

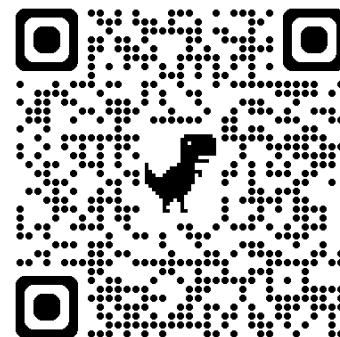


深度學習

Deep Learning

輕量化訓練大型模型的方法

Instructor: 林英嘉 (Ying-Jia Lin)
2025/05/12



[Course GitHub](#)



[Slido # DL_0512](#)

Outline

- 輕量化訓練大型模型的方法
 - Parameter-efficient fine-tuning (PEFT)



The Revolution of ChatGPT

ChatGPT came out in November, 2022.

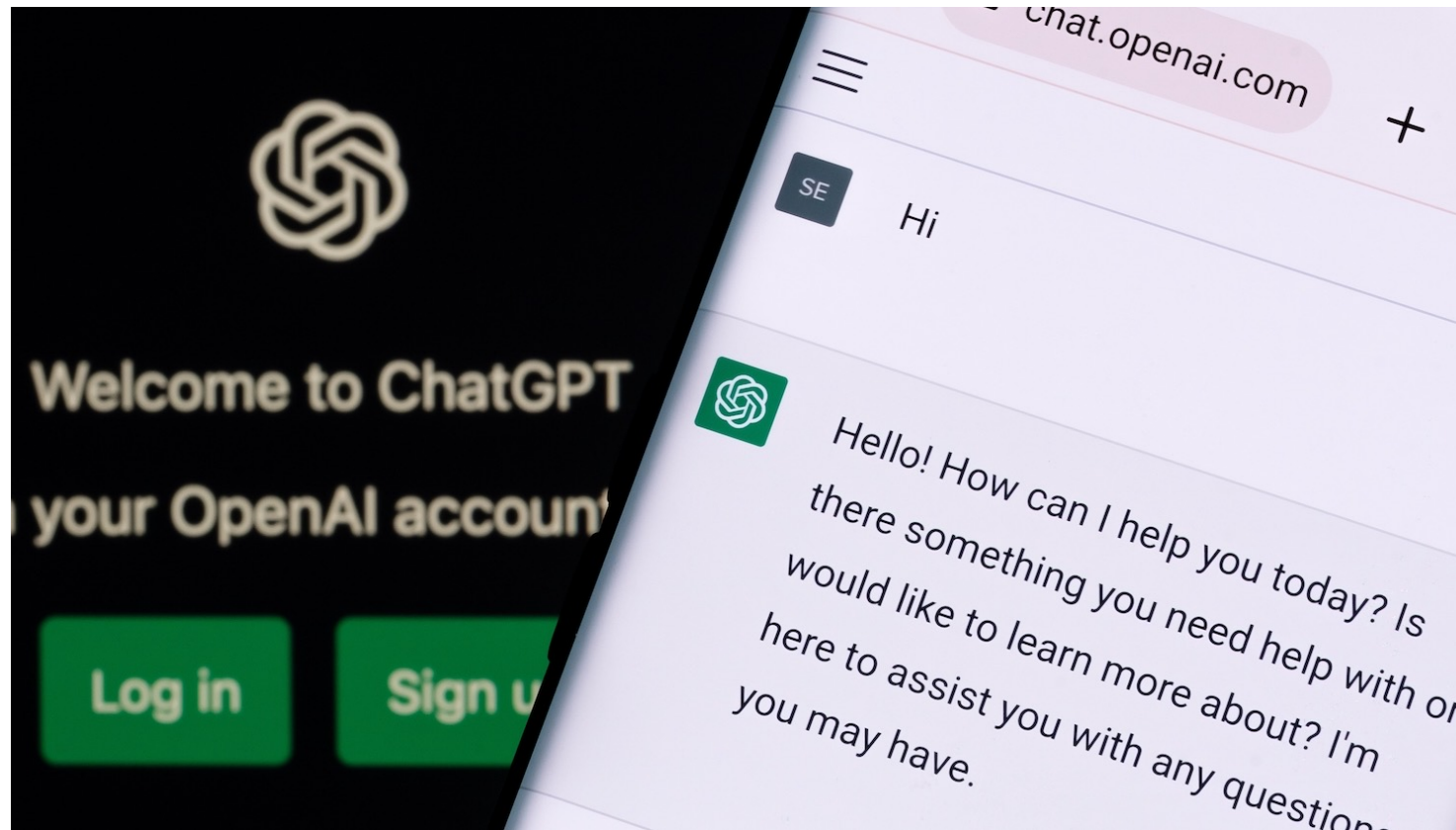
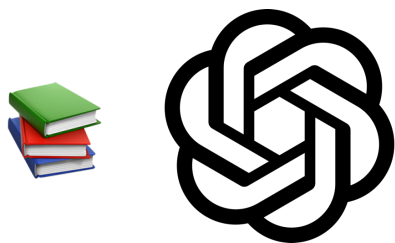


Figure source: <https://poole.ncsu.edu/thought-leadership/article/lets-chat-about-chatgpt/>
<https://openai.com/index/chatgpt/>



讓大型語言模型適用於你的任務？



翻譯、聊天、寫故事 ...



醫學資料、少數語言、網路上查不到的知識

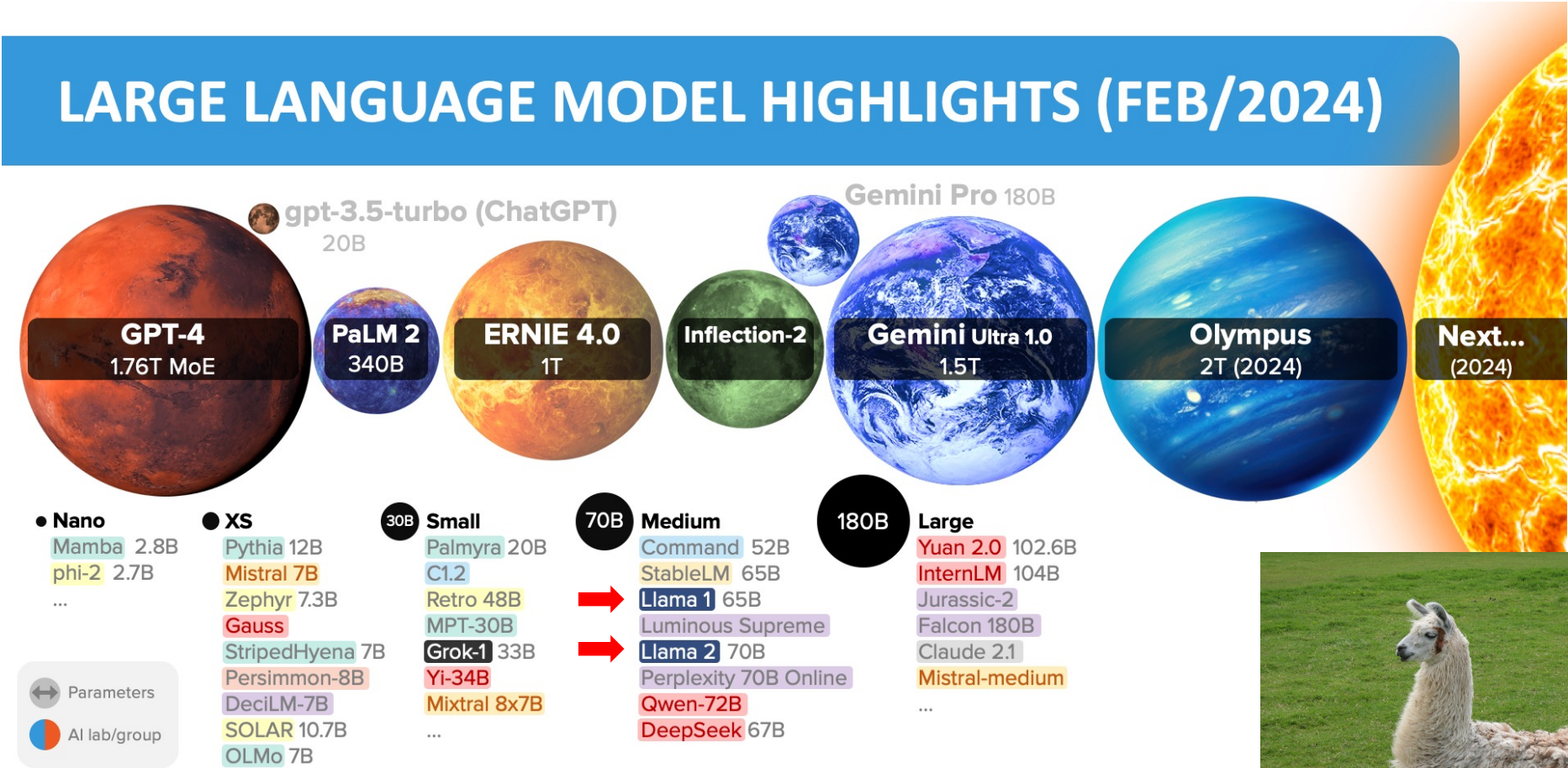


解決方案：Fine-tuning 語言模型



Figure source:
https://gameofthrones.fandom.com/wiki/High_Valyrian?file=Nekeesse_Valyrio.jpg

Full Fine-tuning (全微調) LLM 的困境



Sizes linear to scale. Selected highlights only. All models are Chinchilla-aligned (20:1 tokens:parameters) <https://lilearchitect.ai/chinchilla/> All 200+ models: <https://lilearchitect.ai/models-table/> Alan D. Thompson, 2023-2024.

Source: Inside language models (from GPT-4 to PaLM) – Dr Alan D. Thompson – Life Architect



Source: https://commons.wikimedia.org/wiki/File:Llama_lying_down.jpg



GPU Memory Estimated (Model weights)

以 Llama 2-7B 為例，且儲存參數的精度為 float 32 (FP32)：

在 float 32 (FP32) 的情況下，1 個參數需要 4 bytes 儲存

-> 7B = 70億個參數 = $7,000,000,000 * 4 \text{ bytes}$ = (扣掉3個逗號從B換成GB) **28GB**



Introduction to Quantization

(這頁只是示意圖，數值不精準)

$$\begin{bmatrix} -0.4 & 1.3 & 3.73 \\ -4.7 & -3.2 & -6.4 \\ 8.5 & 14.3 & 13.5 \end{bmatrix} \xrightarrow[\text{FP32} \rightarrow \text{int8}]{\text{降低儲存精度}} \begin{bmatrix} 0 & 1 & 4 \\ -5 & -3 & -6 \\ 9 & 14 & 14 \end{bmatrix}$$

36 bytes **8 bytes**

32-bit floating point (FP32): 1個
值需要**4個bytes**才能儲存

8-bit Integer (int8): 1個值需要
1個bytes才能儲存

誤差：

$$\begin{bmatrix} 0.4 & -0.3 & 0.27 \\ -0.3 & 0.2 & 0.4 \\ 0.5 & -0.3 & 0.5 \end{bmatrix}$$



GPU Memory Estimated (Model weights)

以 Llama 2-7B 為例，且儲存參數的精度為 float 32 (FP32)：

在 float 32 (FP32) 的情況下，1 個參數需要 4 bytes 儲存

-> 7B = 70億個參數 = $7,000,000,000 * 4 \text{ bytes}$ = (扣掉3個逗號從B換成GB) **28GB**

模型	參數量	Memory (FP32)	Memory (FP16)
DeepSeek v3	685B	2740 GB	1370 GB
Llama 4 Scout	109B	436 GB	218 GB
GPT-2 XL	1.5B	6 GB	3 GB



GPU Memory Estimated (Full Fine-tuning)

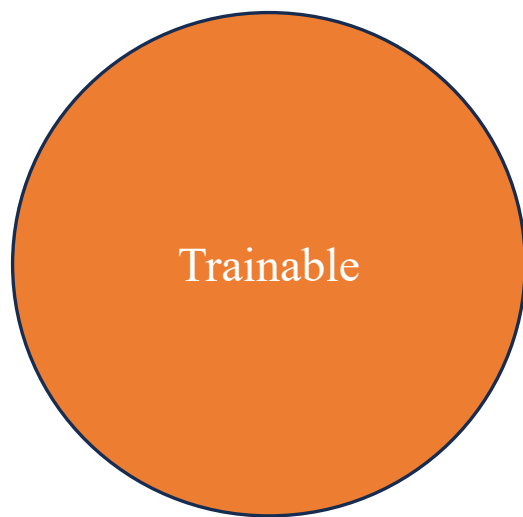
Llama 2-7B		
16-bit float, max_length (seq) = 4096, hidden_size = 4096, batch_size (bs) = 1		
	算法	Memory
CUDA	-	~1 GB
Model weights	$\text{size(float)} * N_{\text{parameter}}$	13.03 GB
Gradients	$\text{size(float32)} * N_{\text{trainable}}$	26.06 GB
Hidden states	$\sim \text{size(float)} * \text{seq} * \text{hidden_size} * L$	1.07 GB
Optimizer states (Adam)	$2 * \text{size(float)} * N_{\text{trainable}}$	26.06 GB

L : Number of layers in model (eq. 32 layers)

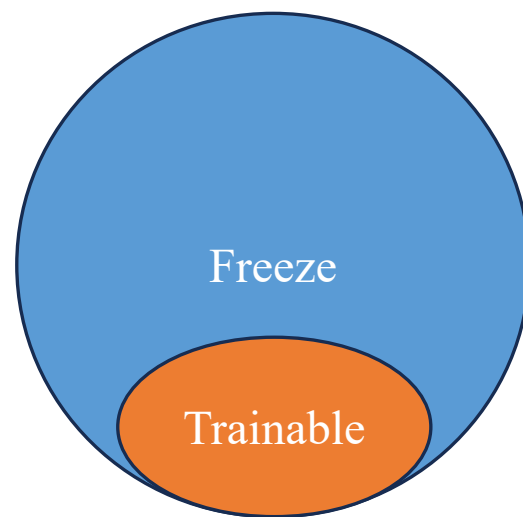
Estimate: 67.22 GB
*NVIDIA 5090: 32GB



有沒有可能只微調 LLM 一部分的參數呢？



Full Fine-tuning
Update **all model parameters**



Parameter-efficient Fine-tuning (PEFT)
Update a **small subset** of model parameters



GPU Memory Estimated (PEFT)

- 假設我們只更新 2% 的參數

Llama 2-7B 16-bit float, max_length (seq) = 4096, hidden_size = 4096, batch_size (bs) = 1		
	算法	Memory
CUDA	-	~1 GB
Model weights	$\text{size(float)} * N_{\text{parameter}}$	13.03 GB
Gradients	$\text{size(float32)} * N_{\text{trainable}}$	$13.03 * 0.02 = 0.2606 \text{ GB}$
Hidden states	$\sim \text{size(float)} * \text{seq} * \text{hidden_size} * L$	1.07 GB
Optimizer states (Adam)	$2 * \text{size(float)} * N_{\text{trainable}}$	0.5212 GB

L : Number of layers in model (eq. 32 layers)

Estimate: **15.88 GB**

*NVIDIA 4060 Ti: 16 GB



PEFT Outline

Adapters

LoRA

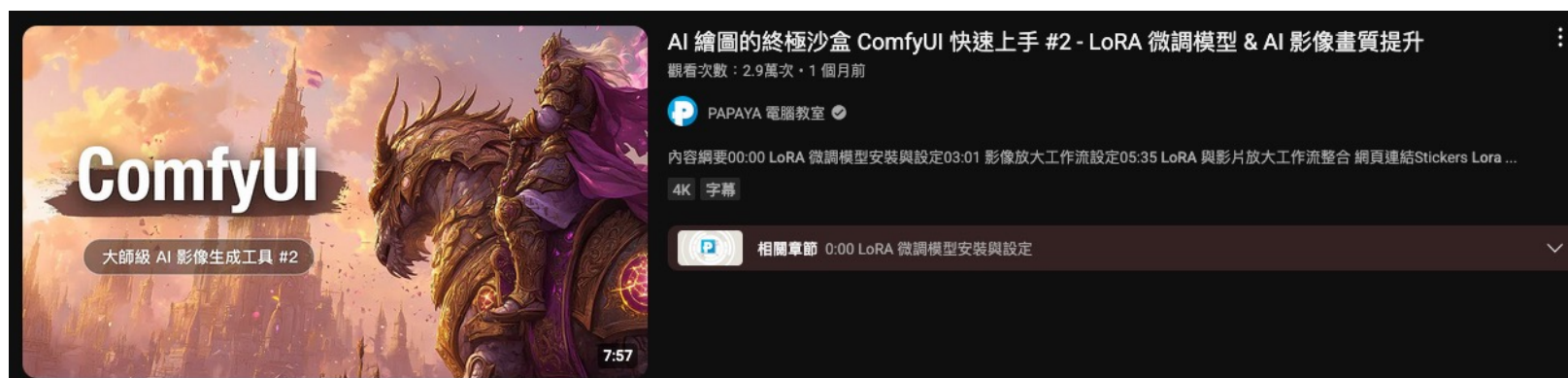
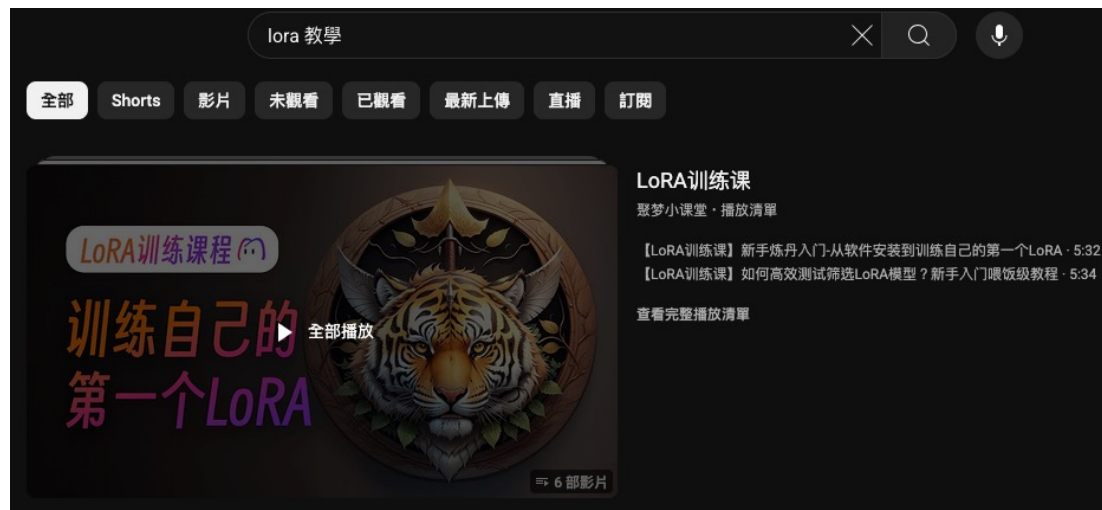
Prefix-
Tuning

BitFit

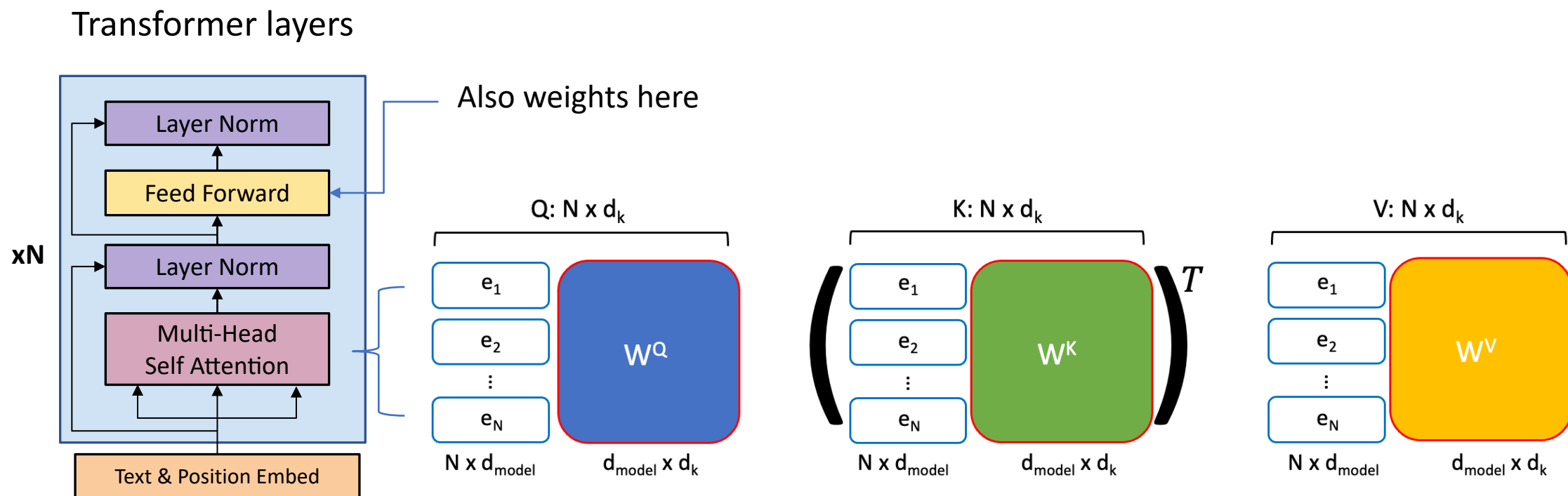
LoRA: **Lo**w-**R**ank **A**daptation



LoRA is common ...



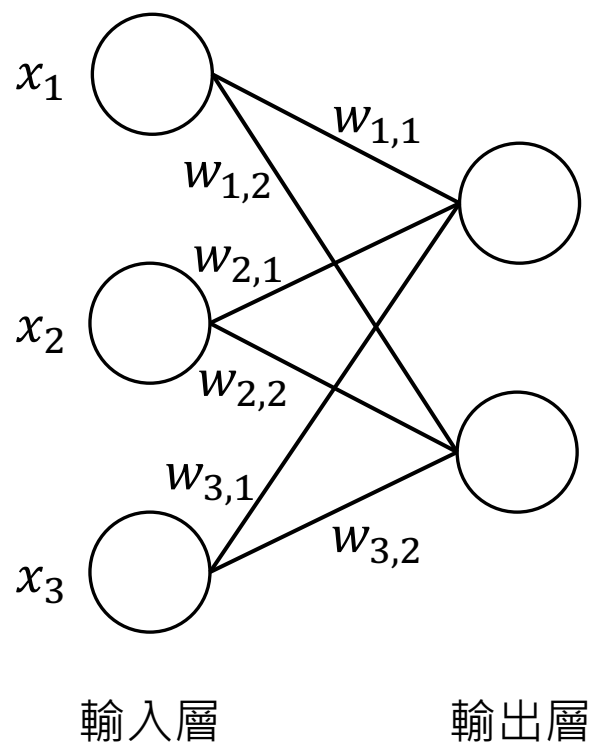
[Prerequisite] Weights in Transformer layers



$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$



[Recap] MLP is composed of weight matrices



$$\begin{matrix} [x_1 & x_2 & x_3] \\ \mathbb{R}^{1 \times 3} \end{matrix} \times \begin{matrix} \boxed{\begin{bmatrix} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \\ w_{3,1} & w_{3,2} \end{bmatrix}} \\ \mathbb{R}^{3 \times 2} \end{matrix} + \text{bias} \quad \mathbb{R}^2$$



[Prerequisite] 矩陣分解



$$d = 1000$$



$$r = 10$$

參數量

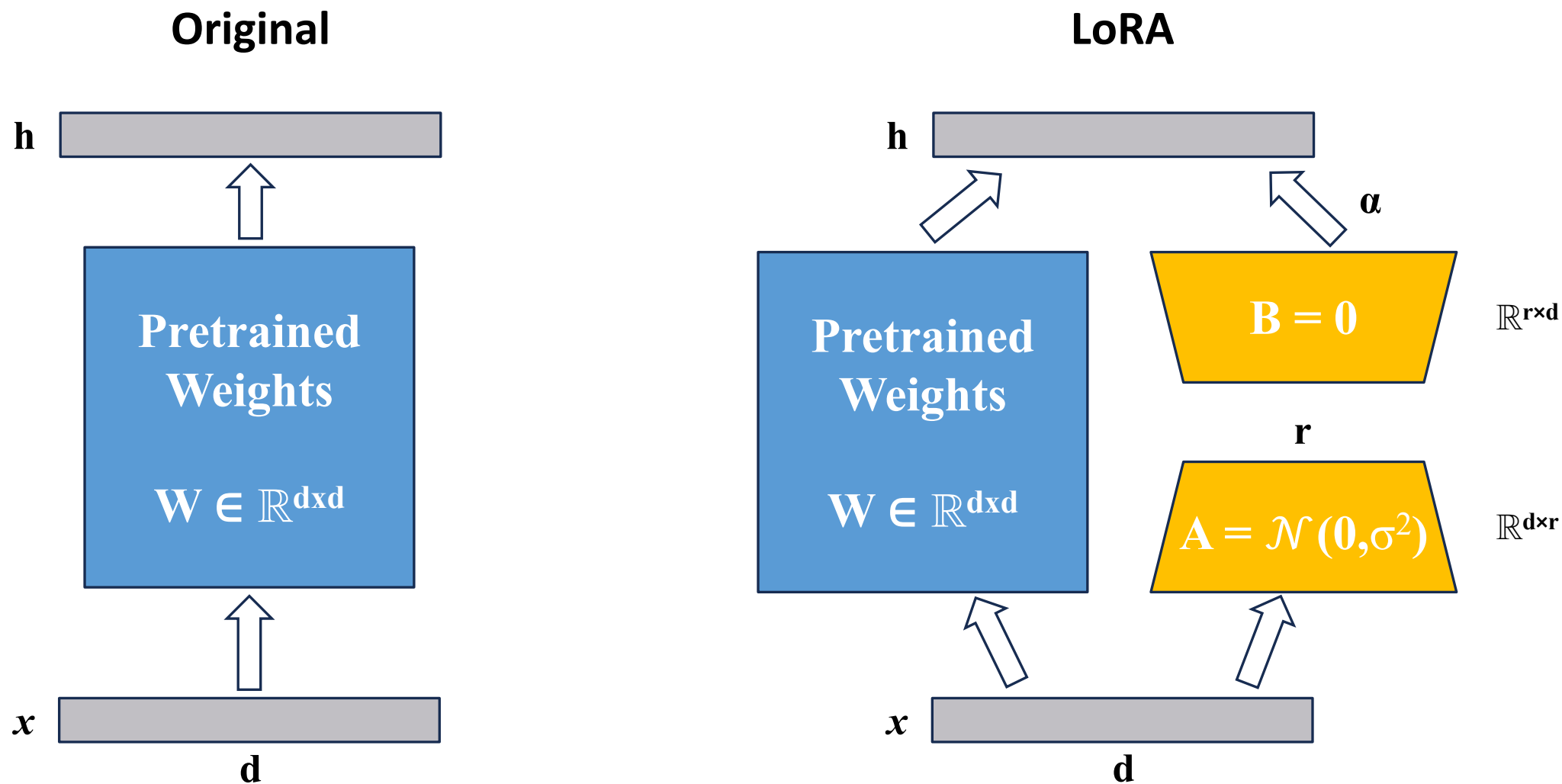
$$1000 \times 1000 = 1 \text{百萬}$$

$$1000 \times 10 + 10 \times 1000 = 20000$$

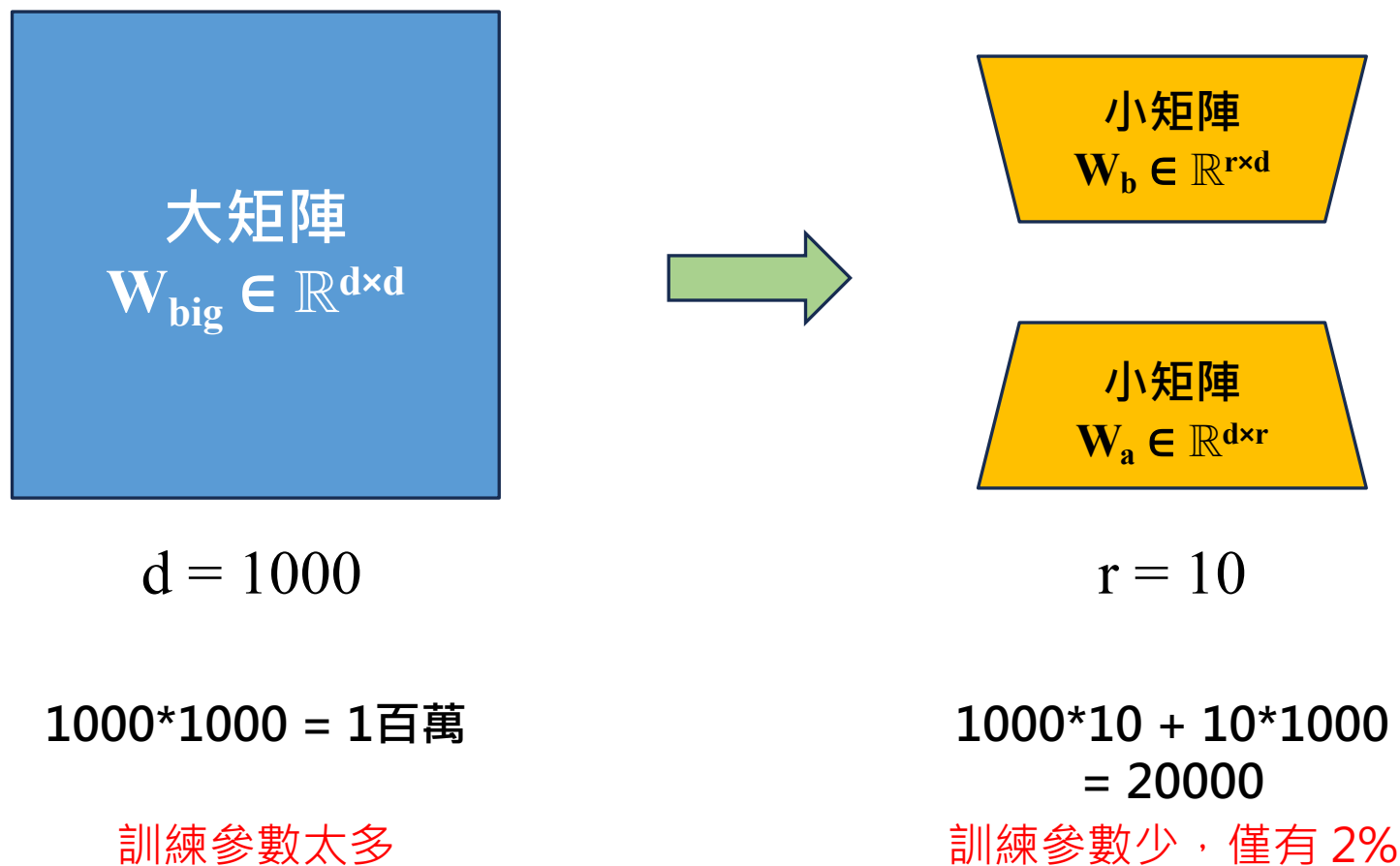


LoRA: Low-Rank Adaptation

只有黃色部分會被訓練
 α 是超參數



為什麼 LoRA 要這樣做？



Low-Rank 的部分在哪？(1)

$$\begin{bmatrix} \checkmark & \checkmark & & \checkmark \\ 1 & 0 & -1 & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Reduced row-
echelon form

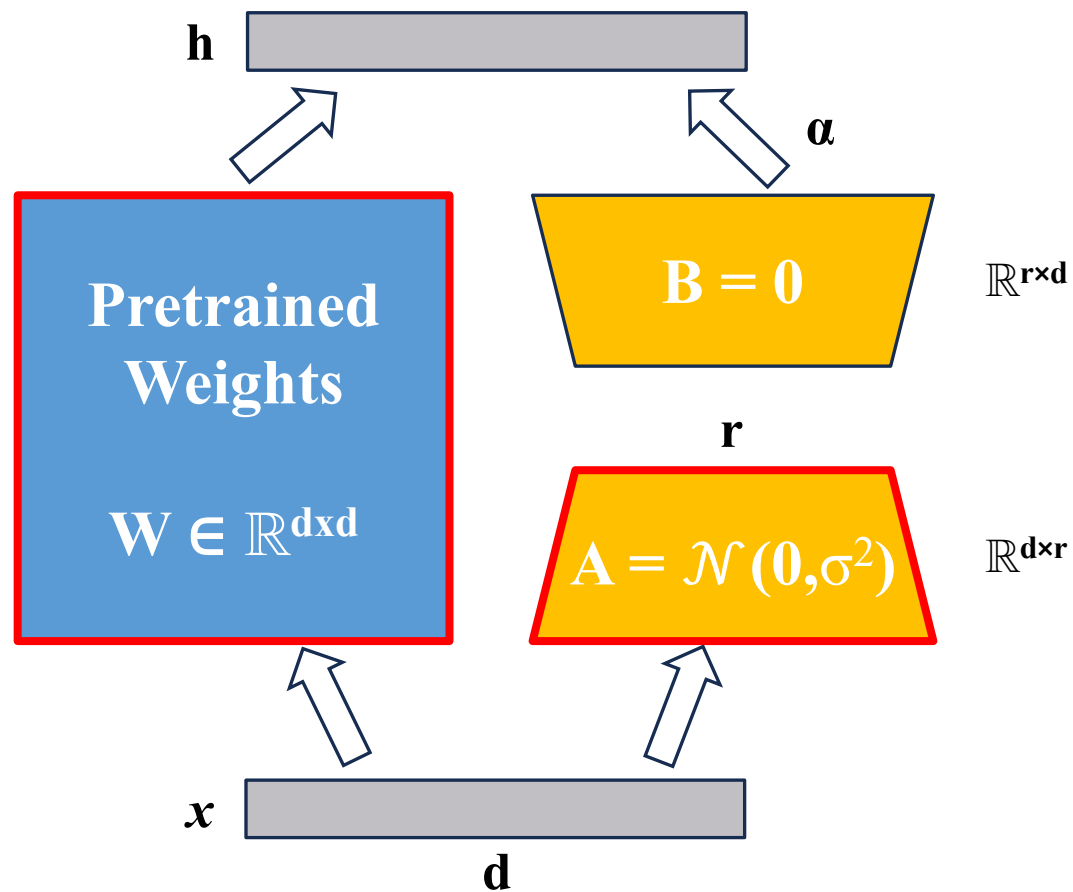
\checkmark : Linearly-independent vectors
rank = 3

rank 大小最多等於 column vectors
(或 row vectors) 的數量



Low-Rank 的部分在哪？(2)

LoRA



$d > r$ 的情況下 (通常 $d \gg r$)，**A 或 B** 的 rank 一定遠小於 **W** 的 rank
故為 low-rank 的由來



LoRA: Low-Rank Adaptation (w / pseudo code)

Pseudocode:

```
input_dim = 768 # the hidden size of the pre-trained model
output_dim = 768 # the output size of the layer
rank = 8 # The rank 'r' for the low-rank adaptation

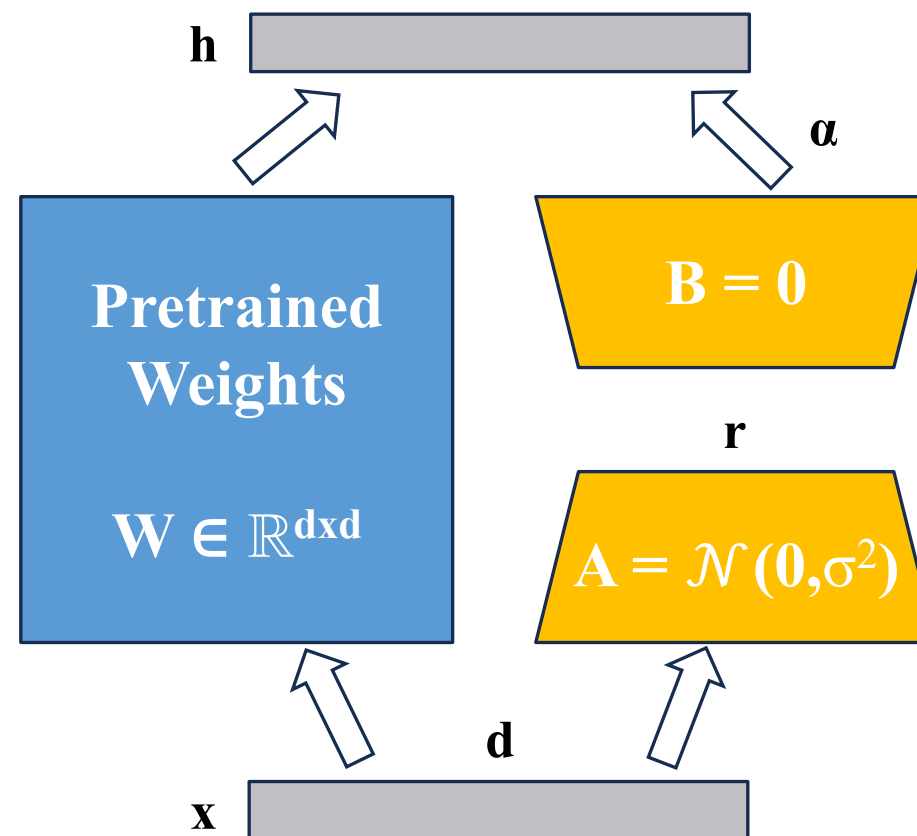
W = ... # from pretrained network with shape input_dim x
output_dim

W_A = nn.Parameter(torch.empty(input_dim, rank)) # LoRA weight A
W_B = nn.Parameter(torch.empty(rank, output_dim)) # LoRA weight B

# Initialization of LoRA weights
nn.init.kaiming_uniform_(W_A, a=math.sqrt(5))
nn.init.zeros_(W_B)

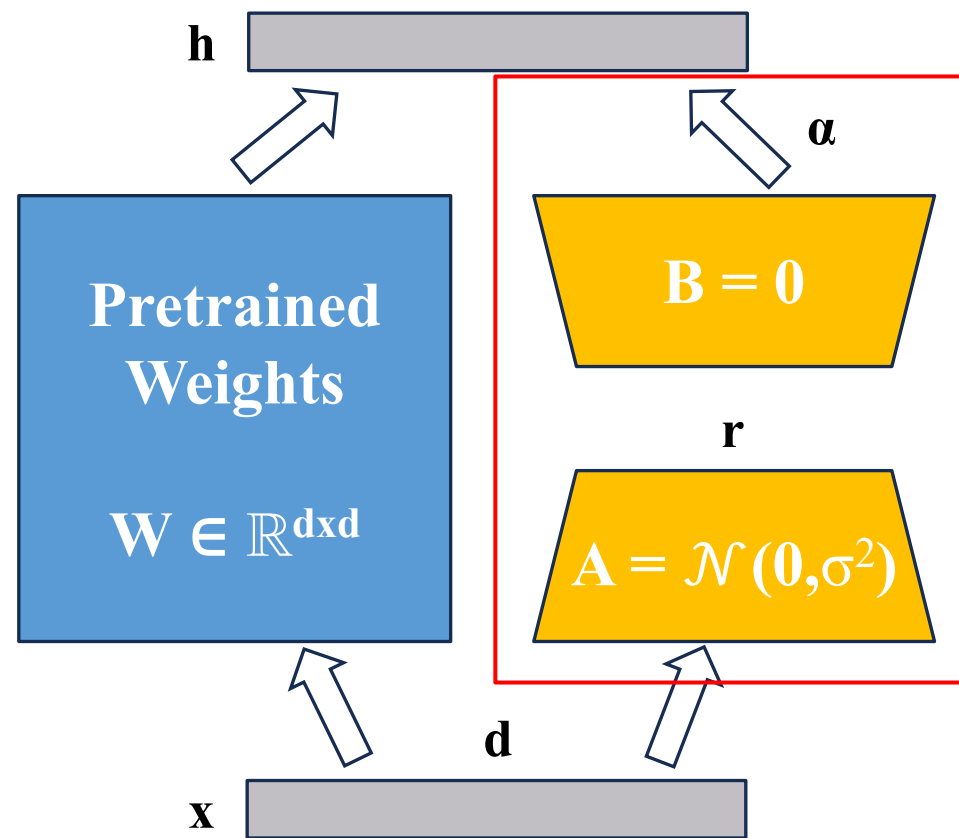
def regular_forward_matmul(x, W):
    h = x @ W
    return h

def lora_forward_matmul(x, W, W_A, W_B):
    h = x @ W # regular matrix multiplication
    h += x @ (W_A @ W_B) * alpha # use scaled LoRA weights
    return h
```



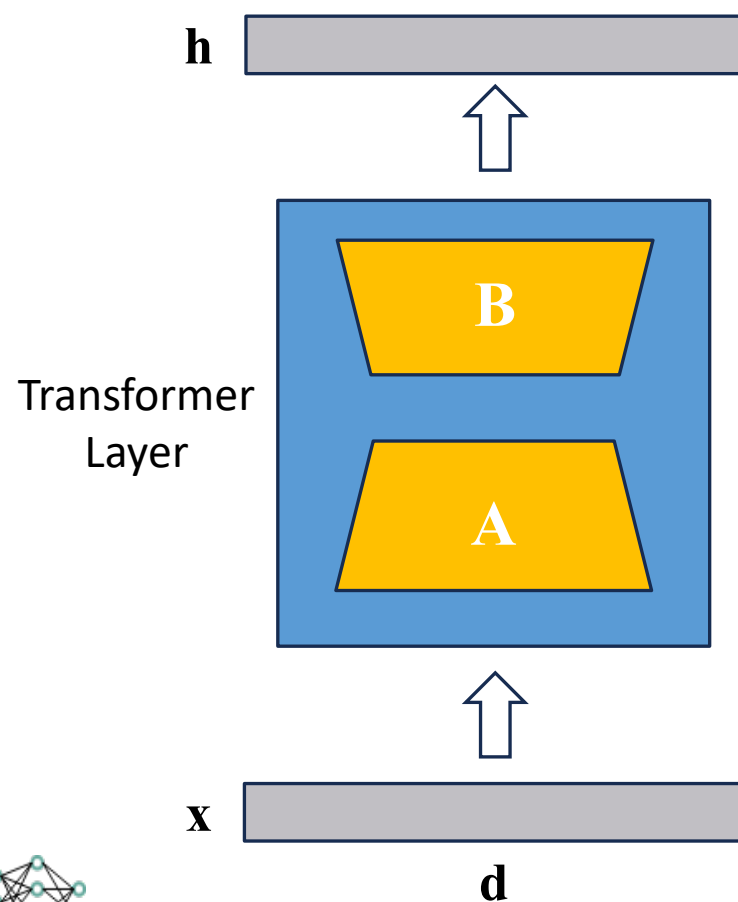
[注意事項] LoRA 的問題

- r 的數值大小需要手動調整
 - r 越小，訓練參數量越少
- 相較於原本沒有 LoRA 的模型，
LoRA 其實會讓 inference 速度變慢



可不可以不要外加？

Adapter: Parameter-Efficient Transfer Learning for NLP. ICML 2019



優點：模型設計更靈活

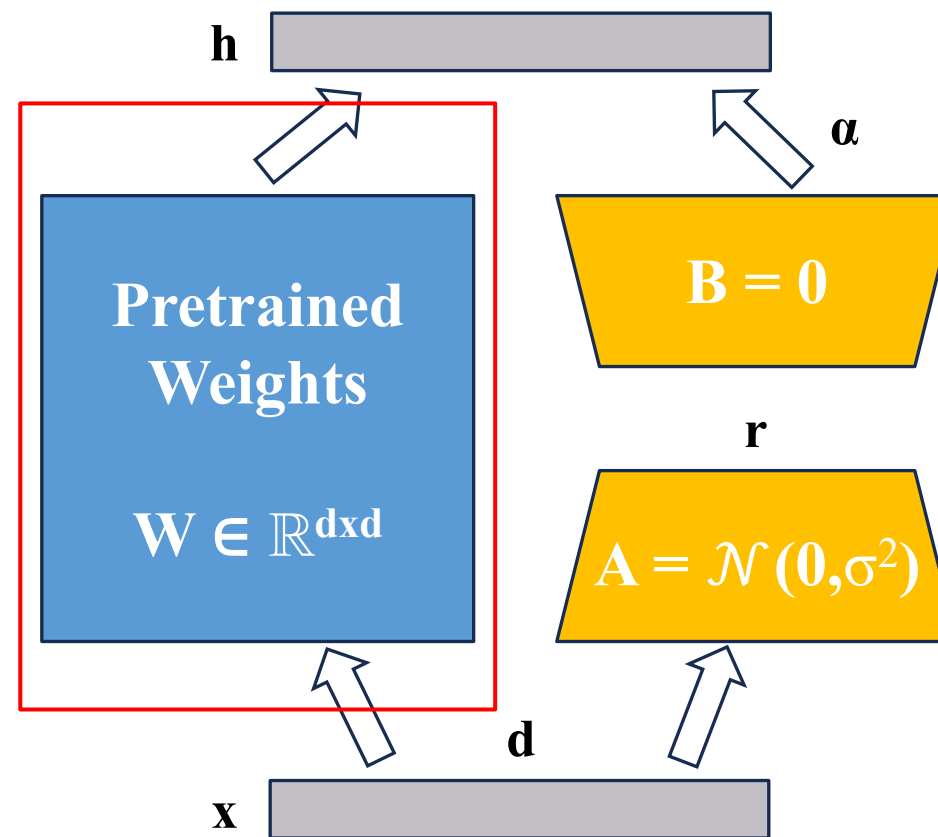
缺點：

- 直接變更了模型架構 (串聯)
 - -> 計算時間比 LoRA 長



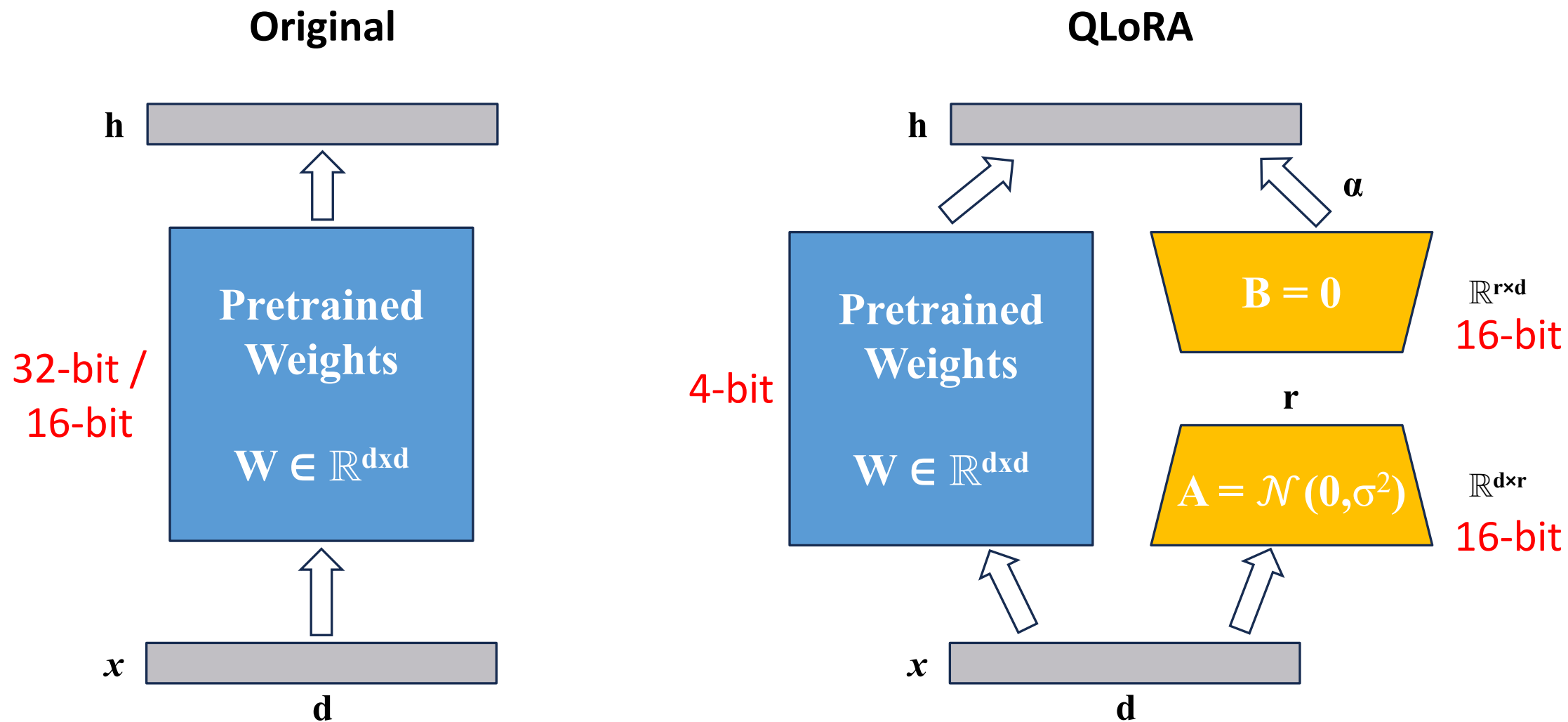
[注意事項] LoRA 的問題

- r 的數值大小需要手動調整
 - r 越小，訓練參數量越少
- 相較於原本沒有 LoRA 的模型，LoRA 其實會讓 inference 速度變慢
- 訓練和 inference 時，大模型的部分其實還是被載入記憶體...



QLoRA: Quantized LoRA

只有黃色部分會被訓練
 α 是超參數

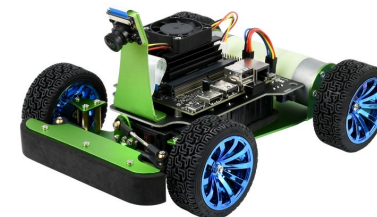
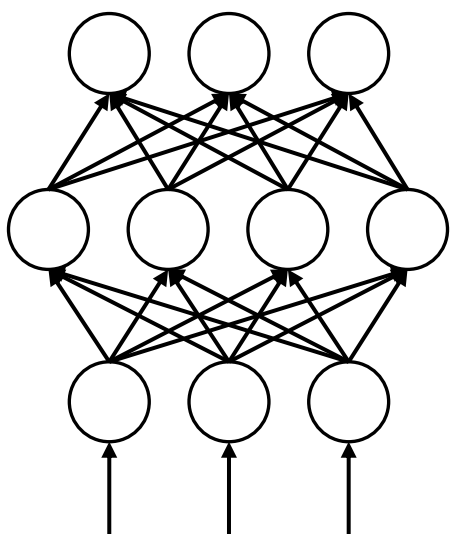


[注意事項] QLoRA

- 記憶體使用量：QLoRA < LoRA
- 模型效能：QLoRA <= LoRA



How about model compression?



PEFT vs. Model compression

	PEFT	Model Compression
目標	讓模型適應新的任務， 但模型大小不變	加速模型運算或模型儲存空間
相較於原始模型的 改變內容	插入少量可訓練參數	減少整體模型結構或權重
參數更新	只更新少量新參數	先減少整體模型結構後針對新的模型進行訓練
使用情境	需要模型學會新的任務時	手機、邊緣裝置



Thank you!

Instructor: 林英嘉

 yjlin@cgu.edu.tw

TA: 林君襄

 becky890926@gmail.com