Multiplicative Mixture of Experts for Rank-Efficient LLM Finetuning

Francesco Carzaniga [1] **Michael Hersche** [1] **David Cox** [1] **Abbas Rahimi** [1]

Abstract

Large language models (LLMs) have achieved impressive results in many general-purpose domains, but their performance on specific tasks can still be improved through finetuning. Parameter-efficient finetuning (PEFT) tailors an LLM to one or more tasks through a small amount of trainable parameters, requiring reduced computational resources. On one hand, techniques like low-rank adaptation (LoRA) provide the required parameter efficiency with *additive* adapters of low, and fixed, rank, which limits their flexibility. On the other hand, mixture of experts (MoEs) enhance the flexibility of a model at the cost of parameter count and memory budget. The combination of the two approaches, parameter-efficient MoEfication, has shown promise in addressing the issues of both. In this work, we show that **replacing additive with multiplicative interactions** improves the rank-efficiency of PEFT adapters, increasing the flexibility and reducing the number of parameters involved in MoEfication. In particular, **our multiplicative method is especially effective when memory becomes the bottleneck**, an increasingly common scenario with large-scale MoEs. We find that our quantum-inspired multiplicative method, OperA, is optimal given the same parameter budget for 7/8 models considered, using fewer or equal parameters than the baseline. In a multi-task setting, OperA fully surpasses all baselines on all models. We also propose an algorithm to optimize the use of OperA in large-scale deployments by alleviating its computational complexity. Finally, we provide evidence that OperA surpasses the effective rank of competing solutions by more than two orders of magnitude.

[1]IBM Research. Correspondence to: Francesco Carzaniga <frc@zurich.ibm.com>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

1. Introduction

Large language models (LLMs) have demonstrated impressive capabilities (OpenAI, 2025; 2024; Qwen AI, 2025a; Meta AI, 2024) when generalizing to novel tasks, making them the models of choice for a variety of domains (Wang et al., 2024; Liu et al., 2024a; 2023). Still, LLMs can be further finetuned to improve their performance in challenging scenarios (Hu et al., 2022; Parthasarathy et al., 2024). However, full finetuning remains prohibitively expensive in many cases (Xia et al., 2024b), motivating parameter-efficient finetuning (PEFT). Among PEFT methods (Han et al., 2024; Hu et al., 2022; Liu et al., 2024b), low-rank adaptation (LoRA; Hu et al. 2022) is particularly effective but relies on fixed low-rank additive adapters, which limits its flexibility. Indeed, while LoRA is often on par with its full-rank counterparts, its performance degrades when tasks can no longer be comfortably represented in low-rank subspaces (Feng et al., 2024; Xia et al., 2024a)

At the same time, mixture of experts (MoEs; Shazeer et al. 2017; Fedus et al. 2022) has been widely adopted in recent LLM architectures as a technique to improve performance across tasks without increasing the active parameter count (Jiang et al., 2024; IBM, 2024; DeepSeek-AI, 2025). MoEs leverage multiple linear models (i.e., experts) that process tokens in parallel, offering a factorization of large spaces into manageable chunks. By instantiating a large number of experts, but only activating a few of them for each token, LLMs can be internally specialized while keeping throughput high. However, in practice MoEs still suffer from increased memory requirements, as all experts need to be kept in memory even when not in use.

Finally, LoRAs and MoEs can be combined to obtain a PEFT approach that is parameter-efficient, memory-efficient, and highly flexible (Tian et al., 2024; Li et al., 2024; Zeng et al., 2025). In particular, each expert is represented as a low-rank linear model, while maintaining the same gating and routing structure of regular MoEs (see Figure 1a), which have already been shown to be sufficiently powerful.

In this work, we show that replacing the additive interactions of LoRA with multiplicative interactions **increases the *effective rank* (Roy & Vetterli, 2007) of the parameter-efficient MoE layer without increasing its parameter**

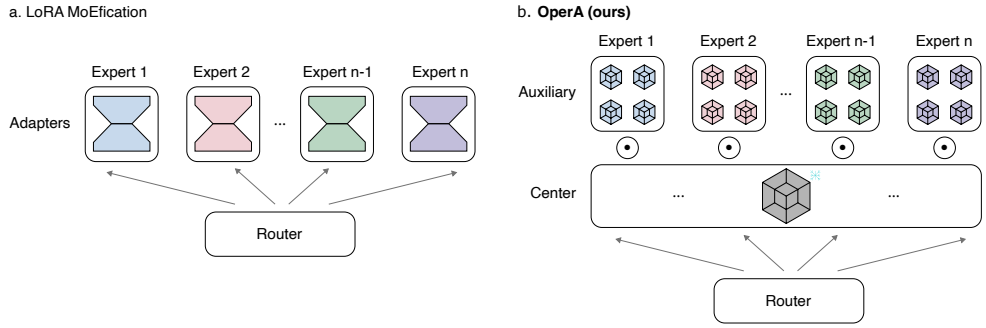


Figure 1. **Rank-efficient mixture of experts. a.** Regular LoRA MoEfication uses one low-rank adapter per expert. **b.** The experts of OperA are instead built through a multiplicative combination of 4D cores, obtained by matrix product operator (MPO) decomposition.

count, thereby increasing the overall efficiency of PEFT. In particular, multiplicative interactions offer a compute bound, rather than memory bound architecture. In this context, we provide an algorithm that is designed to take advantage of modern accelerators by leveraging the reordering of tensor contractions, notably decreasing the required number of operations. We find that our proposed multiplicative approach is most advantageous when memory, not compute, is the dominant bottleneck: a regime increasingly common in large-scale MoE.

In particular, we introduce OperA (see Figure 1b), a PEFT technique that **does away with the fixed-(and low-)rank adapters of LoRA by leveraging multiplicative interactions** with the model weights. We borrow matrix product operators (MPOs; Pirvu et al. 2010; Schollwöck 2011; Orús 2014) from the quantum physics literature to provide an efficient multiplicative decomposition of arbitrary matrices (originally, Hamiltonians) into *core* and *auxiliary* 4D tensors. Through the MPO configuration we can concentrate the majority of the weights onto a frozen core tensor, while learning the leaner auxiliary tensors during training. Therefore, the original weights participate in the construction of the OperA adapter.

We evaluate OperA on a comprehensive set of 8 tasks, including ARC (Clark et al., 2018), PIQA (Bisk et al., 2020), OpenBookQA (Mihaylov et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), and PubmedQA (Jin et al., 2019) comparing them with MoLA (Gao et al., 2025), LoRAMoE (Dou et al., 2024), MixLoRA (Li et al., 2024), and HydraLoRA (Tian et al., 2024), a variety of openly available recent techniques for LoRA MoEfication. We test all approaches on multiple models of different sizes and from different sources to ensure the consistency of our evaluation, including Llama3, Llama3.1, Llama3.2 (Meta AI, 2024), Phi4 (Abdin et al., 2024), and Qwen2.5 (Qwen AI, 2025b). We find that OperA achieves comparable or superior results to the additive adapters and notably increases the effective rank as well, all while reducing the required number of parameters by up to 61% against the most parameter-efficient alternative (MixLoRA) and by up to 70% against the least efficient (MoLA). Moreover, OperA achieves higher accuracy on 7/8 models when given the same parameter budget and also higher accuracy on all models in a multi-task setting.

2. Background

Low-rank adaptation (LoRA). LoRA is based on low-rank approximations of a matrix. In particular, given a pretrained weight $W \in \mathbb{R}^{M\times N}$, LoRA consists of two matrices $A \in \mathbb{R}^{M\times r}$ and $B \in \mathbb{R}^{r\times N}$, with $r \ll M, N$ being the LoRA *rank*. Crucially, while the LoRA rank places an upper bound on the dimensionality of the adapter, the actual subspace spanned by the adapter might be notably smaller. We explore this effect in detail through the use of the *effective rank* in Section 4.5.

Next, the product $AB \in \mathbb{R}^{M\times N}$ results in a shape-compatible matrix with W that only requires a fraction of its parameters to store, producing considerable space savings. This comes to cost of the rank of AB, which is limited to r, and therefore to its flexibility. The final weight matrix becomes

$$W' = W + AB \tag{1}$$

Mixture of experts (MoEs). MoEs increase the flexibility and expressivity of LLMs, while maintaining a constrained increase in the number of parameters. Experts are matrices $E_1, ..., E_n \in \mathbb{R}^{M\times N}$ that independently operate on a given input vector $\boldsymbol{x} \in \mathbb{R}^M$, factorizing a large representation space into multiple additive components. The most popular implementation of MoEs uses a noisy top-k routing mechanism that selects k out of n experts for each $\boldsymbol{x}$, resulting in considerable computational savings at the cost of memory. Specifically, given a router T

$$y = \sum_i^n T(\boldsymbol{x})_i E_i(\boldsymbol{x}) \tag{2}$$

Table 1. **Accuracy of PEFT MoEfication on the downstream tasks.** For each model, we report the number of parameters and the task accuracy.

Model	Adapter	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	HS ↑	WG ↑	PMQA ↑	Avg ↑
Llama3-8B	None	/	66.04	81.19	62.24	67.80	59.27	48.43	67.80	54.40	63.40
	LoRAMoE	246M	77.56	86.62	87.60	87.80	74.33	95.21	84.29	73.10	83.31
	MixLoRA	241M	75.76	87.08	86.18	89.00	73.61	94.63	83.11	55.00	80.55
	OperA	173M	77.39	86.28	88.30	87.60	76.36	96.15	81.45	74.10	**83.45**
Llama3.1-8B	None	/	68.34	81.19	67.08	69.20	57.37	55.35	52.72	58.30	63.69
	LoRAMoE	246M	79.52	85.50	87.32	86.20	74.46	92.50	82.32	73.20	82.63
	MixLoRA	241M	76.96	86.82	86.40	83.20	74.65	94.77	84.14	54.40	80.17
	OperA	173M	76.88	86.32	87.92	85.20	75.23	96.10	80.58	74.70	**82.87**
Llama3.2-3B	None	/	53.41	66.79	63.82	55.00	57.61	45.98	51.93	55.30	56.23
	LoRAMoE	133M	67.92	83.29	82.97	77.00	71.68	90.62	74.22	69.90	77.20
	MixLoRA	131M	68.26	83.12	83.02	77.40	71.50	92.13	75.14	68.10	**77.33**
	OperA	92M	66.98	81.73	83.19	77.20	70.98	92.93	74.19	71.10	77.29
Phi4-14B	None	/	79.95	84.76	90.04	79.40	68.72	87.19	77.74	72.90	80.09
	LoRAMoE	342M	92.02	94.11	92.44	94.00	74.57	95.61	87.53	71.40	87.71
	MixLoRA	336M	91.21	94.28	91.90	92.60	75.47	95.95	88.79	73.40	**87.95**
	OperA	182M	88.14	92.84	91.84	91.00	74.49	95.25	86.27	70.60	86.30
Qwen2.5-3B	None	/	77.65	87.58	75.46	75.00	63.64	72.08	57.38	58.80	70.95
	LoRAMoE	192M	80.11	88.34	85.58	85.30	70.70	91.57	78.93	70.50	81.38
	MixLoRA	188M	80.12	88.64	86.62	87.60	70.92	91.67	80.03	70.40	**82.00**
	OperA	74M	80.89	89.40	86.02	87.60	70.06	92.99	74.82	70.60	81.55
Qwen2.5-7B	None	/	85.07	91.33	81.99	84.60	63.09	86.00	58.64	63.10	76.73
	LoRAMoE	258M	87.03	91.79	89.83	92.00	74.22	95.30	84.93	71.60	85.84
	MixLoRA	253M	87.88	94.19	88.74	92.80	74.52	95.29	83.27	52.70	83.67
	OperA	192M	87.71	93.39	90.42	91.40	74.74	95.91	83.66	72.50	**86.22**
Qwen2.5-14B	None	/	89.42	93.10	88.79	86.80	70.61	91.12	69.22	67.60	82.08
	LoRAMoE	384M	90.66	93.42	91.46	96.20	76.69	97.01	91.32	74.10	88.86
	MixLoRA	376M	90.78	93.73	92.27	95.60	77.58	96.82	90.21	53.10	86.26
	OperA	200M	90.61	93.73	93.42	96.00	78.10	97.43	90.13	75.20	**89.33**
Qwen2.5-32B	None	/	92.24	94.23	89.77	89.80	70.03	94.21	77.35	62.10	83.72
	LoRAMoE	858M	93.43	94.32	94.23	96.40	78.26	97.24	92.23	53.40	87.37
	MixLoRA	841M	93.26	95.75	94.07	97.00	78.35	97.29	90.92	73.20	89.98
	OperA	343M	93.94	94.49	94.29	97.40	79.51	97.38	90.53	74.20	**90.22**

with

$$T(\boldsymbol{x})_i = \begin{cases} t_i & \text{if expert } i \text{ has been selected} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

and $\sum_i^n t_i = 1$ are the routing weights, usually enforced via a *softmax* applied over the router outputs.

Matrix product operators (MPO). MPOs, in the setup used in this work, form a multiplicative representation of an arbitrary matrix $W \in \mathbb{R}^{M \times N}$ into 4D tensors (or cores) with a specific configuration. Specifically,

$$W = L_1 L_2 ... L_m C R_1 R_2 ... R_m \tag{4}$$

with $L_i,\ i \leq m$ being the left auxiliary cores, C being the center core, and $R_i,\ i \leq m$ being the right auxiliary cores. The left and right auxiliary cores are analogous; we will refer to them simply as auxiliary cores from now on. Therefore, specifying a decomposition with m left (or right) auxiliary cores results in a total of $2m + 1$ tensors. Each tensor has shape $\mathbb{R}^{d_{in}, r, c, d_{out}}$, with r, c being the *physical row resp. column legs* and d_{in}, d_{out} being the *bond dimensions*. The specification of r and c for each core represents the main hyperparameter choice of the MPO system, as we do not restrict the bond dimensions here (while it is more typical in quantum physics applications; (Cui et al., 2015)). Finally, a straightforward tensor contraction of all cores recovers the original matrix.

3. OperA: multiplicative mixture of experts

The aim of this work is to show that multiplicative interactions notably improve the parameter-efficiency of MoE-based PEFT by increasing the size of the subspace spanned by the experts, thereby also reducing the memory required by the adapters. Our OperA technique focuses on building the experts (or adapters) to obtain greater rank efficiency at lower parameter count.

As mentioned above, OperA leverages multiplicative interactions, which have been shown to be key to deep neural networks (Jayakumar et al., 2020). Here, these interactions are described by MPO, and help OperA achieve an even greater effective rank with lower parameter count compared to additive adapters. In contrast to LoRA, OperA does not use randomly-initialized fixed-rank adapters, but builds arbitrarily ranked adapters directly from the original weight matrix W. The first step of OperA is to obtain the MPO representation of W using tensor-train singular value decomposition (TT-SVD; Oseledets 2011). TT-SVD needs to be performed only once prior to training for each model, and does not represent a notable performance bottleneck.

Table 2. **Comparison of OperA and competing solutions given the same parameter budget.** For each model, we report the number of parameters and the task accuracy.

Model	Adapter	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	HS ↑	WG ↑	PMQA ↑	Avg ↑
Llama3-8B	LoRAMoE	111M	77.55	86.49	87.76	86.20	74.42	94.73	82.11	75.10	83.05
	MixLoRA	106M	78.75	87.12	83.81	85.20	74.62	94.75	83.90	75.30	82.93
	OperA	95M	77.64	86.41	87.60	84.20	74.74	95.71	83.03	76.10	**83.18**
Llama3.1-8B	LoRAMoE	111M	77.33	86.32	88.52	85.60	74.68	95.61	82.35	75.50	83.24
	MixLoRA	106M	78.33	86.45	83.60	87.40	75.69	95.16	83.42	76.10	83.27
	OperA	95M	77.73	87.29	88.19	85.40	75.69	95.91	80.03	76.70	**83.37**
Llama3.2-3B	LoRAMoE	60M	64.76	82.53	83.68	79.40	71.12	93.42	76.32	73.00	78.03
	MixLoRA	59M	69.33	83.42	83.17	79.40	70.98	92.15	76.64	72.40	78.44
	OperA	59M	69.08	84.64	82.64	79.40	71.53	93.60	75.16	73.00	**78.63**
Phi4-14B	LoRAMoE	154M	91.81	96.70	92.71	93.20	76.02	95.26	88.08	75.00	88.60
	MixLoRA	148M	91.47	96.68	92.27	93.60	76.02	95.81	87.85	76.2	**88.74**
	OperA	96M	87.29	92.80	90.91	91.00	75.17	95.38	87.21	75.10	86.86
Qwen2.5-3B	LoRAMoE	87M	80.20	89.60	84.71	87.20	71.35	92.62	78.85	69.90	81.80
	MixLoRA	83M	82.08	88.39	85.70	84.93	70.80	92.76	79.79	69.90	81.79
	OperA	74M	80.89	89.40	86.02	87.60	70.06	92.99	77.82	71.80	**82.07**
Qwen2.5-7B	LoRAMoE	116M	87.46	92.21	90.91	91.80	73.98	95.17	83.90	72.30	85.97
	MixLoRA	111M	86.77	91.92	88.93	92.40	75.17	95.36	85.79	72.50	86.11
	OperA	93M	87.63	92.51	90.59	94.00	74.07	95.76	83.98	72.20	**86.34**
Qwen2.5-14B	LoRAMoE	173M	91.33	93.50	93.09	97.00	77.80	97.09	90.21	74.60	89.33
	MixLoRA	166M	91.47	93.64	92.82	96.00	77.74	97.09	91.63	74.80	89.40
	OperA	153M	91.04	93.81	93.36	96.80	77.92	97.30	90.90	74.60	**89.47**
Qwen2.5-32B	LoRAMoE	439M	93.69	94.65	94.40	96.40	78.44	97.53	91.87	73.70	90.09
	MixLoRA	422M	93.52	94.49	93.85	96.20	79.11	97.29	91.27	73.20	89.87
	OperA	343M	93.94	94.49	94.29	97.40	79.51	97.38	90.53	74.20	**90.22**

Without loss of generality, consider three cores

$$W = LCR \tag{5}$$

Given n desired experts, we define the multiplicative adapters $L_i, R_i,\ i \leq n$ by initializing them to equal L, R. Next, each expert E_i becomes

$$E_i = L_i C R_i \tag{6}$$

with C fixed. Finally, given the same router T as before the input ($\boldsymbol{x}$)-dependent OperA layer becomes

$$W' = \frac{1}{2}W + \sum_i^n T(\boldsymbol{x})_i \frac{1}{2} E_i \tag{7}$$

with $1/2$ necessary to achieve equality at the beginning of training, analogous to setting the B matrix to 0 in LoRA.

The key to the parameter-efficiency of OperA is to concentrate most of the parameters into C, which is not trainable. We remark that the main hyperparameters of TT-SVD consist in the sizes of the *physical legs* r, c of the MPO system. In particular, we set row legs r_1, r_2, r_3 (with $r_1 \times r_2 \times r_3 = M$) and column legs c_1, c_2, c_3 (with $c_1 \times c_2 \times c_3 = N$). With these parameters set, we can now fully describe the shapes of the MPO:

$$L \in \mathbb{R}^{1 \times r_1 \times c_1 \times d_1}, C \in \mathbb{R}^{d_1 \times r_2 \times c_2 \times d_2}, R \in \mathbb{R}^{d_2 \times r_3 \times c_3 \times 1} \tag{8}$$

The bond dimensions d_1, d_2 are an effect of TT-SVD and cannot be directly lowered without resulting in a lossy reconstruction of W as in regular SVD. Therefore, we can simply choose $r_2 \gg r_{i \neq 2}$ and $c_2 \gg c_{i \neq 2}$ to obtain the desired parameter-reduction effect.

The multiplicative adapters L_i, R_i do not directly limit the rank of the expert, allowing the system to exhibit greater variety and flexibility. In particular, the physical operator is completely determined by the contraction of auxiliary tensors (see also MPOE; Gao et al. 2022). Concretely, the subspace of representable weight matrices is limited by the bond dimensions resulting from TT-SVD (Pirvu et al., 2010), which are in practice orders of magnitude larger than the LoRA rank (see Figure 2).

Nonetheless, the greater parameter efficiency comes at a computational cost, as each expert is now full-rank and must be obtained from a tensor contraction at every step. Therefore, OperA can be seen as a compute-bound but memory-efficient alternative to LoRA. This is highly beneficial for GPU implementations where compute is abundant while memory bandwidth is often the bottleneck.

3.1. Efficient implementation of OperA

The multiplicative nature of OperA necessarily implies higher computational costs with respect to additive solutions like LoRA, in a classic memory-computation tradeoff scenario (we evaluate this effect concretely in Appendix C.4). At the same time, it is well-established that memory represents the main bottleneck for accelerator-based training, both in terms of price and communication overheads within and among GPUs. In large-scale deployments, LoRA's ability to address this bottleneck has been a key to its success.

Table 3. **Accuracy of PEFT MoEfication on multiple downstream tasks simultaneously.** For each model, we report the number of parameters and the task accuracy.

Model	Adapter	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	HS ↑	WG ↑	PMQA ↑	Avg ↑
Llama3-8B	LoRAMoE	246M	79.44	86.91	88.25	86.80	75.78	95.18	83.27	73.40	83.63
	MixLoRA	241M	76.45	86.99	88.14	82.20	75.23	94.24	83.43	54.60	80.16
	OperA	173M	78.13	87.42	88.68	86.60	74.56	96.00	84.16	73.90	**83.68**
Llama3.1-8B	LoRAMoE	246M	79.01	86.99	88.74	84.80	75.99	94.53	84.45	74.30	83.60
	MixLoRA	241M	76.71	86.45	88.19	85.80	73.24	94.76	80.43	54.50	80.01
	OperA	173M	78.88	87.96	88.36	86.00	74.79	95.95	84.00	73.80	**83.72**
Llama3.2-3B	LoRAMoE	133M	67.41	84.51	84.98	77.80	71.44	93.67	75.45	73.30	78.57
	MixLoRA	131M	68.09	83.92	83.62	78.80	70.49	92.40	77.03	71.20	78.19
	OperA	92M	69.62	84.72	83.46	79.20	70.76	93.71	76.32	72.30	**78.76**
Phi4-14B	LoRAMoE	342M	91.47	94.11	92.66	92.60	76.97	96.15	87.69	75.70	88.42
	MixLoRA	336M	91.89	94.91	91.89	93.40	75.50	95.93	88.40	75.80	88.47
	OperA	182M	92.53	96.27	91.99	93.00	77.10	97.03	88.77	75.50	**89.02**
Qwen2.5-3B	LoRAMoE	192M	82.51	89.86	86.11	87.40	70.46	92.34	79.16	69.30	82.14
	MixLoRA	188M	80.89	88.76	86.60	88.00	71.44	93.29	78.37	70.20	82.19
	OperA	74M	80.38	89.35	86.72	88.60	70.60	93.42	78.98	70.10	**82.27**
Qwen2.5-7B	LoRAMoE	258M	87.37	91.96	89.77	92.60	74.95	95.24	84.21	72.70	86.10
	MixLoRA	253M	87.88	93.94	90.64	91.40	75.44	94.63	85.08	52.30	83.91
	OperA	192M	87.71	92.13	91.13	92.00	74.50	95.85	83.74	72.80	**86.23**
Qwen2.5-14B	LoRAMoE	384M	91.30	93.73	92.93	94.80	78.01	97.12	89.74	74.10	88.97
	MixLoRA	376M	90.96	93.73	92.11	95.20	78.38	96.75	89.58	74.00	88.84
	OperA	200M	91.90	93.73	93.10	95.50	78.11	96.90	90.55	74.70	**89.31**

Specifically, a widely-used setup involves loading multiple adapters on the same set of accelerators, which then serve multiple finetuned models through a single matrix-vector multiplication with the base model and multiple cheaper additions. This setup can be translated to parameter-efficient MoEfication as well to notably improve training and generation throughput, as shown by MixLoRA (Li et al., 2024).

In particular, reusing the notation of Section 2, we see that Algorithm 1 is equivalent to LoRA MoEfication.

Algorithm 1 Efficient computation of LoRA experts for PEFT MoEfication.

Input: data $\boldsymbol{x}$, weight W, adapters $A_i, B_i \ \forall i \in [1, ..., K]$, number of experts K
Initialize $\boldsymbol{o} = W\boldsymbol{x}$
for $i = 1$ **to** K **do**
  $E_i(\boldsymbol{x}) = \boldsymbol{o} + A_i B_i \boldsymbol{x}$
end for

While this technique is not directly applicable to OperA, an analogue exists by leveraging reordering of operations: it is not necessary to perform the tensor contraction left-to-right. This principle is also one of the key components of the density matrix renormalization group (DMRG; Schollwöck 2011) algorithm, again a staple of quantum physics used to optimize the MPO decomposition of an Hamiltonian. We borrow from this vast literature and propose Algorithm 2 for the efficient deployment of OperA in large-scale settings.

As with LoRA, we now need to perform the most expensive computation using the central core only once, while each expert can be generated iteratively by multiplication with the much smaller auxiliary cores. This algorithm requires a GPU-native implementation using block-level matrix multiplications to reduce data movement within the accelerator and keep memory bounded. For the context of this work, we find that it is sufficient to rely on the optimal reordering algorithm provided by PyTorch.

Algorithm 2 Efficient computation of OperA experts for PEFT MoEfication.

Input: data $\boldsymbol{x}$, central core C, auxiliary cores $L_i, R_i \ \forall i \in [1, ..., K]$, number of experts K
Initialize $O = C\boldsymbol{x}$
for $i = 1$ **to** K **do**
  $E_i(\boldsymbol{x}) = OL_iR_i$
end for

4. Results

4.1. Experimental setup

Adapters. We compare MoLA, LoRAMoE, MixLoRA, HydraLoRA, and OperA on a variety of models to ensure the consistency of our performance results. As MPOE (Gao et al., 2022) requires the same peak memory as full finetuning, we do not include it here. To push the rank-efficiency of LoRA methods further, we compare our baselines with another additive adapter, SharedLoRA, that we develop based on a two-tier structure (see Appendix A). We report the results using LoRAMoE and MixLoRA in the main text, as they achieved best average accuracy. The full results are reported in Appendix C.

Models. From Meta, we test: Llama3-8B, Llama3.1-8B, and Llama3.2-3B. From Qwen, we test: Qwen2.5-3B, Qwen2.5-7B, Qwen2.5-14B, and Qwen2.5-32B. Finally, from Microsoft we test Phi4-14B. This choice of the 8 mod-

els covers a representative range of sizes (from 3B to 32B parameters) and of model families.

Tasks. For each dataset, we first finetune each adapter on the training split, and then evaluate the accuracy on the testing split. The task ensemble consists of question-and-answer tasks (ARC-e, ARC-c, PIQA, OpenBookQA), classification tasks (BoolQ), and completion tasks (HellaSwag, WinoGrande, PubmedQA). Moreover, we also test the adapters' ability in a multi-task context, where all the previously mentioned tasks are tested simultaneously. All tasks are evaluated on accuracy.

MoEs. For all experiments, we use 8 total experts with 2 active experts per token. Following best practices, we use noisy top-k routing with an auxiliary router balancing loss.

LoRA. For all additive (LoRA-based) approaches, the rank of the adapters is the main hyperparameter. In particular, the rank is set to either 16 (original, see Section 4.2), 12 (optimal, see Appendix C.2), or 8 (same memory budget as OperA, see Section 4.3).

OperA. We reuse the basic architecture of LoRA MoEs of MixLoRA (Li et al., 2024) (see Appendix C.7 for an ablation of MixLoRA's fusion). In particular, we replace the MLP layers OperA MoEs and use standard LoRAs for the attention layers, fixing the rank to 6 to give enough flexibility for adaptation without contributing a large amount of parameters. In contrast to LoRAs, MPOs, as used in this work, do not have rank. We instead focus on the sizes of the physical legs to achieve the desired parameter efficiency. Specifically, we confirm that 5 legs is the optimal choice (Gao et al., 2022). We also find that a 40% to 60% concentration of the legs onto the central core provides a well-balanced solution, leading to empirical Pareto optimality. Indeed, if too little parameters are given to the auxiliary cores then no adaptation can take place, while too large auxiliary cores do not achieve the required parameter efficiency. The specific sizes can be found in Appendix B.

Environment. All experiments are performed on Nvidia A100 80GB or Nvidia H200 140GB using PyTorch 2.6 and the MoE-PEFT (Li et al., 2024) codebase, using 10 to 500 GPU hours per task depending on the model size. All models are trained using mixed fp32 and bf16.

4.2. Baseline

We replicate the setup of MixLoRA to obtain comparable results with established methods, and follow the original hyperparameter selection which found rank 16 to be optimal. In Table 1 we show that OperA achieves competitive accuracy across all models and tasks, also surpassing all baselines on 6/8 models, while obtaining on average a 36% space saving across all tested models with respect to MixLoRA, the most efficient additive adapter. The full results are reported in Appendix C.1. In particular, OperA achieves 61% parameter reduction against MixLoRA on Qwen2.5-3B and a 70% reduction against MoLA on the same model. Overall, OperA surpasses the baseline on all the models tested, with notable parameter efficiency gains (see Appendix B for the configuration of each adapter).

4.3. Optimized additive adapters with a fixed parameter budget

We have demonstrated that OperA is competitive with competing solutions while requiring notably less parameters. At the same time, in Appendix C.2 we also further optimize the additive LoRAs, and MixLoRA in particular, both for accuracy and for parameter count, and show that OperA outperforms the baselines in this setting as well.

However, the question remains whether OperA can actually leverage its multiplicative interactions to be more parameter-efficient than the additive baselines. Table 2 indicates that OperA is Pareto-optimal on 7 out of the 8 models, achieving higher accuracy at a lower parameter count than the baseline.

4.4. Multi task

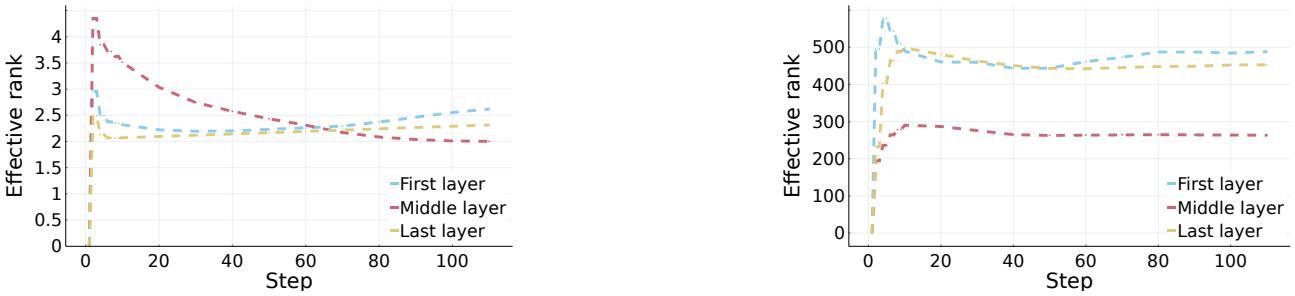
We have shown that OperA maintains a high accuracy while notably reducing the parameter count across all tasks. At the same time, we wish to stress the representation capabilities of all PEFT methods by also considering a multi-task scenario, where the required dimensionality of the modified subspace increases substantially. Qwen2.5-32B cannot be trained on a single GPU in this scenario, so it is excluded. In Table 3 we report the results of OperA on all the previous tasks at the same time (full results in Appendix C.3), consistent with the experimental setup of MoE-PEFT (Li et al., 2024). Here, OperA surpasses all baselines across all models, validating the superior generalization capabilities of multiplicative adapters against additive ones.

4.5. Effective rank

While the LoRA rank puts an upper bound to the dimensionality of the PEFT, whether the adapters do actually make use of all available space can be directly investigated. Here, we choose the effective rank (Roy & Vetterli, 2007) as the measure of interest due to its inherent robustness. The effective rank is the measure of the entropy of the singular values of a matrix, computed as:

$$\mathrm{erank}(W) = \frac{\|W\|_F^2}{\|W\|_2^2} \tag{9}$$

where $\|\cdot\|_F$ is the Frobenius norm and $\|\cdot\|_2$ is the L2 norm. The erank quantifies the concentration of the singular values



(a) Effective rank of MixLoRA.

(b) Effective rank of OperA.

Figure 2. **Effective rank of the adapters.** Development of the effective rank as finetuning progresses.

and therefore the effective dimensionality of the matrix.

We compute the effective rank of MixLoRA adapters as training progresses as the effective rank of the full LoRA matrix, i.e. $\text{erank}(E_i)$. In the case of OperA, we compute $\text{erank}(W - E_i)$ instead, as each E_i is always full rank and we wish to evaluate the additional dimensionality brought by E_i to the system, rather than its raw dimensionality.

Figure 2 shows that the effective rank quickly rises at the beginning of training, as expected, but then progressively settles. In the case of MixLoRA, the effective rank degrades quickly, hovering around 2.5 for both shallow and deep layers, consistent with the findings in (Li et al., 2024) that performance is maintained down to a LoRA rank of 2. On the other hand OperA's effective rank is notably higher and more stable than the additive approaches, indicating that the additional information gained by OperA spans a much greater subspace even with fewer parameters.

4.6. Ablations

Optimal rank. The main hyperparameter that determines the parameter count of LoRA-based approaches is the rank of the adapters. As mentioned earlier, to obtain baseline results we first apply all adapters with rank 16, reported to be optimal for MixLoRA (Li et al., 2024). However, we also perform a fine-grained search using modern architectures to ensure as strong a baseline as possible. Figure 3 shows that performance peaks at rank 12 for Llama3-8B, while it remains stable after rank 12 for Qwen2.5-3B. Therefore, we choose rank 12 as the most parameter-efficient solution, and extend this evaluation to all architectures and other adapters to establish an optimized baseline in Appendix C.2.

Optimal MPO size. The parameter count of OperA is mainly determined by the configuration of the physical legs, which are specific to each model and are evaluated empirically for effectiveness: this is analogous to the choice of rank for LoRA. Nonetheless, the amount of legs also contributes to the downstream performance by varying the degree of factorization. To obtain the optimal number, we sweep from 3 to 7 legs on Qwen2.5-3B and Llama3-8B and check the average performance across 5 tasks. Figure 4 shows that 5 legs provide the best accuracy.

5. Related works

Parameter-efficient finetuning (PEFT). Modern open-weights frontier LLMs can reach tens of billions to hundreds of billions of parameters. Finetuning such models for downstream tasks can prove unfeasible in most situations, where parameter-efficient finetuning (PEFT) becomes a necessity. The most popular approach to PEFT for LLMs is LoRA (Hu et al., 2022), that leverages low-rank matrices to finetune a low-dimensional subspace of the weights and improve performance while reducing the required number of parameters. Further gains on parameter efficiency have been obtained by weight sharing (VeRA; Kopiczko et al. 2024), weight tying (Tied-LoRA; Renduchintala et al. 2024), quantization (QLoRA; Dettmers et al. 2023), and other techniques (Yang et al., 2025; Hayou et al., 2025). The downstream performance of LoRA has also been improved, especially through normalization (DoRA; Liu et al. 2024b).

Mixture of experts (MoEs). MoEs (Shazeer et al., 2017; Fedus et al., 2022) allow LLMs to efficiently scale the number of parameters without proportionally scaling their compute budgets. In particular, MoEs are part of the conditional computing paradigm, where subsets of the full model are activated in an input-dependent manner. There are multiple variants of MoEs, specifying various routing (Fedus et al., 2022; Ruiz et al., 2021; Yue et al., 2025) and gating mechanisms (Li et al., 2023; Puigcerver et al., 2024). Recent works (DeepSeek-AI, 2025; Qwen AI, 2025a) have also shown that very large mixtures with few experts active show remarkable performance across a wide variety of tasks.

Parameter-efficient MoEfication. The combination of PEFT with MoEs has recently started gaining traction (Dou et al., 2024; Tian et al., 2024; Li et al., 2024; Zeng et al., 2025; Gao et al., 2025), to address the limitations of

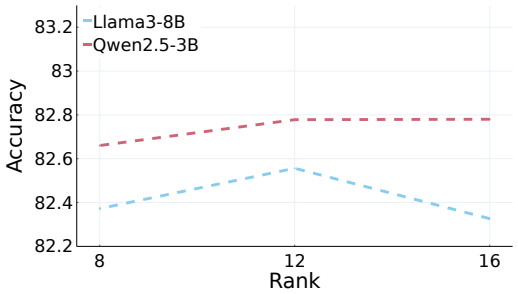


Figure 3. **MixLoRA rank ablation.** We report the average accuracy at varying LoRA ranks, using MixLoRA.

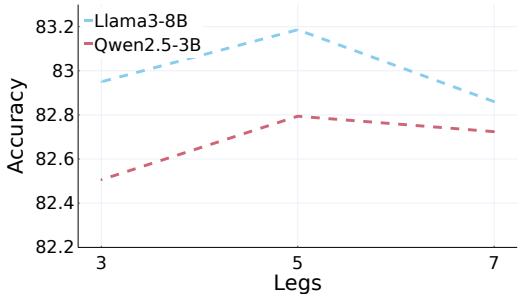


Figure 4. **OperA legs ablation.** We report the average accuracy at various leg configurations, using OperA.

both approaches. Most parameter-efficient MoEfication techniques (Luo et al., 2024; Gao et al., 2024), like LoRAMoE (Dou et al., 2024), replace all linear layers with MoEs. Even greater efficiency is achieved in HydraLoRA by splitting the LoRA adapters into heads (Tian et al., 2024), driving further factorization. Greater specialization is achieved by MoLA (Gao et al., 2025), that targets Transformer layers with different numbers of experts. MixLoRA (Li et al., 2024) selectively replaces the QKV matrices in the attention layers with simple LoRAs, while fusing the MLP layers with parameter-efficient MoEs. More recently, S'MoRE (Zeng et al., 2025) proposes hierarchical mixtures with structured graphs. We adopt the architecture of MixLoRA, as it has been shown to be highly effective, and aim to improve the rank efficiency of each adapter.

Multiplicative adapters. Multiplicative adapters for PEFT have seen some adoption, for example with Kronecker-Product (Yu et al., 2025). A combination of additive and multiplicative adapters using Hadamard product has been shown (Singhal et al., 2026) to notably increase the resulting rank, but still within the framework of additive LoRAs. We explore here Matrix product operators (MPOs), a particularly effective multiplicative decomposition from quantum physics (Pirvu et al., 2010; Schollwöck, 2011; Orús, 2014). MPOs have been used for compressing linear layers (Gao et al., 2020) and as a direct alternative to LoRAs (Liu et al., 2021). More similar to this line of work, MPOE (Gao et al., 2022) also propose an MPO-based approach to parameter-efficient MoEfication. MPOE decomposes the weights using MPO, and then finetunes both the auxiliary and the core tensors. Each expert is recovered by contracting the shared core with the specific auxiliary tensors. The key differences between MPOE and our OperA lie in both the construction of the experts and the routing mechanism. In particular, as MPOE also trains the core tensor, it requires as much computational expense as full finetuning and is thus not applicable to the setting of this work. Additionally, OperA keeps the residual path as the shared expert, and fuses the adapters with the weights.

6. Limitations

OperA strikes a different balance than LoRA in the compute-memory trade-off, lowering memory costs at the expense of compute. While we have proposed a DMRG-style algorithm to mitigate this drawback, we have also characterized the throughput (see Appendix C.4) of OperA to be slower than additive adapters in its current state. Moreover, we have found that OperA surpasses all alternatives on 7/8 models tested, with Phi4-14B being the only exception. As scale is not the issue (OperA is SoTA on the larger Qwen2.5-32B), we speculate that this is due to a combination of pre-training data and architectural design. Nonetheless, OperA is SoTA on Phi4-14B in the multi-task setting, suggesting that more challenging scenarios are particularly beneficial.

7. Conclusion

In this work, we show that multiplicative interactions, which are fundamental to learning in deep neural networks (Jayakumar et al., 2020), notably benefit parameter-efficient MoEfication by improving parameter efficiency and increasing the effective rank of the adapters. OperA leverages matrix product operators (MPOs), a multiplicative decomposition stemming from quantum physics, to produce low-parameter adapters to obtain powerful high-rank experts. Moreover, we highlight the usefulness of the effective rank as a quantification of the dimensionality of an adapter, and show that targeting it leads to improved performance without an increase in parameter count.

Overall, we show that OperA reaches competitive results against 4 other established methods on most of the models and tasks tested. Specifically, OperA reaches a higher accuracy than all baselines at a lower parameter cost on 7/8 models tested, making it the optimal choice for a given parameter budget. Moreover, it also surpasses the baselines on all models in the multi-task setting. Therefore, OperA provides an effective compute bound, rather than memory bound, parameter-efficient MoEfication technique that is appealing for large-scale deployment where memory is scarce.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Abdin, M., Aneja, J., Behl, H., Bubeck, S., Eldan, R., Gunasekar, S., Harrison, M., Hewett, R. J., Javaheripi, M., Kauffmann, P., Lee, J. R., Lee, Y. T., Li, Y., Liu, W., Mendes, C. C. T., Nguyen, A., Price, E., de Rosa, G., Saarikivi, O., Salim, A., Shah, S., Wang, X., Ward, R., Wu, Y., Yu, D., Zhang, C., and Zhang, Y. Phi-4 technical report. *arXiv preprint arXiv:2412.08905*, 2024.

Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. PIQA: Reasoning about physical commonsense in natural language. In *AAAI*, 2020.

Clark, C., Lee, K., Chang, M.-W., Kwiatkowski, T., Collins, M., and Toutanova, K. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*, 2019.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457v1*, 2018.

Cui, J., Cirac, J. I., and Bañuls, M. C. Variational matrix product operators for the steady state of dissipative quantum systems. *Phys. Rev. Lett.*, 114, 2015.

DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2025.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. QLoRA: Efficient finetuning of quantized LLMs. In *NeurIPS*, 2023.

Dou, S., Zhou, E., Liu, Y., Gao, S., Shen, W., Xiong, L., Zhou, Y., Wang, X., Xi, Z., Fan, X., Pu, S., Zhu, J., Zheng, R., Gui, T., Zhang, Q., and Huang, X. LoRAMoE: Alleviating world knowledge forgetting in large language models via MoE-style plugin. In *Proceedings of the Association for Computational Linguistics (ACL)*, 2024.

Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research (JMLR)*, 23(120):1–39, 2022.

Feng, W., Hao, C., Zhang, Y., Han, Y., and Wang, H. Mixture-of-loras: An efficient multitask tuning method for large language models. In *International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING)*, 2024.

Gao, C., Chen, K., Rao, J., Sun, B., Liu, R., Peng, D., Zhang, Y., Guo, X., Yang, J., and Subrahmanian, V. Higher layers need more LoRA experts. *arXiv preprint arXiv:2402.08562*, 2024.

Gao, C., Chen, K., Rao, J., Liu, R., Sun, B., Zhang, Y., Peng, D., Guo, X., and Subrahmanian, V. MoLA: MoE LoRA with layer-wise expert allocation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, 2025.

Gao, Z.-F., Cheng, S., He, R.-Q., Xie, Z. Y., Zhao, H.-H., Lu, Z.-Y., and Xiang, T. Compressing deep neural networks by matrix product operators. *Physical Review Research*, 2(2), 2020.

Gao, Z.-F., Liu, P., Zhao, W. X., Lu, Z.-Y., and Wen, J.-R. Parameter-efficient mixture-of-experts architecture for pre-trained language models. In *Iternational Conference on Computational Linguistics (ICCL)*, 2022.

Han, Z., Gao, C., Liu, J., Zhang, J., and Zhang, S. Q. Parameter-efficient fine-tuning for large models: A comprehensive survey. *Transactions on Machine Learning Research (TMLR)*, 2024.

Hayou, S., Ghosh, N., and Yu, B. PLoP: Precise LoRA placement for efficient finetuning of large models. *arXiv preprint arXiv:2506.20629*, 2025.

Hu, E. J., yelong shen, Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

IBM. Granite 3.0 language models. https://github.com/ibm-granite/granite-3.0-language-models/, 2024.

Jayakumar, S. M., Czarnecki, W. M., Menick, J., Schwarz, J., Rae, J., Osindero, S., Teh, Y. W., Harley, T., and Pascanu, R. Multiplicative interactions and where to find them. In *ICLR*, 2020.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de las Casas, D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

Jin, Q., Dhingra, B., Liu, Z., Cohen, W., and Lu, X. PubMedQA: A dataset for biomedical research question answering. In *International Joint Conference on Natural Language Processing (JCNLP)*, 2019.

Kopiczko, D. J., Blankevoort, T., and Asano, Y. M. VeRA: Vector-based random matrix adaptation. In *ICLR*, 2024.

Li, B., Shen, Y., Yang, J., Wang, Y., Ren, J., Che, T., Zhang, J., and Liu, Z. Sparse mixture-of-experts are domain generalizable learners. In *ICLR*, 2023.

Li, D., Ma, Y., Wang, N., Ye, Z., Cheng, Z., Tang, Y., Zhang, Y., Duan, L., Zuo, J., Yang, C., and Tang, M. MixLoRA: Enhancing large language models fine-tuning with LoRA-based mixture of experts. *arXiv preprint arXiv:2404.15159*, 2024.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. Visual instruction tuning. In *NeurIPS*, 2023.

Liu, H., Li, C., Li, Y., Li, B., Zhang, Y., Shen, S., and Lee, Y. J. LLaVA-NeXT: Improved reasoning, OCR, and world knowledge, January 2024a. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

Liu, P., Gao, Z.-F., Zhao, W. X., Xie, Z.-Y., Lu, Z.-Y., and Wen, J.-R. Enabling lightweight fine-tuning for pre-trained language model compression based on matrix product operators. In *International Joint Conference on Natural Language Processing (IJCNLP)*, 2021.

Liu, S.-Y., Wang, C.-Y., Yin, H., Molchanov, P., Wang, Y.-C. F., Cheng, K.-T., and Chen, M.-H. DoRA: Weight-decomposed low-rank adaptation. In *ICML*, 2024b.

Luo, T., Lei, J., Lei, F., Liu, W., He, S., Zhao, J., and Liu, K. MoELoRA: Contrastive learning guided mixture of experts on parameter-efficient fine-tuning for large language models. *arXiv preprint arXiv:2402.12851*, 2024.

Meta AI. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2024.

OpenAI. GPT-5 system card. https://openai.com/index/gpt-5-system-card/, 2025.

Orús, R. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of physics*, 349:117–158, 2014.

Oseledets, I. V. Tensor-Train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

Parthasarathy, V. B., Zafar, A., Khan, A., and Shahid, A. The ultimate guide to fine-tuning LLMs from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities. *arXiv preprint arXiv:2408.13296*, 2024.

Pirvu, B., Murg, V., Cirac, J. I., and Verstraete, F. Matrix product operator representations. *New Journal of Physics*, 12(2), 2010.

Puigcerver, J., Ruiz, C. R., Mustafa, B., and Houlsby, N. From sparse to soft mixtures of experts. In *ICLR*, 2024.

Qwen AI. Qwen3-Next. https://qwen.ai/blog?id=4074cca80393150c248e508aa62983f9cb7d27cd&from=research.latest-advancements-list, 2025a.

Qwen AI. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025b.

Renduchintala, A., Konuk, T., and Kuchaiev, O. Tied-LoRA: Enhancing parameter efficiency of LoRA with weight tying. In *NAACL*, 2024.

Roy, O. and Vetterli, M. The effective rank: A measure of effective dimensionality. In *European signal processing conference*, 2007.

Ruiz, C. R., Puigcerver, J., Mustafa, B., Neumann, M., Jenatton, R., Pinto, A. S., Keysers, D., and Houlsby, N. Scaling vision with sparse mixture of experts. In *NeurIPS*, 2021.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: an adversarial winograd schema challenge at scale. *Communications ACM*, 64(9):99–106, 2021.

Schollwöck, U. The density-matrix renormalization group in the age of matrix product states. *Annals of physics*, 326(1):96–192, 2011.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*, 2017.

Singhal, R., Ponkshe, K., Vartak, R., and Vepakomma, P. ABBA-adapters: Efficient and expressive fine-tuning of foundation models. In *International Conference on Learning Representations (ICLR)*, 2026.

Tian, C., Shi, Z., Guo, Z., Li, L., and zhong Xu, C. HydraloRA: An asymmetric loRA architecture for efficient fine-tuning. In *NeurIPS*, 2024.

Wang, J., Jiang, H., Liu, Y., Ma, C., Zhang, X., Pan, Y., Liu, M., Gu, P., Xia, S., Li, W., Zhang, Y., Wu, Z., Liu, Z., Zhong, T., Ge, B., Zhang, T., Qiang, N., Hu, X., Jiang, X., Zhang, X., Zhang, W., Shen, D., Liu, T., and Zhang, S. A comprehensive review of multimodal large language models: Performance and challenges across different tasks. *arXiv preprint arXiv:2408.01319*, 2024.

Xia, Y., Fu, F., Zhang, W., Jiang, J., and CUI, B. Efficient multi-task LLM quantization and serving for multiple loRA adapters. In *NeurIPS*, 2024a.

Xia, Y., Kim, J., Chen, Y., Ye, H., Kundu, S., Hao, C., and Talati, N. Understanding the performance and estimating the cost of LLM fine-tuning. *arXiv preprint arXiv:2408.04693*, 2024b.

Yang, J., Ma, Y., Di, D., Li, H., Chen, W., Xie, Y., Cui, J., Yang, X., and Zuo, W. QR-LoRA: Efficient and disentangled fine-tuning via QR decomposition for customized generation. *arXiv preprint arXiv:2507.04599*, 2025.

Yu, B., Yang, Z., and Yi, X. MoKA: Parameter efficiency fine-tuning via mixture of Kronecker product adaption. In *International Conference on Computational Linguistics (ICCL)*, 2025.

Yue, T., Guo, L., Cheng, J., Gao, X., Huang, H., and Liu, J. Ada-K routing: Boosting the efficiency of MoE-based LLMs. In *ICLR*, 2025.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the Association for Computational Linguistics*, 2019.

Zeng, H., Xia, Y., Zhao, Z., Jiang, G., Zhang, Q., Liu, J., Zhang, L., Fan, X., and Zhang, B. S'more: Structural mixture of residual experts for llm fine-tuning. *arXiv preprint arXiv:2504.06426*, 2025.

A. Improving the effective rank of additive adapters

While this work aims to show that multiplicative adapters are remarkably effective in terms of effective rank and performance, an interesting question that can be raised is whether it is possible to improve the effective rank of purely additive adapters. To answer this question, we propose a hierarchical PEFT technique, SharedLoRA(see Figure 5, which exhibits a higher effective rank than other solutions.

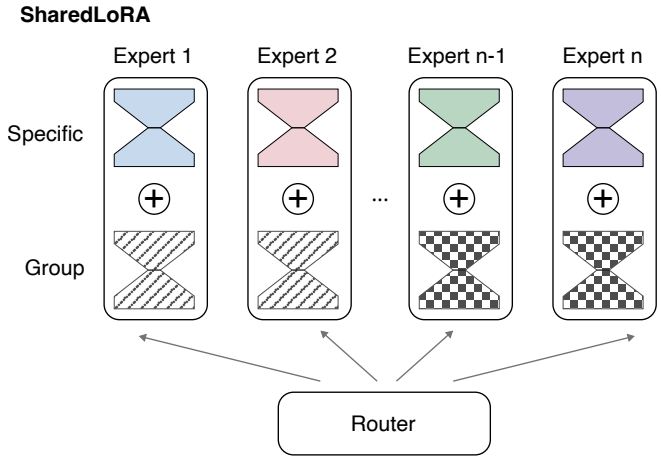
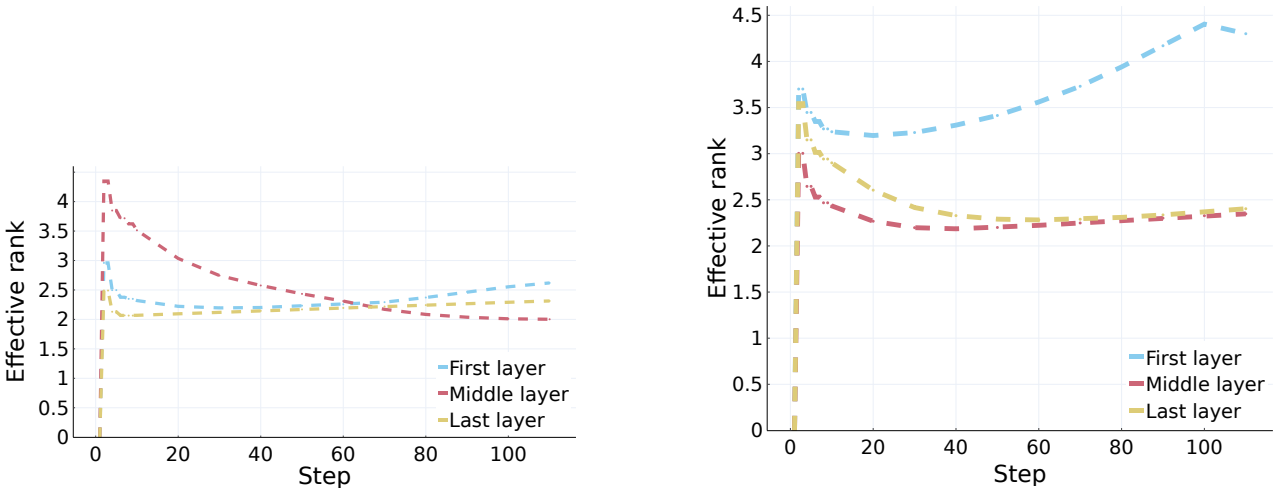


Figure 5. **SharedLoRA.** The experts of SharedLoRA are built by additively combining shared basis LoRAs with expert-specific LoRAs of the same rank. The SharedLoRA rank is therefore the sum of the two ranks.



(a) Effective rank of MixLoRA.

(b) Effective rank of SharedLoRA.

Figure 6. **The effective rank of additive adapters can be improved.** Development of the effective rank of SharedLoRA as finetuning progresses.

SharedLoRA uses a combination of two tiers of low-rank adapters to obtain experts with a higher effective rank without impacting the parameter count. Given a weight matrix W and n desired experts, we first initialize n first-tier (or specific) adapters $A_i^s, B_i^s,\ i \leq n$ of rank r. Next, we initialize g second-tier (or group) adapters $A_i^b, B_i^b,\ i \leq g$, with g a divisor of n such that $n = g \times m$. We then assign the first group adapters A_1^b, B_1^b to the first m experts, the second group adapters to the second m experts, and so on. Next, each expert E_i is defined as

$$E_i = A_i^s B_i^s + A_{i/m}^b B_{i/m}^b \tag{10}$$

Together with its noisy top-k router, the input ($\boldsymbol{x}$)-dependent SharedLoRA layer becomes

$$W' = W + \sum_{i}^{n} T(\boldsymbol{x})_i (A_i^s B_i^s + A_{i/m}^b B_{i/m}^b) \tag{11}$$

as is usual in MoEs.

Each E_i has a maximum rank of $2r$, for a total number of parameters of $(n+g) \times (M+N) \times r$. In contrast, flat (non-tiered) LoRA MoEs require $2 \times n \times (M+N) \times r$ parameters to achieve the same maximum rank. The two-tier structure results in a $\frac{2n}{n+g} > 1$ (for $g < n$) reduction of parameters at the same rank.

We evaluate SharedLoRA on the same tasks as OperA, and the results can be found in Tables 7 and 12. These results indicate that SharedLoRA is an effective additive adapter that reduces the parameter count while maintaining accuracy. Moreover, Figure 6 shows that SharedLoRA does indeed improve the rank efficiency with respect to competing additive methods.

B. Adapter details

B.1. OperA

The hyperparameters for OperA depend on the chosen model, specifically the size of the physical legs. We report in Table 4 both the row legs (Row) and the column legs (Column) in the format $r_1 : r_2 : ... : r_l$ and $c_1 : c_2 : ... : c_l$, with l the number of legs. The number of legs here is always 5, as found in Section 4.6.

Table 4. **Hyperparameters of OperA.** Legs and rank configuration of OperA change depending on the experimental setup.

Model	Column	Row	Rank	Alpha	Dropout	Results
Llama3-8B	2:8:32:4:2	4:7:32:4:4	6	12	0.2	Section 4.2
	2:4:32:8:2	4:7:32:4:4	6	12	0.2	Section 4.5, 4.6, Appendix C.2
Llama3.1-8B	2:8:32:4:2	4:7:32:4:4	6	12	0.2	Section 4.2
	2:4:32:8:2	4:7:32:4:4	6	12	0.2	Section 4.5, 4.6, Appendix C.2
Llama3.2-3B	2:8:12:8:2	2:8:32:8:2	6	12	0.2	Section 4.2
	2:8:24:4:2	2:8:32:8:2	6	12	0.2	Section 4.5, 4.6, Appendix C.2
Phi4-14B	2:8:40:4:2	4:8:35:4:4	6	12	0.2	Section 4.2
	2:5:32:8:2	4:7:40:4:4	6	12	0.2	Section 4.5, 4.6, Appendix C.2
Qwen2.5-3B	2:8:16:4:2	2:8:43:8:2	6	12	0.2	Section 4.2
	2:8:16:4:2	2:8:43:8:2	6	12	0.2	Section 4.5, 4.6, Appendix C.2
Qwen2.5-7B	2:8:28:4:2	4:8:37:8:2	6	12	0.2	Section 4.2
	2:8:28:4:2	2:8:37:8:4	6	12	0.2	Section 4.5, 4.6, Appendix C.2
Qwen2.5-14B	2:8:40:4:2	4:6:36:4:4	6	12	0.2	Section 4.2
	2:5:32:8:2	4:6:48:4:3	6	12	0.2	Section 4.5, 4.6, Appendix C.2
Qwen2.5-32B	2:8:40:4:2	4:6:36:8:4	6	12	0.2	Section 4.2
	2:5:32:8:2	4:6:48:8:3	6	12	0.2	Section 4.5, 4.6, Appendix C.2

B.2. Additive adapters

We report the hyperparameters chosen for the baseline additive adapters in Table 5. The hyperparameters are chosen based on the findings in MixLoRA (Li et al., 2024), and then further finetuned as described in this work.

Table 5. **Hyperparameters of the additive adapters.** The configuration of the additive adapters change depending on the experimental setup.

Model	Rank	Alpha	Dropout	Results
All	16	32	0.05	Section 4.2
All	12	24	0.05	Appendix C.2
All	8	16	0.05	Section 4.6
All	7	14	0.05	Section 4.5

B.3. SharedLoRA

We report the hyperparameters chosen for SharedLoRA in Table 5. The hyperparameters are chosen based on an search conducted on the rank and dropout ratio.

Table 6. **Hyperparameters of SharedLoRA.** The configuration of SharedLoRA changes depending on the experimental setup.

Model	Rank	Alpha	Dropout	Results
All	12	24	0.2	Appendix C
All	8	16	0.2	Appendix C.2

C. Additional results

C.1. Full comparison with additive adapters

We report here the complete set of results in the single-task scenario including all additive adapters. We also evaluate the accuracy of our two-tier additive schema, SharedLoRA, using the same setup as Section 4.2. In fact, we have seen in Appendix A that SharedLoRA increases the effective rank of the additive adapters while reducing the parameter count. Table 7 shows that our hypothesis holds in practice, with SharedLoRA having a lower parameter count than other adapters while maintaining high performance.

C.2. Rank optimization

We perform an hyperparameter search on Llama3-8B and Qwen2.5-3B to obtain the optimal rank of MixLoRA on more recent architectures. Our rank sweep is more fine grained that in the original work, and our results indicate that substantial space savings can be achieved with rank 12 (see Section 4.5 for the ablation results). We extend this evaluation to the other additive baselines, and in parallel we reduce the parameters of OperA and SharedLoRA to produce comparable results. The results in Table 8 indicate that OperA is stable and amenable to different configurations, providing evidence for its flexibility. This is especially remarkable given the multiplicative nature of our method, which shows similar robustness to already established algorithms such as LoRA while employing a notably different technique. Specifically, the performance of OperA in Table 8 is, similarly to the main results with a higher rank, equal or superior to the additive baselines, with even more impressive space savings of 58% against MixLoRA, the most efficient additive adapter once again.

C.3. Multi-task

We evaluate OperA also in a multi-task scenario to further test the rank-efficiency of our methods. This task includes all previous tasks at the same time, again following the setup of MoE-PEFT (Li et al., 2024). As shown in Table 9, OperA once more surpasses all additive alternatives in average accuracy with a notable reduction in parameters.

Rank optimized setup. As before, we use the rank-optimized version of the additive adapters to confirm the performance of our models. In Table 10, OperA surpasses all additive alternatives here as well.

C.4. Throughput

In the PEFT literature it is often the case that a tradeoff is made between computational complexity and performance, together with parameter efficiency. In Table 11, we evaluate the relative slowdown of our multiplicative approach with respect to the additive baselines. The results show that OperA is $1.8 - 4.1\times$ slower than the additive approaches, depending on the model configuration. As mentioned in Section 3.1, additive adapters make use of an optimized implementation based on the pre-computation of intermediate steps that can be reused across experts to improve throughput. At the same time, OperA is not further optimized in this work and only leverages the automatic tensor reordering performed by PyTorch. Again in Section 3.1 we offer an optimized solution inspired by DMRG that could provide notable speedups.

C.5. Rank ablation

Table 12 reports all the results of the rank ablation performed on Qwen2.5-3B and Llama3-7B. Too high and too low a rank harm performance, with 12 being the optimal rank.

C.6. MPO size ablation

Table 13 reports all the results of the ablation on the number of legs of the MPO performed on Qwen2.5-3B. A moderate number of legs, 5, provides the best performance.

C.7. Fusion

The adapter fusion approach of MixLoRA (Li et al., 2024) has been shown to notably improve the performance of additive adapters with LoRA. In Table 14 we report that such approach is helpful for multiplicative adapters such as OperA as well.

Table 7. **Accuracy of all tested adapters.** For each model, we report the number of parameters and the task accuracy.

Model	Adapter	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	HS ↑	WG ↑	PMQA ↑	Avg ↑
Llama3-8B	None	/	66.04	81.19	62.24	67.80	59.27	48.43	67.80	54.40	63.40
	HydraLoRA	241M	76.62	81.65	83.79	81.20	70.80	92.78	79.79	54.70	77.67
	LoRAMoE	246M	77.56	86.62	87.60	87.80	74.33	95.21	84.29	73.10	83.31
	MoLA	346M	76.62	88.13	85.75	85.20	75.23	92.30	82.00	73.00	82.28
	MixLoRA	241M	75.76	87.08	86.18	89.00	73.61	94.63	83.11	55.00	80.55
	SharedLoRA	224M	75.51	86.28	86.02	86.60	74.13	94.03	82.79	54.70	80.01
	OperA	173M	77.39	86.28	88.30	87.60	76.36	96.15	81.45	74.10	**83.45**
Llama3.1-8B	None	/	68.34	81.19	67.08	69.20	57.37	55.35	52.72	58.30	63.69
	HydraLoRA	241M	77.65	84.09	82.37	81.00	70.92	91.09	81.93	55.60	78.08
	LoRAMoE	246M	79.52	85.50	87.32	86.20	74.46	92.50	82.32	73.20	82.63
	MoLA	346M	77.65	86.78	86.45	84.80	74.37	93.76	83.58	72.50	82.49
	MixLoRA	241M	76.96	86.82	86.40	83.20	74.65	94.77	84.14	54.40	80.17
	SharedLoRA	224M	76.11	87.21	86.18	83.00	74.43	94.52	83.74	74.40	82.45
	OperA	173M	76.88	86.32	87.92	85.20	75.23	96.10	80.58	74.70	**82.87**
Llama3.2-3B	None	/	53.41	66.79	63.82	55.00	57.61	45.98	51.93	55.30	56.23
	HydraLoRA	131M	67.52	83.71	81.56	78.20	71.87	90.07	74.61	70.70	77.28
	LoRAMoE	133M	67.92	83.29	82.97	77.00	71.68	90.62	74.22	69.90	77.20
	MoLA	200M	68.52	81.94	82.86	78.00	71.96	90.33	74.30	70.10	77.25
	MixLoRA	131M	68.26	83.12	83.02	77.40	71.50	92.13	75.14	68.10	77.33
	SharedLoRA	121M	69.28	83.29	84.00	79.00	72.20	91.70	76.72	69.50	**78.21**
	OperA	92M	66.98	81.73	83.19	77.20	70.98	92.93	74.19	71.10	77.29
Phi4-14B	None	/	79.95	84.76	90.04	79.40	68.72	87.19	77.74	72.90	80.09
	HydraLoRA	336M	91.38	95.20	90.86	90.80	75.50	95.53	89.50	73.00	87.72
	LoRAMoE	342M	92.02	94.11	92.44	94.00	74.57	95.61	87.53	71.40	87.71
	MoLA	389M	91.38	97.10	91.46	93.80	76.97	96.30	88.40	73.40	**88.60**
	MixLoRA	336M	91.21	94.28	91.90	92.60	75.47	95.95	88.79	73.40	87.95
	SharedLoRA	314M	90.36	95.20	92.44	92.00	76.18	96.12	88.32	75.50	88.27
	OperA	182M	88.14	92.84	91.84	91.00	74.49	95.25	86.27	70.60	86.30
Qwen2.5-3B	None	/	77.65	87.58	75.46	75.00	63.64	72.08	57.38	58.80	70.95
	HydraLoRA	188M	79.99	88.17	84.33	85.40	69.97	90.29	77.74	69.20	80.64
	LoRAMoE	192M	80.11	88.34	85.58	85.30	70.70	91.57	78.93	70.50	81.38
	MoLA	246M	80.03	88.26	85.64	86.00	70.92	92.44	77.24	69.90	81.30
	MixLoRA	188M	80.12	88.64	86.62	87.60	70.92	91.67	80.03	70.40	**82.00**
	SharedLoRA	175M	79.95	88.05	86.23	87.40	71.47	91.96	81.21	68.80	81.88
	OperA	74M	80.89	89.40	86.02	87.60	70.06	92.99	74.82	70.60	81.55
Qwen2.5-7B	None	/	85.07	91.33	81.99	84.60	63.09	86.00	58.64	63.10	76.73
	HydraLoRA	253M	86.18	91.13	86.51	90.00	71.83	95.20	84.37	53.10	82.29
	LoRAMoE	258M	87.03	91.79	89.83	92.00	74.22	95.30	84.93	71.60	85.84
	MoLA	332M	86.18	92.34	88.96	91.20	72.81	94.95	83.03	71.30	85.10
	MixLoRA	253M	87.88	94.19	88.74	92.80	74.52	95.29	83.27	52.70	83.67
	SharedLoRA	235M	87.03	92.04	88.36	92.20	74.43	94.85	85.16	52.90	83.37
	OperA	192M	87.71	93.39	90.42	91.40	74.74	95.91	83.66	72.50	**86.22**
Qwen2.5-14B	None	/	89.42	93.10	88.79	86.80	70.61	91.12	69.22	67.60	82.08
	HydraLoRA	376M	89.94	92.98	91.24	93.80	76.70	96.25	90.84	54.30	85.76
	LoRAMoE	384M	90.66	93.42	91.46	96.20	76.69	97.01	91.32	74.10	88.86
	MoLA	568M	91.47	93.44	91.12	95.20	75.75	95.50	91.08	74.20	88.47
	MixLoRA	376M	90.78	93.73	92.27	95.60	77.58	96.82	90.21	53.10	86.26
	SharedLoRA	348M	90.96	93.86	91.84	95.60	76.94	94.81	90.84	73.80	88.58
	OperA	200M	90.61	93.73	93.42	96.00	78.10	97.43	90.13	75.20	**89.33**
Qwen2.5-32B	None	/	92.24	94.23	89.77	89.80	70.03	94.21	77.35	62.10	83.72
	HydraLoRA	841M	93.00	95.20	93.63	96.20	76.61	96.06	91.08	74.70	89.64
	LoRAMoE	858M	93.43	94.32	94.23	96.40	78.26	97.24	92.23	53.40	87.37
	MoLA	1104M	93.60	94.74	93.74	97.20	77.43	97.42	91.24	51.00	87.05
	MixLoRA	841M	93.26	95.75	94.07	97.00	78.35	97.29	90.92	73.20	89.98
	SharedLoRA	783M	93.86	95.62	93.04	96.20	78.26	97.15	91.48	43.90	86.19
	OperA	343M	93.94	94.49	94.29	97.40	79.51	97.38	90.53	74.20	**90.22**

Table 8. **Accuracy of rank-optimized MixLoRA, SharedLoRA, and OperA**. For each model, we report the number of parameters and the task accuracy.

Model	Adapter	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	HS ↑	WG ↑	PMQA ↑	Avg ↑
Llama3-8B	HydraLoRA	181M	76.62	84.13	83.95	83.40	72.75	93.21	81.45	75.60	81.39
	LoRAMoE	186M	78.67	85.86	86.94	86.20	74.98	95.78	82.95	74.90	**83.29**
	MoLA	262M	77.05	86.07	86.18	87.00	74.74	94.94	84.29	74.70	83.12
	MixLoRA	181M	77.13	87.84	85.90	87.20	74.71	94.70	83.35	74.50	83.17
	SharedLoRA	149M	78.92	86.24	86.02	86.60	74.13	94.92	83.19	74.00	83.00
	OperA	95M	77.64	86.41	87.60	84.20	74.74	95.71	83.03	76.10	83.18
Llama3.1-8B	HydraLoRA	181M	75.34	85.73	85.15	82.60	74.37	92.87	83.19	75.60	81.86
	LoRAMoE	186M	78.07	87.88	88.79	84.60	74.40	95.64	83.43	74.50	83.41
	MoLA	262M	78.41	87.21	86.29	86.60	74.98	94.50	84.93	74.40	83.42
	MixLoRA	181M	76.11	85.27	87.76	84.20	75.50	94.93	84.61	52.20	80.07
	SharedLoRA	149M	76.62	87.46	86.62	86.60	74.65	94.92	84.29	74.40	83.20
	OperA	95M	77.73	87.29	88.19	85.40	75.69	95.91	74.70	83.03	**83.49**
Llama3.2-3B	HydraLoRA	98M	65.19	84.34	83.08	78.80	70.83	91.21	77.35	70.90	77.71
	LoRAMoE	101M	67.49	84.68	83.90	80.00	72.32	92.42	75.69	71.80	78.54
	MoLA	152M	68.77	82.87	82.92	78.20	70.37	92.35	78.53	73.20	78.40
	MixLoRA	98M	69.28	84.85	83.57	79.80	72.20	92.53	73.09	72.10	78.43
	SharedLoRA	81M	68.00	82.66	83.79	79.00	72.20	92.87	76.24	71.00	78.22
	OperA	59M	69.08	84.64	82.64	79.40	71.53	93.60	75.16	73.00	**78.63**
Phi4-14B	HydraLoRA	252M	92.83	93.86	91.24	91.20	74.80	95.28	89.03	72.30	87.57
	LoRAMoE	258M	91.47	92.90	93.04	92.40	75.54	96.36	88.63	75.50	88.23
	MoLA	294M	92.58	97.31	91.57	93.60	75.93	95.96	88.56	76.80	**89.04**
	MixLoRA	252M	90.36	96.84	91.89	94.00	75.41	95.81	87.32	76.20	88.48
	SharedLoRA	209M	90.87	96.70	91.78	92.80	75.72	96.01	88.08	76.50	88.56
	OperA	96M	87.29	92.80	90.91	91.00	75.17	95.38	87.21	75.10	86.86
Qwen2.5-3B	HydraLoRA	141M	80.20	88.89	85.20	86.80	70.24	91.78	78.22	68.70	81.25
	LoRAMoE	145M	80.97	89.35	85.36	86.60	70.98	92.69	79.79	70.70	82.06
	MoLA	186M	79.61	88.85	84.49	88.60	71.56	92.02	78.22	68.20	81.44
	MixLoRA	141M	81.06	88.59	86.56	86.40	71.28	91.94	79.08	70.60	81.94
	SharedLoRA	117M	81.57	89.31	85.96	87.60	71.56	92.16	79.32	70.30	**82.22**
	OperA	74M	80.89	89.40	86.02	87.60	70.06	92.99	77.82	71.80	82.07
Qwen2.5-7B	HydraLoRA	190M	86.01	90.95	88.25	91.80	75.00	93.67	84.06	52.90	82.83
	LoRAMoE	195M	87.20	92.00	90.15	91.20	74.80	95.32	84.69	72.60	86.00
	MoLA	251M	87.46	93.98	89.61	92.60	74.43	95.22	83.50	51.50	83.54
	MixLoRA	190M	87.54	92.17	88.90	92.80	74.10	95.60	84.14	53.00	83.53
	SharedLoRA	157M	87.03	91.54	89.99	92.20	72.90	95.10	84.14	73.10	85.75
	OperA	93M	87.63	92.51	90.59	94.00	74.07	95.76	83.98	72.20	**86.34**
Qwen2.5-14B	HydraLoRA	283M	89.59	93.56	91.68	94.60	75.75	96.34	90.69	53.10	85.66
	LoRAMoE	290M	91.21	93.60	93.20	96.60	77.25	97.04	89.50	73.00	88.93
	MoLA	430M	91.21	93.76	93.20	96.60	77.25	97.04	89.50	65.60	88.02
	MixLoRA	282M	91.21	93.77	92.93	95.80	77.86	96.92	90.06	72.90	88.93
	SharedLoRA	232M	91.56	93.60	92.76	96.20	77.30	97.14	89.82	72.90	88.91
	OperA	153M	91.04	93.81	93.36	96.80	77.92	97.30	89.90	74.60	**89.34**
Qwen2.5-32B	HydraLoRA	632M	93.26	95.75	93.36	97.20	77.74	97.11	91.00	75.20	90.08
	LoRAMoE	649M	93.43	94.82	93.85	96.60	78.35	97.20	91.08	75.70	90.13
	MoLA	835M	93.69	94.57	93.42	96.80	79.20	97.09	91.95	75.60	90.29
	MixLoRA	632M	93.26	94.40	93.09	96.40	78.69	97.02	91.55	75.10	89.94
	SharedLoRA	523M	93.77	94.61	93.96	96.00	78.90	97.15	92.03	74.20	90.08
	OperA	330M	93.77	95.83	93.91	96.80	78.32	97.39	91.79	75.70	**90.44**

Table 9. **Accuracy of all tested adapters on multiple tasks simultaneously.** For each model, we report the number of parameters and the task accuracy.

Model	Adapter	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	HS ↑	WG ↑	PMQA ↑	Avg ↑
Llama3-8B	HydraLoRA	241M	64.08	62.71	52.56	25.80	47.09	24.22	50.28	54.30	47.63
	LoRAMoE	246M	79.44	86.91	88.25	86.80	75.78	95.18	83.27	73.40	83.63
	MoLA	346M	75.77	86.91	88.63	85.20	74.86	94.14	83.11	74.30	82.87
	MixLoRA	241M	76.45	86.99	88.14	82.20	75.23	94.24	83.43	54.60	80.16
	SharedLoRA	224M	77.65	86.91	87.65	86.20	75.72	94.21	85.48	67.00	82.60
	OperA	173M	78.13	87.42	88.68	86.60	74.56	96.00	84.16	73.90	**83.68**
Llama3.1-8B	HydraLoRA	241M	72.01	85.65	82.10	79.40	70.12	86.69	67.32	74.10	77.17
	LoRAMoE	246M	79.01	86.99	88.74	84.80	75.99	94.53	84.45	74.30	83.60
	MoLA	346M	78.58	86.95	88.03	84.80	74.46	94.48	84.61	73.80	83.21
	MixLoRA	241M	76.71	86.45	88.19	85.80	73.24	94.76	80.43	54.50	80.01
	SharedLoRA	224M	78.07	86.62	87.92	82.60	73.39	95.01	80.98	72.90	82.19
	OperA	173M	78.88	87.96	88.36	86.00	74.79	95.95	84.00	73.80	**83.72**
Llama3.2-3B	HydraLoRA	131M	61.09	77.82	77.81	73.20	66.21	89.73	64.56	72.30	72.84
	LoRAMoE	133M	67.41	84.51	84.98	77.80	71.44	93.67	75.45	73.30	78.57
	MoLA	200M	67.49	84.39	84.17	80.20	71.13	92.89	76.48	72.50	78.66
	MixLoRA	131M	68.09	83.92	83.62	78.80	70.49	92.40	77.03	71.20	78.19
	SharedLoRA	121M	69.20	85.14	83.95	77.20	71.19	92.32	77.19	71.60	78.47
	OperA	92M	69.62	84.72	83.46	79.20	70.76	93.71	76.32	72.30	**78.76**
Phi4-14B	HydraLoRA	336M	90.53	93.22	49.35	95.20	77.25	96.07	87.53	53.00	80.27
	LoRAMoE	342M	91.47	94.11	92.66	92.60	76.97	96.15	87.69	75.70	88.42
	MoLA	389M	92.41	95.41	91.95	93.40	77.16	96.09	89.58	75.80	88.98
	MixLoRA	336M	91.89	94.91	91.89	93.40	75.50	95.93	88.40	75.80	88.47
	SharedLoRA	314M	90.78	97.56	91.89	92.00	76.15	96.03	89.19	74.50	88.51
	OperA	182M	92.53	96.27	91.99	93.00	77.10	97.03	88.77	75.50	**89.02**
Qwen2.5-3B	HydraLoRA	188M	81.31	89.65	49.51	86.40	69.17	25.30	74.59	70.30	68.28
	LoRAMoE	192M	82.51	89.86	86.11	87.40	70.46	92.34	79.16	69.30	82.14
	MoLA	246M	80.20	88.80	86.34	85.00	71.28	92.88	78.69	69.90	81.64
	MixLoRA	188M	80.89	88.76	86.60	88.00	71.44	93.29	78.37	70.20	82.19
	SharedLoRA	175M	80.55	88.89	87.21	87.40	71.04	92.70	77.58	71.20	82.07
	OperA	74M	80.38	89.35	86.72	88.60	70.60	93.42	78.98	70.10	**82.27**
Qwen2.5-7B	HydraLoRA	253M	87.97	91.79	87.76	90.60	69.45	91.86	75.69	50.90	80.75
	LoRAMoE	258M	87.37	91.96	89.77	92.60	74.95	95.24	84.21	72.70	86.10
	MoLA	332M	86.01	91.96	91.35	91.60	75.29	94.69	85.87	49.40	83.27
	MixLoRA	253M	87.88	93.94	90.64	91.40	75.44	94.63	85.08	52.30	83.91
	SharedLoRA	235M	87.54	91.88	90.32	91.40	74.80	94.65	84.69	71.80	85.89
	OperA	192M	87.71	92.13	91.13	92.00	74.50	95.85	83.74	72.80	**86.23**
Qwen2.5-14B	HydraLoRA	376M	92.15	94.07	93.31	95.20	77.92	97.39	91.00	73.20	89.28
	LoRAMoE	384M	91.30	93.73	92.93	94.80	78.01	97.12	89.74	74.10	88.97
	MoLA	568M	91.64	93.52	93.14	94.20	77.31	97.08	90.84	74.40	89.02
	MixLoRA	376M	90.96	93.73	92.11	95.20	78.38	96.75	89.58	74.00	88.84
	SharedLoRA	348M	90.61	93.35	92.60	96.60	77.92	97.04	90.29	75.00	89.18
	OperA	200M	91.90	93.73	93.10	95.50	78.11	96.90	90.55	74.70	**89.31**

Table 10. **Accuracy of rank-optimized MixLoRA, SharedLoRA, and OperA on multiple tasks at once.** For each model, we report the number of parameters and the task accuracy.

Model	Adapter	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	HS ↑	WG ↑	PMQA ↑	Avg ↑
Llama3-8B	HydraLoRA	181M	73.46	85.02	78.67	79.60	62.45	90.61	71.90	72.20	76.74
	LoRAMoE	186M	77.41	87.37	88.90	86.80	75.81	95.45	83.98	73.10	83.60
	MoLA	262M	77.22	86.99	86.94	81.60	73.09	95.02	84.69	75.00	82.57
	MixLoRA	181M	77.73	86.36	88.68	85.80	74.80	94.18	83.35	73.10	83.00
	SharedLoRA	149M	76.96	87.88	87.92	86.60	73.82	94.97	82.64	73.80	83.07
	OperA	95M	77.13	87.33	89.28	86.40	75.35	95.54	84.29	74.30	**83.70**
Llama3.1-8B	HydraLoRA	181M	74.15	85.90	81.77	80.80	71.35	90.80	73.88	75.30	79.24
	LoRAMoE	186M	78.07	88.47	88.96	85.80	73.79	95.50	83.98	73.70	83.53
	MoLA	262M	77.22	86.99	86.94	81.60	73.09	95.02	84.69	75.00	82.57
	MixLoRA	181M	78.07	86.49	87.16	83.00	73.06	95.16	82.08	54.70	79.97
	SharedLoRA	149M	77.47	86.53	88.57	84.80	73.36	94.97	82.48	74.50	82.84
	OperA	95M	76.54	86.20	88.74	86.80	75.14	96.23	83.50	75.80	**83.62**
Llama3.2-3B	HydraLoRA	98M	61.69	80.22	79.33	76.40	68.29	90.72	67.48	72.10	74.53
	LoRAMoE	101M	68.60	83.50	85.36	78.60	71.19	93.53	75.14	73.80	78.72
	MoLA	152M	68.09	84.43	84.33	79.60	70.70	92.63	77.11	69.30	78.27
	MixLoRA	98M	69.80	84.72	85.26	79.40	70.31	92.12	77.58	71.60	78.85
	SharedLoRA	81M	68.77	83.29	85.15	77.60	70.00	92.84	78.69	72.80	78.64
	OperA	59M	67.96	84.51	84.68	79.00	70.76	93.46	77.69	73.20	**78.91**
Phi4-14B	HydraLoRA	252M	91.55	93.43	91.29	92.40	74.19	93.99	83.74	75.00	86.95
	LoRAMoE	258M	91.64	94.15	91.82	92.80	76.61	96.00	86.58	76.60	88.28
	MoLA	294M	91.98	96.76	91.73	94.40	75.50	96.16	87.92	75.50	88.74
	MixLoRA	252M	91.13	95.29	92.71	94.20	76.85	95.94	88.24	75.10	88.68
	SharedLoRA	209M	90.87	93.94	91.51	92.60	76.88	96.02	88.40	76.10	88.29
	OperA	96M	91.44	95.55	91.73	94.80	76.60	96.79	88.30	75.50	**88.84**
Qwen2.5-3B	HydraLoRA	141M	82.08	89.69	83.57	88.20	68.72	92.20	74.27	68.80	80.94
	LoRAMoE	145M	81.66	90.03	86.29	87.00	70.21	93.38	79.16	69.90	82.20
	MoLA	186M	80.46	89.65	86.29	86.00	70.43	92.61	79.24	71.90	82.07
	MixLoRA	141M	81.06	89.27	86.89	87.80	70.34	92.05	78.93	71.10	82.18
	SharedLoRA	117M	81.23	89.98	86.83	87.20	70.52	92.50	78.30	70.90	82.18
	OperA	74M	80.38	89.35	86.72	87.60	69.60	93.42	79.98	71.10	**82.27**
Qwen2.5-7B	HydraLoRA	190M	88.65	92.26	89.23	91.20	70.95	25.20	76.01	71.40	75.61
	LoRAMoE	195M	86.52	92.30	90.81	92.60	74.59	95.63	85.08	71.00	86.07
	MoLA	251M	86.86	92.34	90.70	93.20	75.44	94.87	83.90	53.30	83.83
	MixLoRA	190M	87.80	91.54	89.45	91.00	74.77	94.92	85.64	53.20	83.54
	SharedLoRA	157M	87.12	92.17	90.42	92.40	75.14	95.07	84.45	70.00	85.85
	OperA	93M	87.46	91.92	90.75	93.20	75.41	95.91	83.90	72.00	**86.32**
Qwen2.5-14B	HydraLoRA	283M	91.89	93.52	92.33	94.40	77.19	86.90	87.37	52.80	84.55
	LoRAMoE	290M	91.47	93.43	92.76	95.20	77.22	97.44	89.98	73.00	88.81
	MoLA	430M	91.64	93.27	93.09	94.60	77.37	97.02	90.69	53.30	86.37
	MixLoRA	282M	91.47	93.43	92.49	95.40	77.77	97.00	90.77	74.90	89.15
	SharedLoRA	232M	90.44	93.90	92.98	96.00	78.04	97.13	89.58	74.90	89.12
	OperA	153M	91.64	93.33	92.78	95.90	77.37	97.22	90.97	74.50	**89.21**

Table 11. **Throughput of PEFT MoEfication.** We report the throughput of all tested models and adapters in seconds per batch.

Model	HydraLoRA	LoraMoE	MoLA	MixLoRA	SharedLoRA	OperA
Llama3-3B	0.9108	1.3354	1.4880	0.8456	1.0349	3.5019
Llama3.1-3B	0.8870	1.1037	1.5214	0.8423	1.1197	2.6230
Llama3.2-3B	0.6281	0.7581	0.8475	0.5028	0.7328	1.3363
Phi4-14B	1.3103	1.3313	1.0475	1.0760	1.3441	5.5511
Qwen2.5-3B	0.7249	0.9529	1.3986	0.6451	0.8959	1.5588
Qwen2.5-7B	0.8898	0.9745	1.3755	0.8194	0.9443	3.2040
Qwen2.5-14B	1.5238	1.3492	1.4427	1.2482	1.8647	4.9189
Qwen2.5-32B	1.5613	1.7872	2.0970	1.5837	1.9863	5.1422

Table 12. **Results of the rank ablation**. We sweep the rank of MixLoRA from 16 to 8 on Qwen2.5-3B and Llama3-8B, to determine the optimal configuration.

Adapter	Model	Rank ↓	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	Avg ↑
MixLoRA	Qwen2.5-3B	16	188M	80.12	88.64	86.62	87.60	70.92	**82.78**
		12	142M	81.06	88.59	86.56	86.40	71.28	**82.78**
		8	95M	81.91	89.01	83.19	87.60	71.59	82.66
MixLoRA	Llama3-8B	16	241M	75.76	87.08	86.18	89.00	73.61	82.33
		12	181M	77.13	87.84	85.90	87.20	74.71	**82.56**
		8	121M	78.84	87.33	83.59	86.80	75.30	82.37

Table 13. **Results of the MPO size ablation**. We sweep the number of legs in OperA from 3 to 7 on Llama3-8B and Qwen2.5-3B, to determine the optimal configuration.

Adapter	Model	Column	Row	No. params ↓	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	Avg ↑
OperA	Llama3-8B	2:2:4:32:2:2:2	2:2:7:32:2:2:4	172M	77.22	86.24	87.65	86.80	76.39	82.84
		2:8:32:4:2	4:7:32:4:4	172M	77.39	86.28	88.30	87.60	76.36	**83.19**
		16:32:8	28:32:16	172M	78.07	86.36	87.38	87.40	75.54	82.95
OperA	Qwen2.5-3B	2:2:4:16:2:2:2	2:2:4:43:4:2:2	74M	80.55	89.52	85.53	88.20	69.82	82.72
		2:8:16:4:2	2:8:43:8:2	74M	80.89	89.40	86.02	87.60	70.06	**82.79**
		16:16:8	16:43:16	74M	80.03	89.35	85.53	88.60	69.02	82.51

Table 14. **Results of the fusion ablation**. We evaluate whether the adapter fusion approach is successful for OperA as well on Llama3-8B and Qwen2.5-3B.

Model	Adapter	ARC-c ↑	ARC-e ↑	PIQA ↑	OBQA ↑	BoolQ ↑	Avg ↑
Llama3-8B	OperA (w/o fusion)	73.46	84.30	87.11	86.6	74.07	81.11
	OperA	77.64	86.41	87.6	84.2	74.74	**84.19**
Qwen2.5-3B	OperA (w/o fusion)	76.54	87.84	82.92	85.8	69.17	80.45
	OperA	80.89	89.40	86.02	87.60	70.06	**83.54**