method is task-specific, as the gradient is calculated by the backpropagation of the loss for a specific task, while the magnitude-based method use a set of same parameters for all downstream tasks. However, our ablation study in the main body has shown gradient-based method perform bwt-ter and the Fig. 12 also show that each task has it own task-specific parameters.

11. Limitations and societal impacts

Limitations Several studies [44, 67, 78] have demonstrated that certain similar tasks can be optimized together through parameter sharing, resulting in improved performance across all individual tasks. However, our work focuses on selecting distinct parameters for various tasks. Although we already tune affordable parameters, we do not fully exploit the potential of parameter sharing across different tasks. Therefore, we posit that our work can be extended to a multitask setting, where tasks share tuning parameters and thus further reduce the total number of learnable parameters.

Societal impacts Our method can effectively fine-tune pre-trained models for downstream tasks by adjusting less than 1% of the network's parameters. This is particularly beneficial when dealing with large pre-trained models and multiple downstream tasks, as it saves computational resources, memory costs, and reduces carbon emissions. Our approach maintains the model's original structure without introducing any additional parameters during both the training and inference stages, distinguishing it from other methods. However, similar to other fine-tuning approaches, our method relies on a pre-trained model. If this upstream pre-trained model is trained on illicit data, it may also violate the use of fine-tuning methods.

12. Extended related work

12.1. Visual parameter efficient fine-tuning

In the field of computer vision, current work endeavors to pre-train larger models [11, 14, 59, 71, 95, 96] on extensive datasets, followed by fine-tuning diverse downstream tasks to achieve superior performance and faster convergence. Conventional arts set all the network parameters learnable and adapt them to the target tasks. However, as foundation models become increasingly large and the number of downstream tasks increases, it becomes impractical due to the significant computational and storage requirements that it entails. Parameter-efficient fine-tuning (PEFT) methods are proposed to alleviate such a burden, which tunes only a tiny portion of the parameters. The general PEFT can be categorized into addition-based and selection-based methods.

Addition-based methods introduce additional parameters to the pre-trained backbone. Adapter methods keep most of the parameters in the model frozen and update only small-scale injected parameters. Bottleneck-structured adapters [1, 34, 67, 68, 73, 74, 76, 78, 85, 93] adopt a residual pathway to leverage both original and task-specific knowledge by learning down-projection and up-projection with a nonlinear activation. Others [61] propose a hyper-network to generate model weights or decompose the dense weighted matrix into the low-rank matrix to reduce parameters [44]. Instead of introducing extra modules, prompt methods [12, 20, 37, 43, 55, 57, 58] wrap the input with

context. A representative work VPT [41] prepend learnable prompts to the input tokens before feeding it into each Transformer block. VPT includes two variants VPT-Shallow and VPT-Deep associated with the number of inserted layers. VPT-Shallow simply prepends prompts to the first transformer layer while VPT-Deep prepends prompts to all the layers. However, it's inflexible when applying the method to new tasks since it relies on hand-crafted prompt length selection. Apart from the adapter and prompt tuning, a recent study SSF [56] introduces two learnable vectors to scale and shift the feature map in each transformer operation and achieves promising results. These extra parameters will lead to a substantial increase in computational overhead and hinder the rate of convergence. Our method solves these issues without adding parameters or changing the network topology so it can effectively alleviate such problems.

Selection-based methods [24, 91, 94] do not introduce any new parameters but directly select part of the parameters to be optimized without modifying the intrinsic architecture of the model. Bitfit [91] only fine-tunes bias vectors in the pre-trained model. Other methods only fine-tune the top-K layers [34] or the last linear layer [41] with other layers freeze. Despite efficiency, they suffer a significant accuracy drop compared to the full fine-tuning since the manually specified parameters tend to be a non-optimal solution. Our gradient-based parameter selection method falls into this category. Since the gradient can serve as a tool for determining parameter significance, our method is intuitive but surprisingly effective.

12.2. Subset network training

Standard pruning technique [19, 26, 27, 50, 54, 87] naturally uncovers subnetworks whose initializations made them capable of training effectively. The lottery ticket hypothesis [17] articulate that subnetworks can reach test accuracy comparable to the original network. Drawing from the theory, fine-tuning methods based on subset network are widely studied. SpotTune [25] designs a policy network to make routing decisions for subset networks. Child-tuning [89] iteratively updates a subset of parameters by masking out the gradients of the non-child network during the backward process. However, the computing overhead led by hyper networks or iterative parameter selection makes none of these methods parameter-efficient. We fix the position of parameters that will be updated by simple gradient weights sorting before training which makes our method parameter efficient.