利用微型数据集实现实用的网络加速：假设、理论与算法

Guo-Hua Wang 和 Jianxin Wu，*IEEE* 会士

摘要——由于数据隐私问题，利用微型训练集加速网络已成为实践中的迫切需求。以往方法通过滤波器级剪枝，在实验中取得了令人鼓舞的结果。本文从理论上研究这一问题，并提出一种与理论结果高度吻合的有效算法。首先，我们提出微调凸性假设，以解释近期的少样本压缩算法为何不会遭遇过拟合问题。在此基础上，我们进一步首次建立理论来解释这些方法。与直接微调剪枝网络相比，特征模仿被证明能降低参数方差，因此更易于优化。根据理论结论，我们指出，就优化凸性更强、加速比更高而言，丢弃块是一种本质上更优的少样本压缩方案。为选择要丢弃的块，我们提出一种新的度量指标——可恢复性，用以有效衡量恢复压缩网络的难度。最后，我们提出一种名为 PRACTISE 的算法，仅使用微型训练集即可加速网络。PRACTISE 的表现大幅优于以往方法。在延迟降低 22% 的情况下，它在 ImageNet-1k 上的成绩平均高出以往方法 7 个百分点。该方法在无数据或域外数据设置下也表现良好。我们的代码位于 https://github.com/DoctorKey/Practise。

索引词—深度神经网络、网络加速、少样本压缩、知识蒸馏、图像分类。

arXiv:2303.00972v1 [cs.CV] 2 Mar 2023

1 引言

DEEP 度神经网络取得了显著的成功，但其计算成本高昂。为加速神经网络，人们提出了许多方法，例如网络剪枝 [1], [2], [3], [4], 解耦 [5], [6] 和量化 [7], [8]。然而，大多数以往方法都依赖原始训练集（即全部训练数据）来恢复压缩模型的准确率。但是，为了保护数据隐私和/或实现快速部署，可用的训练数据可能十分有限。例如，客户经常要求算法提供商加速其神经网络，但出于隐私考虑，无法提供全部训练数据。客户只提供未压缩的模型和少量训练样本。在某些极端情况下，甚至无法获取任何数据点。算法工程师需要自行合成图像，或收集一些域外训练图像。因此，仅用极少量样本加速深度网络，正逐渐成为一个亟待解决的关键问题。

先前工作 [9], [10], [11] 已取得令人鼓舞的成果。然而，其中大多数工作通过经验性地设计算法来解决少样本压缩问题，却留下了许多尚未解答的基本问题，例如

- 为什么剪枝模型可以仅用极少量样本进行微调，而不会遭受过拟合?
- 为什么微调时必须采用知识蒸馏（知识蒸馏），尤其是基于特征的模仿？知识蒸馏或特征模仿有哪些优势?
- 哪种压缩方案最适合少样本压缩？压缩方案应具备哪些特性

• *G.-H. Wang* 和 *J. Wu* 就职于中国南京市 *210023* 南京大学新软件技术国家重点实验室。*J. Wu* 为通讯作者。本文部分得到国家自然科学基金委员会项目编号 *62276123* 和项目编号 *61921006* 的资助。电子邮箱：*wangguohua@lamda.nju.edu.cn*，*wujx2001@gmail.com*

方案具备哪些优势，能够实现更好的性能?

本文旨在从理论和实证两个角度回答所有这些关键问题。

从头开始训练深度网络具有高度非凸性，并会形成复杂的优化景观 [12], [13], [14]。但我们发现，微调剪枝模型的过程却呈现出令人惊讶的凸性。基于大量实验，我们提出了微调凸性假设，推测微调剪枝模型的损失景观关于参数是凸的。此外，凸性的程度决定了微调所需的训练数据量。凸性使得剪枝模型能够进行少样本微调，而不会出现严重过拟合。为了更好地理解这种凸性为何出现，我们重新构建模型压缩过程，在剪枝结构周围插入线性适配器模块，并且仅 用少量样本微调这些模块。实验表明，其优化景观和准确率与端到端微调几乎相同。也就是说，原始模型与剪枝模型（微调前）之间的差异，在潜在表示/特征空间中是凸的，甚至接近线性。

尽管凸性降低了少样本微调的难度，但不同少样本压缩方法之间的准确率差距仍然很大。与常规微调相比，现有研究发现，知识蒸馏在实践中效果显著，尤其是特征模仿（让学生模仿教师的倒数第二层特征 [11]）。然而，这一重要的经验发现缺乏理论基础。基于我们的微调凸性假设，我们为特征模仿建立了一套理论，并研究学习参数的方差。我们证明，与常规微调相比，通过特征模仿，参数的方差会更低。因此，特征模仿能让最终模型更加稳定。据我们所知，它

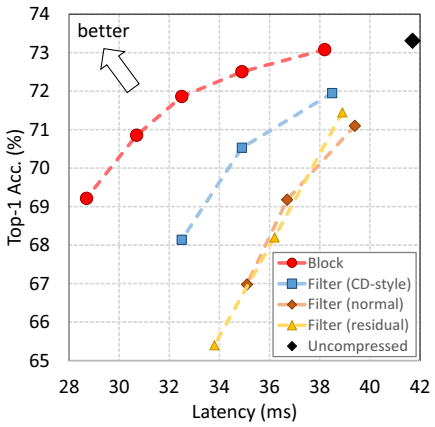


图 1. 使用 500 张训练图像对不同结构压缩方案的比较。就延迟-准确率权衡而言，我们的方案（“块”）显著优于先前方案。ResNet-34 模型在 ImageNet-1k 上进行压缩，并在 NVIDIA TITAN Xp GPU 上进行测试。

表 1 不同压缩方案在 ImageNet-1k 上的 Top-1 验证准确率（%）。使用 50 张训练图像，将 ResNet-34 的延迟降低了 16% 。先前的方法以“常规”方式剪枝滤波器。我们的块级方案“BP（块）”仅移除最前面的几个块，并通过反向传播对剪枝网络进行微调。

知识蒸馏 [15]	FSKD [9]	对比蒸馏 [10]	MiR [11]	BP（块）
44.5	45.3	56.2	64.1	**66.5**

是首个成功阐明特征模仿为何优于常规微调的理论。

下一个关键问题是如何剪枝网络结构。几乎所有先前的少样本压缩工作 [9],[10], [11] 都采用了滤波器级剪枝。本文首次揭示，块级剪枝在少样本压缩中具有根本性的优势。首先，与剪枝滤波器相比，剪枝块拥有更加凸的优化景观，因此所需数据更少。其次，在浮点运算次数相同的情况下，它在实际计算设备上的运行速度快得多。图 1 显示，在延迟与准确率的权衡方面，删除块明显优于以往的压缩方案。表 1 进一步表明，一个简单得令人难以置信的删除块基线方法（即，不做任何其他处理，只进行微调）就已经胜过采用复杂技术的现有方法。该基线方法“BP（块）”仅移除最前面的几个块，并仅使用交叉熵损失对剪枝网络进行微调。

为了实现更好的块剪枝，我们研究了如何仅凭少量训练样本选择要移除的块。已有多种剪枝块的标准 [16], [17], [18] ，但有些 [16], [17] 需要大量数据，另一些 [18]则只评估移除块前后的输出差异。我们注意到，尽管移除某些块会显著改变特征图，但即使训练集很小，也能通过微调轻松恢复，这是因为微调凸性假设。仅测量剪枝后网络与原始网络之间的差异，充其量只能得到次优结果。我们不仅需要考虑损失景观的性质以提高准确率，还需要考虑实际加速比。为了将损失景观的因素纳入考量，本文提出了一个新概念可恢复性，只需少量训练图像即可高效计算。我们的可恢复性度量指标与微调后网络的准确率惊人地一致。结合加速比后，我们最终的剪枝分数显著优于以往的方法。

最后，我们提出 PRACTISE，即“利用少量图像实现实用网络加速”，借助稀缺数据有效加速网络。PRACTISE 的表现显著优于以往的少样本剪枝方法。在延迟降低22.1% 的情况下，PRACTISE 在 ImageNet-1k 上的 Top-1 准确率平均超过此前最先进水平（SOTA）方法*7.0%* （百分点，而非相对提升）。它具有很强的泛化能力，可用于合成图像和域外图像。我们的贡献如下：

- **假设**。 我们提出微调凸性假设，揭示了为何对剪枝模型进行少样本微调不会遭遇过拟合问题。我们指出，凸性与微调的有效性密切相关。

- **理论**。我们为特征模仿建立了一套理论，证明与常规微调相比，可学习参数通过特征模仿能够实现更低的方差。这与经验结果高度吻合。

- **算法**。尽管丢弃块早已为人所知，但我们率先揭示了它在少样本压缩中的巨大潜力，其凸性高于滤波器剪枝。通过考量优化景观的属性（例如凸性和初始准确率），我们提出了一个新概念可恢复性，用于衡量恢复各个块的难度，并确定丢弃块的优先级。随后，我们提出了 PRACTISE，这是一种使用微型图像集加速神经网络的算法。在延迟降低 22.1% 的情况下，PRACTISE 在 ImageNet 上的 Top-1 准确率比此前最先进的方法高出 7.0%。

PRACTISE 算法的初步版本已作为会议论文发表 [19]。

2 相关工作

滤波器级剪枝通过移除卷积层中的滤波器来加速神经网络。研究者提出了多种选择滤波器的准则 [1], [2], [3], [18], [20], [21],，以及不同的训练策略 [22], [23], [24], [25]。然而，这些方法大多依赖完整的训练数据来训练神经网络。在面临数据隐私问题时，这些滤波器级剪枝方法在微型训练集上的延迟-准确率性能较差 [9], [10]。本文认为，滤波器剪枝的缺点在于其优化景观平坦，加速比低。它需要剪除大量参数和浮点运算次数才能降低延迟。这会导致剪枝网络与原始网络之间存在较大的容量差距，因此仅用微型训练集难以恢复剪枝网络的准确率。相比之下，我们主张将块级剪枝用于少样本压缩。丢弃块既能带来更凸的优化景观，也能实现更高的加速比。

块级剪枝会移除网络中的若干个完整块（例如残差块）。已有一些研究针对完整训练数据的情况提出了

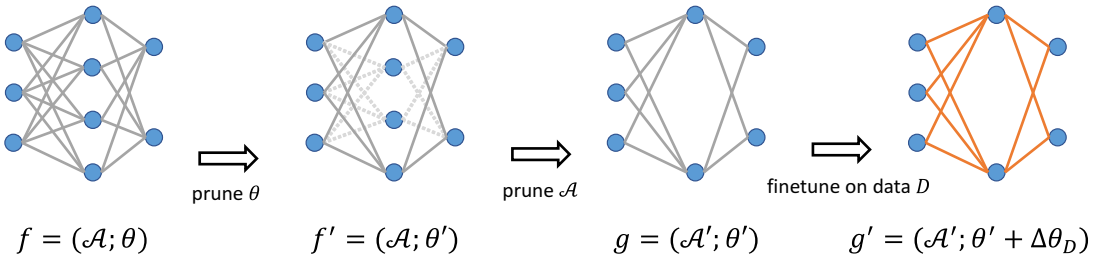


图 2. 传统少样本压缩流程。

相关方法。DBP [16] 提出使用线性探测评估每个块的特征准确率，并丢弃准确率较低的块。ϵ-ResNet则加入稀疏性促进函数，在某个块的所有响应都低于阈值 ϵ 时将其丢弃。DBP 和 ϵ-ResNet 都需要大型数据集进行训练和测试。CURL [18] 使用代理数据集评估块移除前后的 KL 散度变化。然而，它忽略了微调过程。事实上，剪枝网络微调后的准确率至关重要，但目前没有现成的标准能够对其进行准确估计。本文提出一种新的可恢复性，用于评估通过丢弃块进行剪枝后的网络能否充分恢复准确率。

少样本或无数据知识蒸馏旨在利用预训练教师网络，在原始训练数据有限甚至完全缺失的情况下训练学生网络。少样本知识蒸馏 (FSKD) [9] 会在被剪枝的卷积层后插入 1×1 卷积层，并通过使剪枝网络的特征图模仿原始网络的特征图来逐层训练。逐层训练能够从教师网络获取更多监督信息，但会导致误差累积。对比蒸馏 [10] 提出交叉蒸馏，以减少累积误差。MiR [11] 提出先模仿后替换框架，以整体优化剪枝网络。当原始训练数据中甚至没有一个样本可用时，无数据知识蒸馏 (DFKD) [26]会合成原始训练数据的替代数据。后续的 DFKD 方法通过多教师 [28]加速合成过程 [27]并提升性能。剪枝和量化 [29], [30], [31] 是 DFKD 的两大主要应用。然而，大多数 FSKD 和 DFKD 方法采用滤波器级剪枝，导致延迟-准确率表现不佳。我们的 PRACTISE 在合成图像和域外图像上均表现稳健，并在无数据场景下大幅提升准确率。

3 微调凸性假设

针对少样本压缩，已有多种有效算法 [9],[10], [11] 。然而，这些算法是基于经验设计的，缺乏理论分析。关键问题是：为什么仅用少量训练样本微调网络，也不会出现严重的过拟合？本节研究参数对应的损失景观。令人惊讶的是，我们发现，使用少量样本进行微调似乎表现得像一个凸问题，从而缓解了过拟合问题。我们观察到，不同压缩方案的优化景观凸性各不相同。剪枝块具有更高的凸性，因此比剪枝滤波器所需的训练数据更少。

如图2所示，我们将传统的少样本压缩归纳为三个步骤。首先，将一些参数设为零，从而对其进行剪枝。对于移除一个块的操作，我们会剪去残差连接内的所有参数。对于剪枝滤波器，则将被剪枝的参数设为零。其次，根据这些零参数对网络架构进行剪枝，同时不改变网络功能。第三，在数据 D上对压缩网络进行微调。

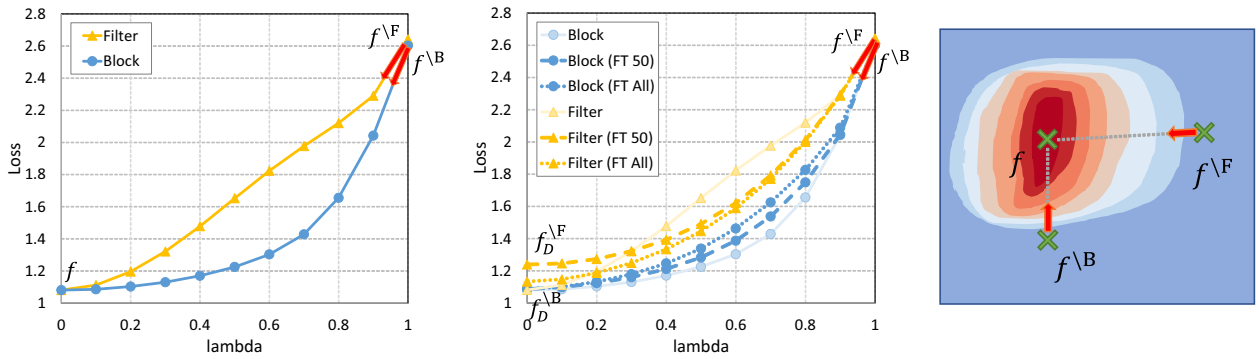
继 [12], [13], [14],之后，我们通过对两个模型进行插值来衡量凸性。考虑图2中的第一步 $(\mathcal{A};\ \theta) \to (\mathcal{A};\ \theta')$，我们简单地将一些参数设为零，同时保持架构不变。因此，可以通过线性插值合并原始模型和剪枝模型。给定 $\lambda \in [0,1]$，插值模型为

$$f_\lambda = \lambda f' + (1-\lambda) f \tag{1}$$
$$\iff (\mathcal{A}; \theta_\lambda) = (\mathcal{A}; \lambda\theta' + (1-\lambda)\theta)\,. \tag{2}$$

我们将 ResNet-34 作为原始模型（f），并分别通过移除滤波器（$f^{\setminus F}$）和块（$f^{\setminus B}$）对其进行剪枝。图3a展示了插值模型的测试损失。损失曲线大致描述了损失景观的性质。令人惊讶的是，这些曲线平滑且单调。更重要的是，它们大致呈凸性。与剪枝滤波器相比，剪枝块实现了更加凸的曲线。图3c定性地展示了这一观察结果。与路径 $f \to f^{\setminus F}$相比，路径 $f \to f^{\setminus B}$更加凸。基于这些观察，我们提出以下猜想。

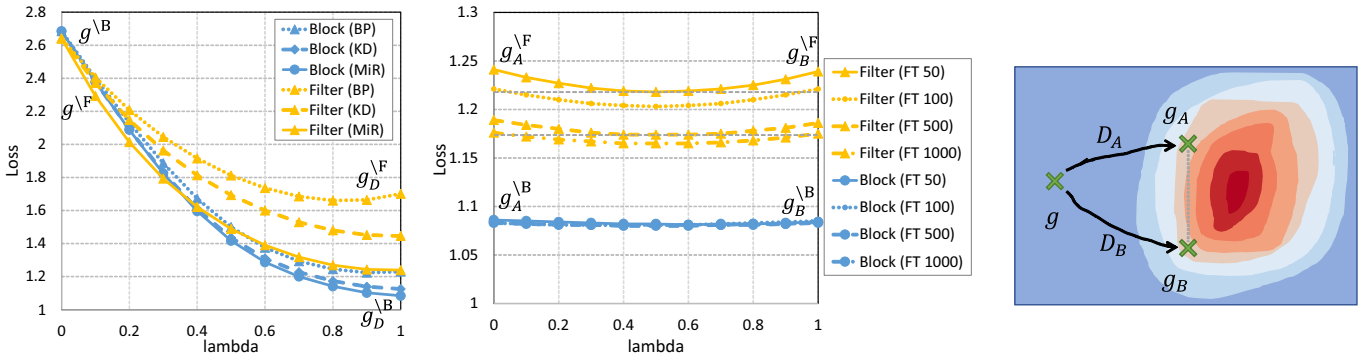
猜想***1***。通过在原始模型与剪枝模型之间进行线性插值，损失景观的凸性表明了微调的难度，这反映了所需训练数据的数量、收敛速度等。

为验证这一猜想，我们在训练集 D 上分别微调 $f^{\setminus B}$ 和 $f^{\setminus F}$，得到微调模型 $f_D^{\setminus B}$ 和 $f_D^{\setminus F}$。然后，我们评估剪枝模型与微调模型之间的插值。图3b展示了结果。首先，在所有情况下，损失景观更凸时，$f_D^{\setminus B}$ 的损失都低于 $f_D^{\setminus F}$。直观而言，损失景观越凸，随机梯度下降就越容易快速优化模型。其次，对于 $\lambda \in [0.9, 1]$，虚线与实线重合，这是因为插值公式 $\theta_\lambda = \lambda\theta^{\setminus B} + (1-\lambda)\ \theta$ 可视为随机梯度下降的更新公式，其中 θ 是最优梯度方向，$1-\lambda$ 是较小的学习率。这进一步基于线性插值与随机梯度下降微调之间的密切关系，证明了猜想1的有效性。



(a) (b) (c)

图 3。（a）展示了原始模型 f（$\lambda = 0$）与剪枝模型 f'（$\lambda = 1$）之间插值模型的损失。随后，我们在包含 50 张（“FT 50”）或全部（“FT All”）训练图像的数据集 D 上微调剪枝模型 f'，得到微调模型 f'_D。（b）展示了这两个模型之间插值模型的损失。（c）定性展示了损失景观。微调剪枝模型时，损失景观关于模型参数呈凸性。与剪枝滤波器相比，剪枝块的损失景观凸性更强。



(a) (b) (c)

图 4。（a）展示了采用不同微调方法时插值模型的损失。所有模型均使用 50 个样本进行微调。对于（b），我们分别在两个数据集 A 和 B 上使用相同数量的样本对模型进行微调，然后评估插值模型的损失。（c）展示了微调剪枝模型的损失景观。

接下来，我们考虑少样本压缩中的其余步骤，在这些步骤中，架构将被剪枝。假设 $f^{\setminus B}$ 和 $f^{\setminus F}$ 经过架构剪枝后分别变为 $g^{\setminus B}$ 和 $g^{\setminus F}$。我们在少量数据 D 上微调这些剪枝模型，得到 $g_D^{\setminus B}$ 和 $g_D^{\setminus F}$。图4a展示了线性插值的结果。不同微调方法的所有曲线均呈凸形；在显著降低损失方面，剪枝块优于剪枝滤波器。基于图3b和图4a，我们提出以下猜想。

猜想 *2.* [微调凸性假设（弱）]: 剪枝模型的微调损失景观关于参数是凸的。这意味着

$$\mathcal{L}(g'_{\triangle_D}) \leq \lambda\mathcal{L}(g) + (1-\lambda)\mathcal{L}(g'_D)\,, \tag{3}$$

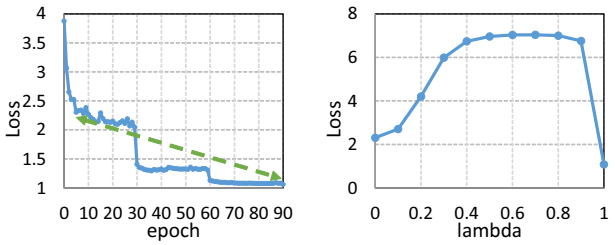
其中 $g'_{\triangle_D}$ 是一个插值模型：

$$g'_{\triangle_D} = (\mathcal{A}'; \theta' + (1-\lambda)\Delta\theta_D)\,, \tag{4}$$

$$g = (\mathcal{A}'; \theta')\,, \tag{5}$$

$$g'_D = (\mathcal{A}'; \theta' + \Delta\theta_D)\,. \tag{6}$$

这一猜想回答了为何仅凭少量训练样本就能够对整个网络进行微调。借助凸性，即使只有极其有限的训练数据，优化也几乎不会受到过拟合的影响。请注意



(a) (b)

图5。（a）展示了剪枝模型 $g^{\setminus B}$ 在整个 ImageNet-1k 训练集上的测试损失曲线。（b）展示了第 5 个训练轮次模型（$\lambda = 0$）与最后一个训练轮次模型（$\lambda = 1$）之间的插值模型的损失，即（a）中绿色箭头所示的模型。

，这一结论并不适用于使用全部训练数据从头训练整个网络。正如先前工作 [12],[13], [14], 所介绍的，在原始模型与微调模型之间进行线性插值时存在损失屏障。我们还通过从头训练一个剪枝网络对其进行了评估。图 5 展示了结果。从头训练会面临复杂的优化景观。线性插值无法实现凸损失曲线。

推论2仅考虑模型g 与其微调版本 g'之间的联系。我们进一步使用不同的数据集 A 和 B 对剪枝模型 g 进行微调，以研究

不同微调版本之间的关系。图4b展示了 g'_A 与 g'_B之间线性插值的结果。使用不同的微型数据集时，微调模型位于同一盆地中，插值过程中不存在损失屏障。图4c以定性的方式展示了这一观察结果。因此，我们提出以下强猜想：

猜想 ***3***。 **[微调凸性假设（强）]:** 对于少样本训练数据集 A 和 B，微调解位于同一个优化盆地中，且泛函 $\mathcal{L}$ 关于参数 θ 是凸的。这意味着

$$\mathcal{L}(g'_{A\hat{+}B}) \leq \lambda\mathcal{L}(g'_A) + (1-\lambda)\mathcal{L}(g'_B)\,, \tag{7}$$

其中

$$g'_{A\hat{+}B} = (\mathcal{A}; \theta' + \lambda\Delta\theta_A + (1-\lambda)\Delta\theta_B)\,, \tag{8}$$
$$g'_A = (\mathcal{A}; \theta' + \Delta\theta_A)\,, \tag{9}$$
$$g'_B = (\mathcal{A}; \theta' + \Delta\theta_B)\,. \tag{10}$$

我们将损失泄漏定义为

$$\mathcal{L}_{leak}(g'_{A\leftrightarrow B}) = \max_{\lambda} \lambda\mathcal{L}(g'_A) + (1-\lambda)\mathcal{L}(g'_B) - \mathcal{L}(g'_\lambda)\,. \tag{11}$$

如果训练数据从 A 增加到 $A \cup B$，损失的减少量与损失泄漏成正比。即

$$\mathcal{L}(g'_A) - \mathcal{L}(g'_{A\cup B}) \propto \mathcal{L}_{leak}(g'_{A\leftrightarrow B})\,. \tag{12}$$

我们的损失泄漏反映了剪枝网络对训练数据的渴求。损失泄漏较大时，剪枝网络需要更多数据进行微调。如图 4b 所示，与剪枝块相比，剪枝滤波器需要更多数据才能降低损失。具体而言，对于各含有 50 张图像的 A 和 B，线性插值模型（$\lambda = 0.5$）取得的损失与使用 100 张图像微调的模型相近。换言之，分别在每个数据集上微调后再对模型求均值，其效果与在两个数据集的并集上微调模型相当。这一观察结果支持我们的推论2，并进一步支持我们的推论2。请注意，通过令数据集为空 A，可以将猜想 3视为推论2的扩展。

总之，我们已经揭开了本节开头提出的谜团。得益于优化景观的凸性，使用少量训练样本进行微调成为可能，且不会出现严重的过拟合。与剪枝滤波器相比，剪枝块的损失景观具有更强的凸性。因此，它所需的训练数据更少，并且收敛速度更快。接下来，我们将基于凸性假设，为不同的微调方法提出一个理论。

4 微调理论

如图 4a 所示，在实践中，不同的微调方法会带来不同的最终准确率。朴素方法（“BP”）仅使用交叉熵损失，正如先前工作所指出的那样，这会导致较低的准确率 [9], [10]。知识蒸馏 [15]（KD），尤其是特征级蒸馏 [32],

[33], 可以提高准确率。MiR [11] 提出让剪枝模型模仿全局平均池化之前的特征，并且仅用少量训练样本就取得了最先进水平的性能。需要指出的是，所有这些算法都是凭经验设计的，没有理论能够解释

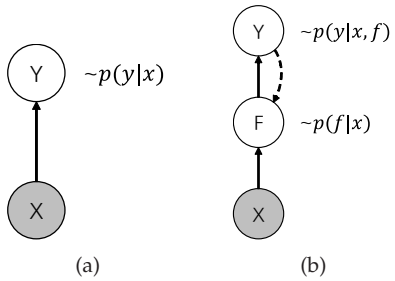


图 6。（a）和（b）分别展示了原始判别模型和特征模仿模型。

为什么特征模仿方法 MiR 优于其他方法。本节提出一种理论，用于解释现有小样本蒸馏方法的行为，并表明 MiR 的损失函数在这些方法中是最优的。此外，我们还表明，与优化交叉熵相比，特征模仿能够获得更稳定的模型。

我们将传统分类表述为图 6a 中的判别模型。给定输入 x，该模型估计条件概率分布 $p(y|x)$。在知识蒸馏中，$p(y|x)$ 可视为教师的分布，而学生分布为 $q_\theta(y|x)$，其中 θ 表示参数。采用 KL散度来衡量这两个分布之间的距离：

$$KL\left[p(y|x)||q_\theta(y|x)\right]\,. \tag{13}$$

为简化起见，我们只考虑单变量情形，但很自然地可以将分析扩展到多变量情形。

关于中间特征的更多信息将在特征模仿中介绍，如图 6b 所示。除了教师的预测 $p(y|x,f)$外，我们还知道特征分布 $p(f|x)$ 。现在，目标是求二者 f 和 y 的联合分布：

$$KL\left[p(y,f|x)||q_\theta(y,f|x)\right]\,, \tag{14}$$

其中，p 和 q 分别表示教师和学生。直观地说，一旦我们成功优化了这一目标，就可以假设 $q_\theta(y,f|x)$ 趋近于 $p(y, f|x)$，记作 $q_\theta(y,f|x) \to p(y,f|x)$。因此，

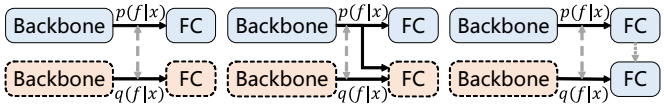
$$q_\theta(y|x) = \int q_\theta(y,f|x)df \to \int p(y,f|x)df = p(y|x)\,, \tag{15}$$

这意味着我们也成功优化了公式 13，并获得了用于分类的判别模型。下面的命题建立了公式 13 与公式 14 之间的严格联系。证明见附录 A。

命题 ***1***。公式 13 的上界为公式 14，即，

$$KL\left[p(y|x)||q_\theta(y|x)\right] \leq KL\left[p(y,f|x)||q_\theta(y,f|x)\right]\,. \tag{16}$$

命题 1 告诉我们，优化公式 14 有助于获得分类器。在本节余下部分，我们将说明，优化公式 14 比优化公式 13 更容易，而且得到的模型更稳定。首先，我们通过以下命题介绍如何优化公式 14，其证明见附录 B。



(a) 原始知识蒸馏 (b) 交叉蒸馏 (c) MiR

图 7. 知识蒸馏中优化分类器（“FC”）的不同方式。带实线边界的蓝色模块表示教师模块，而带虚线边界的橙色模块表示学生模块。

命题 ***2***。公式 14 可以分解为分类损失和特征模仿损失，形式如下

$$KL\left[p(y,f|x)||q_\theta(y,f|x)\right] = C + \mathop{\mathbb{E}}_{f\sim p(f|x)}\left\{\underbrace{-\ln q_\theta(f|x)}_{\text{Feature Mimicking}} + \underbrace{KL\left[p(y|f,x)||q_\theta(y|f,x)\right]}_{\text{Classification}}\right\}, \tag{17}$$

其中 C 是一个常数。假设 $q_\theta(f|x)$ 服从均值为 $\mu_\theta(x)$、标准差为 $\frac{1}{\sqrt{2\beta}}$ 的高斯分布，则

$$\begin{aligned}&\mathbb{E}_{f\sim p(f|x)}[-\ln q_\theta(f|x)]\\ =&\mathbb{E}_{f\sim p(f|x)}\left[\beta\left(f-\mu_\theta(x)\right)^2\right] + C'.\end{aligned} \tag{18}$$

也就是说，在某些合理假设下，对特征模仿部分使用 L2 损失是最优的。在实践中，这种损失通常被用作基于特征的蒸馏损失。β 随后是一个用于平衡特征模仿与分类的超参数，而 $\mu(x)$ 则是我们的学生骨干网络，它根据输入 x生成特征。

对于分类损失 $\mathbb{E}_{f\sim p(f|x)}KL\left[p(y|f,x)||q_\theta(y|f,x)\right]$，我们注意到，由教师 $p(f|x)$ 生成的 f出现在 $q_\theta(y|f,x)$ 中。因此，学生可以在训练期间利用教师特征。我们总结了实现这一损失的三种方式，如图 7 所示。

1）忽略教师的 f，并根据输入 x直接对 y进行建模（如图 7a 所示）。这是先前工作 [9],[15], [32], [33]中广泛采用的传统方式。在没有教师特征帮助的情况下，模型只能取得较低的准确率。2）其次是交叉蒸馏（如图 7b 所示）。学生分类器利用教师特征来降低分类损失。交叉蒸馏优于普通蒸馏。这种方法曾用于 CD [10] 和嫁接 [34] 方法中。3）直接使用教师分类器（如图 7c 所示）。由于我们已经知道 $p(y|f)$（即教师分类器或“FC”），因此可以直接将其用于 $q_\theta(y|f)$。这是最优解，使 $\mathbb{E}_{f\sim p(f|x)}KL\left[p(y|f,x)||q_\theta(y|f,x)\right]=0$。这种方法曾用于 MiR [11]，本文也将其用作微调方法。

4.1 模型参数方差

接下来，我们分析公式 14 为什么优于公式 13。我们的主要结论是：特征模仿（公式 14）能够降低模型参数的方差。换言之，它只需更少的训练数据，就能获得更准确的参数估计。我们从以下命题开始（证明见附录 C）。

命题 ***3.*** 给定 n 个训练样本 $x_1, x_2, \cdots, x_n$，假设微调的损失函数遵循规范形式

$$\mathcal{L} = -\ln q_\theta(f|x). \tag{19}$$

那么近似有

$$\hat{\theta} \sim \mathcal{N}(\tilde{\theta}, \frac{1}{n\mathcal{I}(\tilde{\theta})}), \tag{20}$$

其中，θ 表示教师参数，并且

$$\mathcal{I}(\theta) = \mathbb{E}_{x\sim p(x)}\int\left(\frac{\partial q_\theta(f|x)}{\partial\theta}\right)^2\frac{1}{q_\theta(f|x)}df. \tag{21}$$

命题 3 给出了方差的估计。根据公式 20，$\hat{\theta}$ 的方差计算为 $\frac{1}{n\mathcal{I}(\theta_t)}$。下文将分别在断言 4 和断言 5 中计算特征模仿和分类对应的这一项。证明分别见附录 D 和附录 E。

断言 ***4.*** 如果通过特征模仿进行网络微调，且使用公式 18 中的损失，并假设输出特征之前的最后一层是线性的（即卷积层）：

$$\mu_{w,b}(x) = w\cdot x + b, \tag{22}$$

那么根据命题 3，有

$$\hat{w} \sim \mathcal{N}(\tilde{w}, \frac{1}{2n\beta\mathbb{E}[x^2]}), \tag{23}$$

$$\hat{b} \sim \mathcal{N}(\tilde{b}, \frac{1}{2n\beta}). \tag{24}$$

声明 ***5.*** 如果通过优化公式 13 中的损失对神经网络进行微调，并假设最后一层为

$$q_{\boldsymbol{w},\boldsymbol{b}}(\boldsymbol{y}|f) = \sigma(\boldsymbol{w}\cdot f + \boldsymbol{b}), \tag{25}$$

其中 σ 是 softmax 操作

$$\sigma(\boldsymbol{x})_i = \frac{\exp(x_i)}{\sum_{j=1}^{C}\exp(x_j)}, \tag{26}$$

那么根据命题 3，对于 i中 $\boldsymbol{w}$ 和 $\boldsymbol{b}$的第 i个条目，我们有

$$\hat{w}_i \sim \mathcal{N}(\tilde{w}_i, \frac{1}{n\epsilon_i\mathbb{E}[f^2]}), \tag{27}$$

$$\hat{b}_i \sim \mathcal{N}(\tilde{b}_i, \frac{1}{n\epsilon_i}), \tag{28}$$

其中，ϵ_i 是 $\boldsymbol{\epsilon}$ 的第 i个条目，并且

$$\boldsymbol{\epsilon} = q_{\boldsymbol{w},\boldsymbol{b}}(\boldsymbol{y}|f)[1-q_{\boldsymbol{w},\boldsymbol{b}}(\boldsymbol{y}|f)]. \tag{29}$$

根据命题 4 和声明 5，特征模仿和分类的权重参数方差分别为 $\frac{1}{2n\beta\mathbb{E}[x^2]}$ 和 $\frac{1}{n\epsilon\mathbb{E}[f^2]}$。由于神经网络中存在归一化层，我们可以假设 $\mathbb{E}[x^2]$ 和 $\mathbb{E}[f^2]$ 均有界于某个常数。如果有大规模训练集可用，由于 n较大，这两个方差都很小。

但是，如果 n 较小，分类的方差（声明 5）就会很大，因为 $\epsilon\to 0$ 在经验上约为 10^{-3}。对于特征模仿，β 的典型值为1 2，因此方差（断言 4）仍然很小。这意味着在少样本场景中，特征模仿比分类更稳定。

最后，我们的理论也解释了传统的知识蒸馏 [15]，它在 softmax 运算中采用温度 T：

$$q_{\boldsymbol{w},\boldsymbol{b}}(\boldsymbol{y}|f) = \sigma(\frac{\boldsymbol{w}\cdot f + \boldsymbol{b}}{T}) . \tag{30}$$

如果 $T > 1$，就会使预测更加平缓，并增大 ϵ。这解释了为什么知识蒸馏对少样本压缩非常有帮助。

简而言之，我们的理论表明，与优化交叉熵损失相比，特征模仿能使模型参数方差更低。这解释了为什么特征模仿是一种更优的微调方法。

5 PRACTISE 算法

现在，我们继续提出一种与我们的假设和理论相契合的少样本压缩新算法。首先，我们分析丢弃块（例如 ResNet 中的残差块）的优势。与以往的少样本压缩方案相比，丢弃块具有更凸的优化景观，并且加速比更高，因此即使使用极少量训练集，也能取得更优的延迟-准确率性能。其次，我们提出一种新的可恢复性准则，用于选择要丢弃的块。与以往的准则不同，可恢复性衡量的是微调剪枝网络的难度；它不仅与模型准确率密切相关、关系更直接，而且计算效率高，也非常适合少样本场景。基于可恢复性，我们提出 PRACTISE，这是一种利用极少量训练集实现实用网络加速的算法。PRACTISE 不需要图像标签，甚至可以在无数据设置下运行。

5.1 丢弃块的动机

在真实应用中加速神经网络时，延迟和准确率是最重要的两个度量指标。延迟更低（而非浮点运算次数更少）意味着模型在实际运行中更快。为了用小型训练集加速模型，人们提出了许多压缩方案。图 8 对比了这些方案的加速比。FSKD [9] 根据 L_1 范数（“normal”）剪除残差块中的滤波器。CD [10] 提出只剪除浅层中的卷积层，并保持较深层不变（“CD-style”）。MiR [11]裁剪残差连接（“residual”）。所有这些剪枝方案的加速比都不理想。如图 8 所示，浮点运算次数降低约30% 时，压缩模型的延迟仅降低 16.1%（41.7→35 毫秒）。简单地调整输入图像的分辨率（即“Resolution”）就能实现更好的加速。最后，丢弃块优于所有这些方法。要达到 35 ms 的延迟，它只需减少 12.7% 的浮点运算次数，显著低于剪枝滤波器中的30% 。

我们同样关注微调模型的准确率。如第 3 节所述，与剪枝滤波器相比，剪枝块具有更加凸的优化景观，因此所需的训练样本更少（参见图 4）。

显然，相比剪枝滤波器，丢弃块对模型加速更有效，但在少样本压缩中却常被忽视。一个可能的原因是，剪枝滤波器在完整训练数据集上已经取得了卓越的性能。然而，当只有极少量训练集可用时，我们应更加重视剪枝效率

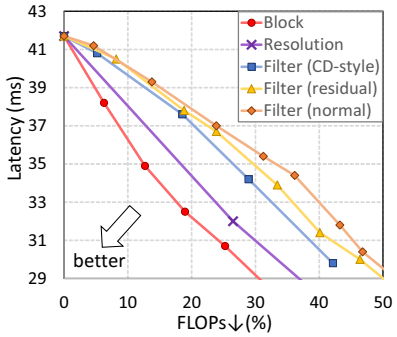


图8。延迟降低与浮点运算次数降低之间的关系。与其他压缩方案相比，丢弃块具有最高的加速比。

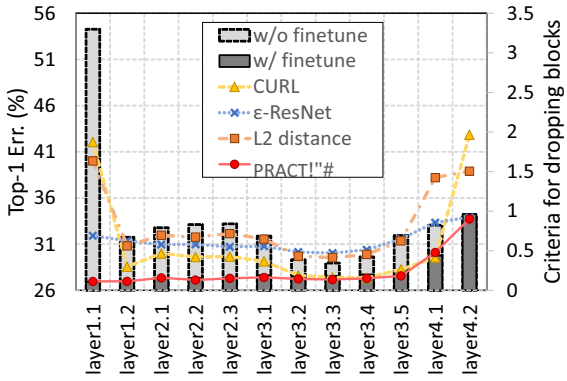


图9。每个块的可恢复性以及丢弃块的不同标准示意图。ResNet-34 有 12 个可以丢弃的块（从“layer1.1”到“layer4.2”）。我们每次丢弃一个块，并评估微调前后的 Top-1错误率。我们为每个块计算不同标准的分数，并在该图中以不同曲线呈现。请注意，Top-1错误率（%）在 ImageNet-1k 验证集（50000 张图像）上进行评估，而微调和标准评估仅使用 500 张训练图像。我们的 PRACTISE 几乎完美地预测了微调网络的错误率。

地进行考量。现有工作 [9], [11]能够在大幅降低浮点运算次数的同时剪枝滤波器，但会导致优化景观不够凸，因此需要更多训练数据。另一方面，丢弃块具有较高的加速比，并且能够保留原始模型更多的参数和容量。由于损失景观更加凸，它只需较少的数据进行微调，便能在极少量训练集上实现更优的延迟-准确率权衡（参见图1）。我们已经证明，即使是朴素的丢弃块方法，也已经超越了大多数现有方法。

5.2 剪枝模型的可恢复性

为了进一步改进丢弃块的方法，我们研究了如何选择要丢弃的块。现有的标准在少样本场景中表现不佳，因为它们要么需要大型训练集 [16], [17] ，要么无法预测微调准确率 [18]。我们认为，有效的度量指标应与微调准确率保持一致。但正如第 3 节所讨论的，它与损失景观的凸性、初始准确率等因素存在复杂的关系。此外，图2中的传统剪枝步骤会改变模型的优化景观。因此，

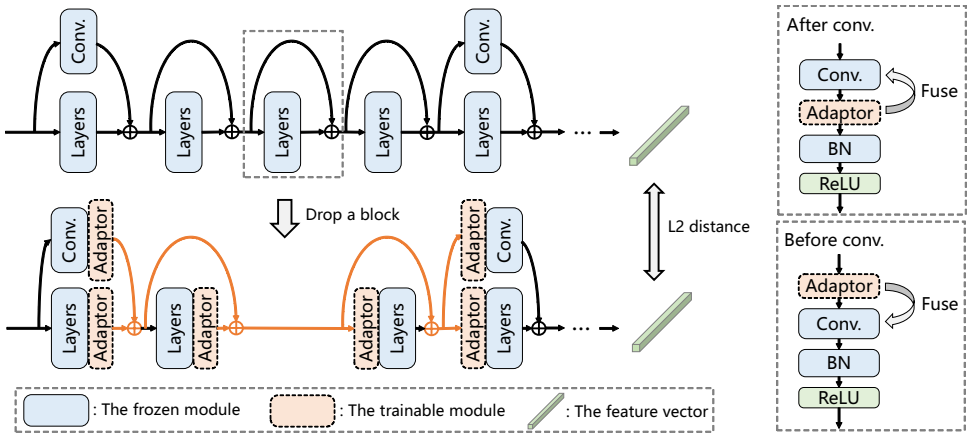


图10。我们用于确定应移除哪个块的方法示意图。移除一个块，并在被移除位置周围插入适配器，以从移除该块造成的影响中恢复。随后，计算剪枝网络与原始网络的特征之间的 L2距离，将其作为该被移除块的可恢复性。最后，我们移除若干易于恢复的块，从而获得剪枝网络。

仅评估 $f \leftrightarrow f'$ 的属性来表明 $g \leftrightarrow g'$ 的微调难度并不理想。相反，我们将提出一种名为 PRACTISE 的算法，以高效评估微调难度。简而言之，PRACTISE 会在剪枝结构周围插入适配器层，并在其他部分保持固定的情况下优化适配器。令人惊讶的是，对于少样本微调，仅训练适配器就能达到几乎与端到端训练所有参数相同的优化特性。

首先，我们提出了一个称为可恢复性的新概念，用于衡量剪枝模型恢复准确率的能力。如图9所示，我们以“layer1.1”块为例。直接移除该块会导致 54.3% 的 Top-1错误率。移除块前后的 KL散度（“CURL”）和“L2距离”变化都很大，表明模型的输出确实发生了显著变化。然而，即使仅使用极小的训练集进行微调，也能非常有效地恢复准确率。现有指标忽略了微调，但真正直接决定压缩质量的是微调准确率。为克服这一困难，我们提出了一种评估每个被移除块可恢复性的方法。如图9所示，我们的方法（“PRACTISE”）与微调模型的 Top-1错误率具有高度一致性。

图10展示了我们的方法。给定原始模型$\mathcal{M}_O$，通过移除一个块 $\mathcal{B}_i$得到剪枝模型 $\mathcal{M}_{P(\mathcal{B}_i)}$ 。我们在与该块相连的位置插入适配器。这些适配器是卷积层，卷积核大小为 1×1，并根据位置不同放置在原始卷积层之前或之后。对于位于被移除块之前的块，适配器插入在卷积层之后；反之，则插入在卷积层之前。由于卷积是线性的，因此所有适配器都可以融合到相邻的卷积层中，同时保持网络功能不变。也就是说，这些适配器不会给剪枝模型带来额外开销。我们使用小型训练集优化这些适配器，以最小化原始模型与剪枝模型特征之间的距离（即特征模仿）。可恢复性计算如下：

$$\mathcal{R}(\mathcal{B}_i) = \min_{\alpha} \mathbb{E}_{x\sim p(x)} \|\mathcal{M}_O(x;\theta) - \mathcal{M}_{P(\mathcal{B}_i)}(x;\theta \setminus b_i, \alpha)\|_F^2 , \tag{31}$$

其中， θ 表示原始模型中的参数， $\setminus b_i$表示不包括被移除块 $\mathcal{B}_i$中的参数， α 表示适配器中的参数。如图9所示，该度量指标能够很好地估计每个块的可恢复性。

如图 11 所示，其成功与我们的假设和理论高度一致。比较图 11a 和 11b，我们得出结论：剪枝架构（图2中的第二步）改变了损失景观。相反，仅优化适配器（$\mathcal{M}_{P(B)+A'} \leftrightarrow \mathcal{M}_{P(B)+A}$）与微调整个模型（$\mathcal{M}'_{P(B)} \leftrightarrow \mathcal{M}_{P(B)}$）的效果几乎相同。如图 11d 所示，微调模型 $\mathcal{M}'_{P(B)}$和 $\mathcal{M}_{P(B)+A'}$甚至位于同一个优化盆地中，且不存在明显的损失屏障。换句话说，仅优化适配器与端到端优化几乎相同。这些结果提供了有力的实证证据，表明 PRACTISE 确实能够精确地找到可恢复性。在图 11c 中，移除“layer1.1”会得到最凸的曲线，这意味着只需少量样本即可轻松恢复准确率。相比之下，“layer4.2”的曲线较为平坦。在初始准确率较低的情况下，其性能表现不佳。

图 12 进一步考察了丢弃更多块时的结果。结果表明，这些适配器能够弥补大部分准确率损失，并在少样本场景下达到与微调所有参数相当的准确率。

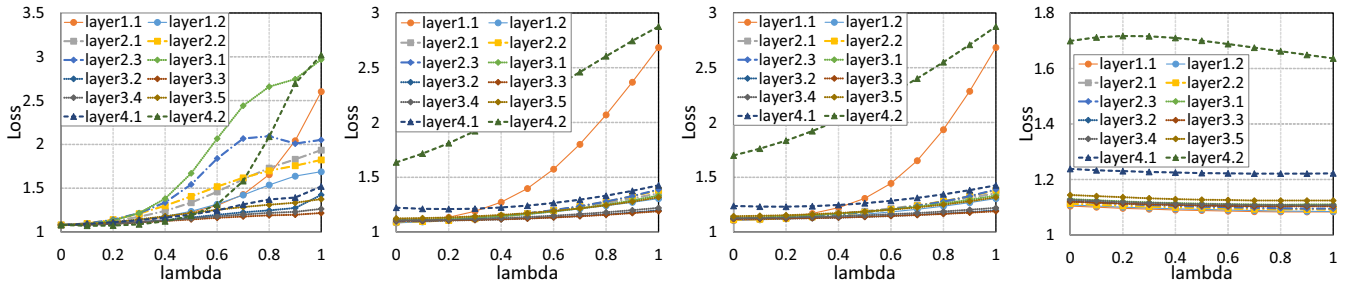
我们的方法也很高效。得益于适配器所需的计算量和参数有限，根据公式 31 计算可恢复性只需少量训练样本和很短的训练时间。

丢弃块时还必须考虑的另一个因素是不同块的延迟。在可恢复性相同的情况下，延迟较高的块应优先丢弃。为此，我们计算块 $\mathcal{B}_i$ 的加速比，公式如下

$$\tau(\mathcal{B}_i) = \frac{lat_{\mathcal{M}_O} - lat_{\mathcal{M}_{P(\mathcal{B}_i)}}}{lat_{\mathcal{M}_O}} , \tag{32}$$

其中， lat 表示延迟。最后，我们将每个块 $\mathcal{B}_i$ 的剪枝分数定义为

$$s(\mathcal{B}_i) = \frac{\mathcal{R}(\mathcal{B}_i)}{\tau(\mathcal{B}_i)} . \tag{33}$$



(a) $\mathcal{M}_O \leftrightarrow \mathcal{M}_{P(B)}$ (b) $\mathcal{M}'_{P(B)} \leftrightarrow \mathcal{M}_{P(B)}$ (c) $\mathcal{M}_{P(B)+A'} \leftrightarrow \mathcal{M}_{P(B)+A}$ (d) $\mathcal{M}_{P(B)+A'} \leftrightarrow \mathcal{M}'_{P(B)}$

图 11. 不同剪枝模型之间插值模型的损失。每条曲线表示剪除一个块。给定原始模型 $\mathcal{M}_O$，通过移除块 B 得到剪枝模型 $\mathcal{M}_{P(B)}$，然后在一个微型数据集上微调得到 $\mathcal{M}'_{P(B)}$。(a) 和 (b) 对比了这些模型的损失景观。由于架构发生变化，损失景观也随之显著改变，正如 (a) 和 (b) 所示，两者存在显著差异。注意，$\mathcal{M}'_{P(B)}$ 是期望的目标。通过插入适配器得到模型 $\mathcal{M}_{P(B)+A}$ 后，我们的 PRACTISE 只训练这些适配器即可实现 $\mathcal{M}_{P(B)+A'}$。优化适配器几乎可以完美实现期望的目标曲线——(b) 和 (c) 精确对齐。特别是，(d) 揭示了这两种训练方法得到的微调模型位于同一个优化盆地中，且不存在明显的损失屏障。

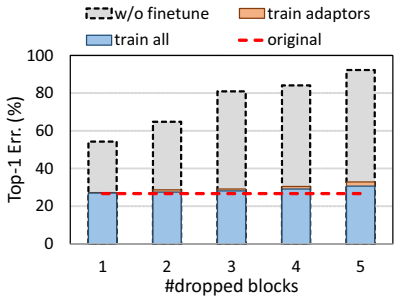


图 12. 全量训练与仅训练适配器的比较。通过移除不同数量的块对 ResNet-34 进行剪枝，且仅使用 500 个训练样本。Top-1错误率（%）在 ImageNet-1k 验证集上评估。注意，原始模型的 Top-1错误率为 26.98%。

我们的剪枝分数同时考虑了每个块的可恢复性和延迟。分数越低，表示该块越优先被移除。我们计算每个块的剪枝分数，并移除分数最低的前 k 个块。

5.3 恢复剪枝模型的准确率

确定剪枝模型的结构后，最后一个问题是如何恢复准确率。第 4 节已证明 MiR 的优势。因此，我们遵循 MiR，通过最小化以下目标对剪枝模型进行微调：

$$\mathcal{L} = \|\mathcal{M}_O(x;\theta_O) - \mathcal{M}_P(x;\theta_P)\|_F^2, \tag{34}$$

其中，θ_O 表示原始模型中被冻结的参数，θ_P 表示剪枝模型中可训练的参数。

PRACTISE 的完整算法见算法 1。我们的 PRACTISE 无需 额外超参数。通过特征模仿，我们能够在不 使用训练标签的情况下加速模型。

PRACTISE 甚至可以在无数据场景下发挥作用。一种做法是采用 DFKD [26]生成的合成图像作为训练图像。大多数现有的 DFKD 工作 [26], [28]采用滤波器剪枝，而 PRACTISE 在延迟-准确率表现上取得了显著提升。这在无数据场景下很有帮助。另一种做法是收集域外数据。另一方面，使用大量域外图像时，剪枝后网络的准确率甚至接近使用原始训练图像时的准确率。这证明了 PRACTISE 具有很强的泛化能力。

算法 1： PRACTISE

输入： 原始模型 $\mathcal{M}_O$，
被丢弃的块数量 k，微量训练数据 $\mathcal{D}_\mathcal{T}$
测试 $\mathcal{M}_O$的延迟；
对于每个块 $\mathcal{B}_i$ **执行**
  丢弃 $\mathcal{B}_i$ 以获得剪枝模型 $\mathcal{M}_{P(\mathcal{B}_i)}$；
  测试 $\mathcal{M}_{P(\mathcal{B}_i)}$的延迟，并求得 $\tau(\mathcal{B}_i)$（式 32）；
  插入适配器；
  计算 $\mathcal{R}(\mathcal{B}_i)$，使用 $\mathcal{D}_\mathcal{T}$（公式 31）；
  计算分数 $s(\mathcal{B}_i)$（公式 33）；
  加回 $\mathcal{B}_i$ 并移除所有适配器；
选择分数最小的前 k 个块；
丢弃这些 k 个块，以获得 $\mathcal{M}_P$；
通过最小化 $\mathcal{L}$（公式34），使用 $\mathcal{D}_\mathcal{T}$对 $\mathcal{M}_P$ 进行微调；
返回剪枝模型 $\mathcal{M}_P$

6 实验结果

本节将介绍 PRACTISE 在少样本压缩和无数据压缩方面的结果。遵循先前工作 [9],[10], [11], ，我们将在 ImageNet-1k [37] 的小型训练集上对 ResNet-34 [35] 和 MobileNetV2 [36] 进行剪枝。随后，为了测试 PRACTISE 的泛化能力，我们将分别使用合成图像和域外数据对 ResNet-50 进行剪枝。最后，我们将开展消融研究。

实现细节。如算法1所示，PRAC-TISE 需要计算每个剪枝模型的延迟和可恢复性$\mathcal{M}_{P(\mathcal{B}_i)}$，然后对剪枝模型 $\mathcal{M}_P$进行微调。本文仅考虑移除输入和输出维度相同的块。对于延迟，我们使用 $64 \times 3 \times 224 \times 224$ 输入对模型进行500次测试，并记录平均延迟。所有延迟数据均在同一台配备 NVIDIA TITAN Xp GPU 的计算机上测得。为计算可恢复性，我们使用随机梯度下降，以批量大小64 优化适配器，共迭代1000次。初始学习率为0。02，每经过40%的迭代次数便缩小为原来的十分之一。当训练数据的大小 $\mathcal{D}_\mathcal{T}$小于

表 2 剪枝 ResNet-34 在 ImageNet-1k 上的 Top-1/Top-5 验证准确率（%）。所有模型均降低了16% 延迟，使用 50、100、500 和 1000 个训练样本。以往方法会剪除残差块内的滤波器。原始方法 ResNet-34 的 Top-1/Top-5 准确率为73.31%/91.42%。

方法	延迟（毫秒）	50	100	500	1000
BP（滤波器）	35.1（15.8%↓）	$39.0_{\pm 1.41}/68.9_{\pm 1.17}$	$41.0_{\pm 0.33}/70.5_{\pm 0.66}$	$51.8_{\pm 0.30}/78.1_{\pm 0.38}$	$57.8_{\pm 0.30}/81.5_{\pm 0.18}$
BP（块）	**34.9 (16.3%↓)**	$\mathbf{66.5}_{\pm 0.81}/\mathbf{78.4}_{\pm 0.44}$	$\mathbf{66.8}_{\pm 0.23}/\mathbf{87.7}_{\pm 0.23}$	$\mathbf{68.6}_{\pm 0.18}/\mathbf{88.8}_{\pm 0.09}$	$\mathbf{69.8}_{\pm 0.12}/\mathbf{89.3}_{\pm 0.07}$
知识蒸馏 [15]	35.1 (15.8%↓)	$44.5_{\pm 1.20}/72.3_{\pm 0.87}$	$46.4_{\pm 0.34}/74.0_{\pm 0.58}$	$54.7_{\pm 0.26}/79.7_{\pm 0.19}$	$57.9_{\pm 0.21}/81.6_{\pm 0.12}$
FSKD [9]	35.1（15.8%↓）	$45.3_{\pm 0.77}/71.5_{\pm 0.62}$	$51.2_{\pm 0.30}/76.8_{\pm 0.23}$	$57.6_{\pm 0.21}/81.6_{\pm 0.15}$	$59.4_{\pm 0.13}/82.7_{\pm 0.06}$
对比蒸馏 [10]	35.1（15.8%↓）	$56.2_{\pm 0.37}/80.8_{\pm 0.31}$	$59.1_{\pm 0.22}/82.8_{\pm 0.11}$	$63.7_{\pm 0.18}/86.0_{\pm 0.05}$	$64.4_{\pm 0.03}/86.3_{\pm 0.07}$
MiR [11]	35.1（15.8%↓）	$64.1_{\pm 0.10}/86.3_{\pm 0.11}$	$65.1_{\pm 0.19}/87.0_{\pm 0.11}$	$67.0_{\pm 0.09}/88.1_{\pm 0.07}$	$67.8_{\pm 0.06}/88.5_{\pm 0.02}$
PRACTISE	**34.9 (16.3%↓)**	$\mathbf{70.3}_{\pm 0.16}/\mathbf{89.6}_{\pm 0.06}$	$\mathbf{71.5}_{\pm 0.74}/\mathbf{90.3}_{\pm 0.37}$	$\mathbf{72.5}_{\pm 0.04}/\mathbf{90.9}_{\pm 0.03}$	$\mathbf{72.5}_{\pm 0.05}/\mathbf{91.0}_{\pm 0.02}$

表 3 剪枝 ResNet-34 在 ImageNet-1k 上的 Top-1/Top-5 验证准确率（%）。我们的方法将延迟降低了 22.1% 。以往的方法会剪除残差连接内部和外部的滤波器（即图 8 中的“残差”）。原始 ResNet-34 的 Top-1/Top-5 准确率为 73.31%/91.42%。

方法	延迟（毫秒）	50	100	500	1000
BP（滤波器）	33.8 (18.9% ↓)	$24.2_{\pm 0.92}/52.7_{\pm 1.36}$	$27.6_{\pm 0.41}/56.7_{\pm 0.62}$	$42.9_{\pm 0.28}/70.5_{\pm 0.27}$	$51.2_{\pm 0.32}/76.5_{\pm 0.16}$
反向传播（块）	**32.5(22.1% ↓)**	$\mathbf{60.6}_{\pm 0.62}/\mathbf{83.5}_{\pm 0.42}$	$\mathbf{61.6}_{\pm 0.31}/\mathbf{84.3}_{\pm 0.36}$	$\mathbf{65.0}_{\pm 0.19}/\mathbf{86.5}_{\pm 0.20}$	$\mathbf{66.8}_{\pm 0.18}/\mathbf{87.5}_{\pm 0.13}$
知识蒸馏 [15]	33。8（18。9% ↓）	$30.1_{\pm 0.69}/57.7_{\pm 1.10}$	$33.1_{\pm 0.43}/61.0_{\pm 0.53}$	$45.7_{\pm 0.26}/72.2_{\pm 0.25}$	$50.5_{\pm 0.29}/75.9_{\pm 0.23}$
FSKD [9]	33.8 (18.9% ↓)	$31.1_{\pm 0.90}/56.5_{\pm 1.10}$	$36.6_{\pm 0.44}/63.1_{\pm 0.46}$	$42.8_{\pm 0.49}/69.1_{\pm 0.58}$	$44.9_{\pm 0.20}/70.5_{\pm 0.29}$
MiR [11]	33.8 (18.9% ↓)	$59.9_{\pm 0.30}/83.2_{\pm 0.31}$	$62.1_{\pm 0.22}/84.8_{\pm 0.18}$	$65.4_{\pm 0.07}/87.0_{\pm 0.03}$	$66.6_{\pm 0.05}/87.7_{\pm 0.04}$
PRACTISE	**32.5(22.1% ↓)**	$\mathbf{68.0}_{\pm 1.36}/\mathbf{88.2}_{\pm 0.77}$	$\mathbf{70.4}_{\pm 0.42}/\mathbf{89.7}_{\pm 0.23}$	$\mathbf{71.8}_{\pm 0.07}/\mathbf{90.5}_{\pm 0.02}$	$\mathbf{71.9}_{\pm 0.05}/\mathbf{90.6}_{\pm 0.04}$

表 4 使用 500 个训练样本对 ResNet-34 进行剪枝时，在 ImageNet-1k 上的 Top-1 验证准确率（%）。模型在三种不同压缩率下进行剪枝。先前的方法采用 CD 风格剪除滤波器，而我们则移除块。原始 ResNet-34 的 Top-1 准确率和延迟分别为 73.31% 和 41.7 毫秒。

延迟（毫秒）	BP（滤波器）	知识蒸馏 [15]	FSKD [9]	对比蒸馏 [10]	MiR [11]	PRACTISE
32.5（22.1%↓）	$47.54_{\pm 0.41}$	$49.34_{\pm 0.25}$	$33。19_{\pm 0.60}$	$59。65_{\pm 0.12}$	$68。14_{\pm 0.04}$	$\mathbf{71}。\mathbf{75}_{\pm 0.07}$
34.9（16.3%↓）	$58。94_{\pm 0.36}$	$61。01_{\pm 0.24}$	$62。56_{\pm 0.13}$	$68。17_{\pm 0.07}$	$70。53_{\pm 0.10}$	$\mathbf{72}。\mathbf{52}_{\pm 0.04}$
38.3 (8.2%↓)	$65。02_{\pm 0.30}$	$67。22_{\pm 0.18}$	$69。59_{\pm 0.09}$	$71。12_{\pm 0.06}$	$71。95_{\pm 0.07}$	$\mathbf{73}。\mathbf{04}_{\pm 0.06}$

64时，批量大小设为数据集大小。对于最终微调，默认情况下，使用随机梯度下降更新剪枝网络中的所有参数，通过2000次迭代最小化公式34（即特征模仿）。批量大小和学习率调度方式与适配器相同。我们采用 PyTorch 官方示例提供的数据增强策略，该策略与先前工作 [9], [10], [11]相同。所有实验均使用 PyTorch [38]进行。

为了与其他少样本剪枝方法进行比较，我们主要关注剪枝模型的延迟-准确率权衡。我们测试了采用先前方法剪枝后的神经网络的延迟，并直接引用这些方法报告的准确率。PRACTISE 的结果是通过使用五个不同抽样的微型训练集运行得到的。我们报告准确率均值及标准差。

6.1 不同数量的训练数据

我们将 PRACTISE 与最先进的少样本剪枝方法进行比较。按照先前工作的做法，我们使用数量不同的 ImageNet-1k 训练图像对 ResNet-34 进行剪枝。表 2 总结了结果。所有先前方法都在残差块内剪除滤波器（参见图 8 中的“normal”），而我们的 PRACTISE 和块丢弃方案都取得了显著更好的结果。首先，我们移除了前几个块以达到延迟目标，然后仅使用交叉熵损失对网络进行微调。这个简单的块丢弃基线方法“BP（块）”已经优于先前方法。值得注意的是，这也是我们的贡献，它揭示了在少样本场景中丢弃块的优势。PRACTISE 进一步提升了结果。在延迟降低幅度相近的情况下，它以 ImageNet-1k 上平均5.7% 的 Top-1 准确率，大幅超越了此前最先进水平。表 3 对比了这些方法在更大幅度延迟降低下的表现。PRACTISE 再次以显著优势超过 MiR，平均7.0% 的 Top-1 准确率。值得注意的是，我们的模型也比先前模型快1.3 毫秒。

6.2 不同的加速比

表 4 比较了 PRACTISE 与以往方法在不同加速比下的表现。当延迟降低8.2% 时，PRACTISE 的 Top-1 准确率比 MiR 高 1.1% 。当延迟进一步降低22.1%时，PRACTISE 的 Top-1 准确率比 MiR 高3.6% 。这表明，随着加速比增大，PRACTISE 的准确率高于其他方法。图 1 展示了不同方法的延迟-准确率权衡曲线。我们的 PRACTISE 显著优于以往方法。

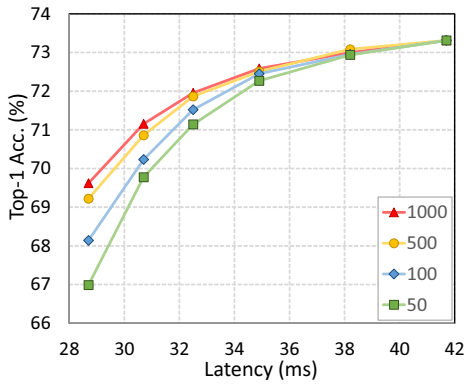


图 13. 数据-延迟-准确率权衡。使用 PRACTISE 对 ImageNet-1k 上的 ResNet-34 进行剪枝。图例中的数字表示不同数量的训练图像。

表 5 PRACTISE 与少样本剪枝方法在 ImageNet-1k 上的比较。使用 500 个训练样本对 MobileNetV2 进行剪枝。

方法	延迟（毫秒）	Top-1/Top-5
原始方法	37.6	71.9/90.3
反向传播（滤波器）	31.5 (16.2% ↓)	$45.0_{\pm 0.34}/71.8_{\pm 0.38}$
知识蒸馏 [15]	31.5 (16.2% ↓)	$48.4_{\pm 0.34}/73.9_{\pm 0.32}$
MiR [11]	31.5 (16.2% ↓)	$67.6_{\pm 0.05}/87.9_{\pm 0.04}$
PRACTISE	**30.4(19.1% ↓)**	$\mathbf{69.3}_{\pm 0.05}/\mathbf{88.9}_{\pm 0.05}$
反向传播（滤波器）	34.1（9.3% ↓）	$55.5_{\pm 0.16}/80.3_{\pm 0.26}$
知识蒸馏 [15]	34.1 (9.3% ↓)	$59.1_{\pm 0.17}/82.5_{\pm 0.15}$
MiR [11]	34.1 (9.3% ↓)	$69.7_{\pm 0.04}/89.2_{\pm 0.03}$
PRACTISE	**31.9（15.2% ↓）**	$\mathbf{70.3}_{\pm 0.03}/\mathbf{89.5}_{\pm 0.03}$

6.3 数据-延迟-准确率权衡

接下来，为了更好地理解少样本压缩场景下的数据-延迟-准确率权衡，我们使用 PRACTISE 对 ResNet-34 进行剪枝，设置不同的延迟降低幅度和训练图像数量。图 13 展示了结果。延迟降低幅度较小时，不同训练数据量下的准确率相近。但延迟降低幅度较大时，仅依靠少量训练数据很难恢复准确率。不同训练数据量之间的准确率差距也会变大。

6.4 MobileNetV2 上的结果

MobileNetV2 [36] 是一种轻量级模型，广泛应用于移动设备。它包含 10 个可由 PRACTISE 移除的块。表 5 总结了 MobileNetV2 的结果。PRACTISE 的表现显著优于以往方法。与 MiR 相比，PRACTISE 获得的剪枝模型延迟更低、准确率更高。

6.5 使用合成/域外图像训练 ages

在某些极端场景中，甚至不会提供任何原始训练样本，因此必须采用零样本剪枝。由于 PRACTISE 不需要真实标签，因此能够利用数据无关知识蒸馏方法合成的图像来加速神经网络。表 6 总结了相关结果。请注意，大多数零样本剪枝方法采用

表 6 PRACTISE 与无数据方法在 ImageNet-1k 上的比较。报告了 ImageNet-1k 上的延迟（ms）和 Top-1 验证准确率（%）。* 表示该方法由我们自行重新实现。

网络	方法	剪枝	延迟	Top-1
ResNet-50	原始方法		83.8	76.1
	DI [26]	滤波器	-	72.0
	MixMix [28]	滤波器	-	69.8
	ADI [26]	滤波器	-	73.3
	ADI* [26]	滤波器	79.9（4.7%↓）	73。5
	PRACTISE	块	**66.2(21.0%↓)**	**74.8**
MobileNetV2	原始方法		37.6	71.9
	DI [26]	滤波器	-	15.3
	MixMix [28]	滤波器	-	42.5
	ADI* [26]	滤波器	30.8（18.1%↓）	62.8
	PRACTISE	块	**30.4(19.1%↓)**	**68.0**

表7 不同域外训练数据集在 ImageNet-1k 上的 Top-1 验证准确率（%）。PRACTISE 将 ResNet-50 的延迟降低了 21% 。

数据集	50	500	1000	5000	All
ImageNet [37]	74.22	74.58	74.58	75.14	75.24
ADI [26]	69.85	72.68	73.01	74.40	74.79
CUB [39]	72.49	73.71	73.94	74.86	74.92
Place365 [40]	**72.80**	**74.10**	**74.18**	**75.05**	**75.21**

滤波器剪枝，而正如我们所展示的，这种方法不如剪枝块。我们采用 ADI [26] 生成的合成图像作为训练集。对于 ResNet-50 和 MobileNetV2，PRACTISE 在实现更大延迟降低的同时，取得了更高的 Top-1 准确率。我们主张未来在 DFKD 中采用丢弃块。

零样本剪枝的另一种选择是收集域外训练图像。PRACTISE 同样能够稳健地处理这些数据。表7展示了相关结果。原始 ResNet-50 在 ImageNet-1k 上进行训练，然后我们使用 PRACTISE 在其他数据集上对其进行剪枝。ADI 由 DeepInversion 合成的图像组成 [26]。CUB [39]包含来自 200 个类别的鸟类图像。Place365 [40]由场景图片组成。我们的 PRACTISE 具有很强的泛化能力，能够处理所有这些数据集。域外数据的另一个优势是图像数量不受限制。我们注意到，使用更多训练样本可以提升准确率，甚至使其接近使用完整的原始 ImageNet-1k训练集时的准确率。

6.6 丢弃块的不同标准

最后，我们将 PRACTISE 与其他丢弃块的标准进行比较。图 9已经展示了仅移除一个块时的结果。显然，PRACTISE 优于其他方法，并且与微调模型的准确率高度一致。大多数现有方法主要衡量原始网络与未经微调的剪枝网络之间的差距。它们忽略了每个被丢弃块的可恢复性，因此导致了较差的丢弃块策略。我们的 PRACTISE 重视每个块的可恢复性和加速比。因此，剪枝网络能够实现更高的准确率和更低的延迟。

表 8 比较了丢弃更多块的不同标准。随着被丢弃的块数量增加，我们的

表8 使用500张图像对 ResNet-34 进行剪枝时，在 ImageNet-1k 上的 Top-1 验证准确率（%）。我们比较了针对丢弃不同数量块的不同标准。

#被丢弃的块	1	2	3	4	5
CURL [18]	72.33	71.08	69.14	65.48	64.97
ϵ-ResNet [17]	72.51	71.20	69.00	67.90	64.75
L2距离	72.51	71.20	69.00	68.61	64.93
PRACTISE	**73.02**	**72.52**	**71.92**	**70.86**	**69.35**

PRACTISE 的表现进一步超越其他方法。总而言之，与其他方法相比，PRACTISE 能找到最适合丢弃的块集合。

7 结论与未来工作

本文旨在利用极少量训练集加速神经网络。我们在少样本压缩中揭示了一项关键观察：微调剪枝网络几乎是一个凸问题。这一猜想成功解释了相关算法为何不会遭遇过拟合问题。我们认为，这一发现推动了少样本模型压缩领域的重大进展。我们还建立了特征模仿的理论。与原始方法相比，该方法的参数方差更低。基于理论结果，我们首次揭示，在这种场景下，相比以往的滤波器级剪枝，丢弃块在根本上更优，因为它具有更高的凸性和加速比。为确定应丢弃哪些块，我们提出了一个新概念——可恢复性，用于衡量利用少量样本恢复剪枝网络准确率的难度。结合可恢复性与特征模仿，我们提出了 PRACTISE。该方法具有出色的延迟-准确率性能，并能稳健地处理合成图像和域外图像。大量实验表明，PRACTISE 显著优于以往方法（在 ImageNet-1k 上，平均 Top-1 准确率提高 7%，延迟降低 22%）。

我们的猜想存在一些局限，例如仅适用于对剪枝模型进行微调。对于迁移学习、领域自适应等其他场景，仍有待进一步探索。我们的理论主要针对判别模型提出。如何蒸馏生成模型仍是一个悬而未决的问题。在网络加速方面，如何计算其他压缩方案的可恢复性也是一个值得探讨的问题。近年来，在下游任务上微调并加速大型预训练网络已成为迫切需求。在下游任务上使用 PRACTISE 调优预训练模型也值得探索。

参考文献

[1] H. Li、A. Kadav、I. Durdanovic、H. Samet 和 H. P. Graf，“用于高效卷积网络的滤波器剪枝”，*arXiv* 预印本 *arXiv:1608.08710*，2016年。[2] J.-H. Luo、J. Wu 和 W. Lin，“ThiNet：一种用于深度神经网络压缩的滤波器级剪枝方法”，载于国际计算机视觉会议，2017年，第 5058–5066 页。[3] Z. Liu、J. Li、Z. Shen、G. Huang、S. Yan 和 C. Zhang，“通过网络瘦身学习高效卷积网络”，载于国际计算机视觉会议，2017年，第 2736–2744 页。[4] Z. Liu、X. Zhang、Z. Shen、Y. Wei、K.-T. Cheng 和 J. Sun，“通过数值梯度更新进行联合多维剪枝”，*IEEE* 图像处理汇刊，第 30 卷，第 8034–8045 页，2021年。

[5] J. Guo、Y. Li、W. Lin、Y. Chen 和 J. Li，“网络解耦：从常规卷积到深度可分离卷积”，载于《英国机器视觉会议》，2018年，第248页。[6] S. Lin、R. Ji、C. Chen、D. Tao 和 J. Luo，“通过结合知识迁移的低秩分解实现整体式CNN压缩”，《IEEE 模式分析与机器智能汇刊》，第41卷，第12期，第2889–2905页，2018年。[7] S. Han、H. Mao 和 W. J. Dally，“深度压缩：通过剪枝、训练式量化和霍夫曼编码压缩深度神经网络”，载于《国际学习表征会议》，2016年，第1–14页。[8] R. Banner、I. Hubara、E. Hoffer 和 D. Soudry，“神经网络8位训练的可扩展方法”，载于《神经信息处理系统进展第31卷》，2018年，第5151–5159页。[9] T. Li、J. Li、Z. Liu 和 C. Zhang，“用于高效网络压缩的少样本知识蒸馏”，载于《IEEE 计算机视觉与模式识别会议》，2020年，第14 639–14 647页。[10] H. Bai、J. Wu、I. King 和 M. Lyu，“通过交叉蒸馏实现少样本网络压缩”，载于《人工智能促进协会人工智能会议》，第04卷，2020年，第3203–3210页。[11] H. Wang、J. Liu、X. Ma、Y. Yong、Z. Chai 和 J. Wu，“少样本模型压缩：先模仿后替换”，载于《IEEE 计算机视觉与模式识别会议》，2022年，第701–710页。[12] F. Draxler、K. Veschgini、M. Salmhofer 和 F. Hamprecht，“神经网络能量景观中几乎不存在障碍”，载于《国际机器学习会议》，2018年，第1309–1318页。[13] J. Frankle、G. K. Dziugaite、D. Roy 和 M. Carbin，“线性模式连通性与彩票假设”，载于《国际机器学习会议》，2020年，第3259–3269页。[14] S. K. Ainsworth、J. Hayase 和 S. Srinivasa，“Git Re-Basin：根据置换对称性合并模型”，载于《国际学习表征会议》，2023年，第1–29页。[15] G. E. Hinton、O. Vinyals 和 J. Dean，“神经网络中的知识蒸馏”，arXiv 预印本 arXiv:1503.02531，2015年。[16] W. Wang、S. Zhao、M. Chen、J. Hu、D. Cai 和 H. Liu，“DBP：用于深度模型加速的基于判别的块级剪枝”，arXiv 预印本 arXiv:1912.10178，2019年。[17] X. Yu、Z. Yu 和 S. Ramalingam，“在深度残差网络中学习严格恒等映射”，载于《IEEE 计算机视觉与模式识别会议》，2018年，第4432–4440页。[18] J.-H. Luo 和 J. Wu，“结合残差连接和有限数据的神经网络剪枝”，载于《IEEE 计算机视觉与模式识别会议》，2020年，第1458–1467页。[19] G.-H. Wang 和 J. Wu，“实用网络加速与微型数据集”，载于《IEEE 计算机视觉与模式识别会议》，2023年，即将出版。[20] Y. He、P. Liu、Z. Wang、Z. Hu 和 Y. Yang，“通过几何中位数进行滤波器剪枝以加速深度卷积神经网络”，载于《IEEE 计算机视觉与模式识别会议》，2019年，第4340–4349页。[21] P. Molchanov、A. Mallya、S. Tyree、I. Frosio 和 J. Kautz，“神经网络剪枝的重要性估计”，载于《IEEE 计算机视觉与模式识别会议》，2019年，第11 264–11 272页。[22] Z. Liu、M. Sun、T. Zhou、G. Huang 和 T. Darrell，“重新思考网络剪枝的价值”，载于《国际学习表征会议》，2018年，第1–21页。[23] X. Ding、T. Hao、J. Tan、J. Liu、J. Han、Y. Guo 和 G. Ding，“ResRep：通过解耦记忆与遗忘实现无损CNN剪枝”，载于《国际计算机视觉会议》，2021年，第4510–4520页。[24] Y. Li、K. Adamczewski、W. Li、S. Gu、R. Timofte 和 L. Van Gool，“重新审视神经网络的随机通道剪枝”，载于《IEEE 计算机视觉与模式识别会议》，2022年，第191–201页。[25] G.-H. Wang、J. Li、B. Li 和 Y. Lu，“EVC：迈向具有掩码衰减的实时神经图像压缩”，载于《国际学习表征会议》，2023年，第1–23页。[26] H. Yin、P. Molchanov、J. M. Alvarez、Z. Li、A. Mallya、D. Hoiem、N. K. Jha 和 J. Kautz，“梦境式蒸馏：通过DeepInversion 实现无数据知识迁移”，载于《IEEE 计算机视觉与模式识别会议》，2020年，第8715–8724页。[27] G. Fang、K. Mo、X. Wang、J. Song、S. Bei、H. Zhang 和 M. Song，“速度提升至100倍的无数据知识蒸馏”，载于《人工智能促进协会人工智能会议》，第36卷，第6期，2022年，第6597–6604页。[28] Y. Li、F. Zhu、R. Gong、M. Shen、X. Dong、F. Yu、S. Lu 和 S. Gu，“MixMix：无数据压缩所需的一切就是特征混合与数据混合”，载于《国际计算机视觉会议》，2021年，第4410–4419页。[29] Y. Liu、W. Zhang 和 J. Wang，“零样本对抗量化”，载于《IEEE 计算机视觉与模式识别会议》，2021年，第1512–1521页。[30] X. Zhang、H. Qin、Y. Ding、R. Gong、Q. Yan、R. Tao、Y. Li、F. Yu 和 X. Liu，“为精确的无数据

量化实现多样化样本生成”，载于 *IEEE* 计算机视觉与模式识别会议，2021年，第15 658–15 667页。[31] Y. Cai、Z. Yao、Z. Dong、A. Gholami、M. W. Mahoney 和 K. Keutzer，“ZeroQ：一种新颖的零样本量化框架”，载于 *IEEE* 计算机视觉与模式识别会议，2020年，第13 169–13 178页。[32] A. Romero、N. Ballas、S. E. Kahou、A. Chassang、C. Gatta 和 Y. Bengio，“FitNets：轻量深度网络的提示”，载于 国际学习表征会议，2015年，第1–13页。[33] G.-H. Wang、Y. Ge 和 J. Wu，“通过模仿特征进行知识蒸馏”，*IEEE* 模式分析与机器智能汇刊，第44卷，第11期，第8183–8195页，2022年。[34] C. Shen、X. Wang、Y. Yin、J. Song、S. Luo 和 M. Song，“面向少样本知识蒸馏的渐进式网络嫁接”，载于 人工智能促进协会人工智能会议，第35卷，第3期，2021年，第2541–2549页。[35] K. He、X. Zhang、S. Ren 和 J. Sun，“用于图像识别的深度残差学习”，载于 *IEEE* 计算机视觉与模式识别会议，2016年，第770–778页。[36] M. Sandler、A. Howard、M. Zhu、A. Zhmoginov 和 L.-C. Chen，“MobileNetV2：倒残差与线性瓶颈”，载于 *IEEE* 计算机视觉与模式识别会议，2018年，第4510–4520页。[37] O. Russakovsky、J. Deng、H. Su、J. Krause、S. Satheesh、S. Ma、Z. Huang、A. Karpathy、A. Khosla、M. Bernstein、A. C. Berg 和 L. Fei-Fei，“ImageNet 大规模视觉识别挑战赛”，国际计算机视觉期刊，第115卷，第3期，第211–252页，2015年。[38] A. Paszke、S. Gross、F. Massa、A. Lerer、J. Bradbury、G. Chanan、T. Killeen、Z. Lin、N. Gimelshein、L. Antiga 等，“PyTorch：一种命令式风格的高性能深度学习库”，载于 神经信息处理系统进展第*32*卷，2019年，第8026–8037页。[39] C. Wah、S. Branson、P. Welinder、P. Perona 和 S. Belongie，“Caltech-UCSD birds-200-2011 数据集”，加州理工学院，技术报告 CNS-TR-2011-001，2011年。[40] B. Zhou、A. Khosla、A. Lapedriza、A. Torralba 和 A. Oliva，“Places：用于深度场景理解的图像数据库”，*arXiv* 预印本 *arXiv:1610.02055*，2016年。



Guo-Hua Wang 获南京大学管理与工程学院学士学位。目前，他是中国南京大学计算机科学与技术系的博士生。他的研究兴趣包括计算机视觉和机器学习。



JianxinWu（IEEE 会员）获得南京大学计算机科学学士和硕士学位，并获得佐治亚理工学院计算机科学博士学位。他目前是中国南京大学计算机科学与技术系及人工智能学院的教授，并隶属于中国新型软件技术国家重点实验室。他的研究兴趣包括计算机视觉和机器学习。他曾担任 CVPR、ICCV、ECCV 的（高级）领域主席，人工智能促进协会、国际人工智能联合会议，以及《IEEE模式分析与机器智能汇刊》的副编辑。

附录 A 命题 1 的证明

为简洁起见，我们省略参数 θ in $q_\theta(y|x)$ 和 $q_\theta(y|f,x)$ 。

$$KL\left[p(y,f|x)||q(y,f|x)\right] \tag{35}$$
$$= \iint p(y,f|x)\ln\frac{p(y,f|x)}{q(y,f|x)}dydf \tag{36}$$
$$= \iint p(f|y,x)p(y|x)\ln\frac{p(f|y,x)p(y|x)}{q(f|y,x)q(y|x)}dydf \tag{37}$$
$$= \int p(y|x)\int p(f|y,x)\left[\ln\frac{p(y|x)}{q(y|x)}+\ln\frac{p(f|y,x)}{q(f|y,x)}\right]dydf \tag{38}$$
$$= \int p(y|x)\int p(f|y,x)\ln\frac{p(y|x)}{q(y|x)}dydf + \int p(y|x)\int p(f|y,x)\ln\frac{p(f|y,x)}{q(f|y,x)}dydf \tag{39}$$
$$= \int p(y|x)\ln\frac{p(y|x)}{q(y|x)}dy + \int p(y|x)KL\left[p(f|y,x)||q(f|y,x)\right]dy \tag{40}$$
$$= KL\left[p(y|x)||q(y|x)\right] + \int p(y|x)KL\left[p(f|y,x)||q(f|y,x)\right]dy \tag{41}$$
$$\geq KL\left[p(y|x)||q(y|x)\right]. \tag{42}$$

附录 B 命题 2 的证明

为简洁起见，我们省略了 θ 中的参数 q_θ（$y|x$）以及 q_θ（$y|f$， x）。

$$KL\left[p(y,f|x)||q(y,f|x)\right] \tag{43}$$
$$= \iint p(y,f|x)\ln\frac{p(y,f|x)}{q(y,f|x)}dydf \tag{44}$$
$$= \iint p(y|f,x)p(f|x)\ln\frac{p(y|f,x)p(f|x)}{q(y,f|x)}dydf \tag{45}$$
$$= \int p(f|x)\int p(y|f,x)\ln\frac{p(y|f,x)p(f|x)}{q(y,f|x)}dydf \tag{46}$$
$$= \mathbb{E}_{f\sim p(f|x)}\int p(y|f,x)\ln\frac{p(y|f,x)p(f|x)}{q(y,f|x)}dy \tag{47}$$
$$= \mathbb{E}_{f\sim p(f|x)}\left[\int p(y|f,x)\ln p(f|x)dy + \int p(y|f,x)\ln\frac{p(y|f,x)}{q(y,f|x)}dy\right] \tag{48}$$
$$= \mathbb{E}_{f\sim p(f|x)}\left[\ln p(f|x)+\int p(y|f,x)\ln\frac{p(y|f,x)}{q(y|f,x)q(f|x)}dy\right] \tag{49}$$
$$= \mathbb{E}_{f\sim p(f|x)}\left[\ln p(f|x)-\int p(y|f,x)\ln q(f|x)dy + \int p(y|f,x)\ln\frac{p(y|f,x)}{q(y|f,x)}dy\right] \tag{50}$$
$$= \mathop{\mathbb{E}}_{f\sim p(f|x)}\left\{-\ln q(f|x) + KL\left[p(y|f,x)||q(y|f,x)\right]\right\}+C. \tag{51}$$

对于特征模仿，我们令 q_θ（$f|x$）服从高斯分布，其分布参数由 x 通过深度神经网络计算得到。因此，我们有

$$q_\theta(f|x)=\frac{1}{\sqrt{2\pi}\sigma_\theta(x)}\exp\left(-\frac{1}{2}\left(\frac{f-\mu_\theta(x)}{\sigma_\theta(x)}\right)^2\right). \tag{52}$$

因此，特征模仿损失变为

$$\mathbb{E}_{f\sim p(f|x)}[-\ln q_\theta(f|x)] \tag{53}$$
$$= \mathbb{E}_{f\sim p(f|x)}\left[-\ln\frac{1}{\sqrt{2\pi}\sigma_\theta(x)}\exp\left(-\frac{1}{2}\left(\frac{f-\mu_\theta(x)}{\sigma_\theta(x)}\right)^2\right)\right] \tag{54}$$
$$= \mathbb{E}_{f\sim p(f|x)}\left[\frac{1}{2}\left(\frac{f-\mu_\theta(x)}{\sigma_\theta(x)}\right)^2-\ln\frac{1}{\sqrt{2\pi}\sigma_\theta(x)}\right]. \tag{55}$$

如果假设方差为常数，并令 $\sigma(x)=\frac{1}{\sqrt{2\beta}}$，则有

$$\mathbb{E}_{f\sim p(f|x)}[-\ln q_\theta(f|x)] \tag{56}$$
$$= \mathbb{E}_{f\sim p(f|x)}\left[\beta\left(f-\mu_\theta(x)\right)^2-\ln\frac{\sqrt{2\beta}}{\sqrt{2\pi}}\right] \tag{57}$$
$$= \mathbb{E}_{f\sim p(f|x)}\left[\beta\left(f-\mu_\theta(x)\right)^2\right]+C. \tag{58}$$

附录 C
命题 3 的证明

给定 n 训练样本 $x_1,x_2,\cdots,x_n$ 以及损失函数

$$\mathcal{L}=-\ln q_\theta(f|x), \tag{59}$$

我们可以通过微调得到局部极小解。根据猜想 2，损失景观关于参数是凸的，局部极小解不会逃离当前的优化盆地。因此，可以通过以下方式求得该解:

$$\arg\max_\theta\sum_{i=1}^{n}\ln q_\theta(f_i|x_i). \tag{60}$$

请注意，这等价于最大似然估计（MLE）。每个样本的对数似然为

$$l_i(\theta)=\ln q(f_i|x_i), \tag{61}$$

且总对数似然为

$$l(\theta)=\sum_{i=1}^{n}l_i(\theta). \tag{62}$$

ˆ对于最大似然估计，达到最优解的条件 θ 为 $\frac{\partial l(\theta)}{\partial\theta}|_{\theta=\hat{\theta}}=0$。对 $\tilde{\theta}$ 进行泰勒展开并省略高阶项后，

$$0=\frac{\partial l(\theta)}{\partial\theta}|_{\theta=\hat{\theta}}\approx\frac{\partial l(\tilde{\theta})}{\partial\tilde{\theta}}+\frac{\partial^2 l(\tilde{\theta})}{\partial\tilde{\theta}^2}(\hat{\theta}-\tilde{\theta}). \tag{63}$$

因此，我们有

$$\hat{\theta}\doteq\tilde{\theta}-\frac{\frac{\partial l(\tilde{\theta})}{\partial\tilde{\theta}}}{\frac{\partial^2 l(\tilde{\theta})}{\partial\tilde{\theta}^2}}. \tag{64}$$

注意，在我们的 ˜猜想 2 下，这个等式近似成立。为简便起见，在后文中我们将 θ 记作 θ

ˆ在本证明中。接下来，我们将分析 θ 的方差。对于每个样本，

$$\frac{\partial l_i(\theta)}{\partial \theta} = \frac{\dot{q}_\theta(f_i|x_i)}{q_\theta(f_i|x_i)}, \tag{65}$$

其中 $\dot{q}_\theta(f_i|x_i)$表示 $\frac{\partial q_\theta(f_i|x_i)}{\partial \theta}$。

$$\mathbb{E}_{f\sim q(f|x_i)}\left[\frac{\partial l_i(\theta)}{\partial \theta}\right] = \int \frac{\dot{q}_\theta(f|x_i)}{q_\theta(f|x_i)} q_\theta(f|x_i) df \tag{66}$$

$$= \int \dot{q}_\theta(f|x_i) df \tag{67}$$

$$= \frac{\partial}{\partial \theta} \int q_\theta(f|x_i) df \tag{68}$$

$$= \frac{\partial 1}{\partial \theta} = 0\,. \tag{69}$$

$$Var_{f\sim q(f|x_i)}\left[\frac{\partial l_i(\theta)}{\partial \theta}\right] = \int \dot{l}_i(\theta)^2 q_\theta(f|x_i) df\,. \tag{70}$$

费 Fisher 信息为 $\mathcal{I}_i(\theta) = \int \dot{l}_i(\theta)^2 q_\theta(f|x_i) df$，并且

$$\mathcal{I}(\theta) = \mathbb{E}_{x\sim p(x)} \int \dot{l}_i(\theta)^2 q_\theta(f|x) df\,. \tag{71}$$

注意，$\frac{\dot{l}(\theta)}{n} = \frac{1}{n}\sum_{i=1}^{n} \dot{l}_i)\,(\theta)$，根据中心极限定理（CLT），我们有

$$\frac{\dot{l}(\theta)}{n} \sim \mathcal{N}(0, \frac{\mathcal{I}(\theta)}{n})\,. \tag{72}$$

对于 $\frac{\partial^2 l_i(\theta)}{\partial \theta^2}$，我们有

$$\frac{\partial^2 l_i(\theta)}{\partial \theta^2} = \frac{\ddot{q}_\theta(f_i|x_i)}{q_\theta(f_i|x_i)} - \left(\frac{\dot{q}_\theta(f_i|x_i)}{q_\theta(f_i|x_i)}\right)^2\,. \tag{73}$$

然后，

$$\mathbb{E}_{f\sim q(f|x_i)}\left[\frac{\partial^2 l_i(\theta)}{\partial \theta^2}\right] \tag{74}$$

$$= \mathbb{E}_{f\sim q(f|x_i)}\left[\frac{\ddot{q}_\theta(f|x_i)}{q_\theta(f|x_i)}\right] - \mathbb{E}_{f\sim q(f|x_i)}\left(\frac{\dot{q}_\theta(f|x_i)}{q_\theta(f|x_i)}\right)^2 \tag{75}$$

$$= -\mathcal{I}_i(\theta)\,. \tag{76}$$

根据大数定律，我们有

$$\frac{\ddot{l}(\theta)}{n} = \frac{1}{n}\sum_{i=1}^{n} \ddot{l}_i(\theta) \xrightarrow{P} -\mathcal{I}(\theta)\,. \tag{77}$$

根据式 72、77 和 64，我们有

$$\hat{\theta} \sim \mathcal{N}(\tilde{\theta}, \frac{1}{n\mathcal{I}(\tilde{\theta})})\,. \tag{78}$$

附录 D 断言 4 的证明

为使用公式 18 中的损失进行微调，我们假设 $q\,(f|x)$ 服从高斯分布，且 $\sigma\,(x) = \frac{1}{\sqrt{2\beta}}$，因此

$$q(f|x) = \sqrt{\frac{\beta}{\pi}} \exp\left(-\beta\left(f-\mu_\theta(x)\right)^2\right)\,. \tag{79}$$

我们假设 $\mu(x) = w\cdot x + b$，其中 w 和 b 是参数θ。首先，我们考虑 w:

$$\dot{q}_w(f|x) = \sqrt{\frac{\beta}{\pi}} \exp\left(-\beta\left(f-\mu(x)\right)^2\right) 2\beta\left(f-\mu(x)\right)x\,. \tag{80}$$

And,

$$\dot{l}_i(w) = \frac{\dot{q}_w(f_i|x_i)}{q_w(f_i|x_i)} \tag{81}$$

$$= 2\beta\left(f_i - \mu(x_i)\right)x_i \tag{82}$$

$$= 2\beta x_i(f_i - w\cdot x_i - b)\,, \tag{83}$$

$$\ddot{l}_i(w) = -2\beta x_i^2\,. \tag{84}$$

$$\mathcal{I}(w) = -\mathbb{E}_x \ddot{l}_i(w) = 2\beta\mathbb{E}[x^2]\,. \tag{85}$$

根据命题 3，我们得到

$$\hat{w} \sim \mathcal{N}(\tilde{w}, \frac{1}{2n\beta\mathbb{E}[x^2]})\,. \tag{86}$$

然后，对于 b，我们也有

$$\dot{q}_b(f|x) = \sqrt{\frac{\beta}{\pi}} \exp\left(-\beta\left(f-\mu(x)\right)^2\right) 2\beta\left(f-\mu(x)\right)\,. \tag{87}$$

$$\dot{l}_i(b) = \frac{\dot{q}_b(f_i|x_i)}{q_b(f_i|x_i)} = 2\beta\left(f_i-\mu(x_i)\right) = 2\beta(f_i - w\cdot x_i - b)\,. \tag{88}$$

$$\ddot{l}_i(b) = -2\beta\,. \tag{89}$$

$$\mathcal{I}(b) = -\mathbb{E}_x \ddot{l}_i(b) = 2\beta\,. \tag{90}$$

根据命题 3，我们得到

$$\hat{b} \sim \mathcal{N}(\tilde{b}, \frac{1}{2n\beta})\,. \tag{91}$$

附录 E
命题 5 的证明

公式 13 中的分类损失可以写为

$$\mathbb{E}_{x\sim p(x)} KL\left[p(y|x)||q(y|x)\right]\,. \tag{92}$$

采用独热编码 $\boldsymbol{y}$时，该损失等于

$$KL\left[p(y|x)||q(y|x)\right] = \sum_{j=1}^{C} p(y_j|x)\ln\frac{p(y_j|x)}{q(y_j|x)} \tag{93}$$

$$= -\ln q(y_c|x)\,, \tag{94}$$

其中，c 表示真实类别。该损失符合命题 3 中的规范形式。在深度神经网络中，x 首先被转换为特征 f，然后 $q(y_c|f)$ 的计算方式为

$$q(y_c|f) = \sigma(\boldsymbol{w}\cdot f + \boldsymbol{b})_c = \frac{\exp(w_c\cdot f + b_c)}{\sum_{j=1}^{C}\exp(w_j\cdot f + b_j)}\,. \tag{95}$$

如果第 i 个样本属于类别 c，则对数似然为

$$l_i(\boldsymbol{w},\boldsymbol{b}) = \ln\sigma(\boldsymbol{w}\cdot f_i + \boldsymbol{b})_c \tag{96}$$

$$= w_c\cdot f_i + b_c - \ln\sum_{j=1}^{C}\exp(w_j\cdot f_i + b_j)\,. \tag{97}$$

首先，我们考虑 $\boldsymbol{w}$。我们有

$$\dot{l}_i(w_c) = f_i - \frac{\exp(w_c\cdot f_i + b_c)f_i}{\sum_{j=1}^{C}\exp(w_j\cdot f_i + b_j)} = f_i - \sigma(\boldsymbol{w}\cdot f_i + \boldsymbol{b})_c f_i\,, \tag{98}$$

$$\dot{l}_i(w_{c'}) = -\frac{\exp(w_{c'}\cdot f_i + b_{c'})f_i}{\sum_{j=1}^{C}\exp(w_j\cdot f_i + b_j)} = -\sigma(\boldsymbol{w}\cdot f_i + \boldsymbol{b})_{c'} f_i\,, \tag{99}$$

其中 $c' \neq c$。然后，

$$\ddot{l}_i(w_j) = \frac{\partial \sigma(\boldsymbol{w} f_i + \boldsymbol{b})_j f_i}{\partial w_j} \tag{100}$$

$$= -\sigma(\boldsymbol{w} f_i + \boldsymbol{b})_j [1 - \sigma(\boldsymbol{w} f_i + \boldsymbol{b})_j] f_i^2 \,. \tag{101}$$

若记 $\epsilon_j = \sigma(\boldsymbol{w} \cdot f_i + \boldsymbol{b})_j [1 - \sigma(\boldsymbol{w} \cdot f_i + \boldsymbol{b})_j]$，则在实践中，$\epsilon_j$ 始终趋近于零。我们有 $\ddot{l}_i(w_j) = -\epsilon_j f_i^2$，且

$$\mathcal{I}(w_j) = -\mathbb{E}_f \ddot{l}_i(w_j) = \epsilon_j \mathbb{E}[f^2] \,. \tag{102}$$

最后，我们得到

$$\hat{w}_j \sim \mathcal{N}(\tilde{w}_j, \frac{1}{n\epsilon_j \mathbb{E}[f^2]}) \,. \tag{103}$$

现在，我们考虑 $\boldsymbol{b}$。

$$\dot{l}_i(b_c) = 1 - \frac{\exp(w_c \cdot f_i + b_c)}{\sum_{j=1}^{C} \exp(w_j \cdot f_i + b_j)} = 1 - \sigma(\boldsymbol{w} \cdot f_i + \boldsymbol{b})_c \,, \tag{104}$$

$$\dot{l}_i(b_{c'}) = -\frac{\exp(w_{c'} \cdot f_i + b_{c'})}{\sum_{j=1}^{C} \exp(w_j \cdot f_i + b_j)} = -\sigma(\boldsymbol{w} \cdot f_i + \boldsymbol{b})_{c'} \,, \tag{105}$$

其中 $c' \neq c$。并且，

$$\ddot{l}_i(b_j) = \frac{\partial \sigma(\boldsymbol{w} f_i + \boldsymbol{b})_j}{\partial b_j} = -\sigma(\boldsymbol{w} f_i + \boldsymbol{b})_j [1 - \sigma(\boldsymbol{w} f_i + \boldsymbol{b})_j] \,. \tag{106}$$

$$\mathcal{I}(b_j) = -\mathbb{E}_f \ddot{l}_i(b_j) = \epsilon_j \,. \tag{107}$$

最后，我们得到

$$\hat{b}_j \sim \mathcal{N}(\tilde{b}_j, \frac{1}{n\epsilon_j}) \,. \tag{108}$$