Що таке LoRA

Як пам’ятаємо з трансформерів, у механізмі attention використовуються матриці Q (query), K (key), V (value) — вони відповідають за те, як модель уважно дивиться на різні частини вхідного тексту.

LoRA — це метод тюнінгу, який не змінює основні (pretrained) ваги моделі,
а вставляє додаткові маленькі low-rank матриці (A та B) в ключові лінійні шари, такі як q_proj, k_proj, v_proj, o_proj

У цих шарах ми фактично не змінюємо основну матрицю W, а додаємо LoRA-доповнення
W(x) + A @ B @ x — і навчаємо тільки A та B.

Ми зберігаємо усі pretrain-знання, адаптуємо до нової задачі + економимо час та ресурси :)

Trainable Матриці пояснення

Назва шару	**Що це таке**
`q_proj`	Query-проєкція — формує запит (що шукати у self-attention)
`k_proj`	Key-проєкція — що є в пам’яті (які ознаки/позиції зіставляються із query)
`v_proj`	Value-проєкція — значення, що зчитуються при увазі
`o_proj`	Output-проєкція — перетворення результату після self-attention
`up_proj`	Внутрішній шар feedforward-мережі — розширення розмірності (MLP)
`down_proj`	Вихідний шар feedforward-мережі — стискання назад до розміру ембедінгу
`gate_proj`	Gated activation — допомагає моделі керувати активацією
`lm_head`	Генератор токенів — лінійний шар, який перетворює hidden state у токен

Ціль	target_modules
Швидкий базовий fine-tune	`["q_proj", "v_proj"]`
Більше впливу, краща якість	`["q_proj", "k_proj", "v_proj", "o_proj"]`
Глибока модифікація (експерт)	`["q_proj", "k_proj", "v_proj", "o_proj", "up_proj", "down_proj", "gate_proj", "lm_head"]`

Параметризація LoRA

Параметр	Що це таке	Типове значення	Просте пояснення
`r`	Ранг low-rank матриць LoRA	8, 16, 32	Скільки “нових знань” LoRA може вивчити
`lora_alpha`	Масштаб LoRA-вкладу	= `r` або `2*r`	Наскільки сильно LoRA впливає на результат
`target_modules`	Які шари моделі тюнити через LoRA	`["q_proj", "v_proj"]`	Де саме вставити LoRA (attention, FFN тощо)
`lora_dropout`	Dropout перед LoRA-обчисленням	`0.05`, `0.1`	Регуляризація під час навчання (захист від оверфіту)

`bias`	Що робити з bias-термінами	`"none"`, `"lora_only"`	Чи оновлювати bias у Linear шарах
`modules_to_save`	Що ще (окрім LoRA) зберігати під час тюнінгу	`["lm_head"]` або `["decode_head"]`	Наприклад, класифікаційний head або генератор
`task_type`	Тип задачі (для PEFT/Trainer)	`"CAUSAL_LM"`	Вказує, яку архітектуру/режим тренування використовувати

Рекомендації

Рекомендую почати з базових налаштувань, а далі пробувати ускладнювати

```
lora_config = LoraConfig(
    r=8, // Мінімум параметрів для навчання, але достатньо щоб вже побачити ефект
    lora_alpha=16, // Регулює вплив LoRA-частини на загальний шар
    target_modules=["q_proj", "v_proj"],
    lora_dropout=0.05, // Запобігає оверфіту, особливо на малих датасетах
    bias="none",
    task_type="CAUSAL_LM"   // Для моделей на кшталт GPT — генерація тексту
)
```

https://arxiv.org/pdf/2106.09685

https://huggingface.co/docs/peft/main/conceptual_guides/lora#common-lora-parameters-in-peft