第 1 章　结构化剪枝综述

吴建鑫　王环宇　张永顺

计算机软件新技术国家重点实验室（南京大学），南京　210023

1　引言

卷积神经网络（Convolutional Neural Networks）特别是深度神经网络（Deep Neural Networks）在计算机视觉任务上发挥着愈发重要的作用。在一个神经网络模型中，通常包含卷积层、汇合层、全连接层、非线形层等基本结构，通过这些基本结构的堆叠，最终形成我们所常用的深度神经网络。早在 1998 年，LeCun 等人使用少数几个基本结构组成 5 层的 LeNet-5 网络[1]，并在 MNIST 数据集上得到了 98.9%的分类精度，但此时的深度神经网络还相对简单，并且只能用于简单的任务上；在 2012 年的 ImageNet 图像分类竞赛中，AlexNet[2]将深度提高到了 8 层，并且达到了远超传统方法的结果；此后，VGG 团队提出的 VGG-Net[3]进一步加深了网络，使网络最高达到了 19 层。虽然增加网络的深度能够带来性能的提升，但也不能无限制的增加网络深度，随着网络的加深，梯度消失会愈发严重，并且模型会变得愈发难以训练。因此在 2016 年，He 等人提出 ResNet[4]，在模型中加入残差结构，并一举将网络的深度提高到 152 层。至此，随着深度学习的研究逐步推进，神经网络可以变得更宽更深更复杂，与此同时带来更好的表示能力和性能表现。

当一些研究者将模型变得更大、更深时，另一些则考虑在保持模型精度的同时使模型变得更小、更快，其中一类重要的方法为模型压缩。模型压缩大致上可以分为四类：模型量化、模型剪枝、低秩近似和知识蒸馏。

通常来说我们用 32 位浮点数来保存模型，模型量化主要考虑用更小位数来保存模型参数，通常使用的有 16 位浮点数和 8 位整数，其参数量和计算量都会相应地随着存储位数而成倍降低；更有甚者，将模型量化成二值网络[5]，三元权重[6]或者同或网络[7]。例如，经过简单量化之后的 MobileNetV1[8]仅仅只有 4-5MB，能够轻松部署在各种移动平

台上。

模型剪枝[9,10,11,12]主要分为结构化剪枝和非结构化剪枝，非结构化剪枝去除不重要的神经元，相应地，被剪除的神经元和其他神经元之间的连接在计算时会被忽略。由于剪枝后的模型通常很稀疏，并且破坏了原有模型的结构，所以这类方法被称为非结构化剪枝。非结构化剪枝能极大降低模型的参数量和理论计算量，但是现有硬件架构的计算方式无法对其进行加速，所以在实际运行速度上得不到提升，需要设计特定的硬件才可能加速。与非结构化剪枝相对应的是结构化剪枝，结构化剪枝通常以滤波器或者整个网络层为基本单位进行剪枝。一个滤波器被剪枝，那么其前一个特征图和下一个特征图都会发生相应的变化，但是模型的结构却没有被破坏，仍然能够通过 GPU 或其他硬件来加速，因此这类方法被称之为结构化剪枝。

低秩近似[13,14,15]将一个较大的卷积运算或者全连接运算替换成多个低维的运算。常用的低秩近似方法有 CP 分解法[13]，Tucker 分解[14]和奇异值分解[15]。例如，一个$M \times N$的全连接操作若能近似分解为$M \times d$和$d \times N$（其中$d \ll M, N$）那么这一层全连接操作的计算量和参数量将被极大地缩减。

知识蒸馏（Knowledge Distillation）[16]通过使用一个足够冗余的教师模型，来将其知识 “传授”给紧凑的学生模型。在训练时同时使用教师模型的软标签和真实标记的硬标签来共同训练学生模型，从而能够使学生模型达到接近教师模型的性能，也因此能够降低达到目标精度所需的计算量和模型大小。

上述模型压缩方法能配合使用，一个模型经过结构化剪枝之后，由于其结构没有发生重要变化，所以能紧接着进行低秩近似以减少参数量和计算量，最后再通过参数量化进一步减少参数量并加速。近些年来，随着物联网的发展，企业将深度学习模型部署在嵌入式设备的需求在快速增长，而嵌入式设备计算能力有限，并且由于成本原因希望部署的模型尽可能地小。模型压缩的意义在于保证精度的同时尽可能减少计算量和参数量，因此对于嵌入式设备的部署有切实的价值。 模型压缩包含很多内容，这里我们将主要关注剪枝算法中的结构化剪枝。在本章的剩余部分，我们将首先介绍结构化剪枝的一些基本方式，然后介绍一些经典的和最新的结构化剪枝算法，最后对结构化剪枝的应用和未来发展进行总结和展望。

2 剪枝方式介绍

在结构化剪枝中，最基本的方式是滤波器剪枝（也称之为通道剪枝），本节将首先介绍滤波器剪枝的内容，并以此为基础介绍三种常见的模型剪枝方式。

FLOPs（Floating-point operations）是浮点计算量的简称[17]，通常使用 FLOPs 来表

示模型的计算复杂度。将一个输入通道数为C_{in}，输出通道数为C_{out}的卷积层简化记为$[C_{out}, C_{in}, K]$，表示这个卷积层里有C_{out}个$C_{in} \times K \times K$的卷积单元，分别和输入特征进行卷积操作。这里我们将一次乘加(multiply and accumulate)算作两个浮点运算，所以对于输入为$H \times W \times C_{in}$的特征，经过这一层卷积的浮点运算量为FLOPs $= 2HWC_{in}K^2C_{out} + HWC_{out}$，其中$HWC_{out}$表示偏置加法带来的计算量。参数量即为所含参数的数量，一个卷积层的参数量Params $= K^2C_{in}C_{out} + C_{out}$，等号右边的两项分别表示卷积核和偏置的参数量。我们使用一个两层的卷积网络来展示滤波器剪枝的细节，为简化模型，我们均使用卷积核为 3、步长为 1 的普通卷积，并且不考虑卷积中的偏置操作。在我们的简化模型中，每层卷积的浮点计算量为$18HWC_{in}C_{out}$，参数量为$9C_{in}C_{out}$。

如图 1 所示，我们以这个简化的两层卷积网络为例来对剪枝过程进行分析。假定第一层卷积的输入和输出通道数分别为C_1和C_2，第二层卷积的输入和输出通道数分别为C_2和C_3。那么一个高度为H、宽度为W、通道数为C_1的输入特征$F_1 \in \mathbb{R}^{H\times W\times C_1}$ 经过第一层卷积将得到中间特征$F_2 \in \mathbb{R}^{H\times W\times C_2}$，然后再经过第二层卷积操作得到输出特征$F_3 \in \mathbb{R}^{H\times W\times C_3}$。这个双层卷积网络的参数量为$9C_2(C_1 + C_3)$，计算量为$18HWC_2(C_1 + C_3)$。我们对这个简单的双层卷积模型进行剪枝，将中间层的特征F_2由C_2个通道减少到C_2'个通道，那么相应地，第一个卷积层中产生$C_2 - C_2'$个对应通道特征的卷积单元变得不再需要，所以新的卷积层可以表示为$[C_2', C_1, 3]$；第二个卷积层中和相应通道特征进行卷积的参数也变得不再需要，所以新的卷积层可以表示为 $[C_3, C_2', 3]$。因此，第一个卷积的输出通道数量和第二个卷积的输入通道数量都由C_2相应减少到C_2'，至此一个通道剪枝过程结束。

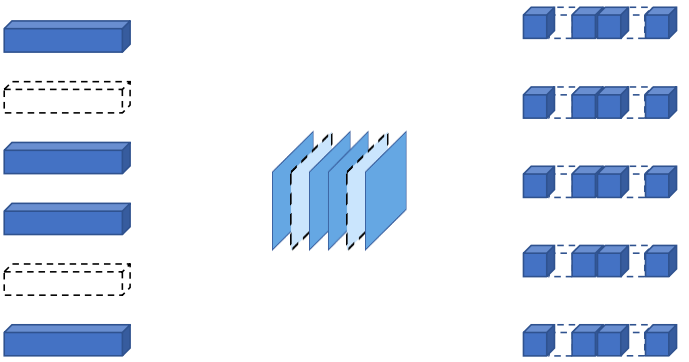


图 1　滤波器剪枝

剪枝之后的双层卷积网络，其参数量变为$9C_2'(C_1+C_3)$，计算量变为$18HWC_2'(C_1+C_3)$。当C_2'减小到C_2的一半时，这个简单双层网络的参数量和计算量都将减半。

2.2 滤波器级别剪枝

在上述的例子中，相邻的两个卷积层紧紧联系在一起，第一个卷积层输出维度的变化将引起第二个卷积层的输入维度发生相应的变化。这样的卷积操作可以由第一个卷积层和第二个卷积层的组合，一直进行到倒数第二个卷积层和最后一个卷积层的组合。

由于卷积层是用于视觉任务的神经网络的基础，所以对于几乎所有的网络结构，都可以使用滤波器级别剪枝（filter-level pruning）来减少参数量和计算量。大多数的剪枝算法也是基于滤波器级别剪枝来精简模型。

滤波器级别剪枝的核心在于减少一个中间特征的数量，其前一个和后一个卷积层需要发生相应的变化。

3.3 阶段级别的剪枝

现在所使用的卷积神经网络大多都参考了 ResNet[4]中的残差结构。残差结构的具体形式不唯一，在 ResNet 中的残差结构由三层卷积和一个跨层连接构成；在 MobileNetV2[18]中，采用逐点卷积和逐层卷积来减少参数量和计算量，也有一个跨层连接。除此之外也有很多其他的形式，但是不管具体的构建形式如何，残差结构都可以表示为：

$$y=f(x)+x。$$

即残差结构的输出由两部分逐点相加得到：一部分是残差结构内最后一层卷积的输出，另一部分是残差结构的输入。由于需要进行逐点操作，所以这两部分的张量维度必须一致。当一个残差结构的输出直接作为下一个残差结构的输入时，等于说这两个残差结构的输出通道数是相等的，那么这两个残差结构紧密的联系在一起。通常会有其他操作夹在两个残差结构块之间（比如普通的卷积层，汇合层等），这样两个残差结构块之间的联系便被打破了。在残差结构块的联系没有被打破之前，多个残差结构块将紧密联系在一起，这些残差结构块的输出通道是对应起来的。我们将这样相互影响的多个残差结构块看成一个“阶段”（stage），常用的 ResNet-50 有四个阶段，这四个阶段分别有 3，4，6，3 个残差结构块。

滤波器级别的剪枝只能作用于残差结构块内部的卷积层，CURL [19]中指出只进行滤波器级别的剪枝会导致模型形成一个沙漏状、两头宽中间窄的结构，这样的结构会影响

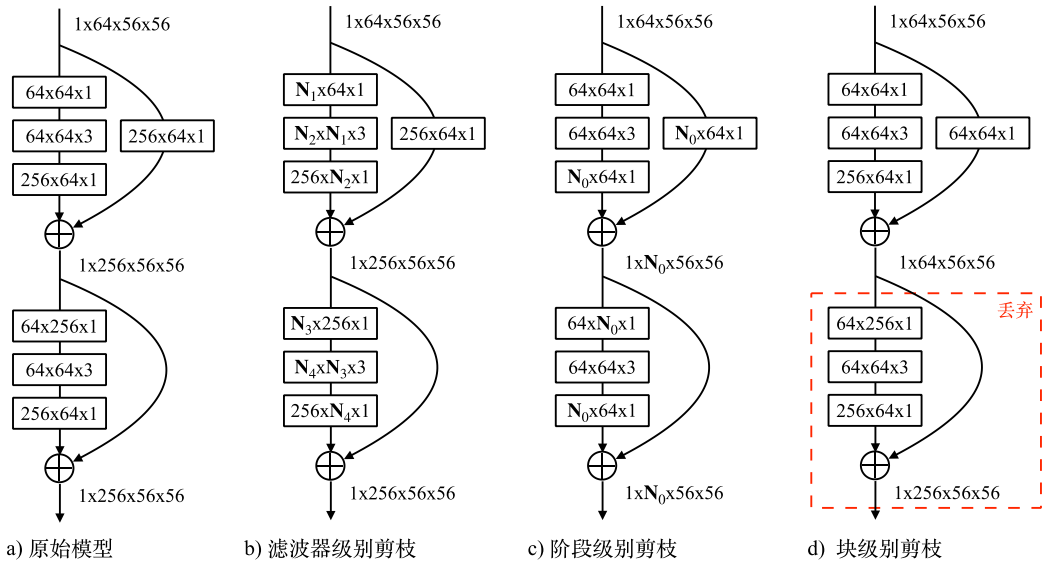


图 2　剪枝方法的对比

的参数量和计算量。在这种情况下，阶段级别的剪枝能弥补滤波器级别剪枝的不足。正如上文所介绍的，一个阶段中的残差结构块是紧密联系在一起的，如图 2 所示，当一个阶段的输出特征发生变化时（一些特征被抛弃），其对应的每个残差结构的输入特征和输出特征都要发生相应的变化，所以整个阶段中，每个残差结构的第一个卷积层的输入通道数，以及最后一个卷积层的输出通道数都要发生相同的变化。由于这样的影响只限定在当前的阶段，不会影响之前和之后的阶段，因此我们称这个剪枝过程为阶段级别的剪枝（stage-level pruning）。

阶段级别的剪枝加上滤波器级别的剪枝能够使网络的形状更均匀，而避免出现沙漏状的网络结构。此外，阶段级别的剪枝能够剪除更多的网络参数，这给网络进一步压缩提供了支持。

3.3　块级别的剪枝

当我们想得到比较小的剪枝模型（计算量是原模型的 10%甚至更少）时，我们需要大量缩减各层特征的通道数，这样模型的宽度会大幅减少，而深度不会发生改变，最终将形成一个特别窄但是仍然比较深的模型。此时，模型的深度会限制模型速度，并且一味减少模型的宽度可能会导致在某些层出现极端窄的情况。此时块级别的剪枝（block-

level pruning）能帮助解决这些困难。

块级别的剪枝是直接丢弃某些残差结构块，由于残差结构的数学形式可以表达为 $y = f(x) + x$，丢弃残差结构后等于这一层变为 $y = x$。以 ResNet 为例，其每个阶段的第一个残差结构通常会降低特征图的分辨率并提高特征图的通道数，所以除了每个阶段第一个残差结构块之外，其他的残差结构块都可以直接地被丢弃并且不影响整个网络的运行。这样的好处在于能降低网络的深度，从而获得相同大小的剪枝模型时，使用块级别剪枝的方法不至于过多减少每一层的通道数。加上块级别剪枝之后，一个 ResNet-50 模型能够很容易地剪枝得到计算量比例为 5%甚至更少的子模型。

3.4 小结

本节主要介绍了三种用于深度模型结构化剪枝的方法。三种剪枝方法的对比在图 2 中显示。这三种方法可以一起使用，从而使剪枝之后的模型结构更正常，并且得到更小的剪枝模型。

本节主要以 ResNet 模型为例进行分析，其他的一些主流模型的剪枝方法可以参考 ResNet 进行相应的处理，在此不作赘述。

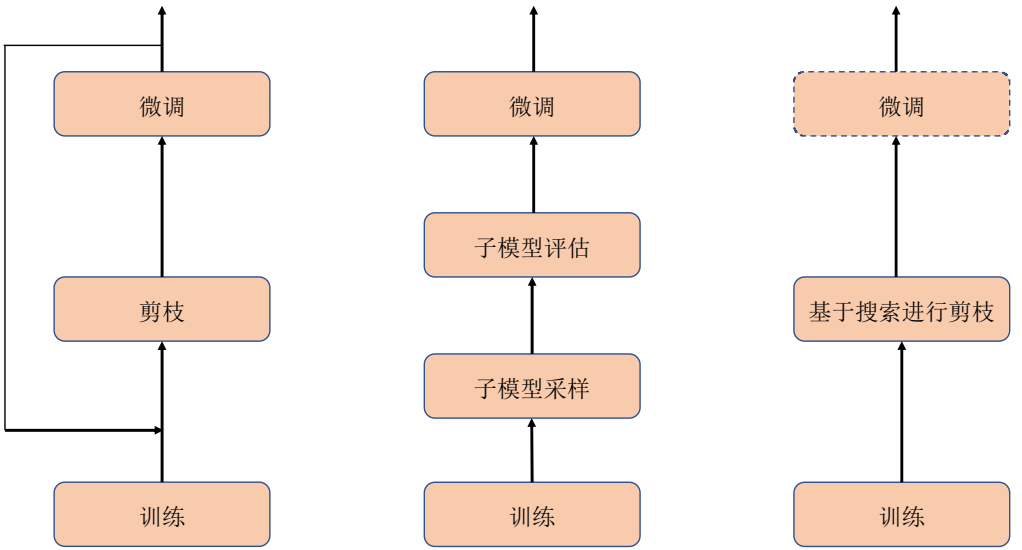
3 剪枝算法

3. 1 剪枝算法的运行流程及评价方式

剪枝算法的目的在于减少原模型的参数以及计算量，同时尽可能保证得到的子模型表达能力相比原模型来说损失较少。本节介绍当前主要的几种剪枝流程以及其典型的代表方法，同时在剪枝实验上，由于本文只针对基于卷积神经网络的图像分类领域模型剪枝，所以在实验方面本文只介绍此领域主要的评价框架，包括数据集、常用模型以及评估准则。

3. 1. 1 剪枝的具体流程

剪枝是减少模型参数量和计算量的经典方法。随着深度学习的兴起以及卷积神经网络在图像分类领域的大量应用，各种各样的剪枝方法也不断涌现出来。虽然剪枝的方法种类很多，但是其核心思想还是对神经网络的结构进行剪枝，目前剪枝算法的总体流程大同小异，可以归结为三类：标准剪枝、基于子模型采样的剪枝、以及基于搜索的剪枝，如图 3 所示。



a）标准剪枝　　b）基于子模型采样的剪枝　　c）基于搜索的剪枝

图 3　剪枝流程

标准剪枝主要包含三个部分：训练、剪枝、以及微调，如图 3a) 所示。对于标准剪枝的流程详细说明如下：

1) 训练。在剪枝流程中，训练部分只需进行一次即可，训练的目的是为剪枝算法获取在特定任务上训练好的原始模型。
2) 剪枝。剪枝最重要的环节是对网络结构进行重要性评估，而这一重要性评估的环节也是各种剪枝算法最主要的区别之一。评估的模型结构主要包含滤波器、块等结构。对网络结构的重要性评估可以分为网络参数驱动的评估以及数据驱动的评估两类方法。

- 基于网络参数驱动的方法利用模型本身的参数信息来衡量模型结构的重要性，如参数的l_1 正则化或者l_2 正则化[9,10,11,12]，该类方法评估过程不依赖输入数据。
- 基于数据驱动的方法通过利用训练数据来对网络结构的重要性作出评估，如通过统计滤波器输出结果经过激活层之后 0 值的个数来评价该滤波器的重要性[20]。

在第 3.2 节中，我们会对参数驱动以及数据驱动两类评估方法做更详细的介绍。在网络结构评估完成之后，只需根据网络结构的评估结果按照第 2 节“剪枝方式介绍”中的模型剪枝方式，修剪掉不重要的网络结构即可。

3) 微调。微调是恢复被剪枝操作影响的模型表达能力的必要步骤。结构化模型剪枝会对原始模型结构进行调整，因此剪枝后的模型参数虽然保留了原始的模型参数，但是由于模型结构的改变，剪枝后模型的表达能力会受到一定程度的影响。微调过程通过将剪枝后的子模型在训练集进行微调训练，能够恢复子模型的表达能力。

4) 再剪枝。再剪枝过程将微调之后的子模型再送到剪枝模块中，再次进行模型结构评估和剪枝过程。通过再剪枝过程，使得每次剪枝都在性能更优的模型上面进行，不断阶段性的优化剪枝模型，直到模型能够满足剪枝目标需求。

标准剪枝流程是目前剪枝算法的主要流程[10,20,21,22,23,24,25,26]，同时在标准剪枝的基础上，有些相关工作对标准剪枝过程进行改进[17,19]。[17]将剪枝过程集成到模型微调中，不再区分微调和剪枝两部分；提出一个新的可训练的网络层用于剪枝过程，该网络层生成二进制码，二进制码中的 0 值对应的网络结构将被剪掉。[19]通过计算原始模型和去掉对应网络结构的子模型之间的 KL 散度来衡量每一个网络结构的重要性，这种计算方式使得网络结构评估不局限在局部特征或者参数中，而是利用全局特征，使得评估结果更为精确，因此其无需再剪枝过程便能达到很好的剪枝效果。

除标准剪枝之外，基于子模型采样的剪枝[27]最近也表现出很好的剪枝效果。基于子模型采样的剪枝流程如图 3b) 所示，得到训练好的模型之后，进行子模型采样过程。一次子模型采样过程为：1）对训练好的原模型中可修剪的网络结构按照剪枝目标进行采样，采样可以是随机的，也可以按照网络结构的重要性进行概率采样；2）对采样出的网络结构进行修剪，得到采样子模型。子模型采样过程通常进行n次，得到n个子模型（$n \geq 1$），之后对每一个子模型进行性能评估。子模型评估结束之后，选取最优的子模型进行微调以得倒最后的剪枝模型。

基于搜索的剪枝主要依靠强化学习或者神经网络结构搜索相关理论，其主要流程如

图 3c) 所示。给定剪枝目标之后，基于搜索的剪枝在网络结构中搜索较优的子结构，这个搜索过程往往伴随着网络参数的学习过程，因此一些基于搜索的剪枝算法在剪枝结束后无需再进行微调。在第 3.3 节“基于搜索的剪枝算法中”，我们会对其进一步做详细的介绍和探讨。

3. 1. 2 评价剪枝效果使用的数据集、模型和评估准则

图像分类领域的剪枝实验常用的数据集包括 CIFAR-10[28]、ILSVRC-2012[29]。

- CIFAR-10 数据集包括 10 个类别，其训练集和验证集分别包含 50000 张和 10000 张 RGB 彩色图像。CIFAR-10 数据集类别平衡，即每个类别包含 5000 张训练图像和 1000 张验证图像，每张图像的分辨率为32×32。
- ILSVRC-2012 是 ImageNet 2012 竞赛数据集，其总类别数量为 1000。ILSVRC-2012 的训练集包含 1281167 张训练图像，每个类别的训练图像数量从 732 到 1300 不等。同时其验证集包含 50000 张图像，验证集类别平衡，每一个类别包含 50 张验证图像。

在图像分类领域卷积神经网络的众多种类中，VGG-Net [3]和 ResNet[4]应用广泛。由于移动端设备部署的需求越来越强烈，MobileNet 系列[8][18]也逐渐成为剪枝实验的主要网络结构之一。

在实际实验中，VGG-Net 通常采用 VGG-16（16 层网络结构的 VGG-Net）作为实验网络结构, 而 MobileNet 通常采用 MobileNet-V1[8]、MobileNet-V2[18]作为 MobileNet 在剪枝实验中的网络结构。同时对于 ResNet，CIFAR-10 实验上通常采用 ResNet-52（针对 CIFAR-10 数据集设计的 52 层网络结构的 ResNet），而 ILSVRC-2012 数据集上通常采用 ResNet-50（50 层网络结构的 ResNet）。

我们通常从三个方面评价神经网络剪枝算法的优劣：准确率及准确率变化、模型大小变化、以及网络前向时间变化。在实际实验中，这三种指标都需要统计，来综合体现剪枝算法的效果。

- 准确率及准确率变化。在测试剪枝算法的实验结果时，需要在数据集的验证集上统计剪枝子模型的 Top-1 验证集准确率以及 Top-5 验证集准确率作为剪枝子网络的性能指标。同时，需要记录剪枝子模型准确率相对于原始模型的准确率变化（通常是负数），准确率变化大表明该子模型相比原模型准确率下降明显。
- 模型大小变化。剪枝实验的一个重要目的是减少原始模型的参数量，因此需要统计剪枝子模型的模型参数量相比于原始模型的变化量。该模型参数变化量作为模型大小变化的指标，变化量越大表示模型大小压缩的越多。

- 网络前向时间变化。对神经网络进行剪枝的最终目的在于保证子模型准确率的前提下，减少子模型的部署需求以及其运行时间，而运行时间仅靠模型大小、甚至 FLOPs 大小来衡量都是不准确的，因为有些模型参数量、FLOPs 虽然小，但是其结构不适应于硬件架构，所以其实际运行时间可能并未提升。模型前向时间变化可以由模型在硬件上的前向时间来体现，FLOPs 可供参考。

3.2 剪枝重要性评估准则

剪枝过程通过定义重要性指标来剪除相对不重要的参数。剪枝的指标往往是启发式的，根据评估指标时所利用的信息，基于重要性的模型剪枝算法评估准则可分为数据驱动和参数驱动两类。数据驱动和参数驱动的评估准则部分方法可见表 1。

表 1 结构化剪枝中的网络结构重要性评估准则

	具体方法	评价准则简介
参数驱动	[10,11]	l_1正则化
	[30]	l_2正则化
	[3,11]	批归一化层[39]中的γ参数
	[24]	几何中位数
	[32]	滤波器的谱聚类
数据驱动	[20]	激活输出值中 0 的激活值数量
	[26]	特征层的秩分解
	[33]	对特征图进行子空间聚类
	[23,34]	网络特征重建误差
	[35]	利用主成分分析来确定滤波器重要性
	[36]	网络结构重要值
	[37]	根据网络梯度进行剪枝
	[22]	利用剪枝前后网络特征熵值
	[19]	子模型和原模型之间的特征 KL 散度
	[27]	子模型进行 AdaptiveBN [27]操作后的准确率
	[38]	为网络每一层通过训练选择不同的评价准则

3.2.1 参数驱动的评估准则

基于参数驱动的评估准则，其网络结构评估无需依赖输入数据，最常用的方式为l_1正则化和l_2正则化。

[10]中对于每一个卷积层的滤波器，计算其权重绝对值之和（l_1正则化）作为其重要性分数，之后根据每一个滤波器的重要性分数排序，修剪掉重要性较低的滤波器。Network slimming[11]将l_1正则化加在损失函数中，用以约束批归一化层（batch normalization）[39]的参数γ稀疏程度，之后通过参数γ来评估滤波器的重要性，选取不重要的滤波器进行剪枝。SFP [30]中利用滤波器权重的l_2正则化来衡量每一个滤波器的重要性，在迭代剪枝的过程中将l_2正则化值较小的滤波器权重赋值为 0，网络剪枝结束时将网络中权重为 0 的滤波器修剪掉来获得最终的剪枝模型。[31]从变分贝叶斯方法的角度，说明批归一化层中参数γ可作为滤波器的评估指标，该方法也将参数γ加入损失函数中予以更新，最后修剪掉较小γ值对应的滤波器。

FPGM[24]从几何中位数的角度来进行滤波器重要性评估。FPGM 针对每一个卷积层包含的滤波器，先计算其几何中位数，认为如果某个滤波器接近几何中位数，可以认为这个滤波器的信息跟其他滤波器重合，于是可以去掉这个滤波器而不对网络产生大的影响。SCSP[32]利用谱聚类的思想，先对各层滤波器进行谱聚类，聚类之后根据滤波器的贡献进行排序，贡献小的滤波器会被剪枝。

基于参数驱动的评估准则部分代表方法总结在表 1 中，由于其评估过程不需要输入数据，只依赖模型本身的参数，因此其对计算资源要求较少。但基于参数驱动的方法通常需要预先设定一个参数剪枝阈值来决定要修剪掉的网络结构，这个阈值通常会根据网络结构变化而变化，需要按照实际任务进行调整，给剪枝工作带来了一定的时间损耗。

3. 2. 2 数据驱动的评估准则

基于数据驱动的评估准则进行网络结构重要性评估时需要利用输入数据，通常在特征层面、梯度层面、网络输出结果等方面进行分析。

APoZ[20]为数据驱动的经典方法之一，其将滤波器经过激活层后输出的特征层当中数值为 0 所占百分比作为该滤波器重要性的评估依据，值为 0 所占百分比越多则该滤波器重要性越低。HRank[26]发现单个滤波器生成的特征层的平均秩总是变化不大的，同时其证明了秩越低的特征层对精度的贡献越小，因此 HRank 通过修剪掉这些生成低秩特征层的滤波器达到剪枝目的。[33]通过实验发现特征层之间具有线性关系，因此利用子空间聚类的思想对滤波器产生的特征层进行聚类来消除滤波器的冗余，达到剪枝目的。

除了对特征层进行直接利用之外，[23,34]通过特征重建的思想来达到网络剪枝目的。ThiNet[23]根据下一层的特征输出来修剪当前层的滤波器，其主要思想是通过重构下一层的输入特征，即用下一层的输入的子集代替原来的输入得到尽可能类似原来的输出，这样子集以外的输入就可以去掉，同时其对应的前一层的滤波器也就可以修剪掉。[34]中也

采用了类似的特征层重建思想，通过迭代算法进行逐层剪枝，将网络每一层的特征重建问题转换为 LASSO 回归和最小二乘法重建误差，通过特征重建过程来修剪掉对特征层重建贡献较小的滤波器。PFA[35]将主成分分析（PCA）用于特征分析中，对于每一层滤波器输出的特征层，先将特征层进行池化输出为向量，而后对各个特征向量进行 PCA 分析，去掉信息含量少的特征层对应的滤波器来达到剪枝目的。

最近的一些工作[19,22,27,36,37]认为只单独考虑深度网络某一层或相邻两层滤波器的重要性并裁剪掉看似不重要的滤波器，可能会对后续层甚至更深层的响应输出产生影响，而且误差会逐层累积。因此需要从网络整体考虑网络结构的重要性。

SNIP[36]在全连接层前加入 Inf-FS 滤波，利用矩阵幂级数的性质有效地计算各个特征相对于所有其他特征的重要性，而后将重要性分数逐层反推，得到整个网络所有层的重要性评估结果，最后根据评估结果对原有的模型进行剪枝。[37]将滤波器的剪枝过程看作是一个优化问题，将代价函数反向传播以泰勒展开作为评估准则来决定滤波器的重要性，目的是为了使得剪枝之后的子模型在训练集上的代价值和原始模型在训练集上的代价值接近，对应在梯度层面上即为滤波器的重要性可由滤波器内的每一个权重的梯度表示。[22]认为滤波器产生的特征之间熵越大则该滤波器的重要性越高，对于每一个卷积层产生的特征，通过全局平均池化转换为特征向量，在训练子集上，对于该卷积层中每一个滤波器对应的特征向量集合，计算其熵作为对应滤波器的重要性指标。

[19,27]在模型层面上对网络结构重要性进行评估。CURL[19]对于网络中每一个滤波器，计算去掉该滤波器之后的子网络最后一层输出的池化特征和原始网络最后一层输出的池化特征之间的 KL 散度，文中认为该滤波器越重要，则去掉该滤波器之后的网络输出特征和原始特征之间相差应该越大，即 KL 散度越大，这种计算方式利用全局特征进行重要性评估。EagleEye[27]直接将剪枝过程转化为对可剪枝的网络结构采样的过程，直接评估每一个采样出来的子网络的重要性来完成剪枝目标，而不直接评估每一个滤波器的重要性。[27]对原始网络中可剪枝的滤波器进行采样，将采样出的子网络除批归一化层之外的参数固定住之后经过训练子集调节批归一化层参数 （AdaptiveBN），AdaptiveBN 之后的子网络在训练子集上的 Top-1 准确率作为该子网络的重要性，文中通过实验展示了子网络经过 AdaptiveBN 之后的准确率和该子网络经过微调之后的模型准确率之间的正相关关系，但由于其采样过程是随机的，所以通常需要针对一个剪枝目标采样上千次子模型，这种子模型采样及评估过程计算量极大。

以上基于数据驱动的剪枝方法中对网络中的每一层都应用了相同的评价准则，LFPC[38]认为不同的卷积层中特征表达意义不同，应该为其设置不同的评价准则来适应卷积层的多样性。LFPC 设计可以学习的评价准则组合，每一层对应一组评价准则组合系

数，该系数为可学习的参数。

和基于参数驱动的评估准则相比，基于数据驱动的评估准则评估方式不限于网络本身的参数，虽然需要的计算资源和评估时间增加，但是其可利用的空间由于输入数据的参与而变得更多。同时，针对不同的场景输入数据能够使得评估结果符合该应用场景下的数据特点，避免基于参数驱动的评估方法只局限于网络本身参数的缺点。通过表 1 中总结的代表方法可以看出，基于数据驱动的评估准则方法受剪枝领域欢迎的程度高于基于参数驱动的方法，近几年提出的评估准则多为数据驱动的方法。

3.3 基于搜索的剪枝方法

神经网络结构搜索（Neural Architecture Search）近些年来在模型结构设计上得到了广泛的关注和研究。在传统的模式下，研究者基于经验手工设计神经网络的结构，这不仅需要巨大的人力成本，同时对于一些普通的研究者来说也是一个困难的任务。Zoph 和 Le[40]首先提出使用基于强化学习的神经网络结构搜索来代替人工设计，但搜索过程需要耗费巨大的计算资源。随着 DARTS[41]的提出，基于梯度的搜索方法凭借其速度优势逐渐成为主流。此时，开始有研究者将结构化剪枝建模成一个基于原型网络的搜索过程，基于原型网络搜索得到的子模型就是剪枝后的小模型。本节我们将简要介绍一些基于搜索的剪枝方法。

AMC[42]早在 2018 年就使用强化学习方法来搜索剪枝结构，在 AMC 中，将每一层的输入特征维度、卷积核大小、步长等信息编码成一个状态，使用 DDPG[43]智能体来预测行动，使用分类错误和 FLOPs 作为奖赏。

ABCPruner[44]使用人工蜂群算法来寻找最优的剪枝结构。和常规的剪枝方法不同，ABCPruner 寻找每层的通道数量，而不是去选择相对重要的通道。具体来说，每层所对应的通道数量组成一个可行解，所有可行解构成目标种群，直接使用子模型在数据集上的表现作为其适应度，使用人工蜂群算法来进行搜索。在 ILSVRC-2012 数据集上，ABCPruner 能够在减少 62.87% FLOPs 和 60%参数量的同时，使 ResNet-152 的精度保持不变。

MetaPruning[45]使用元学习来自动进行通道剪枝。MetaPruning 首先训练一个元网络（meta network），然后使用演化过程来搜索表现好的剪枝模型。MetaPruning 为每层搜索通道的数量，而不是具体的每一个通道，能极大缩小搜索空间；在减少 ResNet-50 模型 25% FLOPs 时，在 ILSVRC-2012 上只比原模型降低 0.4%的 Top-1 准确率。

NetworkAdjustment[46]将模型准确率当成是关于计算量 FLOPs 的一个方程，由此来计算出每层的 FLOPs 利用率，最终根据这个利用率来自动调整每层的通道数。

表 2 不同剪枝算法在 ResNet−50 上的性能表现

	具体方法	模型 FLOPs	剪枝百分比 (%)	子模型 Top1 准确率 (%)	准确率变化 (%)
基于重要性的剪枝方法	SFP[30]	8.17	0	76.15	+0.00
		4.75	42	74.61	-1.54
	FPGM[24]	8.17	0	76.15	+0.00
		4.74	42	75.59	-0.56
		3.80	53	74.83	-1.32
	HRank[26]	8.17	0	76.15	+0.00
		4.60	44	74.98	-0.17
		3.10	62	71.98	-4.17
		1.96	76	69.10	-7.05
	ThiNet[23]	7.72	0	72.88	+0.00
		4.88	37	72.04	-0.84
		3.41	56	71.01	-1.87
		2.20	72	68.42	-4.46
	Entropy[22]	7.72	0	72.88	+0.00
		7.16	7	73.56	+0.68
		6.38	17	72.89	+0.01
		5.04	35	70.84	-2.04
	CURL[19]	8.17	0	76.15	+0.00
		2.22	73	73.39	-2.76
	EagleEye[27]	8.17	0	76.60	+0.00
		6.00	27	77.10	+0.50
		4.00	51	76.40	-0.20
		2.00	76	74.20	-2.40
	LFPC[38]	8.17	0	76.15	+0.00
		3.20	61	74.46	-1.69
基于搜索的剪枝方法	ABCPruner[44]	8.17	0	76.01	+0.00
		5.12	37	74.84	-1.17
		2.60	68	72.58	-3.42
		1.88	77	70.29	-5.72
	MetaPruning[45]	8.17	0	76.60	+0.00
		6.00	27	76.20	-0.40
		4.00	51	75.40	-1.20
		2.00	76	73.40	-3.20
	TAS[47]	8.17	0	77.46	+0.00
		2.31	72	76.20	-1.26
	AutoSlim[48]	8.17	0	76.15	+0.00
		6.00	27	76.10	-0.15
		4.00	51	75.60	-0.55
		2.00	76	74.00	-2.15
	DMCP[25]	8.17	0	76.60	+0.00
		5.60	31	77.00	+0.40
		4.40	46	76.20	-0.40
		2.20	73	74.40	-2.20

基于演化算法的搜索虽然对搜索过程进行了优化，但仍然有巨大的搜索时间花销。DMCP[25]将剪枝建模成一个可微分的马尔可夫过程，然后直接对网络进行梯度下降进行优化。每一层通道的剪枝构造成一个单独的马尔可夫过程，通道数量的增加和减少对应着马尔可夫状态的转移。在 ILSVRC-2012 数据集上，DMCP 能减少 ResNet-50 接近 46%FLOPs 的同时，得到精度降低 0.4%的剪枝模型。

我们使用表 2 来展示不同剪枝算法的性能表现。为了方便比较，我们使用 ResNet-50 作为基础网络，展示不同算法在 ILSVRC-2012 数据集上的剪枝表现。由于不同算法使用的深度学习框架不同，所以模型的基准结果会不同，我们比较其相对基准结果的变化。

3.4 小结

本节首先介绍了模型剪枝的基本流程，并且介绍了评估剪枝算法所常用的数据集，模型和一些评估准则。由于剪枝算法基本上使用相同的数据集和模型进行实验说明验证，所以不同剪枝算法之间的比较十分方便和直接。

本节还介绍了一些常见的基于重要性的结构化剪枝算法，并且按照参数驱动和数据驱动对其分别进行介绍。这类方法认为不同的通道有着不同的重要性，其核心区别在于重要性的定义方式不同。

此外，本节还介绍了一些基于搜索的结构化剪枝算法。这些方法直接对各层通道数量进行搜索。总结起来，无论是哪种剪枝算法，剪枝之后的模型都可以看作是原模型的一个子模型。这些子模型从原模型中继承部分参数，再辅以一定的微调训练，能达到和原模型相当，甚至超过原模型的分类结果，与此同时用到更少的参数量和计算量。

4 讨论与展望

现在的结构化剪枝算法主要在图像分类任务上进行研究，但由于神经网络结构的通用性，在分类任务上证明了有效性的剪枝算法能够方便地移植到其他视觉任务上。Ghosh[49]等人在 YOLOv3 模型上运用剪枝算法，进一步提升模型在目标检测任务上的运行速度；Hou 和 Kung[50]在图像超分辨率重建任务上使用剪枝算法得到更小的模型。也有一些研究者在特定任务上进行优化，Luo 和 Wu[19]在细粒度图像分类和小数据集上优化剪枝算法，比起直接运用大规模图像分类任务上的算法，优化过后的剪枝算法能够进一步带来精度的提升。

随着物联网的兴起和智能手机的发展，越来越多的移动设备搭载深度学习模型来提供更智能的服务。而这些移动设备的计算能力相对较低、存储空间相对不足，轻量级

的模型变得愈发重要。结构化剪枝是一个重要并且有效的精简模型的方法，通常会和参数量化以及低秩近似等其他方法共同使用来最大化压缩模型。非结构化剪枝破坏了模型原有的结构，虽然参数量和理论计算量都极大地降低了，但实际运行速度却并不乐观。和非结构化剪枝不同的是，结构化剪枝算法能在通用平台上运行，并且由于保持了原模型的结构，能在剪枝的基础上进行低秩近似，这是非结构化剪枝所无法做到的。

然而结构化剪枝也存在一些问题，通常需要经过评估-选择-微调这三个步骤才能得到一个精简的模型，更有甚者会逐层进行选择和微调，获取一个精简模型需要花费极长的时间；此外，绝大多数的研究都集中在图像分类的研究上，在目标检测、语义分割等更难的视觉任务上，成果还相对欠缺。

参考文献

[1] Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems 25 (NIPS), pages 1097–1105, 2012.

[3] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In The International Conference on Learning Representations (ICLR), pages 1–14, 2015.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016.

[5] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In Advances in Neural Information Processing Systems 29 (NIPS), pages 4107–4115, 2016.

[6] Fengfu Li, Bo Zhang, and Bin Liu. Ternary weight networks. arXiv preprint arXiv:1605.04711, 2016.

[7] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. XNOR-Net: ImageNet classification using binary convolutional neural networks. In The European Conference on Computer Vision (ECCV), volume 9908 of LNCS, pages 525–542. Springer, 2016.

[8] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.

[9] Vadim Lebedev and Victor Lempitsky. Fast convnets using groupwise brain damage. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2554–2564, 2016.

[10] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In The International Conference on Learning Representations (ICLR), pages 1–13, 2017.

[11] Zhuang Liu, JianguoLi, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In Proceedings of the IEEE International Conference on Computer Vision (CVPR), pages 2736–2744, 2017.

[12] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In Advances in Neural Information Processing Systems 29 (NIPS), pages 2074–2082, 2016.

[13] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan Oseledets, and Victor Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. In The International Conference on Learning Representations (ICLR), pages 1–11, 2015.

[14] Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. In The International Conference on Learning Representations (ICLR), pages 1–16, 2016.

[15] Xiangyu. Zhang, Jianhua Zou, Kaiming He, and Jian Sun. Accelerating very deep convolutional networks for classification and detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 38(10):1943–1955, 2016.

[16] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In NIPS Deep Learning and Representation Learning Workshop, 2015.

[17] Jian-Hao Luo and Jianxin Wu. AutoPruner: An end-to-end trainable filter pruning method for efficient deep model inference. Pattern Recognition (PR), 107, 2020.

[18] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 4510–4520, 2018.

[19] Jian-Hao Luo and Jianxin Wu. Neural network pruning with residual-connections and limited-data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1458–1467, 2020.

[20] Hengyuan Hu and Rui Peng and Yu-Wing Tai and Chi-Keung Tang. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. arXiv preprint arXiv:1607.03250, 2016.

[21] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. In The International Conference on Learning Representations (ICLR), pages 1–14, 2016.

[22] Jian-Hao Luo and Jianxin Wu. An entropy-based pruning method for cnn compression. arXiv preprint arXiv:1706.05791, 2017.

[23] Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 5058–5066, 2017.

[24] Yang He, Ping Liu, Ziwei Wang, Zhilan Hu, and Yi Yang. Filter pruning via geometric median for deep convolutional neural networks acceleration. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 4340–4349, 2019.

[25] Shaopeng Guo, Yujie Wang, Quanquan Li, and Junjie Yan. DMCP: Differentiable Markov channel pruning for neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1539– 1547, 2020.

[26] Mingbao Lin, Rongrong Ji, Yan Wang, Yichen Zhang, Baochang Zhang, Yonghong Tian, and Ling Shao. HRank: Filter pruning using high-rank feature map. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1529–1538, 2020.

[27] Bailin Li, Bowen Wu, Jiang Su, Guangrun Wang, and Liang Lin. EagleEye: Fast sub-net evaluation for efficient neural network pruning. In The European Conference on Computer Vision (ECCV), volume 12347 of LNCS, pages 639– 654. Springer, 2020.

[28] Alex Krizhevsky, and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[29] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. International Journal of Computer Vision, 115(3):211–252, 2015.

[30] Yang He, Guoliang Kang, Xuanyi Dong, Yanwei Fu, and Yi Yang. Soft filter pruning for accelerating deep convolutional neural networks. In The International Joint Conference on Artificial Intelligence (IJCAI), pages 2234–2240, 2018.

[31] Chenglong Zhao, Bingbing Ni, Jian Zhang, Qiwei Zhao, Wen-Jun Zhang, and Qi Tian. Variational convolutional neural network pruning. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2775–2784, 2019.

[32] Huiyuan Zhuo, Xuelin Qian, Yanwei Fu, Heng Yang, and Xiangyang Xue. SCSP: Specral clustering filter pruning with soft self-adaption manners. arXiv preprint arXiv:1806.05320, 2018.

[33] Dong Wang, Lei Zhou, Xueni Zhang, Xiao Bai, and Jun Zhou. Exploring linear relationship in feature map subspace for convnets compression. arXiv preprint arXiv:1803.05729, 2018.

[34] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 1389–1397, 2017.

[35] Xavier Suau, Luca. Zappella, Vinay Palakkode, and Nicholas Apostoloff. Principal filter analysis for guided network compression. arXiv preprint arXiv:1807.10585, 2018.

[36] Ruichi Yu, Ang Li, Chun-Fu Chen, Jui-Hsin Lai, Vlad I Morariu, Xintong Han, Mingfei Gao, Ching-Yung Lin, and Larry S Davis. NISP: Pruning networks using neuron importance score propagation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 9194–9203, 2018.

[37] Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference. arXiv preprint arXiv:1611.06440, 2016.

[38] Yang He, Yuhang Ding, Ping Liu, Linchao Zhu, Hanwang Zhang, and Yi Yang. Learning filter pruning criteria for deep convolutional neural networks acceleration, In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2006–2015, 2020.

[39] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on Machine Learning (ICML), pages 448–456, 2015.

[40] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. In The International Conference on Learning Representations (ICLR), pages 1–16, 2017.

[41] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: Differentiable architecture search. In The International Conference on Learning Representations (ICLR), pages 1–13, 2019.

[42] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. AMC: AutoML for model compression and acceleration on mobile devices. In The European Conference on Computer Vision (ECCV), volume 11211 of LNCS, pages 815–832. Springer, 2018.

[43] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971, 2015.

[44] Mingbao Lin, Rongrong Ji, Yuxin Zhang, Baochang Zhang, Yongjian Wu, and Yonghong Tian. Channel

pruning via automatic structure search. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), pages 673 – 679, 2020.

[45] Zechun Liu, Haoyuan Mu, Xiangyu Zhang, Zichao Guo, Xin Yang, Kwang-Ting Cheng, and Jian Sun. MetaPruning: Meta learning for automatic neural network channel pruning. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 3296–3305, 2019.

[46] Zhengsu Chen, Jianwei Niu, Lingxi Xie, Xuefeng Liu, Longhui Wei, and Qi Tian. Network adjustment: Channel search guided by FLOPs utilization ratio. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 10658–10667, 2020.

[47] Xuanyi Dong, and Yi Yang. Network pruning via transformable architecture search. In Advances in Neural Information Processing Systems 32 (NeurIPS), pages 760–771, 2019.

[48] Jiahui Yu, and Thomas Huang. AutoSlim: Towards one-shot architecture search for channel numbers. arXiv preprint arXiv:1903.11728, 2019.

[49] Sanjukta Ghosh, Shashi K. K. Srinivasa, Peter Amon, A. Hutter, Andreas Hutter, and Andr é Kaup. Deep network pruning for object detection. In IEEE International Conference on Image Processing (ICIP), pages 3915–3919, 2019.

[50] Zejiang Hou and Sun-Yuan Kung. Efficient image super resolution via channel discriminative deep neural network pruning. In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 3647–3651, 2020.