On the Duality between Gradient Transformations and Adapters

Lucas Torroba-Hennigen Hunter Lang Han Guo Yoon Kim

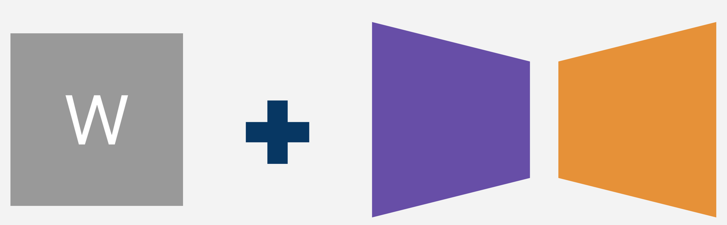
Background

Training LLMs requires a lot of memory. Two memory-efficient training methods stand out.

Adapter methods

Key idea: Freeze base weights and train only an additive perturbation. Fewer parameters means less optimizer and gradient memory.

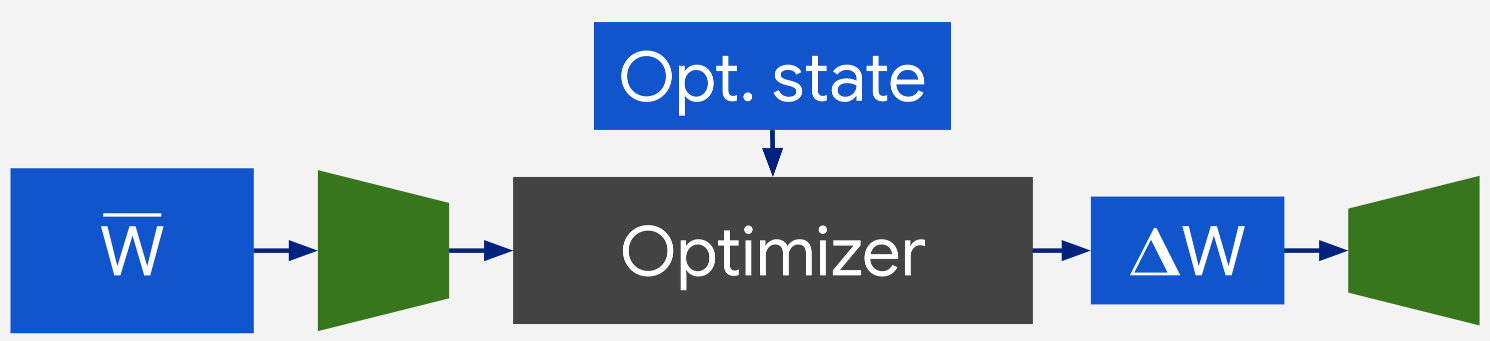
Example: Low-rank (LoRA; Hu et al., 2021)



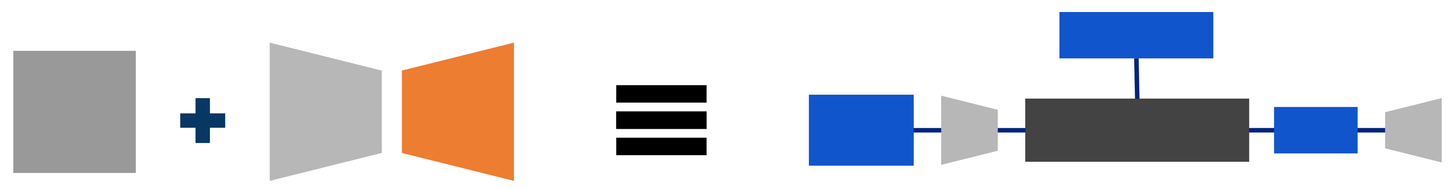
Gradient transformation methods

Key idea: Perform optimizer update in lower-dimensional space. Reduces optimizer states, and hence memory.

Example: GaLore (Zhao et al., 2024)



Previous work: The two methods above are equivalent under certain conditions



Key contributions:

1. Generalize the equivalence between adapters and gradient transformations
2. Exploit this equivalence to improve memory-efficient pretraining
3. Explore this equivalence in the context of memory-constrained distributed pretraining

1 Equivalence theorem

Main result: Training with linear gradient transformations is equivalent to training with a linear adapter.

We explore a general setting:

- Model is a d-dim vector, e.g., $\Theta = \text{vec}(\mathbf{W})$
- Gradient transformation is a linear map to a smaller space, i.e., $\mathbf{S} \in \mathbb{R}^{r\times d}$
- Optimizer takes gradient $\overline{\Theta}^{(t)}$ and states $\xi_\Theta^{(t)}$ at step t as inputs (e.g., SGD, Adam)

Theorem 1 (Duality theorem) shows that training a model with gradient transformations, i.e.,

$$(\Delta_{\mathbf{S}\Theta}^{(t)}, \xi_{\mathbf{S}\Theta}^{(t+1)}) = \text{Optimizer}(\mathbf{S}\overline{\Theta}^{(t)}, \xi_{\mathbf{S}\Theta}^{(t)})$$
$$\Theta^{(t+1)} = \Theta^{(t)} + \mathbf{S}^\top \Delta_{\mathbf{S}\Theta}^{(t)},$$

is equivalent to training it with a linear adapter, i.e., substitute original parameter with $\Theta^{(0)} + \mathbf{S}^\top \Lambda$ where $\Lambda \in \mathbb{R}^r$ is our new parameter. From this:

$$\mathbf{W}^{(0)} + \mathbf{L}^\top \mathbf{A}\mathbf{R}^\top \equiv \mathbf{L}\overline{\mathbf{W}}\mathbf{R}$$

(Adapter view; **A** is parameter) (Gradient view)

New result: If **S** is Kronecker-factored, i.e., $\mathbf{S} = \mathbf{R}^\top \otimes \mathbf{L}$, then this establishes an equivalence between MoRA (Jiang et al., 2024) and a two-sided version of GaLore.

Remark: In GaLore, the gradient transformation is periodically swapped out; this is equivalent to ReLoRA (Lialin et al., 2023).

Experiments: We conduct two studies.

In 2 we investigate two key knobs: choice of transformation and base weight quantization.

In 3 we explore whether worker-specific transformations help in distributed training.

Setup: 200M/1B pretraining; Llama architecture

2 Memory-efficient pretraining

Goal: Retain perplexity but reduce memory use

Finding 1: Rematerializable transformations often reduce memory without big impact to perplexity

Model	Adapter form	PPL	Mem.
Full pretraining	$\mathbf{W}$	12.44	8.04
ReLoRA	$\mathbf{W}+\mathbf{BA}$	13.94	5.77
SVD (GaLore)	$\mathbf{W}+\mathbf{P}^\top\mathbf{A},\ \mathbf{P}^\top = \text{SVD}(\overline{\mathbf{W}})$	13.62	5.27
Gauss. (Flora)	$\mathbf{W}+\mathbf{P}^\top\mathbf{A},\ \mathbf{P} \sim k\mathcal{N}(\mathbf{0},\mathbf{I})$	13.88	5.02
Rademacher	$\mathbf{W}+\mathbf{P}^\top\mathbf{A},\ \mathbf{P} \sim k\,\text{Unif}(\{-1,1\})$	13.86	5.02
Semi-orthogonal	$\mathbf{W}+\mathbf{P}^\top\mathbf{A},\ \mathbf{P}^\top\mathbf{P} = k\mathbf{I}$	13.71	5.27
Two-side Gauss.	$\mathbf{W}+\mathbf{L}^\top\mathbf{A}\mathbf{R}^\top,\ \mathbf{L},\mathbf{R} \sim k\mathcal{N}(\mathbf{0},\mathbf{I})$	15.28	5.02
Two-side SVD	$\mathbf{W}+\mathbf{L}^\top\mathbf{A}\mathbf{R}^\top, \mathbf{L}^\top,\ \mathbf{R}^\top = \text{SVD}(\overline{\mathbf{W}})$	14.27	6.55

Finding 2: INT8 quantization can be done without major degradation; NF4 incurs ~2-4 PPL penalty

Finding 3: No obvious relationship between gradient reconstruction and PPL

3 Distributed pretraining

Goal: Memory-constrained distributed training

Setting: Train for 500 steps and construct pseudo-gradient (~DiLoCo; Douillard et al., 2023).

Finding: Identical < Random < Semi-orthogonal (i.e., distributes dimensions across workers)

Method	Projection Init.	200M	1B
Dist. Training (DiLoCo)	–	18.00	12.77
Dist. ReLoRA (LTE)	–	20.97	13.72
Identical Random	$\mathbf{P}_i = \mathbf{P}_j$	21.51	14.28
Independent Random	$\mathbb{E}[\mathbf{P}_i\mathbf{P}_j^\top] = \mathbf{0}$	20.11	13.66
Distributed Random	$\mathbf{P}_i\mathbf{P}_j^\top = \mathbf{0}$	19.81	13.51

Approaches shine with many low-rank workers

Method	(Rank, Workers)		
	(128, 8)	(256, 4)	(512, 2)
Dist. Training (DiLoCo)	17.81	18.00	18.56
Dist. ReLoRA (LTE)	23.76	20.97	19.54
Identical Random	23.96	21.51	20.32
Independent Random	20.64	20.11	19.97
Distributed Random	20.32	19.81	19.66