LoRA: Low-Rank adaption of Large Language Models.

An important paradigm of NLP is:

- Large-Scale pretraining on general domain data.
- Adaptation to particular tasks and domains (Finetuning).

Problem: Deploying independent instances of fine tuned models is prohibitively expensive.

LoRA: • Freeze the pre-trained model weights and injects trainable rank decomposition matrices into each layer of the transformer architecture, greatly reducing the number of trainable parameters for downstream tasks.

Advantages:

- A pre-trained model can be shared and used to build many small LoRA modules for different tasks. We can freeze the shared model and efficiently switch tasks by replacing the matrices A and B, reducing the storage requirement and task-switching over-head significantly.
- We don't need gradients or maintain the optimizer states for most parameters. We only optimize the injected, much smaller low-rank matrices.

W_q: Query
W_k: Key
W_v: Value
W_o: Output

→ the weight matrices of LLM are typically full rank
→ Adapting to specific tasks ⇒ Low "intrinsic dimension"

$$W_0 + \Delta W = W_0 + B \cdot A$$

$B \in \mathbb{R}^{d \times r}$ $W_0 \in \mathbb{R}^{d \times k}$
$A \in \mathbb{R}^{r \times k}$

A: Random initialization
B: 0 init