5.总结

1.当前高效微调技术的简述

之前对一些常见的高效微调技术进行了背景介绍及技术原理剖析，下面对每一种高效微调技术的特点进行简要的总结。

2.BitFit

对微调机制的一种积极探索，也很简单，**通过仅调整bias效果就能有不错的效果**，但没有具体阐述原理，就是通过猜测加实验得到的结果。同时，作者提出一个观点：微调的过程不是让模型适应另外的数据分布，而是让模型更好的应用出本身的表征能力。

特点：

- 训练参数量极小（约0.1%）。
- 在大部分任务上效果会差于LoRA、Adapter等方法。

3.Prefix Tuning

在每一个Transformer层都带上一些virtual token作为前缀，以适应不同的任务。

特点：

- 前缀Token会占用序列长度，有一定的额外计算开销。
- Prefix Tuning的线性插值是比较复杂的。

> **Note**
>
> - 对于每一层 Transformer：
>   - 在计算 Attention 时，把 **Key 矩阵和 Value 矩阵前面拼接一段可学习的前缀向量**；
>   - 这些前缀向量是**每层独立的**（不同层可以不同）；
> - 输入层也可以加，但重点在 **中间层的 K/V 前缀。**

4.Prompt Tuning

该方法可以看着是Prefix Tuning的简化版本，针对不同的任务，仅在输入层引入virtual token形式的软提示（soft prompt）。

特点：

- 相对于Prefix Tuning，参与训练的参数量和改变的参数量更小，更节省显存。
- 对一些简单的NLU 任务还不错，但对硬序列标记任务（即序列标注）表现欠佳。

> **Note**
>
> **只在输入层开头加一串可学习的 soft prompt 向量**，其他什么都不改

5.P-Tuning

将Prompt转换为可以学习的Embedding层，并用MLP+LSTM的方式来对Prompt Embedding进行一层处理。相比Prefix Tuning，仅在输入层加入的可微的virtual token；另外，virtual token的位置也不一定是前缀，插入的位置是可选的。

特点：

- 引入一个prompt encoder（由一个双向的LSTM+两层MLP组成）来建模virtual token的相互依赖会收敛更快，效果更好。

> **Note**
>
> - 仍然只在**输入层**加虚拟 token（和 Prompt Tuning 一样，不进每层）；
> - 但这些虚拟 token **不是直接可训练的向量**，而是由一个 **prompt encoder** 生成的；
> - 这个 encoder 通常是：**双向 LSTM + 两层 MLP**；
> - 虚拟 token 的位置**不一定是前缀**！可以插在中间（比如 `[x1, P1, x2, P2, x3]`）。

6.P-Tuning v2

该方法在每一个Transformer层都加入了prompt token作为输入，引入多任务学习，针对不同任务采用不同的提示长度。并且回归传统的分类标签范式，而不是映射器。

特点：

- 解决了Prompt Tuning无法在小模型上有效提升的问题。
- 移除了对模型效果改进较小的重参数化的编码器（如：Prefix Tuning中的MLP、P-Tuning中的LSTM）。
- 对于一些复杂的硬序列标记任务（即序列标注）取得了不错的效果。

7.Adapter Tuning

该方法设计了Adapter结构，并将其嵌入Transformer的结构里面，针对每一个Transformer层，增加了两个Adapter结构，在训练时，固定住原来预训练模型的参数不变，只对新增的Adapter结构和Layer Norm 层进行微调。

特点：

- 通过在Transformer层中嵌入Adapter结构，在推理时会额外增加推理时长。

8.AdapterFusion

一种融合多任务信息的Adapter的变体，在 Adapter 的基础上进行优化，通过将学习过程分为两阶段来提升下游任务表现。

9.AdapterDrop

该方法在不影响任务性能的情况下，对Adapter动态高效的移除，尽可能的减少模型的参数量，提高模型在反向传播（训练）和正向传播（推理）时的效率。

特点：

- 通过从较低的 Transformer 层删除可变数量的Adaper来提升推理速度。 当对多个任务执行推理时，动态地减少了运行时的计算开销，并在很大程度上保持了任务性能。

10.LoRA

该方法通过低秩分解来模拟参数的改变量，从而以极小的参数量来实现大模型的间接训练。

特点：

- 将BA加到W上可以消除推理延迟。
- 可以通过可插拔的形式切换到不同的任务。
- 设计的比较好，简单且效果好。

11.AdaLoRA

对LoRA的一种改进，它根据重要性评分动态分配参数预算给权重矩阵，将关键的增量矩阵分配高秩以捕捉更精细和任务特定的信息，而将较不重要的矩阵的秩降低，以防止过拟合并节省计算预算。

12.QLoRA

使用一种新颖的高精度技术将预训练模型量化为 4 bit，然后添加一小组可学习的低秩适配器权重，这些权重通过量化权重的反向传播梯度进行微调。

特点：

- 使用 QLoRA 微调模型，可以显著降低对于显存的要求。同时，模型训练的速度会慢于LoRA。

13.MAM Adapter

一种在 Adapter、Prefix Tuning 和 LoRA 之间建立联系的统一方法。最终的模型 MAM Adapter 是用于 FFN 的并行 Adapter 和 软提示的组合。

特点：

- 整体上来说，最终的模型MAM Adapter效果会优于单个高效微调方法。

14.UniPELT

一种将不同的PELT方法LoRA、Prefix Tuning和Adapter作为子模块，并通过门控机制学习激活最适合当前数据或任务的方法。

特点：

- 相对于LoRA，BitFit，Prefix-tuning，训练的参数量更大；同时，推理更耗时；并且，输入会占用额外的序列长度。
- 多种 PELT 方法的混合涉及PLM 的不同部分对模型有效性和鲁棒性都有好处。

15.多种不同的高效微调方法对比

总的来说，像P-Tuning v2、LoRA等都是综合评估很不错的高效微调技术。如果显存资源有限可以考虑QLoRA；如果只是解决一些简单任务场景，可以考虑P-Tuning、Prompt Tuning也行。

下表从参数高效方法类型、是否存储高效和内存高效、以及在减少反向传播成本和推理开销的计算高效五个维度比较了参数高效微调方法。

Method	Type	Storage	Memory	Backprop	Inference overhead
Adapters (Houlsby et al., 2019)	A	yes	yes	no	Extra FFN
AdaMix (Wang et al., 2022)	A	yes	yes	no	Extra FFN
SparseAdapter (He et al., 2022b)	AS	yes	yes	no	Extra FFN
Cross-Attn tuning (Gheini et al., 2021)	S	yes	yes	no	No overhead
BitFit (Ben-Zaken et al., 2021)	S	yes	yes	no	No overhead
DiffPruning (Guo et al., 2020)	S	yes	no	no	No overhead
Fish-Mask (Sung et al., 2021)	S	yes	maybe[5]	no	No overhead
LT-SFT (Ansell et al., 2022)	S	yes	maybe[5]	no	No overhead
Prompt Tuning (Lester et al., 2021)	A	yes	yes	no	Extra input
Prefix-Tuning (Li and Liang, 2021)	A	yes	yes	no	Extra input
Spot (Vu et al., 2021)	A	yes	yes	no	Extra input
IPT (Qin et al., 2021)	A	yes	yes	no	Extra FFN and input
MAM Adapter (He et al., 2022a)	A	yes	yes	no	Extra FFN and input
Parallel Adapter (He et al., 2022a)	A	yes	yes	no	Extra FFN
Intrinsinc SAID (Aghajanyan et al., 2020)	R	no	no	no	No overhead
LoRa (Hu et al., 2021)	R	yes	yes	no	No overhead
UniPELT (Mao et al., 2021)	AR	yes	yes	no	Extra FFN and input
Compacter (Karimi Mahabadi et al., 2021)	AR	yes	yes	no	Extra FFN
PHM Adapter (Karimi Mahabadi et al., 2021)	AR	yes	yes	no	Extra FFN
KronA (Edalati et al., 2022)	R	yes	yes	no	No overhead
KronA$^{B}_{res}$ (Edalati et al., 2022)	AR	yes	yes	no	Extra linear layer
(IA)^3 (Liu et al., 2022)	A	yes	yes	no	Extra gating
Attention Fusion (Cao et al., 2022)	A	yes	yes	yes	Extra decoder
LeTS (Fu et al., 2021)	A	yes	yes	yes	Extra FFN
Ladder Side-Tuning (Sung et al., 2022)	A	yes	yes	yes	Extra decoder
FAR (Vucetic et al., 2022)	S	yes	maybe[6]	no	No overhead
S4-model (Chen et al., 2023)	ARS	yes	yes	no	Extra FFN and input

Table 1: Comparing PEFT methods across storage efficiency, memory efficiency, and computational efficiency in terms of reducing backpropagation costs and having inference overhead. Method types: A – additive, S – selective, R – reparametrization-based.

下表展示了各种参数高效方法的参与训练的参数量、最终模型与原始模型的改变参数（delta值）以及论文中参与评估的模型的范围（<1B、<20B、>20B）。

Method	% Trainable parameters	% Changed parameters	Evaluated on <1B	<20B	>20B
Adapters (Houlsby et al., 2019)	0.1 - 6	0.1 - 6	yes	yes	yes
AdaMix (Wang et al., 2022)	0.1 - 0.2	0.1 - 0.2	yes	no	no
SparseAdapter (He et al., 2022b)	2.0	2.0	yes	no	no
BitFit (Ben-Zaken et al., 2021)	0.05 - 0.1	0.05 - 0.1	yes	yes	yes
DiffPruning (Guo et al., 2020)	200	**0.5**	yes	no	no
Fish-Mask (Sung et al., 2021)	0.01 - 0.5	0.01 - 0.5	yes	yes	no
Prompt Tuning (Lester et al., 2021)	0.1	0.1	yes	yes	yes
Prefix-Tuning (Li and Liang, 2021)	0.1 - 4.0	0.1 - 4.0	yes	yes	yes
IPT (Qin et al., 2021)	56.0	56.0	yes	no	no
MAM Adapter (He et al., 2022a)	0.5	0.5	yes	no	no
Parallel Adapter (He et al., 2022a)	0.5	0.5	yes	no	no
Intrinsinc SAID (Aghajanyan et al., 2020)	0.001 - 0.1	**∼0.1 or 100**	yes	yes	no
LoRa (Hu et al., 2021)	0.01 - 0.5	**∼0.5 or ∼30**	yes	yes	yes
UniPELT (Mao et al., 2021)	1.0	1.0	yes	no	no
Compacter (Karimi Mahabadi et al., 2021)	0.05-0.07	**∼0.07 or ∼0.1**	yes	yes	no
PHM Adapter (Karimi Mahabadi et al., 2021)	0.2	**∼0.2 or ∼1.0**	yes	no	no
KronA (Edalati et al., 2022)	0.07	**∼0.07 or ∼30.0**	yes	no	no
KronA$^{B}_{res}$ (Edalati et al., 2022)	0.07	**∼0.07 or ∼1.0**	yes	no	no
(IA)^3 (Liu et al., 2022)	0.02	0.02	no	yes	no
Ladder Side-Tuning(Sung et al., 2022)	7.5	7.5	yes	yes	no
FAR (Vucetic et al., 2022)	6.6-26.4	6.6-26.4	yes	no	no
S4-model (Chen et al., 2023)	0.5	**more than 0.5**	yes	yes	no

Table 2: What model sizes PEFTmethods have been evaluated on and their typical amount of trainable parameters used in the papers. By trainable parameter count we specifically mean the number parameters that are updated by a gradient optimization algorithm, not the delta between the original and final model which we denote "changed parameters". For reprametrization-based method we report parameters before and after reparametrization. Estimating updated parameter count for S4 is complicated as the model uses different methods at different layers. We report the range at which the methods were evaluated in the published literature.

从表中可以看到，Prompt Tuning、Prefix Tuning、LoRA等少部分微调技术针对不同参数规模的模型进行过评估，同时，这几种方式也是目前应用比较多的高效微调方法。

16.总结

本文针对之前介绍的几种参数高效微调方法进行了简单的概述，主要有如下几类：

- 增加额外参数，如：Prefix Tuning、Prompt Tuning、Adapter Tuning及其变体。
- 选取一部分参数更新，如：BitFit。
- 引入重参数化，如：LoRA、AdaLoRA、QLoRA。
- 混合高效微调，如：MAM Adapter、UniPELT。

并比较了不同的高效微调方法之间的差异；同时，还指出当前大多数高效微调方法存在的一些问题并给出了最佳实践。