ReFTA: Breaking the Weight Reconstruction Bottleneck in Tensorized Parameter-Efficient Fine-Tuning

Jingjing Zheng[1,2,3,5] jjzheng233@gmail.com
Anda Tang[1] tanganda@pku.edu.cn
Qiangqiang Mao[3,4] maoq@student.ubc.ca
Zhouchen Lin[1,6,7,*] zlin@pku.edu.cn
Yankai Cao[2,3,4,*] yankai.cao@ubc.ca

[1]State Key Lab of General AI, School of Intelligence Science and Technology, Peking University
[2]Institute of Applied Mathematics, The University of British Columbia
[3]Centre for AI Decision-Making and Action, The University of British Columbia
[4]Department of Chemical and Biological Engineering, The University of British Columbia
[5]Department of Mathematics, The University of British Columbia
[6]Institute for Artificial Intelligence, Peking University
[7]Pazhou Laboratory (Huangpu), Guangzhou, Guangdong, China

Abstract

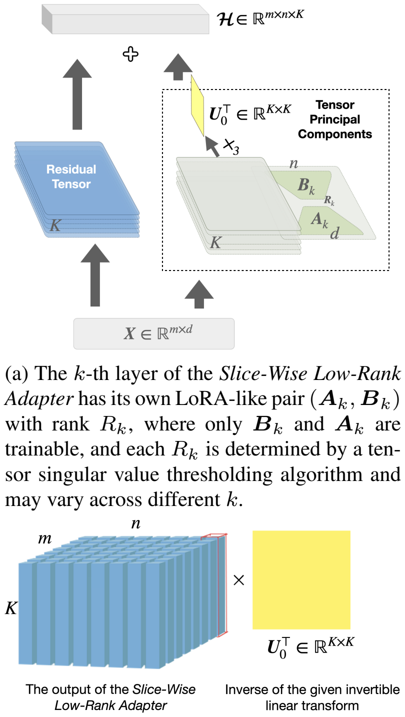
Tensor–based fine-tuning has attracted growing interest for its ability to reduce trainable parameters beyond matrix-based approaches such as LoRA and PiSSA, while capturing inter-layer correlations within networks. However, existing tensor methods typically require repeated reconstruction of model weights during training, leading to substantial computational and memory overhead. To overcome these limitations, we propose ***Re****construction-****F****ree* ***T****ensor-Based* ***A****daptation (ReFTA), which offers four key advantages: (1) it eliminates repeated explicit tensor reconstruction by exploiting the algebraic properties of tensors; (2) it achieves lower quantization error by fine-tuning only the principal tensor components; (3) it is supported by a rigorous generalization guarantee rooted in the algebraic foundations of tensor product–based approaches; and (4) it adopts a unified design controlled by a single tensor rank configuration. Extensive experiments on both image classification (IC) and natural language understanding (NLU) tasks demonstrate that ReFTA achieves the best accuracy–efficiency trade-off among all evaluated methods. Across most cases, ReFTA attains the highest average accuracy with the fewest trainable parameters. On IC tasks with the `ViT-Large` model, ReFTA surpasses LoRA by 5.6% in average accuracy, while reducing the number of trainable parameters by up to 96% compared to LoRA. On NLU tasks with `RoBERTa-Large`, ReFTA improves the average accuracy by approximately 5% over most existing methods while using only 86.4% fewer parameters than LoRA (*$r = 1$*) and 97.5% fewer than PiSSA. The code for ReFTA is available at* https://github.com/jzheng20/ReFTA.

1. Introduction

The ascent of large-scale foundation models has delivered unprecedented capabilities across natural language processing, vision, and multimodal tasks. However, these advances rely on an exponential increase in model size [4, 5, 8], posing severe challenges to efficiency and scalability. This has driven the development of numerous parameter-efficient fine-tuning (PEFT) methods that reduce the number of trainable parameters, including prefix-tuning [21, 23], adapter-based methods [15], sparse methods [12, 41], and low-rank decomposition-based adaptation [7, 16, 27, 42].

Among these, low-rank decomposition–based adaptation has emerged as one of the most effective PEFT strategies, with Low-Rank Adaptation (LoRA) [16] being its most prominent representative. LoRA introduces low-rank updates to pre-trained weights and has inspired a wide range of matrix decomposition–based PEFT variants [3, 16, 27, 28, 37, 38, 42]. For example, PiSSA [27] decomposes each layer's pre-trained weight matrix into a residual matrix $\boldsymbol{W}^{\text{res}}$ and a principal matrix $\boldsymbol{W}^{\text{pri}} = \boldsymbol{AB}$, where only the factors $\boldsymbol{A} \in \mathbb{R}^{d\times r}$ and $\boldsymbol{B} \in \mathbb{R}^{r\times n}$ are updated during training, and $r \ll \min(d, n)$. By initializing these factors from principal components, the fine-tuning process converges more rapidly toward promising local optima.

[*] Corresponding authors


$\mathcal{H} \in \mathbb{R}^{m\times n\times K}$
+
$U_0^\top \in \mathbb{R}^{K\times K}$
Tensor Principal Components
$\times_3$
Residual Tensor
K
n
B_k
R_k
A_k
d
K
$X \in \mathbb{R}^{m\times d}$
m
n
K
×
$U_0^\top \in \mathbb{R}^{K\times K}$
The output of the Slice-Wise Low-Rank Adapter
Inverse of the given invertible linear transform

(a) The k-th layer of the *Slice-Wise Low-Rank Adapter* has its own LoRA-like pair $(\boldsymbol{A}_k, \boldsymbol{B}_k)$ with rank R_k, where only $\boldsymbol{B}_k$ and $\boldsymbol{A}_k$ are trainable, and each R_k is determined by a tensor singular value thresholding algorithm and may vary across different k.

(b) *Feature Fusion Operation*: feature fusion along the third dimension via multiplication with $\boldsymbol{U}_0^\top$, where $\boldsymbol{U}_0$ denotes any fixed invertible linear transform, *e.g.*, Discrete Cosine Transform (DCT).

Figure 1. Illustration of ReFTA, where the Tensor Principal Components part consists of a *Slice-Wise Low-Rank Adapter* and a *Feature Fusion Operation* $\times_3 \boldsymbol{U}_0^\top$.

Despite their success, matrix decomposition–based PEFT methods are limited by their reliance on layer-wise low-rank structures. As model size grows, such singular value decomposition (SVD)-grounded approaches struggle to handle the rapidly increasing number of trainable parameters. This limitation has motivated a shift toward tensor decomposition-based methods, which have demonstrated strong potential for compressing high-dimensional data [14]. Recent works such as LoTR [2], FedTT [13], and LoRETTA [40] apply tensor decomposition techniques, *e.g.*, Tucker Decomposition [11, 22, 34] and Tensor-Train Decomposition (TT) [29], and leverage the inherent low rank tensor structure in the model weights to achieve more compact representations of the update weights, further reducing parameter counts. However, existing tensor methods introduce two critical limitations: (1) they often require repeated reconstruction of tensorized weights during training, incurring substantial computational and memory overhead; and (2) they introduce multiple interdependent rank hyperparameters, creating a heavy tuning burden for large-scale models. Together, these issues expose a key gap: while tensor-based PEFT improves parameter efficiency, its complex tensor structures restricts their practicality in large-scale settings due to the additional computational, memory, and implementation overhead they introduce.

To address this gap, we propose **Re**construction-**F**ree Low-rank **T**ensor **A**daptation (ReFTA), an efficient fine-tuning framework grounded in Tensor SVD (T-SVD) [25, 26, 30, 43], which is defined under the tensor-product (t-product). Compared with existing approaches, our method offers the following advantages:

- **Lower Quantization Error:** ReFTA leverages T-SVD and decouples the model weights into independent tensor components, thereby enabling, to the best of our knowledge, the principal-components-only tensor-based fine-tuning scheme and yielding substantially lower quantization error and more targeted parameter updates.
- **Simplified Tensor-Based Adaptation:** Rather than naively applying T-SVD to derive a structurally cumbersome tensor-based adaptation, we leverage the algebraic properties of tensors to formulate a concise and efficient adaptation design (Figure 1), which eliminates the need for explicit tensor reconstruction during forward and backward propagation, achieving higher computational and memory efficiency.
- **Theoretical Guarantee:** We provide the first theoretical guarantee for a tensor-based PEFT by deriving an explicit upper bound on the expected test error of ReFTA, establishing its robustness and providing a level of theoretical support previously absent in this line of work.
- **Single Rank Configuration:** In contrast to Tucker and TT-based methods that rely on multiple rank hyperparameters for different tensor modes, ReFTA adopts a unified design governed by a single tensor-rank hyperparameter, greatly simplifying configuration and tuning.

Extensive experiments on both vision and language benchmarks demonstrate the efficiency and generality of ReFTA. On `ViT` models, ReFTA achieves the best average accuracy among all compared PEFT methods while using the fewest trainable parameters. Specifically, on the `ViT-Large` model, ReFTA surpasses both LoRA and LoRETTA by 5.6% and 7.1% in average accuracy, respectively, while reducing the number of trainable parameters by up to 96% compared to LoRA and by 50% compared to LoRETTA. For `ViT-Huge`, evaluated on the challenging datasets OxfordPets, StanfordCars, and FGVC, ReFTA (r=15) achieves a 1.1% higher average accuracy while using only 5.4% of the trainable parameters of LoRA (r=8). On `RoBERTa-Large`, ReFTA (0.020M) also achieves the highest average accuracy across NLU tasks among all PEFT methods with the smallest parameter budget (over 97.5% fewer than PiSSA), outperforming leading approaches, including LoRA, PiSSA, LoRA-PRO [38], LoRETTA, and WeGeFT [31], by around 5% on average. Overall, ReFTA

Notations	Descriptions
a, $\boldsymbol{a}$	Scalar and vector, respectively
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \cdots$	Matrices
$\boldsymbol{\mathcal{A}}, \boldsymbol{\mathcal{B}}, \boldsymbol{\mathcal{C}}, \cdots$	Tensors
$\boldsymbol{A}^\top$	Transpose of $\boldsymbol{A}$
$\boldsymbol{X} \in \mathbb{R}^{m\times d}$	Matrix constructed from the input features, where m is the batch size, and d denotes the input feature dimension.
$\boldsymbol{\mathcal{W}}_0$	a tensor constructed from the pretrained network weight matrices.
$[\boldsymbol{\mathcal{A}}]_{i_1,i_2,i_3}$	The (i_1, i_2, i_3)-th element of $\boldsymbol{\mathcal{A}}$
$[\boldsymbol{\mathcal{A}}]_{i_1,i_2,:}$, $[\boldsymbol{\mathcal{A}}]_{:,:,i_3}$	The (i_1, i_2)-th tube and i_3-th frontal slice of $\boldsymbol{\mathcal{A}}$, respectively
$\lVert\boldsymbol{\mathcal{A}}\rVert_F$	$\sqrt{\sum_{1\le i_k \le I_k (k=1,2,3)} [\boldsymbol{\mathcal{A}}]^2_{i_1,i_2,i_3}}$ for $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1\times I_2\times I_3}$
$\boldsymbol{\mathcal{A}} \times_3 \boldsymbol{U}$	Mode-3 product of $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1\times I_2\times I_3}$ and $\boldsymbol{U} \in \mathbb{R}^{I_3\times I_3}$, *i.e.*, $[\boldsymbol{\mathcal{A}} \times_3 \boldsymbol{U}]_{i_1,i_2,:} = \boldsymbol{U}[\boldsymbol{\mathcal{A}}]_{i_1,i_2,:}$ for $i_1 = 1, 2, \cdots, I_1$ and $i_2 = 1, 2, \cdots, I_2$.
$\boldsymbol{\mathcal{A}} \times_1 \boldsymbol{B}$	Mode-1 product of $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1\times I_2\times I_3}$ and $\boldsymbol{B} \in \mathbb{R}^{I_1\times I_1}$, *i.e.*, $[\boldsymbol{\mathcal{A}} \times_1 \boldsymbol{B}]_{:,i_2,i_3} = \boldsymbol{B}[\boldsymbol{\mathcal{A}}]_{:,i_2,i_3}$ for $i_2 = 1, 2, \cdots, I_2$ and $i_3 = 1, 2, \cdots, I_3$.
$\boldsymbol{U}(\boldsymbol{\mathcal{A}}), \boldsymbol{U}^{-1}(\boldsymbol{\mathcal{A}})$	$\boldsymbol{U}(\boldsymbol{\mathcal{A}}) = \boldsymbol{\mathcal{A}} \times_3 \boldsymbol{U}, \boldsymbol{U}^{-1}(\boldsymbol{\mathcal{A}}) = \boldsymbol{\mathcal{A}} \times_3 \boldsymbol{U}^{-1}$

Table 1. Notations

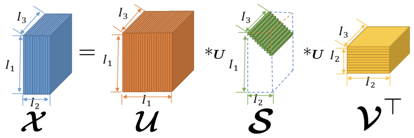

I_3
I_1
I_2
$\boldsymbol{\mathcal{X}}$
$=$
I_3
I_1
I_1
$\boldsymbol{\mathcal{U}}$
$*_{\boldsymbol{U}}$
I_3
I_1
I_2
$\boldsymbol{\mathcal{S}}$
$*_{\boldsymbol{U}}$
I_3
I_2
I_2
$\boldsymbol{\mathcal{V}}^\top$

Figure 2. Illustrations of T-SVD (the top-view perspective).

delivers state-of-the-art adaptation performance, extreme parameter efficiency, and strong cross-domain generalization, fully aligned with and supported by our theoretical analysis.

2. Notations and Preliminaries

Before introducing the methods, in Table 1, we summarize the symbols used in this paper.

2.1. Preliminary definitions and results

Definition 1. *(t-product) [18] Let $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1\times r\times I_3}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{r\times I_2\times I_3}$. Then the* t-product *$\boldsymbol{\mathcal{A}} *_{\boldsymbol{U}} \boldsymbol{\mathcal{B}}$ is defined to be a tensor of size $I_1 \times I_2 \times I_3$,*

$$\boldsymbol{\mathcal{A}} *_{\boldsymbol{U}} \boldsymbol{\mathcal{B}} = \boldsymbol{U}^{-1}(\boldsymbol{U}(\boldsymbol{\mathcal{A}}) \odot \boldsymbol{U}(\boldsymbol{\mathcal{B}})), \tag{1}$$

where $\boldsymbol{U}$ is given invertible linear transform satisfying $\boldsymbol{U}\boldsymbol{U}^\top = \boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{I}$, and $\boldsymbol{\mathcal{A}} \odot \boldsymbol{\mathcal{B}}$ is the slice-wise product of $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1\times r\times I_3}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{r\times I_2\times I_3}$, i.e., *$[\boldsymbol{\mathcal{A}} \odot \boldsymbol{\mathcal{B}}]_{:,:,i_3} = [\boldsymbol{\mathcal{A}}]_{:,:,i_3}[\boldsymbol{\mathcal{B}}]_{:,:,i_3}$ for $i_3 = 1, 2, \cdots, I_3$.*

Definition 2. *(f-diagonal tensor) [19] Tensor $\boldsymbol{\mathcal{S}}$ is called f-diagonal if each of its frontal slices is a diagonal matrix.*

Definition 3. *(Identity tensor) [18] Let $\boldsymbol{U}$ be orthogonal matrix in Definition 1. The tensor $\boldsymbol{\mathcal{I}} \in \mathbb{R}^{I_1\times I_1\times I_3}$ is called the identity tensor if each frontal slice of $\boldsymbol{U}(\boldsymbol{\mathcal{I}})$ is the identity matrix.*

Definition 4. *(Conjugate transpose) [18] Let $\boldsymbol{U}$ be orthogonal matrix in Definition 1. The conjugate transpose of a tensor $\boldsymbol{\mathcal{Q}} \in \mathbb{R}^{I_1\times I_2\times I_3}$ is the tensor $\boldsymbol{\mathcal{Q}}^\top$ that satisfies $[\boldsymbol{U}(\boldsymbol{\mathcal{Q}}^\top)]_{:,:,i_3} = [\boldsymbol{U}(\boldsymbol{\mathcal{Q}})]^\top_{:,:,i_3}$.*

Algorithm 1: Tensor PCA (TPCA)

```
Input: $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1\times I_2\times I_3}$, $\boldsymbol{U} \in \mathbb{R}^{I_3\times I_3}$, and $R > 0$.
Output: $\mathrm{TPCA}_R(\boldsymbol{\mathcal{Y}}, \boldsymbol{U})$, $\boldsymbol{U}(\boldsymbol{\mathcal{U}})$, $\boldsymbol{U}(\boldsymbol{\mathcal{S}})$, $\boldsymbol{U}(\boldsymbol{\mathcal{V}})$,
        and $\{R_k\}_{k=1}^{I_3}$.
Step 1: Compute $\boldsymbol{U}(\boldsymbol{\mathcal{Y}}) = \boldsymbol{\mathcal{Y}} \times_3 \boldsymbol{U}$;
Step 2: Perform matrix PCA on each frontal slice of
  $\boldsymbol{U}(\boldsymbol{\mathcal{Y}})$ by
for $i_3 = 1, ..., I_3$ do
    $[[\boldsymbol{U}(\boldsymbol{\mathcal{U}})]_{:,:,i_3}, [\boldsymbol{U}(\boldsymbol{\mathcal{S}})]_{:,:,i_3}, [\boldsymbol{U}(\boldsymbol{\mathcal{V}})]_{:,:,i_3}] =$
      $\mathrm{SVD}([\boldsymbol{U}(\boldsymbol{\mathcal{Y}})]_{:,:,i_3})$;
Step 3: Let $\mathbb{I}_R$ denotes the index set corresponding
  to the top $R$ values in $\boldsymbol{U}(\boldsymbol{\mathcal{S}})$;
Step 4: Compute tensor $\bar{\boldsymbol{\mathcal{X}}}$ by
for $k = 1, ..., I_3$ do
    $R_k = |\{(i_1, i_2, i_3) \in \mathbb{I}_R | i_3 = k\}|$;
    $[\bar{\boldsymbol{\mathcal{X}}}]_{:,:,k} =$
      $[\boldsymbol{U}(\boldsymbol{\mathcal{U}})]_{:,1:R_k,k}[\boldsymbol{U}(\boldsymbol{\mathcal{S}})]_{1:R_k,1:R_k,k}[\boldsymbol{U}(\boldsymbol{\mathcal{V}})]_{1:R_k,:,k}$;
Step 5: $\mathrm{TPCA}_R(\boldsymbol{\mathcal{Y}}, \boldsymbol{U}) = \bar{\boldsymbol{\mathcal{X}}} \times_3 \boldsymbol{U}^\top$;
```

Definition 5. *(Orthogonal tensor) [18] A tensor $\boldsymbol{\mathcal{Q}} \in \mathbb{R}^{I_1\times I_1\times I_3}$ is orthogonal if it satisfies $\boldsymbol{\mathcal{Q}}^\top *_{\boldsymbol{U}} \boldsymbol{\mathcal{Q}} = \boldsymbol{\mathcal{Q}} *_{\boldsymbol{U}} \boldsymbol{\mathcal{Q}}^\top = \boldsymbol{\mathcal{I}}$.*

Theorem 1. *(T-SVD) [18] Let $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times I_2\times I_3}$. Then it can be factorized as $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{U}} *_{\boldsymbol{U}} \boldsymbol{\mathcal{S}} *_{\boldsymbol{U}} \boldsymbol{\mathcal{V}}^\top$, where $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{I_1\times I_1\times I_3}$, $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{I_2\times I_2\times I_3}$ are orthogonal, and $\boldsymbol{\mathcal{S}} \in \mathbb{R}^{I_1\times I_2\times I_3}$ is a f-diagonal tensor.*

Definition 6. *(Tensor rank) For $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times I_2\times I_3}$, the tensor rank of $\boldsymbol{\mathcal{X}}$, denoted by $\mathrm{rank}(\boldsymbol{\mathcal{X}})$, is defined as the number of non-zero singular tubes of $\boldsymbol{\mathcal{S}}$, where $\boldsymbol{\mathcal{S}}$ is from the t-SVD of $\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{U}} *_{\boldsymbol{U}} \boldsymbol{\mathcal{S}} *_{\boldsymbol{U}} \boldsymbol{\mathcal{V}}^\top$. We can write*

$$\mathrm{rank}(\boldsymbol{\mathcal{X}}) = |\{(i_1, i_2, i_3) | [\boldsymbol{U}(\boldsymbol{\mathcal{S}})]_{i_1,i_2,i_3} \neq 0\}|.$$

Denote $\sigma(\boldsymbol{\mathcal{X}}) = \{[\boldsymbol{U}(\boldsymbol{\mathcal{S}})]_{i_1,i_2,i_3} | [\boldsymbol{U}(\boldsymbol{\mathcal{S}})]_{i_1,i_2,i_3} \neq 0\}$.

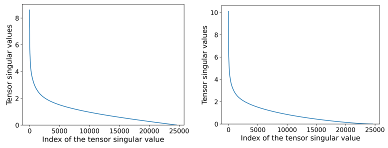
Definition 7. *(Tensor 2-norm) [26] For $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times I_2\times I_3}$, the tensor 2-norm of $\boldsymbol{\mathcal{X}}$, denoted by $\lVert\boldsymbol{\mathcal{X}}\rVert_2$, is defined as $\lVert\boldsymbol{\mathcal{X}}\rVert_2 = \max_{\sigma_i \in \sigma(\boldsymbol{\mathcal{X}})} \sigma_i$.*

Property 1. *[6] For $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{I_1\times I_2\times I_3}$, $\boldsymbol{B} \in \mathbb{R}^{I_i\times I_i}$, and $\boldsymbol{C} \in \mathbb{R}^{I_j\times I_j}$, then $\boldsymbol{\mathcal{A}} \times_i \boldsymbol{B} \times_j \boldsymbol{C} = \boldsymbol{\mathcal{A}} \times_j \boldsymbol{C} \times_i \boldsymbol{B}$ when $i \neq j$. (In this study, we will only use the case of $i = 1$ and $j = 3$, and the definitions of $\times_1$ and $\times_3$ are given in Table 1.)*

Theorem 2. *(Tensor PCA) For $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1\times I_2\times I_3}$ and non-negative integer R, the solution of*

$$\mathrm{TPCA}_R(\boldsymbol{\mathcal{Y}}, \boldsymbol{U}) = \arg\min_{\boldsymbol{\mathcal{X}}} \lVert\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{X}}\rVert_F^2 \quad s.t. \, \mathrm{rank}(\boldsymbol{\mathcal{X}}) \le R \tag{2}$$

is given by Algorithm 1.



(a) (b)

Figure 3. Tensor singular values of the weight tensor constructed by stacking the query projection matrices from the pretrained `ViT-Large` [9], under $\boldsymbol{U}$ set to (a) DCT and (b) LSM-3.

Based on the Eckart–Young–Mirsky theorem [10] and the orthogonality of $\boldsymbol{U}$, the solution to (2) can be derived as shown in Algorithm 1.

3. The Proposed Tensor-Based Adaptation

3.1. Empirical Observation of Low-Rank Tensor Structures

Let $\boldsymbol{\mathcal{W}}_0 \in \mathbb{R}^{d\times n\times K}$ be a third-order tensor constructed by stacking the pretrained weight matrices (e.g., query, key, or value matrices) across all attention layers. This formulation provides a unified representation that captures cross-layer parameter sharing and inter-layer dependencies. Here, d and n denote the input and output feature dimensions, and K is the number of layers. Given an input $\boldsymbol{X} \in \mathbb{R}^{m\times d}$, the stacked linear projections induced by $\boldsymbol{\mathcal{W}}_0$ can be written as[1]

$$\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{W}}_0 \times_1 \boldsymbol{X}, \tag{3}$$

where m is the batch size, and $[\boldsymbol{\mathcal{W}}_0 \times_1 \boldsymbol{X}]_{:,:,k} = \boldsymbol{X}[\boldsymbol{\mathcal{W}}_0]_{:,:,k}$ for $k = 1, 2, \cdots, K$ (see the definition of the mode-1 product $\times_1$ given in Table 1).

To investigate the underlying structure of pretrained weights, we analyze the weight tensor obtained by stacking the query matrices of `ViT-Large`. It is examined using T-SVD based on different invertible linear transforms $\boldsymbol{U}$, including Discrete Cosine Transform (DCT) [18] and the left singular matrix of the mode-3 unfolding of the weight tensor [20] (abbreviated as LSM-3), as illustrated in Fig. 3. Although the weight tensor is not strictly low-rank, most of its energy is concentrated in a small number of principal components. This empirical finding motivates our approach: we leverage T-SVD to develop a new PEFT method that directly operates on these principal tensor components.

3.2. Reconstruction-Free Low-Rank Tensor-Based Adaptation (ReFTA)

Motivated by the observations in the previous subsection, we decompose the weight tensor into a residual tensor $\boldsymbol{\mathcal{W}}_0^{\text{res}}$ and a principal tensor $\boldsymbol{\mathcal{W}}_0^{\text{pri}}$, while updating only the principal component tensor during fine-tuning. As will be analyzed in the Section 4, this design also helps to mitigate quantization error.

```
Algorithm 2: Initialization of ReFTA
  Input: $\boldsymbol{\mathcal{W}}_0$, invertible transform $\boldsymbol{U}_0$, and $R$.
  Output: $\boldsymbol{\mathcal{W}}_0^{\text{pri}}$, $\boldsymbol{\mathcal{W}}_0^{\text{res}}$, and the initialization of
    $\{\boldsymbol{A}_k\}_{k=1}^K$, and $\{\boldsymbol{B}_k\}_{k=1}^K$ in (8).
  Step 1: Calculate $\boldsymbol{\mathcal{W}}_0^{\text{pri}} = \text{TPCA}_R(\boldsymbol{\mathcal{W}}_0, \boldsymbol{U}_0)$, and
    obtain $\boldsymbol{U}_0(\boldsymbol{\mathcal{U}})$, $\boldsymbol{U}_0(\boldsymbol{\mathcal{S}})$, $\boldsymbol{U}_0(\boldsymbol{\mathcal{V}})$, and $\{R_k\}_{k=1}^K$ by
    Algorithm 1;
  Step 2: Obtain $\boldsymbol{\mathcal{W}}_0^{\text{res}}$:
    $\boldsymbol{\mathcal{W}}_0^{\text{res}} = \boldsymbol{\mathcal{W}}_0 - \text{TPCA}_R(\boldsymbol{\mathcal{W}}_0, \boldsymbol{U}_0)$;
  Step 3: Initialize $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$ as $[\boldsymbol{U}_0(\boldsymbol{\mathcal{U}})]_{:,1:R_k,k}$,
    $[\boldsymbol{U}_0(\boldsymbol{\mathcal{S}})]_{1:R_k,1:R_k,k}$ and $[\boldsymbol{U}_0(\boldsymbol{\mathcal{V}}^\top)]_{1:R_k,:,k}$,
    respectively, for $k = 1, 2, \cdots, K$.
```

For a given $\boldsymbol{U} = \boldsymbol{U}_0$, we employ a T-SVD–based tensor principal component analysis (TPCA) to obtain $\boldsymbol{\mathcal{W}}_0^{\text{pri}} = \text{TPCA}_R(\boldsymbol{\mathcal{W}}_0, \boldsymbol{U}_0)$ (see Algorithm 1), and thus obtain

$$\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{W}}_0^{\text{pri}} \times_1 \boldsymbol{X} + \boldsymbol{\mathcal{W}}_0^{\text{res}} \times_1 \boldsymbol{X}.$$

Using the orthogonality property $\boldsymbol{U}_0\boldsymbol{U}_0^\top = \boldsymbol{U}_0^\top\boldsymbol{U}_0 = \boldsymbol{I}$, we have

$$\begin{aligned}\boldsymbol{\mathcal{H}} &= \boldsymbol{U}_0(\boldsymbol{\mathcal{W}}_0^{\text{pri}}) \times_3 \boldsymbol{U}_0^\top \times_1 \boldsymbol{X} + \boldsymbol{\mathcal{W}}_0^{\text{res}} \times_1 \boldsymbol{X} \\ &= \boldsymbol{U}_0(\boldsymbol{\mathcal{W}}_0^{\text{pri}}) \times_1 \boldsymbol{X} \times_3 \boldsymbol{U}_0^\top + \boldsymbol{\mathcal{W}}_0^{\text{res}} \times_1 \boldsymbol{X}. \end{aligned} \tag{4}$$

The second equality in (4) is obtained by applying Property 1. The exchange of the two operations, $\times_3\boldsymbol{U}_0^\top$ and $\times_1\boldsymbol{X}$, is the key to avoiding explicit tensor reconstruction and distinguishes our formulation from the naive tensor-based adaptation directly derived via T-SVD. We will further elaborate on the benefits of this operation in Section 5.

For $\boldsymbol{U}_0(\boldsymbol{\mathcal{W}}_0^{\text{pri}})$ in (4), each frontal slice can be expressed as

$$\begin{aligned}&[\boldsymbol{U}_0(\boldsymbol{\mathcal{W}}_0^{\text{pri}})]_{:,:,k} \\ =&[\boldsymbol{U}_0(\boldsymbol{\mathcal{U}})]_{:,1:R_k,k}[\boldsymbol{U}_0(\boldsymbol{\mathcal{S}})]_{1:R_k,1:R_k,k}[\boldsymbol{U}_0(\boldsymbol{\mathcal{V}}^\top)]_{1:R_k,:,k}.\end{aligned}$$

Accordingly, we define $\boldsymbol{A}_k := [\boldsymbol{U}_0(\boldsymbol{\mathcal{U}})]_{:,1:R_k,k}[\boldsymbol{U}_0(\boldsymbol{\mathcal{S}})]_{1:R_k,1:R_k,k}$, $\boldsymbol{B}_k := [\boldsymbol{U}_0(\boldsymbol{\mathcal{V}}^\top)]_{1:R_k,:,k}$ for $1 \leq k \leq K$, and thus obtain

$$[\boldsymbol{U}_0(\boldsymbol{\mathcal{W}}_0^{\text{pri}})]_{:,:,k} = \boldsymbol{A}_k\boldsymbol{B}_k. \tag{5}$$

Furthermore, from the definition of the mode-1 product $\times_1$ (see Table 1), we have

$$[\boldsymbol{U}_0(\boldsymbol{\mathcal{W}}_0^{\text{pri}}) \times_1 \boldsymbol{X}]_{:,:,k} = \boldsymbol{X}[\boldsymbol{U}_0(\boldsymbol{\mathcal{W}}_0^{\text{pri}})]_{:,:,k} \tag{6}$$

for $k = 1, 2, \cdots, K$. Combining (6) with (5), we obtain

$$[\boldsymbol{U}_0(\boldsymbol{\mathcal{W}}_0^{\text{pri}}) \times_1 \boldsymbol{X}]_{:,:,k} = \boldsymbol{X}\boldsymbol{A}_k\boldsymbol{B}_k \tag{7}$$

[1] This formulation also applies to the multi-head attention within a single layer, where each head can be treated as an individual slice along the third mode.

Methods	ReFTA	Merged Weight Form I of ReFTA	Merged Weight Form II of ReFTA
Forward pass	$\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{H}}_{\text{int}} \times_3 \boldsymbol{U}_0^\top + \boldsymbol{\mathcal{W}}_0^{\text{res}} \times_1 \boldsymbol{X}$	$\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{W}}_{\text{int}} \times_3 \boldsymbol{U}_0^\top \times_1 \boldsymbol{X} + \boldsymbol{\mathcal{W}}_0^{\text{res}} \times_1 \boldsymbol{X},$	$\boldsymbol{\mathcal{H}} = (\boldsymbol{\mathcal{W}}_{\text{int}} \times_3 \boldsymbol{U}_0^\top + \boldsymbol{\mathcal{W}}_0^{\text{res}}) \times_1 \boldsymbol{X},$
Time Cost	$\mathcal{O}(mdnK \ + \ mnK^2)$	$\mathcal{O}(dRn \ + \ K^2 dn \ + \ mdnK)$	$\mathcal{O}(dRn \ + \ K^2dn \ + \ mdnK)$
Backward pass	$\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(3)}(\boldsymbol{\mathcal{H}}_{\text{int}})} = \boldsymbol{U}_0\Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(3)}(\boldsymbol{\mathcal{H}})}\Big)$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{A}_k} = \boldsymbol{X}^\top\Big(\frac{\partial \mathcal{L}}{\partial [\boldsymbol{\mathcal{H}}_{\text{int}}]_{:,:,k}}\Big)\boldsymbol{B}_k^\top$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{B}_k} = \boldsymbol{A}_k^\top \boldsymbol{X}^\top\Big(\frac{\partial \mathcal{L}}{\partial [\boldsymbol{\mathcal{H}}_{\text{int}}]_{:,:,k}}\Big)$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{X}} = \Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(1)}(\boldsymbol{\mathcal{H}}_{\text{int}})}\Big)\text{Unfold}_{(1)}(\boldsymbol{\mathcal{W}}_{\text{int}})^\top$ $+\Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(1)}(\boldsymbol{\mathcal{H}})}\Big)\text{Unfold}_{(1)}(\boldsymbol{\mathcal{W}}_0^{\text{res}})^\top$	$\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(1)}(\boldsymbol{\mathcal{W}}^{\text{pri}})} = \boldsymbol{X}^\top\Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(1)}(\boldsymbol{\mathcal{H}})}\Big)$ $\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(3)}(\boldsymbol{\mathcal{W}}_{\text{int}})} = \boldsymbol{U}_0\Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(3)}(\boldsymbol{\mathcal{W}}^{\text{pri}})}\Big)$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{A}_k} = \Big(\frac{\partial \mathcal{L}}{\partial [\boldsymbol{\mathcal{W}}_{\text{int}}]_{:,:,k}}\Big)\boldsymbol{B}_k^\top$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{B}_k} = \boldsymbol{A}_k^\top\Big(\frac{\partial \mathcal{L}}{\partial [\boldsymbol{\mathcal{W}}_{\text{int}}]_{:,:,k}}\Big)$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{X}} = \Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(1)}(\boldsymbol{\mathcal{H}})}\Big)(\text{Unfold}_{(1)}(\boldsymbol{\mathcal{W}}^{\text{pri}} + \boldsymbol{\mathcal{W}}_0^{\text{res}}))^\top$	$\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(1)}(\boldsymbol{\mathcal{W}}^{\text{pri}})} = \boldsymbol{X}^\top\Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(1)}(\boldsymbol{\mathcal{H}})}\Big)$ $\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(3)}(\boldsymbol{\mathcal{W}}_{\text{int}})} = \boldsymbol{U}_0\Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(3)}(\boldsymbol{\mathcal{W}}^{\text{pri}})}\Big)$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{A}_k} = \Big(\frac{\partial \mathcal{L}}{\partial [\boldsymbol{\mathcal{W}}_{\text{int}}]_{:,:,k}}\Big)\boldsymbol{B}_k^\top$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{B}_k} = \boldsymbol{A}_k^\top\Big(\frac{\partial \mathcal{L}}{\partial [\boldsymbol{\mathcal{W}}_{\text{int}}]_{:,:,k}}\Big)$ $\frac{\partial \mathcal{L}}{\partial \boldsymbol{X}} = \Big(\frac{\partial \mathcal{L}}{\partial \text{Unfold}_{(1)}(\boldsymbol{\mathcal{H}})}\Big)(\text{Unfold}_{(1)}(\boldsymbol{\mathcal{W}}^{\text{pri}} + \boldsymbol{\mathcal{W}}_0^{\text{res}}))^\top$
Time Cost	$\mathcal{O}(mnK^2 + mdnK + dnR)$	$\mathcal{O}(mdnK + dnR + dnK^2)$	$\mathcal{O}(mdnK + dnR + dnK^2)$

Table 2. Comparison of the forward and backward formulations among ReFTA and its two merged-weight variants.

for $k = 1, 2, \cdots, K$

Defining $\boldsymbol{\mathcal{H}}_{\text{int}}$ by $[\boldsymbol{\mathcal{H}}_{\text{int}}]_{:,:,k} := \boldsymbol{X}\boldsymbol{A}_k\boldsymbol{B}_k$ for $k = 1, 2, \cdots, K$, we obtain our final formulation

$$\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{H}}_{\text{int}} \times_3 \boldsymbol{U}_0^\top + \boldsymbol{\mathcal{W}}_0^{\text{res}} \times_1 \boldsymbol{X} \tag{8}$$

by combining (4) and (7).

Equation (8) forms the proposed Tensor-Based Adaptation (ReFTA) as illustrated in Figure 1, in which we fine-tune only the parameters $\{\boldsymbol{A}_k\}_{k=1}^K$ and $\{\boldsymbol{B}_k\}_{k=1}^K$, while keeping the invertible transform $\boldsymbol{U}_0$ and the residual tensor $\boldsymbol{\mathcal{W}}_0^{\text{res}}$ fixed. The component $[\boldsymbol{\mathcal{H}}_{\text{int}}]_{:,:,k} = \boldsymbol{X}\boldsymbol{A}_k\boldsymbol{B}_k$ for $1 \leq k \leq K$, correspond to the *Slice-Wise Low-Rank Adapter* illustrated in Figure 1. The initialization process for ReFTA is summarized in Algorithm 2.

As shown in the algorithm, ReFTA requires only a single tensor-rank configuration to determine $\{R_k\}_{k=1}^K$, unlike Tucker or TT-based methods that rely on multiple rank hyperparameters across tensor modes.

4. Lower Quantization Error

As stated in [27], fine-tuning the principal components can effectively reduce quantization error. In the tensor case, a similar conclusion can be derived. To demonstrate ReFTA's advantage, we consider the following variant:

$$\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{H}}_{\text{int}} \times_3 \boldsymbol{U}_0^\top + \boldsymbol{\mathcal{W}}_0 \times_1 \boldsymbol{X}, \tag{9}$$

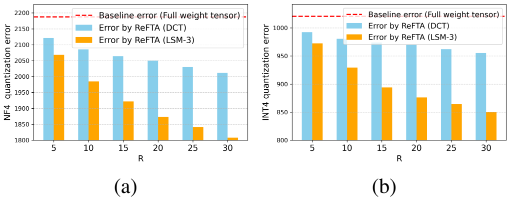
where $\{\boldsymbol{A}_k\}_{k=1}^K$ and $\{\boldsymbol{B}_k\}_{k=1}^K$ in $\boldsymbol{\mathcal{H}}_{\text{int}}$ are initialized using Gaussian-Zero initialization. The corresponding initialization quantization error is given as

$$\|\boldsymbol{\mathcal{W}}_0 - \mathrm{Q}(\boldsymbol{\mathcal{W}}_0)\|_F^2, \tag{10}$$

where $\mathrm{Q}(\cdot)$ denotes the quantization operator. The initialization quantization error of the original ReFTA becomes

$$\|\boldsymbol{\mathcal{W}}_0 - (\mathrm{Q}(\boldsymbol{\mathcal{W}}_0^{\text{res}}) + \boldsymbol{\mathcal{W}}_0^{\text{pri}})\|_F^2 = \|\boldsymbol{\mathcal{W}}^{\text{res}} - \mathrm{Q}(\boldsymbol{\mathcal{W}}_0^{\text{res}})\|_F^2. \tag{11}$$

We compare (10) and (11) in Fig. 4 under both **NF4** [7] and **INT4** [39] quantization schemes. As shown in the figure, when $\boldsymbol{U}_0$ is taken as either the LSM-3 or DCT, the quantization errors of ReFTA are consistently lower than those given in (10), indicated by the red dashed line. Moreover, as the rank R increases, the both ReFTA (LSM-3) and ReFTA (DCT) exhibit a monotonic decrease in quantization error, indicating that higher-rank approximations capture more principal information and leave smaller quantization-sensitive residuals. These results clearly demonstrate ReFTA's inherent robustness to quantization.



(a)

(b)

Figure 4. Comparison of NF4 and INT4 quantization errors from ReFTA and its variant (9) for the weight tensor by stacking the query projection matrices in `ViT-Large`.

5. Why Exchanging $\times_3\boldsymbol{U}_0^\top$ and $\times_1\boldsymbol{X}$ Eliminates Tensor Reconstruction?

In this section, we explain why exchanging the operations $\times_1\boldsymbol{X}$ and $\times_3\boldsymbol{U}_0^\top$ eliminates the need for explicit tensor reconstruction during training and helps improve runtime and memory efficiency in both the forward and backward passes. To analyze this, we examine the three formulations summarized in Table 2, where the merged-weight forms are obtained by naively applying T-SVD, the tensor $\boldsymbol{\mathcal{W}}_{\text{int}} \in \mathbb{R}^{d\times n\times K}$ is defined as $[\boldsymbol{\mathcal{W}}_{\text{int}}]_{:,:,k} := \boldsymbol{A}_k\boldsymbol{B}_k$ for $k = 1, 2, \cdots, K$, and $\boldsymbol{\mathcal{W}}^{\text{pri}} := \boldsymbol{\mathcal{W}}_{\text{int}} \times_3 \boldsymbol{U}_0^\top$.

By comparing these formulations, we observe that, unlike the merged-weight forms that require explicit reconstruction of weight tensor during each forward and backward pass, ReFTA performs adaptation directly in the feature space. This design avoids redundant tensor–matrix multiplications and enables efficient gradient propagation through lightweight intermediate representations ($\boldsymbol{\mathcal{H}}_{\text{int}}$). The corresponding computational complexity analysis is presented in Table 2, from which it can be seen that ReFTA significantly reduces both forward and backward costs when $m \ll d$.

Model	Method	#Params	Accuracy (%)					Avg.
			OxfordPets	StanfordCars	FGVC	RESISC45	CIFAR100	
ViT-Base	FF	85.8M	93.14$_{\pm0.40}$	79.78$_{\pm1.15}$	54.84$_{\pm1.23}$	96.13$_{\pm0.13}$	92.38$_{\pm0.13}$	83.25
	LP	-	90.28$_{\pm0.43}$	25.76$_{\pm0.28}$	17.44$_{\pm0.43}$	74.22$_{\pm0.10}$	84.28$_{\pm0.11}$	58.39
	LoRA ($r = 16$)	581K	93.19$_{\pm0.36}$	45.38$_{\pm0.41}$	25.16$_{\pm0.16}$	92.70$_{\pm0.18}$	92.02$_{\pm0.12}$	69.69
	PiSSA ($r = 8$)	313K	93.84$_{\pm0.3}$	78.43$_{\pm0.5}$	51.56$_{\pm1.8}$	**93.81**$_{\pm1.8}$	**93.31**$_{\pm0.2}$	82.19
	PiSSA ($r = 1$)	55K	93.83$_{\pm0.1}$	60.29$_{\pm0.3}$	29.61$_{\pm0.2}$	92.87$_{\pm0.2}$	91.98$_{\pm0.2}$	73.71
	LoRA-PRO	313K	**94.03**$_{\pm0.1}$	72.12$_{\pm0.4}$	43.39$_{\pm0.7}$	93.66$_{\pm0.2}$	92.54$_{\pm0.1}$	79.14
	WeGeFT	49K	92.71$_{\pm0.2}$	76.18$_{\pm0.2}$	51.82$_{\pm0.9}$	93.03$_{\pm0.2}$	91.46$_{\pm0.2}$	81.04
	LoRETTA ($r = 5$)	57K	93.39$_{\pm0.4}$	74.15$_{\pm0.8}$	48.86$_{\pm0.5}$	93.36$_{\pm0.1}$	91.87$_{\pm0.1}$	80.32
	ReFTA ($R = 15$)	**46K**	93.93$_{\pm0.21}$	**80.23** $_{\pm0.19}$	**52.79**$_{\pm2.71}$	93.35 $_{\pm0.18}$	91.83 $_{\pm0.15}$	**82.42**
ViT-Large	FF	303.3M	94.43$_{\pm0.56}$	88.90$_{\pm0.26}$	68.25$_{\pm1.63}$	96.43$_{\pm0.07}$	93.58$_{\pm0.19}$	88.31
	LP	-	91.11$_{\pm0.30}$	37.91$_{\pm0.27}$	24.62$_{\pm0.24}$	82.02$_{\pm0.11}$	84.28$_{\pm0.11}$	63.98
	LoRA ($r = 16$)	1.57M	**94.82**$_{\pm0.09}$	73.25$_{\pm0.36}$	42.32$_{\pm0.98}$	94.71$_{\pm0.25}$	**94.87**$_{\pm0.10}$	79.99
	PiSSA ($r = 8$)	835K	94.04$_{\pm0.4}$	**84.19**$_{\pm0.7}$	59.81$_{\pm0.6}$	94.99$_{\pm0.2}$	92.42$_{\pm0.1}$	85.09
	LoRA-PRO	835K	94.67$_{\pm0.1}$	83.57$_{\pm0.3}$	57.71$_{\pm0.5}$	**95.12**$_{\pm0.1}$	93.53$_{\pm0.1}$	84.92
	PiSSA ($r = 1$)	147K	93.98$_{\pm0.3}$	83.04$_{\pm0.3}$	56.72$_{\pm0.6}$	94.64$_{\pm0.2}$	93.25$_{\pm0.1}$	84.32
	WeGeFT ($r = 16$)	65K	94.37 $_{\pm0.13}$	75.17 $_{\pm0.59}$	58.04 $_{\pm0.68}$	94.30 $_{\pm0.15}$	93.00 $_{\pm0.08}$	82.97
	LoRETTA ($r = 5$)	132K	78.28$_{\pm0.3}$	68.44$_{\pm0.3}$	58.04$_{\pm0.9}$	94.53$_{\pm0.1}$	93.28$_{\pm0.1}$	78.51
	ReFTA($R = 15$)	**61K**	94.80$_{\pm0.22}$	84.01$_{\pm0.35}$	**61.69**$_{\pm0.92}$	94.59$_{\pm0.26}$	93.26 $_{\pm0.08}$	**85.67**

Table 3. Performance of different fine-tuning methods on various image classification datasets using `ViT` models. The FF results and the results of methods with the smallest parameter counts are shaded in gray and light blue, respectively.

Furthermore, ReFTA also provides a significant memory advantage. In the merged-weight forms, the transformed tensor $\boldsymbol{\mathcal{W}}^{\text{pri}} = \boldsymbol{\mathcal{W}}_{\text{int}} \times_3 \boldsymbol{U}_0^\top$ must be explicitly reconstructed and stored in memory for each iteration, along with its gradient graph, resulting in a memory complexity of $\mathcal{O}(dnK)$. By contrast, ReFTA require only the storage of the intermediate feature tensor $\boldsymbol{\mathcal{H}}_{\text{int}} \in \mathbb{R}^{m \times n \times K}$, whose size scales as $\mathcal{O}(mnK)$ with $m \ll d$. As a result, ReFTA achieves a considerably smaller memory footprint compared to both merged-weight formulations, which is further validated by our experimental results.

Model	Method	#Params	Accuracy (%)			Avg.
			OxfordPets	StanfordCars	FGVC	
ViT-Huge	LoRA ($r = 8$)	1392K	91.26$_{\pm0.34}$	78.03$_{\pm0.27}$	56.41$_{\pm1.72}$	75.23
	PiSSA ($r = 8$)	1392K	89.44$_{\pm0.72}$	75.49$_{\pm0.92}$	52.45$_{\pm1.80}$	72.46
	LoRA ($r = 1$)	245K	90.97$_{\pm0.20}$	74.97$_{\pm1.60}$	53.40$_{\pm0.74}$	73.11
	PiSSA ($r = 1$)	245K	90.74$_{\pm0.35}$	73.10$_{\pm0.35}$	50.71$_{\pm2.19}$	71.51
	LoRETTA ($r = 5$)	194K	90.56$_{\pm0.84}$	74.57$_{\pm0.38}$	51.26$_{\pm1.85}$	72.13
	ReFTA ($R = 15$)	76K	**92.56**$_{\pm0.14}$	**79.77**$_{\pm0.29}$	**56.65**$_{\pm0.60}$	**76.32**
	ReFTA ($R = 5$)	**25K**	92.09$_{\pm0.09}$	76.66$_{\pm0.31}$	54.82$_{\pm0.54}$	74.52

Table 4. Performance of different methods on various image classification datasets using `ViT-Huge` model.

6. Theoretical Guarantee for ReFTA

To provide theoretical insight into the generalization behavior of the proposed ReFTA, we establish an upper bound on the expected test error based on its tensor low-rank structure. Specifically, by analyzing the hypothesis class $\mathcal{F}_{\text{ReFTA}} = \{\phi(\boldsymbol{\mathcal{W}} \times_1 \boldsymbol{x}^\top) \mid \|\boldsymbol{\mathcal{W}}\|_2 \leq B, \boldsymbol{\mathcal{W}} \in \mathbb{R}^{d \times n \times K}\}$ for $\boldsymbol{x} \in \mathbb{R}^{d \times 1}$ and leveraging the spectral norm constraint on the tensorized parameters, we derive the following generalization bound that characterizes how the model complexity scales with the tensor rank R, the number of attention heads K, and the number of samples m.

Theorem 3. *Let $g(\cdot)$ be a $l(g)$-Lipschitz loss function from $(f_{\boldsymbol{\mathcal{W}}}(\boldsymbol{x}), \boldsymbol{y})$ to $[0, 1]$, where $f_{\boldsymbol{\mathcal{W}}} \in \mathcal{F}_{\text{ReFTA}} = \{\phi(\boldsymbol{\mathcal{W}} \times_1 \boldsymbol{x}^\top) \mid \|\boldsymbol{\mathcal{W}}\|_2 \leq B, \boldsymbol{\mathcal{W}} \in \mathbb{R}^{d \times n \times K}\}$ and $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{X} \times \mathbb{Y}$, $\mathbb{X} \subseteq \mathbb{R}^d$ and $\mathbb{Y}$ are feature space and output space, respectively. For any $\delta > 0$, the following holds with probability at least $1 - \delta$ for a randomly chosen i.i.d. samples $\mathbb{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^m$:*

$$\mathbb{E}[g(f_{\boldsymbol{\mathcal{W}}}(\boldsymbol{x}), \boldsymbol{y})] \leq \frac{1}{m} \sum_{i=1}^{m} g(f_{\mathbf{w}}(\mathbf{x}_i), \mathbf{y}_i) + \sqrt{l(g)\pi}\, l(\phi) \hat{R} B \sqrt{\frac{RnK}{m}} + \sqrt{\frac{9 \log \frac{2}{\delta}}{2m}}, \quad (12)$$

where $R = \operatorname{rank}(\boldsymbol{\mathcal{W}})$, $l(\phi)$ is Lipschitz constant for function ϕ, $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m]^\top \in \mathbb{R}^{m \times d}$ for the samples $\{\boldsymbol{x}_i\}_{i=1}^m$, and $\max_{i=1,2,\cdots,m} \|\boldsymbol{x}_i\| \leq \hat{R}$.

This theorem shows that the generalization gap of ReFTA is bounded by a term proportional to $\sqrt{RnK/m}$, indicating that a smaller tensor rank R effectively reduces the model complexity. Therefore, ReFTA not only improves parameter efficiency but also provides theoretical generalization guarantees that favor low-rank tensor adaptation in large-scale models.

7. Experiments

In this section, we compare ReFTA with state-of-the-art PEFT methods, including Linear Probing (LP), Bitfit [41], LoRA [16], DyLoRA [35], AdaLoRA [42], FourierFT [12], PiSSA [27], LoRETTA [40], LoRA-PRO [38], and WeGeFT [31], to evaluate its effectiveness across image classification, natural language understanding, and commonsense reasoning tasks. Following the same experimental settings as [12], we prioritize reusing their reported results whenever applicable for consistency and fairness. Furthermore, we conduct ablation studies to investigate the effects of different invertible transforms and tensor ranks R on model performance.

For all tasks, we report the average performance over five random seeds. The best results among PEFT methods are highlighted in **bold**. All experiments were conducted on an NVIDIA Tesla V100 (32 GB) using Python 3.12.3. Except for the learning rate and weight decay, which are individually tuned to their optimal values, all other hyperparameters and training settings, including batch size, number of epochs, optimizer, and random seeds, follow the configurations in [12]. Unless otherwise specified, all reported results are obtained by fine-tuning only the *query* and *value* projection matrices, as well as the classification head.

7.1. Image Classification

We evaluate all methods on image classification using the widely adopted Vision Transformer (`ViT`) [9], a representative foundation model in computer vision, across five

Model	**Method**	**#Params**	SST-2	MRPC	CoLA	QNLI	RTE	STS-B	**Avg.**
			Acc.	Acc.	MCC	Acc.	Acc.	PCC	
Base	FF	125M	94.8	90.2	63.6	92.8	78.7	91.2	85.2
	LoRA	0.3M	**95.1**$_{\pm 0.2}$	89.7$_{\pm 0.7}$	63.4$_{\pm 1.2}$	**93.3**$_{\pm 0.3}$	78.4$_{\pm 0.8}$	**91.5**$_{\pm 0.2}$	85.2
	AdaLoRA	0.3M	94.5$_{\pm 0.2}$	88.7$_{\pm 0.5}$	62.0$_{\pm 0.6}$	93.1$_{\pm 0.2}$	**81.0**$_{\pm 0.6}$	90.5$_{\pm 0.2}$	85.0
	DyLoRA	0.3M	94.3$_{\pm 0.5}$	89.5$_{\pm 0.5}$	61.1$_{\pm 0.3}$	92.2$_{\pm 0.5}$	78.7$_{\pm 0.7}$	91.1$_{\pm 0.6}$	84.5
	PiSSA ($r = 8$)	0.3M	93.9$_{\pm 0.1}$	89.3$_{\pm 0.8}$	62.1$_{\pm 2.9}$	91.3$_{\pm 0.1}$	77.3$_{\pm 1.4}$	90.5$_{\pm 0.2}$	84.1
	LoRA-PRO	0.3M	94.2$_{\pm 0.3}$	**90.1**$_{\pm 0.5}$	**64.3**$_{\pm 0.72}$	92.0$_{\pm 0.2}$	80.2$_{\pm 1.8}$	90.9$_{\pm 0.22}$	**85.3**
	LoRA ($r = 1$)	0.055M	93.7$_{\pm 0.5}$	89.2$_{\pm 0.3}$	62.3$_{\pm 3.6}$	90.6$_{\pm 0.4}$	79.5$_{\pm 0.4}$	80.8$_{\pm 20.6}$	82.7
	PiSSA ($r = 1$)	0.055M	93.3$_{\pm 0.2}$	89.3$_{\pm 0.6}$	62.6$_{\pm 1.4}$	90.6$_{\pm 0.4}$	74.9$_{\pm 1.2}$	90.0$_{\pm 0.3}$	83.4
	WeGeFT	0.049M	94.1$_{\pm 0.5}$	89.5$_{\pm 0.5}$	63.5$_{\pm 1.3}$	91.2$_{\pm 0.4}$	78.6$_{\pm 1.6}$	90.5$_{\pm 0.1}$	84.6
	LoRETTA	0.057M	94.6$_{\pm 0.5}$	88.3$_{\pm 0.7}$	61.8$_{\pm 1.3}$	92.7$_{\pm 0.2}$	75.1$_{\pm 5.3}$	90.5$_{\pm 0.1}$	83.8
	ReFTA ($R = 15$)	**0.046M**	94.8 ± 0.26	90.0 ± 0.41	63.4 ± 1.53	92.9 ± 0.19	77.3 ± 1.69	90.6 ± 0.13	84.8
Large	FF	356M	96.4	90.9	68	94.7	86.6	92.4	88.2
	LoRA	0.8M	96.2$_{\pm 0.5}$	90.2$_{\pm 1.0}$	68.2$_{\pm 1.9}$	**94.8**$_{\pm 0.3}$	85.2$_{\pm 1.1}$	92.3$_{\pm 0.5}$	87.8
	PiSSA ($r = 8$)	0.8M	95.5$_{\pm 0.2}$	86.9$_{\pm 2.6}$	61.1$_{\pm 3.4}$	92.1$_{\pm 1.7}$	56.8$_{\pm 8.2}$	91.8$_{\pm 0.4}$	80.7
	LoRA-PRO	0.8M	95.9$_{\pm 0.2}$	**90.9**$_{\pm 0.4}$	66.7$_{\pm 2.0}$	93.0$_{\pm 0.5}$	60.5$_{\pm 13.5}$	**92.0**$_{\pm 0.1}$	83.2
	LoRA ($r = 1$)	0.147M	95.7$_{\pm 0.4}$	88.3$_{\pm 0.7}$	62.2$_{\pm 2.4}$	93.9$_{\pm 0.2}$	82.2$_{\pm 2.5}$	78.2$_{\pm 29.7}$	83.4
	PiSSA ($r = 1$)	0.147M	95.2$_{\pm 0.2}$	84.9$_{\pm 3.4}$	56.6$_{\pm 6.2}$	93.4$_{\pm 0.3}$	65.9$_{\pm 11.3}$	91.3$_{\pm 0.2}$	81.2
	WeGeFT	0.065M	95.0$_{\pm 0.3}$	75.7$_{\pm 7.7}$	64.0$_{\pm 2.0}$	93.7$_{\pm 0.3}$	53.6$_{\pm 1.2}$	91.4$_{\pm 0.3}$	78.9
	LoRETTA	0.132M	96.2$_{\pm 0.2}$	90.5$_{\pm 0.4}$	**69.5**$_{\pm 0.6}$	94.1$_{\pm 0.9}$	53.0$_{\pm 0.5}$	92.0$_{\pm 0.2}$	82.6
	ReFTA ($R = 5$)	**0.020M**	**96.3**$_{\pm 0.3}$	90.8$_{\pm 0.5}$	68.6$_{\pm 1.3}$	**94.8**$_{\pm 0.1}$	**87.1**$_{\pm 0.7}$	91.4$_{\pm 0.1}$	**88.2**

Table 5. Performance of different fine-tuning methods on six datasets of the GLUE benchmark using RoBERTa models. The FF results and the results of methods with the smallest parameter counts are shaded in gray and light blue, respectively.

public datasets: OxfordPets[2], StanfordCars[3], FGVC[3], RESISC45[4], and CIFAR100[3]. We train all methods for 10 epochs on each dataset. For ReFTA, DCT is used as the invertible transform.

Table 3 presents the results on all five image classification datasets using ViT-Base and ViT-Large. As shown, ReFTA consistently achieves the best overall performance among all compared PEFT methods while using the fewest trainable parameters. In particular, on ViT-Large, ReFTA outperforms both the tensor-based method LoRETTA and LoRA by more than 5% in accuracy, while requiring only half as many parameters as LoRETTA and about 3.9% of LoRA's parameter count. Among all methods with the smallest parameter counts (highlighted in light blue), ReFTA consistently outperforms the second-best by at least 1.3% on both ViT-Base and ViT-Large.

To further evaluate the effectiveness of ReFTA under different parameter budgets, we also conduct experiments on ViT-Huge. The results are summarized in Table 4. Across all datasets, ReFTA $(R = 15)$ consistently outperforms existing methods, including LoRA, PiSSA, and LoRETTA, while requiring substantially fewer trainable parameters. Notably, it surpasses LoRA $(r = 8)$ and PiSSA $(r = 8)$ while using only 5.4% of their trainable parameters.

These results demonstrate that ReFTA consistently achieves strong generalization and parameter efficiency across both model scales, aligning well with our theoretical analysis on the upper bound of low-rank tensor adaptation, and validating the consistency between theory and practice.

Method	Mistral-7B		LLaMA-2-7B		LLaMA-3-8B		Avg.
	Acc. (%)	Params (K)	Acc. (%)	Params (K)	Acc. (%)	Params (K)	
LoRA (r=8)	**86.90**	3407	**85.37**	4194	**86.08**	3407	**86.11**
PiSSA (r=8)	83.34	3407	**85.37**	4194	85.18	3407	84.63
LoRA (r=1)	86.08	425	84.60	524	85.75	425	85.47
PiSSA (r=1)	85.26	425	83.37	524	81.35	425	83.32
LoRETTA (r=5)	73.92	353	82.47	359	85.91	353	80.76
ReFTA (r=15)	86.81	199	84.86	245	**86.08**	119	85.91
ReFTA (r=5)	85.75	66	84.85	81	84.36	66	84.98

Table 6. Comparison of accuracy (%) on CommonsenseQA.

7.2. Natural Language Understanding

In this subsection, we evaluate all methods on the General Language Understanding Evaluation (GLUE) benchmark [36], where LSM-3 is used as the invertible linear transform for ReFTA. This benchmark encompasses a diverse set of natural language understanding tasks, including single-sentence classification, similarity and paraphrase, and natural language inference. Besides, we use the robustly optimized BERT model, including RoBERTa-Base and RoBERTa-Large [24], for the evaluation. We evaluate the performance of the fine-tuned models using three key metrics: Matthew's correlation coefficient (MCC) for CoLA, Pearson correlation coefficient (PCC) for STS-B, and accuracy (Acc.) for all other tasks. Following [12], we set the maximum number of training epochs to 100 and select the best epoch for each run.

Table 5 presents the fine-tuning performance of RoBERTa-Base and RoBERTa-Large on six datasets from the GLUE benchmark. On RoBERTa-Base, our proposed ReFTA achieves performance comparable to other PEFT methods while using the fewest trainable parameters. Notably, on RoBERTa-Large, ReFTA matches the performance of full fine-tuning (FF) and exceeds that of almost all other PEFT methods by more than 5% in average accuracy, while maintaining the smallest parameter count. Among methods with the smallest number of trainable parameters (highlighted in light blue), ReFTA outperforms WeGeFT by approximately 9% in average accuracy on RoBERTa-Large.

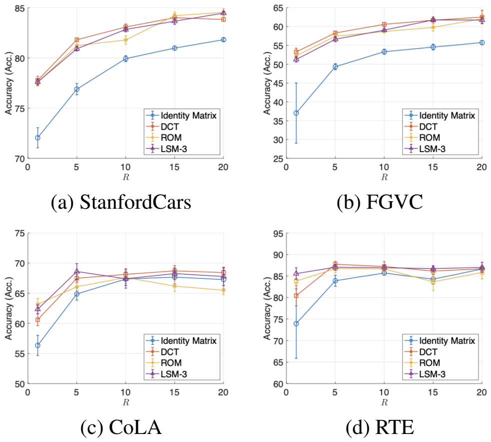
7.3. Commonsense Reasoning

To assess the generalization ability of ReFTA in commonsense reasoning, we evaluate three typical low-rank adaptation methods, including LoRA, PiSSA, and LoRETTA, on CommonsenseQA [32] using three widely adopted LLM backbones: LLaMA 2-7B [33], and LLaMA 3-8B [1], and Mistral-7B [17]. The results are summarized in Table 6. Across models, ReFTA $(r = 15)$ consistently surpasses the performance of PiSSA and LoRETTA, and achieves performance comparable to LoRA $(r = 8)$ while reducing the number of trainable parameters by approximately 94%. These findings highlight the strong parameter efficiency and effectiveness of ReFTA.

[2] https://huggingface.co/datasets/timm/oxford-iiit-pet

[3] https://huggingface.co/datasets/Multimodal-Fatima

[4] https://huggingface.co/datasets/timm/resisc45



(a) StanfordCars

(b) FGVC

(c) CoLA

(d) RTE

Figure 5. Ablation study of different invertible transforms on four datasets: (a) StanfordCars and (c) FGVC using `ViT-Large`, and (b) CoLA and (d) RTE using `RoBERTa-Large`.

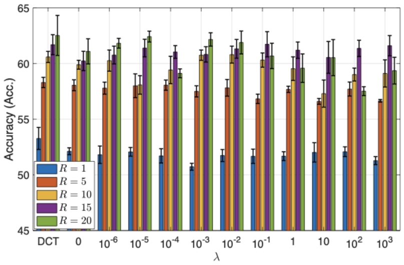


Figure 6. Effect of learning $\boldsymbol{U}_0$ on performance for the FGVC dataset using `ViT-Large`. λ denotes the coefficient of the regularization term $\lambda\|\boldsymbol{I} - \boldsymbol{U}_0^\top \boldsymbol{U}_0\|_F$, which is added to the original loss to encourage orthogonality. The bars labeled "DCT" correspond to the case where $\boldsymbol{U}_0$ is fixed to DCT and not trainable.

7.4. Ablation Study

We conduct an ablation study to investigate the impact of different $\boldsymbol{U}_0$ and R on the parameter efficiency and downstream task performance of ReFTA.

7.4.1. When the Invertible Linear Transform Is Given

We perform experiments with `RoBERTa-Large` and `ViT-Large` on four datasets: CoLA, RTE, StanfordCars, and FGVC. Four common invertible linear transforms are considered: (1) Identity Matrix, (2) DCT, (3) Random Cosine Transform (ROM), and (4) LSM-3. The comparison results are summarized in Fig. 5. As shown in the figure, DCT and LSM-3 consistently outperform the Identity Matrix baseline across all tasks, indicating that incorporating low-rank or frequency-domain invertible transforms can significantly enhance the parameter efficiency of ReFTA. The effect is more pronounced in vision tasks such as StanfordCars and FGVC, where DCT and LSM-3 yield improvements of up to 3–5%. In contrast, for NLU tasks like CoLA and RTE, the performance gap narrows as the rank R increases, but DCT and LSM-3 still demonstrate superior stability under smaller ranks. Overall, these findings highlight that carefully designed invertible transforms enable ReFTA to achieve better generalization while maintaining parameter efficiency.

7.4.2. When the Invertible Linear Transform Is Learned

To further examine whether it is necessary to learn the invertible transform, we conduct experiments where $\boldsymbol{U}_0$ is initialized with the DCT but set to be trainable. The results are presented in Fig. 6. As shown in the figure, allowing $\boldsymbol{U}_0$ to be learned brings only marginal improvements, suggesting that the predefined invertible transforms (e.g., DCT or LSM-3) are already well suited for ReFTA and that additional learning of $\boldsymbol{U}_0$ is not necessary.

8. Conclusions and Future Works

In this work, we present ReFTA, a tensor-based parameter-efficient fine-tuning (PEFT) approach with a single tensor-rank parameter for large models that achieves extremely low trainable parameter counts by leveraging the low-rank structure of weight tensors. To avoid the computational and memory overhead caused by repeated tensor reconstruction, ReFTA reformulates the fine-tuning process in a reconstruction-free manner by exploiting tensor algebraic properties rather than naively applying T-SVD. We provide a theoretical cost analysis and empirical memory comparisons, both confirming the efficiency of this design. Further quantization error analysis reveals ReFTA's remarkable potential for acceleration through quantization. In addition, we establish an upper bound on the expected test error of ReFTA, thereby providing a formal generalization guarantee for our approach. Experimental results further confirm that ReFTA achieves superior performance on downstream tasks such as image classification, natural language understanding, and commonsense reasoning, while requiring significantly fewer trainable parameters than existing PEFT methods. It is worth noting that ReFTA is primarily designed for stacking layers with compatible tensor shapes, which is the dominant setting in Transformer-based models. For layers with incompatible shapes, ReFTA does not enforce stacking but applies independent adaptations to each layer according to its native structure.

As T-SVD naturally generalizes SVD from matrices to tensors, many matrix-based PEFT methods can be readily extended to their tensorized forms. Our work provides valuable insights and techniques that help facilitate and inspire such extensions, making them more effective and principled. In future work, we plan to explore multi-subspace tensor methods to further enhance model expressiveness and generalization in parameter-efficient adaptation.

Acknowledgements

Yankai Cao acknowledges funding from discovery program of the Natural Science and Engineering Research Council of Canada (RGPIN-2019-05499) and New Frontiers in Research Fund (NFRFE-2022-00663). Zhouchen Lin was supported by the NSF China (No. 62276004). The authors also gratefully acknowledge the computing resources and services provided by Digital Research Alliance of Canada (www.alliancecan.ca), and Advanced Research Computing at the University of British Columbia.

References

[1] AI@Meta. Llama 3 model card. 2024. 7

[2] Daniel Bershatsky, Daria Cherniuk, Talgat Daulbaev, and Ivan Oseledets. Lotr: low tensor rank weight adaptation. *arXiv preprint arXiv:2402.01376*, 2024. 2

[3] Shubhankar Borse, Shreya Kadambi, Nilesh Pandey, Kartikeya Bhardwaj, Viswanath Ganapathy, Sweta Priyadarshi, Risheek Garrepalli, Rafael Esteves, Munawar Hayat, and Fatih Porikli. Foura: Fourier low-rank adaptation. In *Neural Information Processing Systems*, 2024. 1

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, and et al. Language models are few-shot learners. 2020. 1

[5] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, and et al. Palm: scaling language modeling with pathways, 2022. 1

[6] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000. 3

[7] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 5

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: pre-training of deep bidirectional transformers for language understanding, 2019. 1

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4, 6

[10] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936. 4

[11] Silvia Gandy, Benjamin Recht, and Isao Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problems*, 27(2):025010, 2011. 2

[12] Ziqi Gao, Qichao Wang, Aochuan Chen, Zijing Liu, Bingzhe Wu, Liang Chen, and Jia Li. Parameter-efficient finetuning with discrete fourier transform. *arXiv preprint arXiv:2405.03003*, 2024. 1, 6, 7

[13] Sajjad Ghiasvand, Yifan Yang, Zhiyu Xue, Mahnoosh Alizadeh, Zheng Zhang, and Ramtin Pedarsani. Communication-efficient and tensorized federated finetuning of large language models, 2025. 2

[14] Frank Lauren Hitchcock. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, (6):164–189, 1927. 2

[15] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp, 2019. 1

[16] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 1, 6

[17] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023. 7

[18] Eric Kernfeld, Misha Kilmer, and Shuchin Aeron. Tensor–tensor products with invertible linear transforms. *Linear Algebra and its Applications*, 485:545–570, 2015. 3, 4

[19] Misha E Kilmer and Carla D Martin. Factorization strategies for third-order tensors. *Linear Algebra and its Applications*, 435(3):641–658, 2011. 3

[20] Hao Kong, Canyi Lu, and Zhouchen Lin. Tensor q-rank: new data dependent definition of tensor rank. *Machine Learning*, 110:1867 – 1900, 2019. 4

[21] Xiang Lisa Li and Percy Liang. Prefix-tuning: optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021. 1

[22] Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 35(1):208–220, 2013. 2

[23] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: prompt tuning can be comparable to fine-tuning universally across scales and tasks, 2022. 1

[24] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: a robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 7

[25] Canyi Lu, Jiashi Feng, Zhouchen Lin, and Shuicheng Yan. Exact low tubal rank tensor recovery from gaussian measurements. *arXiv preprint arXiv:1806.02511*, 2018. 2

[26] Canyi Lu, Xi Peng, and Yunchao Wei. Low-rank tensor completion with a new tensor nuclear norm induced by invertible linear transforms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5996–6004, 2019. 2, 3

[27] Fanxu Meng, Zhaohui Wang, and Muhan Zhang. Pissa: principal singular values and singular vectors adaptation of large

language models. *arXiv preprint arXiv:2404.02948*, 2024. 1, 5, 6

[28] Mahdi Nikdan, Soroush Tabesh, Elvir Crnčević, and Dan Alistarh. Rosa: accurate parameter-efficient fine-tuning via robust adaptation. In *International Conference on Machine Learning*, pages 38187–38206, 2024. 1

[29] Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011. 2

[30] Wenjin Qin, Hailin Wang, Feng Zhang, Jianjun Wang, Xin Luo, and Tingwen Huang. Low-rank high-order tensor completion with applications in visual data. *IEEE Transactions on Image Processing*, 31:2433–2448, 2022. 2

[31] Chinmay Savadikar, Xi Song, and Tianfu Wu. Wegeft: Weight-generative fine-tuning for multi-faceted efficient adaptation of large models. In *International Conference on Machine Learning*, 2025. 2, 6

[32] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, 2019. 7

[33] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023. 7

[34] Ledyard R Tucker. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change*, 15(122-137):3, 1963. 2

[35] Mojtaba Valipour, Mehdi Rezagholizadeh, Ivan Kobyzev, and Ali Ghodsi. Dylora: parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation. *arXiv preprint arXiv:2210.07558*, 2022. 6

[36] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: a multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018. 7

[37] Shaowen Wang, Linxi Yu, and Jian Li. Lora-ga: Low-rank adaptation with gradient approximation. In *Neural Information Processing Systems*, 2024. 1

[38] Zhengbo Wang, Jian Liang, Ran He, Zilei Wang, and Tieniu Tan. Lora-pro: Are low-rank adapters properly optimized? 2025. 1, 2, 6

[39] Xiaoxia Wu, Cheng Li, Reza Yazdani Aminabadi, Zhewei Yao, and Yuxiong He. Understanding int4 quantization for language models: latency speedup, composability, and failure cases. In *International Conference on Machine Learning*, pages 37524–37539. PMLR, 2023. 5

[40] Yifan Yang, Jiajun Zhou, Ngai Wong, and Zheng Zhang. Loretta: Low-rank economic tensor-train adaptation for ultra-low-parameter fine-tuning of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3161–3176, 2024. 2, 6

[41] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: simple parameter-efficient fine-tuning for transformer-based masked language-models, 2022. 1, 6

[42] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Nikos Karampatziakis, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adalora: adaptive budget allocation for parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.10512*, 2023. 1, 6

[43] Yu-Bang Zheng, Ting-Zhu Huang, Xi-Le Zhao, Tai-Xiang Jiang, Teng-Yu Ji, and Tian-Hui Ma. Tensor n-tubal rank and its convex relaxation for low-rank tensor recovery. *Information Sciences*, 532:170–189, 2020. 2