中国心理学前沿
2024 年 11 月第 6 卷第 11 期

基于敏感性的卷积神经网络通道剪枝算法

杨晨彬

江苏警官学院，南京

摘　要 | 深度神经网络模型（convolutional neural networks，CNNs）从卷积层到全连接层都存在着大量冗余的参数，模型剪枝是一种用于减少深度卷积神经网络模型内存消耗和浮点运算（FLOPs）的优化技术，它通过剔除模型中“不重要”的权重，使得模型的参数量和计算量大大减少，同时尽量保证模型的精度不受影响，可以实现参数量与模型性能之间的最佳平衡。然而，对于较高的剪枝率的情况下，当前的通道剪枝方法往往都是将通道的范数大小用作剪枝的度量标准，缺乏相应的理论支撑，也难以取得理想的剪枝效果，这样就会导致压缩后模型精度的显著下降。为了解决这个问题，本文提出一种新的基于敏感性度量的通道剪枝方法，利用二阶敏感性作为标准来对通道的重要性进行衡量。通过理论推导，将传统敏感性计算的计算方式从针对单一权重拓展到了针对整个网络通道剪枝（Channel Pruning），首先从理论上证明了可以使用通道中所有权重敏感性总和的标准来量化整个通道的敏感性，接着根据得到的通道敏感性的相对大小来删减其中不敏感的通道，以此来完成通道剪枝。在各种不同网络模型结构上的实验表明，相较于传统方法，本文可以在牺牲少量精度的同时显著提高剪枝率。例如，在精度轻微损失的情况下，使CIFAR-10数据集上的FLOPs减少60%以上。并且，在ImageNet数据集上，基于ResNet34的剪枝减少了52.1%的FLOPs，而只损失了0.23%的精度。

关键词 | 卷积神经网络；通道剪枝；敏感性

1　引言

卷积神经网络（convolutional neural networks，CNNs）近年来发展迅速，已经在计算机视觉[1]、自然语言处理[2]和语音识别[3]等多个领域成为主流的应用网络模型。然而，为了获得更好的性能，实际应用时往往需要使用参数量更大、网络层数更深的模型。由于模型过于庞大的尺寸，它们很难直接部署在许多资源受限的智能设备中，因为这些设备一般都需要在有限的内存和计算条件下才能运行。为了提高计算效率和速度，研究者已经提出了许多方法来压缩神经网络的冗余信息。这些研

通讯作者：杨晨彬，江苏警官学院侦查系讲师，研究方向：人工智能。
文章引用：杨晨彬. 基于敏感性的卷积神经网络通道剪枝算法 [J]. 中国心理学前沿，2024，6（11）：46-53.
https://doi.org/10.35534/pc.0611006

究大致可以分为以下几类：网络量化[4]、张量分解[5]、模型剪枝[6]等。在这些方法中，模型剪枝是最常见的技术之一，并受到了相当多的关注，具有广阔的研究前景。

从剪枝结构上，模型剪枝通常分为两类：权重剪枝[7]（即非结构化剪枝）和通道剪枝[8]（即结构化剪枝）。权重剪枝从网络通道中移除特定无结构的权重，导致网络中权重矩阵过于稀疏。如果没有专门的硬件或软件算法支持，网络中非结构化的稀疏可能会影响模型的压缩加速效率，无法获得显著的性能增益。与之相对的，通道剪枝的目标是去除每个卷积层中所选择的一整个通道和相应的滤波器，使模型内部具有规则结构，在理论上可以取得更优的加速效果。在一些流行的通道剪枝方法中，通常采用逐层的方法进行通道剪枝。在每一层中，通过最小化下一层的重建误差类来选择并剪枝最不重要的通道。然后，通过再训练对网络参数进行微调，同时调整所有层中的参数以恢复模型的精度。

无论是权重剪枝还是通道剪枝，最基本的任务都是发现并删除最不重要的权重或通道。其中，最高效的方法是最优脑损伤算法[9]（optimal brain damage，OBD）和最优脑手术算法[10]（optimal brain surgeon，OBS），它们使用泰勒级数来估计每个权重被移除后损失函数的变化，并剪枝其中引起最小损失变化的权重。这些利用二阶敏感性的传统剪枝策略主要被用来处理神经网络中的过拟合问题，但是在最近的一项研究中，这个思路被成功地应用在深度神经网络中，作者提出了一种逐层剪枝的算法L-OBS[11]，使用二阶导数剪枝每层的权重。

上述三种方法使用了敏感性作为剪枝的度量标准，相较于常见的基于范数的度量标准更为高效、准确，但是它们都是权重剪枝的策略，并没有能够将敏感性运用于通道剪枝之中，这就会导致神经网络产生非结构化稀疏，实际应用受限。为了解决这个问题，本文提出了一种新的基于敏感性的通道剪枝方法，利用神经网络的二阶信息来确定每个通道的敏感性，由理论推导可知，通道的敏感性由其中所有权重的敏感性之和来衡量。对每一个卷积层中所有通道的敏感性进行估计和排序，可以消除不敏感的通道，同时保留那些具有相对较高敏感性的通道。通过这样的方式，可以在保持网络精度的同时实现深度卷积神经网络的高效压缩。

2 相关工作

模型剪枝是神经网络压缩优化领域中的最热门的研究方向之一，针对网络剪枝的研究最早可以追溯到90年代[12]。其中，相关研究工作[13, 14]是深度神经网络领域最著名的早期工作之一，其对网络中的权重进行了剪枝并取得了较好的压缩结果。如前所述，权重剪枝会导致网络中的非结构化稀疏，压缩效率较低，因此本文简要介绍通道剪枝的相关研究。

通道剪枝的目的是对每一卷积层的不重要的通道进行剪枝，通过简单高效的结构化剪枝来消除对专用软硬件的依赖。然而，这种方法也具有一定挑战性，因为删除一层中的某些通道可能会对下一层的输入产生重大影响，从而影响最终的网络输出精度。因此，确定一种有效的结构剪枝策略至关重要，如何确定所有通道中最不重要的通道是所有通道剪枝方法中都需要解决的首要问题。

目前，最流行的通道剪枝方法是基于范数的剪枝。在这种方法中，通道的范数大小被用作剪枝的判断标准。基于范数的通道剪枝假设范数小的通道的重要性是比较低的，可以安全地被去除。例如使用最小L2范数准则来选择需要剪枝的通道和滤波器[15]，并以软剪枝方式来进行剪枝操作。近几年，其他类似的基于范数标准的方法也被陆续提出，例如假设如果某些通道激活函数的大多数输出为零[16]，则这些通道可以被认为是冗余的，因此可以使用激活神经元的零值平均百分比（APoZ）作为标准，并去除具有较小APoZ的滤波器和通道。在此基础上进行了改进，其中每一层输出中的批归一化层的缩放因子被用作对应通道的重要性度量，具有较低缩放因子的通道被认为缺乏重要性并被剪枝[17]。假设低秩的输出特征图包含较少的网络信息并提出了HRank算法，从数学上定义特征图中的秩作为剪枝相应输出通道的标准[18]。

上述许多方法通过采用“范数越小越不重要”的准则进行通道剪枝，其中具有较小范数的滤波器因为其较低的重要性而被安全剪枝。然而，这个准则并不总是有效的，其中一个重要的原因是，具有小范数的通道也可能对输出非常敏感，从而对网络

https://doi.org/10.35534/pc.0611006

输出产生较大的影响[19]。使用网络的二阶泰勒级数展开很容易就可以验证这一点，其中通道对于输出的扰动依赖于网络的二阶Hessian矩阵。因此，本文成功地将二阶Hessian矩阵运用到神经网络的通道剪枝当中，采用了基于二阶敏感性的方法，在通道剪枝中取得了优异的压缩性能。

3 基于敏感性的通道剪枝算法

3.1 通道敏感性的定义与推导

在监督学习任务中，具有多个卷积层的深度神经网络的理论目标是最小化经验误差函数 $E(\omega)$，其优化问题可以表示如下：

$$E(\omega)=\frac{1}{N}\sum_{i=1}^{N}e(x_i,y_i,\omega), \tag{1}$$

其中 $\omega\in R^m$ 是每一层中的可训练权重 $\{W_1,W_2,\cdots,W_l\}$ 的组合，W_l 表示第l个卷积层的权重矩阵。$e(x_i,y_i,\omega)$ 是输入数据 x_i 的误差，其中 y_i 是对应的标签，N是训练集中训练样本数量。

然后，需要筛选出那些引起训练误差增量最少的剪枝扰动。对于$\delta\omega$变量使用泰勒级数展开，误差函数对应的增量为：

$$\delta E=g^T\delta\omega+\frac{1}{2}\delta\omega^T H\delta\omega+O\left(\|\delta\omega\|^3\right), \tag{2}$$

其中δ表示对应变量的扰动，$g\in R^m$ 表示误差函数E的梯度，$H\in R^{m\times m}$ 为对应的Hessian矩阵，m为整个权重的个数。对于一个已经训练到局部最小误差的预训练网络，可以认为其中的梯度g=0。此外，由于三阶和高阶项 $O\left(\|\delta\omega\|^3\right)$ 通常都较为复杂且影响较低，因此它们都被近似为0。

由此可知，对于一个由一系列权重组成的通道，通道剪枝是通过判断该通道导致对损失函数的最小扰动δE来完成的。因此，可以建立一个约束优化问题，在剪枝一个通道的约束下最小化误差函数可以表示为：

$$\min_{\delta\omega}\frac{1}{2}\delta\omega^T H\delta\omega=\frac{1}{2}\begin{pmatrix}\delta\omega_p\\ \delta\omega_q\end{pmatrix}^T\begin{pmatrix}H_{p,p} & H_{p,q}\\ H_{q,p} & H_{q,q}\end{pmatrix}\begin{pmatrix}\delta\omega_p\\ \delta\omega_q\end{pmatrix}, \tag{3}$$
$$s.t.\delta\omega_p+\omega_p=0\#$$

其中 ω_p 表示剪枝通道的数量为p的权重，ω_q 表示剩余的其他权重。同样，本文使用$\delta\omega_p$ 和 $\delta\omega_q$分别表示对 ω_p 和 ω_q的扰动。此外，$H_{p,p}$ 和 $H_{q,q}$ 分别表示对于剪枝和未剪枝权重的Hessian矩阵。这个优化问题可以用拉格朗日乘子法求解，对鞍点的两边同时求导：

$$L=\frac{1}{2}\delta\omega^T H\delta\omega+\lambda^T(\delta\omega_p+\omega_p)$$
$$\frac{\partial L}{\partial\delta\omega}=\begin{pmatrix}H_{p,p} & H_{p,q}\\ H_{q,p} & H_{q,q}\end{pmatrix}\begin{pmatrix}\delta\omega_p\\ \delta\omega_q\end{pmatrix}+\begin{pmatrix}\lambda\\ 0\end{pmatrix}=0, \tag{4}$$

其中 λ 为拉格朗日乘子。通过展开该方程，并考虑公式3中的约束 $\delta\omega_p=-\omega_p$，通过计算可以得到：

$$\delta\omega_q=H_{q,q}^{-1}H_{q,p}\omega_p \tag{5}$$

利用公式5的结果并重新排列公式3，可以得到如下结果：

$$L_{\omega_p}=\frac{1}{2}\delta\omega^T H\delta\omega$$
$$=\frac{1}{2}\omega_p^T\left(H_{p,p}-H_{p,q}H_{q,q}^{-1}H_{q,p}\right)\omega_p \tag{6}$$
$$=\frac{1}{2}\omega_p^T\left[H^{-1}\right]_{p,p}^{-1}\omega_p,$$

其中$\left[H^{-1}\right]_{p,p}$ 表示p维的Hessian逆矩阵。L_{ω_p} 表示 ω_p 的敏感性，即删除某一通道权重 ω_p 后对误差函数的扰动。然后，给定公式6，可以根据它们的敏感性来确定需要被剪枝的通道。

3.2 通道敏感性的简化与计算

由公式6可知，通道敏感性的计算需要对Hessian矩阵进行求解并且求逆运算，对于深度卷积神经网络来说，这样的计算量非常庞大，无法及时得到运算结果，因此考虑对其进行近似简化。

一个常用的方式是使用对角近似，即假设Hessian矩阵是一个对角矩阵，其非对角位置上的元素均为0，这样公式6就可以简化为：

$$L_{\omega_p}=\frac{1}{2}\omega_p^T Diag\left(H_{p,p}\right)\omega_p \tag{7}$$

其中 $Diag\left(H_{p,p}\right)$ 表示Hessian矩阵 $H_{p,p}$ 的对角元素。通过这样的方式就省略了对Hessian矩阵的求逆运算，计算量即可得到很大程度的压缩。

同时，将公式7改写成累加求和的方式，可得：

$$L_{\omega_p}=\frac{1}{2}\sum_{i=1}^{p}H_{ii}*\left(\omega_p^i\right)^2$$
$$=\sum_{i=1}^{p}L_i\# \tag{8}$$

其中 $L_i = H_{ii} * \omega_p$，表示单个权重的敏感性。H_{ii} 表示第 i 个对角线元素。可以看出，通过对角近似，使得通道的敏感性可以近似为通道内各个权重的敏感性之和。

此外，由于在Hessian矩阵的计算过程中，神经网络中的各个通道之间相互独立、互不影响，因此可以采用并行计算的方式来同步计算各个通道的敏感性，从而减少计算所需时间。

图1显示了把本文算法的示意图，其中根据其二阶敏感性对层进行剪枝。通过对权重按照单独的通道进行分组，并将相应的Hessian近似为对角算子，计算出每个通道的敏感性并根据其敏感性大小进行排序，对相对不敏感的通道进行结构化剪枝，从而实现神经网络的高效压缩。

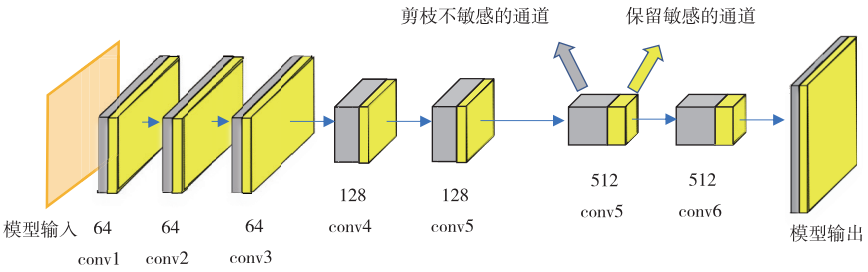

图 1　基于敏感性的剪枝算法示意图

Figure 1　Schematic diagram of sensitivity based pruning algorithm

3.3　算法流程

本文提出的策略通过从经过预训练的模型中剪枝不重要的通道来优化计算效率，同时减少精度损失。通过累加通道其中权重的敏感性，并根据通道敏感性排序，来确定每层通道的相对敏感性，然后根据剪枝率和通道敏感性的排序逐层剪枝不敏感的通道。在通道剪枝之后，对整个网络进行重训练，以此来补偿性能损失，恢复网络的精度。

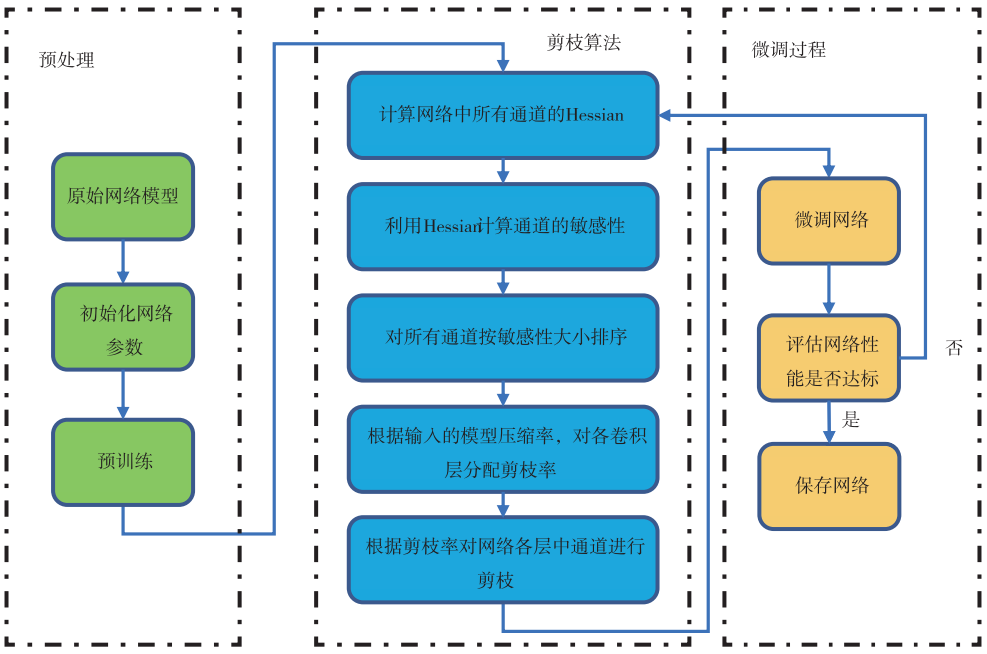

图 2　算法流程图

Figure 2　Algorithm flow chart

https://doi.org/10.35534/pc.0611006

如图2所示，基于敏感性的通道剪枝算法按照，采用预训练–剪枝–微调的策略。算法的具体流程如下所示：

步骤1：输入初始网络模型，并初始化网络模型参数。

步骤2：根据输入的训练集对网络模型进行预训练。

步骤3：利用上节中的算法计算模型卷积层中各个不同通道的Hessian矩阵。

步骤4：通过Hessian矩阵得到通道的敏感性大小。

步骤5：根据得到的敏感性大小对所有的通道进行重要性排序。

步骤6：根据输入的剪枝率确定所需剪枝的通道数量。

步骤7：通过通道剪枝获得紧凑的网络结构及其对应的参数。

步骤8：通过重训练微调网络获得最终的网络参数，并保存网络。

4　实验结果与分析

4.1　数据集及实验参数配置

本文分别在小型数据集CIFAR–10和大型数据集ImageNet上进行了实验。CIFAR–10数据集包括10个类别的60，000张32×32大小的彩色图像，其中50，000张训练图像和10，000张测试图像。ImageNet包含128万张训练图像和5万张验证图像，总共有1000个类别。

本文按照文献[20]中描述的设置对CIFAR–10数据集进行预训练。对于ImageNet数据集，直接使用Torchvision库中的预训练模型。本文还利用了文献[15]中提出的微调方法。对于CIFAR–10数据集，在剪枝后重新训练网络200次，学习率的设置从0.1开始，在分别在第60、120和160次时设置衰减率为0.1，以此获得最佳微调效果。对于ImageNet数据集，本文将剪枝后的网络微调训练了150次，基础学习率为0.1，衰减率为每30次衰减0.1。本文对所有模型使用随机梯度下降算法，批大小、权重衰减和动量分别设置为256、0.0001和0.9。同时，整个实验使用Pytorch 1.8进行测试验证。

4.2　实验结果及分析

在CIFAR–10数据集上剪枝VGG–16模型的结果如表1所示，并与其他文献中的先进算法如GAL[21]、HRank[18]、VarP[22]进行了比较。通过对比，本文始终实现了更优的压缩率和更低的精度损失。特别地，为了与HRank进行更好的对比，本文将剪枝后模型的精度损失设定为与HRank相同的0.53%，此时本文的FLOPs压缩率达到了54.1%，优于HRank的53.5%。此外，在相同的FLOPs压缩（约65.3%）的情况下，本文在准确率下降方面取得了更好的结果（1.46% vs 1.52%）。通过进行更为极端的剪枝，本文能够将FLOPs减少70.1%，而仅损失1.7%的精度。结果表明，本文方法可以高效压缩和加速结构简单的神经网络。

表 1　不同算法对VGG–16模型在CIFAR–10上的实验结果

Table 1　Experimental results of VGG-16 model on CIFAR-10 with different algorithms

不同算法	原始模型精度（%）	剪枝后精度（%）	精度损失（%）	FLOPs 压缩（%）
GAL	93.96	92.03	1.93	39.0
VarP	93.96	93.18	0.78	39.1
SSS	93.96	93.02	0.94	41.6
本文	93.70	93.69	0.01	42.9
GAL	93.96	90.73	3.23	45.2
HRank	93.96	93.43	0.53	53.5
本文	93.70	93.17	0.53	54.1
HRank	93.96	92.34	1.62	65.3
本文	93.70	92.24	1.46	65.8
本文	93.70	91.99	1.71	70.1

ResNet34模型的实验结果如表2所示。与GM[20]、SFP[15]、DMC[23]、NPPM[24]等方法对比，本文方法可以显著降低网络的计算成本，同时对网络性能的影响最小。例如，本文可以在减少48.9%的FLOPs的同时，模型精度性能只下降0.23%。与其他剪枝方法相比，本文方法在ResNet34上剪枝了更多的FLOPs，同时模型性能下降最少。

表 2　不同算法对ResNet34模型在ImageNet上的实验结果

Table 2　Experimental results of ResNet34 model on ImageNet with different algorithms

不同算法	原始模型精度（%）	剪枝后精度（%）	精度损失（%）	FLOPs 压缩（%）
GM	73.92	72.63	1.29	41.1
SFP	73.93	71.84	2.09	41.1
DMC	72.57	72.26	0.31	43.4
NPPM	73.30	73.01	0.29	44.0
本文	72.83	72.60	0.23	48.9

4.3 消融实验

本文进行了消融实验，来验证敏感性剪枝方法的有效性。实验结果如表3所示。为简洁起见，本文使用ResNet56在CIFAR-10上运行所有实验。在其他网络和数据集上也可以找到类似的观察结果。

作为对比，本文按照随机顺序而不考虑它们的敏感度来进行剪枝通道，在表3中被标记为随机顺序。很明显，随机通道剪枝方法在所有情况下都比本文所提出方法的精度下降更大，这表明合理和有序的剪枝指标是必不可少的，本文所提出的基于敏感度的通道选择标准是有效的。

表 3 消融实验

Table 3 Ablation experiment

不同算法	精度损失（%）	参数压缩（%）	FLOPs 压缩（%）
随机顺序	1.91	48.20	47.32
本文	0.00	50.65	47.5
随机顺序	2.35	55.79	58.63
本文	0.82	59.31	59.96
随机顺序	4.65	68.44	71.64
本文	2.12	74.35	72.23

5 结论

在许多情况下，现有的通道剪枝方法在使用高压缩率进行剪枝时，会导致相当大的精度损失。为了克服这个问题，本文提出了一种新的二阶敏感性结构化剪枝方法，该方法利用通道中权重的敏感性之和作为网络模型中剪枝通道的度量。本文的基本思路是确定每一层中应该被剪枝的通道数量，然后根据设定的剪枝率来剪枝不敏感的通道。通过使用不同的模型进行实验测试，证明了本文的方法在降低计算复杂度和压缩模型规模方面的可行性和有效性。本文还与许多现有的方法进行了比较，结果显示本文的方法始终能够以较少的FLOPs产生更佳的模型精度。这些结果表明，本文所提出的剪枝方法在很大程度上明显优于现有方法。

参考文献

[1] Krizhevsky A, Sutskever I, Hinton G E. ImageNet Classification with Deep Convolutional Neural Networks [C] // Advances in Neural Information Processing Systems. Curran Associates, Inc., 2012.

[2] Chowdhary K R. Natural Language Processing [M] //CHOWDHARY K R. Fundamentals of Artificial Intelligence. New Delhi: Springer India, 2020: 603-649.

[3] Deng L, Hinton G, Kingsbury B. New types of deep neural network learning for speech recognition and related applications: an overview [C]. IEEE International Conference on Acoustics, Speech and Signal Processing, 2013.

[4] Zhou S-C, Wang Y-Z, Wen H, et al. Balanced Quantization: An Effective and Efficient Approach to Quantized Neural Networks [J]. Journal of Computer Science and Technology, 2017, 32 (4): 667-682.

[5] Gabor M, Zdunek R. Compressing convolutional neural networks with hierarchical Tucker-2 decomposition [J]. Applied Soft Computing, 2023 (132): 109856.

[6] Wang H, Hu X, Zhang Q, et al. Structured Pruning for Efficient Convolutional Neural Networks via Incremental Regularization [J]. IEEE Journal of Selected Topics in Signal Processing, 2020, 14 (4): 775-788.

[7] Suzuki K, Horiba I, Sugie N. A Simple Neural Network Pruning Algorithm with Application to Filter Synthesis [J]. Neural Processing Letters, 2001, 13 (1): 43-53.

[8] Liu C, Wu H. Channel pruning based on mean gradient for accelerating Convolutional Neural Networks [J]. Signal Processing, 2019 (156): 84-91.

[9] Lecun Y, Denker J, Solla S. Optimal Brain Damage [C] // Advances in Neural Information Processing Systems. Morgan-Kaufmann, 1989: 598-605.

[10] Hassibi B, Stork D, Wolff G. Optimal Brain Surgeon: Extensions and performance comparisons [C]. Advances in Neural Information Processing Systems. Morgan-Kaufmann, 1993: 263-270.

[11] Dong X, Chen S, Pan S. Learning to Prune

https://doi.org/10.35534/pc.0611006

Deep Neural Networks via Layer-wise Optimal Brain Surgeon [C] // Advances in Neural Information Processing Systems. Curran Associates, Inc., 2017.

[12] Hanson S, Pratt L. Comparing Biases for Minimal Network Construction with Back-Propagation [C] // Advances in Neural Information Processing Systems. Morgan-Kaufmann, 1988: 177-185.

[13] Chen W, Wilson J, Tyree S, et al. Compressing Neural Networks with the Hashing Trick [C] //Proceedings of the 32nd International Conference on Machine Learning. Lille, France: PMLR, 2015: 2285-2294.

[14] Han S, Pool J, Tran J, et al. Learning both Weights and Connections for Efficient Neural Network [C] //Advances in Neural Information Processing Systems. Curran Associates, Inc., 2015: 1135-1143.

[15] He Y, Kang G, Dong X, et al. Soft Filter Pruning for Accelerating Deep Convolutional Neural Networks [C] //Proceedings of the 27th International Joint Conference on Artificial Intelligence. Stockholm, Sweden: AAAI Press, 2018: 2234-2240.

[16] Hu H, Peng R, Tai Y-W, et al. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures [J]. arXiv preprint arXiv: 1607. 03250, 2016.

[17] Liu Z, Li J, Shen Z, et al. Learning Efficient Convolutional Networks through Network Slimming [J]. IEEE, 2017.

[18] Lin M, Ji R, Wang Y, et al. HRank: Filter Pruning Using High-Rank Feature Map [C]. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, 2020: 1526-1535.

[19] Liu Z, Sun M, Zhou T, et al. Rethinking the Value of Network Pruning [C]. International Conference on Learning Representations, 2017.

[20] He Y, Liu P, Wang Z, et al. Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration [C] //2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, 2019: 4335-4344.

[21] Lin S, Ji R, Yan C, et al. Towards Optimal Structured CNN Pruning via Generative Adversarial Learning [C] //2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, 2019: 2785-2794.

[22] Zhao C, Ni B, Zhang J, et al. Variational Convolutional Neural Network Pruning [C]. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[23] Gao S, Huang F, Pei J, et al. Discrete Model Compression With Resource Constraint for Deep Neural Networks [C]. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[24] Gao S, Huang F, Cai W, et al. Network Pruning via Performance Maximization [C]. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, 2021: 9266-9276.

Channel Pruning of Convolutional Neural Network based on Sensitivity

Yang Chenbin

Jiangsu Police Institute, Nanjing

Abstract: Convolutional neural networks (CNNs) have a large number of redundant parameters from the convolutional layer to the fully connected layer. Model pruning is an optimization technique used to reduce the memory consumption and floating-point operations (FLOPs) of deep convolutional neural network models. By removing "unimportant" weights from the model, it reduces the number of parameters and computation, while ensuring that the accuracy of the model is not affected as much as possible, achieving the best balance between parameter quantity and model performance. However, for the case of higher pruning rate, current channel pruning methods often use the norm the channel as the criterion of pruning, and lack the corresponding theoretical support. It is difficult to achieve the ideal pruning effect, which will lead to a significant decline in the accuracy of the compressed model. To solve this problem, this paper proposes a new channel pruning method based on sensitivity measurement, which uses second-order sensitivity as a criterion to measure the importance of channels. Through theoretical derivation, the traditional sensitivity calculation is extended from weight to channel, and it is proved that the sensitivity of the entire channel can be quantified by the sum of the weight's sensitivity in the channel, and then the insensitive channel can be deleted to complete the channel pruning. Experiments on a variety of different CNNs architectures show that our paper can significantly improve the pruning rate while losing a small amount of accuracy. For example, with a slight loss of accuracy, FLOPs on the CIFAR-10 dataset were reduced by more than 60%. Additionally, on the ImageNet dataset, ResNet34-based pruning reduces FLOPs by 52.1% while losing only 0.23% accuracy.

Key words: Convolutional neural networks; Channel pruning; Sensitivity

https://doi.org/10.35534/pc.0611006