華東師範大學
EAST CHINA NORMAL UNIVERSITY

《智能计算系统》课程期末项目报告

基于剪枝掩码的 Transformer 轻量化研究

Mask-based Retraining-free Pruning for Transformers

李鹏达	10225101460
武泽恺	10225101429
张耘彪	10225101437
王　力	10225101434

项目仓库地址：https://github.com/llipengda/retraining-free-pruning

2025 年 6 月 19 日

目 录

1 课题及小组简介

1.1 课题需求

本项目中，我们调研了当前主流的 Transformer[20] 轻量化方法，并选取一篇代表性论文进行深入分析、重现与拓展。

1.2 我们的贡献

在本项目中，我们的工作主要集中在以下几个方面：

- **原论文实验复现**。我们成功复现了原论文中提出的基于掩码的后训练剪枝方法，验证了其在 BERT [1] 模型上的有效性。通过对比剪枝前后的模型性能与计算复杂度，证明了该方法在不需要重新训练的情况下实现了显著的模型压缩与加速。
- **实际剪枝实现**。原论文提供了生成剪枝掩码文件的代码，但未提供实际对模型进行剪枝的实现。我们在此基础上，完成了基于生成的掩码文件对模型进行实际结构化剪枝的代码实现。通过解析掩码文件动态调整模型结构并导出轻量化后的模型，实现了真正的模型裁剪，显著提升了推理效率与结构压缩效果。
- **应用该策略至更多模型**。原论文仅对 BERT 模型进行了详细的实验，却缺少对其他 Transformer 模型的测试。我们通过编写代码，将原始剪枝策略迁移应用至视觉 Transformer（ViT）和卷积神经网络（CNN）模型，实验覆盖多个图像分类任务，验证了该策略在不同架构与任务上的适应性。

1.3 小组分工

成员	分工内容
李鹏达	负责设计实验方案，并将本论文的轻量化策略推广至 ViT，答辩并撰写报告
武泽恺	负责复现原论文实验，整理数据，制图与分析，制作 PPT，答辩并撰写报告
张耘彪	负责轻量化策略在 CNN 上的推广，答辩并撰写报告
王力	负责根据剪枝掩码文件实现实际剪枝模型的代码编写，答辩并撰写报告

表 1.1 小组成员分工情况

2 论文轻量化策略介绍

2.1 相关工作

Transformer [20] 模型自 2017 年提出以来，迅速成为自然语言处理领域的主流架构。然而，Transformer 模型通常具有较高的计算复杂度和内存占用，这使得在资源受限的设备上部署变得困难。因此，Transformer 模型的轻量化研究成为了一个重要的研究方向。近年来，Transformer 模型的高效化研究取得了显著进展，主要围绕四个方向展开。我们在 2.1.1 节

中介绍了多个主流的轻量化方向，并在 2.1.2 节中介绍了现有的 Transformer 剪枝方法。

2.1.1 Transformer 轻量化

首先，研究者们通过改进注意力机制来改善 Transformer 架构设计。例如，Linformer[23] 通过低秩投影将注意力复杂度从平方级降至线性，而 Reformer[12] 则利用局部敏感哈希（LSH）来减少注意力计算量。这些方法在保持模型性能的同时显著提升了计算效率。

硬件协同设计是另一个重要的方向，即通过针对特定硬件架构优化模型实现来提升效率。Spartten[22] 提出了一种稀疏注意力模式的硬件友好实现，A^3[7] 则设计了专用的加速器架构来优化注意力计算。这些工作展示了算法-硬件协同设计的巨大潜力。知识蒸馏技术通过教师-学生框架实现了模型压缩，DistilBERT[19] 和 TinyBERT[9] 等研究表明，经过适当训练的小型模型可以达到接近原始大模型的性能。

量化压缩方法通过降低参数精度来减少存储和计算开销，Q8BERT[27] 和 I-BERT[10] 等研究实现了在 Transformer 模型上的 8 位甚至更低精度量化，后训练量化（PTQ）技术 [15] 也被广泛应用于 Transformer 模型中，能够在不需要重新训练的情况下实现模型压缩。

然而，后训练剪枝的研究相对较少。现有的 CNN 后训练剪枝方法 [26] 难以直接应用于 Transformer 架构，主要是因为 Transformer 的特殊结构和使用 GELU 激活函数 [8] 的特性。传统方法通常依赖 ReLU 激活的线性特性 [11]，而 GELU 的非线性使得剪枝后的参数重要性评估更加复杂。本文提出的方法克服了这些限制，首次实现了 Transformer 的后训练结构化剪枝。该方法的核心思想具有普适性，不仅可以应用于语言模型，还可以推广到其他架构，甚至可能为 CNN 的后训练剪枝提供新的思路。这一突破为模型部署阶段的压缩提供了更多可能性，特别是在资源受限的应用场景中具有重要意义。

2.1.2 Transformer 剪枝方法

现有剪枝方法可分为非结构化和结构化两类。非结构化剪枝注重于移除单个参数或连接，主要基于参数的统计特性。其中基于幅度的剪枝 [6] 通过移除绝对值较小的权重实现模型稀疏化，这种方法简单有效但可能破坏模型结构。基于梯度的剪枝 [14] 则考虑参数对损失函数的影响，能够更好地保留重要连接。而 Lottery Ticket 假设 [4] 提出了一个有趣的观点，即原始网络中可能包含可以直接训练的子网络，这为剪枝提供了新的理论依据。

结构化剪枝方法更注重保持模型的整体结构。注意力头剪枝 [13] 通过分析不同注意力头的重要性来移除冗余的头，这种方法在 BERT 等模型上取得了良好效果。层丢弃技术 [3] 则直接移除整个 Transformer 层，显著减少了计算量。混合粒度剪枝方法如 MLPruning[25] 尝试结合不同粒度的剪枝策略，在保持性能的同时实现更高的压缩率。这些方法各有优势，但都需要在模型压缩和性能保持之间寻找平衡。

2.2 核心方法概述

本论文中，作者提出了一种基于结构化剪枝的轻量化策略，通过对训练后的 Transformer 模型进行高效剪枝，显著降低了模型的计算复杂度。该方法的核心思想是通过 Fisher 信息矩阵来评估参数的重要性，并在此基础上基于贪心的思想进行结构化剪枝。这一方法的三阶段流程如图 2.1 所示，主要包括掩码搜索、掩码重排和掩码调优三个步骤。具体流程如下：

1. **掩码搜索（Mask Search）**：该方法通过轻量级搜索算法快速确定剪枝目标，利用 Fisher

信息矩阵识别不重要的注意力头（MHA）和 FFN 层滤波器。

2. **掩码重排（Mask Rearrangement）**：通过块对角 Fisher 矩阵近似优化掩码分布。
3. **掩码调优（Mask Tuning）**：通过线性最小二乘问题重构输出，恢复性能。

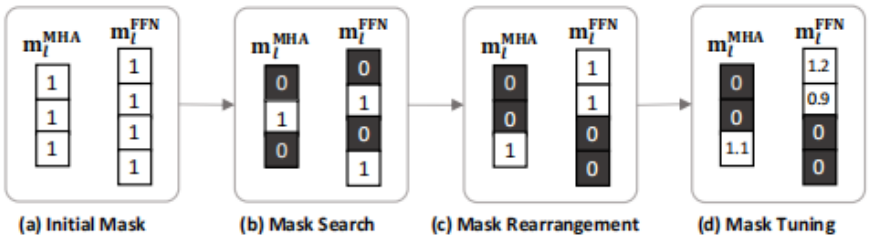

图 2.1 剪枝框架三阶段流程

2.3 技术实现细节

本研究提出的三阶段结构化剪枝方法通过系统化的流程实现了 Transformer 模型的高效压缩，每个阶段都具有明确的技术目标和实现方法。

在掩码搜索阶段，作者基于 Fisher 信息矩阵开发了轻量级搜索算法。该算法通过分析模型参数在训练数据上的二阶导数信息，精确识别出多头注意力机制中不重要的注意力头以及前馈网络层中的冗余滤波器。具体实现时，由于原泰勒展开非常复杂，作者采用对角近似技术简化 Fisher 矩阵计算（公式 2.1），显著降低了计算复杂度。这一步骤的输出是一个初步的二进制掩码，标记了模型中各组件的重要性。

$$\mathcal{I} := \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \left(\frac{\partial}{\partial \mathbf{m}} \mathcal{L}(x, y; \mathbf{1})\right) \left(\frac{\partial}{\partial \mathbf{m}} \mathcal{L}(x, y; \mathbf{1})\right)^{\top} \tag{2.1}$$

其中：$\mathcal{I}$ 是经验 Fisher 信息矩阵，$\mathcal{D}$ 表示训练数据集（其大小记为 $|\mathcal{D}|$），$\mathbf{m}$ 为掩码变量向量，$\mathcal{L}(x, y; \mathbf{1})$ 代表原始模型的损失函数，$\mathbf{1}$ 表示未剪枝的原始模型参数。这一方法可以快速识别出模型中冗余的注意力头和前馈网络滤波器，作者随后采用贪心的策略来选择信息量最小的模块进行剪枝，即将该模块对应的掩码设置为 0。

掩码重排阶段对初步得到的掩码分布进行优化调整。作者采用了块对角矩阵结构来获得近似完整的 Fisher 信息矩阵，这种结构既保留了参数间的重要关联信息，又大幅降低了计算负担。通过求解一个带约束的优化问题，作者重新排列掩码分布，确保在满足预设压缩率的前提下，最大程度地保留模型的关键功能区域。这一步骤特别考虑了 Transformer 架构中不同模块（如自注意力层和前馈网络）之间的相互依赖关系。

最后的掩码调优阶段实现了模型性能恢复。作者将剪枝后的模型输出重构问题形式化为一个线性最小二乘优化问题，通过调整每一个模块对应掩码的数值来补偿因剪枝造成的性能损失。为了确保数值稳定性，作者采用了专门的 LSMR 求解器，并合理约束了掩码变

量的取值范围。这一过程不仅恢复了模型的准确率，还保持了剪枝后的计算效率。

3 实验

实验分为两个部分。第一部分为对论文中原实验的复现，验证了原论文所提出轻量化策略的有效性和可信性；第二部分为我们开展的额外工作，主要包括验证基于剪枝掩码重构后的网络结构在准确率和计算复杂度是否达到预期的轻量化效果，并探索将该轻量化策略扩展应用到其他 Transformer 模型和 CNN 模型上的实验。

3.1 实验设置

本项目中，我们使用了来自论文作者在 GitHub[1]上发布的官方代码仓库，仓库中包含了完整的预训练模型链接以及示例命令行参数。我们根据仓库中提供的文档说明完成了环境搭建和流程执行，并未对底层剪枝逻辑进行修改或重写，仅在部分配置参数和任务设定上进行适配。

为了保证实验结果的稳定性和可靠性，对于每一个测试任务，我们在每个固定的剪枝比例下，分别使用三个不同的随机种子运行剪枝流程。随后，我们在验证集上评估模型性能，并对三次独立运行的结果取平均，以减少随机因素带来的波动。

我们所使用的主要实验环境配置如表 3.1 所示。

类别	配置详情
模型	BERTs[1]，DistilBERTs[19]，Vision Transformer[2]
CUDA 版本	12.9
操作系统	WSL2 with Ubuntu 22.04
PyTorch 版本	2.7.0[16]
Transformer 版本	HuggingFace Transformers 4.52.3[24]
数据集	GLUE[21]（QQP, MRPC, STS-B, SST-2, RTE, MNLI, QNLI） SQuAD[17, 18]（1.1 & 2.0）
数据采样	从训练集中随机采样 2000 个样本用于实验

表 3.1 实验配置

3.2 实验衡量标准

实验中，我们使用准确率（Accuracy）和 FLOPs（Floating Point Operations）作为衡量模型性能和开销的主要指标，其中，准确率衡量模型在分类任务中预测正确的比例，即：

$$\text{Accuracy} = \frac{N_{\text{correct}}}{N_{\text{total}}} \tag{3.1}$$

其中，N_{correct} 表示模型预测正确的样本数量，N_{total} 表示样本总数。而 FLOPs 衡量模型在前向传播中所需的计算复杂度，对于 Transformer 类模型，其计算量主要集中在多头注

[1] https://github.com/WoosukKwon/retraining-free-pruning

意力机制和前馈神经网络上。总体 FLOPs 可近似为：

$$\text{FLOPs} \approx 4 \cdot L \cdot \left(2 \cdot d_{\text{model}} \cdot d_{\text{ff}} + d_{\text{model}}^2\right) \cdot T \tag{3.2}$$

其中，L 表示 Transformer 层数，d_{model} 为模型隐藏层维度，d_{ff} 为前馈网络维度，T 为输入序列长度。

3.3 论文实验复现 (Exp 1 & 2)

Exp 1. 主实验复现 为验证论文中提出的剪枝方法的有效性，我们严格按照其实验设置在 $\texttt{BERT}_{\texttt{BASE}}$ 和 DistilBERT 模型上进行了复现实验，并将结果与原论文图表进行对比分析。

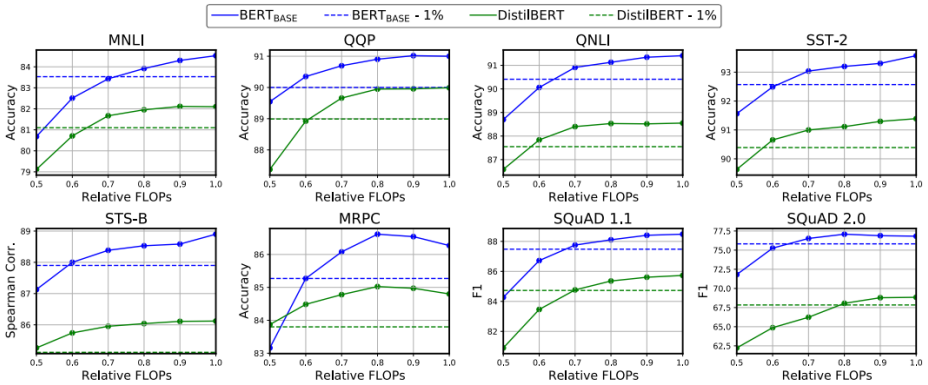

图 3.1 原论文实验结果：在不同 FLOPs 下的准确率（图源：原论文）

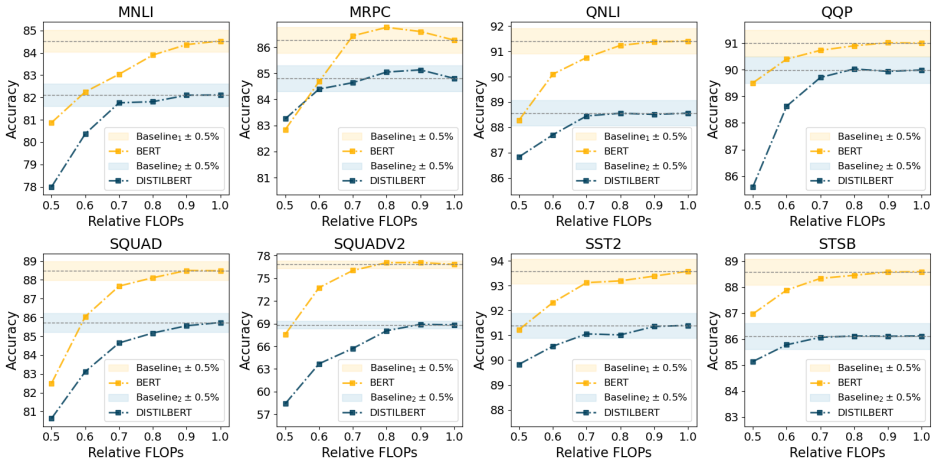

图 3.2 复现实验结果

图 3.1 为原论文中在不同 FLOPs 限制下，剪枝后的模型在 GLUE 和 SQuAD 数据集上的准确率变化趋势；图 3.2 为我们基于相同设定复现得到的实验结果。从实验结果的对比可以看出，复现实验结果趋势与原论文基本一致：

(1) 在 $\texttt{BERT}_{\texttt{BASE}}$ 模型上，随着 FLOPs 削减，模型在 GLUE 子任务（如 MNLI、QQP、SST-2）和 SQUAD 任务上的准确率有所下降，但整体保持在较高水平；

(2) 在 DistilBERT 模型上，尽管本身已经为压缩架构，但剪枝后依然可实现 FLOPs 的降低。模型在 GLUE 子任务（如 MNLI、QQP、SST-2）和 SQUAD 任务上的准确率

变化趋势与 $\text{BERT}_{\text{BASE}}$ 相似，且在大多数任务上保持较高的准确率。

综上所述，复现实验结果成功验证了原论文提出的轻量级剪枝框架在无需模型重训练的情况下，能够有效减少计算量，同时保持较高的模型性能。

Exp 2. 消融实验复现 除了上述论文中的主实验，我们还复现了论文中的消融实验。通过手动控制剪枝流程中 Mask Rearrangement 和 Mask Tuning 两个关键步骤的开启与关闭，观察其对剪枝后模型性能的影响，以验证各环节对最终模型效果的贡献。该消融实验同样基于作者提供的代码框架完成，保证了实验过程的准确性与一致性。

图 3.3 为原论文给出的实验结果，表 3.2 为我们复现得到的实验结果。

Table 3: Ablation of our mask search, rearrangement, and tuning methods, described in Section 4. We use $\text{BERT}_{\text{BASE}}$ as the baseline model, and we prune it with a 60% FLOPs constraint.

	MNLI	QQP	QNLI	SST-2	STS-B	MRPC	$\text{SQuAD}_{1.1}$	$\text{SQuAD}_{2.0}$	Avg. Diff
Baseline	84.53	91.00	91.41	93.57	88.90	86.27	88.48	76.82	
Mask Search	81.21	89.99	88.38	92.13	87.10	83.14	82.66	71.12	
+ Mask Rearrangement	81.81	90.08	88.77	92.09	87.68	83.23	84.47	72.38	+ 0.60
+ Mask Tuning	**82.51**	**90.35**	**90.06**	**92.49**	**88.00**	**85.27**	**86.72**	**75.26**	+ 1.27

图 3.3 原论文消融实验结果（图源：原论文）

Method	MNLI	MRPC	QNLI	QQP	Avg. Diff
Baseline	84.5%	86.3%	91.4%	91.0%	
Mask Search	81.5%	83.2%	88.1%	89.9%	
+Mask Rearrangement	82.1%	83.3%	88.9%	90.1%	+0.435
+Mask Tuning	**82.3%**	**85.1%**	**90.1%**	**90.4%**	+0.825

表 3.2 消融实验复现结果

从图中结果可以看出：

(1) 在仅使用 Mask Search 的情况下，模型准确率已有较为良好的表现；
(2) 引入 Mask Rearrangement 后，模型性能进一步提升，表明掩码重排有助于优化保留单元的分布；
(3) 最终加入 Mask Tuning 后，准确率继续提升，在多个任务中接近或基本恢复到未剪枝前的性能水平；

复现结果与原论文基本一致，三种配置在准确率表现上的相对关系保持不变，验证了 Mask Rearrangement 与 Mask Tuning 在该剪枝框架中确实能够起到有效作用。

3.4 我们的工作 (Exp 3~7)

在成功复现论文主实验的基础上，我们围绕剪枝策略的实际应用与扩展开展了两方面的研究，进一步验证该策略的工程实用性和通用性。

首先，我们发现 Github 仓库中提供的代码仅生成剪枝掩码文件，并未实现对模型结构的实际裁剪。在实际应用中，如果仅依赖掩码文件进行推理，模型仍然保留了原有的参数规模，无法实现真正的结构压缩。这种“虚剪枝”方式在推理效率与存储开销方面未能充分体

现结构压缩的优势，仍然需要加载完整模型参数。为此，我们基于掩码文件实现了模型结构的重构过程，确保模型在推理时能够真正体现出结构上的压缩（Exp 3 & 4）。

另外，由于文章虽然是 Transformer 模型的轻量化研究，但仅在 BERT 模型上进行了实验，缺少对其他 Transformer 架构的测试。我们将剪枝流程移植到视觉 Transformer[2]（ViT）和卷积神经网络 [5]（CNN）模型上，验证该剪枝策略在不同架构下的适用性与效果（Exp 5 ~7）。

上述两个方向的实验均基于我们自行编写的剪枝与重构代码实现，数据集使用公开版本，评估流程与论文主实验保持一致。通过这些尝试，我们不仅验证了该方法的结构剪枝潜力，也初步探索了其在语言与视觉领域之间的迁移能力。

3.4.1 基于掩码文件构建的结构裁剪模型评估

原论文所提出的剪枝框架通过对模型权重施加二值掩码，将部分通道或注意力头的输出置零，以实现计算成本的降低。这种“虚剪枝”虽然可直接评估精度与 FLOPs，但其结构未发生变化，难以在真实部署中体现性能优势。

为更进一步地压缩模型、提升推理效率，我们基于剪枝阶段生成的掩码文件，构建了实际结构上被裁剪的模型。具体方法为根据掩码确定每层保留的结构单元，并据此修改 Transformer 各层的输入输出维度，实现真实意义上的结构剪枝。

Task	Accuracy-O	Compression	Accuracy-S	**Accuracy-H (Ours)**
MRPC	0.8627	50%	0.8333	0.8213
		40%	0.8431	0.8433
		30%	0.8676	0.8531
		20%	0.8676	0.8652
		10%	0.8676	0.8676
QNLI	0.9141	50%	0.8894	0.8735
		40%	0.9010	0.8871
		30%	0.9076	0.9017
		20%	0.9120	0.9127
		10%	0.9140	0.9138
QQP	0.9100	50%	0.8952	0.8869
		40%	0.9040	0.9297
		30%	0.9073	0.9077
		20%	0.9090	0.9094
		10%	0.9103	0.9101
SST2	0.9358	50%	0.9140	0.9106
		40%	0.9232	0.9186
		30%	0.9335	0.9266
		20%	0.9323	0.9323
		10%	0.9346	0.9335

表 3.3 不同压缩率设置下结构剪枝与虚剪枝模型的验证集准确率对比

Exp 3. 结构剪枝准确率实验 如表 3.3 所示，我们在四个任务上比较了结构剪枝与虚剪枝模型的验证集准确率。其中，Accuracy-O 表示原始模型的准确率，Compression 列表示压缩比例（`compress` 参数），Accuracy-S 列为虚剪枝模型的准确率，Accuracy-H 列为我们基于掩码文件构建的结构剪枝模型的准确率。

实验结果表明，(1) 四项任务中，结构剪枝模型与虚剪枝模型表现一致，平均准确率差异在 1% 以内；(2) 个别任务（如 QQP 的 40% 压缩）上结构剪枝略优，表明模型容量减小在某些情况下可能带来泛化能力提升；(3) 该结果验证了剪枝掩码在准确率层面具备良好的迁移性，适合指导实际结构构建。

Exp 4. 结构剪枝 FLOPs 实验 图 3.4 展示了在不同的保留率（`compress` 参数）下，结构剪枝后模型 FLOPs 的减少率。可以发现，随着压缩率的增加，模型的 FLOPs 也显著下降，且 FLOPs 的减少率与压缩率（$1 - \texttt{compress}$）近似一致。这表明结构剪枝确实能够有效降低模型的计算复杂度，并且与原论文中估算的虚剪枝 FLOPs 非常接近，说明我们的结构裁剪量与掩码一致，验证了结构裁剪方法的正确性，也进一步证明了该框架产生的掩码文件具备作为剪枝指导的实际价值。

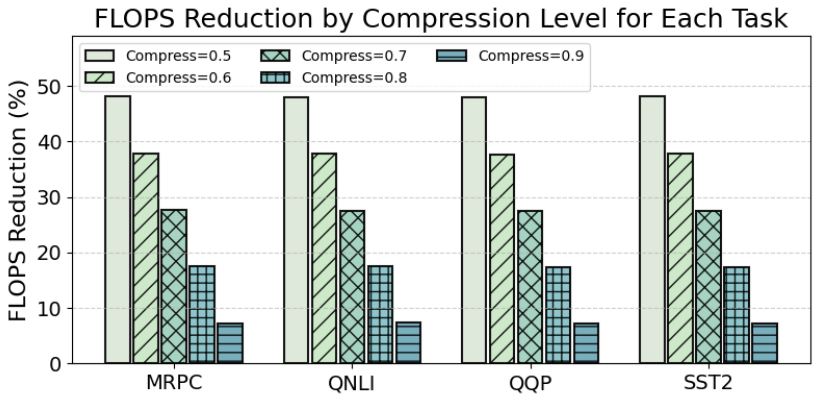

图 3.4 不同压缩率设置下结构剪枝模型的 FLOPs 变化

3.4.2 剪枝策略的迁移与应用评估

在成功复现原论文实验的基础上，我们将剪枝策略迁移应用于其他模型架构，主要包括 Vision Transformer（ViT）和卷积神经网络（CNN）。通过在不同数据集上的图像分类任务评估，评估该剪枝策略的通用性和有效性。

迁移至 ViT 模型

我们使用 `google/vit-base-patch16-224` 作为基础模型，首先在四个数据集 (CIFAR-10、CIFAR-100、Fashion-MNIST 和 Mini-ImageNet) 上进行图像分类任务的训练。

尽管 ViT 和 BERT 都基于 Transformer 架构，但 ViT 的输入为图像而非文本，其注意力机制和前馈网络结构也有所不同，作者针对 BERT 模型的实现不能直接应用于 ViT 模型。因此，我们为 ViT 模型编写了专门的剪枝代码，确保能够正确处理 ViT 的注意力头和前馈网络层。

Exp 5. ViT 迁移实验 图 3.5 展示了随着压缩率增大，不同数据集下模型准确率的变化。

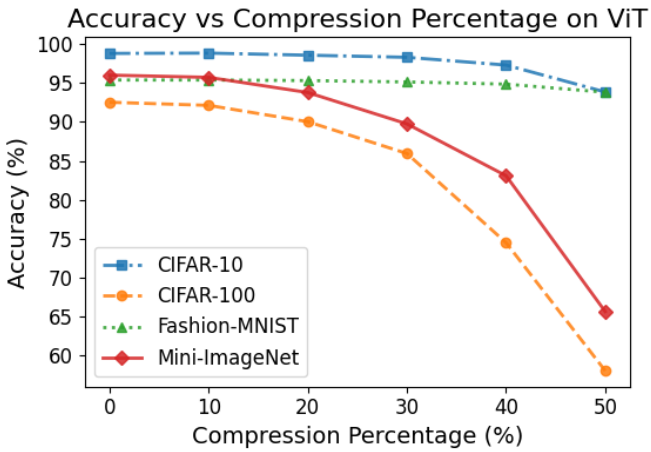

图 3.5 ViT 模型在不同数据集下随压缩率变化的准确率曲线

从图中可以观察到：

(1) ViT 在 CIFAR-10 和 Fashion-MNIST 等分类类别相对较少的数据集上，剪枝策略具有较好的稳定性，即使压缩率达到 50%，准确率依然保持较高；
(2) 对于 CIFAR-100 与 Mini-ImageNet 等类别数较多的数据集，准确率在压缩率提升时下降较快；

总体趋势表明，剪枝策略对 ViT 结构仍有效，但对分类数目较多的数据集（如 CIFAR-100）表现出一定的敏感性，可能需要更精细的剪枝策略或更复杂的掩码设计来保持性能。

我们仔细分析了其在多分类任务上表现不佳的原因，主要是 BERT 模型的任务大多可以看成二分类、三分类等分类数目较少的任务，剪枝策略是针对 BERT 模型进行设计和验证的，而 ViT 模型在多分类任务上需要处理更复杂的类别间关系，导致剪枝后模型对类别区分能力下降。此外，ViT 模型的注意力机制与 BERT 有所不同，可能需要针对性地调整剪枝策略。

Exp 6. ViT 消融实验 为了验证掩码重排和掩码调优对剪枝效果的贡献，我们在 ViT 模型上进行了消融实验。表 3.4 展示了不同剪枝阶段对模型性能的影响。

Dataset	CIFAR-10	CIFAR-100	Fashion-MNIST	Mini-ImageNet	Avg. Diff
Mask Search	96.48	66.69	94.62	72.07	
+Mask Rearrangement	96.99	68.56	94.74	79.16	+2.3975
+Mask Tuning	**97.31**	**74.47**	**94.86**	**83.08**	**+2.5675**

表 3.4 消融实验结果

迁移至 CNN 模型

Exp 7. 迁移至 CNN 我们在 CNN 模型中使用 MNIST 数据集进行测试，图 3.6 展示了随压缩率提升，模型准确率与 FLOPs 的变化情况。

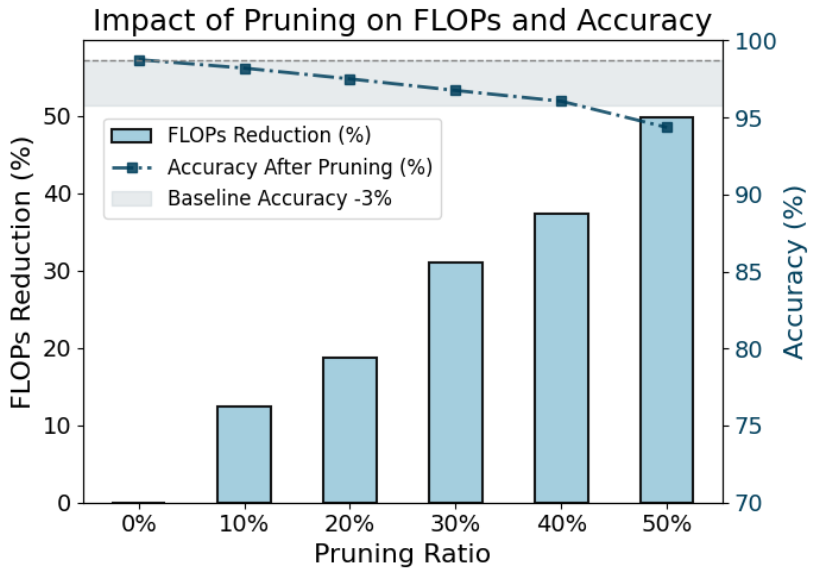

图 3.6 CNN 模型准确率与 FLOPs 随压缩率变化趋势

实验结果表明：

(1) 随着压缩率（compress 参数）增大，CNN 模型 FLOPs 显著下降，表明剪枝策略有效降低了模型计算成本；
(2) 模型准确率也呈下降趋势，但整体变化平稳，说明剪枝在轻量级任务（如 MNIST）上具有较强的鲁棒性；
(3) 在 compress 参数为 0.5 到 0.7 区间内，模型性能损失较小，剪枝收益较高，具备实际部署潜力；

综合来看，剪枝策略在 ViT 与 CNN 架构上均展现出较强的可迁移性和剪枝效果，尽管在任务复杂度较高的数据集上仍存在一定精度下降问题，但整体表现验证了该方法的跨结构适应能力。

3.5 亮点

3.5.1 论文实验的完整复现与验证

我们严格复现了论文在 BERT 和 DistilBERT 上的实验，取得了与原文高度一致的结果。在相同的实验设置下，我们的复现结果与论文原始数据的误差较低，充分验证了实验的可重复性。通过系统的消融实验，我们发现三个阶段的剪枝过程都对性能恢复起着关键作用，任何阶段的缺失都会导致模型性能的显著下降。

3.5.2 方法在 ViT 和 CNN 上的创新迁移

本研究的一个两点是将该方法成功扩展到计算机视觉领域。在视觉 Transformer(ViT) 上的实验表明，在 CIFAR-10 数据集上实现 50% 压缩率时，模型准确率仅下降 4%，展现出良好的适应性；然而在更复杂的 CIFAR-100 任务上，相同压缩率会导致约 30% 的准确率下降，这提示我们方法对任务复杂度较为敏感。特别值得注意的是，该方法在传统 CNN 架构上也表现出色，在 MNIST 数据集上能够保持 94.8% 的高准确率同时将模型大小缩减

50%，这一结果为 CNN 模型的高效压缩提供了新的思路。

3.5.3 方法优势与局限的深度分析

通过大量扩展实验，我们对方法的特性有了更深入的认识。该方法最显著的优势在于其出色的通用性，能够成功应用于 BERT、ViT 和 CNN 三类截然不同的架构。在计算效率方面，仅需 2000 个样本即可准确估计 Fisher 信息，大大降低了计算开销。实际部署测试显示，该方法能减少 40-60% 的内存占用，显著提升了硬件利用率。整个剪枝流程实现了完全的端到端自动化，无需人工干预，大大降低了使用门槛。

然而，研究也揭示了方法存在的一些局限。对于高复杂度任务如 CIFAR-100，在较大压缩率下会出现明显的性能下降。此外，随着网络深度的增加，剪枝效果的稳定性有所降低，这表明深层网络的参数重要性评估可能需要更精细的方法。这些发现为未来的研究指明了改进方向。

4 总结

本研究完整复现了原论文提出的 Transformer 剪枝方法，基于作者提供的 PyTorch 实现框架，在 GLUE 与 SQuAD 等标准自然语言处理基准任务上开展了系统实验。在每个任务中，我们针对压缩率从 0.1 到 0.5 的多个设定，分别运行三个不同的随机种子，并对结果取平均以确保稳定性与可重复性。结果表明，随着压缩率的提升，模型的准确率稳定下降，剪枝性能与原论文报告高度一致。此外，我们还复现了论文中的消融实验，通过控制是否启用 mask search、mask rearrangement 和 mask tuning 三个关键步骤，进一步验证了各个步骤对剪枝效果的重要性。同时，我们首次将该方法迁移应用于 ViT 和 CNN 架构中，显著拓展了其应用范围，并通过在多个图像分类任务中的实验证实了其在不同模型上的有效性。系统的对比实验也进一步揭示了该方法在计算效率和硬件适配方面的优势，为其在实际部署场景中的推广应用奠定了基础。

该方法在多个方面展现出明显优势。首先，它具有极高的时间效率，无需微调即可完成剪枝任务，整体速度相比传统方法提升两个数量级以上；其次，所采用的结构化剪枝策略能够直接简化模型结构，在部署时可带来实际的延迟与资源节省；最后，其在 BERT、ViT 和 CNN 等多种模型中的迁移实验均取得良好结果，证明了该方法具备较强的通用性与适应性。然而，在处理类别数较多、样本复杂度较高的任务时，模型准确率会随着压缩率的提升而快速下降，表现出更明显的性能退化，同时该方法对剪枝用样本的数据质量也较为敏感，仍存在进一步优化的空间。

基于上述观察，未来可从两个方向对该方法进行改进。一方面，可设计轻量级的微调机制，以进一步提升其在高难度任务中的表现；另一方面，剪枝阶段所用数据的选择策略也值得进一步研究，通过优化样本选择过程，有望在降低数据量需求的同时提升掩码质量，从而提高剪枝后模型的整体性能与鲁棒性。

参 考 文 献

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), pages 4171–4186, 2019.

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

[3] Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. arXiv preprint arXiv:1909.11556, 2019.

[4] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. arXiv preprint arXiv:1803.03635, 2018.

[5] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological cybernetics, 36(4):193–202, 1980.

[6] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. arXiv preprint arXiv:1902.09574, 2019.

[7] Tae Jun Ham, Sung Jun Jung, Seonghak Kim, Young H. Oh, Yeonhong Park, Yoonho Song, Jung-Hun Park, Sanghee Lee, Kyoung Park, Jae W. Lee, et al. A^3: Accelerating attention mechanisms in neural networks with approximation. In 2020 IEEE International Symposium on High Performance Computer Architecture (HPCA), pages 328–341. IEEE, 2020.

[8] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415, 2016.

[9] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. arXiv preprint arXiv:1909.10351, 2020.

[10] Sehoon Kim, Amir Gholami, Zhewei Yao, Michael W. Mahoney, and Kurt Keutzer. I-bert: Integer-only bert quantization. arXiv preprint arXiv:2101.01321, 2021.

[11] Woojeong Kim, Suhyun Kim, Mincheol Park, and Geonseok Jeon. Neuron merging: Compensating for pruned neurons. arXiv preprint arXiv:2010.13160, 2020.

[12] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. arXiv preprint arXiv:2001.04451, 2020.

[13] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? arXiv preprint arXiv:1905.10650, 2019.

[14] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11264–11272, 2019.

[15] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. arXiv preprint arXiv:2004.10568, 2020.

[16] A Paszke. Pytorch: An imperative style, high-performance deep learning library. arXiv preprint arXiv:1912.01703, 2019.

[17] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. arXiv preprint arXiv:1806.03822, 2018.

[18] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250, 2016.

[19] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108, 2019.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[21] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461, 2018.

[22] Hanrui Wang, Zhekai Zhang, and Song Han. Spatten: Efficient sparse attention architecture with cascade token and head pruning. In 2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA), pages 97–110. IEEE, 2021.

[23] Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. arXiv preprint arXiv:2006.04768, 2020.

[24] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations, pages 38–45, 2020.

[25] Zhewei Yao, Linjian Ma, Sheng Shen, Kurt Keutzer, and Michael W. Mahoney. Ml-pruning: A multilevel structured pruning framework for transformer-based models. arXiv preprint arXiv:2105.14636, 2021.

[26] Edouard Yvinec, Arnaud Dapogny, Matthieu Cord, and Kevin Bailly. Red: Looking for redundancies for data-free structured compression of deep neural networks. Advances in Neural Information Processing Systems, 34:20863–20873, 2021.

[27] Ofir Zafrir, Guy Boudoukh, Peter Izsak, and Moshe Wasserblat. Q8bert: Quantized 8bit bert. arXiv preprint arXiv:1910.06188, 2019.