NTK-SAP：通过对齐训练动态改进神经网络剪枝

YiteWang[1], **Dawei Li**[1], **RuoyuSun**[2,3*]
[1]美国伊利诺伊大学厄巴纳-香槟分校[2]深圳国际工业与应用数学中心，深圳大数据研究院[3]香港中文大学（深圳）数据科学学院，中国深圳
{yitew2,dawei2}@illinois.edu, sunruoyu@cuhk.edu.cn

arXiv:2304.02840v1 [cs.LG] 6 Apr 2023

摘要

训练前对神经网络进行剪枝，因其有望减少训练时间和内存占用而日益受到关注。一种流行的方法是依据某种度量指标对连接进行剪枝，但尚不清楚哪种度量指标最为合适。神经切线核（NTK）理论的最新进展表明，规模足够大的神经网络的训练动态与神经切线核的谱密切相关。受此发现启发，我们提出剪除对神经切线核谱影响最小的连接。该方法有助于保持神经切线核的谱，从而可能使训练动态与稠密网络的训练动态保持一致。然而，一个潜在问题是，与给定初始点对应的固定权重神经切线核，可能与训练阶段后续迭代所对应的神经切线核存在很大差异。为此，我们进一步提出对多个随机权重实例进行采样，以估计神经切线核的谱。需要指出的是，我们的方法与权重无关，这一点不同于大多数依赖权重的现有方法。此外，我们使用随机输入计算固定权重神经切线核，因此我们的方法也与数据无关。我们将这一前瞻性剪枝算法命名为神经切线核谱感知剪枝（NTK-SAP）。实验结果表明，我们的方法在多个数据集上均优于所有基线方法。代码已发布于 https://github. com/YiteWang/NTK-SAP。

1 引言

过去十年见证了深度神经网络（DNNs）在各种应用中的成功。现代深度神经网络通常具有高度过参数化的特点，这使得训练和部署的计算开销十分昂贵。网络剪枝已成为降低时间和内存开销的有力工具。剪枝方法主要有两类：事后剪枝（Hanet al., 2015；Rendaet al.,2020年；Molchanovet al., 2019；LeCun etal.,1989；Hassibi&Stork,1992）和前瞻剪枝（Lee等人, 2018；Wang 等人,2020年；Alizadeh 等, 2022年；Tanaka 等人, 2020年；de Jorge 等，2020b；Liu & Zenke,2020）。前一类方法在训练大型网络之后对网络进行剪枝，而后一类方法在训练之前对网络进行剪枝。本文聚焦于前瞻剪枝。

SNIP（Lee 等人，2018年）可能是现代神经网络中最早的前瞻剪枝方法。它会剪除对初始损失函数影响最小的初始权重。SNIP 可以被视为基于显著性分数的剪枝的一个特例：剪除具有最低“显著性分数”的连接，其中显著性分数是一种衡量连接重要性的度量指标。对于 SNIP 而言，一个连接的显著性分数是剪除该连接前后的损失函数之差。GraSP（Wang 等人，2020年）和 Synflow（Tanaka 等人，2020年）使用了两种不同的显著性分数。这些显著性分数可能存在的一个问题是，它们与训练初期的几个步骤相关，因此对于训练的后期阶段而言，可能并不是很好的选择。是否存在一种与整个训练动态更直接相关的显著性分数?

*通讯作者。

1.1 我们的方法

近期，已有许多关于神经网络全局优化的研究；参见，例如 Liang 等人（2018b；a；2019；2021）；Venturi 等人（2018）；Safran 和 Shamir（2017）；Li 等人（2022）；Ding 等人（2022）；Soltanolkotabi 等人（2019）；Sun 等人（2020a）；Lin 等人（2021a；b）；Zhang 等人（2021），以及综述 Sun 等人（2020b）；Sun（2020）。其中一条研究路线采用神经切线核（NTK）（Jacot 等人，2018）来描述网络规模足够大时深度神经网络的梯度下降动态。更具体地说，对于规模足够大的深度神经网络，神经切线核在训练过程中渐近保持不变，其收敛行为可以由神经切线核的谱刻画。该理论表明，神经切线核的谱可能是衡量整个训练动态的合理指标，而不只是衡量最初几次迭代。因此，自然可以考虑以下概念性剪枝方法：剪除对神经切线核谱影响最小的连接。

在实施这种概念性的剪枝方法时，有几个问题需要考虑。

首先，应计算什么度量指标？计算神经切线核的完整特征谱过于耗时。遵循数值线性代数和深度学习领域的实践（Lee 等人，2019a；Xiao 等人，2020），我们使用核范数（特征值之和）作为谱的标量指标。

其次，应选择哪个“神经切线核”矩阵？我们将针对给定架构并采用随机初始化所定义的神经切线核矩阵称为“固定权重神经切线核”，并使用“解析神经切线核”（Jacot 等人，2018）来指代网络宽度趋于无穷时固定权重神经切线核的渐近极限。解析神经切线核是神经切线核理论（Jacot 等人，2018）所研究的对象，我们认为，在整个训练过程中，其谱可以作为某种架构性能的指标。[1] 然而，计算其核范数仍然过于耗时（无论是使用 Jacot 等人（2018）给出的解析形式，还是处理超宽网络）。固定权重神经切线核的核范数易于计算，但固定权重神经切线核可能与解析神经切线核存在很大差异。为解决这一问题，我们注意到一个较少被提及的事实：解析神经切线核也是固定权重神经切线核关于随机权重的期望的极限[2]，因此，对于给定宽度，可以通过固定权重神经切线核的期望对其进行近似。相比单个固定权重神经切线核，固定权重神经切线核的期望应当能够更好地近似解析神经切线核。当然，为了估计该期望，我们可以使用少量权重配置作为“样本”，并计算若干固定权重神经切线核的平均值。

还存在一个可能的问题：例如，计算 100 个固定权重神经切线核是否需要多花 100 倍的计算成本？我们采用另一种计算技巧来降低计算成本：对于每个输入小批次，我们使用一个新的权重配置样本来计算一个固定权重神经切线核（更准确地说，是计算其核范数）。与使用 100 个小批次、针对同一个权重配置计算固定权重神经切线核相比，这种做法不会增加计算成本。我们将这一技巧称为“新输入新权重”（NINW）。

我们将提出的前瞻性剪枝算法命名为**神经切线核谱-感知**剪枝（**NTK-SAP**）。我们展示了 NTK-SAP 在多个数据集上的竞争力，其中包括 CIFAR-10、CIFAR-100、Tiny-ImageNet 和 ImageNet。总而言之，我们的贡献如下：

- 我们提出了一种受理论启发的前瞻剪枝方法，名为 NTK-SAP。该方法根据神经切线核的谱对网络进行剪枝。
- 我们引入了一种多重采样公式，使用不同的权重配置来更好地捕捉剪枝后的神经网络的预期行为。我们利用“新输入新权重”（NINW）技巧来降低计算成本，该技巧也可能具有独立的研究价值。
- 通过实证研究，我们表明，NTK-SAP作为一种与数据无关的前瞻性剪枝方法，在多种设置下均达到了最先进的性能。

2 相关工作与背景

训练后的剪枝（事后剪枝）。事后剪枝可以追溯到20世纪80年代（Janowsky，1989；Mozer & Smolensky，1989），通常需要多轮训练-剪枝-重新训练流程（Han et al.，2015；LeCun et al.，1989）。这些剪枝方法大多使用

[1]有关更多经验层面的讨论，请参见附录 B.2。[2]这是一个微妙之处，更多讨论请参见附录 B.1。

幅值（Han 等，2015）、海森矩阵（LeCun 等，1989；Hassibi 与 Stork，1992；Molchanov等，2019）以及其他信息（Dai 等，2018年；Guo 等，2016；Dong等，2017；Yu 等，2018年）来定义评分指标，以确定需要剪枝的权重。一些其他近期工作（Verma与Pesquet，2021）通过迭代优化找出稀疏网络。大多数事后剪枝方法旨在减少推理时间，但彩票票据剪枝（Frankle 与Carbin，2018年；Renda 等，2020年）除外，其剪枝网络的目的是寻找可训练子网络。

训练期间剪枝。另一类剪枝算法会在训练过程中逐步将稠密神经网络稀疏化。一些工作采用显式的 ℓ_0（Louizos 等人，2017年）或 ℓ_1（Wen 等人，2016年）正则化来学习稀疏解。其他工作引入可训练掩码（Liu 等人，2020年；Savarese 等人，2020年；Kang 和 Han，2020年；Kusupati 等人，2020年；Srinivas 等人，2017年；Xiao 等人，2019年），或使用投影梯度下降（Zhou 等人，2021）来专门学习需要保留的连接。动态稀疏训练方法（Mocanu 等人，2018年；Bellec 等人，2017年；Mostafa 和 Wang，2019年；Dettmers 和 Zettlemoyer，2019年；Liu 等人，2021b；Evci 等人，2020年；Liu 等人，2021c；a）在固定参数数量预算下训练神经网络，同时在整个训练过程中逐步调整参数。此外，另一项工作（You 等人，2019年）尝试在早期训练轮次识别中奖彩票。这些方法可以减少部分训练时间，但通常无法降低训练的内存开销。

训练前剪枝（前瞻剪枝）。SNIP（Lee 等人，2018年）可谓是最早的前瞻剪枝方法。之后，GraSP（Wang 等人，2020年）利用海森矩阵-梯度乘积来设计剪枝度量。为了解决 SNIP 和 GraSP 在稀疏率较高时出现的层坍塌问题，Synflow（Tanaka 等人，2020年）以与数据无关的方式迭代剪枝网络。FORCE 和迭代 SNIP（de Jorge 等人，2020a）也旨在提升 SNIP 在高稀疏率下的性能。Lee 等人（2019b）采用正交约束，改善经 SNIP 剪枝后的网络的信号传播。神经切线迁移（Liu & Zenke，2020年）明确地使子网络学习稠密网络完整的经验神经切线核。PropsPr（Alizadeh 等，2022）也指出了现有前瞻性剪枝方法的“短视”问题，并提出利用元梯度来使用前几次迭代中的信息。

3 前瞻性剪枝方法的通用流程

在本节中，我们回顾初始化时剪枝问题的形式化定义，并重新审视几种现有的前瞻剪枝方法。这里我们考虑Synflow（Tanaka 等人，2020年）、GraSP（Wang 等人，2020年）、SNIP（Lee 等人，2018年）和迭代SNIP（de Jorge 等，2020b）。给定一个数据集$\mathcal{D} = (\mathbf{x}_i, \mathbf{y}_i)_{i=1}^n$和一个未经训练的神经网络 $f(\cdot; \boldsymbol{\theta})$，其参数化形式为 $\boldsymbol{\theta} = \{\theta^1, \theta^2, \cdots, \theta^p\} \in \mathbb{R}^p$，我们希望找到一个二值掩码$\mathbf{m} = \{m^1, m^2, \cdots, m^p\} \in \{0,1\}^p$，使其最小化以下方程：

$$\min_{\mathbf{m}} \mathcal{L}(\mathcal{A}(\boldsymbol{\theta}_0, \mathbf{m}) \odot \mathbf{m}; \mathcal{D}) = \min_{\mathbf{m}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(\mathbf{x}_i; \mathcal{A}(\boldsymbol{\theta}_0, \mathbf{m}) \odot \mathbf{m}), \mathbf{y}_i) \tag{1}$$
$$\text{s.t.} \quad \mathbf{m} \in \{0,1\}^p, \quad \|\mathbf{m}\|_0 / p \leq d = 1 - k$$

其中，k和 $d = 1 - k$是目标稀疏度和密度，$\boldsymbol{\theta}_0$是遵循特定权重初始化方法 $\mathcal{P}$（例如 Xavier 初始化）的初始化点，$\odot$ 表示 Hadamard乘积，而 $\mathcal{A}$ 表示一种训练算法（例如 Adam），该算法接收掩码$\mathbf{m}$ 以及初始化 $\boldsymbol{\theta}_0$，并在收敛时输出训练后的权重 $\boldsymbol{\theta}_{\text{final}} \odot \mathbf{m}$。

由于直接最小化公式 (1) 难以实现，现有的前瞻性剪枝方法会在初始化时为每个掩码及其对应的权重 θ_0^j 赋予一个显著性分数 $S(\theta_0^j)$，其一般形式如下：

$$S(m^j) = S(\theta_0^j) = \frac{\partial \mathcal{I}}{\partial m^j} = \frac{\partial \mathcal{I}}{\partial \theta_0^j} \cdot \theta_0^j \tag{2}$$

其中，$\mathcal{I}$ 是参数 $\boldsymbol{\theta}_0$ 和对应掩码$\mathbf{m}$的函数。直观地说，公式（2）衡量了移除连接 m^j 对 $\mathcal{I}$的影响。计算出每个掩码的显著性分数后，我们将只保留排名前d 的掩码。具体而言，显著性分数最高的掩码将被保留（$m^j = 1$），而其他掩码将被剪枝（$m^j = 0$）。

SNIP. SNIP（Lee 等人，2018年）定义 $S_{\text{SNIP}}(m^j) = \left| \frac{\partial \mathcal{L}(\boldsymbol{\theta}_0 \odot \boldsymbol{m};\ D)}{\partial \theta_0^j} \cdot \theta_0^j \right|$，其中 $D \subset \mathcal{D}$ 是整个训练集的一个子集。SNIP 在一轮中剪枝整个网络。

迭代 SNIP。迭代 SNIP[3]（de Jorge 等，2020b）的显著性分数与原始 SNIP 相同。不过，它会执行多轮剪枝。

GraSP。GraSP（Wang 等人，2020）定义了 $S_{\text{GraSP}}(m^j) = -\left[H(\boldsymbol{\theta}_0 \odot \boldsymbol{m};\ D) \frac{\partial \mathcal{L}(\boldsymbol{\theta}_0 \odot \boldsymbol{m};\ D)}{\partial \boldsymbol{\theta}_0} \right]^j \cdot \theta_0^j)$，其中 H 是海森矩阵。此外，GraSP 也是一种单轮前瞻剪枝方法。

Synflow。Synflow（Tanaka 等人，2020）使用 $S_{\text{Synflow}}(m^j) = \left| \frac{\partial \sum_k f_{(\mathbb{1}\ |\boldsymbol{\theta}_0| \odot \boldsymbol{m}\ k}}{\partial |\theta_0^j|} \cdot |\theta_0^j| \right|$;)作为显著性分数，其中 f_k 表示第 k个输出 logit，$\mathbb{1}$ 表示一个全为 1 的输入，而 $f(\cdot;\ |\theta_0^j|)$ 表示将神经网络的所有参数替换为其绝对值。与迭代 SNIP 类似，Synflow 也是一种迭代剪枝方法，但采用与数据无关的方式。

4 神经正切核综述

近期研究（Jacot 等人，2018年；Lee 等人，2019a；Arora 等人，2019年；Xiao 等人，2020年）探讨了大型深度神经网络在梯度下降下的训练动态。形式上，令 $\mathcal{D} = \{(\mathrm{x}_i, \mathrm{y}_i)\}_{i=1}^n \subset \mathbb{R}^{n_0} \times \mathbb{R}^k$ 表示训练集，并令 $\mathcal{X} = \{\mathrm{x} : (\mathrm{x}, \mathrm{y}) \in \mathcal{D}\}$、$\mathcal{Y} = \{\mathrm{y} : (\mathrm{x}, \mathrm{y}) \in \mathcal{D}\}$ 分别表示输入和标签。考虑一个由权重 $\boldsymbol{\theta}$ 参数化的神经网络 $f(\cdot;\ \boldsymbol{\theta})$。令 $\mathcal{L}(\boldsymbol{\theta}) = \sum_{(\mathrm{x},\mathrm{y}) \in (\mathcal{X},\mathcal{Y})} \ell(f(\mathrm{x};\ \boldsymbol{\theta}), \mathrm{y})$ 表示经验损失，其中 ℓ 是损失函数。因此，$f(\mathcal{X};\ \boldsymbol{\theta}) \in \mathbb{R}^{kn \times 1}$ 是所有训练输入对应预测值的拼接向量。在学习率为 η 的连续时间梯度下降下，参数 $\boldsymbol{\theta}_t$ 的演化过程（其中我们强调 $\boldsymbol{\theta}$ 对时间 t 的依赖关系），以及神经网络 $\mathcal{L}$ 的训练损失，可表示为（Lee 等人，2019a）：

$$\dot{\boldsymbol{\theta}}_t = -\eta \nabla_{\boldsymbol{\theta}_t} f(\mathcal{X}; \boldsymbol{\theta}_t)^T \nabla_{f(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L} \tag{3}$$

$$\dot{\mathcal{L}} = \nabla_{f(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L}^T \nabla_{\boldsymbol{\theta}_t} f(\mathcal{X}; \boldsymbol{\theta}_t) \dot{\boldsymbol{\theta}}_t = -\eta \nabla_{f(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L}^T \hat{\boldsymbol{\Theta}}_t(\mathcal{X}, \mathcal{X}) \nabla_{f(\mathcal{X};\boldsymbol{\theta}_t)} \mathcal{L} \tag{4}$$

其中我们在时间 t获得神经切线核:

$$\hat{\boldsymbol{\Theta}}_t(\mathcal{X}, \mathcal{X}; \boldsymbol{\theta}_t) \triangleq \nabla_{\boldsymbol{\theta}_t} f(\mathcal{X}; \boldsymbol{\theta}_t) \nabla_{\boldsymbol{\theta}_t} f(\mathcal{X}; \boldsymbol{\theta}_t)^T \in \mathbb{R}^{kn \times kn}. \tag{5}$$

已有证明表明，当神经网络的宽度变得足够大时（Jacot等，2018年；Arora等，2019年），固定权重神经切线核会收敛到一个确定性矩阵 $\boldsymbol{\Theta}(\mathcal{X}, \mathcal{X})$，我们将其称为解析神经切线核。解析神经切线核在整个训练过程中保持渐近恒定，即$\boldsymbol{\Theta}_t(\mathcal{X}, \mathcal{X}) \approx \boldsymbol{\Theta}_0(\mathcal{X}, \mathcal{X})\ \ \forall t \geq 0$。

神经切线核的条件数与神经网络优化相关。神经切线核的条件数被认为在神经网络优化中十分重要。具体而言，设初始化时神经切线核的特征值分解为 $\boldsymbol{\Theta}_0(\mathcal{X},\ \mathcal{X}) = U^T \Lambda U$其中 Λ =diag($\lambda_1,\ \cdots,\ \lambda_{kn}$) 是包含神经切线核所有特征值的对角矩阵。

对于大宽度网络，神经网络在神经切线核的特征基下的平均预测可以近似描述为 $(U\mathbb{E}[f(\mathcal{X})])_i = (\mathbf{I} - e^{-\eta\lambda_i t})(U\mathcal{Y})_i$(Xiao 等人, 2020年)。因此，当使用尽可能大的学习率 $\eta \sim 2/\lambda_1$ (Lee等人, 2019a)时，最小特征值的收敛速率与 λ_1/λ_{kn}有关，而 λ_1/λ_{kn}是神经切线核的条件数。在经验上，神经切线核的条件数已成功用于在神经架构搜索（神经架构搜索）领域识别有前景的架构（Chen 等人, 2021; Wang 等人, 2022年）。

神经切线核的完整谱是更好的衡量指标。然而，最小和最大特征值只能提供关于深度神经网络优化特性的有限信息。在经典优化理论中，收敛速度取决于海森矩阵的所有特征值，因为不同的特征值控制着不同特征模态的收敛。类似地，对于神经网络，一些研究（Kopitkov & Indelman, 2020；Su & Yang, 2019）表明，收敛速度取决于最小特征值以外的其他特征值。从这个意义上说，与只考虑神经切线核的条件数相比，控制整个特征谱可能是确保深度神经网络获得良好优化性能的更好方法。

[3]由于我们通过实验证发现，FORCE 的稳定性不如迭代 SNIP，因此本文不考虑 FORCE。详细比较请参见附录 F。

5 我们的方法

5.1 使用迹范数考虑整个特征谱的影响

如第 4 节所述，神经切线核的特征谱捕捉了大型深度神经网络的训练动态，因此我们打算利用神经切线核的谱来指导前瞻剪枝。具体而言，我们希望让剪枝后网络的谱接近大型网络的谱，从而使剪枝后网络能够取得与大型网络相似的表现。

正如第3节所讨论的，基于显著性分数的方法需要使用单一度量指标来捕捉优化性能。这一单一度量指标通常是一个标量，因此我们希望生成一个标量来捕捉整个谱的信息。一个非常自然的选择是使用所有特征值的平均值。事实上，Pennington 等人（2017；2018年）；Shu等人（2021）确实通过NTK的平均特征值（或一种称为输入-输出雅可比矩阵的相关矩阵）显式或隐式地分析了深度神经网络的优化性质。此外，使用平均特征值还具有一些良好的性质，使其比其他度量指标更适合用于剪枝：

计算效率。首先，计算所有特征值的计算成本过高。相反，迹范数可以高效计算，我们将在第 5.2 节中展示这一点。

可能保留完整的谱。 如第3节所示，每个权重的显著性分数衡量的是移除该权重时度量指标（此处为神经切线核的迹范数）的变化。更确切地说，移除显著性分数较低的权重不太可能导致特征值分布发生显著变化。因此，使用神经切线核的迹范数作为指标有助于保留完整的谱。

5.2 近似计算神经切线核的迹范数

以核范数作为整体谱的代理指标。 神经切线核是一个对称矩阵。因此，神经切线核的迹等价于参数到输出的雅可比矩阵的 Frobenius 范数的平方，即

$$\|\boldsymbol{\Theta}_0\|_* = \|\boldsymbol{\Theta}_0\|_{\mathrm{tr}} = \|\nabla_{\boldsymbol{\theta}_0} f(\mathcal{X}; \boldsymbol{\theta}_0)\|_F^2. \tag{6}$$

有限差分近似。 计算雅可比矩阵 $\nabla_{\boldsymbol{\theta}_0} f(\mathcal{X};\ \boldsymbol{\theta}_0) \in \mathbb{R}^{kn \times p}$ 的 Frobenius 范数需要大量内存和计算资源。具体而言，对于一批训练数据，显式构造雅可比矩阵的朴素方法可能至少需要 k轮反向传播，其中 k 在 ImageNet 分类任务中可高达 1000。因此，需要采用近似方法。

一种可能的方法是利用损失相对于权重的梯度对其进行下界估计 $\|\nabla_{\boldsymbol{\theta}_0} \mathcal{L}\|_2^2$。FORCE（deJorge等，2020b）研究了这种方法，但发现其效果不如迭代 SNIP。

或者，我们使用有限差分表达式 $\frac{1}{\epsilon} \mathbb{E}_{\Delta\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \epsilon \mathbf{I})} \left[\| f(\mathcal{X}; \boldsymbol{\theta}_0) - f(\mathcal{X};\ \boldsymbol{\theta}_0 + \Delta\boldsymbol{\theta}) \|_2^2 \right]$来近似雅可比范数。其论证请参见附录 O。

使用剪枝集近似训练集。我们使用以下有限近似表达式来定义显著性分数：

$$S_{\text{NTK-SAP}}(m^j) = \left| \frac{\partial \mathbb{E}_{\Delta\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, \epsilon \mathbf{I})} \left[\| f(\mathbf{X}_D; \boldsymbol{\theta}_0 \odot \mathbf{m}) - f(\mathbf{X}_D; (\boldsymbol{\theta}_0 + \Delta\boldsymbol{\theta}) \odot \mathbf{m}) \|_2^2 \right]}{\partial m^j} \right| \tag{7}$$

其中，我们使用剪枝数据集的输入 $\mathbf{X}_D$ 来近似整个训练输入 $\mathcal{X}$。我们的前瞻剪枝方法将采用第3节所示的相同框架，剪去公式（7）中定义的显著性分数最低的权重。不过，要使该方法切实可行，还需要一些额外技巧，下面将对此进行讨论。

5.3 NTK-SAP：多重采样公式

上一小节使用固定权重神经切线核计算显著性分数。我们认为，解析神经切线核的谱可以作为衡量特定架构性能的指标。单个固定权重神经切线核可视为对解析神经切线核的近似，而其期望

（对随机抽取的权重配置取期望）应当是更好的近似。此外，为了近似该期望，我们对权重配置进行若干次独立采样，并计算相应固定权重神经切线核的平均值。这有助于探索参数空间，并更好地捕捉剪枝后的神经网络的预期行为。

具体而言，对于 R 随机权重配置 $\boldsymbol{\theta}_{0,\ r \text{ iid}\sim} \mathcal{P}$，我们使用以下稳定化版本的显著性分数：

$$S_{\text{NTK-SAP}}(m^j) = \left| \frac{\partial \frac{1}{R} \sum_{r=1}^{R} \mathbb{E}_{\Delta\boldsymbol{\theta}\sim\mathcal{N}(\mathbf{0},\epsilon\mathbf{I})} \left[\| f(\mathbf{X}_D; \boldsymbol{\theta}_{0,r} \odot \mathbf{m}) - f(\mathbf{X}_D; (\boldsymbol{\theta}_{0,r} + \Delta\boldsymbol{\theta}) \odot \mathbf{m}) \|_2^2 \right]}{\partial m^j} \right|. \tag{8}$$

所提出的显著性分数与现有的大多数前瞻剪枝分数有一个重要区别：我们的分数与权重无关，即它并非由某个特定的随机权重配置抽样所定义。换言之，该分数主要由掩码结构决定，而非具体权重。[4]

新输入新权重（NINW）技巧。 看起来，使用多个权重配置会大幅增加计算成本。不过，利用剪枝训练子集 $D = (\mathbf{x}_i,\ \mathbf{y}_i)_{i=1}^{|D|} \subset \mathcal{D}$ 通常会以多个批次输入神经网络这一事实，我们进一步用 NINW 技巧近似公式 (8)：

$$S_{\text{NTK-SAP}}(m^j) = \left| \frac{\partial \frac{1}{|D|} \sum_{i=1}^{|D|} \mathbb{E}_{\Delta\boldsymbol{\theta}\sim\mathcal{N}(\mathbf{0},\epsilon\mathbf{I})} \left[\| f(\mathbf{x}_i; \boldsymbol{\theta}_{0,i} \odot \mathbf{m}) - f(\mathbf{x}_i; (\boldsymbol{\theta}_{0,i} + \Delta\boldsymbol{\theta}) \odot \mathbf{m}) \|_2^2 \right]}{\partial m^j} \right|. \tag{9}$$

用随机输入替换剪枝集。 此外，我们发现，用标准高斯噪声 $\mathcal{Z} \sim \mathcal{N}(\mathbf{0},\ \mathbf{I})$ 替换训练子集D可以获得相当的性能，这使得 NTK-SAP 完全不依赖数据。至于高斯扰动，理想情况下，使用多次抽样确实能得到更准确的近似。然而，我们通过实证发现，单次高斯扰动抽样就足以获得令人满意的性能。因此，通过使用一次随机抽样 $\Delta\boldsymbol{\theta}$，我们得到最终的数据无关 NTK-SAP 及其显著性分数

$$S_{\text{NTK-SAP}}(m^j) = \left| \frac{\partial \frac{1}{|D|} \sum_{i=1}^{|D|} \left[\| f(\mathbf{z}_i; \boldsymbol{\theta}_{0,i} \odot \mathbf{m}) - f(\mathbf{z}_i; (\boldsymbol{\theta}_{0,i} + \Delta\boldsymbol{\theta}_i) \odot \mathbf{m}) \|_2^2 \right]}{\partial m^j} \right| \tag{10}$$

其中 $\boldsymbol{\theta}_{0,\ i \text{ 独立同分布}\sim} \mathcal{P}$，$\Delta\boldsymbol{\theta}_{i \text{ 独立同分布}\sim} \mathcal{N}(\mathbf{0},\ \epsilon\mathbf{I})$，且 $\mathbf{z}_{i \text{ 独立同分布}\sim} \mathcal{N}(\mathbf{0},\ \mathbf{I})$。

此外，为了尽可能避免层坍塌，我们也像 Synflow 一样，分 T 轮对神经网络进行迭代剪枝。[5] NTK-SAP 算法的主要步骤见算法 1。

计算成本。我们将 NTK-SAP 与迭代 SNIP 的计算成本进行比较，因为二者都是迭代方法，且都使用多批次输入。由于重新初始化过程的开销可以忽略不计，NTK-SAP 需要的计算成本大约是迭代 SNIP 的两倍，这是因为公式中包含两次带计算图的前向传播。不过需要注意的是，NTK-SAP 是一种与数据无关的方法，可以预先计算；此外，采用更小的 T 可以减少计算时间，因此其计算开销是合理的。关于 T 的影响、剪枝时间的详细比较以及浮点运算次数的比较，分别见第 6.4 节、附录 D.1 和附录 D.2。

6 实验

在本节中，我们通过实验证明所提出的方法 NTK-SAP（红色）的有效性。我们将其与两种基线方法随机剪枝（灰色）和幅值剪枝（洋红色）进行比较，如

[4]关于优秀的剪枝方案是否应与权重无关，请参阅附录 M。[5]关于 NTK-SAP 为何有可能避免层坍塌，请参阅附录 N。

算法 1神经切线核频谱感知剪枝（NTK-SAP）

要求： 稠密网络 $f(\cdot;\boldsymbol{\theta}\odot\mathbf{m})$，最终密度 d，迭代步数 T，噪声输入的批次大小 B，扰动超参数 ϵ

初始化 $\mathbf{m} = \mathbb{1}$
对 t 在 $[1,\cdots,T]$ **执行**
  对于 i 中 $[1,\cdots,B]$ **执行**
    使用 $\boldsymbol{\theta}_0$ 重新初始化神经网络 $,_i \overset{\text{iid}}{\sim} \mathcal{P}$
    样本噪声输入 $\mathbf{z}_i \overset{\text{iid}}{\sim} \mathcal{N}(\mathbf{0},\mathbf{I})$ 和参数扰动 $\Delta\boldsymbol{\theta}_i \overset{\text{iid}}{\sim} \mathcal{N}(\mathbf{0},\epsilon\mathbf{I})$
    根据公式（10）评估显著性分数 $[\mathcal{S}(\mathbf{m})]_i$
  结束循环
  计算 $\mathcal{S}(\mathbf{m}) = \left| \sum_{i=1}^{B} [\mathcal{S}(\mathbf{m})]_i \right|$
  计算 $(1-d^{\frac{t}{T}})$ $\mathcal{S}(\mathbf{m})$ 的百分位数作为阈值 τ
  更新掩码 $\mathbf{m} \leftarrow \mathbf{m} \odot (\mathcal{S}(\mathbf{m}) > \tau)$
结束循环
返回 最终掩码 $\mathbf{m}$

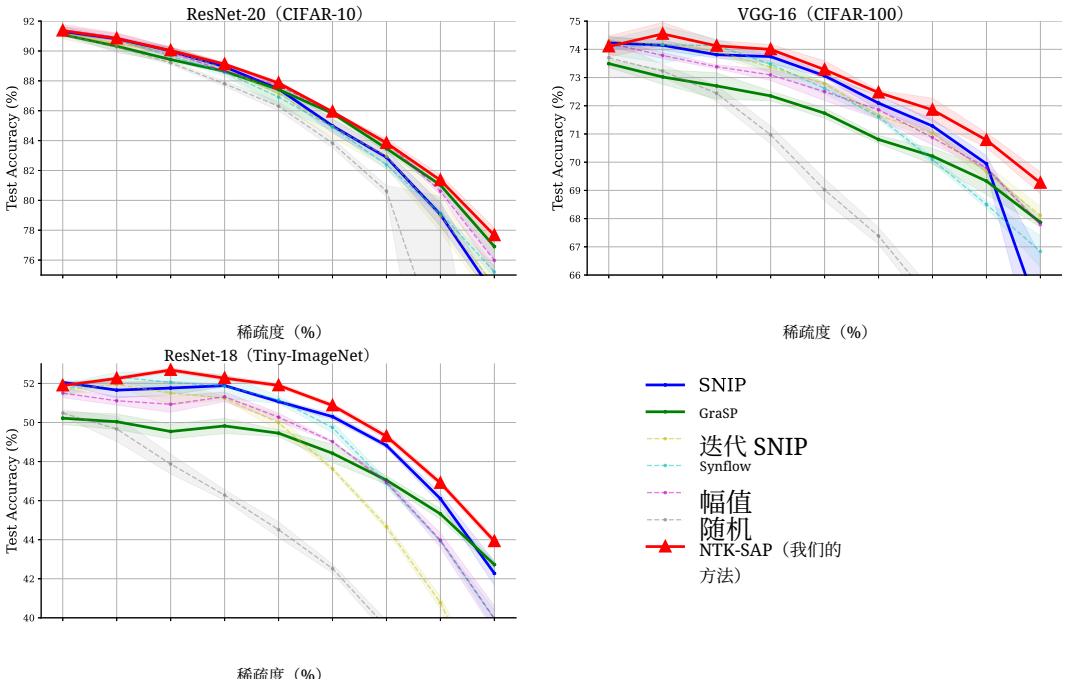


图 1：NTK-SAP 与其他前瞻剪枝方法的性能。结果取 3 个随机种子的平均值，阴影区域表示标准差。我们重点展示 SNIP 和 GraSP，因为它们分别在较温和和极端的密度比下表现最佳，优于其他基线方法。

以及前瞻性剪枝算法 SNIP（蓝色） (Lee 等人，2018年)、迭代 SNIP（黄色） (deJorge 等人，2020b)、GraSP（绿色） (Wang 等人，2020年) 和Synflow（青色）在多种架构和数据集上的表现。

目标数据集、模型和稀疏率。我们的代码基于 Synflow（Tanaka 等人，2020年）的原始代码开发。在较小的数据集上，我们在 CIFAR-10 上使用 ResNet-20（He 等人，2016年），在 CIFAR-100（Krizhevsky 等人，2009年）上使用 VGG-16（Simonyan 和 Zisserman，2014年），并在 Tiny-ImageNet 上使用 ResNet-18。在 ImageNet（Deng 等人，2009年）实验中，我们使用 ResNet-18 和 ResNet-50。训练方案大致遵循 Frankle 等人（2020b）中的设置。训练超参数的详细信息见附录 A。我们将稀疏率分为三个范围：极低稀疏率（36.00%、59.04% 和 73.80%）、中等稀疏率（83.22%、89.30% 和 93.12%）以及极端稀疏率（95.60%、97.17% 和 98.20%）。

实现细节。根据原始实现，SNIP、GraSP 和迭代 SNIP 在将 BatchNorm 层设为训练模式的情况下运行，而 Synflow 指标则在评估模式下进行评估。我们使用的用于剪枝的训练样本数量是类别数的十倍，这与 SNIP 和 GraSP 的原始实现建议一致。对于包括迭代 SNIP 和 Synflow 在内的迭代方法，我们遵循 Tanaka 等（2020年）和 Frankle 等人（2020b）的做法，采用指数调度，在 100 轮中对所有权重进行剪枝。此外，遵循 Frankle 等人（2020b）的做法，所有模型均使用 Kaiming 正态初始化。对于 CIFAR-10/100 和 Tiny-ImageNet 数据集，我们使用批次大小为 256 的批次对网络进行剪枝；对于 ImageNet 实验，则使用批次大小为 128 的批次。

对于 NTK-SAP，我们使用与迭代 SNIP 相同数量的高斯噪声批次，并针对 CIFAR-100 和 Tiny-ImageNet 在 $T = 100$ 轮中逐步对神经网络进行剪枝。由于 CIFAR-10 的剪枝数据集仅包含 1 个训练样本批次，因此我们改为采样$R = 5$ 个独立的初始化点，并在 $T = 20$ 轮中逐步进行剪枝，同时使总体计算成本大致保持不变。更多详情请参见附录 A。

6.1 CIFAR 和 Tiny-ImageNet 上的结果

在图 1中，我们分别使用 CIFAR-10 上的 ResNet-20、CIFAR-100 上的 VGG-16 以及Tiny-ImageNet 上的 ResNet-18 对NTK-SAP 进行了评估。可以观察到，NTK-SAP 在所有稀疏率下都具有竞争力。

极低稀疏率和中等稀疏率。对于极低稀疏率，除 GraSP、幅值和随机剪枝外，所有方法都表现良好。在 ResNet-20 上，所有方法表现相近，而随机剪枝和 GraSP 的表现略差。在 VGG-16（ResNet-18）上，可以观察到类似趋势，但性能差距更大。NTK-SAP 始终表现良好，尤其是在 ResNet-18（TinyImageNet）上。

对于中等稀疏率，NTK-SAP 是在所有数据集和模型上都表现良好的唯一方法。在 ResNet-20 上，SNIP 和迭代SNIP 与 NTK-SAP一同领先，但从93.12% 的稀疏度开始性能下降。超过 89.30% 后，幅值剪枝和 GraSP 的性能与 NTK-SAP 相近。在 VGG-16和 ResNet-18 上，NTK-SAP 和 SNIP 仍占据主导地位，而 NTK-SAP 的表现优于 SNIP。

极端稀疏率。 NTK-SAP 的卓越性能在极端稀疏率下尤为明显，尤其是在 98.20% 时。在 CIFAR-10 数据集上，GraSP 和幅值剪枝在相同稀疏率下表现相当。在 VGG-16 上，S NIP 起初与 NTK-SAP 不相上下，之后则明显落后。在最稀疏的 98.20% 水平下，幅值、迭代 SNIP 和 GraSP 的结果相近，比 NTK-SAP 低约 1 个百分点。

6.2 更大型数据集上的结果

我们进一步评估 NTK-SAP 在更困难任务上的性能：我们使用 ResNet-18 和 ResNet-50，在 ImageNet数据集上以剪枝率 {89.26%, 95.60%} 进行实验。[6] 我们将 NTK-SAP与 Synflow、SNIP、迭代 SNIP、GraSP、幅值和随机剪枝进行比较。

结果如表 1 所示。我们发现，SNIP在较小数据集上表现尚可，但已不再具有竞争力。对于 ResNet-50，在稀疏率分别为89.26% 和95.60% 时，NTK-SAP 相较于其他剪枝方法的优势至少为 0.5%和 1%。对于 ResNet-18，在 89.26%的稀疏度下，NTK-SAP 略优于幅值剪枝，GraSP紧随其后。在稀疏率为95.60% 时，GraSP 优于幅值剪枝，而 NTK-SAP 仍然领先。

需要注意的是，幅值剪枝取得了出人意料的良好表现。我们推测，幅值剪枝选择了恰当的逐层稀疏率，因为Kaiming 正态初始化已经根据不同层的参数数量，为权重赋予了不同的方差。因此，参数更多的层更有可能被剪枝。

6.3 与 SNIP 和 GraSP 的整体比较

In t本节中，我们详细地将 NTK-SAP 与两种有竞争力的基线方法进行明确比较，即 ., SNIP 和 GraSP，因为没有最先进的前瞻剪枝算法（Frankle 等，2020b).

[6]我们选择这些稀疏率，因为它们涵盖了匹配的稀疏率和极端稀疏率，并且接近Iterative SNIP（de Jorge等，2020b）和 ProsPr（Alizadeh 等，2022）所使用的 {90%, 95%}。

表 1：ImageNet 数据集上前瞻剪枝方法的测试性能。最佳结果以**粗体**显示；次佳结果带下划线。

网络	ResNet-18		ResNet-50	
稀疏率	89.26%	95.60%	89.26%	95.60%
（稠密基线）	69.98		76.20	
Synflow	57.30	45.65	66.81	58.88
SNIP	57.41	45.04	60.98	40.69
迭代 SNIP	52.97	37.44	52.53	36.82
GraSP	58.16	<u>49.17</u>	<u>67.74</u>	<u>59.73</u>
幅值	<u>58.75</u>	48.50	66.80	43.79
随机	54.48	42.80	65.30	56.53
我们的方法	**58.87**	**49.43**	**68.28**	**60.79**

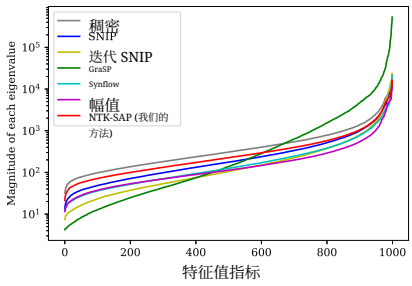


图 2：ResNet-20（CIFAR-10）在稀疏率为 73.80% 时的固定权重神经切线核特征值分布。

我们在图 1中重点介绍了这两种基线算法。具体而言，SNIP被认为具有竞争力，因为其通常表现良好，尤其是在简单和中等稀疏率下。然而，SNIP 在 ImageNet上表现不佳。相反，GraSP在 ImageNet数据集以及较小数据集上的极端稀疏率下确实展现出了良好的性能，但在较小数据集的简单和中等稀疏率下表现不佳。从这个意义上说，NTK-SAP 是首个在所有数据集上都能始终保持良好表现的前瞻性剪枝算法。

6.4 更多消融研究

最后，我们评估了改变迭代步数（T）时 NTK-SAP 的性能，以便更好地了解迭代剪枝如何帮助 NTK-SAP 在更高稀疏率下取得更好的性能。我们在不同的 T 下重复了第 6.1 节中的实验。结果表明，增加 T 能持续提升 NTK-SAP 的性能。此外，当 T 大约达到第 6.1 节所采用值的 $1/5$ 时，这种提升趋于饱和。因此，若采用较小的 T，便可进一步降低 NTK-SAP 的计算成本。更多详情请参见附录 C。我们还针对扰动超参数 ϵ 的影响进行了消融研究。结果表明，NTK-SAP 对 ϵ 的取值具有鲁棒性。更多详情请参见附录 I。

6.5 剪枝后固定权重神经切线核的特征谱

为了直观展示我们提出的 NTK-SAP 方法的有效性，我们展示了使用 ResNet-20 在 CIFAR-10 数据集上、稀疏率为 73.80% 时剪枝子网络的特征谱。图 2 表明，与其他方法相比，NTK-SAP 成功保留了固定权重神经切线核的特征谱。其他稀疏率下固定权重神经切线核的特征谱请见附录 G.1。

7 结论

与训练后剪枝算法相比，初始化时剪枝颇具吸引力，因为它无需训练即可确定掩码。然而，现有的大多数前瞻性剪枝方法都没有捕捉剪枝后的训练动态。我们认为，可以通过使用神经切线核理论来缓解这一问题。因此，我们提出了 NTK-SAP，它能够在不使用任何数据的情况下迭代剪枝神经网络。此外，NTK-SAP 还考虑了多个随机抽取的初始化点。在实证方面，我们的方法在本文研究的所有数据集和模型上都具有竞争力。我们认为，我们的算法能够鼓励更多研究将理论发现应用于实证研究。此外，我们提出的结合多种初始化配置的新方法，或许还能进一步推动未来研究探讨剪枝与神经架构搜索之间的相似性，例如 Liu 等（2018）和 Abdelfattah 等（2021）的研究。

这项工作的局限性包括：与单次剪枝方法相比，迭代剪枝方法的计算开销相对较大。我们还想强调，剪枝方法对于降低深度学习模型训练和应用的能耗具有重要意义。

8 致谢

本研究利用了美国国家科学基金会重大研究仪器计划（资助编号：1725729；Kindratenko 等，2020）以及伊利诺伊大学厄巴纳-香槟分校提供的资源。本文部分得到河套深港科技创新合作区项目（编号：HZQSWS-KCCYB-2022046）、香港中文大学（深圳）大学发展基金 UDF01001491，以及广东省科学技术厅广东省人工智能数学基础重点实验室的资助。

我们感谢项目主席和领域主席处理我们的论文。我们也感谢审稿人投入时间和精力，为我们的论文提出建设性的建议。

参考文献

Mohamed S Abdelfattah、Abhinav Mehrotra、Łukasz Dudziak 和 Nicholas D Lane。轻量级神经架构搜索的零成本代理指标。*arXiv* 预印本 *arXiv:2101.08134*，2021年。Milad Alizadeh、Shyam A Tailor、Luisa M Zintgraf、Joost van Amersfoort、Sebastian Farquhar、Nicholas Donald Lane 和 Yarin Gal。前瞻剪枝：利用元梯度在初始化时找到可训练权重。*arXiv* 预印本 *arXiv:2202.08132*，2022年。Sanjeev Arora、Simon S Du、Wei Hu、Zhiyuan Li、Russ R Salakhutdinov 和 Ruosong Wang。关于无限宽神经网络的精确计算。神经信息处理系统进展，32，2019年。Guillaume Bellec、David Kappel、Wolfgang Maass 和 Robert Legenstein。深度重连：训练极稀疏的深度网络。*arXiv* 预印本 *arXiv:1711.05136*，2017年。Wuyang Chen、Xinyu Gong 和 Zhangyang Wang。在四小时图形处理器时间内于 ImageNet 上进行神经架构搜索：一种受理论启发的视角。*arXiv* 预印本 *arXiv:2102.11535*，2021年。

Bin Dai、Chen Zhu、Baining Guo 和 David Wipf。使用变分信息瓶颈压缩神经网络。载于国际机器学习会议，第 1135–1144 页。机器学习研究论文集，2018年。

Pau de Jorge、Amartya Sanyal、Harkirat S Behl、Philip HS Torr、Gregory Rogez 和 Puneet K Dokania。渐进式骨架化：在初始化时从网络中剔除更多冗余部分。*arXiv* 预印本 *arXiv:2006.09081*，2020年a。

Pau de Jorge、Amartya Sanyal、Harkirat S Behl、Philip HS Torr、Gregory Rogez 和 Puneet K Dokania。渐进式骨架化：在初始化时从网络中裁剪更多冗余。*arXiv* 预印本 *arXiv:2006.09081*，2020b。

Jia Deng、Wei Dong、Richard Socher、Li-Jia Li、Kai Li 和 Li Fei-Fei。《ImageNet：大型分层图像数据库》。载于 *2009 IEEE*计算机视觉与模式识别会议，第248–255页。IEEE，2009年。

Tim Dettmers 和 Luke Zettlemoyer。《从零开始构建稀疏网络：在不损失性能的情况下加快训练》。《*arXiv* 预印本 *arXiv:1907.04840*》，2019。

Tian Ding、Dawei Li 和 Ruoyu Sun。平滑激活函数下宽神经网络存在次优局部极小值。《运筹学数学》，47(4):2784–2814，2022年。Xin Dong、Shangyu Chen 和 Sinno Pan。通过逐层最优脑外科手术学习剪枝深度神经网络。神经信息处理系统进展，30，2017年。Utku Evci、Trevor Gale、Jacob Menick、Pablo Samuel Castro 和 Erich Elsen。操纵彩票：让所有票据都成为赢家。载于国际机器学习会议，第2943–2952页。机器学习研究论文集，2020年。Utku Evci、Yani Ioannou、Cem Keskin 和 Yann Dauphin。稀疏神经网络中的梯度流，以及彩票票据如何获胜。载于人工智能促进会会议论文集，第36卷，第6577–6586页，2022年。Jonathan Frankle 和 Michael Carbin。彩票票据假设：寻找稀疏且可训练的神经网络。*arXiv*预印本 *arXiv:1803.03635*，2018年。Jonathan Frankle、Gintare Karolina Dziugaite、Daniel Roy 和 Michael Carbin。线性模式连通性与彩票票据假设。载于国际机器学习会议，第3259–3269页。机器学习研究论文集，2020a。Jonathan Frankle、Gintare Karolina Dziugaite、Daniel M Roy 和 Michael Carbin。初始化时剪枝神经网络：为什么我们没有达到目标？*arXiv*预印本 *arXiv:2009.08576*，2020b。Yiwen Guo、Anbang Yao 和 Yurong Chen。用于高效深度神经网络的动态网络手术。神经信息处理系统进展，29，2016年。

Song Han、Jeff Pool、John Tran 和 William Dally。学习高效神经网络的权重与连接。神经信息处理系统进展，28，2015年。Babak Hassibi 和 David Stork。网络剪枝的二阶导数：最优脑外科医生。神经信息处理系统进展，5，1992年。何恺明、X. Zhang、任少卿和孙剑。用于图像识别的深度残差学习。*2016*年*IEEE*计算机视觉与模式识别会议（*CVPR*），第770–778页，2016年。Arthur Jacot、Franck Gabriel 和 Clément Hongler。神经切线核：神经网络中的收敛与泛化。神经信息处理系统进展，31，2018年。Steven A Janowsky。神经网络中的剪枝与裁剪。《物理评论*A*》，39(12):6600，1989年。Minsoo Kang 和 Bohyung Han。使用可微掩码的操作感知软通道剪枝。载于国际机器学习会议，第5122–5131页。机器学习研究论文集，2020年。Volodymyr Kindratenko、Dawei Mu、Yan Zhan、John Maloney、Sayed Hadi Hashemi、Benjamin Rabe、徐柯、Roy Campbell、彭健和 William Gropp。Hal：用于可扩展深度学习的计算机系统。载于高级研究计算实践与经验，第41–48页。2020年。Dmitry Kopitkov 和 Vadim Indelman。神经谱对齐：实证研究。载于国际人工神经网络会议，第168–179页。施普林格，2020年。Alex Krizhevsky、Geoffrey Hinton 等。学习微小图像中的多层特征。2009年。Aditya Kusupati、Vivek Ramanujan、Raghav Somani、Mitchell Wortsman、Prateek Jain、Sham Kakade 和 Ali Farhadi。用于可学习稀疏性的软阈值权重重参数化。载于国际机器学习会议，第5544–5555页。机器学习研究论文集，2020年。Yann LeCun、John Denker 和 Sara Solla。最优脑损伤。神经信息处理系统进展，2，1989年。Jaehoon Lee、Lechao Xiao、Samuel Schoenholz、Yasaman Bahri、Roman Novak、Jascha Sohl-Dickstein 和 Jeffrey Pennington。任意深度的宽神经网络在梯度下降下演化为线性模型。神经信息处理系统进展，32，2019a。Namhoon Lee、Thalaiyasingam Ajanthan 和 Philip HS Torr。SNIP：基于连接敏感度的单次网络剪枝。*arXiv*预印本 *arXiv:1810.02340*，2018年。Namhoon Lee、Thalaiyasingam Ajanthan、Stephen Gould 和 Philip HS Torr。从信号传播视角看初始化时的神经网络剪枝。*arXiv*预印本 *arXiv:1906.06307*，2019b。Dawei Li、Tian Ding 和 Ruoyu Sun。论宽度对神经网络的益处：吸引盆的消失。《*SIAM*优化杂志》，32(3):1728–1758，2022年。Shiyu Liang、Ruoyu Sun、Jason D Lee 和 R Srikant。添加一个神经元即可消除所有不良局部极小值。载于神经信息处理系统进展，第4355–4365页，2018a。Shiyu Liang、Ruoyu Sun、李一轩和 Rayadurgam Srikant。理解二元分类神经网络的损失曲面。载于 Jennifer Dy 和 Andreas Krause（编），第*35*届国际机器学习会议论文集，机器学习研究论文集第80卷，第2835–2843页。机器学习研究论文集，2018年7月10–15日。网址 https://proceedings. mlr.press/v80/liang18a.html。Shiyu Liang、Ruoyu Sun 和 R. Srikant。重新审视深度神经网络中的景观分析：消除趋于无穷的递减路径，2019年。Shiyu Liang、Ruoyu Sun 和 R. Srikant。在轻度过参数化神经网络中实现较小的测试误差。*CoRR*，abs/2104.11895，2021。网址 https://arxiv.org/abs/2104. 11895。Dachao Lin、Ruoyu Sun 和张志华。球对称数据下线性神经网络更快的方向收敛。神经信息处理系统进展，34，2021a。

Dachao Lin、Ruoyu Sun 和张志华。单隐层稀疏网络及其扩展的景观，2021b。

Junjie Liu、Zhe Xu、Runbin Shi、Ray CC Cheung 和 Hayden KH So。动态稀疏训练：利用可训练的掩码层从头开始寻找高效的稀疏网络。*arXiv* 预印本 *arXiv:2005.06870*，2020年。

Shiwei Liu、Tianlong Chen、Xiaohan Chen、Zahra Atashgahi、Lu Yin、Huanyu Kou、Li Shen、Mykola Pechenizkiy、Zhangyang Wang 和 Decebal Constantin Mocanu。通过增强剪枝可塑性与神经再生实现稀疏训练。神经信息处理系统进展，34，2021a。

Shiwei Liu、Decebal Constantin Mocanu、Yulong Pei 和 Mykola Pechenizkiy。自私稀疏 RNN 训练。发表于国际机器学习会议，第 6893–6904 页。机器学习研究论文集，2021b。

Shiwei Liu、Lu Yin、Decebal Constantin Mocanu 和 Mykola Pechenizkiy。我们真的需要稠密的过参数化吗？稀疏训练中的及时过参数化。载于国际机器学习会议，第 6989–7000 页。机器学习研究论文集，2021c。

Tianlin Liu 和 Friedemann Zenke。通过神经切线迁移寻找可训练的稀疏网络。载于国际机器学习会议，第 6336–6347 页。机器学习研究论文集，2020年。

Zhuang Liu、Mingjie Sun、Tinghui Zhou、Gao Huang 和 Trevor Darrell。重新思考网络剪枝的价值。*arXiv* 预印本 *arXiv:1810.05270*，2018年。

Christos Louizos、Max Welling 和 Diederik P Kingma。通过*l*_0 正则化学习稀疏神经网络。*arXiv* 预印本 *arXiv:1712.01312*，2017年。

Decebal Constantin Mocanu、Elena Mocanu、Peter Stone、Phuong H Nguyen、Madeleine Gibescu 和 Antonio Liotta。受网络科学启发，采用自适应稀疏连接实现人工神经网络的可扩展训练。《自然通讯》，9(1):1–12，2018年。

Pavlo Molchanov、Arun Mallya、Stephen Tyree、Iuri Frosio 和 Jan Kautz。神经网络剪枝的重要性估计。载于*IEEE/CVF*计算机视觉与模式识别会议论文集，第11264–11272页，2019年。

Hesham Mostafa 和 Xin Wang。通过动态稀疏重参数化实现深度卷积神经网络的参数高效训练。载于国际机器学习会议，第4646–4655页。机器学习研究论文集，2019年。

Michael C Mozer 和 Paul Smolensky。利用相关性自动缩减网络规模。连接科学，1(1):3–16，1989年。

Franco Pellegrini 和 Giulio Biroli。神经网络剪枝对特征进行去噪，并使视觉任务中的局部连接性显现。在国际机器学习会议，第 17601–17626 页。机器学习研究论文集，2022年。

Jeffrey Pennington、SamuelSchoenholz 和 SuryaGanguli。通过动态等距性在深度学习中复兴 sigmoid：理论与实践。神经信息处理系统进展，30，2017。

Jeffrey Pennington、SamuelSchoenholz 和 SuryaGanguli。深度网络中谱普适性的出现。载于 人工智能与统计国际会议，第 1924–1932 页。机器学习研究论文集，2018年。

亚历克斯·伦达、乔纳森·弗兰克尔和迈克尔·卡宾。比较神经网络剪枝中的回溯与微调。*arXiv* 预印本 *arXiv:2003.02389*，2020年。

Itay Safran 和 Ohad Shamir。两层 ReLU 神经网络中常见虚假的局部极小值。*arXiv* 预印本 *arXiv:1712.08968*，2017 年。Pedro Savarese、Hugo Silva 和 Michael Maire。通过连续稀疏化赢得彩票。神经信息处理系统进展，33：11380–11390，2020年。

Yao Shu、Shaofeng Cai、Zhongxiang Dai、Beng Chin Ooi 和 Bryan Kian Hsiang Low。Nasi：初始化时进行标签和数据无关的神经架构搜索。*arXiv* 预印本 *arXiv:2109.00817*，2021。Karen Simonyan 和 Andrew Zisserman。用于大规模图像识别的超深卷积网络。*arXiv* 预印本 *arXiv:1409.1556*，2014。Mahdi Soltanolkotabi、Adel Javanmard 和 Jason D Lee。过参数化浅层神经网络优化景观的理论见解。*IEEE*信息论汇刊，65(2):742–769，2019。Suraj Srinivas、Akshayvarun Subramanya 和 R Venkatesh Babu。训练稀疏神经网络。载于*IEEE*计算机视觉与模式识别会议研讨会论文集，第 138–145 页，2017。Lili Su 和 Pengkun Yang。关于过参数化神经网络的学习：一种函数逼近视角。神经信息处理系统进展，32，2019。Ruo-Yu Sun。深度学习优化：综述。中国运筹学会学报，第 1–46 页，2020。Ruoyu Sun、Tiantian Fang 和 Alexander Schwing。迈向更优的生成对抗网络全局损失景观。载于 H. Larochelle、M. Ranzato、R. Hadsell、M. F. Balcan 和 H. Lin（编），神经信息处理系统进展，第 33 卷，第 10186–10198 页。Curran Associates, Inc.，2020a。网址 https://proceedings.neurips.cc/paper/2020/file/738a6457be8432bab553e21b4235dd97-Paper.pdf。Ruoyu Sun、Dawei Li、Shiyu Liang、Tian Ding 和 Rayadurgam Srikant。神经网络的全局景观：综述。*IEEE*信号处理杂志，37(5):95–108，2020b。Hidenori Tanaka、Daniel Kunin、Daniel L Yamins 和 Surya Ganguli。通过迭代保留突触流，在无数据条件下剪枝神经网络。神经信息处理系统进展，33:6377–6389，2020。Luca Venturi、Afonso Bandeira 和 Joan Bruna。双层神经网络优化景观中的伪谷。*arXiv* 预印本 *arXiv:1802.06384*，2018。Sagar Verma 和 Jean-Christophe Pesquet。通过次微分包含实现网络稀疏化。载于国际机器学习会议，第 10542–10552 页。机器学习研究论文集，2021。Chaoqi Wang、Guodong Zhang 和 Roger Grosse。通过保留梯度流，在训练前挑选中奖彩票。*arXiv* 预印本 *arXiv:2002.07376*，2020。Haoxiang Wang、Yite Wang、Ruoyu Sun 和 Bo Li。模型无关元学习的全局收敛与受理论启发的小样本学习神经架构搜索。载于计算机视觉与模式识别会议，2022。Wei Wen、Chunpeng Wu、Yandan Wang、Yiran Chen 和 Hai Li。在深度神经网络中学习结构化稀疏性。神经信息处理系统进展，29，2016。Lechao Xiao、Jeffrey Pennington 和 Samuel Schoenholz。解耦深度神经网络的可训练性与泛化能力。载于国际机器学习会议，第 10462–10472 页。机器学习研究论文集，2020。Xia Xiao、Zigeng Wang 和 Sanguthevar Rajasekaran。Autoprune：通过正则化辅助参数实现自动网络剪枝。神经信息处理系统进展，32，2019。Haoran You、Chaojian Li、Pengfei Xu、Yonggan Fu、Yue Wang、Xiaohan Chen、Richard G Baraniuk、Zhangyang Wang 和 Yingyan Lin。绘制早鸟中奖彩票：迈向更高效的深度网络训练。*arXiv* 预印本 *arXiv:1909.11957*，2019。Ruichi Yu、Ang Li、Chun-Fu Chen、Jui-Hsin Lai、Vlad I Morariu、Xintong Han、Mingfei Gao、Ching-Yung Lin 和 Larry S Davis。Nisp：使用神经元重要性评分传播剪枝网络。载于*IEEE*计算机视觉与模式识别会议论文集，第 9194–9203 页，2018。

Jiawei Zhang、Yushun Zhang、Mingyi Hong、Ruoyu Sun 和 Zhi-Quan Luo。表达能力与可训练性相遇：少于 n 个神经元也能发挥作用。*神经信息处理系统进展*，34，2021。

Xiao Zhou、Weizhong Zhang、Hang Xu 和 Tong Zhang。《具有全局稀疏度约束的神经网络有效稀疏化》。载于 *IEEE/CVF计算机视觉与模式识别会议论文集*，第 3599–3608 页，2021。

附录概述

附录的组织结构如下：

- 附录 A 介绍了一般实验设置。
- 附录 B 提供了关于我们多重采样公式的直观解释。
- 附录 C 展示了迭代步数的消融研究。
- 附录 D 详细比较了不同前瞻性剪枝方法的计算成本。
- 附录 E 进一步比较了前瞻性剪枝方法与早期剪枝基线。
- 附录 F 对 SNIP变体进行了比较。
- 附录 G 展示了剪枝后的神经网络的特征谱。
- 附录 H 提供了主要实验的完整图表。
- 附录 I 对扰动超参数进行了消融研究 ϵ。
- 附录 J 概述了本文所研究的前瞻性剪枝方法。
- 附录 K 对比了神经切线迁移与 NTK-SAP。
- 附录 L 提供了 ProsPr 与 NTK-SAP 之间的比较。
- 附录 M 讨论了 NTK-SAP 的权重无关性质。
- 附录 N 讨论了为什么 NTK-SAP 有可能避免层坍塌。
- 附录 O 论证了第 5.2 节所采用的有限差分表述的合理性。

A 实验设置

我们的代码基于 Synflow 的原始代码。[7] 原始代码不包含许可证信息。但我们已通过电子邮件获得批准。作者授予学术用途的免费使用权。

A.1 架构详情

这些网络架构如下：

- 对于 CIFAR-20 上的ResNet-20 模型以及CIFAR-100 上的 VGG-16（带 BatchNorm）模型，我们使用 Synflow（Tanaka 等人，2020年）实验中采用的默认设置。彩票票据实验（Frankle 与 Carbin，2018年；Renda等人，2020年；Frankle 等人，2020a）使用相同的架构。
- 对于 ResNet-18 和 ResNet-50，我们使用 ImageNet 版本的 ResNet 模型。[8] 我们使用 torchvision 实现。

A.2 数据集

我们使用 CIFAR-10、CIFAR-100、Tiny-ImageNet 和 ImageNet。这些数据集不包含个人身份信息或冒犯性内容。

- CIFAR-10 和CIFAR-100数据集通过逐通道归一化进行增强，应用随机裁剪（32×32，填充 4）和水平翻转。
- Tiny-ImageNet通过按通道归一化进行增强：选择一个宽高比在0.8到1.25之间的随机图像块，以及一个在0.1到1.0之间的随机缩放比例，将其裁剪为 64 ×64，并进行随机水平翻转。
- ImageNet 数据集经过按通道归一化处理，并应用随机尺寸裁剪至 224×224以及随机水平翻转。

[7]https://github.com/ganguli-lab/Synaptic-Flow.[8]
https://github.com/weiaicunzai/pytorch-cifar100

A.3 训练详情

训练超参数见表 2。这些超参数是在 Synflow（Tanaka 等人，2020年）和 Frankle 等人（2020b）所用参数的基础上修改而来。在下表中，我们用 Nesterov 优化器指代采用 Nesterov 动量的随机梯度下降优化器。

表 2：本工作中使用的训练超参数。

网络	数据集	训练轮数	批次	优化器	动量	LR	学习率衰减	权重衰减	初始化
ResNet-20	CIFAR-10	160	128	SGD	0.9	0.1	在第 80 和 120 训练轮次时为 10 倍	1e-4	Kaiming 正态
VGG-16	CIFAR-100	160	128	Nesterov	0.9	0.1	在第 60、120 个训练轮次时调整为 10 倍	5e-4	Kaiming 正态分布
ResNet-18	Tiny-ImageNet	200	256	SGD	0.9	0.2	在第 100、150 个训练轮次时调整为 10 倍	1e-4	Kaiming 正态初始化
ResNet-18	ImageNet	90	512	SGD	0.9	0.1	在训练轮次 30、60、80 时将学习率乘以 10	1e-4	Kaiming 正态初始化
ResNet-50	ImageNet	90	448	SGD	0.9	0.1	在训练轮次 30、60、80 时进行 10 倍调整	1e-4	Kaiming 正态初始化

A.4 剪枝详情

稀疏率 图 1 的 x 轴采用密度（1−稀疏度）的对数刻度。更具体地说，我们通过两次移除剩余参数的 20% 来获得每个稀疏率。例如，第一个稀疏率为 1−0.8×0.8=0.36。我们选择这些稀疏率来评估，主要遵循 Frankle 等人（2020b）采用的传统做法。其直觉是，在最初几轮中，网络参数过多，因此可以移除很大一部分权重。

选择剪枝样本。 与 GraSP 的实现不同（该实现每个类别恰好使用 10 个样本），我们遵循 SNIP、迭代 SNIP 和 Synflow 的实现，加载多个批次，使每个类别大致包含十个样本。

剪枝批次大小。 为了计算显著性分数，我们在 ImageNet实验中使用 128 的剪枝批次大小，并在其他情况下使用 256。

BatchNorm 层。 正如 Synflow 原论文所指出的，BatchNorm 层对于正确应用不同的剪枝方法至关重要。SNIP、迭代 SNIP 和 GraSP在训练模式下对网络进行剪枝，而 Synflow 使用评估模式。至于NTK-SAP，我们在修改后的评估模式下计算显著性分数。更具体地说，我们首先在不构建计算图的情况下计算批次统计数据，然后使用相同的批次统计数据来近似评估模式下的神经切线核核范数。

迭代方法的剪枝计划。 按照 Synflow，所有迭代方法，包括Synflow、迭代SNIP 和 NTK-SAP，均使用指数剪枝计划。有关如何计算剪枝计划，请参见算法 1。

有关 NTK-SAP 的更多详细信息。对于每轮剪枝，我们使用 40 个批次的噪声输入（而迭代 SNIP 使用 79 个批次），批次大小为 128，用于 ImageNet 数据集上的 ResNet-50。对于其他设置，我们使用的用于剪枝的高斯输入数量是类别数量的十倍。

至于剪枝轮数 T，所有 ImageNet 实验均使用 25 轮。对于 CIFAR-10 上的 ResNet-20，我们使用 $T = 20$ 轮，每轮包含 5 个批次（因为批次大小大于类别数 10）。对于 CIFAR-100 和 Tiny-ImageNet 数据集，我们使用 $T = 100$轮。
对于 ϵ，ImageNet 上的 ResNet-18 使用 $\epsilon = 10^{-6}$，其他情况则使用 $\epsilon = 10^{-4}$。

A.5 硬件

我们的所有实验均在 NVIDIA V100 上运行。CIFAR-10/100 和 Tiny-ImageNet 数据集上的实验每次使用一块图形处理器运行。ImageNet 实验中，ResNet-18 和 ResNet-50 分别使用 2 块和 4 块图形处理器。

B 多重采样公式的直觉

B.1 通过解析神经切线核理解多重采样公式

本节将详细阐述我们在 NTK-SAP 中进行多重采样的直觉依据。

我们仍从公式 7 开始讨论。需要指出的是，公式 7 本身不足以保证良好的性能，因为直接将神经切线核理论应用于剪枝时，实际上存在一个差距：在实际应用中，固定权重神经切线核可能与解析神经切线核大相径庭。

解析神经切线核的标准表达式。需要指出的是，直接使用固定权重神经切线核来近似解析神经切线核，可以写成以下数学形式：

$$\boldsymbol{\Theta}(\mathcal{X},\mathcal{X}) = \lim_{n_l\to\infty,\forall l} \hat{\boldsymbol{\Theta}}^{\{n_l\}}(\mathcal{X},\mathcal{X};\boldsymbol{\theta}) \tag{11}$$

其中， $\boldsymbol{\Theta}$ 是解析神经切线核，而 $\hat{\boldsymbol{\Theta}}^{\{n_l\}}(\cdot,\cdot;\boldsymbol{\theta})$ 是宽度为 n_l、权重为 $\boldsymbol{\theta}$ 的神经网络的固定权重神经切线核。近似解析神经切线核的一种自然方法是去掉"lim"运算，并对给定的足够大宽度 $\{n_l\}$ 直接使用 $\hat{\boldsymbol{\Theta}}^{\{n_l\}}(\mathcal{X},\mathcal{X};\ \boldsymbol{\theta})$。

解析神经切线核与固定权重神经切线核的两个区别。我们指出，除了宽度的选择之外，权重 $\boldsymbol{\theta}$ 的选择也会影响 $\hat{\boldsymbol{\Theta}}^{\{n_l\}}(\mathcal{X},\mathcal{X};\ \boldsymbol{\theta})$。换言之， $\boldsymbol{\Theta}(\mathcal{X},\mathcal{X})$既与宽度无关，也与权重无关，而 $\hat{\boldsymbol{\Theta}}^{\{n_l\}}(\mathcal{X},\mathcal{X};\ \boldsymbol{\theta})$则依赖于宽度和权重。解析神经切线核与固定权重神经切线核之间存在两个区别，但表达式 11 仅体现了宽度方面的区别，而忽略了权重方面的区别。其根本原因在于，在无限宽神经网络中，随机权重的影响会相互抵消，因此无需通过添加期望之类的算子来消除随机性。

解析神经切线核的另一种表达式。虽然不是必需的，但在表达式中添加"期望"算子是合理的。我们将解析神经切线核重写为固定权重神经切线核在宽度趋于无穷时的极限（对随机权重取期望），数学表达如下：

$$\boldsymbol{\Theta}(\mathcal{X},\mathcal{X}) = \mathbb{E}_{\boldsymbol{\theta}}\left[\lim_{n_l\to\infty,\forall l} \hat{\boldsymbol{\Theta}}^{\{n_l\}}(\mathcal{X},\mathcal{X};\boldsymbol{\theta})\right]. \tag{12}$$

此公式是正确的，因为 $\mathbb{E}_{\boldsymbol{\theta}}\left[\lim_{n_l\to\infty,\forall l} \hat{\boldsymbol{\Theta}}^{\{n_l\}}(\mathcal{X},\mathcal{X};\ \boldsymbol{\theta})\right] = \mathbb{E}_{\boldsymbol{\theta}}\left[\boldsymbol{\Theta}(\mathcal{X},\mathcal{X})\right] = \boldsymbol{\Theta}(\mathcal{X},\mathcal{X})$。

解析神经切线核的另一种近似。 尽管公式 12中的期望算子是多余的，但当我们考虑有限宽度近似时，它会产生影响。由公式12得出的一个自然近似如下：首先，移除"lim"操作，并使用一组足够大的固定宽度 $\{n_l\}$；其次，对多个相互独立的权重配置进行采样，并将平均值用作"期望" $\mathbb{E}_{\boldsymbol{\theta}}\left[\hat{\boldsymbol{\Theta}}^{\{n_l\}}(\mathcal{X},\mathcal{X};\ \boldsymbol{\theta})\right]$ 的近似，其中给定一组足够大的宽度 $\{n_l\}$。具体而言，使用 R 个相互独立的初始化点，我们得到显著性准则公式 8 的稳定版本。

B.2 通过固定权重神经切线核演化理解多重采样公式

在上一节中，我们分析了为什么多重采样比单次抽取固定权重神经切线核更能准确估计解析神经切线核。这一分析促成了多重采样方法的设计。然而，为什么要用解析神经切线核作为有限宽度神经网络的性能指标，似乎并不明了。本节将从另一个角度解释多重采样方法。

当宽度趋于无穷时，神经切线核几乎是确定的，并且在训练过程中渐近地保持不变。

在这种情况下，解析神经切线核是衡量足够大的网络优化情况的良好指标，因为它衡量的是随机权重配置下的预期行为以及

训练轨迹。设 $\boldsymbol{\theta}_t(\boldsymbol{\theta}_0; t)$ 表示网络在训练步 t 时、初始随机权重配置为 $\boldsymbol{\theta}_0$的参数，则

$$\boldsymbol{\Theta}(\mathcal{X}, \mathcal{X}) = \mathbb{E}_{\boldsymbol{\theta}_0, t}\left[\lim_{n_l \to \infty, \forall l} \hat{\boldsymbol{\Theta}}^{\{n_l\}}(\mathcal{X}, \mathcal{X}; \boldsymbol{\theta}_t(\boldsymbol{\theta}_0; t))\right]. \tag{13}$$

在有限宽度（现实世界）设定下，固定权重神经切线核不再保持不变，而是时间的函数。尽管我们无法再用单个确定且恒定的核描述整个训练过程，但仍可以用神经正切核序列描述该过程。

为简单起见，我们考虑离散的梯度下降更新规则。我们可以用序列 $\hat{}\ \hat{}\ \hat{}\{\boldsymbol{\Theta}_0(\mathcal{X}, \mathcal{X};\ \boldsymbol{\theta}_0)\ \boldsymbol{\Theta}_1(\mathcal{X}, \mathcal{X};\ \boldsymbol{\theta}_1)\ \cdots, \boldsymbol{\Theta}_E(\mathcal{X}, \mathcal{X};\ \boldsymbol{\theta}_E)\}$ 来刻画有限宽度神经网络的训练动态，该序列包含每个时间步的所有固定权重神经切线核。

在这种情况下，我们希望序列中的所有元素都具有良态性质，从而使神经网络在训练阶段表现良好。因此，我们需要量化移除一个掩码对这些固定权重神经切线核的核范数的影响。

一种可能的度量指标是$\frac{1}{E+1}\sum_{t=0}^{E} \|\hat{}\boldsymbol{\Theta}_t(\mathcal{X}, \mathcal{X};\ \boldsymbol{\theta}_t)\|_*$，它是基于参数序列 $\{\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_E\}$ 计算得到的固定权重神经切线核的核范数平均值。

我们推测，参数空间在训练过程中得到了充分探索。参数序列中的元素 $\{\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_E\}$ 可能分布得较为均匀，因此 $\frac{1}{E+1}\sum_{t=0}^{E} \|\hat{}\boldsymbol{\Theta}_t(\mathcal{X}, \mathcal{X};\ \boldsymbol{\theta}_t)\|_*$ 可以被视为 $\mathbb{E}_{\boldsymbol{\theta}}\left[\|\hat{}\boldsymbol{\Theta}(\mathcal{X}, \mathcal{X};\ \boldsymbol{\theta})\|_*\right]$. 的近似。作为近似这一期望的另一种方式，我们可以使用多个随机权重配置来近似 $\mathbb{E}_{\boldsymbol{\theta}}\left[\|\hat{}\boldsymbol{\Theta}(\mathcal{X}, \mathcal{X};\ \boldsymbol{\theta})\|_*\right]$，即$\sum_{r=1}^{R} \|\hat{}\boldsymbol{\Theta}_{0,r}(\mathcal{X}, \mathcal{X}; \boldsymbol{\theta}_{0,r})\|_*$，这就是显著性准则公式 8。

相比之下，许多现有工作在初始化时使用单个随机权重配置计算固定权重神经切线核，因此无法很好地衡量实际规模网络的优化情况。在我们的工作中，我们计算期望固定权重NTK，并认为这一期望版本的固定权重NTK能够更好地反映子网络在随机权重配置和训练轨迹下的预期行为。因此，我们旨在找到一个具有良态期望固定权重NTK的子网络。

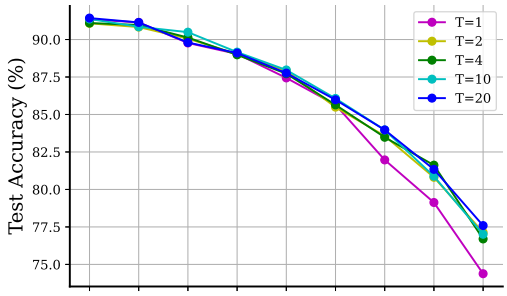
为了使期望固定权重NTK呈良态，我们选择在初始化时将其与稠密网络的固定权重NTK对齐。这样选择有以下两个理由：

- **稠密固定权重神经切线核的特征谱是一个很好的候选**。我们认为，稠密网络的良好性能意味着稠密网络在初始化时的特征谱具有良态（有关稠密网络的特征谱，请参见第 6.5 节和附录 G.1）。
- **稠密固定权重神经切线核的特征谱是一个实用的选择**。对于前瞻剪枝，我们只能在初始化时访问固定权重神经切线核。此外，第3节中提到的前瞻剪枝框架是一个通用框架，用于保留稠密网络的一个标量指标。因此，由于该框架的内在特性，我们只能尝试在初始化时使稀疏网络的特征谱与稠密网络的特征谱对齐，而不能对齐到某个给定的良态特征谱。

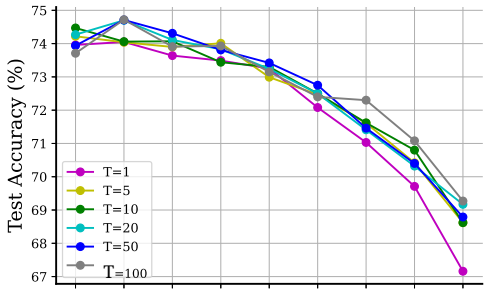
事实上，对于实际网络，稠密网络的特征谱可能会在训练过程中发生变化。因此，我们的方法只能在训练早期近似地使训练行为与稠密网络对齐。在后续训练轮次中，子网络预期具备的良态特性确保了所确定的子网络相比其他前瞻剪枝方法具有更好的优化性能。

C 关于迭代步数的消融研究

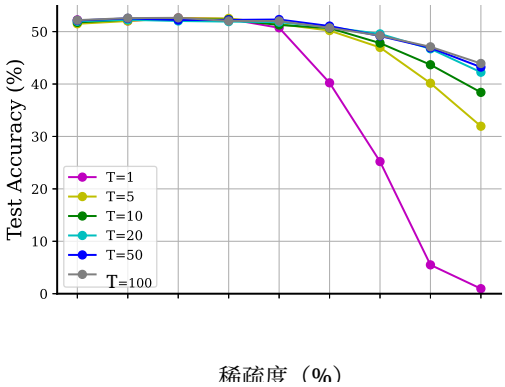
我们在此展示第 6.4 节的实验结果。图 3 表明，对于较小的数据集，NTK-SAP 可以采用较少的迭代次数 T，且性能不会显著下降。



(a) CIFAR-10 数据集上的 ResNet-20。



(b) CIFAR-100 数据集上的 VGG-16。



(c) ResNet-18 在 Tiny-ImageNet 数据集上。

图3：评估迭代次数（T）对 NTK-SAP在 CIFAR-10（ResNet-20）、CIFAR-100（VGG-16）和 Tiny-ImageNet（ResNet-18）上的性能的影响。

D 详细计算成本

D.1 计算时间详细比较

表 3：所考虑方法的剪枝时间与训练 1 个训练轮次所需时间的比较。

数据集	CIFAR-10	CIFAR-100	Tiny-ImageNet	ImageNet	
模型	ResNet-20	VGG-16	ResNet-18	ResNet-18	ResNet-50
1 个训练轮次的训练时间	9.6 秒	11 s	24 s	15 分 42 秒	18 分钟
SNIP	1 s	1 s	1 s	13 s	34 s
GraSP	1 s	1 s	3 s	50 s	5 分 32 秒
Synflow	12 s	34 s	1分2秒	6分34秒	5 分 39 秒
迭代 SNIP	24 s	46 s	1 分 18 秒	16分42秒	23 分 17 秒
NTK-SAP	17 s	1 分 14 秒	2 分 14 秒	17 分 53 秒	22 分 49 秒
NTK-SAP-S	3 s	14 s	31 s	-	-

我们在表 3 中详细比较了所考虑方法的剪枝时间，以及模型训练 1 个训练轮次所需的时间。我们关注将 NTK-SAP 的剪枝时间与 1 个训练轮次的时间进行比较，因为 de Jorge 等（2020b）的实证结果表明，在经过 1 轮训练后，大多数前瞻性剪枝方法的表现不逊于幅值剪枝。我们也在附录 E 中重复了这类实验。

对于 ImageNet 实验，NTK-SAP 的剪枝时间与模型训练 1–2 轮的时间相近。对于较小的数据集，附录 C 表明，我们可以减少剪枝轮数 T。因此，对于较小的数据集，我们改为将其余方法与采用较小 T 的 NTK-SAP 进行比较。我们将这一修改版本命名为 NTK-SAP-S。具体而言，CIFAR-10 实验中我们采用 $T = 4$，CIFAR-100 和 Tiny-ImageNet 实验中则采用 $T = 20$。可以看到，NTK-SAP-S 能够在相当于训练 1–2 轮的时间内完成剪枝。因此，我们认为 NTK-SAP 的计算成本是合理的。

D.2 浮点运算次数详细比较

在本节中，我们从浮点运算次数的角度比较不同的前瞻性剪枝方法。我们使用 Synflow 代码库中的默认浮点运算次数计数器。我们将反向传播的浮点运算次数近似为推理/前向传播浮点运算次数的两倍。剪枝浮点运算次数的计算进行了简化，假设神经网络在计算梯度时是稠密的。

剪枝成本。 表4 展示了不同前瞻剪枝方法的剪枝成本对比。Synflow 的剪枝成本最低，因为其显著性分数是使用全为 1的输入计算的。NTK-SAP 的剪枝成本约为迭代SNIP 的两倍，因此与其他前瞻性剪枝方法相比，该方法需要更高的成本。不过，我们想强调的是，如第 6.4 节和第 C节所示，通过使用更小的 T，可以降低 NTK-SAP 的剪枝成本。

Inference 成本。 我们进一步比较通过不同前瞻方法找到的子网络的推理成本。 t pru剪枝方法。结果见表 5、6 和 7。

总剪枝和训练成本。最后，我们在表 8、9 和 10 中比较了所考虑的前瞻剪枝方法的剪枝与训练总浮点运算次数。如第 6.4 节所述，NTK-SAP 通常需要更高的总计算成本。不过，我们认为额外的计算成本是合理的，尤其是考虑到 NTK-SAP 是一种与数据无关的方法，并且找到掩码后即可轻松应用于不同数据集。此外，我们还想再次强调，通过减少 T，可以降低 NTK-SAP 的计算成本。

E 早期剪枝基线

我们重复了 FORCE（de Jorge 等，2020b）的实验，将前述前瞻剪枝方法与早期剪枝进行比较。具体而言，我们先对稠密神经网络训练 1 个训练轮次，然后使用幅值剪枝对训练后的神经网络进行剪枝。接着，按照 LTH 工作中的做法，将权重恢复到初始值。我们将这一基线称为 Mag-1epoch。图 4 展示了 1 个训练轮次的训练略微提升了幅值剪枝的性能。不过，NTK-SAP 的表现仍优于这种早期剪枝方法。

F SNIP变体

在本节中，我们将研究 SNIP 的变体，包括 SNIP、迭代 SNIP 和 FORCE。 .

不同稀疏度调度的影响。 我们首先研究迭代 SNIP 中所用稀疏度调度的影响。具体而言，我们比较了主论文中使用的迭代 SNIP（指数）与采用线性稀疏度调度的迭代 SNIP（线性）。如图 5 所示，我们发现，线性调度略微提升了迭代 SNIP 在较小数据集上的性能。然而，表 11 显示，线性调度并不一定能提升迭代 SNIP 在较大数据集上的性能。

本文实现的迭代 SNIP 与 de Jorge 等，2020b 中报告的结果之间的性能差距，可能是由所采用的不同初始化方法以及用于剪枝的批次数量造成的。此外，de Jorge 等人（2020b）仅考察了相对较大的神经网络，包括 VGG-19 和 ResNet-50。不过，Frankle 等人也通过实验证明，

Table 4：剪枝浮点运算次数对比。浮点运算次数最大的结果以蓝色显示，结果与 th e sma最少的浮点运算次数以**粗体**标示。

数据集	CIFAR-10	CIFAR-100	Tiny-ImageNet
模型	ResNet-20	VGG-16	ResNet-18
SNIP	2.48×10^{11}	1.88×10^{12}	8.94×10^{11}
GraSP	4.95×10^{11}	3.77×10^{12}	1.79×10^{12}
Synflow	$\mathbf{1.24 \times 10^{10}}$	$\mathbf{9.42 \times 10^{10}}$	$\mathbf{4.47 \times 10^{10}}$
迭代 SNIP	2.48×10^{13}	1.88×10^{14}	8.94×10^{13}
NTK-SAP	4.95×10^{13}	3.77×10^{14}	1.79×10^{14}
NTK-SAP（小 T）	9.91×10^{12}	7.54×10^{13}	3.58×10^{13}

表5：推理浮点运算次数（$\times 10^6$）在 CIFAR-10（ResNet-20）上的比较。浮点运算次数最大的结果以蓝色显示，浮点运算次数最小的结果以**粗体**显示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
SNIP	31.90	19.40	12.30	7.37	3.77
GraSP	**24.20**	**13.50**	8.65	4.95	3.20
Synflow	33.20	21.60	12.20	5.73	2.67
迭代 SNIP	31.30	16.70	**7.80**	**3.46**	**2.10**
NTK-SAP	31.30	18.20	10.10	5.30	2.83

表 6：CIFAR-100（VGG-16）上的推理浮点运算次数（$\times 10^7$）比较。浮点运算次数最大的结果以蓝色标示，浮点运算次数最小的结果以**粗体**标示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
SNIP	25.50	17.10	11.80	8.19	5.34
GraSP	**18.30**	**10.30**	**6.15**	**3.75**	**2.48**
Synflow	25.80	17.60	11.50	6.95	3.71
迭代 SNIP	25.20	15.80	9.67	5.51	2.97
NTK-SAP	26.10	17.50	11.60	7.54	4.58

表 7：推理浮点运算次数（$\times 10^7$）在Tiny-ImageNet（ResNet-18）上的比较。浮点运算次数最大的结果以 蓝色显示，浮点运算次数最小的结果以**粗体**显示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
SNIP	12.30	8.62	6.13	4.14	2.51
GraSP	**8.14**	**5.29**	**3.59**	2.45	1.65
Synflow	12.80	9.49	6.77	4.31	2.36
迭代 SNIP	12.20	7.91	4.40	**2.00**	**0.80**
NTK-SAP	12.70	9.47	6.77	4.53	2.73

表 8：总剪枝 +训练浮点运算次数（$\times 10^{14}$）在 CIFAR-10（ResNet-20）上的比较。浮点运算次数最大的结果以蓝色显示，浮点运算次数最小的结果以**粗体**显示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
SNIP	7.65	4.67	2.95	1.77	0.91
GraSP	**5.81**	**3.24**	**2.08**	1.19	0.77
Synflow	7.98	5.19	2.93	1.38	**0.64**
迭代 SNIP	7.75	4.25	2.12	**1.08**	0.75
NTK-SAP	8.01	4.86	2.91	1.77	1.17

表9：总剪枝 + 训练浮点运算次数（$\times 10^{15}$）在 CIFAR-100（VGG-16）上的比较。浮点运算次数最大的方法以蓝色 显示，浮点运算次数最小的方法以**粗体**显示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
SNIP	6.13	4.09	2.83	1.97	1.28
GraSP	**4.39**	**2.47**	**1.48**	**0.90**	**0.60**
Synflow	6.19	4.23	2.76	1.67	0.89
迭代 SNIP	6.24	3.99	2.51	1.51	0.90
NTK-SAP	6.65	4.58	3.17	2.19	1.48

表10：剪枝总计 + 训练浮点运算次数（$\times 10^{14}$）在Tiny-ImageNet(ResNet-18) 上的比较。浮点运算次数最大的结果以 蓝色 显示，浮点运算次数最小的结果以**粗体**显示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
SNIP	73.70	51.70	36.80	24.80	15.10
GraSP	**48.80**	**31.80**	**21.50**	14.70	9.91
Synflow	76.70	57.00	40.60	25.90	14.10
迭代 SNIP	73.90	48.40	27.30	**12.90**	**5.71**
NTK-SAP	78.20	58.60	42.40	29.00	18.10

表 11：SNIP变体在 ImageNet 数据集上的比较。最佳结果以粗体显示。

网络	**ResNet-18**		**ResNet-50**	
稀疏率	89.26%	95.60%	89.26%	95.60%
（稠密基线）	69.98		76.20	
SNIP Lee 等（2018）	**57.41**	**45.04**	**60.98**	**40.69**
迭代 SNIP（指数）de Jorge 等（2020b）	52.97	37.44	52.53	36.82
迭代 SNIP（线性）	50.29	40.98	56.54	30.55

（2020b）和 Tanaka 等（2020）[9] 指出，迭代 SNIP 并不一定能提升 SNIP 的性能。

FORCE 的性能不稳定。此外，我们还考虑了 FORCE（de Jorge 等，2020b），该方法允许剪枝掩码在迭代剪枝过程中恢复。结果也展示在图 5 中。我们观察到，FORCE 在 VGG-16（CIFAR-100）上的性能显著下降。因此，我们未在正文中报告 FORCE。

[9]请参阅 Synflow 论文的作者反馈文件。

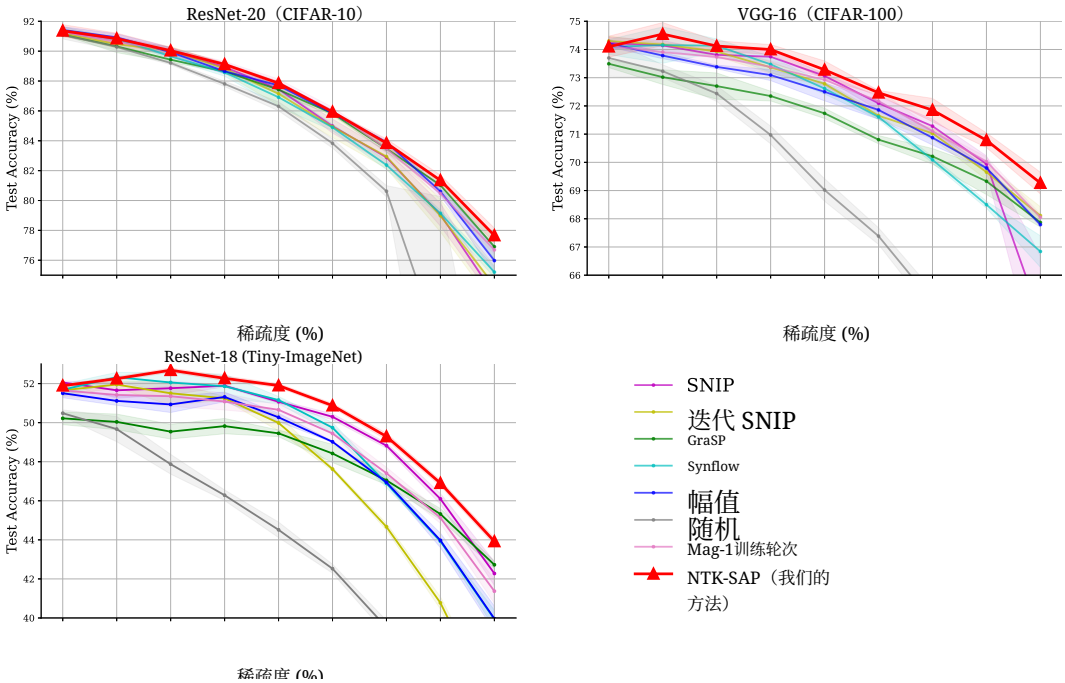


图 4：训练 1 个训练轮次后，前瞻性剪枝方法与幅值剪枝的比较。结果为 3 个随机种子的平均值，阴影区域表示各次运行的标准差。

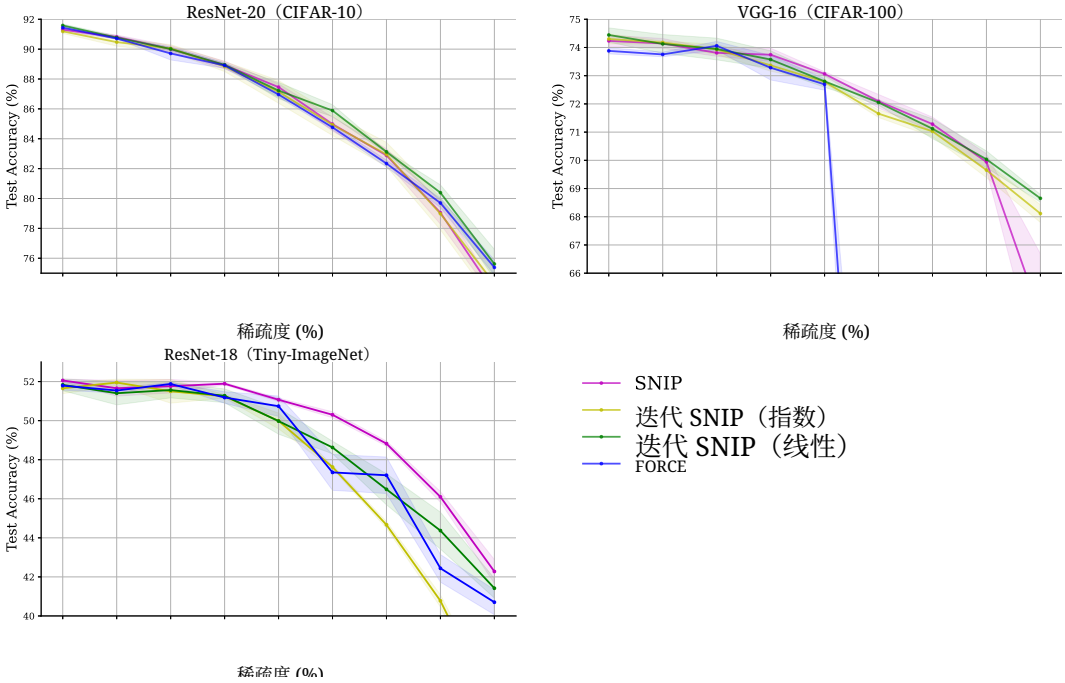
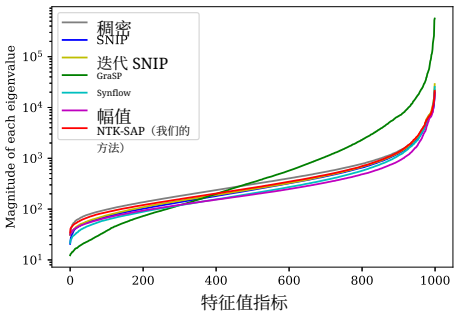
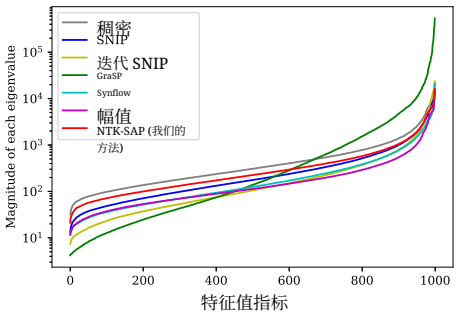


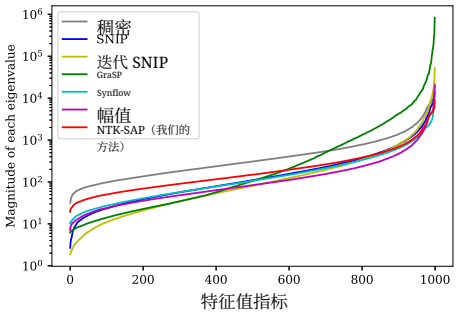
图 5：SNIP变体的比较。结果取 3 个随机种子的平均值，阴影区域表示各次运行的标准差。



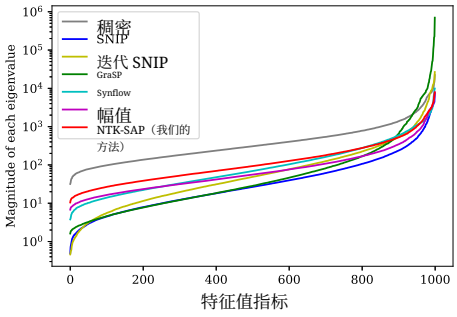
(a) 稀疏度 = 36.00%



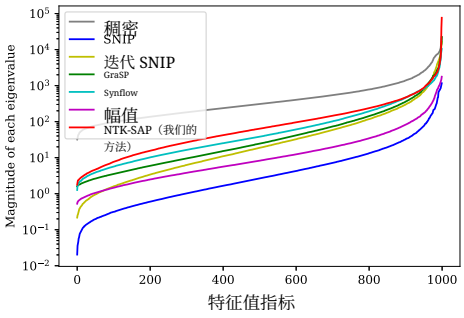
(b) 稀疏度 = 73.80%



(c) 稀疏度 = 89.30%



(d) 稀疏度 = 95.60%



(e) 稀疏度 = 98.20%

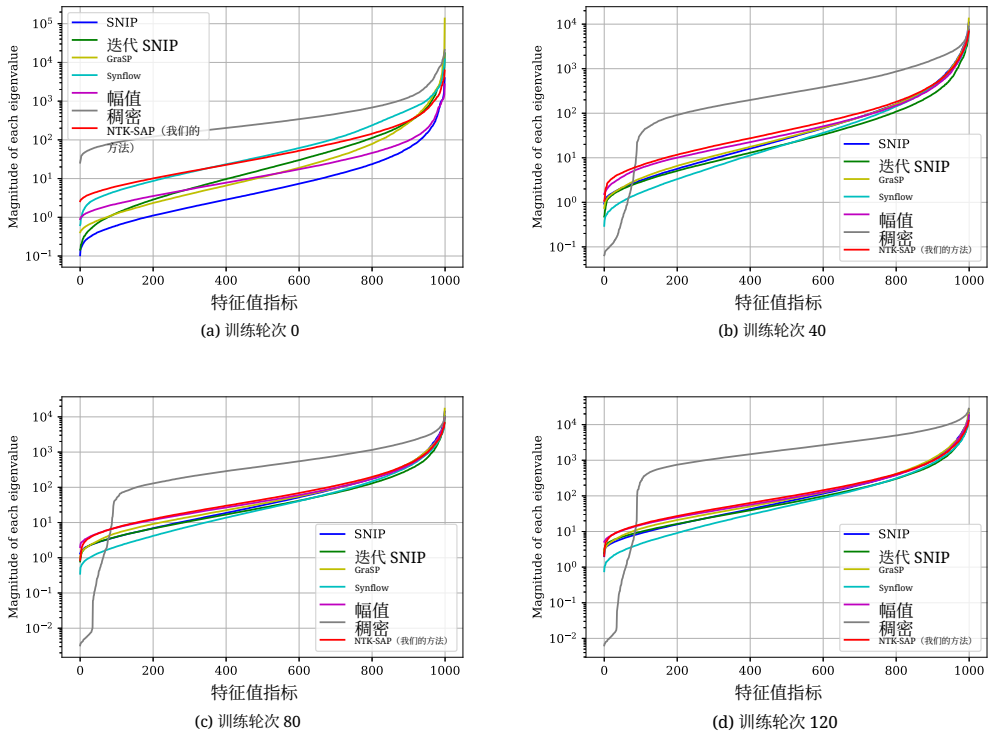
图6：不同稀疏率下ResNet-20（CIFAR-10）上固定权重神经切线核的特征值分布。我们将每个特征值的索引置于 x 轴上，并在 y 轴上呈现相应特征值的幅值。

G 剪枝神经网络的特征谱

G.1 初始化时的特征谱

为了直观展示我们提出的 NTK-SAP 方法的有效性，我们在 CIFAR-10 数据集上使用 ResNet-20 进行实验，稀疏率为 $\{36.00\%\ 73.80\%\ 89.30\%\ 95.60\%\ 98.20\%\}$.[10] 我们使用批次大小为 100 的批次计算固定权重神经切线核。因此，固定权重神经切线核的大小为1000×1000。

[10]其他数据集的 k 值要大得多，因此固定权重神经切线核更难计算。



(a) 训练轮次 0　(b) 训练轮次 40

(c) 训练轮次 80　(d) 训练轮次 120

图 7：稀疏率为 98.20% 时，不同训练轮次下 ResNet-20（CIFAR-10）的固定权重神经切线核的特征值分布。横轴为各特征值的索引，纵轴为对应特征值的幅值。

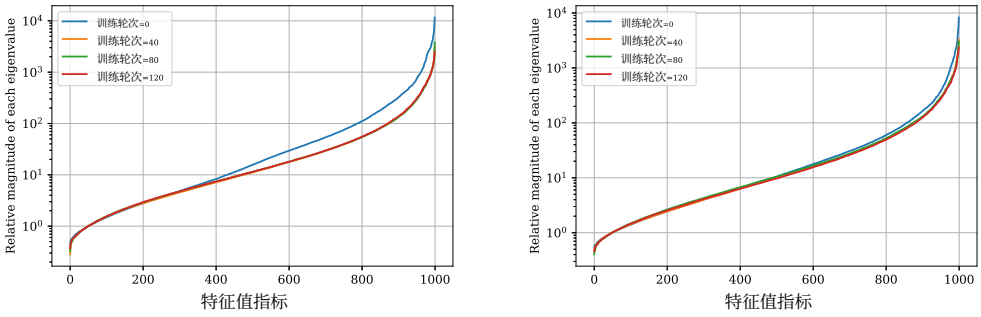
图 6 展示了采用上述前瞻性剪枝方法剪枝后的神经网络的特征值分布。这表明 NTK-SAP 能有效地使谱保持接近稠密网络。

如图 7 所示，稠密网络的谱在整个训练过程中不断变化。相比之下，所有剪枝网络的固定权重神经切线核特征谱表现相似，且偏离了稠密网络。不过，我们表明，与其他前瞻剪枝方法相比，NTK-SAP 的特征谱在最初几个训练轮次中与稠密网络的特征谱对齐（如图所示，在大部分训练过程中，NTK-SAP 的大多数特征值都最接近稠密网络）。其次，NTK-SAP 的固定权重神经切线核在整个训练过程中都保持良态，这与我们的动机一致。

G.2 训练期间的特征谱

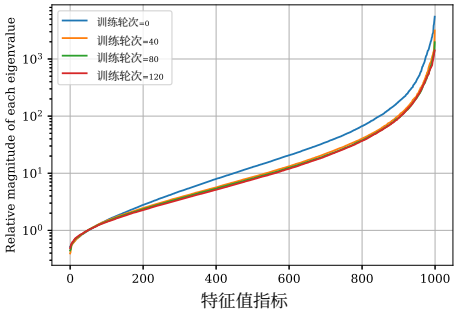
我们还绘制了在训练期间采用这些前瞻剪枝方法进行剪枝的神经网络的特征值分布。由于在 CIFAR-10 数据集上，对于较大范围的稀疏率，不同方法的测试性能差异并不大，因此我们只关注最极端的稀疏度，即 98.20%。我们通过将每个特征值除以最小的 10% 特征值的平均值，报告相对特征值。结果如图 8 所示。

我们观察到，在 CIFAR-10 数据集上表现最好的三种方法，即 NTK-SAP、GraSP 和幅值方法，具有相对较低的条件数。此外，表现最好的两种方法，即 GraSP 和 NTK-SAP，在训练期间的相对特征值分布几乎保持稳定不变。这可能解释了为什么 GraSP 和 NTK-SAP 在该稀疏率下表现良好。

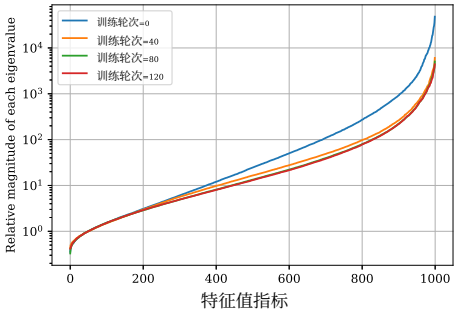


(a) 迭代 SNIP

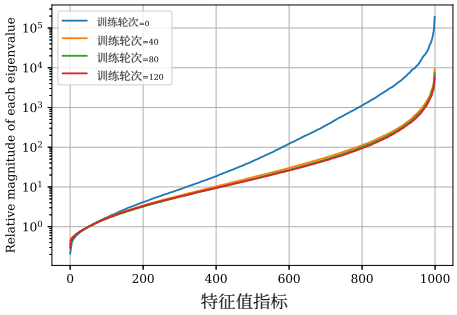
(b) GraSP



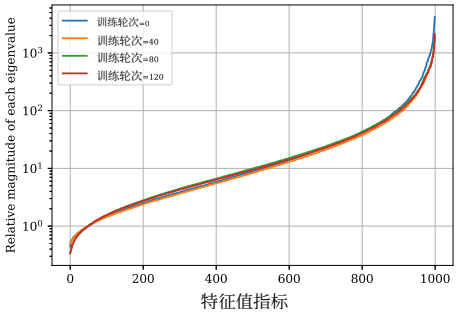
(c) 幅值



(d) SNIP



(e) Synflow



(f) NTK-SAP（我们的方法）

图 8：在稀疏度为 98.20% 的 ResNet-20（CIFAR-10）上训练期间固定权重神经切线核的相对特征值分布。我们将每个特征值的索引置于 x 轴上，并在 y 轴上呈现相应的相对特征值幅值。

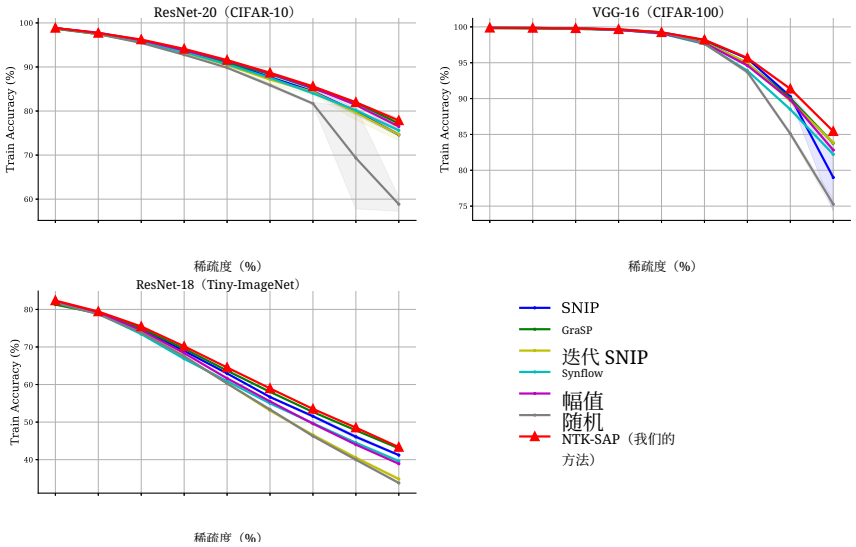


Figure 9: NTK-SAP 与其他前瞻性剪枝方法的训练准确率。结果为平均值 d over 3 个随机种子，阴影区域表示运行结果的标准差。

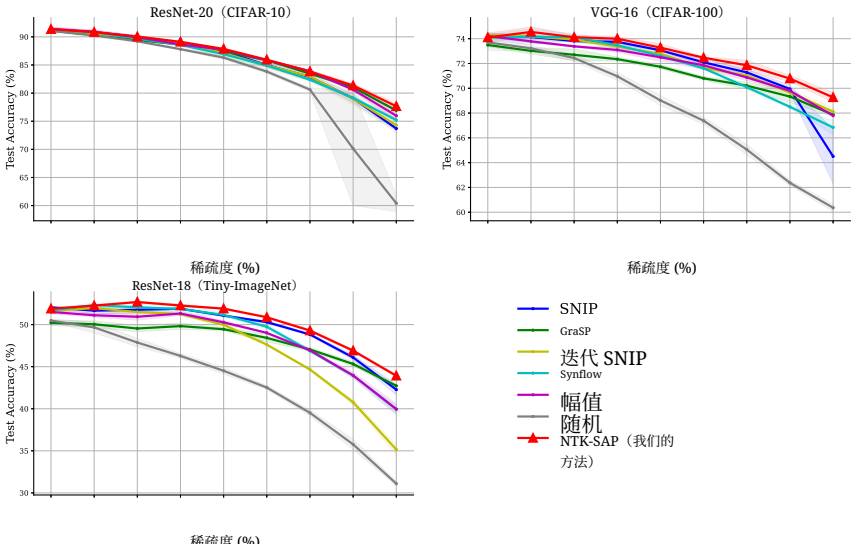


图10：NTK-SAP与其他前瞻性剪枝方法的测试准确率。结果取 3 个随机种子的平均值，阴影区域表示各次运行的标准差。

表 12：关于CIFAR-10（ResNet-20）上扰动超参数 ϵ 的消融研究。最佳结果以**粗体**显示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
$\epsilon = 1.0 \times 10^{-4}$†	91.36 ± 0.41	90.04 ± 0.23	87.86 ± 0.13	83.84 ± 0.28	**77.66 ± 0.50**
$\epsilon = 2.5 \times 10^{-5}$	91.34 ± 0.11	90.30 ± 0.17	87.94 ± 0.20	83.82 ± 0.24	77.33 ± 0.10
$\epsilon = 1.0 \times 10^{-5}$	**91.45 ± 0.13**	90.26 ± 0.06	87.64 ± 0.18	**83.92 ± 0.44**	77.53 ± 0.34
$\epsilon = 1.0 \times 10^{-6}$	91.41 ± 0.03	**90.44 ± 0.33**	**88.00 ± 0.36**	83.66 ± 0.23	76.90 ± 0.44

†用于正文的实验。

表13：在CIFAR-100（VGG-16）上对扰动超参数 ϵ 的消融研究。最佳结果以**粗体**显示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
$\epsilon = 1.0 \times 10^{-4}$†	74.10 ± 0.37	**74.12 ± 0.19**	73.28 ± 0.32	**71.85 ± 0.41**	**69.27 ± 0.39**
$\epsilon = 2.5 \times 10^{-5}$	74.25 ± 0.16	74.09 ± 0.20	73.20 ± 0.24	71.49 ± 0.08	68.97 ± 0.16
$\epsilon = 1.0 \times 10^{-5}$	**74.31 ± 0.41**	74.08 ± 0.20	73.18 ± 0.39	71.68 ± 0.25	69.24 ± 0.21
$\epsilon = 1.0 \times 10^{-6}$	74.27 ± 0.22	73.90 ± 0.25	**73.37 ± 0.37**	71.49 ± 0.13	69.26 ± 0.48

†用于正文的实验。

表 14：在 Tiny-ImageNet（ResNet-18）上对扰动超参数 ϵ 进行消融研究。最佳结果以**粗体**显示。

稀疏度	36.00%	73.80%	89.30%	95.60%	98.20%
$\epsilon = 1.0 \times 10^{-4}$†	51.89 ± 0.38	**52.69 ± 0.14**	51.90 ± 0.12	49.29 ± 0.15	**43.92 ± 0.39**
$\epsilon = 2.5 \times 10^{-5}$	**52.10 ± 0.51**	52.18 ± 0.16	51.95 ± 0.26	**49.48 ± 0.38**	43.63 ± 0.19
$\epsilon = 1.0 \times 10^{-5}$	51.86 ± 0.16	51.97 ± 0.46	**52.00 ± 0.30**	48.83 ± 0.09	43.36 ± 0.53
$\epsilon = 1.0 \times 10^{-6}$	51.94 ± 0.45	52.31 ± 0.19	51.32 ± 0.20	49.15 ± 0.38	42.94 ± 0.22

†用于正文中的实验。

H 主要实验的完整图表

我们进一步提供了第 6.1 节所报告的主要实验的完整训练/测试准确率曲线。结果如图9和图10所示。NTK-SAP 比其他前瞻性剪枝方法具有更高的训练准确率，表明其能够识别出更易于训练的子网络。

通常，所有前瞻性剪枝方法的测试性能都会随着诱导的稀疏度增加而下降。并且，随着稀疏率的提高，性能下降的速度会加快。此外，对于 VGG-16 (CIFAR-100)和 ResNet-18 (Tiny-ImageNet)，可以清楚地看到，NTK-SAP 的测试性能先提高后下降，这表明剪枝带来了泛化能力的提升。然而，在ResNet-20 (CIFAR-10) 中并未观察到这一现象。

I 关于扰动超参数的消融研究

在本节中，我们对选择不同扰动超参数 ϵ 的影响进行了消融研究。我们从 $\{1.0 \times 10^{-4}, 2.5 \times 10^{-5}, 1.0 \times 10^{-5}, 1.0 \times 10^{-6}\}$中选择 ϵ，其中 1.0×10^{-4} 是正文中使用的值。CIFAR-10、CIFAR-100 和Tiny-ImageNet 上的结果分别如表 12、13 和 14 所示。

表格显示，NTK-SAP 对扰动超参数 ϵ的选择具有鲁棒性。

表 15：所考虑的前瞻剪枝方法概览。

剪枝方法	权重无关	无数据	迭代方法
SNIP（Lee 等人，2018年）			
GraSP（Wang 等人，2020年）			
ProsPr（Alizadeh 等，2022年）			
FORCE（de Jorge 等，2020b）			✓
迭代 SNIP（de Jorge 等，2020b）			✓
Synflow（Tanaka 等人，2020年）		✓	✓
NTK-SAP（我们的方法）	✓	✓	✓

J 所考虑的前瞻剪枝方法概述

Synflow 和 NTK-SAP 是仅有的两种无数据前瞻剪枝方法。 本节在表 15 中总结了所有提及的前瞻剪枝方法。Synflow 和 NTK-SAP 是仅有的不需要任何数据的前瞻剪枝方法。这意味着剪枝完成后，找到的子网络可以轻松应用于不同的数据集。此外，据我们所知，NTK-SAP 是首个权重无关的前瞻剪枝方法。

K NTK-SAP 与神经切线迁移的区别

本节简要讨论 NTK-SAP 与神经切线迁移（NTT）（Liu & Zenke，2020年）之间的区别。NTT 同样利用神经切线核理论来指导前瞻剪枝。其目标是联合优化$\mathbf{m}$ 和 $\boldsymbol{\theta}$，使剪枝模型的输出和固定权重神经切线核尽可能接近稠密模型的对应结果。

然而，显式构建固定权重神经切线核的计算成本远高于估计 NTK-SAP中所使用的谱。因此，对于包括 ResNet-18、ResNet-34 和ResNet-50在内的大型模型，使用 NNT的成本可能过高。[11] 事实上，原始 NTT 论文仅考虑了多层感知机、LeNet 和 Conv-4 等小型网络。

另一个问题是：我们能否将NTK-SAP 中使用的近似方法应用于 NTT？我们认为这并非易事。NTK-SAP的思路是移除那些对固定权重神经切线核谱影响最小的掩码。因此，剪枝后谱很可能大致保持不变。相反，NTT作为一种基于优化的方法，可能需要额外修改损失函数，以惩罚固定权重神经切线核谱的变化，因为两个谱可能具有完全相同的总和，但其累积分布函数（CDF）却完全不同。

L 与 ProsPr 的比较

在提交之前，我们注意到 ProsPr（Alizadeh 等，2022）最近作为一种新的、具有竞争力的前瞻剪枝方法发表了。我们尝试复现 ProsPr 的结果，以便对 ProPr 和 NTK-SAP 进行公平比较。[12] 然而，我们未能复现 ProsPr 原论文中报告的结果。我们怀疑这是因为我们使用的超参数与作者所用的不同，因此向作者询问了他们的超参数。作者告诉我们，超参数信息尚未公布，之后会择期发布。待这些信息公布后，我们将进行此项比较。

[11]请参阅 NTT GitHub 页面中问题#2的相关讨论。[12]请注意，ProsPr 使用的架构与其采用的基准（Frankle 等，2020b）之间存在一些差异。

不过，我们想强调的是，ProsPr 是一种依赖数据的方法（见表 15），需要使用数据来执行剪枝过程。相反，NTK-SAP 不需要任何数据。模型完成剪枝后，无需额外计算即可应用于任何数据集。

M 关于 NTK-SAP 的权重无关性质的讨论

在第 5.3 节中，我们讨论了 NTK-SAP 的权重无关性质。这里我们进一步讨论，一个良好的剪枝方案是否也应当具有权重无关性，至少在期望意义上是如此。

我们认为，这取决于剪枝方法的类型。对于事后剪枝，一个良好的剪枝方案可能依赖于权重。例如，在使用一种良好的事后剪枝方法“彩票算法”时（Frankle 与 Carbin，2018年），依赖掩码的权重配置表现优于随机权重配置。

然而，正如（Frankle 等，2020b）所指出的，大多数现有的前瞻性剪枝方法对新采样的初始化并不十分敏感。一个可能的原因是，前瞻性剪枝方法无法获得足够的信息来学习图像的局部特征/模式（Pellegrini 与 Biroli，2022年），或剪枝方案的吸引盆（Evci 等人，2022年）。因此，从这个意义上说，一个良好的前瞻剪枝方案可能具有权重无关性。无论如何，这一论点并不能决定性地证明，最佳的前瞻性剪枝方法应当具有权重无关性。
另一个问题是：既然 NTK-SAP 旨在寻找期望表现良好的子网络，那么如果初始化非常糟糕，与二值掩码的重叠为零，会发生什么情况？这个网络能训练得好吗？我们认为，这样的网络确实不应该训练得很好。不过，这种情况发生的概率为 0，因为已知的初始化方法（例如 Xavier 或 Kaiming 归一化）会从连续随机分布中抽取初始权重。此外，如果允许采用最坏情况的初始化，稠密网络也可能遇到全零初始化，从而导致结果不佳。

N 为什么 NTK-SAP 可能避免层坍塌

我们在第 5.3 节提到，NTK-SAP 使用迭代剪枝来避免层坍塌。以下是对 NTK-SAP 为何能隐式避免层坍塌的一种可能解释：

首先，Synflow（Tanaka 等人，2020年）证明，基于突触显著性的前瞻性剪枝方法普遍遵循守恒定律，而迭代剪枝可以帮助它们避免层坍塌。因此，作为一种迭代的、基于突触显著性的剪枝方法，NTK-SAP 理论上也能够避免层坍塌。

其次，Lee 等人（2019b）提出了一种基于 SNIP 的重新初始化方法，旨在层坍塌即将发生时改善信号传播。这项工作揭示，当层坍塌即将发生时，逐层输入输出雅可比矩阵的谱会极差，从而破坏逐层动态等距性。同样，神经切线核的谱也可以作为此类指标。具体而言，可以证明，当层坍塌即将发生时，神经切线核的谱也会恶化。这里，我们给出一个简单的例子，以便更好地理解。例如，当层坍塌发生时，深度神经网络的权重梯度将为零，而固定权重神经切线核将是一个所有特征值均为零的零矩阵。从这个意义上说，NTK-SAP 的工作方式与迭代 SNIP（de Jorge 等，2020b）非常相似，都是为了避免层坍塌。

有限差分公式的论证

在本节中，我们将论证第 5.2 节所采用的有限差分公式。为论证这一近似方法，我们已经

$$
\begin{aligned}
&\mathbb{E}_{\Delta\boldsymbol{\theta}\sim\mathcal{N}(\mathbf{0},\epsilon\mathbf{I})}\left[\|f(\mathcal{X};\boldsymbol{\theta}_0)-f(\mathcal{X};\boldsymbol{\theta}_0+\Delta\boldsymbol{\theta})\|_2^2\right]\\
\approx&\mathbb{E}_{\Delta\boldsymbol{\theta}}\left[\|\nabla_{\boldsymbol{\theta}_0}f(\mathcal{X};\boldsymbol{\theta}_0)\Delta\boldsymbol{\theta}\|_2^2\right]\\
=&\sum_{i,j}\left[\nabla_{\boldsymbol{\theta}_0}f(\mathcal{X};\boldsymbol{\theta}_0)\right]_{i,j}^2\mathbb{E}\left[\Delta\theta_j\Delta\theta_j\right]\\
=&\epsilon\|\nabla_{\boldsymbol{\theta}_0}f(\mathcal{X};\boldsymbol{\theta}_0)\|_F^2.
\end{aligned}
$$