Tuning Pre-trained Model via Moment Probing

Mingze Gao[1,2,†] Qilong Wang[1,*] Zhenyi Lin[1] Pengfei Zhu[1] Qinghua Hu[1] Jingbo Zhou[2]
[1]Tianjin Key Lab of Machine Learning, College of Intelligence and Computing, Tianjin University, China
[2]Business Intelligence Lab, Baidu Research, China
{gaomingze, qlwang, linzhenyi, zhupengfei, huqinghua}@tju.edu.cn, zhoujingbo@baidu.com

Abstract

Recently, efficient fine-tuning of large-scale pre-trained models has attracted increasing research interests, where linear probing (LP) as a fundamental module is involved in exploiting the final representations for task-dependent classification. However, most of the existing methods focus on how to effectively introduce a few of learnable parameters, and little work pays attention to the commonly used LP module. In this paper, we propose a novel Moment Probing (MP) method to further explore the potential of LP. Distinguished from LP which builds a linear classification head based on the mean of final features (e.g., word tokens for ViT) or classification tokens, our MP performs a linear classifier on feature distribution, which provides the stronger representation ability by exploiting richer statistical information inherent in features. Specifically, we represent feature distribution by its characteristic function, which is efficiently approximated by using first- and second-order moments of features. Furthermore, we propose a multi-head convolutional cross-covariance (MHC^3) to compute second-order moments in an efficient and effective manner. By considering that MP could affect feature learning, we introduce a partially shared module to learn two recalibrating parameters (PSRP) for backbones based on MP, namely MP_+. Extensive experiments on ten benchmarks using various models show that our MP significantly outperforms LP and is competitive with counterparts at lower training cost, while our MP_+ achieves state-of-the-art performance.

1. Introduction

Benefiting from the emergence of huge-scale datasets [8, 42, 41], the rapid development of neural network architectures [10, 17, 43, 35] and self-supervised learning [15, 39, 5], large-scale pre-trained models dependent on sufficient computational resources show the great potential of the transferability on downstream tasks [9, 7, 40, 4], where full fine-tuning as a basic method has achieved promising performance. However, full fine-tuning suffers from a high computational cost and is easy overfitting on small-scale datasets [32, 23]. In contrast, a simpler and more efficient method is only to tune a linear classifier (i.e., linear probing [16]). Compared to full fine-tuning, linear probing (LP) usually suffers from inferior performance. To address this, existing works make a lot of efforts on parameter-efficient strategies [21, 2, 22, 30, 24, 23, 32, 6]. Going beyond LP, they focus on introducing a few learnable parameters to recalibrate features from frozen pre-trained models for downstream tasks. These methods avoid tuning massive parameters and show better efficiency and effectiveness trade-off.

Although many advanced efforts are made on parameter-efficient tuning, little work pays attention to the most fundamental LP, which is involved in all existing tuning methods to learn a task-dependent classification head, and is closely related to the performance of downstream tasks. It can be observed that LP learns a linear classifier on input features, which are generally presented by classification token [10], average pooling (mean) of word tokens [43, 35] or convolution features [36]. From the statistical perspective, LP

Method	IN-1K (%)	NABirds (%)	Params. (M)	Time (ms)	Mem. (G)
Linear probing	82.04	75.9	0.77	**60**	**3.23**
MP (Ours)	**83.15**	**84.9**	3.65	**72**	**3.34**
VPT-Shallow [23]	82.08	78.8	0.92	115	11.39
VPT-Deep [23]	82.45	84.2	1.23	120	11.39
AdaptFormer [6]	83.01	84.7	1.07	125	10.49
SSF [32]	83.10	85.7	0.97	187	13.78
Full fine-tuning	83.58	82.7	86.57	157	11.92
MP_+ (Ours)	**83.62**	**86.1**	4.10	140	11.22

Table 1: Comparison of various tuning methods for pre-trained models in terms of recognition accuracy (%), learnable parameters (Params.), training time (Time) per mini-batch, and GPU memory (Mem.) usage of training on ImageNet-1K (IN-1K) and NABirds, where ViT-B/16 pre-trained on IN-21K is used as basic backbone.

† This work was done when Mingze Gao was an intern at Baidu Research. * Corresponding author

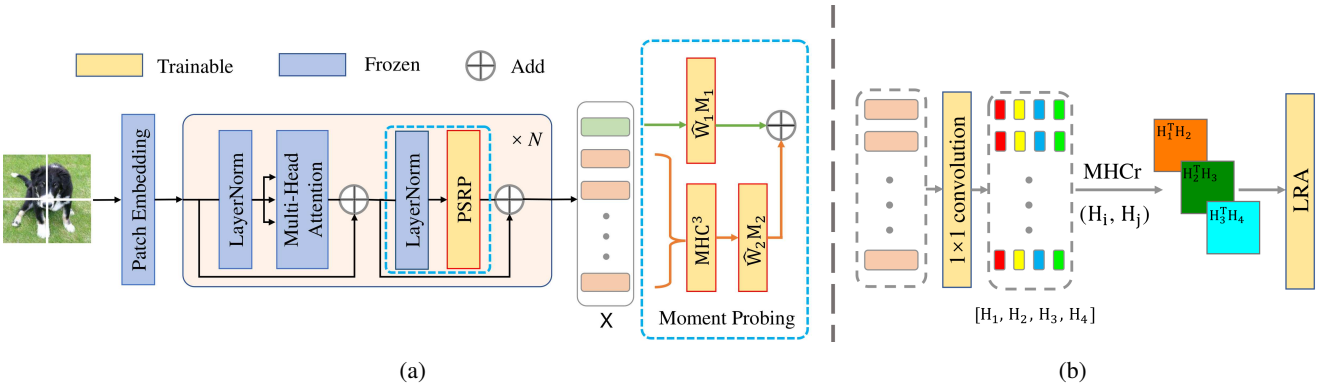


Figure 1: (a) Overview of proposed MP$_+$ method for tuning pre-trained models, whose core is Moment Probing (MP) indicated by blue dashed line instead of the original linear probing. Specifically, our MP performs a linear classifier on powerful representations characterized by feature distribution, which is approximated by using first- and second-order moments of features. To efficiently explore second-order moments, we present a (b) multi-head convolutional cross-covariance (MHC^3) method, whose details can refer to Sec. 3.1. Besides, a partially shared module to learn two recalibrating parameters (PSRP) is introduced for exploring the potential of MP on feature learning.

mainly exploits first-order statistics of features, taking no full merit of rich statistical information inherent in features.

In this paper, we further explore the potential of LP by generating more powerful representations for linear classifier. To this end, we propose a Moment Probing (MP) method, whose core is to perform a linear classifier on feature distribution, which portrays the full picture of features and provides more powerful representations. Specifically, we model probability density of features by their characteristic function, and approximate it by using first- and second-order moments of features for computational efficiency. Furthermore, we present a multi-head convolutional cross-covariance (MHC^3) method to efficiently compute second-order moments, avoiding the issues of computation and generalization brought by high-dimensional second-order representations. For computing MHC^3, we first split features into several groups and compute cross-covariance between each two adjacent groups to extract second-order statistics. Then, a parameter-efficient module based on convolutions is designed to strengthen the interactions among cross-covariances and reduce the size of second-order representations. Compared to the original second-order moments, our MHC^3 performs better in terms of both efficiency and effectiveness. Table 1 shows our MP shares similar training cost with LP, but obtains much higher accuracy. Meanwhile, MP is comparable to or better than existing parameter-efficient methods at lower training cost.

Since deep models are trained in an end-to-end learning manner, the classifier will bring effect on feature learning. To explore the potential of our MP on feature learning, we introduce a partially shared module to learn two recalibrating parameters (PSRP) for pre-trained models, inspired by parameter-efficient methods [6, 32]. By combining MP with PSRP, our MP$_+$ surpasses full fine-tuning while learning much fewer parameters, whose overview is illustrated in Figure 1. The contributions of our work are summarized as follows: (1) To our best knowledge, we make the first attempt to explore the potential of LP for tuning pre-trained models. To this end, we propose a Moment Probing (MP) method, which performs a linear classifier on powerful representations characterized by feature distribution. (2) For the efficiency of MP, we approximate feature distribution by using first- and second-order moments of features, and then present a multi-head convolutional cross-covariance (MHC^3) method to explore second-order moments in an efficient and effective manner. (3) By considering the effect of our MP on feature learning, we introduce a partially shared module to learn two recalibrating parameters (PSRP), resulting in a MP$_+$ method. It further exploits the potential of our MP in tuning pre-trained models. (4) We conduct extensive experiments on ten benchmarks using various models, and results show our MP is superior to LP while generalizing well to pre-training strategies, few-shot and out-of-distribution settings. Besides, our MP$_+$ outperforms existing parameter-efficient methods, while achieving state-of-the-art performance.

2. Related Work

2.1. Transfer Learning

Transfer learning aims to reuse the pre-trained models and re-adapting them to new tasks by fine-tuning them on a specific dataset. In both of natural language processing and computer vision communities, transferring pre-trained large models to downstream tasks has long been a popular paradigm [9, 2, 50, 4, 15, 39, 5, 7], which al-

ways provides rewarding performance with the total amount of pre-trained data and the scale of the model itself increase. During the transfer process, tuning strategies are limited to basic full fine-tuning and linear probing, suffering from low parameter efficiency and inferior performance. To solve above problem, recent works attempt to explore parameter-efficient methods by tuning a few of learnable parameters in the frozen backbone and sharing most of the parameters in the backbone for each downstream task [22, 21, 2, 32, 23, 6]. For example, some works selectively fine-tune the original parameters in the backbone where SpotTune [14] investigates which layers need to tune and Bitfit [2] finds that fine-tuning bias term is enough in some cases. Other works focus on inserting additional modules to adapt feature transfer. Among them, Adaptor-based methods [21, 24] insert an additional non-linear MLP block between frozen layers. VPT [23] adds learnable prompt tokens into the input space of frozen backbone. SSF [32] introduces both the scale and shift factors after each operation in backbone. AdaptFormer [6] proposes a parallel branch adding to the feed-forward network (FFN) of the pre-trained model. Different from aforementioned works, our MP makes the first attempt to explore the potential of LP, while achieving comparable or better performance than above counterparts at much lower training cost.

2.2. Second-order Pooling

Second-order moment involved in our MP is closely related to global covariance pooling (GCP) [34], which has been studied to improve the representation and generalization abilities of deep architectures. Previous works [34, 33, 28, 29] have shown that GCP is an effective alternative to global average pooling (GAP) for visual tasks. For example, B-CNN [34] and MPN-COV [28] insert GCP with different post-normalization methods into deep convolutional neural networks (CNNs) for fine-grained and large-scale visual, respectively. Recently, DropCov [46] proposes an adaptive channel dropout method for GCP normalization, and shows the effectiveness in both deep CNNs and ViTs. Besides, some researches focus on reducing the dimension of GCP representations, resulting in several compact models [11, 26]. Different from the above GCP methods, our MP presents a multi-head convolutional cross-covariance (MHC^3) to efficiently compute the second-order moment, which is designed for parameter-efficient tuning of pre-trained models and outperforms existing counterparts in terms of efficiency and effectiveness.

3. Proposed Method

In this section, we will introduce the proposed MP$_+$ method. As shown in Figure 1, the core of MP$_+$ is a Moment Probing (MP) instead of the original LP for tuning pre-trained models. Besides, a partially shared module to learn two recalibrating parameters (PSRP) is introduced to explore the potential of MP for feature learning. In the following, we will describe our MP and PSRP in detail.

3.1. Moment Probing

Linear Classifier on Feature Distribution. Given a set of N d-dimensional features $\mathbf{X} \in \mathbb{R}^{N\times d}$ output from the block right before classifier of pre-trained models (e.g., final word tokens for ViTs and last convolution features for CNNs), linear probing (LP) builds a linear classifier on a representation $g(\mathbf{X})$ generated by $\mathbf{X}$, i.e.,

$$\mathbf{y}_{pred} = \mathbf{W}g(\mathbf{X}), \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{C\times S}$ indicates weights of classifier. C and S are class number of downstream task and the size of representation $g(\mathbf{X})$, respectively. For the original LP, $g(\mathbf{X})$ is usually generated by classification token [10] (weighted combination of word tokens $\mathbf{X}$), average (mean) pooling of $\mathbf{X}$ (e.g., word tokens [43, 35] or convolution features [36]). From the statistical perspective, $g(\mathbf{X})$ of LP mainly exploits first-order statistics of features $\mathbf{X}$. As shown in Eqn. (1), prediction $\mathbf{y}_{pred}$ is definitely influenced by representation $g(\mathbf{X})$. Therefore, stronger representations potentially make the prediction more precise.

To this end, we propose a Moment Probing (MP) to generate stronger representations by exploiting probability distribution $P(\mathbf{X})$ of features $\mathbf{X}$ instead of simple mean point in $g(\mathbf{X})$, because $P(\mathbf{X})$ can characterize the full picture of $\mathbf{X}$. As such, our MP aims to perform a linear classifier on feature distribution $P(\mathbf{X})$:

$$\mathbf{y}_{pred} = \widehat{\mathbf{W}}P(\mathbf{X}). \tag{2}$$

However, $P(\mathbf{X})$ is usually unknown. Based on the classical probability theory [3], $P(\mathbf{X})$ can be defined by its characteristic function $\varphi_{\mathbf{X}}(t)$, according to Fourier transforms between $\varphi_{\mathbf{X}}(t)$ and probability density function $p(\mathbf{X})$ as

$$\varphi_{\mathbf{X}}(t) = E\left[e^{it\mathbf{X}}\right] = \int_{\mathbb{R}} e^{it\mathbf{x}} dP_{\mathbf{X}}(\mathbf{x}) = \int_{\mathbb{R}} e^{it\mathbf{x}} p_{\mathbf{X}}(\mathbf{x}) d\mathbf{x}, \tag{3}$$

where i is the imaginary unit, and $t \in \mathbb{R}$ is the argument of the characteristic function. Furthermore, we can rewrite $\varphi_{\mathbf{X}}(t)$ with the Taylor series of $e^{it\mathbf{X}}$ as:

$$\begin{aligned}\varphi_{\mathbf{X}}(t) &= E\left[\sum_{k=0}^{\infty} \frac{(it\mathbf{X})^k}{k!}\right] = \sum_{k=0}^{\infty} \frac{(it)^k}{k!} E\left[\mathbf{X}^k\right] \\ &= 1 + (it)E\left[\mathbf{X}\right] + \frac{(it)^2}{2!}E\left[\mathbf{X}^2\right] + \cdots,\end{aligned} \tag{4}$$

where $E\left[\mathbf{X}^k\right]$ indicates k^{th}-order moment ($\mathbf{M}_k$) of $\mathbf{X}$. Let the coefficient of $\mathbf{M}_k$ be ω_k, we can rewrite Eqn. (4) as

$$\varphi_{\mathbf{X}}(t) = 1 + \omega_1\mathbf{M}_1 + \omega_2\mathbf{M}_2 + \cdots = 1 + \sum_{k=1}^{\infty} \omega_k \mathbf{M}_k, \tag{5}$$

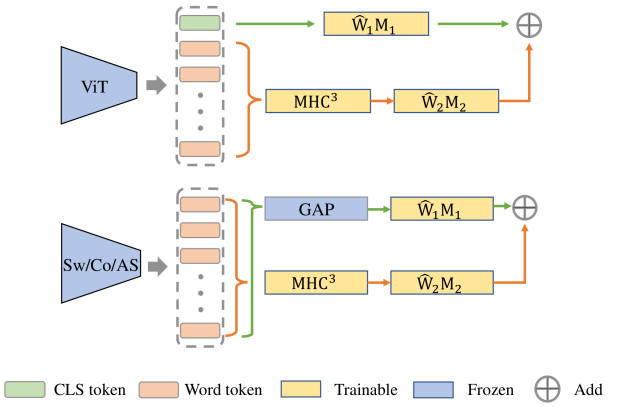


Figure 2: Usage details of MP for ViT, Swin transformer (Sw), ConvNeXt (Co) and AS-MLP (AS).

So far, we can represent feature distribution $P(\mathbf{X})$ by using its characteristic function as for Eqn. (5). By omitting the constant term, we reformulate Eqn. (2) as:

$$\mathbf{y}_{pred} = \widehat{\mathbf{W}}\left(\sum_{k=1}^{\infty}\omega_k\mathbf{M}_k\right) = \sum_{k=1}^{\infty}\left(\widehat{\mathbf{W}}_k\mathbf{M}_k\right), \quad (6)$$

where we can see that $\mathbf{y}_{pred}$ in Eqn. (6) fuses prediction results from different order moments, which are superior to single first-order statistics in the original LP. Although combination of more statistics generally leads to a more precise approximation on feature distribution, it brings much more computational cost and is sensitive to noise [48]. Therefore, our MP exploits the first- and second-order moments to approximate feature distribution for final prediction:

$$\mathbf{y}_{pred} = \sum_{k=1}^{2}\left(\widehat{\mathbf{W}}_k\mathbf{M}_k\right) = \widehat{\mathbf{W}}_1\mathbf{M}_1 + \widehat{\mathbf{W}}_2\mathbf{M}_2, \quad (7)$$

where $\mathbf{M}_1$ can be computed as classification token or average pooling of word tokens as for LP, while $\mathbf{M}_2$ is calculated by $\mathbf{X}^T\mathbf{X}$.

Efficient Second-order Moment. For the input features $\mathbf{X} \in \mathbb{R}^{N\times d}$, their second-order moment ($\mathbf{M}_2 = \mathbf{X}^T\mathbf{X}$) results in a d^2-dimensional representation. As such, $\mathbf{M}_2$ will create large-size representations for high-dimensional features, bringing the issues of computational burden and overfitting. To handle above issues, we present a multi-head convolutional cross-covariance (MHC^3) to efficiently compute the second-order moment. As shown in Figure 1b, we first exploit a 1×1 convolution to reduce dimension of $\mathbf{X}$ from d to $\hat{d}$ ($\hat{d} < d$), which are indicated by $\widehat{\mathbf{X}}$. Then, features $\widehat{\mathbf{X}}$ are split into h heads along feature dimension:

$$[\mathbf{H}_1;\mathbf{H}_2;\cdots;\mathbf{H}_h] = \mathrm{SP}\left(\widehat{\mathbf{X}}\right), \quad (8)$$

where SP indicates a splitting operation, and $\mathbf{H}_i \in \mathbb{R}^{N\times(\hat{d}/h)}$ are split features. To capture second-order statistics, we compute mutli-head cross-covariance between two adjacent split features, i.e.,

$$\mathbf{Z}_i = \mathrm{MHCr}(\mathbf{H}_i, \mathbf{H}_{(i+1)}) = \ell_2\left(\mathbf{H}_i^T\mathbf{H}_{(i+1)}\right), \quad (9)$$

where $\mathbf{Z}_i$ leads to a $(\hat{d}/h)^2$-dimensional cross-covariance representation, which captures second-order statistics between $\mathbf{H}_i$ and $\mathbf{H}_{(i+1)}$. ℓ_2 is an element-wise ℓ_2 normalization to control the magnitude of representation $\mathbf{Z}_i$.

It can be seen that $\mathbf{Z} = \{\mathbf{Z}_1;\mathbf{Z}_2;\cdots;\mathbf{Z}_{h-1}\}$ only capture second-order statistics between adjacent split features. To perform interaction among all features and further reduce representation size, we introduce a parameter-efficient Local Representation Aggregation (LRA) block for $\mathbf{Z}$:

$$\mathrm{LRA}(\mathbf{Z}) = \mathrm{Concat}[\mathrm{Conv}^2_{3\times3}(\sigma(\mathrm{Conv}^2_{3\times3}(\mathbf{Z})))], \quad (10)$$

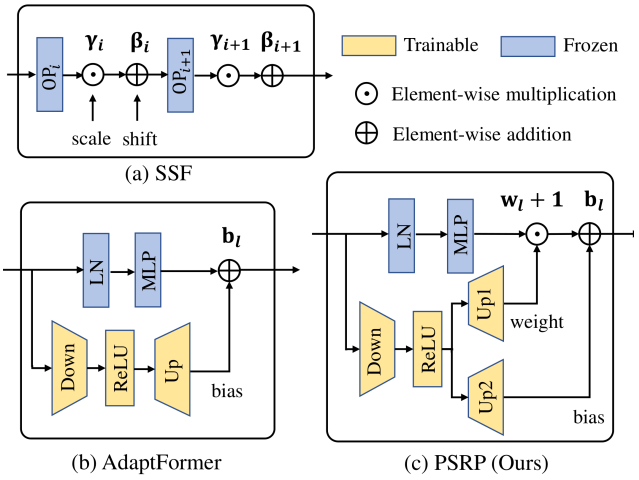
where $\mathrm{Conv}^2_{3\times3}$ indicates a 3×3 convolution with the stride of 2, while input channel and output one of convolution are $h-1$. σ is GELU non-linearity, and Concat is a concatenated operation. By using Eqn. (8)$\sim$Eqn. (10), we can compute the proposed MHC$^3(\mathbf{X})$, which results in a $(h-1)(\hat{d}/4h)^2$-dimensional cross-covariance representation. Compared with d^2-dimensional $\mathbf{M}_2 = \mathbf{X}^T\mathbf{X}$, size of our MHC$^3(\mathbf{X})$ is at least $16h$ times less than the original second-order moment. Based on the proposed MHC^3, our MP is finally computed by

$$\mathbf{y}_{pred} = \widehat{\mathbf{W}}_1\mathbf{M}_1 + \widehat{\mathbf{W}}_2\mathrm{MHC}^3(\mathbf{X}). \quad (11)$$

Usage Details of MP. We provide details on how to apply our MP for different model families. Specifically, as shown in Figure 2, for ViTs [10] we utilize the classification token (CLS_token) as the first-order moment and compute the second-order moment by using MHC^3 of the final word tokens, which constitutes our MP. For Swin transformer (Swin) [35], ConvNeXt [36] and AS-MLP [31], we perform global average pooling (GAP) of the final word tokens (i.e., Swin/AS-MLP) or the final convolution features (ConvNeXt) as the first-order moment, while these features are feed to MHC^3 for computing the second-order moment. Finally, predication scores of first- and second-order moments are summed for fusion.

3.2. PSRP for Feature Recalibration

To further explore the potential of MP, we investigate the effect of MP on learning intermediate features. To avoid tuning a number of parameters in large-scale pretrained models, recently proposed parameter-efficient methods learn a few of additional parameters to recalibrate features from frozen backbones. Particularly, as shown in Figure 3 (a), SSF [32] introduces learnable parameters $\boldsymbol{\gamma}_i \in$



(a) SSF (b) AdaptFormer (c) PSRP (Ours)

Figure 3: Comparison of (c) our PSRP with (a) SSF and (b) AdaptFormer, where SSF respectively introduces parameters $\boldsymbol{\gamma}_i$ and $\boldsymbol{\beta}_i$ for scaling and shifting features after each operation OP_i. AdaptFormer designs a tiny network to learn shifting parameters $\mathbf{b}_i$ for feature recalibration. Differently, our PSRP develops a partially shared module to learn both scaling and shifting parameters.

$\mathbb{R}^{d_i}$ and $\boldsymbol{\beta}_i \in \mathbb{R}^{d_i}$ for scaling and shifting the output from each operation $\mathrm{OP}_i(\mathbf{X}_i)$ (e.g., multi-head self-attention or feed forward network (FFN)) of pre-trained model:

$$\mathbf{Y}_i = \boldsymbol{\gamma}_i \odot \mathrm{OP}_i(\mathbf{X}_i) \oplus \boldsymbol{\beta}_i, \tag{12}$$

where d_i dimension $\mathbf{X}_i$ are inputs of i-operation OP_i and $\mathbf{Y}_i$ are recalibrated features. $\odot$ and $\oplus$ indicate element-wise multiplication and addition along dimension, respectively. As for Eqn. (12), the parameters $\boldsymbol{\gamma}_i$ and $\boldsymbol{\beta}_i$ are irrelevant to $\mathbf{X}_i$, neglecting effect of input. As shown in Figure 3 (b), AdaptFormer [6] designs an input-based tiny network to learn shifting parameters for feature recalibration in FFN, but it ignores the effect of scaling parameters. As suggested in SSF [32], both scaling and shifting operations help model transfer in downstream tasks.

Based on above analysis, we introduce two input-based tiny networks to learn both scaling and shifting parameters for feature recalibration. By considering parameter efficiency, we develop a partially shared module to learn two recalibrating parameters (PSRP). As shown in Figure 3 (c), our PSRP learns scaling ($\mathbf{w}_l$) and shifting ($\mathbf{b}_l$) parameters of l^{th}-layer outputs as

$$\begin{aligned} \mathbf{w}_l &= \mathbf{W}^1_{up}(\mathrm{ReLU}(\mathbf{W}_{down}(\mathbf{X}_l))), \\ \mathbf{b}_l &= \mathbf{W}^2_{up}(\mathrm{ReLU}(\mathbf{W}_{down}(\mathbf{X}_l))), \end{aligned} \tag{13}$$

where $\mathbf{X}_l \in \mathbb{R}^{d_l}$ are inputs of l^{th}-layer, while $\mathbf{W}^1_{up} \in \mathbb{R}^{d_l \times d_h}$, $\mathbf{W}^2_{up} \in \mathbb{R}^{d_l \times d_h}$ and $\mathbf{W}_{down} \in \mathbb{R}^{d_h \times d_l}$ are learnable parameters with hidden dimension of d_h. Finally, we obtain the recalibrated features by using $\mathbf{w}_l$ and $\mathbf{b}_l$ as

$$\mathbf{Y}_l = (\mathbf{w}_l + \mathbf{1}) \odot \mathrm{FFN}(\mathrm{LN}(\mathbf{X}_l)) \oplus \mathbf{b}_l, \tag{14}$$

where LN indicates layer normalization [1], and $\mathbf{1}$ is a vector of all ones.

4. Experiments

In this section, we first describe the experimental settings, and then compare with state-of-the-art (SOTA) methods on various downstream datasets as well as using different pre-trained models. Sec. 4.3 conducts ablation studies to assess the effect of key components on our MP. Additionally, we evaluate generalization ablity of our MP to out-of-distribution datasets, other parameter-efficient methods, pre-training strategies and few-shot setting in Sec. 4.4, Sec. 4.5 Sec. 4.6 and Sec. 4.7, respectively.

4.1. Experimental Settings

Pre-trained Backbone. In our experiments, we adopt four kinds of backbone models, including ViT [10], Swin Transformer [35], ConvNeXt [36], and AS-MLP [31]. Specifically, we first employ the pre-trained ViT-B/16, Swin-B, ConvNeXt-B, and AS-MLP to compare with SOTA methods on downstream tasks. Then, ViT-B/16 pre-trained on ImageNet-21K is used to evaluate the generalization of our MP to out-of-distribution and few-shot settings. Additionally, we employ ViT-Base/16, ViT-Large/14, ViT-Large/16 and ViT-Huge/14 for assessing generalization to different pre-training strategies (e.g., MAE [15] and CLIP [39]).

Datasets. Following the settings in SSF [32], we employ five datasets (i.e., CUB-200-2011 [45], NABirds [44], Oxford Flowers [38], Stanford Dogs [25], and Stanford Cars [12]) for fine-grained visual classification. Besides, CIFAR-100 [27] and ImageNet-1K [8] are used for general image classification. Additionally, ImageNet-A [20], ImageNet-R [18] and ImageNet-C [19] are considered to evaluate the robustness of our method to out-of-distribution.

Implementation Details. For fine-tuning models pre-trained by fully-supervised scheme [10] and CLIP [39], we follow the same settings in [10, 32]. Specifically, for data augmentation, the input images are cropped to 224×224 with a random horizontal flip for FGVC datasets, while stronger data augmentation strategies [10] are adopted for CIFAR-100 and ImageNet-1K. AdamW [37] with warmup and cosine annealing schedule of learning rate is used for network optimization. For fine-tuning models pre-trained by MAE [15], we follow its official configurations of LP on ImageNet-1K, which adopt a linear scaling rule [13]. Refer to the supplementary materials for more details. Source code will be available at **https://github.com/mingzeG/Moment-Probing**

Method \ Dataset	CIFAR-100	CUB-200 -2011	NABirds	Oxford Flowers	Stanford Dogs	Stanford Cars	Mean	Params. (M)
Linear probing	88.7	85.3	75.9	97.9	86.2	51.3	80.88	0.17
MP (Ours)	**93.8**$_{(5.1)}$	89.3$_{(4.0)}$	84.9$_{(9.0)}$	**99.6**$_{(1.7)}$	89.5$_{(3.3)}$	83.6$_{(32.3)}$	90.12$_{(9.24)}$	1.20
Adapter [21]	93.3	87.1	84.3	98.5	89.8	68.6	86.93	0.40
Bias [2]	93.4	88.4	84.2	98.8	**91.2**	79.4	89.23	0.27
VPT-Shallow [23]	90.4	86.7	78.8	98.4	**90.7**	68.7	85.62	0.25
VPT-Deep [23]	93.2	88.5	84.2	99.0	90.2	83.6	89.78	0.81
AdaptFormer [6]	93.6	88.4	84.7	99.2	88.2	81.9	89.33	0.46
SSF [32]	94.0	**89.5**	**85.7**	**99.6**	89.6	**89.2**	**91.27**	0.38
Full fine-tuning	93.8	87.3	82.7	98.8	89.4	84.5	89.42	85.96
MP$_+$ (Ours)	**94.2**	**89.9**	**86.1**	**99.7**	90.2	**89.4**	**91.58**	1.64

Table 2: Comparison of various fine-tuning methods on different downstream tasks (i.e., CIFAR-100 and FGVC datasets), where ViT-B/16 model pre-trained on ImageNet-21K is used as basic backbone.

Method \ Model	ViT-B/16	Swin-B	ConvNeXt-B	AS-MLP-B
Linear probing	82.04	83.25	84.05	79.04
MP (Ours)	83.15$_{(1.11)}$	84.62$_{(1.37)}$	85.09$_{(1.04)}$	84.03$_{(4.99)}$
Adapter [21]	82.72	83.82	84.49	88.01
Bias [2]	82.74	83.92	84.63	87.46
VPT-Shallow [23]	82.08	83.29	-	-
VPT-Deep [23]	82.45	83.44	-	-
AdaptFormer [6]	83.01	84.08	84.79	88.86
SSF [32]	83.10	84.40	84.85	88.28
Full fine-tuning	**83.58**	**85.20**	**85.80**	**89.96**
MP$_+$ (Ours)	**83.62**	**84.95**	**85.37**	**89.82**

Table 3: Comparison of various fine-tuning methods using different backbones, where ViT-B/16, Swin-B, and ConvNeXt-B are pre-trained on ImageNet-21K and fine-tuned on ImageNet-1K. AS-MLP-B is pre-trained on ImageNet-1K and fine-tuned on CIFAR-100.

4.2. Comparison with SOTA

To verify the effectiveness of our MP, we compare with several SOTA fine-tuning methods on seven downstream classification tasks. Specifically, we compare with SOTA methods on CIFAR-100 and FGVC datasets, where ViT-B/16 model pre-trained on ImageNet-21K is used as a basic backbone. Besides, we compare with SOTA methods using various backbone models on ImageNet-1K and CIFAR-100. Top-1 accuracy is used as evaluation metric while the best and second-best results for all methods are highlighted in **red** and **blue**, respectively.

Downstream Tasks. First, we compare our MP with SOTA methods on CIFAR-100 and FGVC datasets by using ViT-B/16 model pre-trained on ImageNet-21K. From Table 2 we can observe that (1) our MP consistently outperforms the original LP on all downstream tasks by a large margin, achieving 9.24% gains on average. We owe these performance gains to exploration of the stronger representations for linear classifier inherent in our MP. (2) Compared to full fine-tuning and other parameter-efficient methods, MP achieves comparable or even better performance at much lower training cost (see Figure 4), showing the potential of MP in tuning pre-trained models. (3) By considering the effect of MP on feature learning, MP$_+$ achieves further improvement and surpasses all compared tuning methods, verifying the effectiveness of MP on feature learning.

Backbone Models. Furthermore, we compare our MP with SOTA methods using four backbone models, including ViT-B/16 [10], Swin-B [35], ConvNeXt-B [36] and AS-MLP [31], which cover Vision Transformers (ViTs), deep CNNs and MLP-Mixer. Specifically, ViT-B/16, Swin-B and ConvNeXt-B are pre-trained on ImageNet-21K and fine-tuned on ImageNet-1K, while AS-MLP is pre-trained on ImageNet-1K and fine-tuned on CIFAR-100. Table 3 gives the compared results, where MP outperforms LP by more than 1% and about 5% on ImageNet-1K and CIFAR-100, respectively. Besides, MP is superior to existing parameter-efficient methods on ImageNet-1K. Since there exists large domain gap between ImageNet-1K and CIFAR-100, feature learning is necessary to achieve promising performance. As such, MP$_+$ brings 5.72% gains over MP for AS-MLP on CIFAR-100. Particularly, our MP$_+$ performs better than its counterparts, while obtaining very similar results with full fine-tuning, but it is much more parameter-efficient. These results verify the effectiveness of MP on various backbones.

Comparison of Computational Complexity. To assess computational efficiency of our MP, we compare with existing methods in terms of training time per batch and GPU memory usage for training, where ViT-B/16 [10] pre-trained on ImageNet-21K is used as a basic backbone. All models are fine-tuned with a batch size of 64 on a single NVIDIA A100 GPU. As shown in Figure 4, our MP shares similar computational cost with one of LP and it is much more efficient than other existing parameter-efficient methods, be-

Method	Params.(M)	Acc.(%)
CLS_token	0.77	82.04
GAP	0.77	79.34
GCP	4.10	80.11
MHC^3	2.88	82.10
CLS_token + GAP	1.54	81.74
CLS_token + GCP	4.86	82.54
MP	3.65	83.15

(a) Probing representations.

Dimension $\hat{d}$	Params. (M)	Acc. (%)
128	0.93	82.56
256	1.44	82.72
384	2.36	82.90
512	3.65	83.15
640	5.44	83.24
768	7.41	83.35

(b) Dimension $\hat{d}$ in MP.

Dim. d_h	AdaptMLP		PSRP	
	Params. (M)	Acc. (%)	Params. (M)	Acc. (%)
4	0.84	82.62	0.88	82.80
8	0.92	82.85	0.99	83.04
16	1.06	83.01	1.21	83.18
32	1.36	83.08	1.65	83.31
64	1.95	83.19	2.53	83.43

(c) Dimension d_h in PSRP.

Table 4: Ablation studies on MP and PSRP. Note that the results highlighted color indicate the default settings of our work.

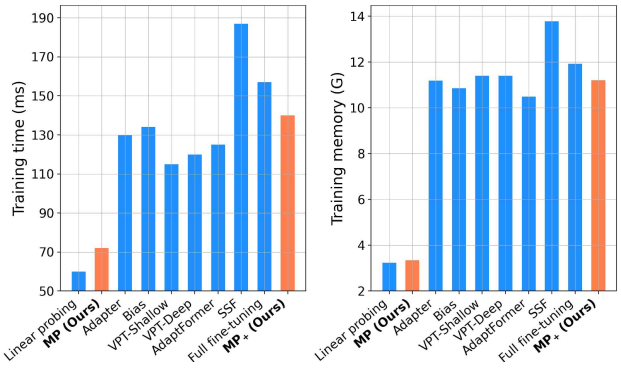


Figure 4: Comparison of computational complexity for various fine-tuning methods in terms of training time and training memory.

cause MP and LP avoid gradient computation and update for the intermediate layers. Our MP$_+$ shares similar computational cost with those of parameter-efficient methods, while being more efficient than full fine-tuning and SSF. According to above results, we can conclude that our MP provides a promising solution to achieve a good trade-off between performance and computational complexity, which is more suitable for low-cost computing resource.

4.3. Ablation Studies

In this subsection, we make ablation studies to evaluate the effect of key components, including probing representations, feature dimension d in MP and hidden dimension d_h in PSRP. Besides, we compare MHC^3 with state-of-the-art second-order representations. Here, we use ViT-B/16 pretrained on ImageNet-21K as backbone and fine-tune it on ImageNet-1K.

Comparison of Different Probing Representations. To assess effect of representations on linear probing, we compare with several representations, including classification token (CLS_token), global average pooling (GAP) of word tokens, the original global covariance pooling (GCP) of word tokens and their combinations. As shown in Table 4a, we can see that CLS_token (weighted combination of word tokens) is superior to GAP. Particularly, the original GCP brings no gain over CLS_token. In contrast, our MHC^3 is slightly superior to CLS_token, while outperforming GCP by $\sim$2% with fewer parameters. It indicates that our MHC^3 explores second-order moments for tuning pre-trained models in a more effective and efficient way. By combining CLS_token with GCP or MHC^3, the performance will further increase. However, combination of CLS_token with GAP leads to inferior performance, which may be caused by both CLS_token and GAP are first-order statistics, suffering from weak complementary. Our MP explores both first- and second-order moment to achieve the best results, performing better than second-best method (CLS_token + GCP) by 0.6% with fewer parameters. Above results clearly demonstrate the effectiveness of representations in our MP for linear probing.

Effect of Dimension on MP and PSRP. For computing our MHC^3, we first exploit a 1×1 convolution layer to reduce dimension of features $\mathbf{X}$ from d to $\hat{d}$ ($\hat{d} < d$). To evaluate effect of dimension $\hat{d}$, we experiment with MP by changing $\hat{d}$ from 128 to 768. As listed in Table 4b, performance of MP consistently increases when $\hat{d}$ becomes larger. However, larger $\hat{d}$ will involve more parameters. To balance efficiency and effectiveness, we set $\hat{d}$ to 512 throughout all experiments. Furthermore, we assess effect of hidden dimension d_h in Eqn. (13) on our PSRP, and compare with AdaptMLP [6]. As compared in Table 4c, larger d_h leads to better performance but more parameters for both AdaptMLP and PSRP. Meanwhile, our PSRP consistently outperforms AdaptMLP with few additional parameters, verifying the effectiveness of PSRP. In our work, d_h is set to 16 for efficiency and effectiveness trade-off.

Comparison of Second-order Moment To evaluate the efficiency and effectiveness of MHC^3 for computing second-order moment, we compare with two state-of-the-art second-order representations (i.e., B-CNN [34] and iSQRT-COV [28]) on ImageNet-1k, where ViT-B/16 pretrained on ImageNet-21K is used as the backbone. To achieve a good efficiency and effectiveness trade-off, we set feature dimension $\hat{d}$ to 128 for all compared second-order representa-

Method	Params.(M)	Acc.(%)
GCP	4.10	80.11
B-CNN [34]	4.10	80.38
iSQRT-COV [28]	4.10	80.39
MHC^3	**2.88**	**82.10**
CLS_token + GCP	4.86	82.54
CLS_token + B-CNN	4.86	82.68
CLS_token + iSQRT-COV	4.86	82.62
MP (Ours)	**3.65**	**83.15**

Table 5: Comparison of second-order representations on IN-1K in terms of tuning parameters (Params.) and Top-1 accuracy (Acc.).

tions. As shown in Table 5, our MHC^3 improves existing second-order representations more than 1.71% in Top-1 accuracy, while having fewer parameters. Furthermore, our MP is superior to all combinations of classification token (CLS_token) with other second-order representations in terms of both efficiency and effectiveness. These results above clearly demonstrate that MHC^3 involved in our MP is more suitable for tuning pre-trained models.

4.4. Robustness to OOD Setting

To verify the robustness of our MP, we conduct experiments on three out-of-distribution (OOD) datasets, including ImageNet-A (IN-A) [20], ImageNet-R (IN-R) [18] and ImageNet-C (IN-C) [19]. Specifically, we first fine-tune ViT-Base/16 model pre-trained on ImageNet-21K by using ImageNet-1K (IN-1K), and directly perform inference on three OOD datasets without any training. Here we compare our MP/MP$_+$ with several SOTA methods in Table 6, where we can observe that our MP improves LP over about 3%$\sim$5% on three OOD datasets. Meanwhile, MP is very competitive to existing parameter-efficient methods at lower training cost. Particularly, full fine-tuning is high-performance on IN-1K, but it shows weak generalization to OOD datasets. On the contrary, our MP$_+$ shows high-performance on both IN-1K and OOD datasets. MP$_+$ achieves the best results on IN-A and IN-C, while being comparable to the recently proposed SSF on IN-R. Above results demonstrate our MP and MP$_+$ have the ability to equip pre-trained models with stronger robustness to OOD.

4.5. Generalization to Parameter-efficient Methods

To verify the generalization of our MP, we further evaluate the effect of our MP by combining it with other parameter-efficient methods (ie, VPT [23], AdaptFormer [6], SSF [32]). Specifically, we keep the experimental settings in Sec. 4.4. As shown in Table 6, we can see that MP brings 0.86%, 0.58% and 0.46% improvement gains for

Method \ Dataset	IN-1K (↑)	IN-A (↑)	IN-R (↑)	IN-C (↓)
Linear probing	82.04	33.91	52.87	46.91
MP (Ours)	83.15$_{(1.11)}$	39.14$_{(5.23)}$	55.91$_{(3.04)}$	41.75$_{(5.16)}$
Adapter [6]	82.72	42.21	54.13	42.65
Bias [2]	82.74	42.12	55.94	41.90
VPT-Shallow [23]	82.08	30.93	53.72	46.88
VPT-Deep [23]	82.45	39.10	53.54	43.10
AdaptFormer [6]	83.01	42.96	54.45	42.35
SSF [32]	83.10	**45.88**	**56.77**	41.47
Full fine-tuning	**83.58**	34.49	51.29	46.47
MP + VPT-Deep	83.31	39.76	54.21	41.92
MP + AdaptFormer	83.43	43.59	54.89	41.33
MP + SSF	83.56	45.56	**56.95**	**41.32**
MP$_+$ (Ours)	**83.62**	**46.11**	56.67	**41.14**

Table 6: Comparison of various fine-tuning methods on out-of-distribution datasets, where Top-1 accuracy (%) is used as metric on IN-1K, IN-A and IN-R and mean corruption error is used as metric on IN-C. Note that ↑ and ↓ indicate the higher and the lower are better, respectively.

VPT-Deep, AdaptFormer, and SSF in IN-1K, respectively. Besides, MP has an ability to improve the robustness of existing parameter-efficient methods to OOD settings. These results above verify the effectiveness and complementarity of our MP to existing parameter-efficient methods. Additionally, our MP$_+$ outperforms all combinations of MP with existing parameter-efficient methods, demonstrating the superiority of our partially shared module to learn two recalibrating parameters (PSRP) over existing parameter-efficient methods.

4.6. Generalization to Pre-training Strategies

To explore the effect of different pre-training strategies, we conduct experiments by using various backbones under supervised (SUP [10]) and self-supervised (MAE [15], CLIP [39]) settings. For SUP setting, we evaluate our methods using five ViT models (i.e., ViT-Small/16, ViT-Small/32, ViT-Base/16, ViT-Base/32, ViT-Large/16) those are pre-trained on ImageNet-21K, and fine-tune them on ImageNet-1K. As shown in the left two columns of Figure 5, our MP consistently outperforms LP by a large margin. Besides, our MP obtains comparable or even better results with full fine-tuning, when model sizes increase. Particularly, MP (85.25%, 4.4M) and MP$_+$ (85.95%, 5.6M) outperform full fine-tuning (85.04%, 304.3M) by 0.21% and 0.71% with tuning much fewer parameters for ViT-L/16. For self-supervised objectives (i.e., MAE and CLIP), we use ViT-Base/16, ViT-Large/14, ViT-Large/16 and ViT-Huge/14 as basic backbones. As shown in the right two columns of Figure 5, MP still improves LP by a clear margin, while MP$_+$ brings further performance gains over MP. Since MAE aims to optimize parameters for image reconstruction, which suffers from a clear gap for classification task and may require to tune amount of parameters for per-

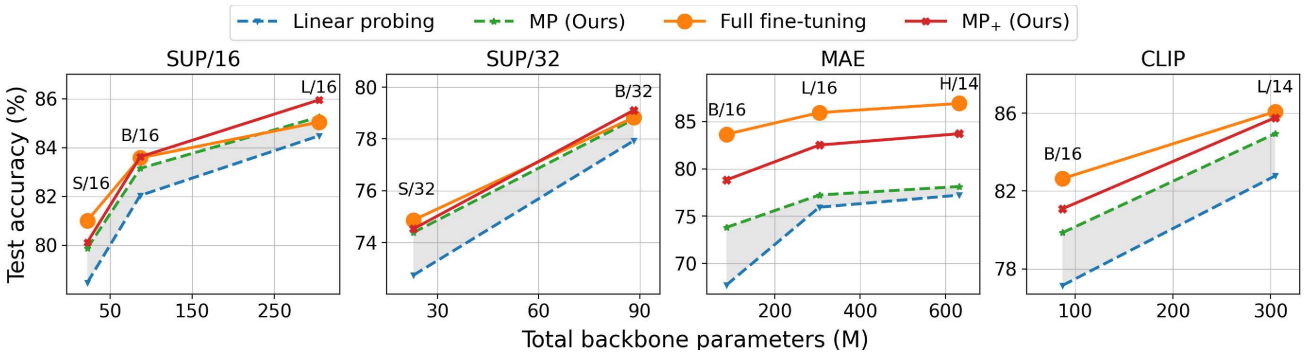


Figure 5: Comparisons across different pre-trained objectives and scales. The shadow region indicates performance gap between MP and LP. The size of markers is proportional to number of trainable parameters in log scale.

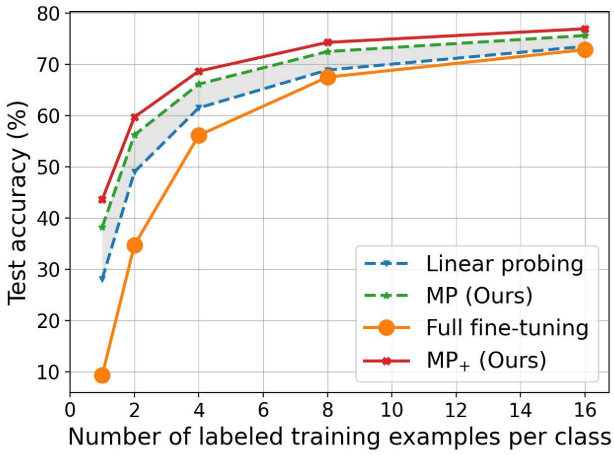


Figure 6: Comparison of various fine-tuning methods under few-shot setting. The shadow region indicates performance gap between MP and LP. The size of markers is proportional to number of trainable parameters in log scale.

forming model transfer. As such, full fine-tuning achieves the best results. For CLIP, our MP$_+$ achieves comparable results with full fine-tuning especially on large-size model, but it is more parameter-efficient. These results show our MP methods can generalize well to different pre-training strategies.

4.7. Generalization to Few-shot Setting

Finally, we evaluate the performance of our MP and MP$_+$ under few-shot setting. Following the setting in [39, 47], we conduct experiments on ImageNet-1K by selecting $\{1, 2, 4, 8, 16\}$ samples for each class as the training set, where ViT-B/16 model pre-trained on ImageNet-21K is used as the backbone. For each setting, we make three trials and report the average results in Figure 6. It can be seen that full fine-tuning is inferior to LP, which may be caused by that full fine-tuning requires to learn amount of parameters and is difficult to optimize on tiny/small training sets. Our MP is superior to LP for all cases, while MP$_+$ achieves further performance improvement by considering parameter-efficient feature learning. The comparisons above demonstrate that parameter-efficient designs for powerful representations and feature learning encourage our MP methods show good generalization to few-shot samples.

5. Conclusion

In this paper, we made an attempt to explore the potential of LP for tuning pre-trained models from the perspective of representations for linear classifiers. Particularly, we propose a Moment Probing (MP) method to feed a powerful representation characterized by feature distribution into classifier, where feature distribution is approximated by combining the original first-order moment of features with an efficient second-order moments (i.e., multi-head convolutional cross-covariance, MHC^3). Extensive experiments on various settings (e.g., FGVC, different backbones, out-of-distribution, few-shot samples and pre-training strategies) demonstrate the effectiveness and efficiency of MP on tuning pre-trained models. By introducing PSRP modules, our MP$_+$ achieves state-of-the-art performance by considering feature learning in a parameter-efficient manner. In the future, we will investigate to extend our MP in prompt learning task [49].

Acknowledgment

The work was sponsored by National Natural Science Foundation of China (Grant No.s 62276186, 61925602), and Haihe Lab of ITAI (NO. 22HHXCJC00002).

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. In *Advances in neural information processing systems*, 2016. 5

[2] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, May 2022. 1, 2, 3, 6, 8

[3] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. 2006. 3

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020. 1, 2

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2021. 1, 2

[6] Shoufa Chen, Chongjian GE, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. In *Advances in Neural Information Processing Systems*, 2022. 1, 2, 3, 5, 6, 7, 8

[7] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 1, 2

[8] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops)*, jun 2009. 1, 5

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, June 2019. 1, 2

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1, 3, 4, 5, 6, 8

[11] Yang Gao, Oscar Beijbom, Ning Zhang, and Trevor Darrell. Compact bilinear pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016. 3

[12] 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

[13] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Scott Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: training imagenet in 1 hour. *Journal of Machine Learning Research*, 18(1):6738–6764, 2017. 5

[14] 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*, 2019. 3

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1, 2, 5, 8

[16] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020. 1

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016. 1

[18] 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*, 2021. 5, 8

[19] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019. 5, 8

[20] 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*, 2021. 5, 8

[21] 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*, 2019. 1, 3, 6

[22] 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. In *International Conference on Learning Representations*, 2022. 1, 3

[23] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, 2022. 1, 3, 6, 8

[24] 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. 1, 3

[25] 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*, 2011. 5

[26] Shu Kong and Charless Fowlkes. Low-rank bilinear pooling for fine-grained classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017. 3

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

[28] Peihua Li, Jiangtao Xie, Qilong Wang, and Zilin Gao. Towards faster training of global covariance pooling networks by iterative matrix square root normalization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018. 3, 7, 8

[29] Peihua Li, Jiangtao Xie, Qilong Wang, and Wangmeng Zuo. Is second-order information helpful for large-scale visual recognition? In *Proceedings of the IEEE international conference on computer vision*, 2017. 3

[30] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Aug. 2021. 1

[31] Dongze Lian, Zehao Yu, Xing Sun, and Shenghua Gao. As-mlp: An axial shifted mlp architecture for vision. In *International Conference on Learning Representations*, 2022. 4, 5, 6

[32] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. In *Advances in Neural Information Processing Systems*, 2022. 1, 2, 3, 4, 5, 6, 8

[33] Tsung-Yu Lin and Subhransu Maji. Improved bilinear pooling with cnns. In *Proceedings of the British Machine Vision Conference*, September 2017. 3

[34] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, 2015. 3, 7, 8

[35] 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*, 2021. 1, 3, 4, 5, 6

[36] 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*, 2022. 1, 3, 4, 5, 6

[37] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 5

[38] 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*, 2008. 5

[39] 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*, 2021. 1, 2, 5, 8, 9

[40] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 1

[41] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021. 1

[42] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, 2017. 1

[43] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. In *Advances in Neural Information Processing Systems*, 2021. 1, 3

[44] 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*, June 2015. 5

[45] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5

[46] Qilong Wang, Mingze Gao, Zhaolin Zhang, Jiangtao Xie, Peihua Li, and Qinghua Hu. Dropcov: A simple yet effective method for improving deep architectures. In *Advances in Neural Information Processing Systems*, 2022. 3

[47] 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. In *European Conference on Computer Vision*, 2022. 9

[48] Xian-Da Zhang. *Modern signal processing*. Walter de Gruyter GmbH & Co KG, 2022. 4

[49] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022. 9

[50] Liu Zhuang, Lin Wayne, Shi Ya, and Zhao Jun. A robustly optimized BERT pre-training approach with post-training. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, Aug. 2021. 2