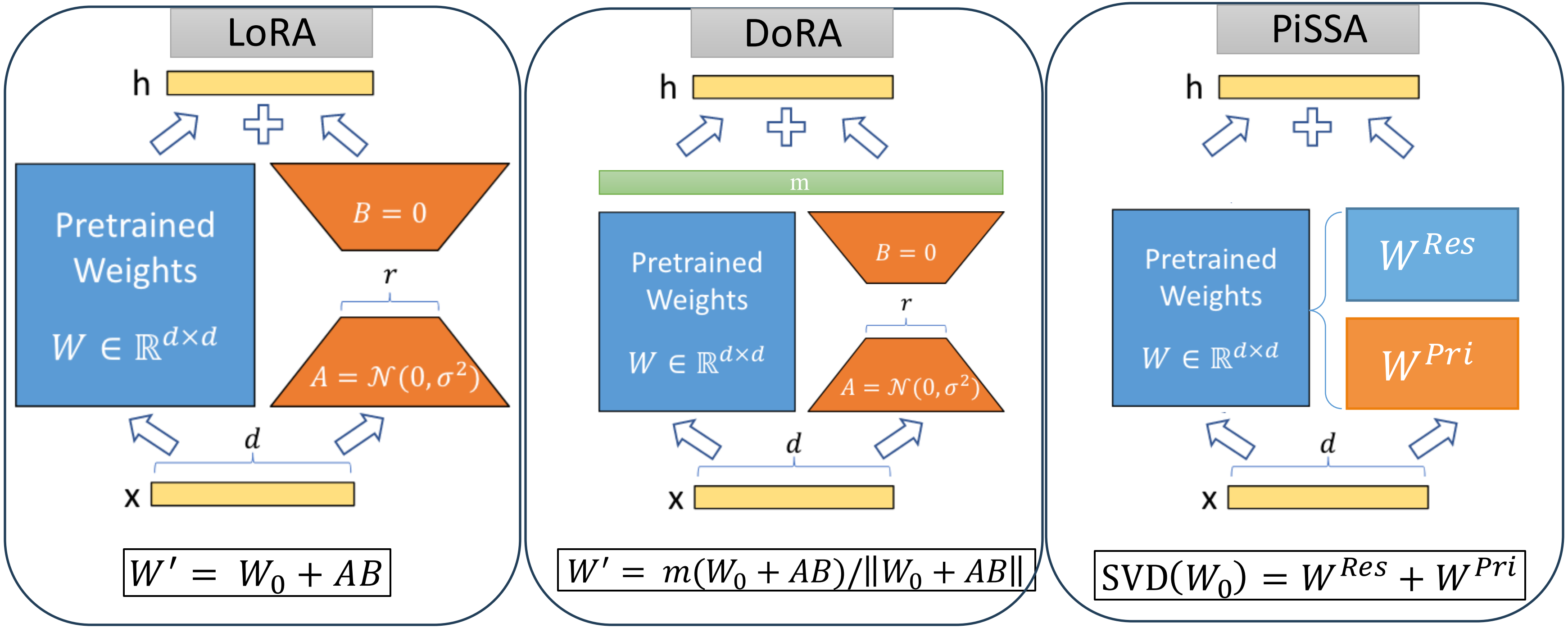
From Weight-Based to State-Based Fine-Tuning: Further Memory Reduction on LoRA with Parallel Control

Chi Zhang, Lianhai Ren, Jingpu Cheng, Qianxiao Li

National University of Singapore

01

Understanding LoRA and PEFT: A Weight-Tuning Perspective

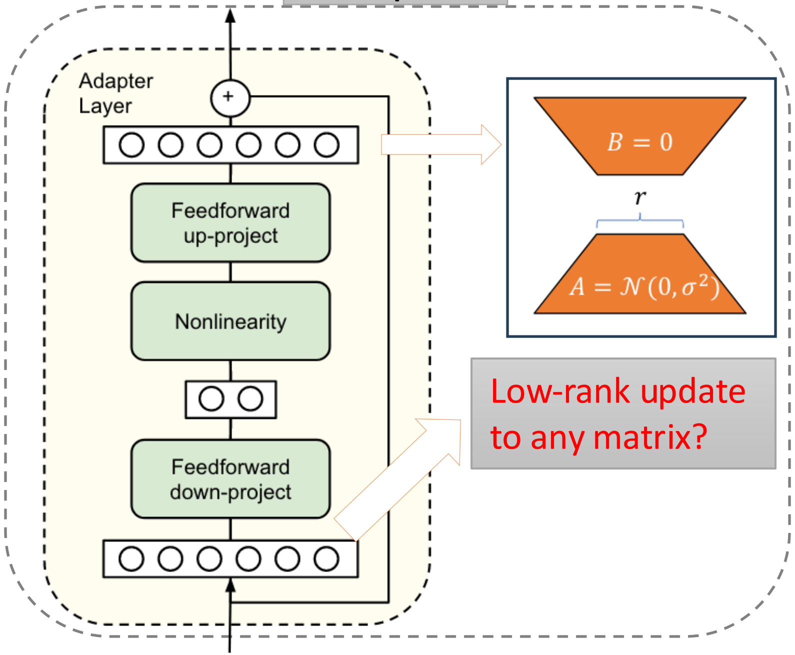


- The key idea is “Weight-Tuning”

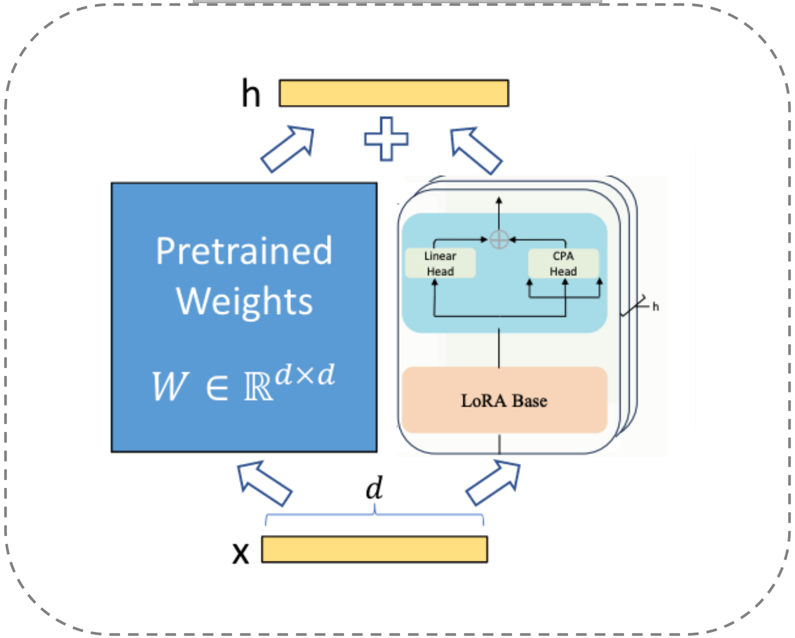
02

Challenging the Weight Tuning View: Counter-Examples

Adapter



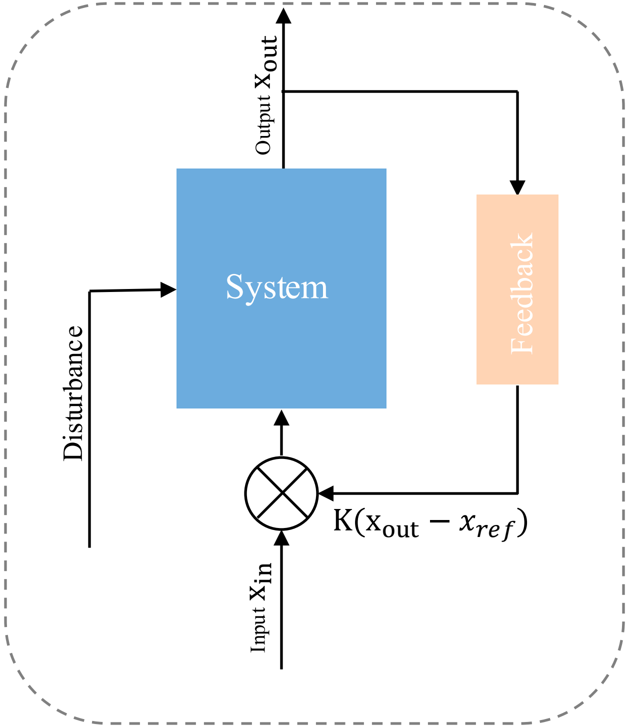
Attention Modules



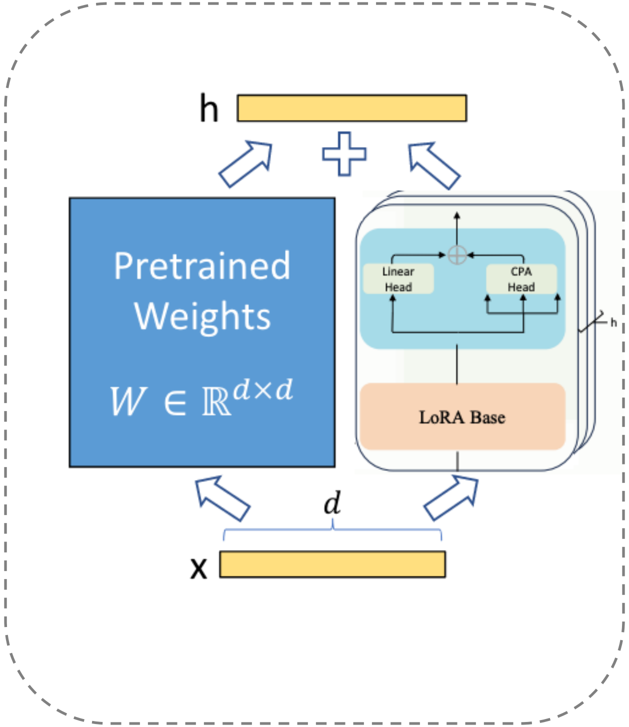
1. Houlsby et al., *Parameter-Efficient Transfer Learning for NLP*, ICML 2019
2. Zhang et al., *Parameter-Efficient Fine-tuning with control,* ICML 2024

03 Change Our Mindset From Finetuning to Control

Feedback Control



PEFT



04 Control Theory Beneath the Surface: Fine-Tuning as Implicit Control

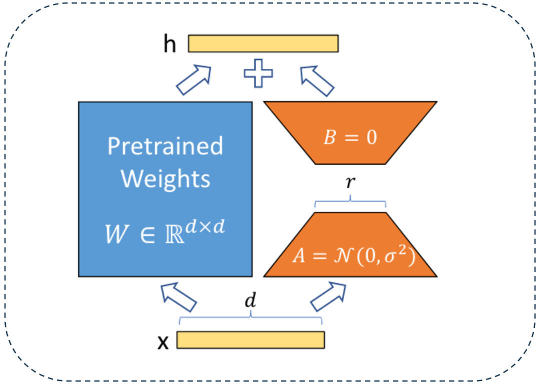
1. A Linear Time-Invariant (LTI) System with Control

$$\dot{x}_t = A\,x_t + B\,u_t$$

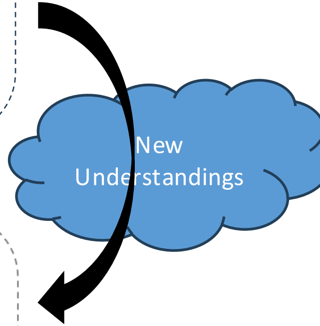
Finetuning a layer of NN models:

$$x_{t+1} = x_t W + x_t U = x_t (W + U)$$

If $\mathrm{U} = \mathrm{A_r B_r}$, it recovers the LoRA case.



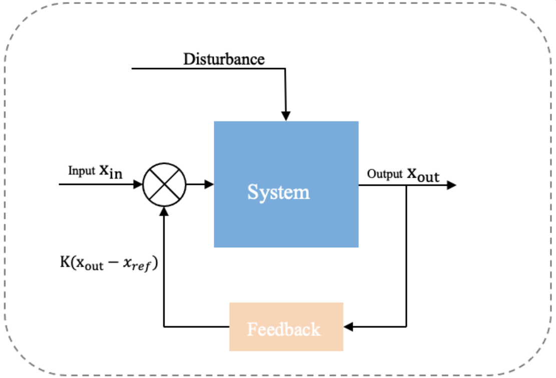
2. Although originally seen as weight-tuning techniques, LoRA and other PEFT methods implicitly design control systems.



3. In the LTI system, the x_t and u_t do not need to have the same shape, e.g., $x_t \in R^n$, $u_t \in R^m$. Similarly, the control matrix U does not need to have the same shape as the original matrix W. Moreover, $x_t U$ can be g(u,x_t).

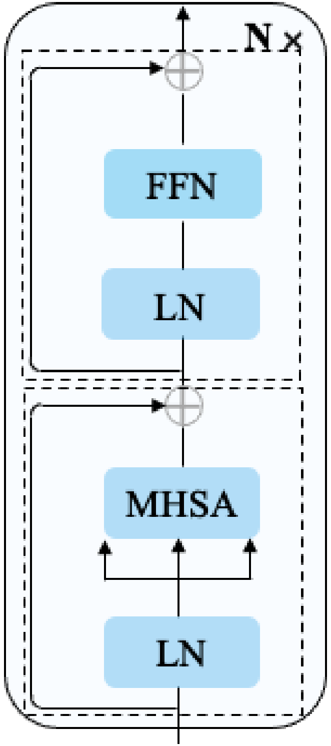
4. Yet, there are still notable distinctions:

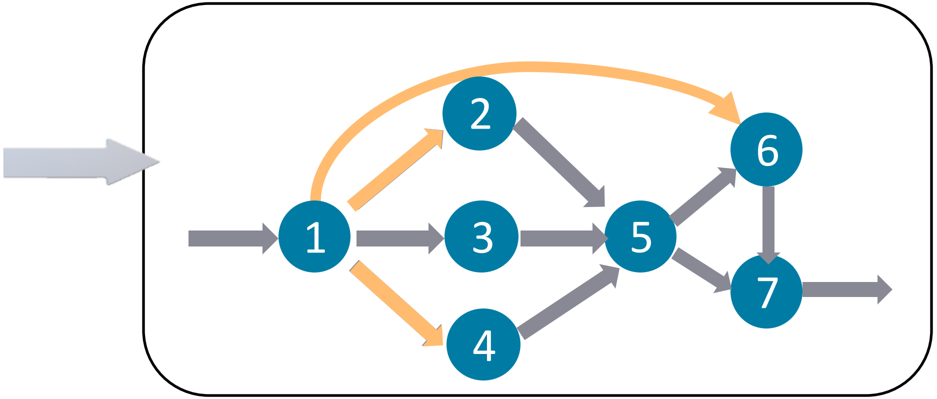
- Control is typically applied to **states** — not directly to weight matrices.
- Feedback control minimally disrupts the original system — instead, it introduces perturbations to the system as a whole.
- Control not only edits edges — it can create new ones.



05

From Weight-Based to State-Based Finetuning





1. Neural network is a directed acyclic graph G = (V,E). Computation on the edge $(u, v) \in E$ is defined as:

$$x_v^u = f_v^u(x_u; W_{u \to v})$$

2. State-based tuning involves modifying the intermediate states x_v with a control function:

$$x_v' = \sum_{\tilde{u}} x_v^{\tilde{u}} + g_v^u(M_{u \to v}, x_u)$$

3. With state-based tuning, we are free to choose any starting/ending state.

ICML International Conference On Machine Learning

06

One Example of State-Based Finetuning: Towards a Control-Affine System

1. Linear Time-Invariant (LTI) dynamics

$$\dot{x}_t = Ax_t + Bu_t$$

2. Nonlinear dynamics with time–varying control matrices

$$\dot{x}_t = f(x_t) + B_t u_t$$

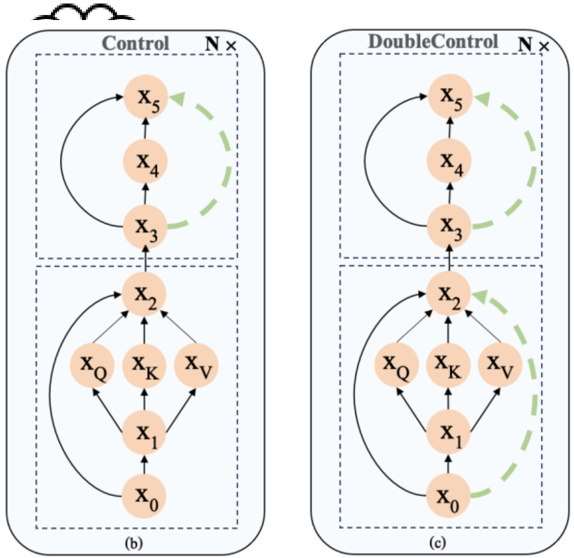
3. Control-Affine Nonlinear System:

$$\dot{x}(t) = f(x(t)) + \sum_{i=1}^{m} g_i(x(t))u_i(t)$$

4. Fully nonlinear control systems:

$$\dot{x}_t = f(x_t, u_t)$$

$$x_{t+1} = \text{softmax}\left(\frac{x_t(W_t^Q + U_t^Q)(x_t W_t^K)^\top}{\sqrt{d_k}}\right) \cdot x_t(W_t^V + U_t^V) \cdot W_t^O + x_t$$



$$x_{t+1} = \text{softmax}\left(\frac{(x_t W_t^Q)(x_t W_t^K)^\top}{\sqrt{d_k}}\right) x_t W_t^V W_t^O + x_t + x_t U_t$$

$$x_{t+1} = \sigma\left(x_t W_{\text{MLP}_1}\right) W_{\text{MLP}_2} + x_t + x_t U_t$$

07 Performance Analysis

Consider a deep linear network defined as:

$$f : x_0 \to x_T, \quad x_{t+1} = x_t W_t, \quad t = 0, \ldots, T,$$

and its low-rank adaptation:

$$\bar{f} : x_0 \to x_T, \quad x_{t+1} = x_t(W_t + U_t), \quad t = 0, \ldots, T-1,$$

where $x_t \in \mathbb{R}^{d_t}$ represents the hidden state, $W_t \in \mathbb{R}^{d_t \times d_{t+1}}$ is the weight matrix at layer t, and $U_t \in \mathbb{R}^{d_t \times d_{t+1}}$ is a low-rank matrix with rank r_t. Then, there exists a weight matrix M satisfying

$$\text{rank}(M) \leq r_0 + \cdots + r_{T-1},$$

such that for all $x_0 \in \mathbb{R}^{d_0}$,

$$\bar{f}(x_0) = f(x_0) + x_0 M.$$

08

Performance Analysis (Cont.)

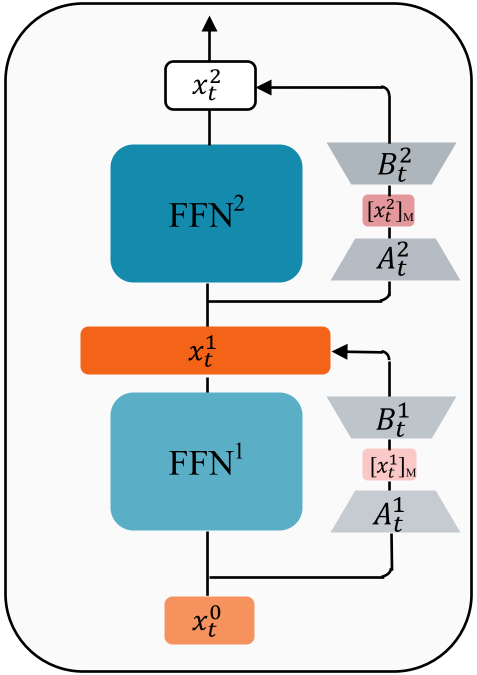
Let F_{x_t} and G_{x_t} be the mappings $U_t \mapsto x_{t+1} \in \mathbb{R}^d$, as defined in equations (1) and (2), respectively.

$$x_{t+1} = x_t + f_t\left(x_t(W_t + U_t)\right), \tag{1}$$

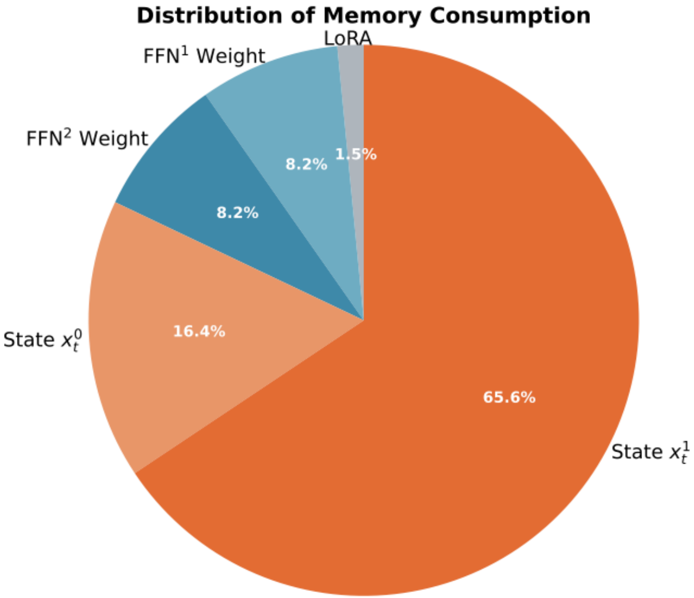
$$x_{t+1} = x_t + f_t\left(x_t W_t\right) + x_t U_t, \tag{2}$$

If $\nabla f(x_t)$ is singular, then the pushforward of the tangent space at 0 under F_{x_t} forms a proper subspace of $\mathbb{R}^d$. In contrast, the pushforward of the tangent space at 0 under G_{x_t} always spans the entire space $\mathbb{R}^d$, as long as x_t is non-zero.

09 Memory Consumption Analysis



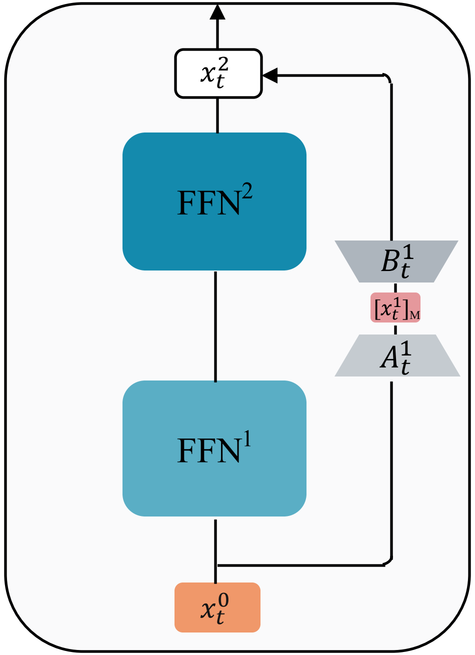
Batch size = 16, Seq Length = 1024, Feature Dimension = 4096

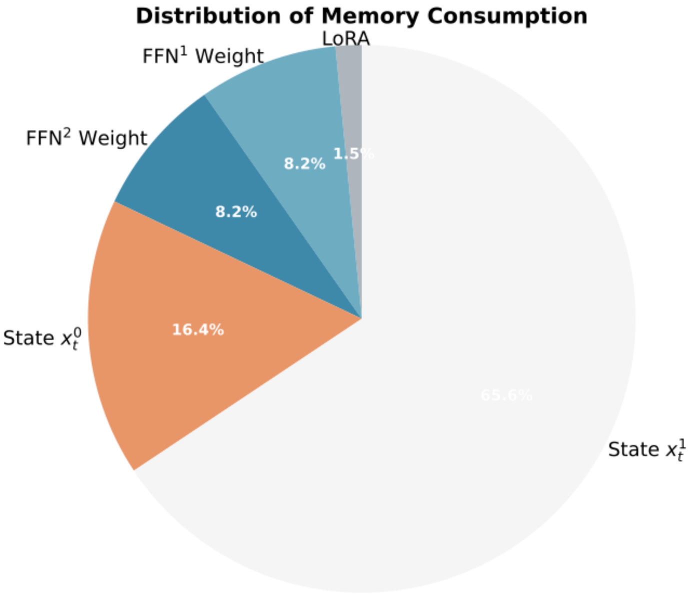


$$\frac{\partial \ell}{\partial A_t^2} = \frac{\partial \ell}{\partial x_t^2} \frac{\partial x_t^2}{\partial [x_t^2]_M} \frac{\partial [x_t^2]_M}{\partial A_t^2} \rightarrow x_t^1$$

10

Memory Reduction by Parallel Control





What consumes the memory?

- Backward States (LoRA)
- Model Weight (QLoRA)
- Forward States (Control)

Train 7B/8B Models on Nvidia-3090

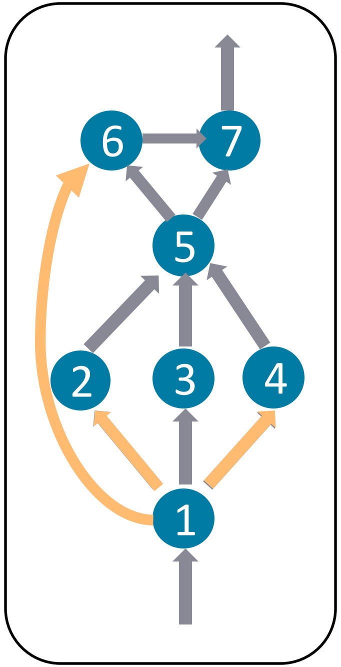
Table 1. Comparison on the Commonsense benchmark.

Model	Method	# of Params	GPU Memory	Training Time	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg
ChatGPT †	-	-	-	-	73.1	85.4	68.5	78.5	66.1	89.8	79.9	74.8	77.0
LLaMA2-7B	LoRA (QKVUD) †	56.10 M	44.204 GB	8h37m	69.8	79.9	79.5	83.6	82.6	79.8	64.7	81.0	77.6
	DoRA (QKVUD) †	56.98 M	59.568 GB	14h50m	71.8	**83.7**	76.0	89.1	82.6	**83.7**	68.2	**82.4**	79.7
	Control(UD)+LoRA(QKV)	41.94 M	38.556 GB	7h36m	**73.0**	83.5	**79.5**	**89.7**	82.6	82.9	**68.6**	80.4	**80.0**
	DoubleControl (QKVUD)	33.55 M	35.214 GB	6h58m	72.3	82.5	79.2	89.1	**83.1**	83.0	68.5	79.0	79.6
LLaMA3-8B	LoRA (QKVUD) †	56.62 M	55.040 GB	9h33m	70.8	85.2	79.9	91.7	84.3	84.2	71.2	79.0	80.8
	DoRA (QKVUD) †	57.41 M	67.284 GB	15h15m	74.6	**89.3**	79.9	**95.5**	85.6	90.5	**80.4**	85.8	85.2
	Control(UD)+LoRA(QKV)	35.65 M	48.550 GB	8h11m	**75.7**	87.9	**80.4**	**95.5**	**86.3**	90.6	79.8	86.2	**85.3**
	DoubleControl (QKVUD)	33.55 M	45.316 GB	7h44m	74.1	87.8	**80.7**	**95.5**	86.0	**90.8**	80.0	**87.8**	**85.3**

Model	Method	# of Params	GPU Memory	Training Time	BoolQ	PIQA	SIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	OBQA	Avg
LLaMA2-7B	Control(UD)+LoRA(QKV)	41.94 M	21.874 GB	21h21m	71.4	81.1	75.7	86.7	82.9	82.3	67.2	80.4	78.4
	DoubleControl (QKVUD)	33.55 M	20.656 GB	20h09m	70.8	83.0	79.2	84.6	81.5	82.8	68.3	81.2	78.9
LLaMA3-8B	Control(UD)+LoRA(QKV)	35.65 M	22.920 GB	21h51m	75.1	87.8	79.9	95.3	85.0	90.0	79.0	85.0	84.6
	DoubleControl (QKVUD)	33.55 M	22.176 GB	20h33m	74.4	86.9	80.4	95.3	85.4	90.1	79.4	85.6	84.7

12 Summary

- **Changing the Mindset:** although initially framed as **weight-tuning** techniques, PEFT methods reveal a deeper connection to **classical control theory**, with each block serving as a controller.
- **From Weight-based to State-based:** similar to control theory, the focus of tuning can shift from **weights to states** — allowing us to modify arbitrary states in the graph by **updating existing edges** or **introducing new ones**.
- **Example:** One particular example is designing a **control-affine** system, where a parallel scheme can be adopted with minimal modification to the original system. This allows for simpler theoretical analysis and reduces memory usage by skipping large components.



czhang24@nus.edu.sg

THANK YOU