Efficient Adaptation of Large Vision Transformer via Adapter Re-Composing

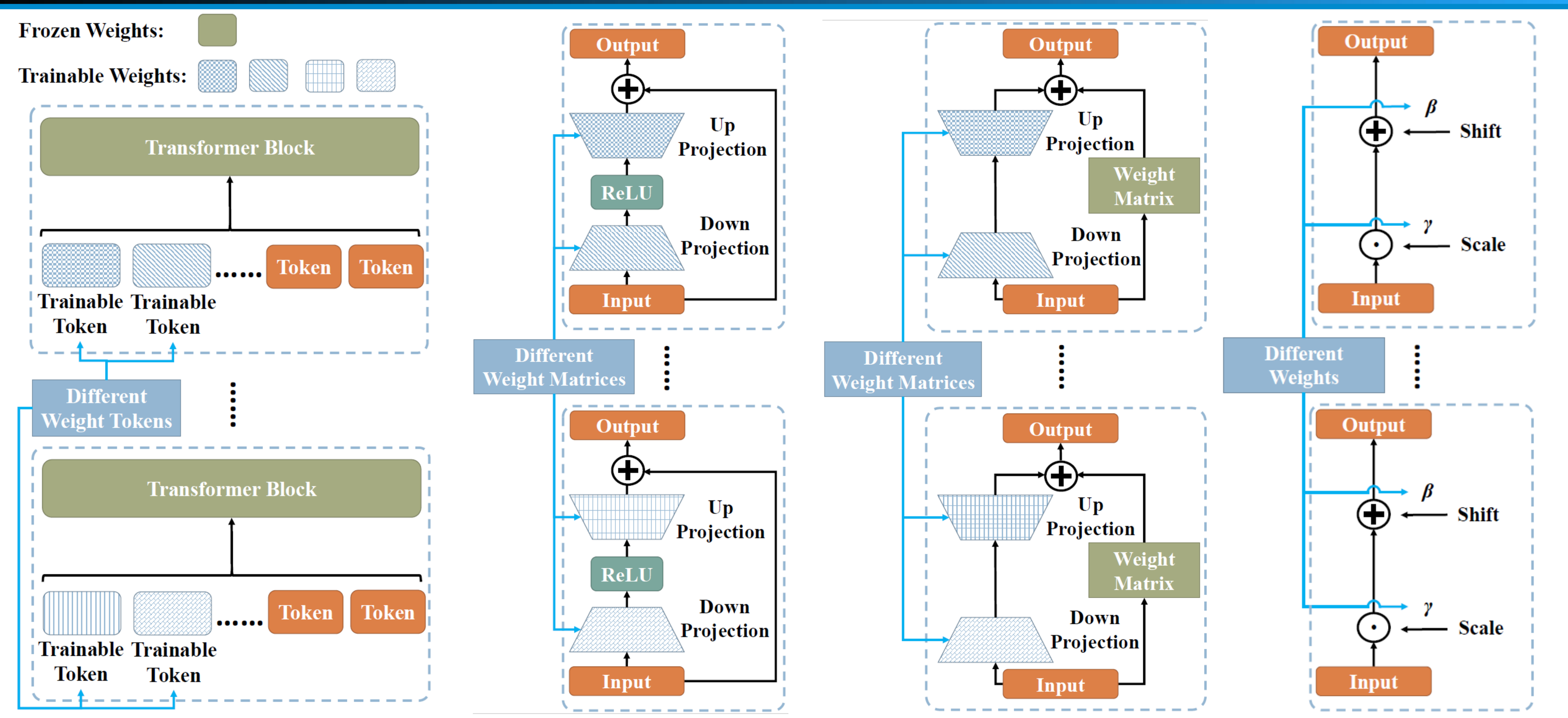
Wei Dong[1] Dawei Yan[1] Zhijun Lin[2] Peng Wang[3]★

1 College of Information and Control Engineering, Xi'an University of Architecture and Technology

2 School of Computer Science, Northwestern Polytechnical University

3 School of Computer Science and Engineering, University of Electronic Science and Technology of China

MOTIVATION



- Fine-tuning large-scale pre-trained models for downstream tasks.
  - Updating full fine-tuning model is expensive.
  - Existing methods keep pre-trained parameters frozen and only update a small of task-specific parameters.
- Adapter Re-Composing.
  - Exploring the importance of adaptation parameter reusability and further compressing the adaptation cost.

CONTRIBUTION

- We approach efficient pre-trained model adaptation from a novel perspective by exploring the reusability of adaptation parameters, which goes beyond existing works that primarily focus on the lightweight design of adapter structures.
- We introduce the Adapter Re-Composing (ARC) strategy, which shares parameters across layers and utilizes lower-dimensional re-composing coefficients to create layer-adaptive adapters, keeping a linear increase in parameter size with the number of layers.
- Through extensive experiments on various Vision Transformer variations and numerous downstream tasks, we show that our method achieves highly competitive transfer learning performance.
- Our codes is available at:

https://github.com/DavidYanAnDe/ARC

METHODOLOGY

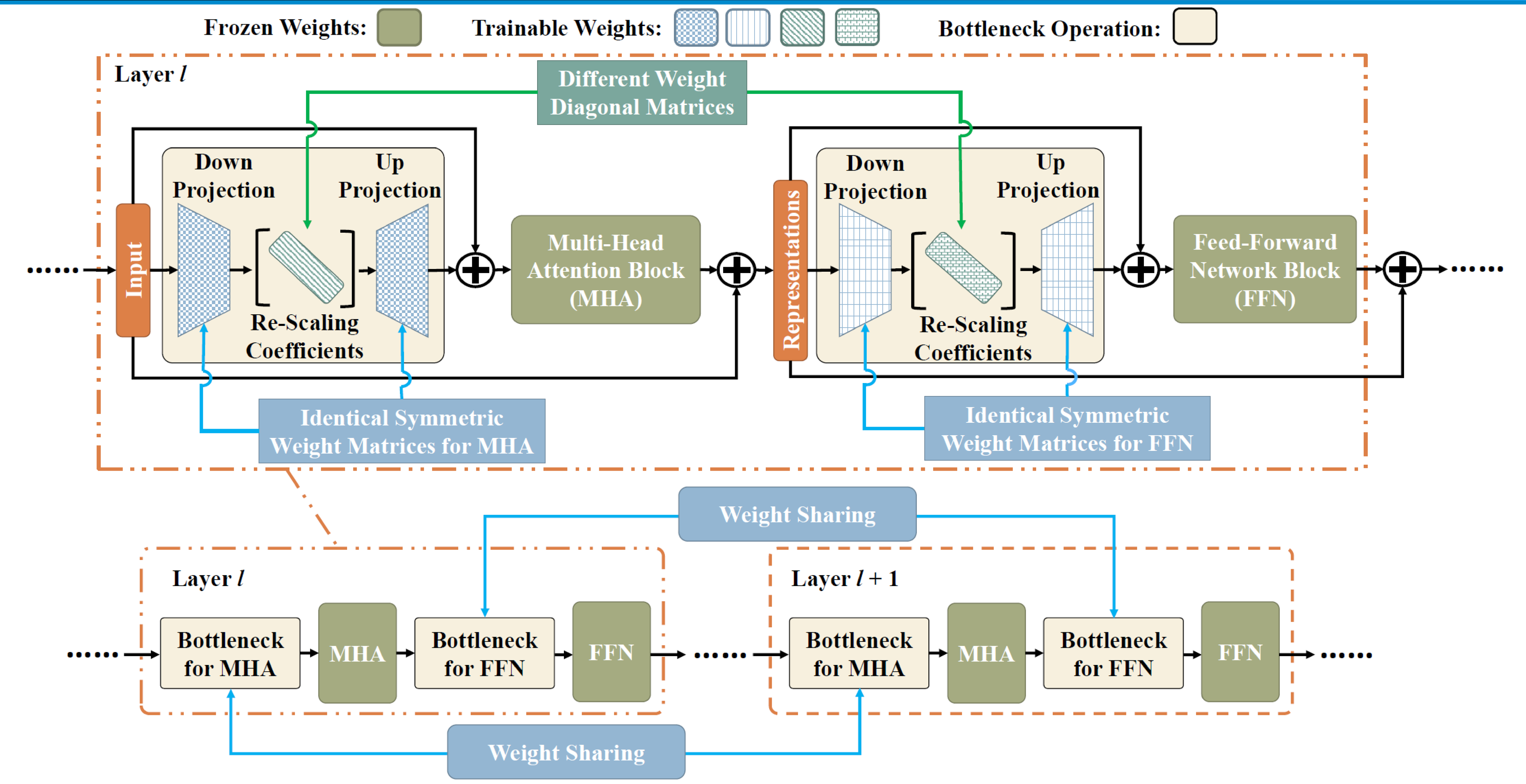


Illustration of the proposed Adapter Re-Composing Method.

Plain Vision Transformer processing flow

$$\mathbf{X}_{\text{emb}} = \left[\vec{x}_{\text{cls}}^{\mathrm{T}}; \mathbf{X}_{\text{patches}}\mathbf{W}\right] + \mathbf{X}_{\text{pos}}$$

$$\mathbf{X}^{(l)\prime} = \mathrm{MHA}\left(\mathrm{LN}\left(\mathbf{X}^{(l-1)}\right)\right) + \mathbf{X}^{(l-1)}$$

$$\mathbf{X}^{(l)} = \mathrm{FFN}\left(\mathrm{LN}\left(\mathbf{X}^{(l)\prime}\right)\right) + \mathbf{X}^{(l)\prime}$$

$$\mathbf{X}_h^{(l)\prime} = \mathrm{AH}_h\left(\mathbf{X}_{\text{norm}}^{(l-1)}\right) = \mathrm{softmax}\left(\frac{\left(\mathbf{X}_{\text{norm}}^{(l-1)}\mathbf{W}_q^{(l)}\right)\left(\mathbf{X}_{\text{norm}}^{(l-1)}\mathbf{W}_k^{(l)}\right)^{\mathrm{T}}}{\sqrt{D_h^{(l)}}}\right)\mathbf{X}_{\text{norm}}^{(l-1)}\mathbf{W}_v^{(l)}$$

$$\mathbf{X}^{(l)\prime} = \mathrm{MHA}(\mathbf{X}_{\text{norm}}^{(l-1)}) = [\mathrm{AH}_1(\mathbf{X}_{\text{norm}}^{(l-1)}),\ \cdots,\ \mathrm{AH}_M(\mathbf{X}_{\text{norm}}^{(l-1)})]\mathbf{W}_o^{(l)}$$

$$\mathbf{X}^{(l)} = \mathrm{FFN}\left(\mathbf{X}_{\text{norm}}^{(l)\prime}\right) = \mathrm{GELU}\left(\mathbf{X}_{\text{norm}}^{(l)\prime}\mathbf{W}_1^{(l)}\right)\mathbf{W}_2^{(l)}$$

Adapter Re-Composing method

We apply adapter by sharing the parameter among up and down matrices as: $\mathbf{W}_{\text{up}} = (\mathbf{W}_{\text{down}})^{\mathrm{T}}$

Adapters are shared between each layer.
(adapter are independent for MHA and FFN)

$$\mathbf{X}_{\text{out}} = \mathrm{ARC}(\mathbf{X}_{\text{in}}) = \mathbf{X}_{\text{in}}\mathbf{W}_{\text{down}}\mathbf{C}^{(l)}\mathbf{W}_{\text{up}} + \mathbf{X}_{\text{in}}$$

$$\mathbf{W}_1^{(l)\prime} = \left(\mathbf{W}_{\text{down}}\mathbf{C}^{(l)}\mathbf{W}_{\text{up}} + \mathbf{I}\right)\mathbf{W}_1^{(l)}$$

$$\mathbf{X}^{(l)\prime} = \mathrm{MHA}\left(\mathrm{ARC}_{\mathrm{MHA}}\left(\mathrm{LN}\left(\mathbf{X}^{(l-1)}\right)\right)\right) + \mathbf{X}^{(l-1)}$$

$$\mathbf{X}^{(l)} = \mathrm{FFN}\left(\mathrm{ARC}_{\mathrm{FFN}}\left(\mathrm{LN}\left(\mathbf{X}^{(l)\prime}\right)\right)\right) + \mathbf{X}^{(l)\prime}$$

$$\mathbf{X}^{(l)} = \mathrm{GELU}\left(\mathrm{ARC}_{\mathrm{FFN}}(\mathbf{X}^{(l)\prime})\mathbf{W}_1^{(l)}\right)\mathbf{W}_2^{(l)}$$

RESULT

(a). Comparison of ARC with baselines and state-of-the-art efficient adaptation methods on five FGVC datasets. All methods utilize ViT-B/16 pre-trained on ImageNet-21k as the backbone.

Method \ Dataset	CUB-200-2011	NABirds	Oxford Flowers	Stanford Dogs	Stanford Cars	Mean	Params.(M)
Full fine-tuning	87.3	82.7	98.8	89.4	84.5	88.5	85.98
Linear probing	85.3	75.9	97.9	86.2	51.3	79.3	0.10
Adapter [7]	87.1	84.3	98.5	89.8	68.6	85.7	0.41
Bias [42]	88.4	84.2	98.8	91.2	79.4	88.4	0.28
VPT-Shallow [6]	86.7	78.8	98.4	90.7	68.7	84.6	0.25
VPT-Deep [6]	**88.5**	84.2	99.0	90.2	83.6	89.1	0.85
LoRA [24]	88.3	85.6	99.2	91.0	83.2	89.5	0.44
SSF* [9]	89.5	85.7	99.6	89.6	89.2	90.7	0.39
SSF [9]	82.7	**85.9**	98.5	87.7	82.6	87.5	0.39
$\mathrm{ARC}_{\mathrm{att}}$	88.4	85.0	**99.4**	90.1	82.7	89.1	0.15
ARC	**88.5**	85.3	99.3	**91.9**	**85.7**	**90.1**	0.20

(b). Comparison of ARC with baselines and state-of-the-art efficient adaptation methods on VTAB-1k benchmark. All methods utilize ViT-B/16 pre-trained on ImageNet-21k as the backbone.

	Natural								Specialized					Structured										
Method \ Dataset	CIFAR-100	Caltech101	DTD	Flowers102	Pets	SVNH	Sun397	Mean	Camelyon	EuroSAT	Resisc45	Retinopathy	Mean	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele	Mean	Mean Total	Params.(M)
Full fine-tuning	68.9	87.7	64.3	97.2	86.9	87.4	38.8	75.9	79.7	95.7	84.2	73.9	83.4	56.3	58.6	41.7	65.5	57.5	46.7	25.7	29.1	47.6	65.6	85.8
Linear probing	63.4	85.0	63.2	97.0	86.3	36.6	51.0	68.9	78.5	87.5	68.6	74.0	77.2	34.3	30.6	33.2	55.4	12.5	20.0	9.6	19.2	26.9	52.9	0.04
Adapter [7]	74.1	86.1	63.2	97.7	87.0	34.6	50.8	70.5	76.3	88.0	73.1	70.5	77.0	45.7	37.4	31.2	53.2	30.3	25.4	13.8	22.1	32.4	55.8	0.27
Bias [42]	72.8	87.0	59.2	97.5	85.3	59.9	51.4	73.3	78.7	91.6	72.9	69.8	78.3	61.5	55.6	32.4	55.9	66.6	40.0	15.7	25.1	44.1	62.1	0.14
VPT-Shallow [6]	77.7	86.9	62.6	97.5	87.3	74.5	51.2	76.8	78.2	92.0	75.6	72.9	79.7	50.5	58.6	40.5	67.1	68.7	36.1	20.2	34.1	47.0	64.9	0.11
VPT-Deep [6]	**78.8**	**90.8**	65.8	98.0	88.3	78.1	49.6	78.5	81.8	**96.1**	83.4	68.4	82.4	68.5	60.0	46.5	72.8	73.6	47.9	**32.9**	37.8	55.0	69.4	0.60
LoRA [24]	65.3	87.9	69.4	98.7	90.7	82.4	53.4	78.2	82.8	94.8	82.5	75.0	83.8	77.6	64.7	45.8	79.0	73.3	44.7	26.3	38.2	56.2	70.1	0.29
SSF* [9]	69.0	92.6	75.1	99.4	91.8	90.2	52.9	81.6	87.4	95.9	87.4	75.5	86.6	75.9	62.3	53.3	80.6	77.3	54.9	29.5	37.9	59.0	73.1	0.24
SSF [9]	58.0	89.8	70.5	98.9	90.2	90.5	52.9	78.7	**86.7**	95.2	86.4	75.4	**85.9**	68.2	61.0	**52.8**	80.7	77.3	48.5	27.6	31.1	55.9	70.6	0.24
$\mathrm{ARC}_{\mathrm{att}}$	70.1	90.5	70.5	98.8	90.8	88.6	53.6	80.4	84.6	95.5	86.6	75.5	85.6	79.0	65.6	48.6	81.3	75.1	48.7	29.1	39.6	58.4	72.2	0.08
ARC	72.2	90.1	**72.7**	**99.0**	**91.0**	**91.9**	**54.4**	**81.6**	84.9	95.7	**86.7**	**75.8**	85.8	**80.7**	**67.1**	48.7	**81.6**	**79.2**	**51.0**	31.4	**39.9**	**60.0**	**73.4**	0.13

(c). Performance comparison on VTAB-1k using ViT-Large and ViT-Huge pre-trained on ImageNet-21k as backbone.

(a) ViT-Large

	Natural (7)	Specialized (4)	Structed (8)	Mean Total	Params.
Full fine-tuning	74.7	83.8	48.1	65.4	303.4
Linear probing	70.9	69.1	25.8	51.5	0.05
Adapter [7]	68.6	73.5	29.0	52.9	2.38
Bias [37]	70.5	73.8	41.2	58.9	0.32
VPT-Shallow [6]	78.7	79.9	40.6	62.9	0.15
VPT-Deep [6]	**82.5**	83.9	54.1	70.8	0.49
LoRA [24]	81.4	85.0	**57.3**	72.0	0.74
ARC	82.3	**85.6**	**57.3**	**72.5**	0.18

(b) ViT-Huge

	Natural (7)	Specialized (4)	Structed (8)	Mean Total	Params.
Full fine-tuning	70.9	83.6	46.0	63.1	630.9
Linear probing	67.9	79.0	26.1	52.7	0.06
Adapter [7]	68.1	76.4	24.5	51.5	5.78
Bias [42]	70.3	78.9	41.7	60.1	0.52
VPT-Shallow [6]	74.8	81.2	43.0	62.8	0.18
VPT-Deep [6]	77.9	83.3	52.2	68.2	0.96
LoRA [24]	77.1	83.5	**55.4**	69.3	1.21
ARC	**79.1**	**84.8**	53.7	**69.6**	0.22

(d). Performance comparison on VTAB-1k using Swin-Base pre-trained on ImageNet-21k as backbone.

	Natural (7)	Specialized (4)	Structed (8)	Mean Total	Params.
Full fine-tuning	79.1	86.2	59.7	72.4	86.8
Linear probing	73.5	80.8	33.5	58.2	0.05
MLP-4 [6]	70.6	80.7	31.2	57.7	4.04
Partial [6]	73.1	81.7	35.0	58.9	12.65
Bias [42]	74.2	80.1	42.4	62.1	0.25
VPT-Shallow [6]	**79.9**	82.5	37.8	62.9	0.05
VPT-Deep [6]	76.8	84.5	53.4	67.7	0.22
ARC	79.0	**86.6**	**59.9**	**72.6**	0.27