Introduction to Two Recent Parameter-Efficient Fine-Tuning Approaches

Shaocong Ma

October 9, 2025

PEFT in a Nutshell

Parameter-Efficient Fine-Tuning (PEFT) updates only a tiny fraction of a large model while keeping most pretrained weights frozen (or nearly so).

- **Why:** cut compute/memory, avoid catastrophic forgetting, enable many task adapters per base model.
- **How:** freeze backbone; add small trainable modules or low-rank updates; optionally merge at inference.

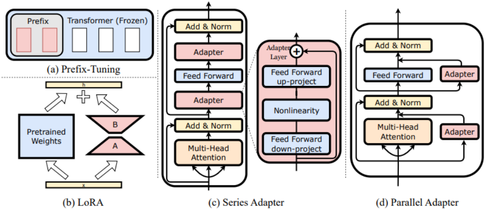


Figure: Many adapters: LoRA, Prefix, Series Adapter, Parallel Adapter, ...

Fig. source: Hu, Zhiqiang, et al. "Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models." arXiv preprint arXiv:2304.01933 (2023).

DoRA: Weight-Decomposed Low-Rank Adaptation

Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, Min-Hung Chen

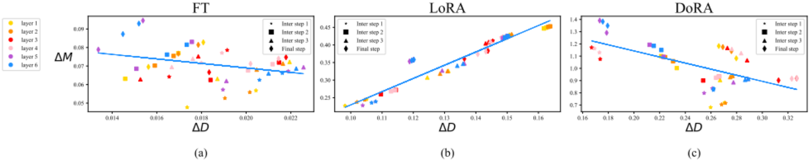
NVIDIA, HKUST

ICML 2024

https://github.com/NVlabs/DoRA

Motivation

- LoRA is popular but still shows a performance gap vs. full fine-tuning (FT).
- **Hypothesis**: the gap is not only about rank/parameter count.
- **Observation**: Weight **update patterns** differ: FT vs. LoRA show distinct magnitude/direction behaviors.
  - *Empirically, FT shows a negative correlation between magnitude and direction changes; vanilla LoRA shows positive.*



- **Goal**: close the gap while keeping **no extra inference cost**.

Key Idea: Decompose & Tune Magnitude and Direction

- Reparameterize a weight matrix $W \in \mathbb{R}^{d \times k}$ into magnitude $m \in \mathbb{R}^{1 \times k}$ and direction $V \in \mathbb{R}^{d \times k}$:

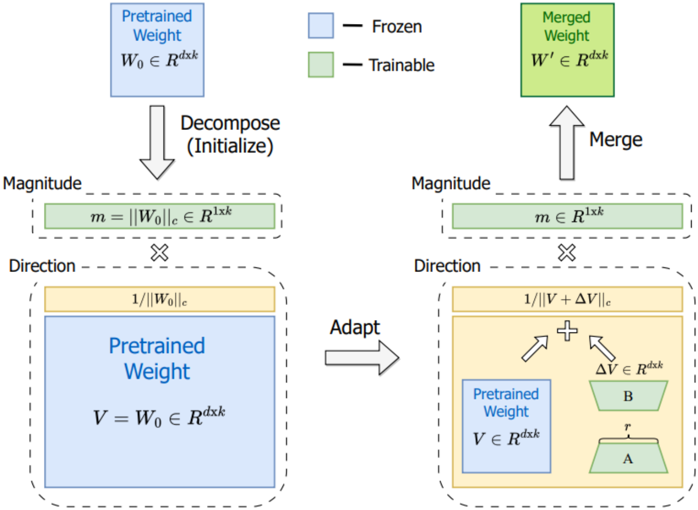
$$W = m \frac{V}{\|V\|_c}, \qquad \text{where } \|\cdot\|_c \text{ is the column-wise norm.}$$

- **DoRA:** Train m directly and adapt the *direction* with a low-rank update (LoRA) BA:

$$W' = m \frac{W_0 + BA}{\|W_0 + BA\|_c}.$$

- Intuition: let LoRA focus on directional changes while m handles scaling.
- Merges into W' for inference $\Rightarrow$ no latency increase.

DoRA Overview Figure



Why DoRA Helps (Analysis)

- **Update patterns:** Empirically, FT shows a *negative* correlation between magnitude and direction changes; vanilla LoRA shows *positive*. DoRA matches FT-like behavior.
- **Gradient view:** Decomposition projects gradients away from the current weight direction and scales them by $m/\|V'\|_c$, improving conditioning and stability.
- **Practical tweak:** Treat $\|V'\|_c$ as a constant in backprop to reduce memory ($\sim$12–24% less during training) with negligible accuracy change.

Results (Highlights)

- **Commonsense reasoning (LLaMA/LLaMA2/LLaMA3):** DoRA consistently outperforms LoRA.
  - $+3.7$ (LLaMA-7B), $+1.0$ (LLaMA-13B), $+2.9$ (LLaMA2-7B), $+4.4$ (LLaMA3-8B) average points across 8 tasks.
- **Multimodal:** On VL-BART multi-task image/video–text understanding, DoRA $\geq$ LoRA while keeping efficiency.
- **Training stability:** Smaller deviations from pretrain in both magnitude and direction, yet better accuracy.
- **Inference cost:** unchanged vs. LoRA (mergeable weights).

Conclusion (DoRA)

- Weight-decomposed PEFT that better matches FFT learning patterns[1].
- Improves accuracy over LoRA across LLM and LVLM tasks.
- Keeps **PEFT virtues**: low trainable params, merge-before-inference, stable training.

[1] Another paper identifies the different learning patterns between FFT and LoRA: *Yen, Jui-Nan, et al. "LoRA Done RITE: Robust Invariant Transformation Equilibration for LoRA Optimization." ICLR 2025*

HydraLoRA: An Asymmetric LoRA Architecture for Efficient Fine-Tuning

Chunlin Tian, Zhan Shi, Zhijiang Guo, Li Li, Chengzhong Xu

University of Macau, UT Austin, University of Cambridge

NeurIPS 2024

https://arxiv.org/abs/2404.19245

Motivation

- PEFT methods underperform FFT, particularly in scenarios involving complex datasets.

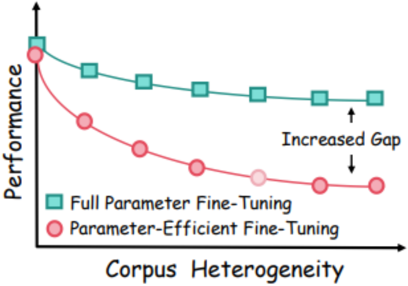


Figure: When dataset is more complicated, PEFT performs increasingly worse than FFT.

Motivation

- **Observation I**: multiple *smaller* LoRA heads per task beat a single monolithic LoRA with the same parameter budget (reduces interference).

Schemes	$r \times n$	MMLU ↑	% Parameter
LoRA	8×1	43.22	0.062
LoRA	16×1	45.45	0.124
LoRA	32×1	46.59	**0.248**
LoRA (Split)	16×2	46.82	0.248
LoRA (Split)	8×4	**46.94**	0.248
LoRA (Split)	4×8	46.83	0.248

Figure: 4×8 heads LoRA is better than 1×32 head LoRA on Dolly-15K dataset; evaluated on MMLU.

Motivation

- **Observation II**: across tasks, LoRA **A** matrices converge (shared commonality), **B** matrices diverge (task-specific).

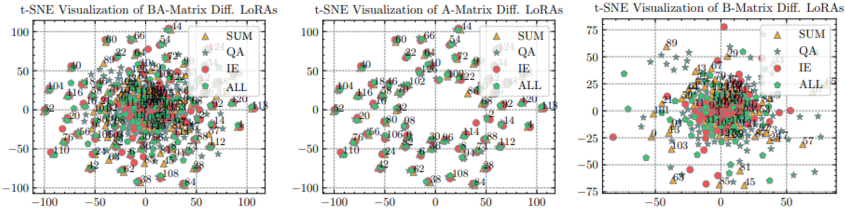


Figure 3: Breakdown analysis of LoRA modules. Compare fine-tuned LoRA modules of Dolly-15K [8] with three subtasks of Dolly-15K including "*summarization (Sum)*", "*closed QA (QA)*" and "*information extraction (IE)*" using t-SNE. Consider LLaMA2-7B (random seed=42), which contains 32 decoder layers, corresponding to 32 adaptive modules. Each module consists of {**0**: q_proj of A, **1**: q_proj of B, **2**: v_proj of A, **3**: v_proj of B} submodules. This makes a total of 32 × 4 submodules. Left displays all submodules. Center shows all even submodules, i.e. the A matrix. Right represents all odd submodules, i.e. the B matrix. It can be seen that the differences in the fine-tuned LoRA modules for different tasks arise mainly from the B matrix.

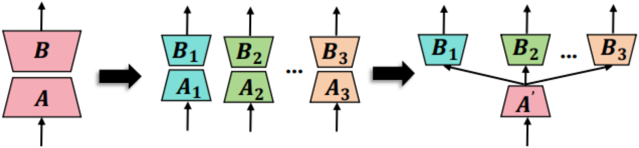
Figure: For different tasks, A matrice concentrates, while B matrices diverge.

HydraLoRA Architecture

- **Asymmetric LoRA:** share one A, learn multiple B_i "heads," and let a router mix them:

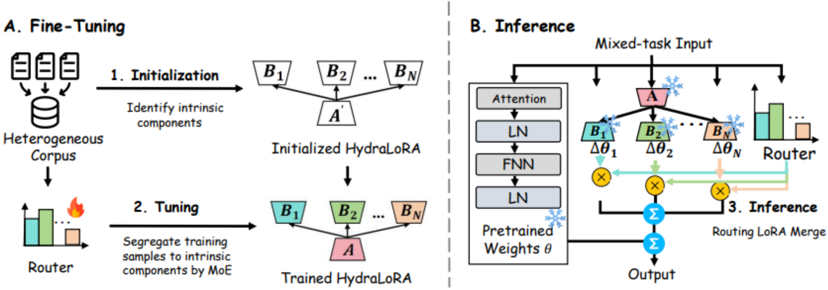
$$W = W_0 + \sum_{i=1}^{N} \omega_i B_i A.$$

- **MoE-style routing:** $\omega = \mathrm{softmax}(W_g^\top x)$ chooses/weights experts (B_i) per input.
- **End-to-end:** discovers intrinsic components (subdomains) automatically; no domain heuristics required.



(a) (b) (c)

HydraLoRA Workflow



Results (Highlights)

- **Single-domain instruction tuning (LLaMA2-7B):** HydraLoRA improves MMLU/Medical/Law, HumanEval@1/@10, GSM8K over LoRA; also surpasses LoRA-Split with fewer params.
- **Multi-task (BBH) with LLaMA2-7B/13B:** HydraLoRA $>$ LoRA, LoraHub, and LoRA-MoE under comparable budgets.
- **Efficiency:** On GSM8K (LLaMA2-7B), $\sim$1.96$\times$ faster training and $\sim$49.6% lower energy vs. LoRA (rank=32) with competitive or better quality.
- **Ablations:** removing MoE or gating or the hydra split degrades performance; full design is best.

Takeaways

- **DoRA:** FT-like learning behavior via magnitude/direction decoupling; higher accuracy than LoRA without inference cost.
- **HydraLoRA:** shared-A + multi-B with routing handles data heterogeneity; better quality/efficiency than monolithic LoRA.
- Both are **drop-in** PEFT upgrades you can try before paying FFT costs.

Thank You!