文章编号：2095-2163(2023)12-0009-10 中图分类号：TP183 文献标志码：A

利用互信息的自适应阈值通道剪枝方法

陈 震，杨 娟
（合肥工业大学 计算机与信息学院，合肥 230000）

摘 要：深度神经网络在计算机视觉、自然语言处理等方面表现出优秀的效果，但同时需要极高的计算运行成本以及适配的硬件资源，网络剪枝则是解决上述问题的有效方法。在已有的研究中多采用正则化方法将参数收敛到 0 值，并且使用统一阈值对网络进行修剪，进而导致剪枝不足或过度剪枝影响网络的精度。对此，本文提出了基于互正则化的层适应阈值剪枝策略（Mutual Regulation and Threshold Selection，MRTS），结合了互信息的正则化方法，将不重要的参数收敛到 0 值，重要的参数收敛到非 0 值，提高了通道的重要性区分度，便于后续网络剪枝。此外，本文根据神经网络层间参数的分布情况，自适应地计算出每一层的剪枝阈值，以修剪不重要的通道，在不影响精度的前提下进一步减少神经网络的计算量。实验表明，在 CIFAR10/100 和 ImageNet 数据集上，对比其他剪枝方法，MRTS 方法在提高模型稀疏度方面可以取得优异的成果。

关键词：网络剪枝；互信息；互正则化方法；深度神经网络

Adaptive threshold channel pruning method based on mutual information

CHEN Zhen，YANG Juan
（School of Computer and Information Technology，Hefei University of Technology，Hefei 230000，China）

Abstract：Deep neural networks have shown excellent results in computer vision and natural language processing，but they require high computational running costs and suitable hardware resources. However，most of the existing studies use regularization methods to converge the parameters to zero and prune the network using a uniform threshold，which leads to insufficient pruning or excessive pruning and affects the accuracy of the network. In this paper，we propose a layer-adaptive threshold pruning strategy Mutual Regulation and Threshold Selection（MRTS）based on mutual regularization，which combines the regularization method of mutual information to converge the unimportant parameters to 0 values and the important parameters to non-zero values，improving the importance distinction of channels for subsequent network pruning. In addition，this paper adaptively calculates the pruning threshold for each layer according to the distribution of parameters among the layers of the neural network to prune the unimportant channels and further reduce the computational effort of the neural network without affecting the accuracy. Experiments show that the structured pruning method using MRTS method achieves certain results on CIFAR10/100 and ImageNet datasets，especially in terms of high sparsity.

Key words：network pruning；mutual information；mutual regularization method；deep neural networks

0 引 言

随着理论不断完善和运算能力的提高，深度神经网络已经在语音、图像、文本等信息处理等领域取得了显著成果，但是过参数化一直是大家所担忧的问题，并且移动端设备对于文件体积较为敏感。因此，想要将深度神经网络模型移植到移动端，需要对其进行优化并保证其运行效率。针对于此，研究者们常用的是网络剪枝方法，该方法能够有效地提升模型性能且同时降低模型计算成本。近期，关于网络剪枝的研究主要分为非结构化剪枝[1-2]及结构化剪枝[3-5]两类。

文献[6]中提出了使用 L1 正则化对参数进行稀疏化训练，使得参数变得稀疏便于修剪，展示出了优异的剪枝效果。在此之后，一系列引入正则化方法训练比例因子进行剪枝的方法[7-8]被提出。这些方法引入 BN 层比例因子，通过 L1 正则化对 BN 层比例因子进行训练，接着对比例因子进行重要性排序，最后使用预定义的全局阈值用于修剪不重要的通道，这类方法在剪枝效果方面取得了很大的进展。但这些剪枝方法仍旧存在一定的局限性：

（1）使用 L1 正则化对比例因子进行训练，会将所有的比例因子收敛向 0 值，导致比例因子之间差

基金项目：国家自然科学基金（62106064）。

作者简介：陈 震（1999-），女，硕士研究生，主要研究方向：神经网络模型压缩。
通讯作者：杨 娟（1983-），女，博士，讲师，主要研究方向：深度学习。Email：yangjuan6985@163.com

收稿日期：2022-12-18

距较小,难以删除非重要通道;

(2)由于神经网络每一层的参数分布都是不同的,因此各层参数对于神经网络的重要性也不同,选择预定义的全局阈值对神经网络进行修剪,通常会导致修剪不足或是过度修剪,无法保留神经网络的原有性能。

针对上述问题,本文提出了互正则化以及阈值选择策略的解决方案。互信息[9]是指两个随机变量间的相互依赖程度,因此可以通过得到比例因子与该层网络输出的互信息来判断比例因子的重要性,这就是互正则化方法。重要的比例因子会通过互正则化方法训练后收敛到非 0 值,而不重要的比例因子收敛到 0 值,因此提高了比例因子的重要性区分度,便于后续的修剪工作。结合互正则化得到训练后的比例因子,根据每层的比例因子分布来获取适合当前层的阈值,对比例因子进行升序排序,小于此阈值的比例因子对应通道则被修剪,因此重要的通道得以保存,并且在保持高精度的同时,避免了修剪不足与过度剪枝的情况。

本文目标是在训练时能够更好地区分比例因子的重要性,其次要根据每层的重要性选择不同的阈值,在达到高稀疏性的同时最大程度保持网络原有性能。

1 相关工作

1.1 网络剪枝

网络剪枝可以在保持原有性能上降低计算成本,其中结构化剪枝方法可以在通道甚至网络层的层次上进行修剪,不仅能够保留原始的卷积结构还不需要特定的硬件或库来支持实现。剪枝方法的重要思想就是按照重要性排序将不重要的部分修剪掉,其中的关键就在于如何衡量重要性。一个最简单的思路就是按照参数绝对值大小来评估重要性,通常较小的参数会被认为是不重要的从而修剪掉。文献[5]中将权重的绝对值作为衡量其重要性的标准,但是原网络训练出的参数不够稀疏从而不利于修剪。解决这个问题的常用方法是在训练时添加正则化约束,通常是使用 L1 正则化训练权重,使得权重稀疏化。对于结构化剪枝来说,想要获得结构化的稀疏权重,常用的还有 LASSO 方法[10],该方法以组为单位对模型进行稀疏化训练。文献[11-12]中使用泰勒展开作为修剪标准,对过滤器进行剪枝;文献[4]中基于 BN 层的广泛使用,向 BN 层添加比例因子,并对其使用 L1 正则化使其稀疏化,之后裁剪比例因子中较小的值。虽然以上方法都在一定程度上实现了权重的稀疏化,但都存在将权重收敛到同一个值的问题,权重重要性之间的区分度不够明显,不利于之后的剪枝。为解决这个问题,文献[7]提出了使用权重与权重均值的差值作为偏项,对 L1 正则化进行改进,从而使得参数分布较为分散。但此方法并未考虑到权重与输出结果精度之间的联系,因此有一定的局限性。文献[13]提出使用 LASSO 回归方法增加参数重要性的区分度,其原理也是添加了 L1 正则化来约束权重,后续使用了最小二乘法来约束输出结果,但该方法使得剪枝步骤变得较为复杂。文献[14]中利用 kl 散度来度量通道的重要性,此方法虽然会将通道比例因子收敛到近似 0 值,但重要性区分度依旧不高。因此本文提出了互正则化,使用基于互信息的正则化方法使权重趋向两边,不仅考虑了权重之间的关系,也考虑到了权重与输出结果之间的关系,并且在稀疏化训练过程中就已经提高了权重重要性的区分度,为后续的剪枝操作简化了步骤,为权重排序以及剪枝操作提供了极大的便利。

网络剪枝过程中最重要的一步就是剪枝,因此将稀疏化训练后的权重按照升序排列后,裁剪较小的权重,不仅能够降低神经网络的计算量,同时可以较大程度保证神经网络原本的训练结果。传统的研究中会预先确定网络的剪枝比例,对于网络使用同一比例进行全局剪枝,这意味着将会有 $x\%$ 的通道将会被裁剪。由于每层网络的权重分布不同,预先设置的全局剪枝比例不可能满足每一层的权重分布情况,将会导致某些层剪枝不足或是剪枝过度。因此本文提出了一个层适应的阈值选择策略,根据每一层的权重分布去选取适合这一层的剪枝比例,可以最大程度提高稀疏度的同时保持原有网络的性能。

1.2 互信息

以往剪枝方法中会过多的考虑权重间的关系,如文献[6,10]中就采用对比权重的绝对值大小来判断重要性,但该方法在文献[15]中已被证明并不科学合理。除此以外,研究者们还会思考权重裁剪对于 Loss 值的影响,但这个想法在文献[16]中被发现需要计算 Hessian 矩阵,会浪费更多计算成本。本文提出的方法则是引入互信息概念,通过得到权重与输出结果间的依赖性来判断此权重的重要性,解决了以上方法的局限性。互信息会度量两个随机变量 X 和 Y 的依赖程度,即一个变量变化对另一个变量的影响程度。当互信息为 0 时,代表毫不相关,值越大相关性越强。因此本文选择使用互信息来获取权重与神经网络的依赖程度,两者之间的互信息值越大则代表当前权重对于神经网络的影响越大,这也代表着经过互正则化训练之后的重要权重会收

敛到一个较大值，使得不重要的权重收敛到 0 值，有效的提高了权重重要性的区分度。

2　MRTS

2.1　预设

典型的网络剪枝流程包括稀疏化训练、剪枝和微调三步骤。在稀疏化训练的过程中，稀疏约束性问题可以用式(1)解决：

$$\hat{\theta} = \arg\min_{\theta} L(\theta) + R(\theta) \tag{1}$$

其中，$L(\cdot)$、$R(\cdot)$ 和 θ 分别代表损失函数、正则化项和训练参数。

遵循以往通道剪枝的思想，稀疏正则化一般选择是 L1 正则化，同时本文为每一个神经元引入一个比例因子，将其表示为向量 $\boldsymbol{\gamma} \in R^n$。由于 BN 层在以往的网络剪枝工作中被广泛使用，因此引入 BN 层的比例因子作为每个神经元的比例因子。引入 BN 层比例因子后的神经网络稀疏训练的目标函数为

$$\hat{\theta} = \arg\min_{\theta} L(\theta) + \lambda \gamma_{L1} \tag{2}$$

式中：λ 控制稀疏度，γ_{L1} 指的是对神经元比例因子的 L1 正则化。

本文引入 BN 层比例因子是因为在卷积层之后经常跟随着 BN 层，所以比较容易找到其中的对应关系，无需额外引用参数。

L1 正则化通常把所有参数收敛到接近 0，使用的剪枝策略则是预先选择一个全局阈值进行剪枝操作。由于参数都接近 0 值导致没有明显区分度，并且使用全局阈值进行修剪，往往会导致过度修剪或是修剪不足，无法发挥网络的最优性能。为了解决上述问题，本文提出了互正则化以及阈值选择两个方法，只将不重要的参数稀疏化为 0 值，实现区分重要与不重要参数的目的，并且针对每一层参数动态选择最适合的阈值进行修剪，最终神经网络稀疏训练的目标函数为

$$\hat{\theta} = \arg\min_{\theta} L(\theta) + \lambda \varphi_{mr}(\boldsymbol{\gamma}) \tag{3}$$

其中，$L(\cdot)$、$\varphi_{mr}(\cdot)$ 和 θ 分别代表损失函数、互正则化和训练参数。

2.2　互正则化

基于稀疏训练的最终目标函数中引入了比例因子向量 $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \cdots, \gamma_n)$，其中 γ_i 代表施加在第 i 个神经元上的比例因子，每个比例因子都需要是正数并且有界，即 $\gamma_i \in [0, \varepsilon]$，$\varepsilon > 0$。L1 正则化要得到 $\min_{\gamma \in [0,\varepsilon]^n} \| \boldsymbol{\gamma} \|_1$，此式最优解为 0，因此 L1 正则化会将所有比例因子推向 0。而本文方法则是避免所有比例因子收敛到 0。互信息越大代表对神经网络影响越大，当前比例因子想要通过互正则化训练后趋向一个较大的值，则需要添加互信息偏项控制稀疏程度。互正则化方法与 L1 正则化方法对比结果如图 1 所示，互正则化表示为

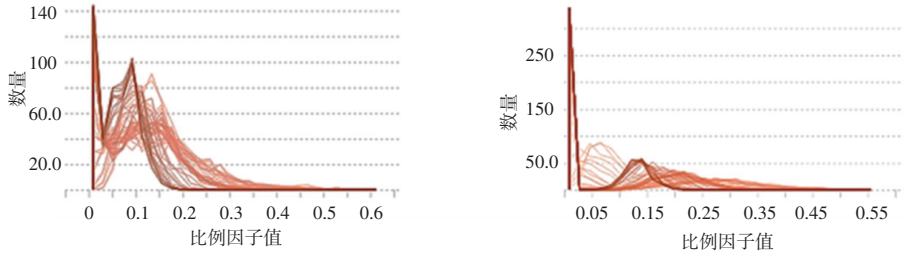
$$\varphi_t(\boldsymbol{\gamma}) = \tau \| \boldsymbol{\gamma} \|_1 - \| I(\boldsymbol{\gamma}; z_{\text{out}}) \|_1 = \sum_{i=1}^{n} \tau |\gamma_i| - |I(\gamma_i; z_{\text{out}}^i)|, (\tau \in R, \gamma_i \in [0, \varepsilon]) \tag{4}$$

式中：$I(\gamma_i; z_{\text{out}}^i)$ 代表 γ_i 对于神经网络的互信息。与 L1 正则化相同的是本文引入了超参数 τ 来控制 $\| \boldsymbol{\gamma} \|_1$ 的权重，不同之处在于增加了一个分值项 $- \| I(\boldsymbol{\gamma}; z_{\text{out}}) \|_1$，其目的是为了将比例因子 $\boldsymbol{\gamma}$ 区分开，避免比例因子收敛到一个值，使其根据互信息的值尽可能地向两边收敛，让不重要的继续向 0 值收敛，重要的则保留较大的数值。本文引入 BN 层比例因子，使用 z_{in}^i 和 z_{out}^i 作为第 i 个神经元 BN 层的输入输出，BN 层发生的变换为

$$\hat{Z} = \frac{z_{\text{in}}^i - E[z_{\text{in}}^i]}{\sqrt{Var[z_{\text{in}}^I]}}, \quad z_{\text{out}}^i = \gamma_i \hat{z} + \beta_i \tag{5}$$

其中，$\gamma_i = \sqrt{Var[z_{\text{in}}^i]}$、$\beta_i = E[z_{\text{in}}^i]$，$Var[z_{\text{in}}^i]$ 和 $E[z_{\text{in}}^i]$ 分别代表 z_{in}^i 的方差和均值。因此，式(4)中的 ε 应该足够大，使得

$$E > \gamma_i, (i \in [1, n]) \tag{6}$$



(a) L1 正则化　　(b) 互正则化

图 1　比例因子分布

Fig. 1　Scale factor distribution

2.3 阈值选择策略

在所有参数使用互正则化方法训练完成,得到向两边收敛的比例因子后,为了避免出现图 2 所示的剪枝不足以及过度剪枝情况,则要根据阈值选择方法来计算每一层合适的阈值。γ_{total} 代表一组比例因子总集合,γ_N 代表要被修剪的比例因子集合,γ_Y 代表要保留的比例因子集合。γ_N 集合里的比例因子都比较小,其和 $\sum_{\gamma\in\gamma_N}|\gamma|$ 也比较小,所以 $e^{\sum_{\gamma\in\gamma_N}|\gamma|}$ 值也会比较小,因此可以根据这个性质来求得阈值。

由于希望得到不重要比例因子所占总比例因子的比率 $\dfrac{\sum_{\gamma\in\gamma_N}|\gamma|}{\sum_{\gamma\in\gamma_{\text{total}}}|\gamma|}$,而 e^x 的值随着 x 的值增加而增加,所以可以通过求 $\dfrac{e^{\sum_{\gamma\in\gamma_N}|\gamma|}}{e^{\sum_{\gamma\in\gamma_{\text{total}}}|\gamma|}}$ 的值,来近似得到不重要因子的比例。则有

$$\frac{e^{\sum_{\gamma\in\gamma_N}|\gamma|}}{e^{\sum_{\gamma\in\gamma_{\text{total}}}|\gamma|}}\leqslant\frac{e^{n_{\gamma_N}|\max\{\gamma_N\}|}}{e^{n_{\gamma_Y}|\min\{\gamma_Y\}|}}=e^{n_{\gamma_N}|\max\{\gamma_N\}|-n_{\gamma_Y}|\min\{\gamma_Y\}|}\tag{7}$$

式中:$\dfrac{\sum_{\gamma\in\gamma_N}|\gamma|}{\sum_{\gamma\in\gamma_{\text{total}}}|\gamma|}$ 代表不重要比例因子占所有比例因子的比率,$\sum_{\gamma\in\gamma_N}|\gamma|$ 代表所有不重要比例因子的和,$\sum_{\gamma\in\gamma_{\text{total}}}|\gamma|$ 代表所有比例因子的和。为得到近似值,本文将分式的值适当进行缩放,n 代表集合里元素的数量,$\max\{\cdot\}$ 代表集合中的最大值,$\min\{\cdot\}$ 代表集合中的最小值。由于不重要的阈值都趋于 0,即令 $\max\{\gamma_N\}=0$,因此可将式(7)近似为

$$\alpha=e^{1-n_{\gamma_Y}|\min\{\gamma_Y\}|}\tag{8}$$

为了得到更准确的近似值,可以得到:

$$\frac{e^{\sum_{\gamma\in\gamma_N}|\gamma|}}{e^{\sum_{\gamma\in\gamma_{\text{total}}}|\gamma|}}\geqslant\frac{e^{n_{\gamma_N}|\max\{\gamma_N\}|}}{e^{n_{\gamma_{\text{total}}}|\max\{\gamma_{\text{total}}\}|}}=e^{n_{\gamma_N}|\max\{\gamma_N\}|-n_{\gamma_{\text{total}}}|\max\{\gamma_{\text{total}}\}|}\tag{9}$$

同上,将式(9)近似为

$$\beta=e^{1-n_{\gamma_{\text{total}}}|\max\{\gamma_{\text{total}}\}|}\tag{10}$$

其中,n 代表集合中的因子数量,所求的不重要比例因子比率 δ 满足 $\beta\leqslant\delta\leqslant\alpha$。

所求阈值 γ_t 满足:

$$e^{\sum_{\gamma\in\gamma_N}|\gamma|}<\delta e^{\sum_{\gamma\in\gamma_{\text{total}}}|\gamma|}\leqslant e^{\sum_{\gamma\in\gamma_N}|\gamma|+\gamma_t}\tag{11}$$

其中,δ 为不重要比例因子所占比率;γ_t 为阈值;$\sum_{\gamma\in\gamma}|\gamma|$ 代表集合 γ 里所有元素的和。

计算阈值首先需要对比例因子进行升序排列,然后找到的第一个满足公式(11)的比例因子(阈值 γ_t),之后就可以根据阈值对不同的 BN 层以及不同的稀疏级别来进行修剪,修剪比例因子小于 γ_t 的通道,剪枝后损失的精度可以通过微调(剪枝后的模型使用小学习率继续训练)或者重新训练(保留剪枝后的结构但不保留剪枝后的权重,随机初始化权重后再进行训练)来进行恢复,达到高稀疏性,并且最大程度保持了原有性能。

图 2 中,蓝色线条代表剪枝不足、红色线条代表过度剪枝、绿色箭头代表与互正则化方法结合后,使用阈值选择方法所选择的最佳阈值。

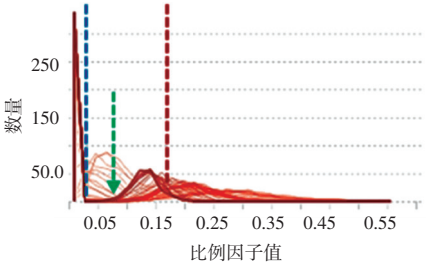


图 2 阈值选择

Fig. 2 Threshold selection

3 实验结果与分析

本文通过大量实验评估 MRTS 方法在分类图像上的效果,在不同的数据集上压缩不同的模型来评估 MRTS 的有效性,在 CIFAR-10、CIFAR-100 以及 ImageNet 上压缩 VGG 和 ResNet 用于图像分类,并且使用计算量(FLOPs)作为衡量标准。

3.1 数据集

根据文献[16]所提观点,在小数据集和大数据集上使用同样的剪枝方法,所呈现的结果不尽相同。因此,本文分别使用小数据集(CIFAR-10/CIFAR-100)和大数据集(ImageNet)来判断方法的有效性。

CIFAR-10 数据集包含 10 类 60 000 张 32×32 彩色图像,其中包括 50 000 张训练图像和 10 000 张测试图像。CIFAR-100 数据集包含 100 类 60 000 张图像,其中包括 50 000 张训练图像和 10 000 张测试图像。ImageNet 数据集一直是评估图像分类算法性能的基准,是一个大规模带标签图像的数据集,大约有 1 500 万张图片,2.2 万个类别。本文使用其子集 ISLVRC-2012,训练集含有 128 万张图片,验证集为 50 000 张图片,这些图片隶属于 1 000 个不同类别。本文使用深度神经网络结构 VGG-Net 与

ResNet，在 CIFAR-10、CIFAR-100 以及 ImageNet 上评估 MRTS 方法，在每个数据集以及不同网络结构上，与已发表过的该数据集和网络结构的实验结果进行了对比。

3.2　实验设置

实验基于 VGG 模型，将 BN 层添加到 VGG 的全连接层，其次基于 ResNet 模型[17]，本文的代码使用 NS 方法[4]作为基线实现。训练稀疏性过程中，采用 Nesterov 动量为 0.9、重量衰减为0.000 1的随机梯度下降法(SGD)进行优化，网络经过 200 个 epoch 的训练，初始学习率为 0.01，学习率在第 60、120、160 个 epoch 处衰减。实验中对模型进行了改良，在 ReLu 前增加 BN 层，用 BN 层来代替 Dropout 层。本实验代码是基于 PyTorch 和 Torchvision 两种工具实现。为简单起见，根据实验组的实验设置，由典型值估计得到 $\tau = 1.5$、$\delta = 0.001$，用于所有实验。对于 ResNet 这种具有跳跃连接的模型，根据 2.3 节计算的阈值直接删除整个分支。虽然有可能不会破坏网络，但与其他并行分支共享输入通道的 BN 层却不能被直接裁剪，对于这种类型的 BN 层，本文仍遵循 NS 中的方法，使用通道选择操作来修剪可忽略的通道。

3.3　实验结果与分析

以往的研究工作通常只会使用单独数据集或单个模型进行实验，这给本实验组与以前实验结果对比造成了困难。因此，在不同的数据集和模型上，本文只选择与在相同数据集以及相同模型上发表过的实验结果进行对比。除此之外，有一些方法使用了额外的辅助器，如 CCPrune[18]，为了公平地对待实验结果，本文将不会对这些方法进行比较。

对于 VGG-Net，本文选择 VGG-16 网络结构，将其在 CIFAR-10 以及 CIFAR-100 上进行训练，使用 MRTS 方法进行剪枝并对剪枝后的网络进行微调；ResNet 选择 ResNet-56 网络结构，将其分别在小数据集(CIFAR-10、CIFAR-100)和大数据集(ImageNet)上进行训练，使用 MRTS 方法进行剪枝并对剪枝后的网络进行微调。实验结果见表 1，其中基线精度代表剪枝之前的准确率，剪枝后精度代表剪枝之后的准确率变化。从表中可以发现，使用 MRTS 结构化剪枝之后的神经网络结构输出精度变化可以忽略不计，这代表本文所使用的互正则化方法，成功的提高了不重要比例因子与重要比例因子的区分度，且阈值选择策略成功选出了较为合理的阈值。因此，修剪掉的基本是对于神经网络输出影响不大的神经元，这些实验结果证明了 MRTS 方法的合理性。

表 1　剪枝前后准确率对比图

Table 1　Comparison of accuracy before and after pruning

模型	数据集	基线精度/%	剪枝后精度/%
VGG-16	CIFAR-10	93.63	93.38
	CIFAR-100	73.3	72.6
ResNet-56	CIFAR-10	93.15	93.36
	CIFAR-100	71.85	70.68
	IMAGENET	76.93	76.15

为了进一步评估 MRTS 方法的有效性，本实验组对比了不同修剪方法在精度和计算量方面降低的性能，展示了在同样的实验设置下使用不同方法，对同一网络结构在同一数据集上所得到的精度下降值以及相对应的计算量下降对应值。精度下降值指的是剪枝后的神经网络输出精度与基线模型输出精度的差值，计算量下降值在此代表的是剪枝率，即剪枝后的模型计算量与基线模型计算量差值的比率值。广泛使用于图像分类的较小数据集是 CIFAR 数据集，该数据集也是神经网络剪枝方法中最常见的数据集之一，因此本文使用的小数据集也是 CIFAR 数据集。在 CIFAR-10 数据集上的实验结果与其他剪枝方法对比情况见表 2。

表 2　CIFAR-10 上不同剪枝方法训练模型对比

Table 2　Comparison of training models with different pruning methods on CIFAR-10

数据集	模型	剪枝方法	精度下降/%	计算量下降/%
CIFAR-10	VGG-16	NS[4]	0.23	52
		FPGM[19]	0.06	34
		SFP[20]	1	62.8
		HRANK[21]	−0.3	54
		Ours	**0.3**	72.8
		AMC[22]	0.9	48
		NISP[23]	0.02	42
		PFEN[6]	−0.03	28
	ResNet-56	NS[4]	0.6	49
		CP[13]	1	50
		FPGM[19]	0.1	53
		SFP[20]	0.3	50
		HRANK[21]	2	73.8
		CPOT[24]	1.3	52.6
		Ours	**−0.2**	62.7

ResNet-56 模型在 CIFAR-10 数据集上的表现对比其他剪枝方法有着明显的优势，表中的精度下降一栏为负值，则代表剪枝后的模型比基线模型得到了更高的输出精度，MRTS 方法获得了精度提高以

及在修剪率为 62.7%时的最佳修剪精度。使用 MRTS 剪枝方法后,对比发现 VGG-16 模型在 CIFAR-10 数据集上的计算量下降值最大,同时精度下降情况与其他方法相差不大。在小数据集 CIFAR-100 上训练 VGG-16 模型及 ResNet-56 模型的实验结果与其他剪枝方法对比情况见表 3。在此之前很少有剪枝工作在 CIFAR-100 上进行实验,所以这方面的实验结果较少。从表中可以看出,ResNet-56 模型在 CIFAR-100 数据集上的精度变化与基线基本持平,但是剪枝率却达到了基线方法的 2 倍,这表明 MRTS 方法依旧达到了最佳结果,在保持精度的同时尽可能减少网络的计算量。

表 3 CIFAR-100 上不同剪枝方法训练模型对比

Table 3 Comparison of training models with different pruning methods on CIFAR-100

数据集	模型	剪枝方法	精度下降/%	计算量下降/%
CIFAR-100	VGG-16	NS[4]	-0.37	38
		COP[25]	0.9	42.8
		Ours	**0.7**	58.6
	ResNet-56	NS[4]	1.1	25
		FPGM[19]	6.04	51.8
		CPOT[24]	1.3	52.1
		Ours	**1.2**	54.2

由于 ImageNet 数据集与 CIFAR 数据集相比要大很多,因此受资源所限,仅仅对 ResNet-56 模型进行了训练。在 CIFAR-10 数据集上训练 ResNet-56 的实验结果与其他剪枝方法对比情况见表 4。在表中不难看出,各方法的计算量下降值相差较小,但是 MRTS 方法在精度下降方面达到了最优的效果,得到了最小的精度下降值和最好的修剪精度。

表 4 ImageNet 上不同剪枝方法训练模型对比图

Table 4 Comparison of training models with different pruning methods on ImageNet

数据集	模型	方法	精度下降/%	计算量下降/%
ImageNet	ResNet-56	NS[5]	1.21	52
		FPGM[19]	1.93	51.8
		SFP[20]	16.3	53
		ours	**0.78**	53.5

总之,对比以往的剪枝方法,MRTS 方法在小数据集、大数据集上都获得了最优的结构化剪枝结果。

为了进一步展示互正则化的有效性,使用互正则化训练后的 ResNet-56 与 VGG-16 中,BN 层比例因子分布如图 3 所示。由此可见,每个子图中比例因子分布会存在两个峰值,一个峰值在 0 值附近,另一峰值在一个较大非 0 值附近,两峰值所处位置有着明显差异,这种差异证明了互正则化方法是有效的。

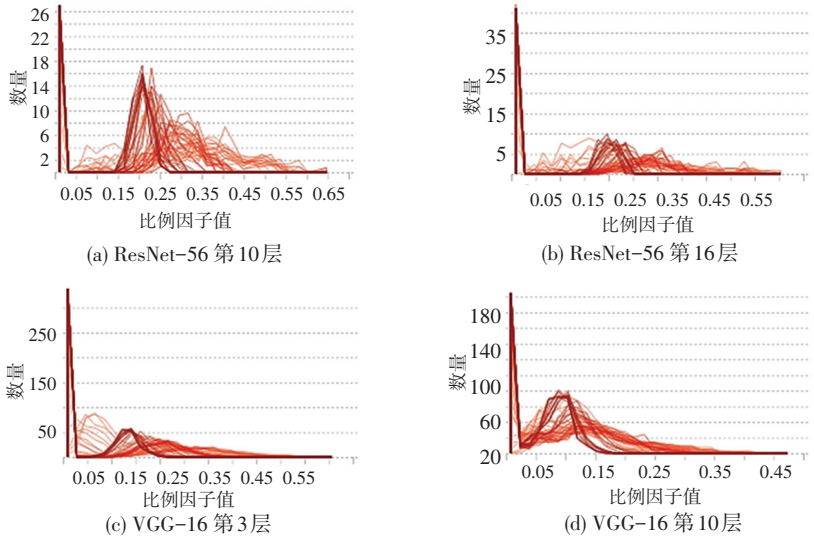
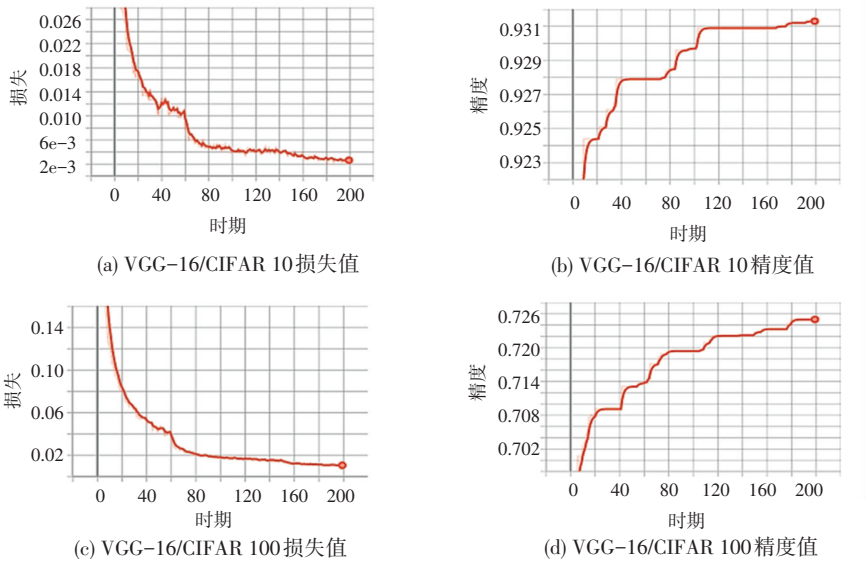


图 3 互正则化训练后的比例因子分布直方图

Fig. 3 Histograms of the scale factor distributions after mutual regularization training

为了更直观的展现实验结果,VGG-16 与 ResNet 模型在 CIFAR-10/CIFAR-100 数据集上的训练损失函数值与精度值变化情况如图 4、图 5 所示。由于互正则化方法可以通过调节超参数 τ 来控制推向 0 值的参数比例,τ 取值越大,收敛到 0 值的参数就越多,则将被修剪的神经元也越多,但并不是 τ 越大越好。通过实验,本文选择 τ = 1.5,在控制精度的情况下降低计算量。其次,动态的选择层相关的阈值,避免剪枝不足与过度修剪情况。从图中可以看出,MRTS 方法在提高精度以及降低损失方面都得到了较为优异

的成果。通过互正则化对比例因子进行重要性区分，后续的阈值选择方法则是选出层相关的最合适阈值，两个方法结合后才能得到 MRTS 方法优于其他剪枝方法的表现。

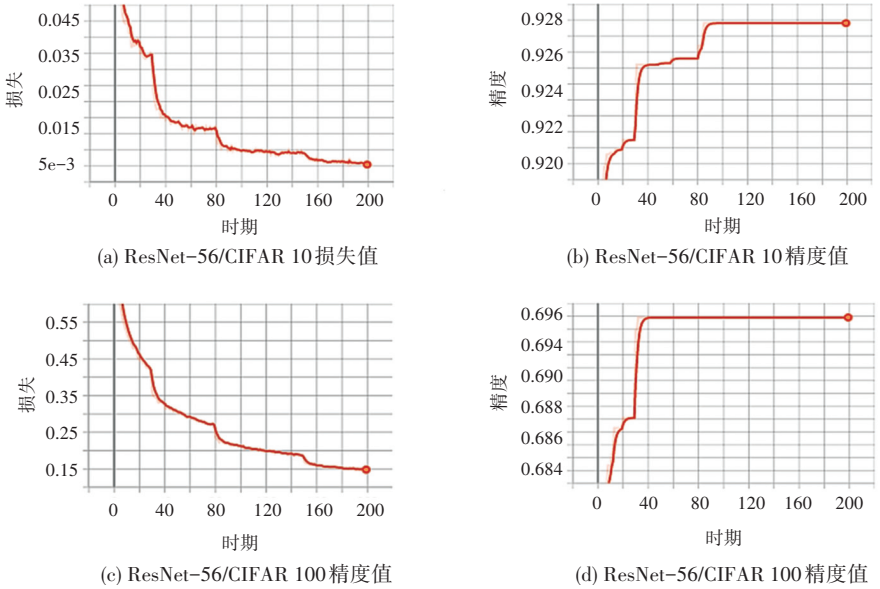


(a) VGG−16/CIFAR 10损失值　(b) VGG−16/CIFAR 10精度值

(c) VGG−16/CIFAR 100损失值　(d) VGG−16/CIFAR 100精度值

图 4　VGG 在 CIFAR−10/CIFAR−100 上训练情况

Fig. 4　VGG training on the CIFAR−10/CIFAR−100



(a) ResNet−56/CIFAR 10损失值　(b) ResNet−56/CIFAR 10精度值

(c) ResNet−56/CIFAR 100精度值　(d) ResNet−56/CIFAR 100精度值

图 5　ResNet 在 CIFAR−10/CIFAR−100 上训练情况

Fig. 5　ResNet training on the CIFAR−10/CIFAR−100

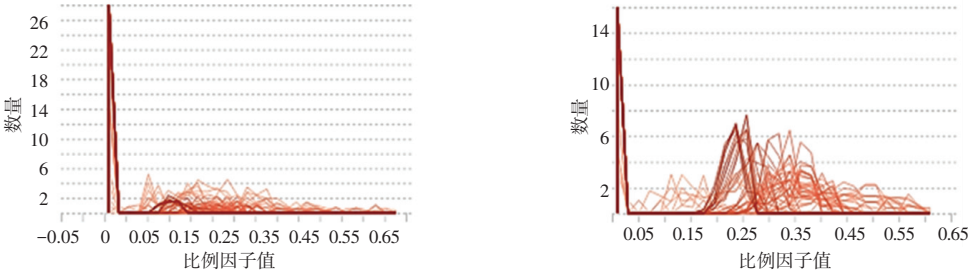
3.4　消融实验

为了验证互正则化与阈值策略的有效性，本文在 CIFAR−10 上对 VGG−16 网络模型进行了消融实验来验证剪枝方法的有效性。

本文使用 NS 方法[4]作为基线方法，将互正则化以及阈值策略分别加入到基线当中，以此来验证两种方法是否都能有效的提高基线的性能。

互正则化的主要功能是改变神经网络中参数的训练方法，令训练后的参数向两级分化，而不是只收敛到 0 值，以此来提高参数的区分度。如图 6 所示，将互正则化方法添加到基线中，改变了基线原有的稀疏化方法，图 6(a)为基线训练后的某两层参数分布情况，图 6(b)则是加入了互正则化方法后与基线方法同层对应的参数分布情况。通过两图的参数分布图对比可以发现，基线方法经过训练后的参数基本都趋于 0 值，而经过互正则化方法训练后的参数则呈现

两极分化趋势,在 0 值以及一个较大的非 0 值周围都有强烈的波动。通过对比可以得出,基线训练后的参数与添加了互正则化方法相比,参数分布更集中趋向于 0 值,互正则化成功的实现了提高参数重要性区分度的功能,为后续的剪枝过程提供了极大的便利。

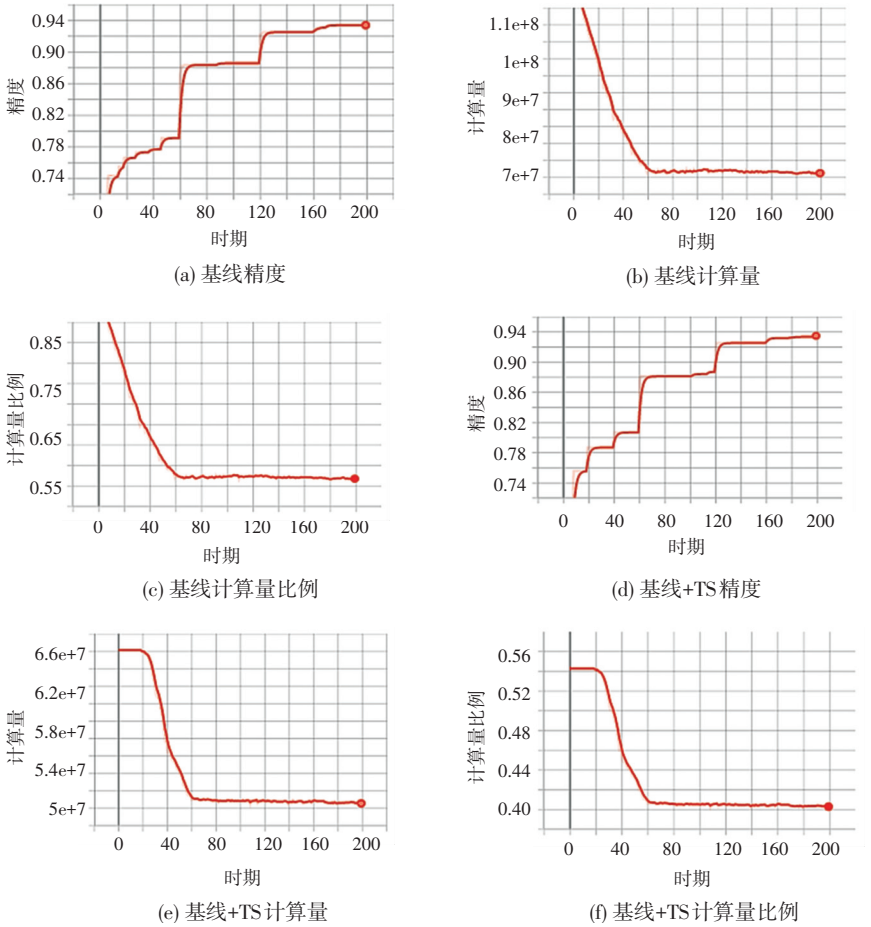


(a) L1 正则化 (b) 互正则化

图 6 不同稀疏化方法训练后的参数分布对比图

Fig. 6 Comparison of parameter distribution after training with different thinning methods

阈值选择的功能主要是根据每层参数分布,选择出合适的阈值对神经网络参数进行修剪,因此可以在保留原有网络精度的前提下,最大程度的修剪参数减少计算量。如图 7 所示,图 7 中(a)、(b)、(c)是基线剪枝后的结果,(d)、(e)、(f)是使用了阈值选择方法对模型进行剪枝后的结果展示。从剪枝后的模型训练精度对比发现,使用基线方法进行剪枝后的模型精度与使用阈值选择方法进行剪枝后的模型精度几乎没有差别;由计算量的变化情况可以明显看出,使用了阈值选择方法对模型进行剪枝的力度更强,基线方法将模型计算量降低到原模型计算量的 55%左右,而本文提出的阈值选择方法则是将计算量压到了 41%左右。因此,从精度对比以及计算量对比可得出,阈值选择算法的剪枝比例更高,且精度保留完整。



(a) 基线精度 (b) 基线计算量

(c) 基线计算量比例 (d) 基线+TS 精度

(e) 基线+TS 计算量 (f) 基线+TS 计算量比例

图 7 基线方法修剪网络与使用 TS 方法修剪网络后的准确率与修剪率对比

Fig. 7 Comparison of accuracy and pruning rate after pruning networks with baseline method and TS method

本文提出的 MRTS 是互正则化与阈值选择策略方法的结合，保留了两者各自的优点。通过互正则化对参数进行稀疏化训练提高参数重要性的区分度，使用阈值选择方法获得最适合每层的参数分布的阈值，然后对参数从小到大进行排列，修剪小于阈值的参数，保留了神经网络的原有性能，并且大程度的降低了计算量。图 8 显示了与基线、单独使用 MR 方法或单独使用 TS 方法相比 MRTS 方法的优越性，主要从精度以及计算量两方面来进行对比。由图 8 中可见，4 种方法的精度变化相差不大，即剪枝后都可以实现保留原有模型精度的功能；与基线相比互正则化提高了参数重要性的差异，便于后续修剪操作，同时阈值选择方法动态的选择出适应模型每一层的阈值进行剪枝，极大的优化了模型的计算量。由此可以得出，结合两种方法优势的 MRTS 在保持精度基本不变的同时，极大地降低了计算量，优化了模型性能，因此该方法无论在精度指标还是模型压缩，以及计算效率下都取得了较好的效果。

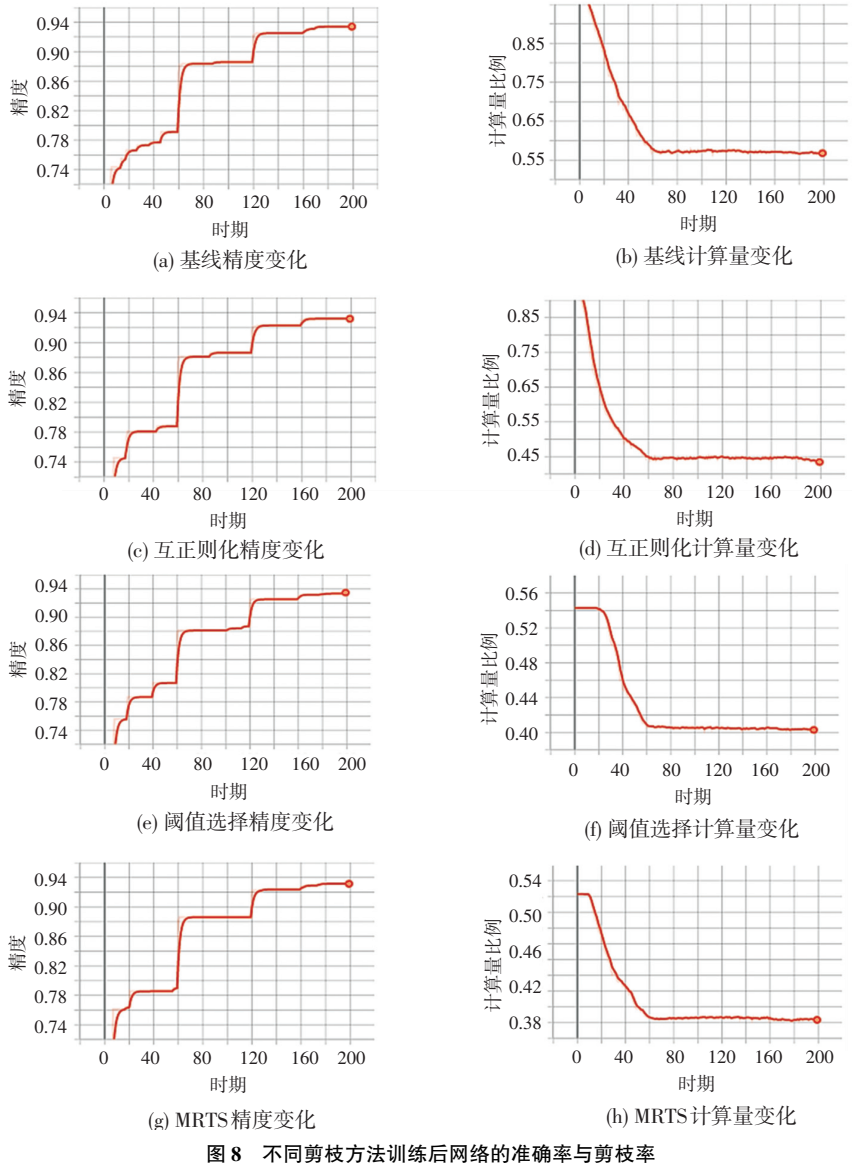


图 8　不同剪枝方法训练后网络的准确率与剪枝率

Fig. 8　Accuracy and pruning rate of the network trained by different pruning methods

4　结束语

本文提出了 MRTS 方法，其中包含两点，第一点是提出了互正则化，使用基于互信息的正则化方法对比例因子进行稀疏训练，将比例因子收敛到 0 以及一个较大的非 0 值，提高了不重要神经元与重要神经元的区分度。第二点是提出了阈值选择方法，与之前的全局阈值不同，其会根据每层比例因子的分布来确定

阈值。经过大量实验对比,本文的 MRTS 方法易于实现且剪枝率高,可以较好的保持网络的优良性能,后续将会在更多模型以及数据集上进行实验证明方法的有效性。

参考文献

[1] LECun Y, DENKER J, SOLLA S. Optimal brain damage[J]. Advances in Neural Information Processing Systems, 1989, 2:598-605.

[2] HAN S, MAO H, DALLY W J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding[J]. arXiv preprint arXiv:1510.00149, 2015.

[3] LUO J H, WU J, LIN W. Thinet: A filter level pruning method for deep neural network compression[C]//Proceedings of the IEEE International Conference on Computer Vision. 2017: 5058-5066.

[4] LIU Z, LI J, SHEN Z, et al. Learning efficient convolutional networks through network slimming[C]//Proceedings of the IEEE International Conference on Computer Vision. 2017: 2736-2744.

[5] 陈靓,钱亚冠,何志强,等.深度卷积神经网络的柔性剪枝策略[J]. 电信科学,2022,38(1):83-94.

[6] LI H, KADAV A, DURDANOVIC I, et al. Pruning filters for efficient convnets[J]. arXiv preprint arXiv:1608.08710, 2016.

[7] YE Y, YOU G, FWU J K, et al. Channel pruning via optimal thresholding[C]//Proceedings of the International Conference on Neural Information Processing. Springer, Cham, 2020: 508-516.

[8] ZHUANG T, ZHANG Z, HUANG Y, et al. Neuron-level structured pruning using polarization regularizer[J]. Advances in Neural Information Processing Systems, 2020, 33: 9865-9877.

[9] GIERLICHS B, BATINA L, TUYLS P, et al. Mutual information analysis[C]//International Workshop on Cryptographic Hardware and Embedded Systems. Springer, Berlin, Heidelberg, 2008: 426-442.

[10] WEN W, WU C, WANG Y, et al. Learning structured sparsity in deep neural networks[J]. Advances in Neural Information Processing Systems, 2016, 29.

[11] MOLCHANOV P, TYREE S, KARRAS T, et al. Pruning convolutional neural networks for resource efficient inference[J]. arXiv preprint arXiv:1611.06440, 2016.

[12] 张彪,杨朋波,桑基韬,等. 基于特征归因重要度评价的卷积网络剪枝[J]. 中国科学:信息科学,2021,51(1):13-26.

[13] HE Y, ZHANG X, SUN J. Channel pruning for accelerating very deep neural networks[C]//Proceedings of the IEEE International Conference on Computer Vision. 2017: 1389-1397.

[14] 杨鑫,袁晓彤. 利用 KL 散度度量通道冗余度的深度神经网络剪枝方法[J]. 计算机应用与软件,2021,38(11):300-306.

[15] YE J, LU X, LIN Z, et al. Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers[J]. arXiv preprint arXiv:1802.00124, 2018.

[16] GALE T, ELSEN E, HOOKER S. The state of sparsity in deep neural networks[J]. arXiv preprint arXiv:1902.09574, 2019.

[17] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016: 770-778.

[18] PENG H, WU J, CHEN S, et al. Collaborative channel pruning for deep networks[C]//Proceedings of the International Conference on Machine Learning. PMLR, 2019: 5113-5122.

[19] HE Y, LIU P, WANG Z, et al. Filter pruning via geometric median for deep convolutional neural networks acceleration[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019: 4340-4349.

[20] HE Y, KANG G, DONG X, et al. Soft filter pruning for accelerating deep convolutional neural networks[J]. arXiv preprint arXiv:1808.06866, 2018.

[21] LIN M, JI R, WANG Y, et al. Hrank: Filter pruning using high-rank feature map[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 1529-1538.

[22] HE Y, LIN J, LIU Z, et al. Amc: Automl for model compression and acceleration on mobile devices[C]//Proceedings of the European Conferenceon Computer Vision (ECCV). 2018: 784-800.

[23] YU R, LI A, CHEN C F, et al. Nisp: Pruning networks using neuron importance score propagation[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 9194-9203.

[24] SHEN Y, SHEN L, HUANG H Z, et al. Cpot: Channel pruning via optimal transport[J]. arXiv preprint arXiv:2005.10451, 2020.

[25] WANG W, YU Z, FU C, et al. COP: Customized correlation-based Filter level pruning method for deep CNN compression[J]. Neurocomputing, 2021, 464: 533-545.