CS3512 Paper Sharing

大语言模型的高效参数微调技术

UNIkeEN 2024/04/02

1 BACKGROUND / 应用背景

- 具有一定专业性、包含私有数据的特定领域需要微调
- 全模型微调（FFT）更新所有参数
- 近年大语言模型参数不断增大，达超过百亿规模（GPT-3 175B）
- FFT 计算、存储、数据传输面临挑战

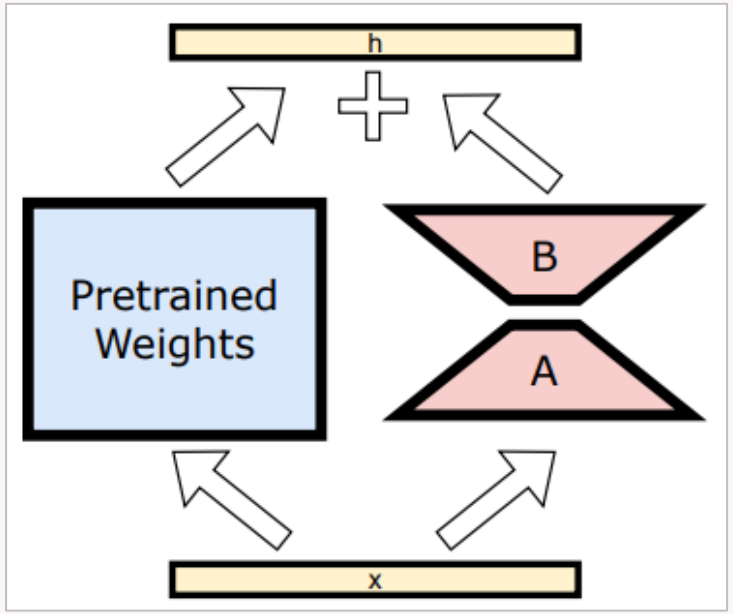
- 只微调部分参数、或为新任务学习外部模块——高效参数微调（PEFT）

低秩方法　基于适配器的方法　基于层前缀的方法　基于提示的方法

2 LoRA / 低秩微调

- **低秩矩阵注入与更新**
- 权重矩阵通常满秩，先前工作显示 LLM 具有较低“内在维度”
- 本文假设，微调过程中的参数更新也存在“低秩”
- $W = W_0 + \Delta W = W_0 + BA$
- $W_0 \in \mathbb{R}^{d\times k}$ $B \in \mathbb{R}^{d\times r}$ $A \in \mathbb{R}^{r\times k}$ $r \ll \min(d,k)$
- 原始权重固定，仅更新 A 和 B
- 随机高斯分布初始化 A，全零矩阵初始化

- 在 Transformer 层应用 LoRA



2 LoRA / 低秩微调

- **计算资源、效率优化与准确度（以 GPT-3 175B 为例）**
- 空间
  - VRAM 使用量减少到 FFT 的 1/3（1.2TB -> 350GB）
  - Checkpoint 约减少 10000 倍（350GB -> 35MB）
- 时间
  - 训练速度较 FFT 提升 25%
- 质量
  - 在保证较低参数量时，效果均达到最佳，大部分情况下优于 FFT

2 LoRA / 低秩微调

Model&Method	# Trainable Parameters	WikiSQL Acc. (%)	MNLI-m Acc. (%)	SAMSum R1/R2/RL
GPT-3 (FT)	175,255.8M	**73.8**	89.5	52.0/28.0/44.5
GPT-3 (BitFit)	14.2M	71.3	91.0	51.3/27.4/43.5
GPT-3 (PreEmbed)	3.2M	63.1	88.6	48.3/24.2/40.5
GPT-3 (PreLayer)	20.2M	70.1	89.5	50.8/27.3/43.5
GPT-3 ($\text{Adapter}^{\text{H}}$)	7.1M	71.9	89.8	53.0/28.9/44.8
GPT-3 ($\text{Adapter}^{\text{H}}$)	40.1M	73.2	**91.5**	53.2/29.0/45.1
GPT-3 (LoRA)	4.7M	73.4	**91.7**	**53.8/29.8/45.9**
GPT-3 (LoRA)	37.7M	**74.0**	**91.6**	53.4/29.2/45.1

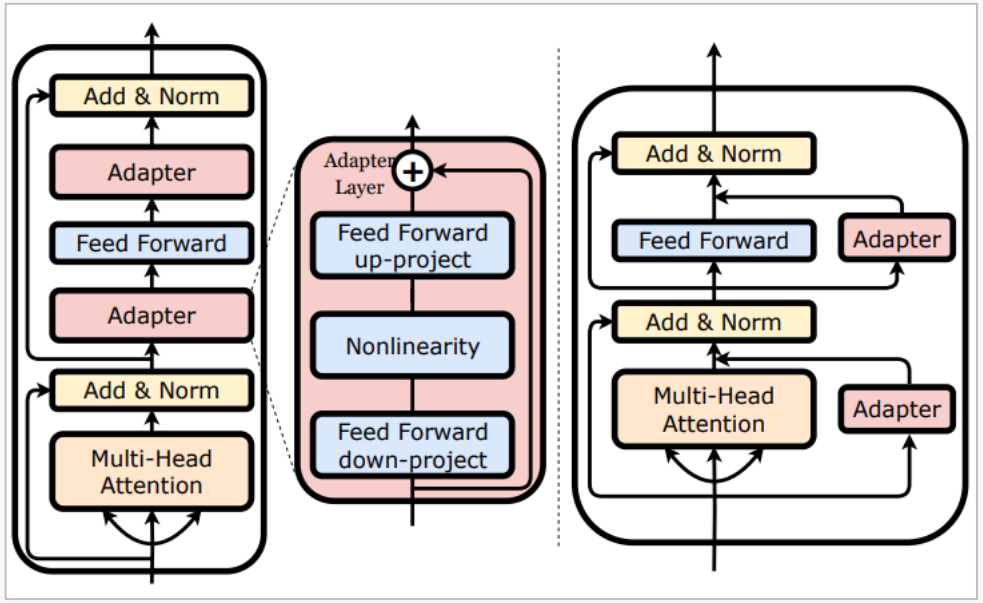
2 LoRA / 低秩微调

- **秩参数 r 的选择**
- 即使是非常小的 r , LoRA 的性能已经非常具有竞争力，继续增加 r , 验证准确率不再上升
- 支持“内在秩”假设，即低秩矩阵对于微调已然足够
- 补充实验

	Weight Type	$r=1$	$r=2$	$r=4$	$r=8$	$r=64$
WikiSQL(±0.5%)	W_q	68.8	69.6	70.5	70.4	70.0
	W_q, W_v	73.4	73.3	73.7	73.8	73.5
	W_q, W_k, W_v, W_o	74.1	73.7	74.0	74.0	73.9
MultiNLI (±0.1%)	W_q	90.7	90.9	91.1	90.7	90.7
	W_q, W_v	91.3	91.4	91.3	91.6	91.4
	W_q, W_k, W_v, W_o	91.2	91.7	91.7	91.5	91.4

3 LLM-Adapters / 基于适配器的方法

- 向基座预训练模型的已有子层添加新的模块并训练之



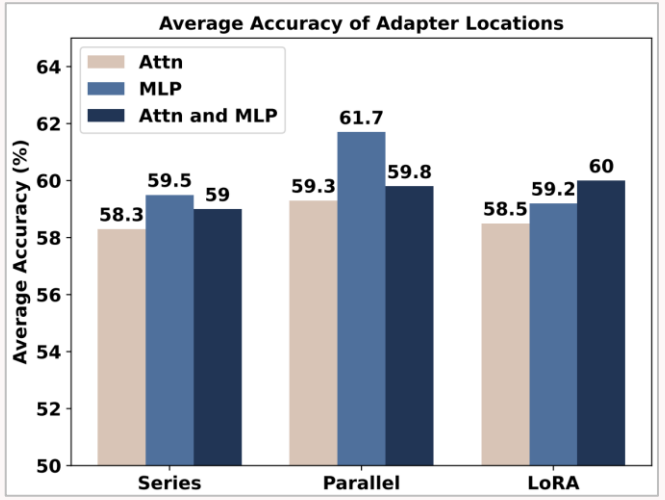
串行适配器

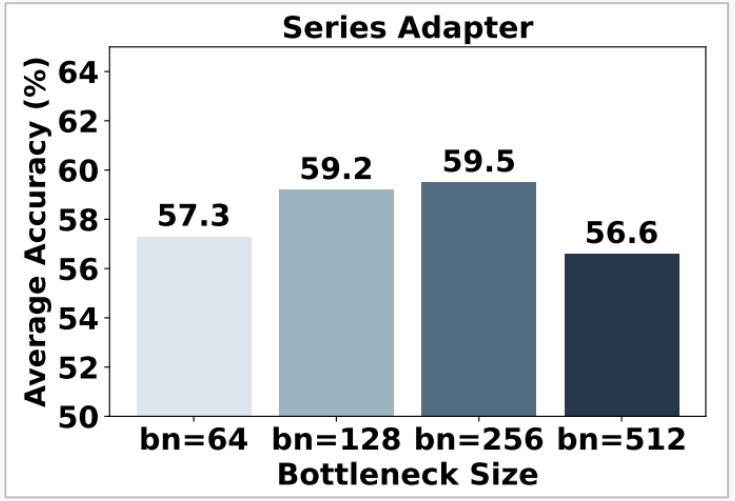
$H_o \leftarrow H_o + f(H_o W_{\text{down}})W_{\text{up}}$　　$f(\cdot)$是非线性激活函数

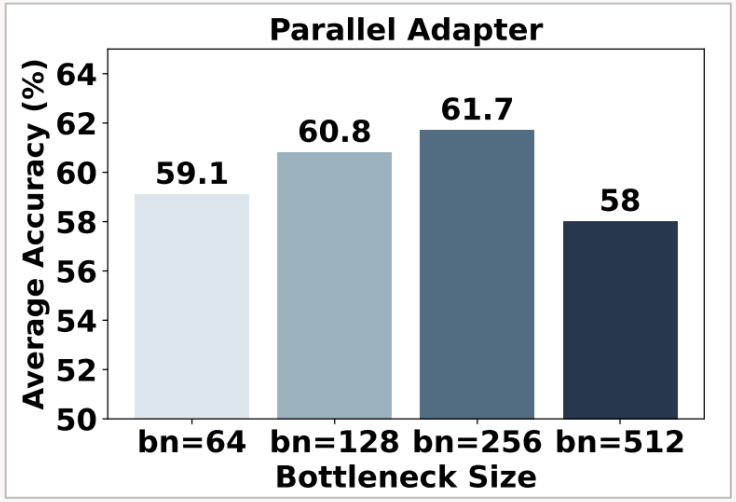
并行适配器

$H_o \leftarrow H_o + f(H_i W_{\text{down}})W_{\text{up}}$

3 LLM-Adapters / 基于适配器的方法







- 串行适配器最佳位置于 MLP 后，并行适配器、LoRA 最佳应用位置于多头注意力层+ MLP 层
- 实验显示，适配器方法和 LoRA 效果各有千秋，并行适配器通常优于串行适配器

4 Comparison and Thoughts / 比较与思考

- LoRA 只需存储低秩矩阵、没有修改模型结构
- 适配器引入推理延迟，结构可能不同导致任务切换效率低下
- 实践 case study——适配器略差于 LoRA
- 检索增强生成方法（RAG）
- 低秩矩阵与并行适配器概念易混淆

Thanks for Watching