# Low-Rank Rescaled Vision Transformer Fine-Tuning: A Residual Design Approach

Wei Dong[1]*, Xing Zhang[2], Bihui Chen[2], Dawei Yan[2], Zhijun Lin[3], Qingsen Yan[3], Peng Wang[1]†, Yang Yang[1]
[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China
[2]College of Information and Control Engineering, Xi'an University of Architecture and Technology
[3]School of Computer Science, Northwestern Polytechnical University

## Abstract

*Parameter-efficient fine-tuning for pre-trained Vision Transformers aims to adeptly tailor a model to downstream tasks by learning a minimal set of new adaptation parameters while preserving the frozen majority of pre-trained parameters. Striking a balance between retaining the generalizable representation capacity of the pre-trained model and acquiring task-specific features poses a key challenge. Currently, there is a lack of focus on guiding this delicate trade-off. In this study, we approach the problem from the perspective of Singular Value Decomposition (SVD) of pre-trained parameter matrices, providing insights into the tuning dynamics of existing methods. Building upon this understanding, we propose a Residual-based Low-Rank Rescaling (RLRR) fine-tuning strategy. This strategy not only enhances flexibility in parameter tuning but also ensures that new parameters do not deviate excessively from the pre-trained model through a residual design. Extensive experiments demonstrate that our method achieves competitive performance across various downstream image classification tasks, all while maintaining comparable new parameters. We believe this work takes a step forward in offering a unified perspective for interpreting existing methods and serves as motivation for the development of new approaches that move closer to effectively considering the crucial trade-off mentioned above. Our code is available at https://github.com/zstarN70/RLRR.git.*

## 1. Introduction

In response to the remarkable capabilities demonstrated by large pre-trained models, the paradigm in computer vision and natural language processing has shifted from training task-specific models to fine-tuning a shared pre-trained model [5, 19]. Within this trajectory, Parameter-Efficient Fine-Tuning (PEFT) has emerged as an active research area, seeking to adeptly tailor a model to downstream tasks by learning a minimal set of new adaptation parameters while keeping the majority of pre-trained parameters frozen.

The central challenge of PEFT lies in efficiently adapting the pre-trained model to downstream tasks without compromising its generalization capacity. Existing work [8, 9, 12, 17] has predominantly focused on the efficient adaptation aspect of PEFT, devising various strategies to adjust pre-trained model parameters. However, less attention has been given to the crucial task of striking a balance between preserving the pre-trained model's capacity and enabling effective task adaptation. We believe that the pre-trained model inherently possesses robust generalization capabilities, and the phenomenon of prevalent low-rank strategies [4, 9, 13, 17] surpassing full fine-tuning corroborates the existence of significant redundancy within the parameter matrix tuning process. In this work, our aim is to take a step forward and explore how to achieve a better trade-off, offering a unified perspective to comprehend this critical balance.

We approach the analysis by viewing each pre-trained parameter matrix through the lens of Singular Value Decomposition (SVD), breaking down the raw matrix into a series of terms. Each term is the product of a left-singular column vector, a right-singular row vector, and a corresponding singular value. We then examine mainstream PEFT strategies such as adaptation-based methods [8], LoRA [9], prompt-tuning [12], scaling and shifting [17], using this framework. This perspective enhances our understanding of these methods, shedding light on how they tune parameters toward downstream tasks and the extent of their tuning.

Building on this analysis, we propose a low-rank rescaled fine-tuning strategy with a residual design. Our fine-tuning is formulated as a combination of a frozen matrix and a low-rank-based rescaling and shifting of the matrix. The low-rank rescaling strategy tunes the frozen matrix

*W. Dong's participation was in part supported by the Natural Science Basic Research Program of Shaanxi (Program No.2024JC-YBMS-464).

†Corresponding author: Peng Wang.

both row-wise and column-wise, providing enhanced flexibility in matrix tuning. The inclusion of the residual term proves crucial in preventing the tuned parameters from deviating excessively from the pre-trained model.

Extensive experiments demonstrate that our method achieves competitive performance across various downstream image classification tasks while maintaining comparable new parameters. The contributions of this work can be summarized as follows:

- **Unified Analytical Framework**: We introduce a unified analytical framework based on SVD to view pre-trained parameter matrices, providing a comprehensive understanding of mainstream PEFT strategies.
- **Trade-off Exploration**: Addressing a gap in existing research, we take a significant step forward by exploring the trade-off between preserving the generalization capacity of pre-trained models and efficiently adapting them to downstream tasks in PEFT.
- **Proposed Method**: We propose a novel Low-Rank Rescaled Fine-Tuning strategy with a Residual Design. This method formulates fine-tuning as a combination of a frozen matrix and a low-rank-based rescaling and shift, offering enhanced flexibility in matrix tuning.
- **Comprehensive Experiments**: Extensive experiments on various downstream image classification tasks showcase the competitiveness of our proposed method, achieving comparable performance with existing strategies while maintaining a minimal set of new parameters.

## 2. Related Work

### 2.1. Pre-training and Transfer Learning

Transfer learning, as demonstrated by various studies [11, 22, 29, 33], has proven its adaptability across diverse domains, modalities, and specific task requirements. It has significantly improved performance and convergence speed by pre-training on large-scale datasets and leveraging acquired parameters as initialization for downstream tasks. Large-scale datasets play a pivotal role in this paradigm, contributing to the performance and convergence speed of pre-trained models in downstream tasks. They endow these models with robust generalization capabilities that enhance learning efficiency. Additionally, self-supervised pre-training [2, 7] offers further benefits by mitigating costs, time, and quality issues associated with manual data labeling.

In the field of computer vision, earlier studies favor pre-training by the ImageNet-1K dataset [3] to attain quicker convergence and enhanced performance in downstream tasks. However, with the advent of larger-scale models like Vision Transformer [5] (ViT) and Swin Transformer [19], researchers have shifted toward utilizing more extensive datasets such as ImageNet-21K [3] and JFT-300M [25], for pre-training to pursue enhanced training efficiency and robustness. Nevertheless, the adoption of large-scale models presents substantial challenges due to the computational resources required during fine-tuning for downstream tasks. Consequently, researchers have begun exploring methods to achieve efficient fine-tuning.

### 2.2. Parameter-Effcient Fine-Tuning

To mitigate the computational resource challenges posed by exponential parameter growth when fine-tuning the entire network on downstream tasks, PEFT [4, 9, 12, 17] endeavors to facilitate the transition of pre-trained models to downstream tasks while significantly reducing the number of trainable parameters compared to full fine-tuning. This reduction aims to minimize training and storage expenses while addressing the risk of overfitting.

In the field of NLP, various PEFT methods have been proposed and have attained significant success [9, 10, 16, 18, 20, 32]. Adapter [8], as one of the primary fine-tuning approaches for large models, introduces a paradigm for fine-tuning through bottleneck structures, entailing the insertion of trainable adapter components into the network structure. Additionally, LoRA [9] employs low-rank decomposition to reduce parameters and treats adapters as side paths to simulate parameter matrix increments during fine-tuning. Subsequently, a multitude of PEFT methods tailored for pre-training ViT models emerged. VPT [12] employs a limited number of trainable parameters in the input and intermediate layers of ViT. It fine-tunes solely these lightweight parameters while maintaining the backbone frozen, resulting in notable performance improvements compared to full fine-tuning. SSF [17] introduces a feature modulation method that efficiently transfers features in pre-trained models by scale and shift operations. Unlike sequential adapter insertion approaches, AdaptFormer [1] explores a parallel adapter solution on ViT for various downstream tasks. FacT [13], based on a tensor decomposition framework, decomposes and reassembles parameter matrices in ViT, allowing lightweight factors to dominate the fine-tuning increment, and only updates the factors during fine-tuning for downstream tasks, resulting in lower fine-tuning costs. ARC [4] approaches fine-tuning from the perspective of the cross-layer similarity in ViT, using parameter-sharing adapter structures and independent scaling factors, offering a lesser fine-tuning cost than other methods.

### 2.3. Discussion to the Proposed Method

The proposed method incorporates a unique residual structure. Diverging from alternative parallel-structured methods, such as LoRA [9], which introduces solely low-rank learnable adaptors and can lead to challenges in fine-tuning, our approach navigates the model toward a nuanced balance between optimizing for downstream tasks and preserving the model's intrinsic representational capacity. In contrast to SSF [17], we extend our consideration to the adjustment of the singular column vector through a framework rooted in SVD, a dimension that SSF does not encompass. In summary, our study provides a cohesive perspective on past methodologies and presents compelling motivations for this specific strategy.

## 3. Methodology

In this section, we provide a comprehensive overview of the fundamental concepts related to PEFT methods. We leverage SVD to analyze the pre-trained weight matrices, delving into the underlying mechanisms of popular PEFT approaches within the SVD framework. Our scrutiny is centered on the delicate balance between retaining the generalization capacity of pre-trained parameters and facilitating task-specific adaptation. Concluding this analysis, we introduce our Residual-based Low-Rank Rescaling (RLRR) strategy, designed to optimize this trade-off for enhanced fine-tuning performance.

### 3.1. Preliminary Knowledge on PEFT Methods

ViT is a deep learning model that applies the Transformer [27] architecture to computer vision tasks like image classification, originally designed for natural language processing. The ViT model comprises two primary components: a patch embedding layer and a Transformer encoder. The patch embedding layer splits an input image $\mathbf{X} \in \mathbb{R}^{H\times W\times C}$ into a sequence of fixed-size patches, and projects each patch into a high-dimensional vector, *i.e.*, $\mathbf{X}_{\text{patches}} \in \mathbb{R}^{N\times(P^2\cdot C)}$, where $H$ and $W$ are respectively the height and width of the image resolution $(H, W)$, $(P, P)$ is the resolution of each patch, $C$ is the number of input channels, and $N = H \cdot W/P^2$ is the number of tokens. The entire patch embedding layer can be described as follows:

$$\mathbf{X}_e = [\vec{\boldsymbol{x}}_{\text{cls}}^\top; \mathbf{X}_{\text{patches}}\mathbf{W}_{\text{patches}}] + \mathbf{X}_{\text{pos}}, \tag{1}$$

where a learnable *class* token $\vec{\boldsymbol{x}}_{\text{cls}} \in \mathbb{R}^D$ is concatenated to $\mathbf{X}_{\text{patches}}\mathbf{W}_{\text{patches}}$ using a linear projection $\mathbf{W}_{\text{patches}} \in \mathbb{R}^{(P^2\cdot C)\times D}$ and the concatenation operation $[\cdot;\cdot]$. Additionally, position embeddings $\mathbf{X}_{\text{pos}} \in \mathbb{R}^{(N+1)\times D}$ are incorporated. The Transformer encoder then processes the patch embeddings using Multi-Head Attention (MHA) and Feed-Forward Network (FFN) blocks. In MHA block, Attention Head (AH) module is defined as:

$$\begin{aligned}&\text{AH}_h(\mathbf{X}^{(l-1)}) = \\ &\text{softmax}\Big(\frac{(\mathbf{X}^{(l-1)}\mathbf{W}_q^{(l)})(\mathbf{X}^{(l-1)}\mathbf{W}_k^{(l)})^\top}{\sqrt{D_h^{(l)}}}\Big)\mathbf{X}^{(l-1)}\mathbf{W}_v^{(l)},\end{aligned} \tag{2}$$

where the weight matrices $\mathbf{W}_q^{(l)} \in \mathbb{R}^{D^{(l-1)}\times D_h^{(l)}}$, $\mathbf{W}_k^{(l)} \in \mathbb{R}^{D^{(l-1)}\times D_h^{(l)}}$, and $\mathbf{W}_v^{(l)} \in \mathbb{R}^{D^{(l-1)}\times D_h^{(l)}}$ are respectively the *query*, *key*, and *value* operations with the feature dimensionality $D_h^{(l)} = \frac{D^{(l)}}{M}$ of the output of $\text{AH}_h(\cdot)$ and the number of attention heads $M$. Hence, the whole MHA block is defined as:

$$\begin{aligned}&\text{MHA}(\mathbf{X}^{(l-1)}) = \\ &\qquad [\text{AH}_1(\mathbf{X}^{(l-1)}), \cdots, \text{AH}_M(\mathbf{X}^{(l-1)})]\mathbf{W}_o^{(l)},\end{aligned} \tag{3}$$

with a linear projection $\mathbf{W}_o^{(l)} \in \mathbb{R}^{(M\cdot D_h^{(l)})\times D^{(l)}}$. We then feed the normalized output $\mathbf{X}^{(l)'}$ of the MHA block into FFN block:

$$\text{FFN}(\mathbf{X}^{(l)'}) = \text{GELU}(\mathbf{X}^{(l)'}\mathbf{W}_1^{(l)})\mathbf{W}_2^{(l)}, \tag{4}$$

where $\mathbf{W}_1^{(l)} \in \mathbb{R}^{D^{(l)}\times 4\cdot D^{(l)}}$ and $\mathbf{W}_2^{(l)} \in \mathbb{R}^{4\cdot D^{(l)}\times D^{(l)}}$ denote two linear projection matrices respectively. The whole process of $(l)$-th Transformer encoder layer is defined as:

$$\begin{aligned}\mathbf{X}^{(l)'} &= \text{MHA}(\text{LayerNorm}(\mathbf{X}^{(l-1)})) + \mathbf{X}^{(l-1)}, \\ \mathbf{X}^{(l)} &= \text{FFN}(\text{LayerNorm}(\mathbf{X}^{(l)'})) + \mathbf{X}^{(l)'},\end{aligned} \tag{5}$$

with $\text{LayerNorm}(\cdot)$ function to layer representation normalization.

In downstream tasks involving ViT and its variants, three primary types of visual PEFT methods are employed. These methods fine-tune the pre-trained model by utilizing a minimal set of new parameters, and they encompass adaptation-based, prompt-based, and scaling & shifting-based strategies. More specifically, when considering any weight matrix:

$$\mathbf{W}^{(l)} \in \{\mathbf{W}_q^{(l)}, \mathbf{W}_k^{(l)}, \mathbf{W}_v^{(l)}, \mathbf{W}_o^{(l)}, \mathbf{W}_1^{(l)}, \mathbf{W}_2^{(l)}\}, \tag{6}$$

the general idea of adaptation-based methods [8] can be defined as Eq. (7) from Table 1, in which $\text{Act}(\cdot)$ is the activation function, $\vec{\boldsymbol{b}}^{(l)}$ is the bias weights, and $\mathbf{W}_{\text{down}} \in \mathbb{R}^{D^{(l)}\times D^{'}}$ and $\mathbf{W}_{\text{up}} \in \mathbb{R}^{D^{'}\times D^{(l)}}$ are down- and up-adapting projection matrices across different layers with the dimensionality $D^{(l)} \gg D^{'}$. A prominent example of an adaptation-based method is Low-Rank Adaptation (LoRA)[9], which can be expressed as Eq.(9) in Table 1.

The second type comprises prompt-based methods [12], represented by Eq.(11), where $\boldsymbol{\Theta}^{(l-1)} \in \mathbb{R}^{T\times D^{(l-1)}}$ constitutes learnable parameters with $T$ virtual tokens. Finally, the third type encompasses scaling & shifting-based strategies, illustrated by Eq.(13), featuring learnable scaling parameters $\vec{\boldsymbol{s}}^{(l)}$, shifting parameters $\vec{\boldsymbol{f}}^{(l)}$, and element-wise Hadamard product denoted by $\odot$.

Table 1. Examples of PEFT methods and their interpretation under the SVD framework.

| Visual PEFT Method | Strategy | Spectral Analysis |
|---|---|---|
| Adaptation-based [8] | $\mathbf{X}_{\text{FT}}^{(l-1)} = \text{Act}\left(\left(\mathbf{X}^{(l-1)}\mathbf{W}^{(l)} + \vec{\boldsymbol{b}}^{(l)\top}\right)\mathbf{W}_{\text{down}}\right)\mathbf{W}_{\text{up}}, \quad (7)$ | $\begin{aligned}\mathbf{X}_{\text{FT}}^{(l-1)} &= \text{Act}\left(\left(\mathbf{X}^{(l-1)}\left(\lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots + \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}\right) + \vec{\boldsymbol{b}}^{(l)\top}\right)\mathbf{W}_{\text{down}}\right)\mathbf{W}_{\text{up}} \\ &= \text{Act}\left(\mathbf{X}^{(l-1)}\left(\lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top}\mathbf{W}_{\text{down}} + \cdots + \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}\mathbf{W}_{\text{down}}\right) + \vec{\boldsymbol{b}}^{(l)\top}\mathbf{W}_{\text{down}}\right)\mathbf{W}_{\text{up}} \qquad (8) \\ &= \text{Act}\left(\mathbf{X}^{(l-1)}\left(\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top}\lambda_1^{(l)}\mathbf{W}_{\text{down}} + \cdots + \vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}\lambda_D^{(l)}\mathbf{W}_{\text{down}}\right) + \vec{\boldsymbol{b}}^{(l)\top}\mathbf{W}_{\text{down}}\right)\mathbf{W}_{\text{up}},\end{aligned}$ |
| LoRA adaptation [9] | $\mathbf{X}_{\text{FT}}^{(l-1)} = \mathbf{X}^{(l-1)}(\mathbf{W}^{(l)} + \mathbf{W}_{\text{down}}\mathbf{W}_{\text{up}}) + \vec{\boldsymbol{b}}^{(l)\top}, \quad (9)$ | $\mathbf{X}_{\text{FT}}^{(l-1)} = \mathbf{X}^{(l-1)}\left(\lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots + \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top} + \mathbf{W}_{\text{down}}\mathbf{W}_{\text{up}}\right) + \vec{\boldsymbol{b}}^{(l)\top}, \qquad (10)$ |
| Prompt-based [12] | $\mathbf{X}_{\text{FT}}^{(l-1)} = \begin{pmatrix}\mathbf{X}^{(l-1)} \\ \boldsymbol{\Theta}^{(l-1)}\end{pmatrix}\mathbf{W}^{(l)} + \vec{\boldsymbol{b}}^{(l)\top}, \quad (11)$ | $\begin{aligned}\mathbf{X}_{\text{FT}}^{(l-1)} &= \begin{pmatrix}\mathbf{X}^{(l-1)} \\ \boldsymbol{\Theta}^{(l-1)}\end{pmatrix}\left(\lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots + \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}\right) + \vec{\boldsymbol{b}}^{(l)\top} \qquad (12) \\ &= \begin{pmatrix}\lambda_1^{(l)}\mathbf{X}^{(l-1)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots + \lambda_D^{(l)}\mathbf{X}^{(l-1)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top} \\ \lambda_1^{(l)}\boldsymbol{\Theta}^{(l-1)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots + \lambda_D^{(l)}\boldsymbol{\Theta}^{(l-1)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}\end{pmatrix} + \vec{\boldsymbol{b}}^{(l)\top},\end{aligned}$ |
| scaling&shifting-based [17] | $\mathbf{X}_{\text{FT}}^{(l-1)} = \left(\mathbf{X}\mathbf{W}^{(l)} + \vec{\boldsymbol{b}}^{(l)\top}\right) \odot \vec{\boldsymbol{s}}^{(l)\top} + \vec{\boldsymbol{f}}^{(l)\top}, \quad (13)$ | $\begin{aligned}\mathbf{X}_{\text{FT}}^{(l-1)} &= \left(\mathbf{X}\left(\lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots + \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}\right) + \vec{\boldsymbol{b}}^{(l)\top}\right) \odot \vec{\boldsymbol{s}}^{(l)\top} + \vec{\boldsymbol{f}}^{(l)\top} \\ &= \mathbf{X}\left(\lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} \odot \vec{\boldsymbol{s}}^{(l)\top} + \cdots + \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top} \odot \vec{\boldsymbol{s}}^{(l)\top}\right) + \vec{\boldsymbol{b}}^{(l)\top} \odot \vec{\boldsymbol{s}}^{(l)\top} + \vec{\boldsymbol{f}}^{(l)\top} \qquad (14) \\ &= \mathbf{X}\left(\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top}\lambda_1^{(l)} \odot \vec{\boldsymbol{s}}^{(l)\top} + \cdots + \vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}\lambda_D^{(l)} \odot \vec{\boldsymbol{s}}^{(l)\top}\right) + \vec{\boldsymbol{b}}^{(l)\top} \odot \vec{\boldsymbol{s}}^{(l)\top} + \vec{\boldsymbol{f}}^{(l)\top},\end{aligned}$ |

## 3.2. Revisiting Existing PEFT Methods through singular value decomposition

In this section, we revisit the working mechanisms of existing PEFT methods mentioned above through the lens of SVD. Our goal is to establish a unified framework for understanding the delicate trade-off between retaining the generalization capacity of the pre-trained model and facilitating task-specific adaptation. We initiate our exploration by employing SVD to decompose the weight matrix $\mathbf{W}^{(l)}$ to:

$$\mathbf{W}^{(l)} = \lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots + \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}, \tag{15}$$

with the spectrum (*i.e.* singular values) $\{\lambda_d^{(l)}\}$, the left singular vector $\vec{\boldsymbol{d}}_d^{(l)}$ coming from the left unitary matrix, and the right singular vector $\vec{\boldsymbol{u}}_d^{(l)\top}$ coming from the right unitary matrix. Under this SVD framework, Eqs. (7), (9), (11), and (13) can be rewritten as Eqs. (8), (10), (12), and (14) in Table 1.

Upon examining these redefined equations, it becomes evident that general adaptation-based methods involve each singular item under the spectrum, denoted as $\lambda_d^{(l)}\vec{\boldsymbol{d}}_d^{(l)}\vec{\boldsymbol{u}}_d^{(l)\top}\mathbf{W}_{\text{down}}$. The down-adapting projection matrix $\mathbf{W}_{\text{down}}$ is directly applied to the right singular vector $\vec{\boldsymbol{u}}_d^{(l)}$. However, this direct application compromises the spatial structure, including the orthogonality of these right singular matrix $[\vec{\boldsymbol{u}}_1^{(l)}, \vec{\boldsymbol{u}}_2^{(l)}, \ldots, \vec{\boldsymbol{u}}_d^{(l)}]^\top$, thereby affecting the representation capacity of the pre-trained model. Similarly, in prompt-based methods, the learnable tokens $\boldsymbol{\Theta}$ directly interat with the left singular vector $\vec{\boldsymbol{d}}_d^{(l)}$ in Eq. (12). However, this direct interaction has the potential to excessively influence the tuning, deviating significantly from the pre-trained model. Scaling&shifting-based methods has the same defect due to the element-wise multiplication $\vec{\boldsymbol{u}}_d^{(l)\top} \odot \vec{\boldsymbol{s}}^{(l)\top}$ in Eq. (14). Additionally, over-adaptation may perturb the spectrum, affecting one side of the weight capacity. Specifically, $\lambda_d^{(l)}\mathbf{W}_{\text{down}}$ in Eq. (8), $\lambda_d^{(l)}\boldsymbol{\Theta}^{(l-1)}$ in Eq. (12), and $\lambda_d^{(l)} \odot \vec{\boldsymbol{s}}^{(l)\top}$ in Eq. (14) demonstrate the impact to the singular spectrum. Improper initialization of parameters, such as $\mathbf{W}_{\text{down}}$, $\boldsymbol{\Theta}^{(l-1)}$, and $\vec{\boldsymbol{s}}^{(l)}$, can lead to spectrum distortion and the loss of the original weight capacity.

In contrast, from Eq. (10), we observe that the sole low-rank adaption item $\mathbf{W}_{\text{down}}\mathbf{W}_{\text{up}}$ of LoRA method adapts weakly to each of all singular items $\{\lambda_d^{(l)}\vec{\boldsymbol{d}}_d^{(l)}\vec{\boldsymbol{u}}_d^{(l)\top}\}$ when the dimensionality $D$ of the weight matrix $\mathbf{W}^{(l)}$ is large. This slight perturbation may marginally change the weight spectrum and singular vectors in which the representation capacity of the pre-trained model can not be smoothly adapted to downstream tasks.

## 3.3. Residual-based Low-Rank Rescaling (RLRR) Method

To balance the trade-off between over-adaptation and under-adaptation in downstream tasks, we propose a simple yet effective method, namely, the RLRR strategy as shown in Fig. 1. It can be derived from the aforementioned unified framework:

$$\begin{aligned}\mathbf{X}_{\text{FT}}^{(l-1)} &= \mathbf{X}^{(l-1)}(\mathbf{W}^{(l)} + \triangle\mathbf{W}^{(l)}) + \vec{\boldsymbol{b}}^{(l)\top} + \vec{\boldsymbol{f}}^{(l)\top} \\ &= \mathbf{X}^{(l-1)}(\mathbf{W}^{(l)} + \vec{\boldsymbol{s}}_{\text{left}}^{(l)} \odot \mathbf{W}^{(l)} \odot \vec{\boldsymbol{s}}_{\text{right}}^{(l)\top}) \\ &\quad + \vec{\boldsymbol{b}}^{(l)\top} + \vec{\boldsymbol{f}}^{(l)\top},\end{aligned} \tag{16}$$

in which we add scales $\vec{\boldsymbol{s}}_{\text{left}}^{(l)}$ and $\vec{\boldsymbol{s}}_{\text{right}}^{(l)}$ to both side of weight matrix $\mathbf{W}^{(l)}$, making it more flexible compared to SSF [17] when learning the features of downstream tasks.

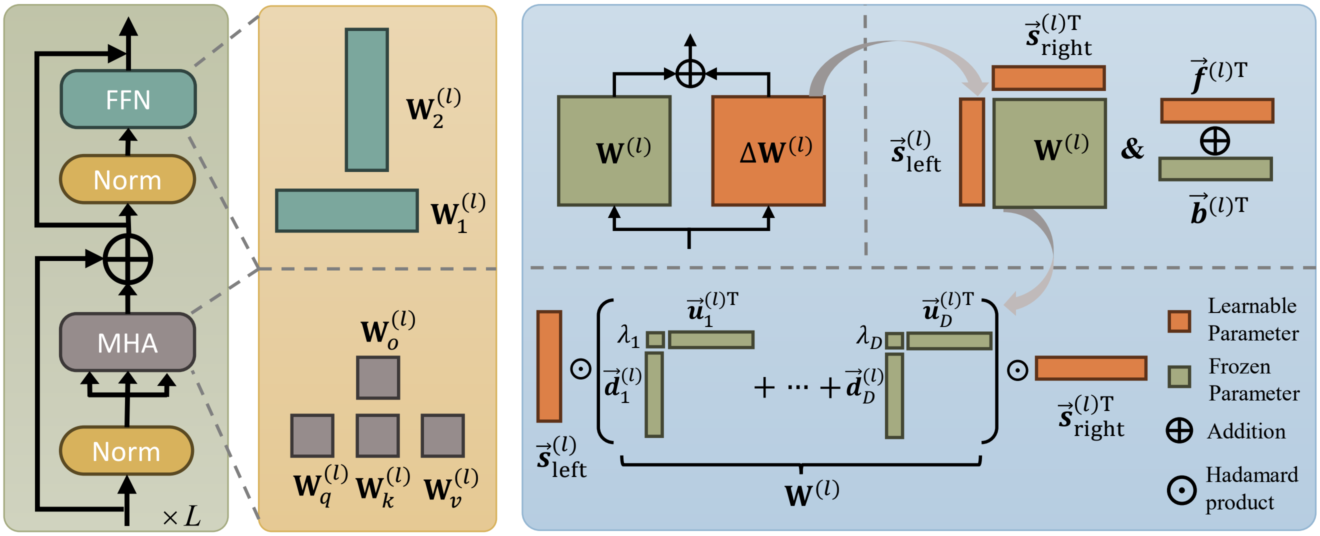
FFN
Norm
MHA
Norm
$\times L$
$\mathbf{W}_2^{(l)}$
$\mathbf{W}_1^{(l)}$
$\mathbf{W}_o^{(l)}$
$\mathbf{W}_q^{(l)}$ $\mathbf{W}_k^{(l)}$ $\mathbf{W}_v^{(l)}$
$\mathbf{W}^{(l)}$
$\Delta\mathbf{W}^{(l)}$
$\vec{\boldsymbol{s}}_{\text{right}}^{(l)\text{T}}$
$\vec{\boldsymbol{s}}_{\text{left}}^{(l)}$
$\mathbf{W}^{(l)}$
&
$\vec{\boldsymbol{f}}^{(l)\text{T}}$
$\vec{\boldsymbol{b}}^{(l)\text{T}}$
$\lambda_1$
$\vec{\boldsymbol{u}}_1^{(l)\text{T}}$
$\vec{\boldsymbol{d}}_1^{(l)}$
$+\cdots+$
$\lambda_D$
$\vec{\boldsymbol{u}}_D^{(l)\text{T}}$
$\vec{\boldsymbol{d}}_D^{(l)}$
$\vec{\boldsymbol{s}}_{\text{right}}^{(l)\text{T}}$
$\vec{\boldsymbol{s}}_{\text{left}}^{(l)}$
$\mathbf{W}^{(l)}$
Learnable Parameter
Frozen Parameter
Addition
Hadamard product

Figure 1. Illustration of the proposed RLRR method. For any weight matrix $\mathbf{W}^{(l)}$ in the MHA and FFN modules, we fine-tune the frozen pre-training parameter matrix using a residual structure. This involves combining the frozen matrix with a low-rank-based scaling and shifting operation *i.e.*, $\triangle\mathbf{W}^{(l)}$. From the perspective of SVD, scaling vectors $\vec{\boldsymbol{s}}_{\text{left}}^{(l)}$ and $\vec{\boldsymbol{s}}_{\text{right}}^{(l)}$ and shifting vector $\vec{\boldsymbol{f}}_{(l)}$ can also be interpreted as adjustments to the rows and columns of the pre-training matrix $\mathbf{W}^{(l)}$.

In Eq. (16), we also add the frozen weights $\mathbf{W}^{(l)}$ to the fine-tuning item $\triangle\mathbf{W}^{(l)} = \vec{\boldsymbol{s}}_{\text{left}}^{(l)} \odot \mathbf{W}^{(l)} \odot \vec{\boldsymbol{s}}_{\text{right}}^{(l)\top}$ with learnable parameters $\vec{\boldsymbol{s}}_{\text{left}}^{(l)}$, $\vec{\boldsymbol{s}}_{\text{right}}^{(l)}$, and $\vec{\boldsymbol{f}}^{(l)}$. By doing this, RLRR strategy can trade off the over- and under-adaption. Concretely, we expand Eq. (16) to:

$$\begin{aligned}\mathbf{X}_{\text{FT}}^{(l-1)} =& \mathbf{X}^{(l-1)}(\lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots + \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top} \\ &+ \vec{\boldsymbol{s}}_{\text{left}}^{(l)} \odot (\lambda_1^{(l)}\vec{\boldsymbol{d}}_1^{(l)}\vec{\boldsymbol{u}}_1^{(l)\top} + \cdots \\ &+ \lambda_D^{(l)}\vec{\boldsymbol{d}}_D^{(l)}\vec{\boldsymbol{u}}_D^{(l)\top}) \odot \vec{\boldsymbol{s}}_{\text{right}}^{(l)\top}) + \vec{\boldsymbol{b}}^{(l)\top} + \vec{\boldsymbol{f}}^{(l)\top},\end{aligned} \tag{17}$$

in which we get the singular item:

$$\begin{aligned}&\mathbf{W}_{\text{item}} = \\ &\lambda_d^{(l)}\vec{\boldsymbol{d}}_d^{(l)}\vec{\boldsymbol{u}}_d^{(l)\top} + \lambda_d^{(l)}\vec{\boldsymbol{s}}_{\text{left}}^{(l)} \odot \vec{\boldsymbol{d}}_d^{(l)}\vec{\boldsymbol{u}}_d^{(l)\top} \odot \vec{\boldsymbol{s}}_{\text{right}}^{(l)\top},\end{aligned} \tag{18}$$

and each element therein is:

$$\begin{aligned}&\mathbf{W}_{\text{item}[i,j]} = \lambda_d^{(l)}\vec{\boldsymbol{d}}_{d[i]}^{(l)}\vec{\boldsymbol{u}}_{d[j]}^{(l)} + \lambda_d^{(l)}\vec{\boldsymbol{s}}_{\text{left}[i]}^{(l)}\vec{\boldsymbol{d}}_{d[i]}^{(l)}\vec{\boldsymbol{u}}_{d[j]}^{(l)}\vec{\boldsymbol{s}}_{\text{right}[j]}^{(l)} \\ &= (1 + \vec{\boldsymbol{s}}_{\text{left}[i]}^{(l)}\vec{\boldsymbol{s}}_{\text{right}[j]}^{(l)})\lambda_d^{(l)}\vec{\boldsymbol{d}}_{d[i]}^{(l)}\vec{\boldsymbol{u}}_{d[j]}^{(l)}.\end{aligned} \tag{19}$$

There is a constant term 1 in Eq. (19) that can fix the intrinsical representation capacity to the pre-trained model and meanwhile leverage the fine-tuning item $\vec{\boldsymbol{s}}_{\text{left}[i]}^{(l)}\vec{\boldsymbol{s}}_{\text{right}[j]}^{(l)}$ to adaptively adjust such model capacity to learn the downstream tasks.

**Re-parameterization**. Similar to previous methods [17], our adjustments to the parameter matrices are linear operations. This allows us to seamlessly absorb the scaling and shifting operations into the original parameter matrices by re-parameterizing as follows:

$$\begin{aligned}\mathbf{W}_{\text{re-param}}^{(l)} &= \mathbf{W}^{(l)} + \Delta\mathbf{W}^{(l)} \\ &= \left(\mathbf{1} + \vec{\boldsymbol{s}}_{\text{left}}^{(l)}\vec{\boldsymbol{s}}_{\text{right}}^{(l)\top}\right) \odot \mathbf{W}^{(l)}, \\ \vec{\boldsymbol{b}}_{\text{re-param}}^{(l)} &= \vec{\boldsymbol{b}}^{(l)} + \vec{\boldsymbol{f}}^{(l)},\end{aligned} \tag{20}$$

where $\mathbf{1}$ denotes a matrix involving all elements to 1, with its dimensions consistent with the original parameter matrix $\mathbf{W}^{(l)}$ in the $(l)$-th layer. The vectors $\vec{\boldsymbol{s}}_{\text{left}}^{(l)}$ and $\vec{\boldsymbol{s}}_{\text{right}}^{(l)}$ denote the scaling parameters and the shifting parameters is the $\vec{\boldsymbol{f}}$ vector. Eq. (20) implies that we can merge $\vec{\boldsymbol{s}}_{\text{left}}^{(l)}$, $\vec{\boldsymbol{s}}_{\text{right}}^{(l)}$, and $\vec{\boldsymbol{f}}^{(l)}$ into the original parameter matrix $\mathbf{W}^{(l)}$ by linear operations without requiring extra storage space during inference.

## 4. Experiments

### 4.1. Experimental Settings

**Downstream Tasks**. Following the previous works [4, 12, 17], we evaluate RLRR on a collection of five Fine-Grained Visual Classification (FGVC) datasets and the VTAB-1k benchmark. FGVC consists of *CUB-200-2011* [28], *NABirds* [26], *Oxford Flowers* [21], *Stanford Dogs* [14], and *Stanford Cars* [6]. We follow the data partitioning scheme established in VPT [12] to maintain consistency. VTAB-1k [31] is a benchmark that contains 19 diverse visual classification tasks, which are divided into three groups: *Natural*, *Specialized*, and *Structured*. *Natural* group corresponds

Table 2. Performance comparison of RLRR with the baseline and state-of-the-art efficient adaptive methods on the VTAB-1k benchmark. All methods leverage ViT-B/16 pre-trained on ImageNet-21k as the backbone. Furthermore, SSF, ARC*, and RLRR* utilize the augmented ViT backbone by AugReg [24]. Bold font denotes state-of-the-art performance, while underlined results indicate sub-optimal performance.

| **Datasets**<br>**Methods** | **Natural** | | | | | | | | **Specialized** | | | | | **Structed** | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **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 | 85.80 |
| 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 [8] | 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 [30] | 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 [12] | 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 [12] | **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 [9] | 67.1 | 91.4 | 69.4 | 98.8 | 90.4 | 85.3 | 54.0 | 79.5 | 84.9 | 95.3 | 84.4 | 73.6 | 84.6 | **82.9** | **69.2** | 49.8 | 78.5 | 75.7 | 47.1 | 31.0 | **44.0** | 59.8 | 72.3 | 0.29 |
| AdaptFormer [1] | 70.8 | 91.2 | 70.5 | 99.1 | 90.9 | 86.6 | 54.8 | 80.6 | 83.0 | 95.8 | 84.4 | **76.3** | 84.9 | 81.9 | 64.3 | 49.3 | 80.3 | 76.3 | 45.7 | 31.7 | 41.1 | 58.8 | 72.3 | 0.16 |
| FacT-TK$_{\leq 32}$[13] | 70.6 | 90.6 | 70.8 | 99.1 | 90.7 | 88.6 | 54.1 | 80.6 | 84.8 | **96.2** | 84.5 | 75.7 | 85.3 | 82.6 | 68.2 | 49.8 | 80.7 | 80.8 | 47.4 | 33.2 | 43.0 | 60.7 | 73.2 | 0.07 |
| ARC [4] | 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 |
| RLRR | 75.6 | **92.4** | **72.9** | **99.3** | **91.5** | 89.8 | **57.0** | **82.7** | **86.8** | 95.2 | 85.3 | 75.9 | **85.8** | 79.7 | 64.2 | **53.9** | **82.1** | **83.9** | **53.7** | **33.4** | 43.6 | **61.8** | **74.5** | 0.33 |
| SSF [17] | 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 |
| ARC* [4] | 71.2 | 90.9 | 75.9 | 99.5 | 92.1 | 90.8 | 52.0 | 81.8 | 87.4 | **96.5** | 87.6 | **76.4** | 87.0 | **83.3** | 61.1 | **54.6** | 81.7 | 81.0 | **57.0** | 30.9 | 41.3 | 61.4 | 74.3 | 0.13 |
| RLRR* | **76.7** | **92.7** | **76.3** | **99.6** | **92.6** | **91.8** | **56.0** | **83.7** | **87.8** | 96.2 | **89.1** | 76.3 | **87.3** | 80.4 | **63.3** | 54.5 | **83.3** | **83.0** | 53.7 | **32.0** | **41.7** | **61.5** | **75.1** | 0.33 |

to images from daily life, *Specialized* group includes medical and remote sensing images captured by specialized devices, and *Structured* group contains synthetic images from simulated environments. Each task contains only 1000 images for training, covering various potential downstream tasks such as classification, object counting, and depth estimation. Consequently, it serves as a comprehensive measurement for evaluating the efficacy of fine-tuning methodologies.

**Pre-trained Backbones**. We employ ViT [5] and Swin Transformer [19] as backbones to evaluate our approach. Furthermore, we employ three different variants of ViT (*i.e.* ViT-Base, ViT-Large, ViT-Huge) to demonstrate the versatility of RLRR. All of these backbone architectures leverage parameters pre-trained on the ImageNet21K dataset [3], preserving the default configurations, which include the number of image patches and the dimensions of the features in the hidden layers. Moreover, we note that the SSF [17] employs a ViT backbone that is augmented with AugReg [24]. To guarantee a fair comparison, we have carried out independent experiments with this augmentation strategy, as presented in Table 2 and Table 3.

**Baselines and Existing PEFT methods**. We evaluate the performance of RLRR by comparing it with two baseline methods and several well-known PEFT approaches including Adapter [8], Bias [30], LoRA [9], VPT [12], AdaptFormer [1], FacT [13] and ARC [4]. The two baseline methods are (1) Full Fine-tuning, which updates all parameters of the pre-trained model using the training data from the downstream task, and (2) Linear Probing, which involves training only the linear classification head for the downstream task while keeping the rest of the pre-trained parameters frozen.

**Implementation Details**. In this work, we implement standard data augmentation following VPT [12] during the training phase. For five FGVC datasets, we apply random horizontal flips and randomly resize crop to 224 × 224 pixels. For the VTAB-1k benchmark, images are resized to 224 × 224 pixels, and we employ random horizontal flips on the 19 datasets. We conduct a grid search to optimize hyper-parameters specific to tuning, such as learning rate and weight decay. All experiments are conducted using the PyTorch framework [23] on an NVIDIA A800 GPU with 80 GB of memory.

## 4.2. Experimental Comparisons

In this section, we conduct a comprehensive comparison of our RLRR method with baseline models and other state-of-the-art approaches using two sets of visual adaptation benchmarks. We evaluate the classification accuracy of each method across a range of downstream tasks and examine the number of trainable parameters during the fine-tuning phase. The outcomes of these evaluations are detailed in Table 2 and Table 3. Based on the findings, we make the following observations:

(1) RLRR approach yields results that are competitive with both baseline methods and prior state-of-the-art PEFT methods. Notably, RLRR attains superior performance on the majority of datasets across two visual adaptation benchmarks, outperforming most existing fine-tuning approaches. It also maintains a competitive number of trainable parameters, suggesting that RLRR achieves high efficiency without incurring excessive computational costs. In particular, on the VTAB-1k benchmark, our method excels in more than half of the 19 datasets, achieving a 1.1% improvement (74.5% *vs.* 73.4%) over the plain pre-trained model and a 0.8% increase (75.1% *vs.* 74.3%) over the AugReg-enhanced model relative to the latest PEFT methods. Moreover, RLRR demonstrates optimal performance in 7 out of 10 assessments on the FGVC dataset using two versions of

Table 3. Performance comparison of RLRR with baseline and state-of-the-art PEFT methods on five FGVC datasets. All experiments use ViT-B/16 pretrained on ImageNet-21k as the backbone. SSF, ARC*, and RLRR* leverage the augmented ViT backbone by AugReg [24].

| Methods \ Datasets | CUB-200-2011 | NABirds | Oxford Flowers | Stanford Dogs | Stanford Cars | Mean Total | 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 | 0.18 |
| Adapter [8] | 87.1 | 84.3 | 98.5 | 89.8 | 68.6 | 85.7 | 0.41 |
| Bias [30] | 88.4 | 84.2 | 98.8 | 91.2 | 79.4 | 88.4 | 0.28 |
| VPT-Shallow [12] | 86.7 | 78.8 | 98.4 | 90.7 | 68.7 | 84.6 | 0.25 |
| VPT-Deep [12] | 88.5 | 84.2 | 99.0 | 90.2 | 83.6 | 89.1 | 0.85 |
| LoRA [9] | 88.3 | **85.6** | 99.2 | 91.0 | 83.2 | 89.5 | 0.44 |
| ARC [4] | 88.5 | 85.3 | 99.3 | 91.9 | 85.7 | 90.1 | 0.25 |
| RLRR | **89.3** | 84.7 | **99.5** | **92.0** | **87.0** | **90.4** | 0.47 |
| SSF [17] | 89.5 | **85.7** | 99.6 | 89.6 | 89.2 | 90.7 | 0.39 |
| ARC* [4] | 89.3 | **85.7** | **99.7** | 89.1 | 89.5 | 90.7 | 0.25 |
| RLRR* | **89.8** | 85.3 | 99.6 | **90.0** | **90.4** | **91.0** | 0.47 |

Table 4. Performance comparison on VTAB-1k using VIT-Large and VIT-Huge pre-trained on ImageNet-21k as the backbone. "(·)" indicates the number of tasks in the subgroup. Detailed results are presented in the Appendix.

| Methods \ Datasets | (a) ViT-Large | | | | | (b) ViT-Huge | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Natural (7) | Specialized (4) | Structed (8) | Mean | Params.(M) | Natural (7) | Specialized (4) | Structed (8) | Mean | Params.(M) |
| Full fine-tuning | 74.7 | 83.8 | 48.1 | 65.4 | 303.40 | 70.9 | 83.6 | 46.0 | 63.1 | 630.90 |
| Linear probing | 70.9 | 69.1 | 25.8 | 51.5 | 0.05 | 67.9 | 79.0 | 26.1 | 52.7 | 0.06 |
| Adapter [8] | 68.6 | 73.5 | 29.0 | 52.9 | 2.38 | 68.1 | 76.4 | 24.5 | 51.5 | 5.78 |
| Bias [30] | 70.5 | 73.8 | 41.2 | 58.9 | 0.32 | 70.3 | 78.9 | 41.7 | 60.1 | 0.52 |
| VPT-Shallow [12] | 78.7 | 79.9 | 40.6 | 62.9 | 0.15 | 74.8 | 81.2 | 43.0 | 62.8 | 0.18 |
| VPT-Deep [12] | 82.5 | 83.9 | 54.1 | 70.8 | 0.49 | 77.9 | 83.3 | 52.2 | 68.2 | 0.96 |
| LoRA [9] | 81.4 | 85.0 | 57.3 | 72.0 | 0.74 | 77.1 | 83.5 | 55.4 | 69.3 | 1.21 |
| SSF [17] | 81.9 | 85.2 | 59.0 | 73.0 | 0.60 | 79.0 | 83.1 | 56.6 | 70.4 | 0.97 |
| ARC [4] | 82.3 | 85.6 | 57.3 | 72.5 | 0.18 | 79.1 | 84.8 | 53.7 | 69.6 | 0.22 |
| RLRR | **83.9** | **86.4** | **61.9** | **75.2** | 0.82 | **79.4** | **85.1** | **59.0** | **72.0** | 1.33 |

pre-trained model, underscoring its consistent adaptability and robustness across varied downstream tasks.

Additionally, it is noteworthy that RLRR and LoRA have comparable parameter counts. However, RLRR significantly outperforms LoRA in downstream tasks. This underscores the superior design of RLRR, which leverages the pre-trained parameter matrix as the foundation for the residual term, thus preventing the potential pitfalls of over or under adaptation in downstream tasks. The tuning of the residual term also benefits from the high-efficiency parameter adjustment inherent in the well-structured design of SSF. Furthermore, our approach incorporates a rescaling weight matrix design, which provides greater flexibility than that of SSF. In contrast to VPT, our method obviates the need for designing complex, task-specific trainable parameters or for intricate injection selections within the partial modules of ViT, thereby avoiding additional computational overhead.

(2) In contrast to PEFT solutions, Full fine-tuning does not yield significant improvements. In fact, performance can decline even with the increase in the number of updated parameters. We attribute this to the loss of the generalization ability of the pre-trained model, which was acquired from large-scale datasets, leading to overfitting on the training set for downstream tasks. In practice, as a commonly adopted strategy in transfer learning, full fine-tuning necessitates extensive data and meticulous experimental setups to prevent overfitting. Especially on the VTAB-1k benchmark with only 1000 images for training, besides fine-tuning the entire model, numerous adaptation methods often find themselves in the dilemma of overfitting. This underscores the effectiveness and promise of lightweight adaptation designs.

**Experiments on larger-scale ViT backbones.** Beyond the commonly employed ViT-B/16 backbone for evaluations, we expand our experiments to include larger backbones, ViT-L/16 and ViT-H/14, to verify the scalability and generalizability of our RLRR method. As indicated in Tables 4 (a) and (b), RLRR consistently outperforms other state-of-the-art adaptation methods, maintaining exceptional performance even when applied to these larger-scale backbones. Specifically, our method surpasses the latest state-of-the-art by 2.7% on the ViT-L/16 and by 2.6% on the ViT-H/14 backbones. These findings demonstrate the capability of RLRR to effectively scale to larger models, confirming its robustness for efficient adaptation across diverse Transformer-based architectures.

**Experiments on hierarchical Vision Transformers.** To further demonstrate the efficacy of RLRR, we apply it to the Swin Transformer [19], a Transformer-based architecture distinguished by its hierarchical structure. The Swin

Table 5. Performance comparison on VTAB-1k using Swin Transformer pre-trained on ImageNet-21k as the backbone. ”(·)” indicates the number of tasks in the subgroup. Detailed results are presented in the Appendix.

| Datasets / Methods | Natural (7) | Specialized (4) | Structed (8) | Mean Total | Params.(M) |
|---|---|---|---|---|---|
| Full fine-tuning | 79.1 | 86.2 | 59.7 | 72.4 | 86.80 |
| Linear probing | 73.5 | 80.8 | 33.5 | 58.2 | 0.05 |
| MLP-4 [12] | 70.6 | 80.7 | 31.2 | 57.7 | 4.04 |
| Partial [12] | 73.1 | 81.7 | 35.0 | 58.9 | 12.65 |
| Bias [30] | 74.2 | 80.1 | 42.4 | 62.1 | 0.25 |
| VPT-Shallow [12] | 79.9 | 82.5 | 37.8 | 62.9 | 0.05 |
| VPT-Deep [12] | 76.8 | 84.5 | 53.4 | 67.7 | 0.22 |
| ARC [4] | 79.0 | 86.6 | **59.9** | 72.6 | 0.27 |
| RLRR | **81.3** | **86.7** | 59.0 | **73.0** | 0.41 |

Table 6. Ablation study on the FGVC dataset to examine the impact of the various RLRR combinations.

| scaling | | residual | FGVC Datasets | | | | | Mean | Params. (M) |
|---|---|---|---|---|---|---|---|---|---|
| left | right | | CUB | NABirds | Flowers | Dogs | Cars | | |
| ✓ | × | × | 86.9 | 84.2 | 99.5 | 91.0 | 84.3 | 89.2 | 0.39 |
| × | ✓ | | 87.3 | 84.4 | 99.3 | 91.1 | 84.5 | 89.3 | 0.39 |
| ✓ | ✓ | | 86.6 | 83.9 | 99.3 | 91.0 | 83.5 | 88.9 | 0.47 |
| ✓ | × | ✓ | 87.1 | 84.5 | 99.5 | 91.5 | 85.1 | 89.5 | 0.39 |
| × | ✓ | | 87.9 | 84.5 | 99.4 | 91.3 | 85.4 | 89.7 | 0.39 |
| ✓ | ✓ | | 89.3 | 84.7 | 99.5 | 92.0 | 87.0 | 90.4 | 0.47 |

Transformer is organized into discrete stages, each with transformer blocks of consistent feature dimensions, though the dimensions vary across stages. Table 5 showcases that RLRR upholds competitive adaptation accuracy, even when adapted to this specialized Transformer architecture, thereby affirming its robustness to a range of visual adaptation tasks.

### 4.3. Ablation Studies

To gain deeper insights into the proposed method, we conduct comprehensive ablation studies on RLRR to elucidate its critical features and to carry out pertinent analyses. The ablation studies examining module deployment are performed using the CIFAR-100 dataset [15]. Concurrently, we assess the impact of various components on FGVC dataset.

**Effect of RLRR Adaptation Insertion.** To assess the impact of RLRR adaptation, we experiment with its insertion into different layers and Transformer modules, including MHA, FFN, and LayerNorm. Notably, for LayerNorm, as its weights are not stored in matrix form, we follow the same approach as SSF [17]. The specific results are illustrated in Fig. 2. We observe that as the number of deployed layers increases, the accuracy improves across all settings. Moreover, the configuration where the residual and rescaling design are applied to all modules, as we employed, consistently outperforms other configurations. Consequently, we choose to deploy the residual and rescaling design across all modules.

**Effects of Different RLRR Combinations.** To further illustrate the importance of the residual and rescaling design, we evaluate the ablation effects of the various components within our proposed method. The findings are delineated in Table 6. The results reveal that one-sided (*i.e.* left or right) scaling tuning leads to better performance compared to dual-sided (*i.e.* left and right) tuning in the absence of the residual term. This suggests that excessive rescaling of the original pre-trained parameter matrix will compromise the generalizability learned by pre-trained models, especially without additional constraints. Intriguingly, when the residual term is included, this trend reverses, which not only demonstrates that rescaling can introduce flexible perturbations but also emphasizes the importance of the residual term in maintaining the intrinsic representational capacity of the model.

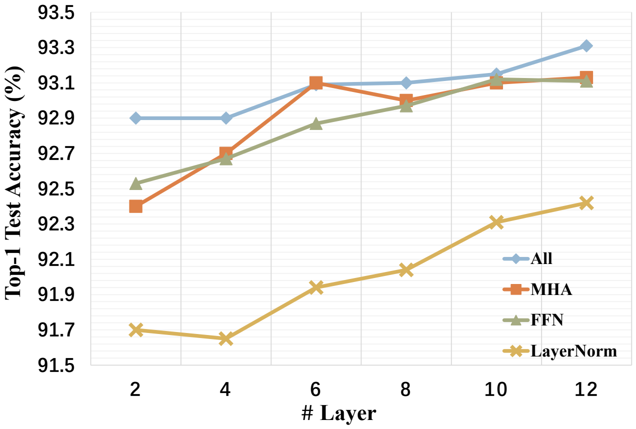

Figure 2. Ablation study using the VIT-B/16 backbone on the CIFAR-100 dataset to evaluate the impact of incorporating RLRR adaptation across different module and layer combinations.

## 5. Conclusions

In this study, we addressed the challenge of PEFT for pre-trained Vision Transformers, with a focus on achieving a delicate balance between retaining the generalization capacity of the pre-trained model and adapting effectively to downstream tasks. Our approach involved viewing PEFT through a novel SVD perspective, offering a unified framework for understanding the working mechanisms of various PEFT strategies and their trade-offs.

To achieve a more favorable trade-off, we introduced a RLRR fine-tuning strategy. RLRR incorporates a residual term, providing enhanced adaptation flexibility while simultaneously preserving the representation capacity of the pre-trained model. Through extensive experiments on two downstream benchmark datasets, our RLRR method demonstrated highly competitive adaptation performance and exhibited other desirable properties. This work contributes valuable insights into the PEFT landscape and proposes an effective strategy for achieving a more nuanced balance between generalization and task-specific adaptation in pre-trained Vision Transformers.

## References

[1] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022. 2, 6

[2] X Chen, S Xie, and K He. An empirical study of training self-supervised vision transformers. In *CVF International Conference on Computer Vision (ICCV)*, pages 9620–9629. 2

[3] 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, 6

[4] Wei Dong, Dawei Yan, Zhijun Lin, and Peng Wang. Efficient adaptation of large vision transformer via adapter re-composing. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 1, 2, 5, 6, 7, 8

[5] 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. In *International Conference on Learning Representations*, 2020. 1, 2, 6

[6] 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

[7] 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*, pages 16000–16009, 2022. 2

[8] 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, 6, 7

[9] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2021. 1, 2, 3, 4, 6, 7

[10] Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. Lorahub: Efficient cross-task generalization via dynamic lora composition, 2023. 2

[11] Mohammadreza Iman, Hamid Reza Arabnia, and Khaled Rasheed. A review of deep transfer learning and recent advancements. *Technologies*, 11(2):40, 2023. 2

[12] 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*, pages 709–727. Springer, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[13] Shibo Jie and Zhi-Hong Deng. Fact: Factor-tuning for lightweight adaptation on vision transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1060–1068, 2023. 1, 2, 6

[14] 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

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 8

[16] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021. 2

[17] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline for efficient model tuning. *Advances in Neural Information Processing Systems*, 35:109–123, 2022. 1, 2, 3, 4, 5, 6, 7, 8

[18] 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. 2

[19] 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. 1, 2, 6, 7

[20] Simian Luo, Yiqin Tan, Suraj Patil, Daniel Gu, Patrick von Platen, Apolinário Passos, Longbo Huang, Jian Li, and Hang Zhao. Lcm-lora: A universal stable-diffusion acceleration module. *arXiv preprint arXiv:2311.05556*, 2023. 2

[21] 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

[22] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. 2

[23] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 6

[24] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021. 6, 7

[25] 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*, pages 843–852, 2017. 2

[26] 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

[27] 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. 3

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

[29] Wei Ying, Yu Zhang, Junzhou Huang, and Qiang Yang. Transfer learning via learning to transfer. In *International Conference on Machine Learning*, pages 5085–5094. PMLR, 2018. 2

[30] 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 (Volume 2: Short Papers)*, pages 1–9, 2022. 6, 7, 8

[31] 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

[32] Jinghan Zhang, Junteng Liu, Junxian He, et al. Composing parameter-efficient modules with arithmetic operation. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[33] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020. 2