SeMi-LoRA: Enhancing Low-Rank Adaptation via Separation and Mixing

Zhenfei Yang[1] , **Beiming Yu**[1] , **Peiqin Lin**[2] , **Yongkang Liu**[3] and **Deyi Xiong**[1]
[1]Tianjin University
[2]LMU Munich
[3]Northeastern University
yichendaojun@gmail.com, beiming403@163.com, lpq29743@gmail.com, misonsky@163.com, dyxiong@tju.edu.cn

Abstract

Low-Rank Adaptation (LoRA) has become a standard paradigm for parameter-efficient fine-tuning (PEFT). However, its low-rank constraint can limit its capacity to express complex weight updates, leaving a performance gap compared with full fine-tuning. Existing extensions improve expressivity, but they often sacrifice parameter mergeability or rely on sparse, block-diagonal updates that restrict global information flow. We propose **SeMi-LoRA** (Separation and Mixing LoRA), a novel framework that enables complete high-rank updates while preserving mergeability. SeMi-LoRA decomposes adaptation into channel-wise compression followed by two-stage latent mixing, namely Rank Mix and Channel Mix. Our structural analysis shows that, under a fixed base rank, the rank capacity of the update increases with the number of channels and supports complete updates rather than partial sparse ones. Extensive experiments on commonsense reasoning, mathematical reasoning, natural language understanding, and dialogue generation benchmarks demonstrate that SeMi-LoRA consistently outperforms strong PEFT baselines with favorable parameter efficiency.

1 Introduction

Adapting large language models (LLMs) to application scenarios usually presents a significant challenge, especially under the growing demands of model training, evaluation, and deployment [Guo *et al.*, 2023; Chen *et al.*, 2026; Jin *et al.*, 2024]. Full fine-tuning (FFT) is often impractical due to its high resource cost. Parameter-Efficient Fine-Tuning (PEFT) techniques have thus become a critical research area [Houlsby *et al.*, 2019], offering efficient solutions by optimizing a small subset of parameters while maintaining performance.

Among various PEFT methods, LoRA (Low Rank Adaptation) has emerged as one of the most widely adopted solutions due to its high flexibility and scalability [Hu *et al.*, 2022]. LoRA approximates the updated weights by adjusting a set of low-rank adapters while keeping the original model weights frozen, thereby effectively reducing the computational resources and memory overhead. Furthermore, unlike methods such as Adapter [Pfeiffer *et al.*, 2021] and P-Tuning [Li and Liang, 2021], LoRA's update parameters can be merged into the original model weights, eliminating additional inference latency. However, its low-rank approximation may introduce performance gaps [Ren *et al.*, 2024]. A straightforward way to address this is by increasing LoRA's rank, but this results in a linear increase in computation and memory costs.

Method	LoRA	LoRA-MoE	VeRA	MELoRA	SeMi-LoRA
High Rank		✓	✓	✓	✓
Efficiency (C)	✓			✓	✓
Efficiency (M)	✓	✓	✓	✓	✓
Mergeable	✓		✓	✓	✓
Complete Update	✓	✓	✓		✓

Table 1: Comparison of existing PEFT methods. SeMi-LoRA incorporates advantages across all dimensions, whereas other methods exhibit weaknesses in certain aspects. Efficiency (C): computation efficiency; Efficiency (M): memory efficiency.

To address this limitation, various solutions have been proposed. [Zadouri *et al.*, 2024] introduce the LoRA-MoE concept, which utilizes the Mixing of Experts (MoE) framework to integrate multiple LoRA modules for more flexible parameter updates. However, LoRA-MoE involves computationally complex nonlinear functions, which prevent the new parameters from being directly merged into the model weights. This increases both the inference cost and the difficulty of deployment. VeRA [Kopiczko *et al.*, 2024] shares high-rank matrices with lightweight scaling, but its restricted expressivity often demands more compute to match performance. Sparse approaches also target higher effective rank. MELoRA [Ren *et al.*, 2024] builds a high-rank block-diagonal update by composing multiple mini low-rank adapters, and MoRA [Jiang *et al.*, 2024] uses non-parametric operators with a shared high-rank matrix. However, these designs structurally restrict $\Delta\mathbf{W}$, limiting global information flow and preventing full updates across $\mathbf{W}$. Overall, existing methods tend to trade off mergeability, efficiency, or complete update support; Table 1 summarizes these limitations.

Our goal is to achieve efficient high-rank updates while preserving parameter mergeability and the capacity for complete updates. Inspired by Depthwise Separable Convolution [Howard *et al.*, 2017], we propose a novel PEFT method: SeMi-LoRA. Depthwise separable convolution separates a

standard convolution into depthwise and pointwise operations, reducing computational complexity while preserving strong representation capacity. Similarly, we separate feature mixing across channels and apply a two-stage mixing strategy to the separated features.

Specifically, SeMi-LoRA consists of four stages: Compress, Rank Mix, Channel Mix, and Decompress. We first split features into non-overlapping channels and apply per-channel low-rank compression. We then perform two linear mixers in the latent space: Rank Mix (intra-channel, along rank) and Channel Mix (inter-channel, at fixed rank indices), enabling both local and cross-channel interactions. Finally, Decompress projects features back to the original dimension, yielding a mergeable update that supports global modifications of $\mathbf{W}$ with only marginal overhead.

Overall, our main contributions are as follows:

- We propose a novel PEFT method, SeMi-LoRA, which achieves high-rank updates by separating and mixing LoRA matrices.
- We provide two complementary perspectives of SeMi-LoRA from feature mixing and matrix multiplication, and show that both MELoRA and LoRA are special cases of SeMi-LoRA.
- We conduct extensive experiments on commonsense reasoning, math reasoning, natural language understanding, and generation. Code and additional implementation details are available for reproducibility[1]. SeMi-LoRA consistently outperforms strong PEFT baselines, improving over LoRA by +5.02% and +7.58% absolute on commonsense and math reasoning, averaged across benchmarks.

2 Related Work

Parameter-Efficient Fine-Tuning (PEFT). With the widespread adoption of LLMs, fine-tuning them often requires substantial resources, which has driven the rapid development of Parameter-Efficient Fine-Tuning (PEFT) techniques. The core idea of PEFT is to enhance fine-tuning efficiency by selectively adjusting certain parameters of the model or introducing a small number of trainable parameters, without significantly increasing memory and computational demands [Houlsby *et al.*, 2019]. Early research includes methods such as adapter [Pfeiffer *et al.*, 2021], prefix tuning [Li and Liang, 2021], and prompt tuning [Lester *et al.*, 2021], however, these approaches may lead to issues such as non-mergeable incremental parameters and increased inference costs. Other methods improve efficiency through partial fine-tuning, such as BitFit [Ben Zaken *et al.*, 2022], or through feature scaling, such as IA3 [Liu *et al.*, 2022], and re-parameterization fine-tuning, such as LoRA [Hu *et al.*, 2022]. Notably, LoRA has become an important method for fine-tuning large-scale language models (LLMs) due to its high parameter efficiency and mergeable incremental parameters.

LoRA Extensions and Variations. LoRA has become a focal point of research owing to its effectiveness in fine-tuning large language models while demanding fewer computational resources. Building on LoRA's potential, researchers have proposed various extensions and optimization methods to enhance its performance and applicability. LoRA+ [Hayou *et al.*, 2024] refines learning rate strategies to improve training effectiveness. DoRA [Liu *et al.*, 2024] improves learning capacity and stability by decomposing weights into magnitude and direction. Furthermore, ReLoRA [Lialin *et al.*, 2023] aims to increase the rank of the final parameters by continuously integrating current-step LoRA updates into the LLM during training. Meanwhile, VeRA [Kopiczko *et al.*, 2024] further improves parameter efficiency by freezing the LoRA matrix and introducing trainable scaling vectors. Methods like LoRA-MoE [Zadouri *et al.*, 2024] and MELoRA [Ren *et al.*, 2024], through multi-LoRA expert ensembles, further enhance LoRA's flexibility and robustness. Related multilingual LLM studies also explore efficient training and LoRA-based knowledge transfer [Sun *et al.*, 2024; Dong *et al.*, 2025].

3 Preliminary

LoRA utilizes low-rank decomposition techniques to reparameterize model weight updates, allowing LLMs to adapt to new tasks. Formally, given a pre-trained weight matrix $\mathbf{W}_0 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, the expression for the weight update can be written as:

$$\mathbf{h} = (\mathbf{W}_0 + \Delta\mathbf{W})\mathbf{x} = \mathbf{W}_0\mathbf{x} + \mathbf{B}\mathbf{A}\mathbf{x} \tag{1}$$

where $\Delta\mathbf{W} \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ represents the low-rank weight updates. In LoRA, $\Delta\mathbf{W}$ is constructed as the product of two low-rank matrices $\mathbf{A} \in \mathbb{R}^{r \times d_{\text{in}}}$ and $\mathbf{B} \in \mathbb{R}^{d_{\text{out}} \times r}$, where the rank $r \ll \min(d_{\text{in}}, d_{\text{out}})$. During the training process, the original weight matrix $\mathbf{W}_0$ remains frozen, while $\mathbf{A}$ and $\mathbf{B}$ are optimized as trainable parameters. During inference, the low-rank weight update matrix can be merged into the original matrix to reduce latency in the inference process.

4 SeMi-LoRA

We propose SeMi-LoRA, a parameter-efficient fine-tuning (PEFT) framework that reformulates separation-and-mixing for low-rank weight updates to approximate high-rank updates while remaining fully linear and mergeable. The architecture decomposes the feature transformation into four linear stages: *Compress*, *Rank Mix*, *Channel Mix*, and *Decompress*. Specifically, input features are partitioned into c channels and projected into an r-dimensional latent subspace. Subsequently, the representation undergoes alternating intra-channel mixing (*Rank Mix*) and inter-channel mixing (*Channel Mix*), both stabilized by residual connections, before being projected back to the output space. Figure 1 illustrates the framework from both feature mixing and matrix multiplication perspectives.

4.1 Feature Mix View of SeMi-LoRA

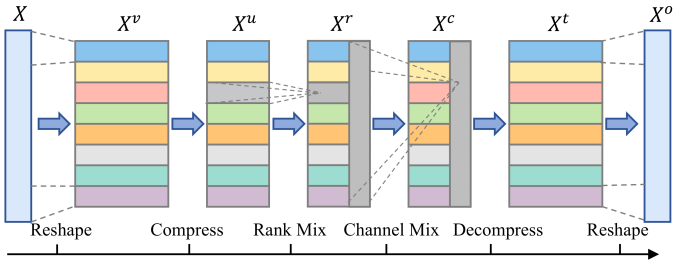
We use column vectors. Let the input feature be $\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$. SeMi-LoRA follows a split-transform-merge paradigm.

Stage 1: Compress

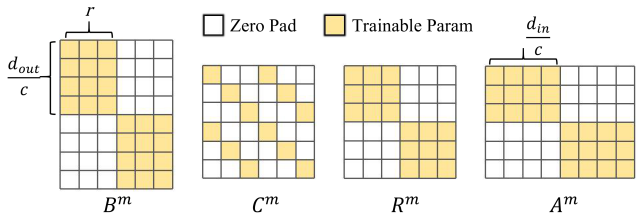
We reshape $\mathbf{x}$ into c non-overlapping channels:

$$\mathbf{x}^v = \eta_{c \times g}(\mathbf{x}) \tag{2}$$

[1]https://github.com/thinkingmanyangyang/peft-code



(a) Feature Mix View of SeMi-LoRA



(b) Matrix Multiplication View of SeMi-LoRA

```
Pseudocode of SeMi-LoRA in a PyTorch-like style

# inputs shape: [batch, h_in]
# c: channel size, r: rank size

# Compress
# A: [c, h_in/c, r]
x = x.reshape(batch, c, h_in/c)
comp_x = torch.einsum("bci,cij->bcj", x, A)

# Rank Mix
# R: [c, r, r], x_r: [b, c, r]
x_r = torch.einsum("bci,cij->bcj", comp_x, R)
x_r = x_r + comp_x

# Channel Mix
# C: [r, c, c], x_c: [b, c, r]
x_c = torch.einsum("bir,rij->bjr", x_r, C)
x_c = x_c + x_r

# Decompress
# B: [c, r, h_out/c]
decomp_x = torch.einsum("bci,cij->bcj", x_c, B)
x_out = decomp_x.reshape(b, h_out)
```

(c) Torch-like pseudo-code of SeMi-LoRA

Figure 1: Overview of SeMi-LoRA: (a) illustrates the feature mixing view of SeMi-LoRA, while (b) presents the matrix multiplication view, demonstrating the operation with a rank of 3 and a channel of 2. (c) presents the pseudo-code.

where $\eta_{c\times g}$ reshapes $\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$ into $\mathbf{x}^v = [\mathbf{x}_1^v, \ldots, \mathbf{x}_c^v] \in \mathbb{R}^{c\times g}$ with $g = d_{\text{in}}/c$. Here c is the number of channels.

For each channel $i \in \{1, \ldots, c\}$, we compress $\mathbf{x}_i^v \in \mathbb{R}^g$ into an r-dimensional latent subspace using a mini compression matrix $\mathbf{A}_i \in \mathbb{R}^{r\times g}$:

$$\mathbf{x}_i^u = \mathbf{A}_i \mathbf{x}_i^v, \quad i = 1, \ldots, c \tag{3}$$

obtaining $\mathbf{x}^u = [\mathbf{x}_1^u, \ldots, \mathbf{x}_c^u] \in \mathbb{R}^{c\times r}$. We refer to r as the *rank dimension* (latent dimension per channel).

Stage 2–3: Rank Mix and Channel Mix

To enable expressive yet efficient interactions in the latent space, we introduce two linear mixers with residual connections, inspired by the MLP-Mixer design [Tolstikhin *et al.*, 2021]. Rank Mix operates *within each channel* along the rank dimension, while Channel Mix operates *across channels* at each fixed rank index (without rank-to-rank mixing).

Rank Mix (intra-channel). For each channel i, we apply a trainable matrix $\mathbf{R}_i \in \mathbb{R}^{r\times r}$ with a residual connection:

$$\mathbf{x}_i^r = \mathbf{R}_i \mathbf{x}_i^u + \mathbf{x}_i^u, \quad i = 1, \ldots, c \tag{4}$$

Let $\mathbf{R} = [\mathbf{R}_1, \ldots, \mathbf{R}_c]$. Stacking all channels yields $\mathbf{x}^r = [\mathbf{x}_1^r, \ldots, \mathbf{x}_c^r] \in \mathbb{R}^{c\times r}$.

Channel Mix (inter-channel). Channel Mix exchanges information across channels while preserving rank indices. We transpose $\mathbf{x}^r \in \mathbb{R}^{c\times r}$ to obtain $\mathbf{x}^{r'} \in \mathbb{R}^{r\times c}$, where the t-th row $\mathbf{x}_t^{r'} \in \mathbb{R}^c$ corresponds to features at rank index t across all channels. We then mix channels independently for each rank index t:

$$\mathbf{x}_t^{c'} = \mathbf{C}_t \mathbf{x}_t^{r'} + \mathbf{x}_t^{r'}, \quad t = 1, \ldots, r \tag{5}$$

where $\mathbf{C} = [\mathbf{C}_1, \ldots, \mathbf{C}_r]$ and each $\mathbf{C}_t \in \mathbb{R}^{c\times c}$. After stacking all rows, we obtain $\mathbf{x}^{c'} \in \mathbb{R}^{r\times c}$ and transpose it back to $\mathbf{x}^c \in \mathbb{R}^{c\times r}$.

Stage 4: Decompress

Finally, we map the mixed latent features back to the output dimension. For each channel i, we use a mini decompression matrix $\mathbf{B}_i \in \mathbb{R}^{k\times r}$ with $k = d_{\text{out}}/c$:

$$\mathbf{x}_i^{o'} = \mathbf{B}_i \mathbf{x}_i^c, \quad i = 1, \ldots, c \tag{6}$$

and aggregate $\mathbf{x}^{o'} \in \mathbb{R}^{c\times k}$ into the output vector

$$\mathbf{x}^o = \gamma(\mathbf{x}^{o'}) \in \mathbb{R}^{d_{\text{out}}} \tag{7}$$

where $\gamma(\cdot)$ concatenates (or stacks) the c channel outputs back to the original output space.

4.2 Matrix Multiplication View and Mergeability

A key advantage of LoRA is *mergeability*: the learned update can be folded into the base weight, incurring no inference overhead. We now show that SeMi-LoRA preserves this property. Since all stages above are linear, the overall transformation is also linear and can be written explicitly as a product of structured matrices.

Define the block-diagonal compression and decompression matrices $\mathbf{A}^m \in \mathbb{R}^{cr\times d_{\text{in}}}$ and $\mathbf{B}^m \in \mathbb{R}^{d_{\text{out}}\times cr}$ as

$$\mathbf{A}^m = \operatorname{diag}_{i=1}^{c}(\mathbf{A}_i), \quad \mathbf{B}^m = \operatorname{diag}_{i=1}^{c}(\mathbf{B}_i). \tag{8}$$

Rank Mix is block-diagonal across channels:

$$\mathbf{R}^m = \operatorname{diag}_{i=1}^{c} \mathbf{R}_i \in \mathbb{R}^{cr\times cr} \tag{9}$$

The residual form in Eq. (4) corresponds to multiplying by $(\mathbf{R}^m + \mathbf{I})$, where $\mathbf{I} \in \mathbb{R}^{cr\times cr}$ is the identity matrix.

Channel Mix as a sparse matrix. In the feature view, Channel Mix transposes $\mathbf{x}^r \in \mathbb{R}^{c\times r}$ to $\mathbf{x}^{r'} \in \mathbb{R}^{r\times c}$ and mixes channels within each rank index. This transpose only permutes the same cr latent entries; therefore Channel Mix can

be represented as a structured sparse matrix $\mathbf{C}^m \in \mathbb{R}^{cr\times cr}$:

$$\mathbf{C}^m = \begin{bmatrix} \mathrm{diag}(\mathbf{C}_{:,1,1}) & \mathrm{diag}(\mathbf{C}_{:,1,2}) & \cdots & \mathrm{diag}(\mathbf{C}_{:,1,c}) \\ \mathrm{diag}(\mathbf{C}_{:,2,1}) & \mathrm{diag}(\mathbf{C}_{:,2,2}) & \cdots & \mathrm{diag}(\mathbf{C}_{:,2,c}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{diag}(\mathbf{C}_{:,c,1}) & \mathrm{diag}(\mathbf{C}_{:,c,2}) & \cdots & \mathrm{diag}(\mathbf{C}_{:,c,c}) \end{bmatrix}. \tag{10}$$

Here $\mathbf{C} \in \mathbb{R}^{r\times c\times c}$ is a tensor, and $\mathbf{C}_{:,i,j} \in \mathbb{R}^r$ stores the mixing weights from channel j to channel i at each rank index. Each (i, j) block is diagonal (no rank-to-rank mixing), matching Eq. (5). The residual form in Eq. (5) corresponds to multiplying by $(\mathbf{C}^m + \mathbf{I})$.

Putting everything together, the overall weight update of SeMi-LoRA can be written as

$$\Delta\mathbf{W} = \mathbf{B}^m(\mathbf{C}^m + \mathbf{I})(\mathbf{R}^m + \mathbf{I})\mathbf{A}^m \tag{11}$$

which is a linear operator and thus can be merged into the base weights. We now characterize its rank capacity. For any feasible parameters,

$$\begin{aligned} \mathrm{Rank}(\Delta\mathbf{W}) &= \mathrm{Rank}(\mathbf{B}^m(\mathbf{C}^m + \mathbf{I})(\mathbf{R}^m + \mathbf{I})\mathbf{A}^m) \\ &\le \min\Big\{ \mathrm{Rank}(\mathbf{B}^m), \mathrm{Rank}(\mathbf{C}^m + \mathbf{I}), \\ &\qquad \mathrm{Rank}(\mathbf{R}^m + \mathbf{I}), \mathrm{Rank}(\mathbf{A}^m)\Big\}. \end{aligned} \tag{12}$$

Since $\mathbf{A}^m = \mathrm{diag}_{i=1}^c \mathbf{A}_i$ and $\mathbf{B}^m = \mathrm{diag}_{i=1}^c \mathbf{B}_i$, the block-diagonal structure gives $\mathrm{Rank}(\mathbf{A}^m) = \sum_{i=1}^c \mathrm{Rank}(\mathbf{A}_i)$ and $\mathrm{Rank}(\mathbf{B}^m) = \sum_{i=1}^c \mathrm{Rank}(\mathbf{B}_i)$. When the mini adapters are full-rank in the low-dimensional subspace, i.e., $\mathrm{Rank}(\mathbf{A}_i) = \mathrm{Rank}(\mathbf{B}_i) = r$, we have $\mathrm{Rank}(\mathbf{A}^m) = \mathrm{Rank}(\mathbf{B}^m) = cr$. The residual mixers $(\mathbf{R}^m + \mathbf{I})$ and $(\mathbf{C}^m + \mathbf{I})$ are square operators in the same cr-dimensional latent space; except for degenerate cancellations, they do not reduce this latent rank. Therefore,

$$\mathrm{Rank}(\Delta\mathbf{W}) \le \min(d_{\mathrm{in}}, d_{\mathrm{out}}, cr). \tag{13}$$

This shows that the channel number c expands the rank capacity of the update while preserving the linear and mergeable structure of SeMi-LoRA.

4.3 Relations with MELoRA and LoRA

We relate SeMi-LoRA to LoRA variants via the expansion:

$$\Delta\mathbf{W} = \mathbf{B}^m(\mathbf{C}^m\mathbf{R}^m + \mathbf{R}^m + \mathbf{C}^m + \mathbf{I})\mathbf{A}^m.$$

Equivalently, this expansion contains the block-diagonal term $\mathbf{B}^m\mathbf{A}^m$ and the cross-mixing term $\mathbf{B}^m\mathbf{C}^m\mathbf{R}^m\mathbf{A}^m$, which introduces cross-block channel interactions. This expands the support of $\Delta\mathbf{W}$ beyond block-diagonal and enables non-local (cross-channel) updates, while the overall mapping remains linear and thus mergeable. The leading term $\mathbf{B}^m\mathbf{C}^m\mathbf{R}^m\mathbf{A}^m$ forms a non-zero $c \times c$ block matrix, allowing cross-channel blocks (i, j) to be updated instead of restricting $\Delta\mathbf{W}$ to independent block-diagonal groups.

MELoRA. Setting $\mathbf{R}^m = \mathbf{0}$ and $\mathbf{C}^m = \mathbf{0}$ disables both mixers and reduces the update to

$$\Delta\mathbf{W} = \mathbf{B}^m\mathbf{A}^m = \mathrm{diag}(\mathbf{B}_1\mathbf{A}_1, \ldots, \mathbf{B}_c\mathbf{A}_c) \tag{14}$$

Even with a high rank, the block-diagonal structure isolates input channels, preventing cross-channel interactions. SeMi-LoRA overcomes this by employing $\mathbf{C}^m$ and $\mathbf{R}^m$ for channel and subspace mixing. This enables non-zero off-diagonals, enhancing the expressivity of $\Delta\mathbf{W}$ without sacrificing mergeability.

LoRA. Standard LoRA uses $\Delta\mathbf{W} = \mathbf{BA}$ with $\mathbf{A} \in \mathbb{R}^{r\times d_{\mathrm{in}}}$ and $\mathbf{B} \in \mathbb{R}^{d_{\mathrm{out}}\times r}$. Let $\mathbf{A} = [\mathbf{A}_1, \ldots, \mathbf{A}_c]$ (column-wise) and $\mathbf{B} = [\mathbf{B}_1^\top, \ldots, \mathbf{B}_c^\top]^\top$ (row-wise), where $\mathbf{A}_j \in \mathbb{R}^{r\times g}$, $\mathbf{B}_i \in \mathbb{R}^{k\times r}$, $g = d_{\mathrm{in}}/c$, and $k = d_{\mathrm{out}}/c$. Set $\mathbf{R}^m = \mathbf{0}$ and choose $\mathbf{C}$ such that $\mathbf{C}_{:,i,i} = \mathbf{0}$ and $\mathrm{diag}(\mathbf{C}_{:,i,j}) = \mathbf{I}_r$ for all $i \ne j$. Then $\mathbf{B}^m(\mathbf{C}^m+\mathbf{I})\mathbf{A}^m$ produces blocks $(i, j) = \mathbf{B}_i\mathbf{A}_j$, yielding $\Delta\mathbf{W} = \mathbf{BA}$. Therefore, SeMi-LoRA exactly reduces to LoRA under this construction.

5 Experiment

We evaluated SeMi-LoRA across commonsense reasoning, mathematical reasoning, text classification, and text generation, covering over 20 benchmarks. Baselines include LoRA [Hu *et al.*, 2022], PiSSA [Meng *et al.*, 2024], DoRA [Liu *et al.*, 2024], VeRA [Kopiczko *et al.*, 2024], MoRA [Jiang *et al.*, 2024], HiRA [Huang *et al.*, 2025], and MELoRA [Ren *et al.*, 2024].

5.1 Commonsense Reasoning

Experiment Setup: We evaluated PEFT methods for commonsense reasoning by fine-tuning LLaMA2-7B [Touvron *et al.*, 2023] and LLaMA3-8B with SeMi-LoRA, LoRA, and other baselines on eight datasets. The datasets include BoolQ [Clark *et al.*, 2019], PIQA [Bisk *et al.*, 2020], SIQA [Sap *et al.*, 2019], HellaSwag [Zellers *et al.*, 2019], WinoGrande [Sakaguchi *et al.*, 2021], ARC-e, ARC-c [Clark *et al.*, 2018], and OBQA [Mihaylov *et al.*, 2018]. These tasks are all in the form of multiple-choice questions. We followed the experimental settings of [Hu *et al.*, 2023; Liu *et al.*, 2024], integrated the training datasets of all eight tasks during the training process, and evaluated the model performance on the independent test set of each task. The performance of all models is measured using accuracy (%), and the best model is selected based on the validation set loss. We followed the optimizer, scheduler, and training protocol used in the corresponding baseline papers.

Result: As summarized in Table 2, SeMi-LoRA achieves the best average performance among the compared PEFT methods on commonsense reasoning benchmarks. On LLaMA2-7B, SeMi-LoRA attains **82.63%** average accuracy, consistently surpassing partial-update approaches (e.g., MELoRA and MoRA) as well as high-rank variants such as HiRA under comparable parameter budgets. These results support our central hypothesis that a *complete* update formulation is crucial for faithful weight approximation in parameter-constrained regimes. Moreover, SeMi-LoRA exhibits a markedly improved efficiency–performance trade-off: it matches standard LoRA while using only one quarter of the trainable parameters. This advantage carries over to larger backbones: on LLaMA3-8B, SeMi-LoRA achieves **87.55%**, demonstrating strong scalability and robustness against competitive baselines.

Models	Method	Param	$r \times c$	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg.
ChatGPT	-	-	-	73.10	85.40	68.50	78.50	66.10	89.80	79.90	74.80	77.00
L2-7B	LoRA	56M	32	69.80	79.90	79.50	83.60	82.60	79.80	64.70	81.00	77.61
	PiSSA	56M	32	67.60	78.10	78.50	76.60	78.00	75.80	60.20	75.60	73.80
	VeRA	1M	1024	59.00	78.00	76.70	48.50	76.90	77.10	59.60	72.80	68.57
	MELoRA	56M	32×8	71.90	83.00	76.10	88.40	82.70	84.50	70.50	79.20	79.54
	DoRA	57M	32	71.80	83.70	76.00	**89.10**	82.60	83.70	68.20	82.40	79.69
	MoRA	56M	32	72.17	80.79	79.53	29.09	80.19	85.31	71.42	81.20	72.46
	HiRA	56M	32	71.22	83.35	79.53	88.12	83.98	86.74	73.81	84.60	81.42
	SeMi-LoRA	14M	8×4	67.82	81.18	77.79	86.64	84.06	86.06	71.84	81.20	79.57
	SeMi-LoRA	57M	32×8	**72.21**	**85.26**	**80.55**	88.91	**86.01**	**87.92**	**74.55**	**85.60**	**82.63**
L3-8B	LoRA	57M	32	70.80	85.20	79.90	91.70	84.30	84.20	71.20	79.00	80.79
	MELoRA	57M	32×8	73.88	89.34	82.14	92.61	88.56	93.31	82.59	**89.80**	86.53
	HiRA	57M	32	75.40	89.70	81.15	95.36	87.70	93.27	82.90	88.32	86.72
	SeMi-LoRA	58M	32×8	**77.22**	**89.72**	**82.45**	**95.65**	**89.58**	**93.34**	**84.52**	87.90	**87.55**

Table 2: Commonsense reasoning results on LLaMA2-7B (L2-7B) and LLaMA3-8B (L3-8B).

Models	Param	$r \times c$	SST-2	MRPC	CoLA	QNLI	RTE	STS-B	Avg.
Full FT	355M	-	96.4	90.9	68.0	94.7	86.6	92.4	88.2
Adpt	0.8M	-	**96.6**	89.7	67.8	94.8	80.1	91.9	86.8
LoRA	0.8M	8	96.2	90.2	68.2	94.8	85.2	92.3	87.8
VeRA	0.06M	256	96.1	90.9	68.0	94.4	85.9	91.7	87.8
MELoRA	0.8M	8×8	96.1	90.5	68.4	94.1	87.1	91.9	88.0
HiRA	0.8M	8×8	96.3	90.8	68.5	94.7	86.8	92.5	88.2
SeMi-LoRA	0.24M	2×4	96.2	91.1	68.2	**95.0**	86.2	92.1	88.1
SeMi-LoRA	0.9M	8×8	96.4	**91.6**	**69.5**	94.9	**87.4**	**92.5**	**88.7**

Table 3: RoBERTa-large model performance on GLUE benchmark. We report Matthew's correlation for CoLA, Pearson correlation for STS-B, and accuracy for the remaining tasks. Higher value is better for all metrics.

These results are consistent with the rank analysis in Section 4.2, which shows that channels can increase adaptation capacity without changing the base rank.

5.2 GLUE Benchmark

Experiment Setup: We employed the RoBERTa-large model on the GLUE benchmark [Wang *et al.*, 2018] to evaluate the effectiveness of our method on natural language understanding (NLU). GLUE is a benchmark that includes classification, similarity/paraphrase, and natural language inference tasks. Following VeRA [Kopiczko *et al.*, 2024], we evaluated on six GLUE tasks: CoLA, SST-2, MRPC, STS-B, QNLI, and RTE. We adopted the same evaluation metrics as [Hu *et al.*, 2022] to ensure a consistent comparison.

We performed five runs using different random seeds and reported the median, following the standard GLUE fine-tuning protocol used by the compared PEFT baselines.

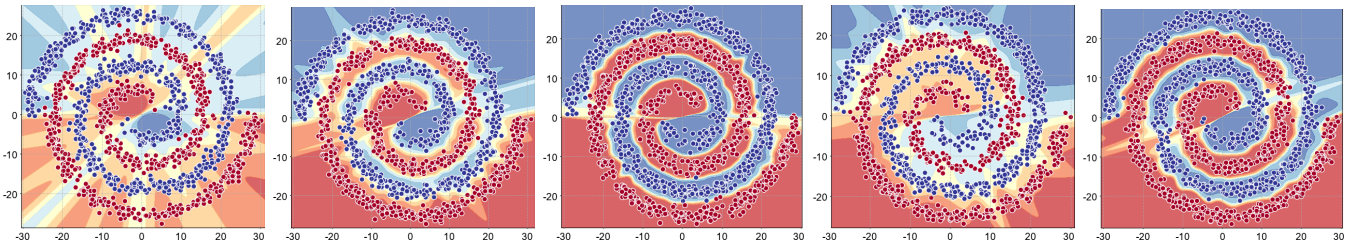
Result: Table 3 reports the GLUE results. With 0.9M trainable parameters, SeMi-LoRA obtains the highest average score among the compared PEFT methods, outperforming LoRA, MELoRA, and HiRA. Moreover, with only 0.24M trainable parameters, SeMi-LoRA remains close to the strongest baselines while using substantially fewer trainable parameters, suggesting favorable parameter efficiency on natural language understanding tasks.

5.3 Math Reasoning

Experiment Setup: Low-rank update methods (such as LoRA) often exhibit shortcomings when faced with math reasoning tasks that require deeper knowledge and reasoning abilities. These tasks depend on the model's knowledge accumulation and complex logical reasoning capabilities. LoRA's limitations lie in its low-rank update mechanism, which makes it difficult to effectively learn and store new knowledge [Jiang *et al.*, 2024]. To evaluate SeMi-LoRA on mathematical reasoning, we fine-tuned **LLaMA2-7B**, **Mistral**, and **Gemma** on GSM8K [Cobbe *et al.*, 2021] and MetaMathQA [Yu *et al.*, 2023], and reported exact-match accuracy on the corresponding test sets using the last checkpoint. All experiments fol-

Backbone	Method	Param	$r \times c$	GSM8K	MATH
L2-7B	LoRA	320M	128	42.30	5.50
	PiSSA	320M	128	53.07	7.44
	VeRA	1.6M	1024	45.92	6.30
	MELoRA	320M	128×8	49.55	6.96
	HiRA	320M	128	42.96	5.85
	SeMi-LoRA	21M	8×16	43.13	5.60
	SeMi-LoRA	351M	128×8	**54.43**	**8.52**
Mistral	LoRA	168M	64	67.70	19.68
	SeMi-LoRA	21.2M	8×8	69.92	20.66
	SeMi-LoRA	176M	64×8	**73.54**	**21.79**
Gemma	LoRA	200M	64	74.90	31.28
	SeMi-LoRA	25.2M	8×8	76.63	31.82
	SeMi-LoRA	207M	64×8	**78.41**	**32.15**

Table 4: Performance of different fine-tuning methods on GSM8K and MATH across backbones.



(a) LoRA Rank=8 (b) LoRA Rank=32 (c) LoRA Rank=64 (d) MELoRA Rank=8×8 (e) SeMi-LoRA Rank=8×8

Figure 2: Visualized fitting results of different PEFT methods on a two-spiral dataset. 8×8 means that rank 8 has been extended by 8 times through the rank extension mechanism.

Model	Method	Param	BLEU	BERT F1	BERT-R	BERT-P	Meteor	R-L	Avg.
L2-7B	P-Tuning	-	0.60	83.29	83.33	83.28	15.11	12.36	46.33
	MoRA	56M	1.09	84.09	84.65	83.59	10.97	9.57	45.66
	LoRA	56M	1.82	84.41	84.71	84.16	11.38	10.55	46.17
	DoRA	56M	1.73	84.18	84.61	83.81	11.25	10.41	46.00
	HiRA	56M	2.70	84.86	84.98	84.77	13.56	12.80	47.28
	SeMi-LoRA	57M	2.82	85.73	85.68	85.62	14.31	13.75	47.99
L3-8B	P-Tuning	-	1.50	81.52	81.07	82.01	15.49	13.55	45.86
	MoRA	57M	1.60	84.22	84.06	84.43	12.37	11.19	46.31
	LoRA	57M	2.26	84.32	84.00	84.67	12.51	11.77	46.59
	DoRA	57M	2.29	84.32	84.06	84.62	12.63	11.78	46.62
	HiRA	57M	3.41	84.81	84.40	85.25	14.87	14.05	47.80
	SeMi-LoRA	58M	3.63	85.63	85.24	85.72	16.47	14.87	48.59

Table 5: ConvAI2 Dialogue Generation Results (BLEU, BERTScore P/R/F1, METEOR, ROUGE-L).

lowed the optimizer, scheduler, and training protocol of [Meng *et al.*, 2024].

Result: Table 4 shows that SeMi-LoRA consistently improves mathematical reasoning across backbones. On LLaMA2-7B, SeMi-LoRA achieves the best GSM8K and MATH scores among the compared PEFT methods. The gains also transfer to Mistral and Gemma, where SeMi-LoRA improves over LoRA under both parameter-efficient and comparable-parameter settings.

5.4 Open-Domain Dialogue Generation (ConvAI2)

Experiment Setup: We evaluated SeMi-LoRA on the ConvAI2 dataset [Dinan *et al.*, 2019], which comprises 17,878 multi-turn dialogues for training and 1,000 for testing. Each example includes persona profiles and the dialogue history. Following prior work [Song *et al.*, 2021], we adopted the self-persona setting, where only the speaker's own persona is provided as context. Experiments were conducted with two backbones, LLaMA-2-7B and LLaMA-3-8B, and we compared SeMi-LoRA against strong PEFT baselines. For evaluation, we reported BLEU [Papineni *et al.*, 2002] and BERTScore [Zhang *et al.*, 2019] to assess generation quality and semantic similarity, and additionally included METEOR and ROUGE-L for a more comprehensive comparison. We followed the optimizer, scheduler, and training protocol used by [Huang *et al.*, 2025].

Result: The results are summarized in Table 5. Overall, SeMi-LoRA delivers strong performance with both backbones and remains competitive with all baselines under comparable parameter budgets. Relative to standard LoRA and its variants (e.g., MoRA and HiRA), SeMi-LoRA consistently improves generation-focused metrics, including BLEU, METEOR, and ROUGE-L. These gains highlight the effectiveness of SeMi-LoRA for persona-grounded open-domain dialogue generation.

6 Analysis

6.1 Does SeMi-LoRA's High Rank Improve Model Fitting Ability?

To examine whether SeMi-LoRA improves fitting ability through higher-rank updates, we constructed a two-spiral dataset with 1500 samples and used a multi-layer MLP to fit the non-linear decision boundary. The weights $\mathbf{W}$ of the MLP linear layers were frozen, and different PEFT methods were used to approximate the update matrix $\Delta\mathbf{W}$. We trained all methods for 200 epochs with learning rate 5e-4.

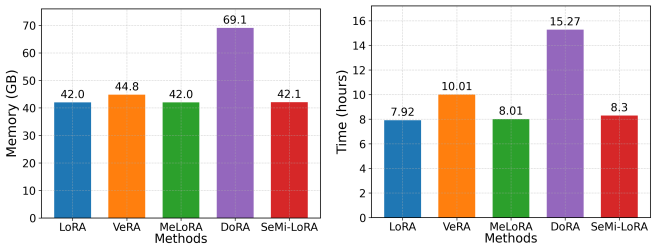
Figure 2 visualizes the fitting results. Comparing LoRA Rank 8, Rank 32, and Rank 64 shows that increasing the rank improves the fitting of non-linear data. However, MELoRA, despite its partial high-rank design, remains limited because its block-wise update restricts global interactions. In contrast, SeMi-LoRA with rank 8×8 produces a decision boundary close to LoRA Rank 64, indicating that its separation-and-mixing design effectively improves high-rank update capacity.

6.2 How is SeMi-LoRA's System Efficiency?

In this section, we explore the practical performance of the SeMi-LoRA method with a focus on system efficiency, primarily examining its performance in terms of GPU memory consumption and training latency. We compare the resource consumption of SeMi-LoRA with LoRA, VeRA, MELoRA, and DoRA when fine-tuning LLaMA2-7B (using the Commonsense Reasoning dataset) on an Nvidia A100 GPU. We use the same setup as the commonsense reasoning experiments.

As shown in Figure 3a, SeMi-LoRA, LoRA, VeRA, and MELoRA exhibit similar GPU memory consumption, indicating that SeMi-LoRA's GPU memory requirements are comparable to those of low-rank methods such as LoRA, without adding extra burden. DoRA, due to its complex parameter learning strategy, has higher memory consumption.

Figure 3b compares the training time of different methods. SeMi-LoRA, LoRA, and MELoRA show similar training durations, indicating that SeMi-LoRA remains computationally efficient despite the additional mixing stages. VeRA incurs slightly higher training time, possibly due to its very high-rank



(a) GPU memory cost (b) Time cost

Figure 3: GPU memory and time cost comparison on the commonsense reasoning dataset. Despite two extra mixing stages, SeMi-LoRA adds only slight overhead over LoRA, while remaining more efficient than other high-rank PEFT methods.

Methods	SST-2	MRPC	CoLA	RTE
w/o C-Mix	96.3	90.8	68.7	87.2
w/o R-Mix	96.2	91.2	69.1	86.1
w/o Res	96.2	91.3	69.3	86.9
C-Mix->R-Mix	96.0	91.4	**69.7**	87.1
R-Mix->C-Mix	**96.4**	**91.6**	69.5	**87.4**

Table 6: Impact of key components on selected GLUE tasks. C-Mix denotes Channel Mix, R-Mix denotes Rank Mix, and Res denotes residual connection.

auxiliary frozen matrix, while DoRA shows more obvious training latency. This efficiency comes from the small additional computation in the compact latent space. Moreover, our `einsum`-based Channel Mix avoids large-scale activation permutes/transposes, reducing extra memory movement.

In summary, SeMi-LoRA surpasses the strongest existing baseline on commonsense reasoning while retaining LoRA-like memory usage and training throughput; compared to DoRA, it cuts training time by 54.4%, highlighting SeMi-LoRA as a practical and efficient high-rank extension of LoRA.

6.3 What's the Component Effect on SeMi-LoRA?

To investigate the impact of feature separation and mixing strategies (Channel Mix and Rank Mix) in SeMi-LoRA on performance, we conducted a series of ablation experiments. All variants used the same backbone and training protocol, isolating the effect of each component. The experimental results are shown in Table 6. After removing the Rank Mix (R-Mix) mechanism and the Channel Mix (C-Mix) mechanism, the model performance decreased. This indicates that Rank Mix and Channel Mix are indispensable components in SeMi-LoRA. The experimental results show that after removing the Residual Connection (Res) structure, the model performance slightly decreased. This suggests that the Residual Connection structure may play a role in integrating the advantages of various mixing methods. We also conducted experiments to swap the order of Rank Mix and Channel Mix. The results show that the order of Rank Mix preceding Channel Mix yields relatively better performance, but overall the performance is comparable.

C-Mix Init.	R-Mix Init.	SST-2	MRPC	CoLA	RTE
Kaiming	Orthogonal	95.9	90.5	68.1	85.8
Kaiming	Kaiming	96.2	91.2	68.5	86.3
All One	Orthogonal	**96.4**	90.8	69.3	**87.8**
All One	Kaiming	**96.4**	**91.6**	**69.5**	87.4

Table 7: Impact of different initialization strategies on selected GLUE tasks.

6.4 How Does Initialization Impact Model Performance?

In this section, to investigate the impact of different initialization strategies on SeMi-LoRA performance, we designed various schemes and evaluated them on the GLUE Benchmark. We followed LoRA's initialization strategy, where matrix $\mathbf{A}$ in the LoRA decomposition is initialized with the Kaiming uniform distribution and matrix $\mathbf{B}$ is initialized with the zero matrix. For the Rank Mix module specific to SeMi-LoRA, we compared the initialization effects of orthogonal initialization and the Kaiming uniform distribution. For the Channel Mix module, we compared the differences between the Kaiming uniform distribution and all-one initialization. The experimental results of Table 7 show that SeMi-LoRA achieves optimal performance when the Rank Mix is initialized with the Kaiming uniform distribution, and the Channel Mix is initialized with all ones.

7 Conclusion

We have presented SeMi-LoRA, a method for improving the performance and efficiency of parameter-efficient fine-tuning on complex tasks. The core idea of SeMi-LoRA lies in a separation-and-mixing strategy, including Rank Mix and Channel Mix, which enables information interaction across feature dimensions, realizing a high-rank approximation of the parameter increment matrix. SeMi-LoRA combines feature-mixing and matrix-multiplication views, and supports merging new parameters into the original model weights during inference, achieving zero additional inference overhead. Extensive experiments on reasoning, understanding, and generation tasks validate the effectiveness of SeMi-LoRA, while analyses explore its mechanisms and efficiency under different settings.

8 Limitations

SeMi-LoRA demonstrates strong performance but introduces the channel number c as an additional hyperparameter, which may require tuning for new tasks or backbones. Rank Mix and Channel Mix also slightly increase implementation complexity compared with standard LoRA, although empirical memory and latency overhead remains small. Future work will investigate automatic channel selection and adaptive rank-channel allocation.

Acknowledgments

The present research was supported by the National Key Research and Development Program of China (Grant No. 2024YFE0203000). We would like to thank the anonymous reviewers for their insightful comments.

Contribution Statement

Zhenfei Yang and Beiming Yu contributed equally to this work. Deyi Xiong is the corresponding author.

References

[Ben Zaken *et al.*, 2022] 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, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[Bisk *et al.*, 2020] Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: reasoning about physical commonsense in natural language. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7432–7439. AAAI Press, 2020.

[Chen *et al.*, 2026] Zhuo Chen, Yuxuan Miao, and Deyi Xiong. Data mixing for large language models pretraining: A survey and outlook. *Data Intelligence*, 8(1):15–55, March 2026.

[Clark *et al.*, 2018] Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

[Clark *et al.*, 2019] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*, 2019.

[Cobbe *et al.*, 2021] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

[Dinan *et al.*, 2019] Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, et al. The second conversational intelligence challenge (convai2). In *The NeurIPS'18 Competition: From Machine Learning to Intelligent Conversations*, pages 187–208. Springer, 2019.

[Dong *et al.*, 2025] Tianyu Dong, Bo Li, Jinsong Liu, Shaolin Zhu, and Deyi Xiong. MLAS-LoRA: Language-aware parameters detection and LoRA-based knowledge transfer for multilingual machine translation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15645–15660, Vienna, Austria, July 2025. Association for Computational Linguistics.

[Guo *et al.*, 2023] Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Supryadi, Linhao Yu, Yan Liu, Jiaxuan Li, Bojian Xiong, and Deyi Xiong. Evaluating large language models: A comprehensive survey, 2023.

[Hayou *et al.*, 2024] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. Lora+: Efficient low rank adaptation of large models. *arXiv preprint arXiv:2402.12354*, 2024.

[Houlsby *et al.*, 2019] 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.

[Howard *et al.*, 2017] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: efficient convolutional neural networks for mobile vision applications (2017). *arXiv preprint arXiv:1704.04861*, 126, 2017.

[Hu *et al.*, 2022] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[Hu *et al.*, 2023] Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276, 2023.

[Huang *et al.*, 2025] Qiushi Huang, Tom Ko, Zhan Zhuang, Lilian Tang, and Yu Zhang. Hira: Parameter-efficient hadamard high-rank adaptation for large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[Jiang *et al.*, 2024] Ting Jiang, Shaohan Huang, Shengyue Luo, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. Mora: High-rank updating for parameter-efficient fine-tuning, 2024.

[Jin *et al.*, 2024] Renren Jin, Jiangcun Du, Wuwei Huang, Wei Liu, Jian Luan, Bin Wang, and Deyi Xiong. A comprehensive evaluation of quantization strategies for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12186–12215, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[Kopiczko *et al.*, 2024] Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. Vera: Vector-based random matrix adaptation. In *The Twelfth International Conference on Learning Representations*, 2024.

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

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

[Lialin *et al.*, 2023] Vladislav Lialin, Sherin Muckatira, Namrata Shivagunde, and Anna Rumshisky. Relora: High-rank training through low-rank updates. In *The Twelfth International Conference on Learning Representations*, 2023.

[Liu *et al.*, 2022] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin A Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *Advances in Neural Information Processing Systems*, 35:1950–1965, 2022.

[Liu *et al.*, 2024] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: Weight-decomposed low-rank adaptation. *arXiv preprint arXiv:2402.09353*, 2024.

[Meng *et al.*, 2024] 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.

[Mihaylov *et al.*, 2018] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. *arXiv preprint arXiv:1809.02789*, 2018.

[Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[Pfeiffer *et al.*, 2021] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, 2021.

[Ren *et al.*, 2024] Pengjie Ren, Chengshun Shi, Shiguang Wu, Mengqi Zhang, Zhaochun Ren, Maarten Rijke, Zhumin Chen, and Jiahuan Pei. Melora: Mini-ensemble low-rank adapters for parameter-efficient fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3052–3064, 2024.

[Sakaguchi *et al.*, 2021] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

[Sap *et al.*, 2019] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan LeBras, and Yejin Choi. Socialiqa: Commonsense reasoning about social interactions. *arXiv preprint arXiv:1904.09728*, 2019.

[Song *et al.*, 2021] Haoyu Song, Yan Wang, Kaiyan Zhang, Weinan Zhang, and Ting Liu. Bob: Bert over bert for training persona-based dialogue models from limited personalized data. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (volume 1: long papers)*, pages 167–177, 2021.

[Sun *et al.*, 2024] Haoran Sun, Renren Jin, Shaoyang Xu, Leiyu Pan, Supryadi, Menglong Cui, Jiangcun Du, Yikun Lei, Lei Yang, Ling Shi, Juesi Xiao, Shaolin Zhu, and Deyi Xiong. FuxiTranyu: A multilingual large language model trained with balanced data. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1499–1522, Miami, Florida, US, November 2024. Association for Computational Linguistics.

[Tolstikhin *et al.*, 2021] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 24261–24272. Curran Associates, Inc., 2021.

[Touvron *et al.*, 2023] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[Wang *et al.*, 2018] 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.

[Yu *et al.*, 2023] Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*, 2023.

[Zadouri *et al.*, 2024] Ted Zadouri, Ahmet Üstün, Arash Ahmadian, Beyza Ermis, Acyr Locatelli, and Sara Hooker. Pushing mixture of experts to the limit: Extremely parameter efficient moe for instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024.

[Zellers *et al.*, 2019] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.

[Zhang *et al.*, 2019] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.