Structured Pruning Learns Compact and Accurate Models——结构化修剪学习紧凑精确的模型

摘要

- 目前模型压缩两种方式：
  - 修剪(pruning)：从预先训练好的模型中去除权重
  - 蒸馏(distillation)：训练一个较小的紧凑模型来匹配一个较大的模型
  - **存在问题**：剪枝方法可以显著减小模型尺寸，但很难达到蒸馏方法那样的推理速度；蒸馏方法需要大量的未标注数据来进行训练
- 模型——CoFi（压缩transformer模型）
  - 提供了高度并行的子网络，实现了与知识蒸馏相当的高精度、低延迟
  - 不需要依赖大量的 $Unlabeled$ 数据
  - **关键点**：
    - 联合修剪粗粒度（层级）和细粒度（head、隐藏单元）模块，用不同粒度的掩码控制每个参数的修剪策略
    - 分层蒸馏策略：在优化过程中，将知识从未修剪的模型转移到修剪后的子网络模型
- 效果

  在GLUE和SQuAD数据集上进行的实验并比较分析，CoFi产生的模型具有超过10倍的加速率，精度下降很小，与以前的修剪和蒸馏方法相比，具有更高的效率和有效性

1. Introduction

- 一些最先进的剪枝和蒸馏方法的比较：
  - 对大型预训练语言模型BERT进行剪枝和蒸馏操作：U 代表$Unlabeled$；T 代表 $Task-specific$
  - MNLI数据集：MNLI(The Multi-Genre Natural Language Inference Corpus, 多类型自然语言推理数据库)，自然语言推断任务，是通过众包方式对句子对进行文本蕴含标注的集合。给定前提（premise）语句和假设（hypothesis）语句，任务是预测前提语句是否包含假设（蕴含, entailment），与假设矛盾（矛盾，contradiction）或者两者都不（中立，neutral）。前提语句是从数十种不同来源收集的，包括转录的语音，小说和政府报告。

	U	T	↗	Params	MNLI
$\text{BERT}_{\text{base}}$ (teacher)	✗	✗	1.0×	85M	84.8
Distillation					
DistillBERT_6	✓	✗	2.0×	43M	82.2
TinyBERT_6	✓	✓	2.0×	43M	84.0
$\text{MobileBERT}^{\ddagger}$	✓	✗	2.3×	20M	83.9
DynaBERT	✗	✓	6.3×	11M	76.3
$\text{AutoTinyBERT}^{\ddagger}$	✓	✓	9.1×	3.3M	78.2
TinyBERT_4	✓	✓	11.4×	4.7M	78.8
Pruning					
Movement Pruning	✗	✓	1.0×	9M	81.2
Block Pruning	✗	✓	2.7×	25M	83.7
CoFi Pruning (ours)	✗	✓	2.7×	26M	84.9
CoFi Pruning (ours)	✗	✓	12.1×	4.4M	80.6

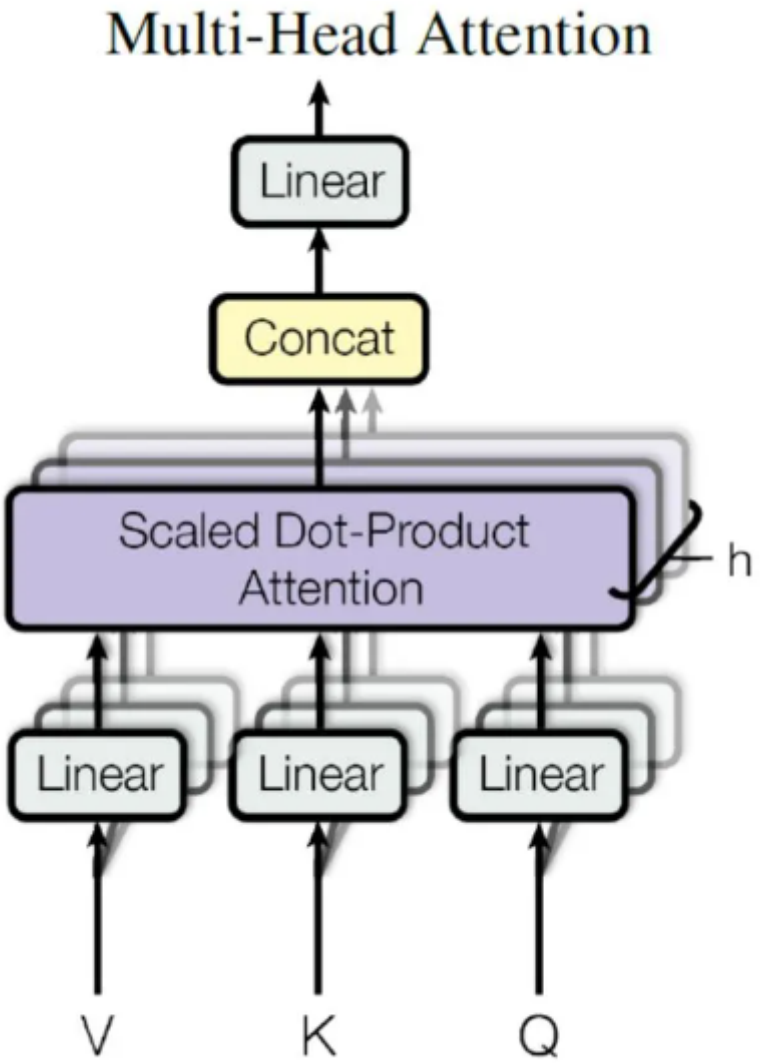
- 剪枝
  - 剪枝方法主要关注于在更大的预训练模型中搜索精确的子网络——网络结构搜索；
  - 相关研究：如何从结构上修剪Transformer网络（结构化修剪）
    - 删除整个层 => 修剪head => 修剪中间维度 => 修剪权重矩阵中的块
    - 趋势：趋向于细粒度单元，以得到更加灵活的结构
    - 局限：修建后的模型很少实现较大的加速（2 - 3倍）
- 知识蒸馏
  - 通常先指定一个固定的模型架构，并在未标记的语料库上执行一般蒸馏步骤，然后对特定于任务的数据进一步微调或者蒸馏
  - 蒸馏得到的模型推理速度快，表现效果好；需要大量未标记数据进行训练，训练速度慢
- CoFi——粗粒度和细粒度剪枝
  - 特定于任务的结构化剪枝
  - 结构化剪枝可以实现高度紧凑的子网络——方便之后进行知识蒸馏
  - 减少计算
  - **创新点**
    - 联合修剪粗粒度单位（self-attention、前馈层）和细粒度单位（head、隐藏维度）
    - 通过不同粒度的多个掩码来控制每个参数的修剪决策
    - 使修剪更加灵活，得到更加灵活的模型
- 分层蒸馏——将知识从未修剪的模型转移到修剪的模型

2. 背景

2.1 Transformers

- MHA

$$\text{MHA}(X) = \sum_{i=1}^{N_h} \text{Att}(W_Q^{(i)}, W_K^{(i)}, W_V^{(i)}, W_O^{(i)}, X),$$

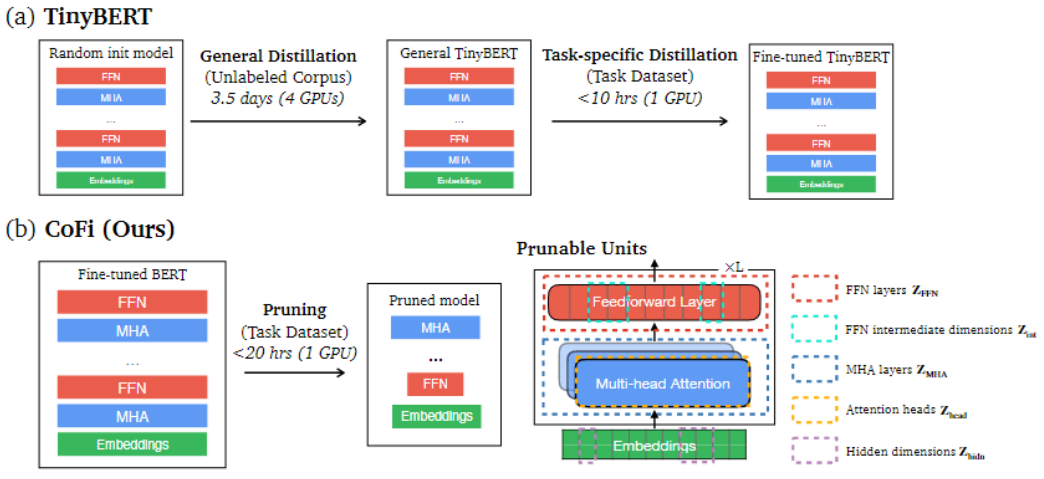


Multi-Head Attention

- FFN——前馈神经网络（输入和输出维度相同）

2.2 Distillation

- 知识蒸馏：一种模型压缩方法，将教师模型的知识转移到学生模型
- 分为一般蒸馏和基于特定任务的蒸馏，两者分别利用未标注数据和基于特定任务的数据，两者结合可以获得更优的模型表现
- 在未标记语料库上对学生网络进行一般蒸馏或预训练对于保持性能至关重要，同时计算成本很高



2.3 Pruning

- **Layer pruning**

  去除整个MHA层和FFN层，大约有一半的层可以被修剪，但精度大幅度下降，加速率提升两倍左右

- **Head pruning**

  引入一个mask，z_{head} 将自注意力层中不重要的head修剪

$$\mathrm{MHA}(X) = \sum_{i=1}^{N_h} \mathbf{z}_{\mathrm{head}}^{(i)} \mathrm{Att}(W_Q^{(i)}, W_K^{(i)}, W_V^{(i)}, W_O^{(i)}, X).$$

- **FFN pruning**

  修剪FFN中不重要的隐藏层

$$\mathrm{FFN}(X) = \mathrm{gelu}(XW_U) \cdot \mathrm{diag}(\mathbf{z}_{\mathrm{int}}) \cdot W_D.$$

- **块和非结构化剪枝**

  很难在硬件上进行加速

- **与蒸馏相结合**

  目前尚不清楚如何在训练期间应用分层蒸馏策略，因为修剪后的学生模型的架构演变。

3. 方法

结构化修剪方法——CoFi，联合修剪粗粒度和细粒度单元，并以分层蒸馏为目标，将知识从未修剪的模型转移到修剪的模型

3.1 粗粒度和细粒度修剪

- 为self-Attention和全连接层引入两个额外的掩码z_{MHA} 和 z_{FFN}；使用这些掩码显示地修剪整个层，而不是修剪一个MHA层中的所有头部或者FFN层中的全部中间维度
- 定义一组掩码 z_{hidden} 修剪$MHA(X)$ 和$FFN(X)$ 的输出维度（隐藏维度），跨层共享，隐藏表示中的每个维度都通过残差连接连接到下一层中的相同维度。这些掩码变量被应用于模型中的所有权重矩阵。
- CoFi与以前的修剪方法的不同之处在于，多个掩码变量联合控制一个单个参数的修剪决策，引入了5个mask

  z_{MHA} 、 z_{FFN}、z_{head}、z_{int}、 z_{hidden}

$$\begin{aligned}\mathrm{MHA}(X) = z_{\mathrm{MHA}} \cdot \sum_{i=1}^{N_h} (\mathbf{z}_{\mathrm{head}}^{(i)} \cdot \\ \mathrm{Att}(W_Q^{(i)}, W_K^{(i)}, W_V^{(i)}, W_O^{(i)}, X)), \\ \mathrm{FFN}(X) = z_{\mathrm{FFN}} \cdot \mathrm{gelu}(XW_U) \cdot \mathrm{diag}(\mathbf{z}_{\mathrm{int}}) \cdot W_D.\end{aligned}$$

  剪枝目标：

$$\hat{s} = \frac{1}{M} \cdot 4 \cdot d_h \cdot \sum_i^L \sum_j^{N_h} \sum_k^d \mathbf{z}_{\text{MHA}}^{(i)} \cdot \mathbf{z}_{\text{head}}^{(i,j)} \cdot \mathbf{z}_{\text{hidden}}^{(k)} + \frac{1}{M} \cdot 2 \cdot \sum_i^L \sum_j^{d_f} \sum_k^d \mathbf{z}_{\text{FFN}}^{(i)} \cdot \mathbf{z}_{\text{int}}^{(i,j)} \cdot \mathbf{z}_{\text{hidden}}^{(k)},$$

3.2 Distillation to Pruned Models——修剪模型的蒸馏

- 分层蒸馏——动态搜索教师模型与学生模型之间的层映射
  - 预测层

$$\mathcal{L}_{\text{pred}} = D_{\text{KL}}(\mathbf{p}_s \parallel \mathbf{p}_t).$$

  - 中间层

$$\mathcal{L}_{\text{layer}} = \sum_{i \in \mathcal{T}} \text{MSE}(W_{\text{layer}} \mathbf{H}_s^{m(i)}, \mathbf{H}_t^i),$$

  - 动态搜索与学生模型隐藏层最相似的教师模型的隐藏层

$$m(i) = \underset{j:\mathbf{z}_{\text{FFN}}^{(j)}>0}{\arg\min} \text{MSE}(W_{\text{layer}} \mathbf{H}_s^j, \mathbf{H}_t^i).$$

  - 知识蒸馏损失函数

$$\mathcal{L}_{\text{distil}} = \lambda \mathcal{L}_{\text{pred}} + (1 - \lambda) \mathcal{L}_{\text{layer}},$$

4 实验

4.1 设置

- **数据集**
  - **GLUE**：包含九项任务涉及到自然语言推断、文本蕴含、情感分析、语义相似等多个任务。
    - SST-2：SST-2(The Stanford Sentiment Treebank，斯坦福情感树库)，单句子分类任务，包含电影评论中的句子和它们情感的人类注释。这项任务是给定句子的情感，类别分为两类正面情感（positive，样本标签对应为1）和负面情感（negative，样本标签对应为0），并且只用句子级别的标签。也就是，本任务也是一个二分类任务，针对句子级别，分为正面和负面情感。——情感分类
    - QNLI：QNLI(Qusetion-answering NLI，问答自然语言推断)，自然语言推断任务。QNLI是从另一个数据集The Stanford Question Answering Dataset(斯坦福问答数据集，SQuAD 1.0)[3]转换而来的。SQuAD 1.0是有一个问题-段落对组成的问答数据集，其中段落来自维基百科，段落中的一个句子包含问题的答案。这里可以看到有个要素，来自维基百科的段落，问题，段落中的一个句子包含问题的答案。通过将问题和上下文（即维基百科段落）中的每一句话进行组合，并过滤掉词汇重叠比较低的句子对就得到了QNLI中的句子对。相比原始SQuAD任务，消除了模型选择准确答案的要求；也消除了简化的假设，即答案适中在输入中并且词汇重叠是可靠的提示。——问答系统

- MNLI：MNLI(The Multi-Genre Natural Language Inference Corpus, 多类型自然语言推理数据库)，自然语言推断任务，是通过众包方式对句子对进行文本蕴含标注的集合。给定前提（premise）语句和假设（hypothesis）语句，任务是预测前提语句是否包含假设（蕴含, entailment），与假设矛盾（矛盾，contradiction）或者两者都不（中立，neutral）。前提语句是从数十种不同来源收集的，包括转录的语音，小说和政府报告。
- QQP：QQP(The Quora Question Pairs, Quora问题对数集)，相似性和释义任务，是社区问答网站Quora中问题对的集合。任务是确定一对问题在语义上是否等效。
- CoLA：CoLA(The Corpus of Linguistic Acceptability，语言可接受性语料库)，单句子分类任务，语料来自语言理论的书籍和期刊，每个句子被标注为是否合乎语法的单词序列。本任务是一个二分类任务，标签共两个，分别是0和1，其中0表示不合乎语法，1表示合乎语法。
- RTE：RTE(The Recognizing Textual Entailment datasets，识别文本蕴含数据集)，自然语言推断任务，它是将一系列的年度文本蕴含挑战赛的数据集进行整合合并而来的，包含RTE1[4]，RTE2，RTE3[5]，RTE5等，这些数据样本都从新闻和维基百科构建而来。将这些所有数据转换为二分类，对于三分类的数据，为了保持一致性，将中立（neutral）和矛盾（contradiction）转换为不蕴含（not entailment）。
- STS-B：STSB(The Semantic Textual Similarity Benchmark，语义文本相似性基准测试)，相似性和释义任务，是从新闻标题、视频标题、图像标题以及自然语言推断数据中提取的句子对的集合，每对都是由人类注释的，其相似性评分为0-5(大于等于0且小于等于5的浮点数，原始paper里写的是1-5，可能是作者失误）。任务就是预测这些相似性得分，本质上是一个回归问题，但是依然可以用分类的方法，可以归类为句子对的文本五分类任务。
- MRPC：MRPC(The Microsoft Research Paraphrase Corpus，微软研究院释义语料库)，相似性和释义任务，是从在线新闻源中自动抽取句子对语料库，并人工注释句子对中的句子是否在语义上等效。类别并不平衡，其中68%的正样本，所以遵循常规的做法，报告准确率（accuracy）和F1值。

- **SQuAD**

  SQuAD是Stanford Question Answering Dataset 的首字母缩写。这是一个阅读理解数据集，由众包工作者在一组维基百科文章上提出的问题组成，其中每个问题的答案都是相应文章中的一段文本，某些问题可能无法回答。

Task	Train Size	Metric
SST-2	67k	accuracy
QNLI	105k	accuracy
MNLI	393k	accuracy
QQP	364k	accuracy
CoLA	8.5k	Matthews corr.
RTE	2.5k	accuracy
STS-B	7k	Spearman corr.
MRPC	3.7k	accuracy
SQuAD	88k	F1

- **实验设置**
  - 对于比较大的数据集：MNLI、QNLI、SST-2和QQP以及SQuAD

    20epoch训练（1epoch以蒸馏为目标 + 2epoch达到剪枝稀疏率）

- 对于较小的GLUE数据集

  100epoch训练（4epoch以蒸馏为目标 + 20epoch达到剪枝稀疏率）

- 即使已经达到了目标稀疏率，在之后的epoch中也会继续寻找最优结构

Hyperparameter	
λ	$0.1, 0.3, 0.5$
distillation temperature t	2
finetuning epochs	20
finetuning learning rate	$1e-5, 2e-5, 3e-5$
training learning rate	$2e-5$ (GLUE), $3e-5$ (SQuAD)
batch size	32 (GLUE), 16 (SQuAD)

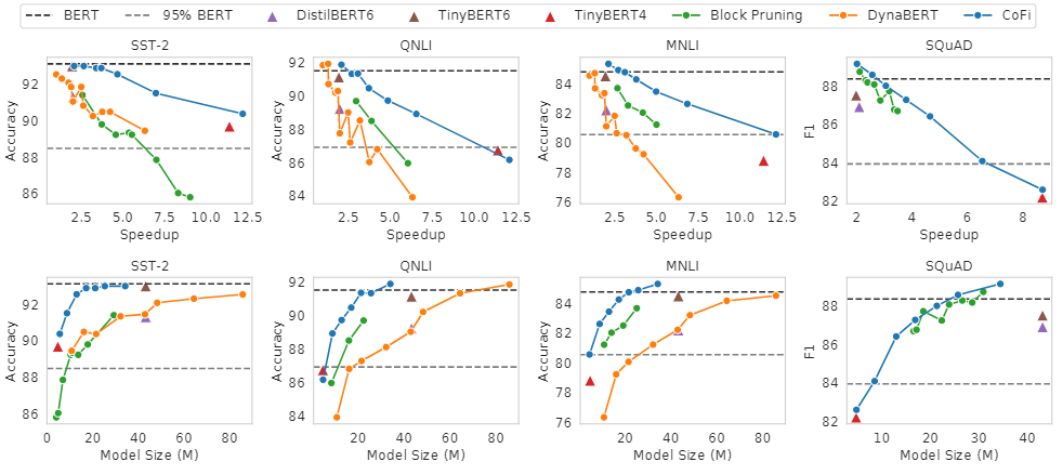
- **baseline**

  BERT、DistillBERT6、TinyBERT6、TinyBERT4、Block Pruning、DynaBERT

- **衡量标准**：Speedup rate——加速率（推理速度）；以未进行剪枝的BERT模型作为baseline，并在单个NVIDIA V100 GPU上评估具有相同硬件设置的所有模型，以对比不同模型的加速率。相较于压缩率更好

4.2 Main Results

- 整体性能



- **与TinyBERT4进行比较**

Task	SST-2 (67k)	QNLI (105k)	MNLI (393k)	QQP (364k)	CoLA (8.5k)	RTE (2.5k)	STS-B (7k)	MRPC (3.7k)	SQuAD (88k)	Train Time
$\text{BERT}_{\text{base}}$ (teacher)	93.1	91.5	84.8	91.2	61.2	70.0	88.7	85.0	88.4	-
TinyBERT_4 w/o GD	87.7	81.8	78.7	89.5	16.6	47.3	17.8	68.9	-	≤ 10
TinyBERT_4	89.7	**86.7**	78.8	90.0	32.5	63.2	**85.0**	81.4	82.1	~ 350
Speedup ↗	11.4×	11.4×	11.4×	11.4×	11.4×	11.4×	11.4×	11.4×	8.7×	-
CoFi Pruning (ours)	**90.6**	86.1	**80.6**	**90.1**	**35.6**	**64.7**	83.1	**82.6**	**82.6**	≤ 20
Speedup ↗	12.0×	12.1×	12.1×	11.0×	11.5×	11.9×	12.9×	11.9×	8.7×	-

Task	TinyBERT_4	CoFi (ours)
SST-2	89.7 → 91.6	90.6 → **92.4**
QNLI	86.7 → **87.6**	86.1 → 86.8
RTE	63.2 → 62.5	64.7 → **67.5**
MRPC	81.4 → 83.6	82.6 → **84.6**

- 消融实验

  探究不同的剪枝粒度对模型表现的影响，在不同数据集制定不同的稀疏率，下表为模型表现

  -hidden：去除z_{hidden}

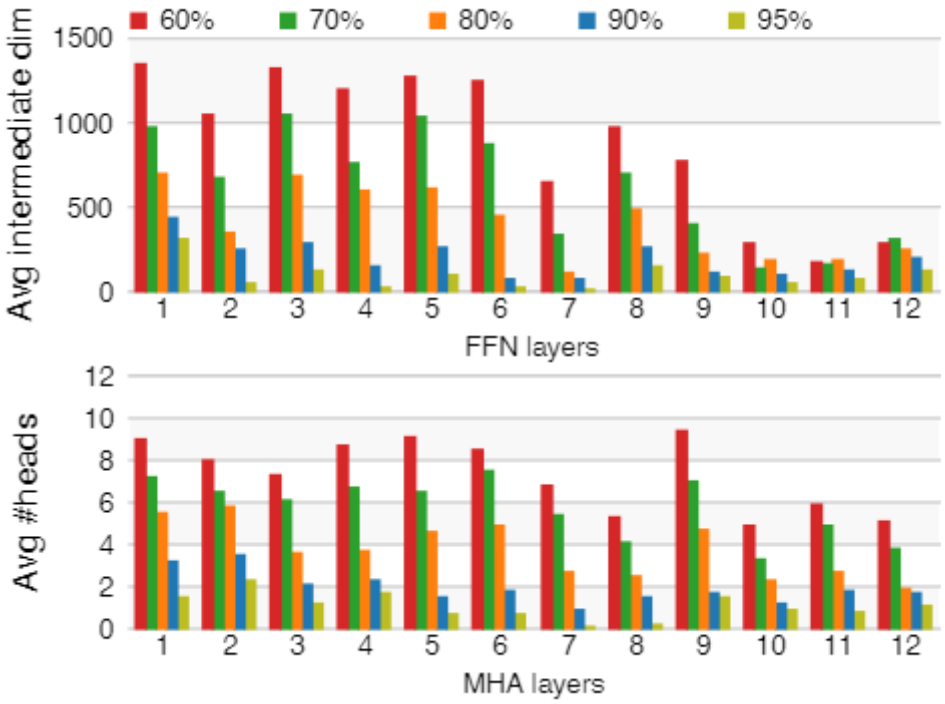
  -layer：去除z_{MHA} 、 z_{FFN}

	QNLI (60%)		QNLI (95%)		MNLI (60%)		MNLI (95%)		SQuAD (60%)		SQuAD (95%)	
	↗	acc	↗	acc	↗	acc	↗	acc	↗	F1	↗	F1
CoFi	2.1×	**91.8**	12.1×	**86.1**	2.1×	85.1	12.1×	**80.6**	2.0×	**89.1**	8.7×	**82.6**
−hidden	2.2×	91.3	13.3×	85.6	2.1×	**85.2**	13.7×	79.8	2.0×	88.7	9.7×	80.8
−layer & hidden	2.2×	91.3	7.2×	84.6	2.1×	84.8	7.0×	78.4	2.1×	88.5	6.4×	74.1
CoFi	2.1×	91.8	**12.1**×	86.1	2.1×	85.1	**12.1**×	80.6	2.0×	89.1	**8.7**×	82.6
−layer	2.1×	91.5	8.3×	86.7	2.1×	85.4	8.4×	80.6	2.0×	89.1	7.9×	80.5

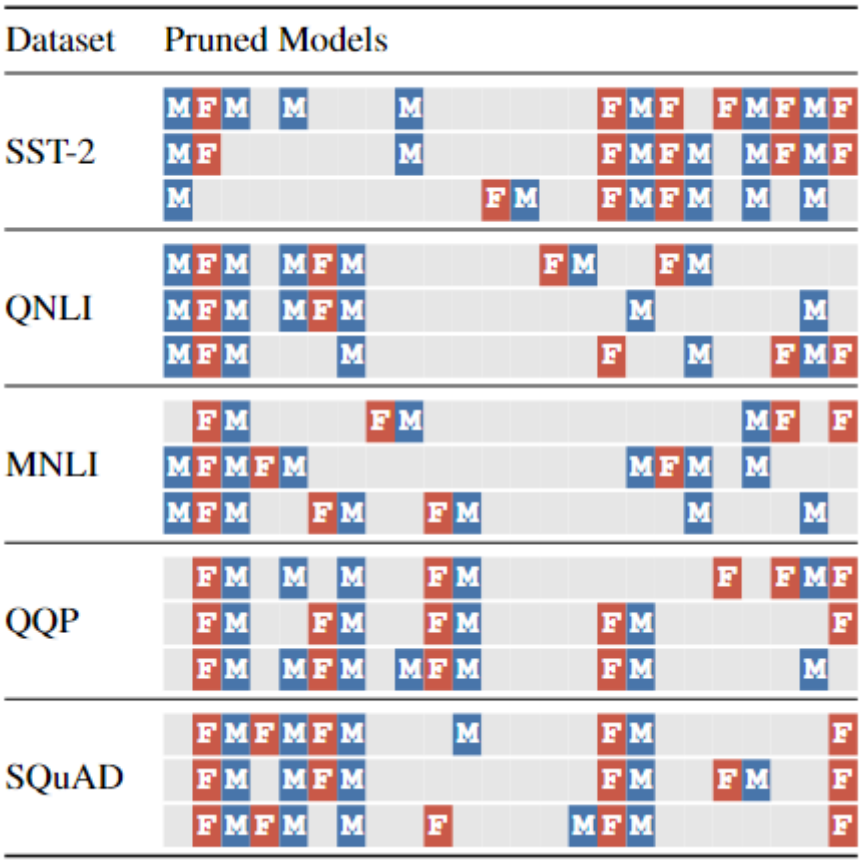
- 蒸馏目标
  - **加入蒸馏的必要性**：移除蒸馏完全会导致所有数据集的性能下降高达 1.9-6.8 个点
  - 一个简单的替代方案：“Fixed Hidden Distillation”，它将教师模型的每一层与学生模型中的相应层进行匹配——如果已经修剪层，则不会添加蒸馏目标
  - 固定的隐藏蒸馏不如 CoFi 中使用的动态层匹配目标。所提出的动态层匹配目标始终收敛到教师模型和学生模型层之间的特定对齐。例如，我们发现在 QNLI 上，训练过程在教师模型中的 3, 6, 9, 12 层与学生模型中的 1, 2, 4, 9 层动态匹配。此外，如表所示，删除它会损害所有数据集的性能除了 SST-2。
  - 在所有稀疏性中将动态层蒸馏添加到预测蒸馏上的消融研究。使用层蒸馏损失显然有助于提高所有稀疏率和不同任务的性能。

	SST-2	QNLI	MNLI	SQuAD
CoFi	90.6	**86.1**	**80.6**	**82.6**
$-\mathcal{L}_{\text{layer}}$	**91.1**	85.1	79.7	82.5
$-\mathcal{L}_{\text{pred}}, \mathcal{L}_{\text{layer}}$	86.6	84.2	78.2	75.8
Fixed Hidn Distil.	90.0	85.8	80.5	80.9

4.4 Structures of Pruned Models



- 前馈层在所有稀疏性上被显著修剪。例如，在60%的稀疏性水平上，修剪后FFN层中中间维度的平均数量减少了71%（3072→ 884），MHA中的平均头数减少了39%（12→ 7.3）。这表明FFN层比MHA层更为冗余。
- CoFi倾向于从上层修剪子模块多于从下层修剪子模块。
- 论文研究了剩余FFN和MHA层的数量，并在表中可视化了高度压缩模型（稀疏度=95%）的结果。尽管模型的压缩率大致相同，但模型结构差异很大，表明不同的数据集有不同的最优子网络



5 相关工作

- 结构化剪枝被广泛应用于计算机视觉领域，通道剪枝也成为CNN标准剪枝方式，也可以被应用于Transformer模型；非结构化剪枝虽然能使模型获得高稀疏率但是对于加速推理效果不明显，最典型的代表是彩票机制
- 除了修剪之外，还探索了许多其他技术来获得 Transformer 模型的推理加速，包括第 2.2 节中介绍的蒸馏、量化、动态推理加速和矩阵分解等方式来加速Transformer模型
- 上述方法都是针对于特定任务，但一些工作探索了上游修剪，用掩码修剪一个大型预训练语言模型；Chen等人(2020a)显示了一个70%的稀疏模型，保留了迭代幅度修剪产生的MLM精度。Zafir等人(2021)展示了上游非结构化修剪对下游修剪的潜在好处。
- 未来工作，将CoFi 应用于上游修剪是一个有前途的未来方向，以生成具有灵活结构的与任务无关的模型

6 总结

CoFi是一种结构化剪枝方法，包含所有级别的剪枝，例如：MHA/FFN层、单个头和基于tansformer模型的隐藏维度，同时为结构化修剪定制蒸馏目标。CoFi将模型压缩成与标准蒸馏模型截然不同的结构，获得了将近10倍的加速。论文中总结，来自大型模型的针对特定任务的结构化剪枝可以在精度损失很小的情况下进行大幅度的模型压缩，使得计算量大大减少，不需要预训练或者数据增强，通过剪枝也可以获得更灵活的模型结构，具有广泛前景。