



## 基于自适应分层阈值判断的神经网络模型压缩

卢 鹏, 万 莹, 邹国良, 陈金字, 郑宗生, 王振华

(上海海洋大学 信息学院, 上海 201306)

**摘 要:** 面对多样化的应用环境, 卷积神经网络(CNN)的架构深度不断增加以提升精度, 但同时需要大量的计算参数和网络存储。针对CNN卷积层参数冗余和运算效率低的问题, 提出一种基于分层阈值的自适应动态剪枝方法。设计自适应分层阈值判断算法, 对批归一化层的尺度因子进行聚类分析, 自适应地找到每层的分类断点并据此确定最终阈值, 利用该阈值修剪正则化后的输入模型, 从而避免根据经验人为定义固定阈值, 减小模型尺寸和运行时占用的内存。分别采用该方法和LIU等提出的使用固定阈值且全局修剪的方法对VGGNet、ResNet、DenseNet和LeNet模型进行压缩, 并在CIFAR、SVHN和MNIST数据集上测试模型性能。实验结果表明, 该方法能够在模型精度与剪枝率之间找到最优平衡, 剪枝后模型的测试错误率较对比方法降低0.02~1.52个百分点, 同时自适应分层阈值判断算法也能避免对比方法在全局修剪时减去整个层的问题。

**关键词:** 深度学习; 图像识别; 卷积神经网络; 模型压缩; 网络剪枝

开放科学(资源服务)标志码(OSID):



**中文引用格式:** 卢鹏, 万莹, 邹国良, 等. 基于自适应分层阈值判断的神经网络模型压缩[J]. 计算机工程, 2022, 48(1): 112-118, 126.

**英文引用格式:** LU P, WAN Y, ZOU G L, et al. Neural network model compression based on adaptive hierarchical threshold judgment[J]. Computer Engineering, 2022, 48(1): 112-118, 126.

### Neural Network Model Compression Based on Adaptive Hierarchical Threshold Judgment

LU Peng, WAN Ying, ZOU Guoliang, CHEN Jinyu, ZHENG Zongsheng, WANG Zhenhua  
(College of Information, Shanghai Ocean University, Shanghai 201306, China)

**[Abstract]** Aiming at diversified application environments, the architecture depth of Convolutional Neural Network (CNN) is increasing to improve the accuracy, but at the same time, it needs a lot of computing parameters and network storage. An adaptive dynamic pruning method based on layered threshold is proposed to solve the problems of CNN convolution parameter redundancy and low operation efficiency. An adaptive hierarchical threshold judgment algorithm is designed to cluster the scale factors of the Batch Normalization (BN) layer, adaptively find the classification cluster points of each layer, and determine the final threshold accordingly. The regularized input model is trimmed by using the threshold, so as to avoid artificially defining a fixed threshold according to experience and reduce the model size and memory occupied during operation. This method and the method of using fixed threshold and global pruning proposed by LIU et al are used to compress VGGNet, ResNet, DenseNet and LeNet models respectively, and the model performance are tested on CIFAR, SVHN and MNIST data sets. Experimental results show that this method can find the optimal balance between model accuracy and pruning rate. The test error rate of the model after pruning is 0.02~1.52 percentage points lower than that of the comparison method. At the same time, the adaptive layered threshold judgment algorithm can also avoid the problem of reducing the whole layer in the global pruning of the comparison method.

**[Key words]** deep learning; image recognition; Convolutional Neural Network (CNN); model compression; network pruning  
**DOI:** 10.19678/j.issn.1000-3428.0060042

## 0 概述

卷积神经网络(Convolutional Neural Network,

CNN)是一类包含卷积计算且具有深度结构的前馈神经网络, 是深度学习的代表算法之一<sup>[1]</sup>。自深度卷积神经网络 AlexNet<sup>[2]</sup>问世并在图像分类领域取

**基金项目:** 国家自然科学基金(41501419, 41671431); 上海市地方院校能力建设项目(19050502100)。

**作者简介:** 卢 鹏(1981—), 男, 讲师、博士, 主研方向为深度学习、图像处理; 万 莹, 硕士研究生; 邹国良, 教授; 陈金字, 硕士研究生; 郑宗生、王振华, 副教授。

**收稿日期:** 2020-11-18 **修回日期:** 2021-01-05 **E-mail:** 568897592@qq.com

得优异成绩后,CNN算法得到了广泛应用<sup>[3]</sup>,如计算机视觉领域中的图像分类、目标检测、语义分割等。同时,在ImageNet比赛带来的分类挑战影响下,CNN技术在架构深度方面获得重大突破<sup>[4]</sup>,优秀模型已经从8层发展到100层以上。此外,CNN在人工智能、自然语言处理、故障诊断上也具有广阔的应用前景。

面对众多挑战,研究者希望构建出快速并且紧凑的CNN网络<sup>[5-6]</sup>。常见的方法有权值修剪<sup>[7-8]</sup>、低秩扩展<sup>[9]</sup>、量化、二值化<sup>[10]</sup>、动态分解等。HAN等首先计算出参数权重,在不影响整体精度的情况下修剪小幅度的权值,然后再次训练。但此方法并不能同时减少计算时间和参数量,因为删除的大部分参数属于全连接层,其在总体浮点运算中占比很小。如在VGG16中,全连接层参数占据总数的90%,但在总体浮点运算中只占比1%。低秩扩展通过对网络层的分解来降低计算成本,量化、二值化、动态分解则进行快速推理,都属于直接学习方法。然而,关于CNN模型加速和压缩的研究成果<sup>[11]</sup>,大多都只能单独提升速度或存储容量,较少有能够同时解决这两个问题的方法。另一个压缩CNN模型的方向是网络稀疏化。稀疏化可以施加于不同层次的结构,这虽然加大了模型压缩率,加快了推理速度,但重复次数多且需要特殊的软件/硬件加速器来节省时间<sup>[10]</sup>。

本文在网络正则化的基础上,提出一种新的阈值判定方法,以去除网络层中的冗余连接,减少分配给参数的存储,从而减少计算成本。文献[12]提出的网络修剪方法使用了固定阈值,从全局进行一次网络剪枝,而本文则提出自适应动态剪枝算法,在预剪枝阶段逐层自适应地分出断点,并采用断点中的最小值作为阈值,针对每层不同的阈值标准进行剪裁。

## 1 相关工作

目前CNN模型已经在计算机视觉领域取得了长足的进步,但是因为一般在一般情况下,深层网络比浅层网络的效果更好,所以卷积神经网络参数巨大,并且计算卷积层和全连接层需要大量的浮点矩阵乘法,导致计算开销也非常大,虽然有的网络可以在GPU上实时运行,但是如此庞大的规模使其无法直接应用于手机等移动设备中。因此,深度学习模型的压缩成为一个亟需解决的重要问题。

目前多数CNN网络结构(如VGGNet、AlexNet、ResNet、DenseNet<sup>[13]</sup>等)在卷积层都会产生大量的计算成本,导致全连接层中包含大量的网络参数。因此,很多加速压缩研究都集中在卷积层和全连接层。

为降低计算成本,研究者提出了近似卷积运算,即将权矩阵表示为2个较小的低秩乘积,如奇异值分解(SVD)<sup>[9]</sup>,但因为CNN中的计算操作主要来自卷积层,所以这种方法在全连接层的作用比卷积层更大。然而,实现低秩分解通常会比较困难,因为分

解过程包括计算复杂度高的分解操作。另一个问题是当前的低秩分解方法是逐层进行的,不能进行全局的参数压缩,因为不同的网络层具备的信息也是不同的。

在基于权值量化的方法中,文献[14]使用8位定点整数来代替32位浮点数,文献[15]提出将实值权值量化为二元/三元权值(限于 $\{-1,1\}$ 或 $\{-1,0,1\}$ 的权值),文献[15]提出了量化卷积神经网络,其量化权重并将计算转换为卷积层中的内积。但是,二值化网络在处理大型CNN时会使准确率明显下降。目前的二值化方式大多采用简化的矩阵近似策略,忽略了二值化对最终性能造成的影响。在权重剪枝方面,文献[12]提出用小的权值来修剪不重要的连接,通过存储稀疏格式的模型来减少存储空间,但这些方法需要专用的稀疏矩阵运算库甚至硬件才能实现加速。针对结构化稀疏问题,文献[16]提出一种结构化稀疏学习(SSL)方法来稀疏化CNN中不同层次的结构(如过滤器、通道或层),文献[17]在训练时通过随机去激活卷积层上的输入-输出通道连接来引入稀疏性,文献[18]在训练过程中施加了神经元水平的稀疏性,使一些神经元可以被修剪以获得紧凑的网络。上述方法都利用训练过程中的群体稀疏化来获得结构化稀疏性,是针对神经元或者通道而不是权重进行修剪,因此不需要太多专用库来实现推理加速,本文所使用的精简方法也属于此类。此外,文献[19]通过组合网络修剪、加权量化和霍夫曼编码来进一步减少存储,文献[20]提出动态网络手术,参考文献[9]的方法动态修剪和拼接连接。

虽然已有很多经典的CNN模型<sup>[1,21-22]</sup>,但也有一些自动学习网络架构的探索,如在给定资源预算的情况下引入网络架构搜索的子模块/超模块优化<sup>[23]</sup>。还有一些研究<sup>[24]</sup>提出利用强化学习来自动学习神经网络结构,这些方法的搜索空间非常大,需要训练数百个模型来区分优劣。虽然选择仅限于每一层的宽度,但网络剪枝也可以被视为一种架构学习方法。

目前除了对BP神经网络的研究外,还有针对非迭代神经网络的研究。文献[25]提出一种通用的几何变换模型,遵循了训练和自我训练原则,为广泛的问题提供了有效的解决方案。文献[26]提出了一种以线性多项式作为构造公式求解多元线性回归任务的新方法,能够实现快速识别同时又能保证高精度。这些方法也是压缩中的重要分支。

在上述的各类方案中,不同层级的稀疏性有不同的特点,最粗的层级在修剪时不需要特殊的包裹获得推理加速,但其灵活性较差,有时会剪掉完整的层,并且只有当深度超过50层时,去除层才有效果。而更细致的修建方式(如权重修剪)虽然具有较高的灵活性和通用性,但却又需要特殊的软件和硬件进行推理<sup>[22]</sup>。相比之下,居中的通道级别的修剪及通道级稀疏,在灵活性和易于实现之间实现了良好的

折衷,其可应用于经典的CNN网络,即将每个神经元视为一个通道。

2 自适应动态剪枝方法

本文设计一种自适应方案来实现CNN的通道级稀疏。稀疏性在不同的层级上有不同的级别,如层级、通道级、内核级或重量级。最粗的层级稀疏不需要特殊的软件或硬件来进行加速,但灵活性较差。最细的重量级稀疏虽然能提供最高的灵活性和通用性,却通常需要特殊的软件或硬件加速器。相比之下,通道级的稀疏性在灵活性和易于实现之间实现了很好的折衷,其可应用于任何典型的CNN或全连接网络(将每个神经元视为一个通道),得到的网络本质上是未修剪网络的“精简版”,可以在传统的CNN平台上进行有效推理,从而达到压缩的目的。在本节中将介绍如何利用批归一化(Batch Normalization, BN)层的权重,通过自适应分层阈值判断算法来确定阈值,从而有效识别和删除网络中不重要的通道。剪枝过程如图1所示。首先将基线模型按文献[14]的方法正则化,获得正则化模型后,通过自适应阈值判断对正则化后的权重进行分类,以最小值作为阈值进行剪枝,移除不重要的权重。由浅到深,直至最后一层,最后对剪枝结束的模型继续训练,以提高精度。

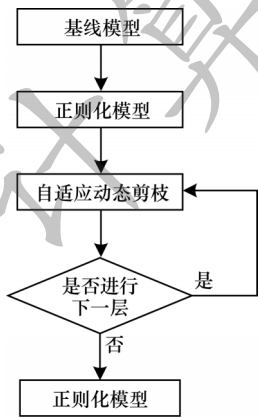


图1 剪枝流程

Fig.1 Pruning process

2.1 正则化必要性分析

在之前的工作中,基线模型进行训练后将再次进行正则化处理,这是用一种相对简单的模型来拟合复杂数据。有一些正则化可以控制神经网络防止过拟合,如L1正则化、L2正则化<sup>[27]</sup>、ElasticNet回归<sup>[28]</sup>、最大范数约束<sup>[29]</sup>、Dropout<sup>[30]</sup>等。在L1正则化中,对每个权重 $w$ ,将 $\lambda|w|$ 添加到目标函数中,其中, $\lambda$ 是正则化强度;在L2正则化中,对于网络中的每个权重 $w$ ,将 $1/2\lambda w^2$ 添加到目标中,其中, $\lambda$ 同为正则化强度;但有时L1回归太过,过多特征被稀疏为0,而当L2回归正则化不够,回归系数衰减太慢时,ElasticNet综合回归则会带来更好的效果。ElasticNet回归综合了L1正则化项和L2正则化项,

其目标函数如下:

$$\operatorname{argmin}_{\beta} (\|y - X_{\beta}\|^2 + \lambda_2 \|\beta\|^2 + \lambda_1 \|\beta\|_1) \quad (1)$$

正则化后可以得到一个稀疏模型,表示只有少数特征对这个模型有贡献,绝大部分特征是没有贡献的,或者贡献微小的可以去掉,为了探究正则化后权重的作用,本文将文献[12]中VGG16基线模型与正则化后进行对比。如图2所示,正则化方法削弱了不重要的特征变量,减小了特征变量的数量级。

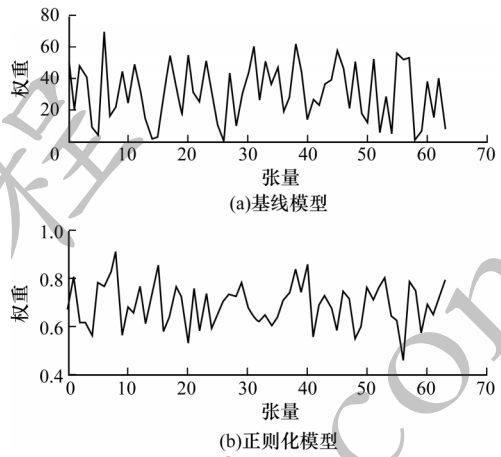


图2 正则化前后第2层通道权重的对比

Fig.2 Comparison of channel weights of the second layer before and after regularization

通过将正则化项引入神经网络的代价函数中,可得到输入数据的稀疏性特征表示。稀疏性特征可以防止过拟合,提高泛化能力,更好地解释模型。如同从生物学的角度来说,人脑中的大量神经元,当受到外界刺激(图像或者声音)时,只有少量的神经元被激活,大部分神经元处于抑制状态。因此,正则化后的权重值全部减小,已有部分通道接近于0。此时再进行裁剪时,能够得到更好的效果。表1中数据显示,采用文献[12]中相同的剪枝方法,正则化前后的模型精度相近,但正则化前的剪枝率仅16%,正则化后的剪枝率可达70%,由此进一步证明了正则化的必要性。

表1 正则化前后在相同阈值下的剪枝率对比

Table 1 Comparison of pruning rates under the same threshold before and after regularization %

模型	阈值	剪枝率	精度
基线模型	断点最小值	17	93.5
正则化模型	断点最小值	70	93.7

虽然正则化能够为剪枝带来优势,但之前的剪枝方法中确定全局阈值需要经过大量实验以及经验,通过逐次比对后选取最优阈值百分比,且无法知道具体的阈值。因此,本文在前期正则化的基础上,提出自适应分层阈值判断算法,根据每层的权重分析对权重重点进行分类,形成断点后选取最小值,逐层修剪加权连接,这种方法可以更准确地对阈值进行



判断,在剪枝率与精度中达到最优平衡,得到更高的精度,从而更好地进行网络模型压缩。

## 2.2 分层断点判断

在文献[12]关于网络修剪的工作中,采用固定的百分比作为阈值进行全局修剪。但是如果固定百分比太高,则会导致一开始就修剪了大量连接,导致性能急剧下降。相反,如果固定百分比太低,则压缩比可能比期望的价值小很多。因此,需要大量实验进行经验积累,才能确定最终阈值。为了解决这个问题,本文使用自适应分层阈值判断算法对BN层尺度因子进行聚类分析。在众多散乱的尺度因子值中,自适应地找到分类簇点,再根据簇点确定最终阈值。这避免了单一的百分比方式去确定阈值,使得每层都将有其自适应的具体阈值。

按照文献[12]中的稀疏方法,利用BN层的缩放因子,通过联合计算后得到每层的尺度因子,并根据这些尺度因子计算每层的阈值进行剪枝。每一层的自适应分层断点算法描述和计算流程如下:

### 算法1 自适应分层断点算法

输入 每层的缩放因子  $X$

输出 每层的断点

1. While  $i < n$  do
2.  $K(x) = K(x) + (X_i - X)/h$
3.  $F(x) = K(x)/(n \times h^d)$
4. 将这些向量相加得到  $M_h(X)$
5.  $m_h(X) = M_h(X) + X$
6.  $X$  向  $M_h(X)$  方向移动
7.  $i = i + 1$
8. 将迭代过程中遇到的所有点分类进集群  $C$
9. if 当前集群簇心距离其他现有集群簇心距离  $\leq 0.5$
10. 合并
11. end if
12. 重复以上步骤直到输出所有断点
13. End while

1) 在每层的尺度因子中,随机选取一个中心点  $X$ ,以半径  $h$  为内核,找出内核内的所有数据点,将这些点归属于一个聚类  $C$ ,同时,在该聚类中记录数据点出现的次数加1,再利用概率密度计算中心点到每个内核数据点的核密度,即加权平均值。目标函数如下:

$$\hat{f}(X) = \frac{\sum_{i=1}^n \left( \frac{X_i - X}{h} \right)}{n \cdot h^d} \quad (2)$$

目标函数的梯度估计如下:

$$\nabla \hat{f}(X) = - \frac{2 \sum_{i=1}^n (X_i - X) K' \left( \frac{X_i - X}{h} \right)}{n \cdot h^d} = - \frac{2}{h^2} \left[ \frac{\sum_{i=1}^n G \left( \frac{X_i - X}{h} \right)}{n \cdot h^2} \right] \left[ \frac{\sum_{i=1}^n (X_i - X) G \left( \frac{X_i - X}{h} \right)}{\sum_{i=1}^n G \left( \frac{X_i - X}{h} \right)} \right] \quad (3)$$

2) 将这些向量相加,得到向量  $M_h(X)$ ,  $X$  将沿着  $M_h(X)$  的方向移动,移动距离为  $\|M_h(X)\|$ ,即  $m_h(X) =$

$M_h(X) + X$ ,然后重复迭代。在每一次迭代中,滑动窗口会移向密度较高的区域,将中心点移动到窗口内的点的平均值,滑动窗口中的密度与它内部的点的数量成比例。通过移向窗口中点的平均值,它将逐渐向更高的点密度方向移动,移动后的样本点迭代如下:

$$M_h(X) = \frac{\sum_{i=1}^n [X_i - X] K \left( \frac{X_i - X}{h} \right)}{\sum_{i=1}^n K \left( \frac{X_i - X}{h} \right)} = \frac{\sum_{i=1}^n (X_i) K \left( \frac{X_i - X}{h} \right)}{\sum_{i=1}^n K \left( \frac{X_i - X}{h} \right)} - X = m_k(X) - X \quad (4)$$

将迭代过程中遇到的点都应归类与簇  $C$ 。若收敛时当前簇  $C$  的中心与其他已存在的簇心距离过小,则会将其合并,数据点出现次数也对应合并。再次重复上述步骤直到所有点都被标记。

最后根据对每个点的访问频率,将访问频率最高的点作为断点。实际上,每一个样本点都需要计算其均值,并根据计算出的均值进行移动,直到满足终止条件,最终得到的数据点也就是该点的聚类中心点。针对不同的BN层,会自动聚类出不同的断点。

## 2.3 阈值判定

在聚类出断点后,将其由小到大进行排序,由断点确定具体阈值。若选取的断点数值越大,剪枝率将越高。由于权重越小,重要性越低,若选取的断点越小,精度就将越高。因此,为了由断点确定最优的阈值,针对不同层进行了对比实验。如图3所示,在CIFAR10数据集上,假设每层均采用断点最小值作为阈值为基础设置,在后续实验中,将第2层选取倒数第二小的断点作为阈值,其他层仍采用最小值。依次再将第3~第6层做同样设置可看出,若增大阈值,提高剪枝率的情况下,精度将大幅下降。并且由于第2层的通道数为64,提高阈值后对精度的影响最大。后续3、4层的通道数为128,5、6层的通道数为256,其精度影响逐渐减小,但仍超出文献[12]中最低精度,因此更加不考虑倒数第三小的断点作为阈值,可见全局采用断点最小值作为阈值为最佳设置。

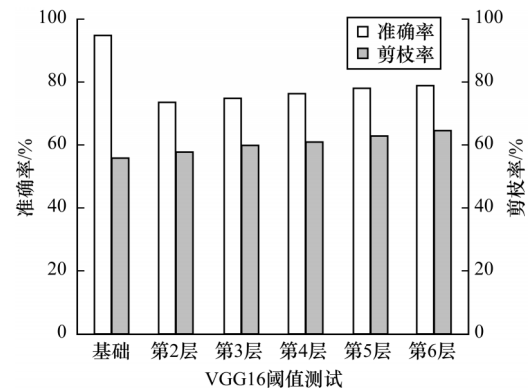


图3 VGG16阈值测试结果

Fig.3 VGG16 threshold test result

### 3 实验

本文实验修剪了简单的 CNN 模型 (VGGNet、ResNet、DenseNet)。与常用的压缩模型 AlexNet 不同, VGG 在全连接层的参数较少, 因此, 在这样的网络中修剪更具有挑战性。在 pytorch 深度学习平台上实现了文献[12]中的基线模型和正则化后的模型, 并根据自适应分层阈值判断结果, 继续修剪网络中的通道。

#### 3.1 数据集

本文实验使用以下数据集:

1) CIFAR<sup>[31]</sup>。CIFAR 包括 CIFAR-10 和 CIFAR-100。CIFAR-10 数据集由 10 个类组成, 每个类包含 60 000 张图像, 50 000 张用于训练, 10 000 张用于测试。CIFAR-100 数据集则由 100 个类组成, 每个类包含 600 张图像, 与 CIFAR-10 不同的是, CIFAR-100 中的 100 个类又被分为 20 个超类, 每个图像都带有一个“精细”标签(它所属的类)和一个“粗糙”标签(它所属的超类)。

2) SVHN。SVHN 是一个现实世界的图像数据集, 用于机器学习和识别算法。分为完整数据与裁剪数据, 其中完整数据是带有字符级边界框的原始图像, 裁剪数据则已调整为固定的 32 像素×32 像素分辨率。按照惯例, 仍然使用 604 388 个裁剪数据图像进行训练。

3) MNIST。MNIST 是一个手写数字数据集, 包含 60 000 张训练图片和 10 000 张测试图像。为了测试本文方法在全连接网络上的有效性, 本文将与文献[12]方法进行对比。

#### 3.2 实验网络

在 CIFAR 和 SVHN 数据集上, 实验在 3 种流行的网络架构 VGGNet、ResNet 和 DenseNet 上评估本文方法。VGGNet 最初是为 ImageNet 分类而设计的。在实验中, 使用包含 VGGNet 的 16 和 19 层网络结构和使用具有瓶颈结构 (ResNet-164) 的 164 层预激活 ResNet。对于 DenseNet, 使用生长速率为 12 (DenseNet-40) 的 40 层 DenseNet。在 MNIST 数据集上, 在与文献[12]相同的三层全连接网络上评估本文方法。

#### 3.3 训练、剪枝与微调

训练、剪枝与微调过程如下:

1) 基线训练。首先从零开始训练所有网络, 作为基线网络数据进行对比。所有网络都使用 SGD (Stochastic Gradient Descent)。在 CIFAR 数据集上进行 160 个 epoch 的训练, 在 SVHN 数据集上进行 20 个 epoch 的训练, 2 个数据集批量大小同为 64, 学习率同为 0.1。在 MNIST 数据集上进行 30 个 epoch

的训练, 批量大小为 256, 初始学习率为 0.1, 经过 20 个 epoch 后初始学习率为 10。为了更好地对比阈值选定对精度结果和剪枝率的影响, 本文方法参数设置与文献[12]方法保持一致。

2) 稀疏训练。在稀疏化训练时仍沿用文献[12]方法, 为每个通道引入比例因子。联合训练网络权重与比例因子, 对比例因子施加稀疏正则化后得到稀疏模型。

3) 剪枝。针对稀疏以后的模型进行剪枝, 需要确定剪枝阈值。在文献[12]中采用统一全局阈值一次修剪, 且阈值定义为所有比例因子值的某个百分比。缺点是由于不同层数的参数量不同, 重要性也不同, 全局阈值需要多次反复实验, 最终确定具体阈值, 且增加 1% 或者降低 1% 剪枝比例对最终精度结果影响不大, 因此阈值的判断仅靠人为选择。而本文将针对不同层, 使用自适应动态剪裁算法, 自动确定不同层的具体阈值, 而非百分比在精度与剪枝率取得平衡。这使得其能更好地进行裁剪, 选取适合该层的阈值, 而无需多次反复实验。

4) 微调。在剪枝后, 将得到一个更窄、更紧凑的模型, 实验将继续对模型微调, 在 CIFAR、SVHN 和 MNIST 数据集上, 使用与基线训练中相同的数据设置, 再次训练, 以取得更好的精度效果。

#### 3.4 结果对比与分析

在以往的经验中, 第 1 个卷积层在原始图像的提取特征中起着重要作用, 只存在较低的计算成本和较小的参数存储(小于整个网络的 1%)。因此, 在本文实验中不修改该层的参数。同时网络的全连接层不包含 BN 层, 同样不进行修剪。

以 VGG16 网络为例, 图 4 给出了剪枝层的断点数。网络层越深, 通道的权重值增多, 断点数也逐步振荡增加, 但每层仍以断点最小值为剪枝阈值。图 5 对比了剪枝前后通道数, 可以看出层数越深剪枝幅度越大, 通常在更深的网络层具有的激活映射更小, 占用的内存也更少, 因此会有更多的通道被修剪。

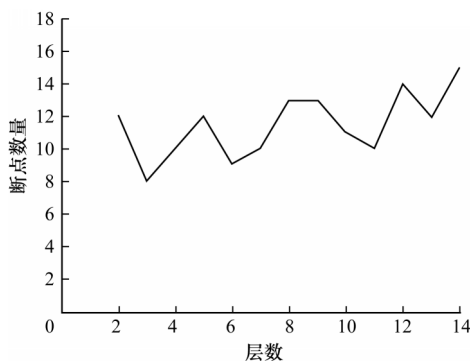


图 4 VGG16 分层断点数

Fig.4 Number of VGG16 layered break points

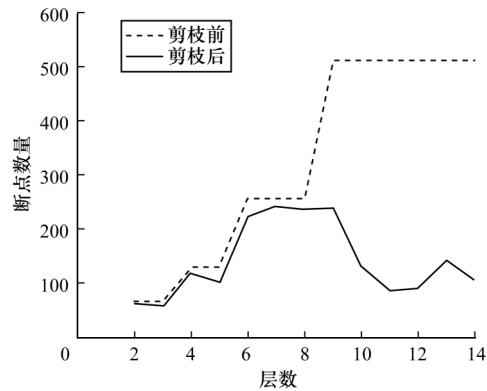


图5 VGG16分层剪枝通道数

Fig.5 Number of VGG16 layered pruning channels

本文给出了VGG网络在数据集上的测试结果,为了更好地进行效果对比,以剪枝前的参数作为基线,将文献[12]方法的剪枝率、测试错误率和参数量与本文方法进行对比,如表2~表4所示。可以看出,本文不必依靠大量实验或者以往经验确定阈值,而是使用自适应分层阈值判断算法可以自适应地根据断点选择阈值进行分层剪枝,可以保持甚至提高至更好的精度。

表2 CIFAR10数据集测试结果

Table 2 Test result in CIFAR10 data set

模型	压缩方法	剪枝率/%	测试错误率/%	参数量/ $10^6$
VGG16	基线模型	0	6.32	20.04
	文献[12]方法	70	6.20	2.30
	本文方法	56	5.21	2.50
VGG19	基线模型	0	6.27	20.42
	本文方法	65	5.00	7.80
DenseNet	基线模型	0	6.11	1.02
	文献[12]方法	40	5.19	0.66
	本文方法	36	4.21	0.69
ResNet	基线模型	0	5.42	1.07
	文献[12]方法	40	5.08	1.44
	本文方法	34	4.93	1.47

表3 CIFAR100数据集测试结果

Table 3 Test result in CIFAR100 data set

模型	压缩方法	剪枝率/%	测试错误率/%	参数量/ $10^6$
VGG16	基线模型	0	26.74	20.04
	文献[12]方法	50	26.52	5.00
	本文方法	43	25.00	5.80
VGG19	基线模型	0	27.30	20.49
	本文方法	57	26.80	9.60
DenseNet	基线模型	0	25.36	1.06
	文献[12]方法	40	25.28	0.66
	本文方法	32	24.66	0.68
ResNet	基线模型	0	23.37	1.73
	文献[12]方法	40	22.87	1.46
	本文方法	32	21.96	1.49

表4 SVHN数据集测试结果

Table 4 Test result in SVHN data set

模型	压缩方法	剪枝率/%	测试错误率/%	参数量/ $10^6$
VGG16	基线模型	0	2.17	20.4
	文献[12]方法	60	2.06	3.04
	本文方法	49	1.97	3.26
VGG19	基线模型	0	2.15	20.42
	本文方法	46	1.95	8.00
DenseNet	基线模型	0	1.89	1.02
	文献[12]方法	40	1.79	0.65
	本文方法	36	1.61	0.68
ResNet	基线模型	0	1.78	1.70
	文献[12]方法	40	1.85	1.46
	本文方法	33	1.75	1.49

CIFAR10数据集上的实验结果显示:在使用本文方法情况下,对VGG16网络模型剪枝率为56%,测试错误率较使用文献[12]方法降低1.01个百分点;对DenseNet网络模型剪枝率为36%,测试错误率较使用文献[12]方法降低0.98个百分点;对ResNet网络模型剪枝率为32%,测试错误率较使用文献[12]方法降低0.15个百分点。

在CIFAR100数据集上的实验结果显示:在使用本文方法情况下,对VGG16网络模型剪枝率为43%,测试错误率较使用文献[12]方法降低1.52个百分点;对DenseNet网络模型剪枝率较使用文献[12]方法增加8%,测试错误率降低0.62个百分点;对ResNet网络模型剪枝率为32%,测试错误率较使用文献[12]方法降低0.91个百分点。

在SVHN数据集上的实验结果显示:在使用本文方法情况下,VGG16网络模型相较于文献[12]方法牺牲了11%的剪枝率和 $0.21 \times 10^6$ 的模型参数量,降低测试错误率0.09个百分点;对DenseNet剪枝率较使用文献[12]方法下降4%,测试错误率降低0.02个百分点,模型参数量增加0.03%;ResNet网络模型剪枝率为33%,测试错误率较使用文献[12]方法降低0.1个百分点。

由此可以看出,本文方法能够降低错误率,调节精度与剪枝率的平衡。虽然剪枝率没有超过文献[12]方法,但本文方法可以更好地在剪枝与精度中找到最优平衡。表3结果显示,在同样的数据集上,网络层越深效果更好,同时剪枝率也会相应提高。

针对LeNet网络模型,本文在MNIST数据集上进行了测试并与文献[12]全局剪枝方法进行了比较,测试结果如表5所示,其中测试错误率降低0.05个百分点,参数剪枝率达到82.8%。尽管正则化的作用主要用于对卷积层中的通道进行剪枝,但它也适用于对全连接层中的神经元进行剪枝。文献[12]中使用全局阈值修剪时有时会剪去整个层,



而本文自适应分层阈值判断算法则避免了这个问题,并获得了略低的测试误差。

表5 MNIST 测试结果 %

模型	剪枝方法	测试错误率	参数剪枝率
LeNet	基线模型	1.43	0.0
	文献[12]方法	1.49	84.4
	本文方法	1.44	82.8

#### 4 结束语

目前,深度学习已经在图像、文本、音频、视频等诸多领域取得了巨大成功,同时也推动了很多相关智能产品的诞生。CNN网络模型虽然结构越深效果越好,但是高容量、高训练成本和推理成本都会影响计算效率。本文提出的自适应分层阈值判断算法,针对正则化后的模型进行剪枝,逐层对通道权重进行断点分层后,选取最小值为阈值进行剪枝,摆脱了以往需要经验人为定义的固定阈值,进而获得更加紧凑的卷积神经网络,基于该算法可减少模型尺寸,且得到的模型不需要特殊的库/硬件来进行有效的推理。在多个数据集上的实验结果证明,本文方法可以减小模型尺寸且精度并没有显著损失。未来将研究多种压缩方法相结合的方法,以获得更好的压缩结果。

#### 参考文献

- [1] GOODFELLOW I, BENGIO Y, COURVILLE A. Deep learning[M]. Cambridge, USA: MIT Press, 2016: 326-366.
- [2] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017, 60(6): 84-90.
- [3] SZEGEDY C, LIU W, JIA Y, et al. Going deeper with convolutions[C]//Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition. Washington D. C. , USA: IEEE Press, 2015: 1-9.
- [4] HE K M, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]//Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Washington D. C. , USA: IEEE Press, 2016: 1-5.
- [5] ZOU Y X, YU J S, CHEN Z H, et al. Convolution neural networks model compression based on feature selection for image classification[J]. Control Theory & Applications, 2017, 34(6): 746-752.
- [6] 叶子, 肖诗斌. 卷积神经网络模型压缩在图像分类中的应用[J]. 北京信息科技大学学报, 2018, 33(3): 52-56.  
YE Z, XIAO S B. Compression of convolutional neural network applied to image classification[J]. Journal of Beijing Information Science & Technology University, 2018, 33(3): 52-56. (in Chinese)
- [7] SONG H, JEFF P, JOHN T, et al. Learning both weights and connections for efficient neural network[J]. International Journal of Neural Systems, 1996, 7(2): 129-147.
- [8] HAN S, LIU X Y, MAO H Z, et al. EIE: efficient inference engine on compressed deep neural network[J]. Computer Architecture News, 2016, 44(3): 243-254.
- [9] DENTON E L, ZAREMBA W, BRUNA J, et al. Exploiting linear structure within convolutional networks for efficient evaluation[C]//Proceedings of the 28th Conference on Neural Information Processing Systems. Montreal, Canada: Morgan Kaufmann Press, 2014: 1269-1277.
- [10] RASTEGARI M, ORDONEZ V, REDMON J, et al. XNOR-Net: ImageNet classification using binary convolutional neural networks[C]//Proceedings of European Conference on Computer Vision. Berlin, Germany: Springer, 2016: 525-542.
- [11] ZHANG X, ZOU J, MING X, et al. Efficient and accurate approximations of nonlinear convolutional networks[C]//Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. Washington D. C. , USA: IEEE Press, 2015: 1984-1992.
- [12] LIU Z, LI J G, SHEN Z Q. Learning efficient convolutional networks through network slimming[C]//Proceedings of the 16th IEEE International Conference on Computer Vision. Washington D. C. , USA: IEEE Press, 2017: 2755-2763.
- [13] HUANG G, LIU Z, LAURENS V D M, et al. Densely connected convolutional networks[C]//Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. Washington D. C. , USA: IEEE Press, 2017: 2261-2269.
- [14] VANHOUCKE V, SENIOR A, MAO M Z. Improving the speed of neural networks on CPUs[C]//Proceedings of Deep Learning and Unsupervised Feature Learning Workshop. Cambridge, USA: MIT Press, 2011: 1-4.
- [15] WU J, CONG L, WANG Y, et al. Quantized convolutional neural networks for mobile devices[C]//Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Washington D. C. , USA: IEEE Press, 2016: 4820-4828.
- [16] WEN W, WU C, WANG Y, et al. Learning structured sparsity in deep neural networks[C]//Proceedings of NIPS'16. Cambridge, USA: MIT Press, 2016: 1-9.
- [17] CHANGPINYO S, SANDLER M, ZHMOGINOV A. The power of sparsity in convolutional neural networks[C]//Proceedings of International Conference on Learning Representations. Toulon, France: [s. n. ], 2017: 1-13.
- [18] HAO Z, ALVAREZ J M, PORIKLI F. Less is more: towards compact CNNs[C]//Proceedings of European Conference on Computer Vision. Berlin, Germany: Springer, 2016: 1-16.
- [19] HAN S, MAO H, DALLY W J. Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding[J]. Fiber, 2015, 56(4): 3-7.
- [20] GUO Y, YAO A, CHEN Y. Dynamic network surgery for efficient DNNs[C]//Proceedings of NIPS'16. Cambridge, USA: MIT Press, 2016: 1379-1387.
- [21] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[C]//Proceedings of ICLR'17. Palais, France: ICLR, 2017: 1-5.
- [22] HE K, ZHANG X, REN S, ET AL. Deep residual learning for image recognition[C]//Proceedings of Conference on Computer Vision and Pattern Recognition. Washington D. C. , USA: IEEE Press, 2016: 1-12.

(下转第126页)

(上接第118页)

- [23] BELLO I, ZOPH B, VASUDEVAN V, et al. Neural optimizer search with reinforcement learning [C]// Proceedings of the 34th International Conference on Machine Learning. New York, USA: ACM Press, 2017: 1-16.
- [24] BAKER B, GUPTA O, NAIK N. Designing neural network architectures using reinforcement learning [C]// Proceedings of ICLR'17. Palais, France: ICLR; 2017: 1-5.
- [25] TKCVHENKO R, IZONIN I. Model and principles for the implementation of neural-like structures based on geometric data transformations [C]// Proceedings of ICCSEE'18. Berlin, Germany: Springer, 2018: 578-587.
- [26] IZONIN I, TKACHENKO R, KRYVINSKA N, et al. Multiple linear regression based on coefficients identification using non-iterative SGTm neural-like structure [C]// Proceedings of the 15th International Work-Conference on Artificial Neural Networks. Gran Canaria, Spain: [s. n.], 2019: 467-479.
- [27] ZHU J, HASTIE T. Classification of gene microarrays by penalized logistic regression [J]. Biostatistics, 2004, 5(3): 427-443.
- [28] MARIO Z, MICHAEL G. Accelerating K-means on the graphics processor via CUDA [C]// Proceedings of the 1st International Conference on Intensive Applications and Services. Washington D. C., USA: IEEE Press, 2009: 7-15.
- [29] KWAK N. Principal component analysis based on L1-norm maximization [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, 30(9): 1672-1680.
- [30] SRIVASTAVA N, HINTON G, KRIZHEVSKY A, et al. Dropout: a simple way to prevent neural networks from overfitting [J]. Journal of Machine Learning Research, 2014, 15(1): 1929-1958.
- [31] GAO H, YU S, ZHUANG L, et al. Deep networks with stochastic depth [C]// Proceedings of ECCV'16. Berlin, Germany: Springer, 2016: 1-13.

编辑 金胡考