计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2020,29(9):16−25 [doi: 10.15888/j.cnki.csa.007632]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

卷积神经网络压缩与加速技术研究进展①

尹文枫[1], 梁玲燕[1], 彭慧民[1], 曹其春[1], 赵　健[1], 董　刚[1], 赵雅倩[1], 赵　坤[2]

[1](浪潮电子信息产业股份有限公司, 济南 250101)
[2](广东浪潮大数据研究有限公司, 广州 510632)
通讯作者: 尹文枫, E-mail: yinwenfeng@inspur.com

摘　要: 神经网络压缩技术的出现缓解了深度神经网络模型在资源受限设备中的应用难题, 如移动端或嵌入式设备. 但神经网络压缩技术在压缩处理的自动化、稀疏度与硬件部署之间的矛盾、避免压缩后模型重训练等方面存在困难. 本文在回顾经典神经网络模型和现有神经网络压缩工具的基础上, 总结参数剪枝、参数量化、低秩分解和知识蒸馏四类压缩方法的代表性压缩算法的优缺点, 概述压缩方法的评测指标和常用数据集, 并分析各种压缩方法在不同任务和硬件资源约束中的性能表现, 展望神经网络压缩技术具有前景的研究方向.

关键词: 神经网络压缩; 参数剪枝; 参数量化; 低秩分解; 知识蒸馏

引用格式: 尹文枫,梁玲燕,彭慧民,曹其春,赵健,董刚,赵雅倩,赵坤.卷积神经网络压缩与加速技术研究进展.计算机系统应用,2020,29(9):16–25. http://www.c-s-a.org.cn/1003-3254/7632.html

Research Progress on Convolutional Neural Network Compression and Acceleration Technology

YIN Wen-Feng[1], LIANG Ling-Yan[1], PENG Hui-Min[1], CAO Qi-Chun[1], ZHAO Jian[1], DONG Gang[1], ZHAO Ya-Qian[1], ZHAO Kun[2]

[1](Inspur Electronic Information Industry Co. Ltd., Jinan 250101, China)
[2](Guangdong Inspur Big Data Research Co. Ltd., Guangzhou 510632, China)

Abstract: The development of neural network compression relieves the difficulty of deep neural networks running on resource-restricted devices, such as mobile or embedded devices. However, neural network compression encounters challenges in automation of compression, conflict of the sparsity and hardware deployment, avoidance of retraining compressed networks and other issues. This paper firstly reviews classic neural network models and current compression toolkits. Secondly, this paper summarizes advantages and weaknesses of representative compression methods of parameter pruning, quantization, low-rank factorization and distillation. This paper lists evaluating indicators and common datasets for the performance evaluation and then analyzes compression performance in different tasks and resource constraints. Finally, promising development trends are stated in this paper as references for promoting the neural network compression technique.

Key words: neural network compression; parameter pruning; parameter quantizatipn; low-rank factorization; knowledge distillation

随着硬件的发展, 如图形处理单元 (GPU)[1] 和张量处理单元 (TPU)[2], 以及深度学习算法的成功, 如 AlexNet[3]、16 层 VGG[4] 和 152 层 ResNet[5], 基于深度学习的应用在计算机视觉、语音识别和推荐系统等广泛领域得到普及. 这些强大的深度学习模型伴随着在延迟、存储、算力和能耗等方面的资源开销增加, 给资源有限的移动

① 收稿时间: 2020-02-26; 修改时间: 2020-03-17, 2020-04-10; 采用时间: 2020-04-17; csa 在线出版时间: 2020-09-04

和嵌入设备实现离线深度感知带来了困难. 性能良好的 VGG-16 模型[4] 采用 8 比特量化之后, 由 ImageNet 数据集[6] 训练, 需要1.5×10^{10}次乘法累加操作, 1.4×10^{8}个参数, 1650 ms 的平均延迟, 在 RedMi 3S Android 平台上能耗为 397.7 mJ[7]. 因此, 压缩神经网络的参数和计算, 有助于将一些典型的基于深度学习算法的应用如语音助手、人脸识别、指纹解锁和文本处理工具等部署在移动平台.

本文将对模型压缩技术中的代表方法进行介绍与分析. 但是, 诸如 MobileNet、Inception、SqueezeNet 等采用紧致的卷积核或高效的计算方式来搭建深度神经网络的轻量化模型设计方法不在本文讨论范围内. 不同于在预训练网络上进行处理, 轻量化模型设计方法另辟蹊径. 轻量化模型设计方法采用紧致的卷积核或高效的计算方式来搭建深度神经网络, 而不是由预训练神经网络进行神经元或神经元连接的删减来实现模型压缩.

1 神经网络模型与压缩工具

本节主要介绍经典的神经网络模型, 这些代表性模型通常应用于评测新兴压缩方法的性能, 此外本节汇总集成最新模型压缩方法的各个压缩工具包的特性, 并简述模型压缩方法在硬件部署方面的进展.

1.1 经典深度神经网络模型回顾

随着 LeNet 的提出, 卷积神经网络进入了大众的视野. 在此基础上形成了 AlexNet 网络, 该经典网络结构与 LeNet-5 的结构类似, 但网络层次进一步加深. 目前演变出的多种卷积神经网络, 如 VGG、GoogleNet、ResNet 等, 虽然模型性能越来越好, 但网络的层数和计算量也随之增大, 不利于边缘设备或云端的部署.

在 2014 年的 ImageNet 挑战赛中脱颖而出的 VGG 网络[4] 具有两种常用拓扑结构 VGG16 和 VGG19. 表 1 列举了 VGG 网络等模型的参数量、模型所需内存大小以及计算量. 其中 flops 表示浮点运算次数, 用来衡量模型的复杂度. 如表 1 所示, VGG 网络结构有上亿的参数, 计算量巨大, 因此在部署过程中, 消耗较大的存储容量和计算资源, 不利于边缘端的部署.

ResNet 网络[5] 结构的核心是残差学习单元, 其解决了增加神经网络深度时精度退化的问题, 让深度神经网络结构能够达到更深的水平, 如 ResNet152 网络就有 152 层卷积.

表 1 深度网络模型的资源需求汇总

主干网络	参数 (百万)	模型内存 (MB)	计算量 (Gflops)
AlexNet	60	233	0.7
VGG19	144	548	19.6
ResNet152	60	230	11.3
GoogleNet	7	27	1.6
MobileNetV1	4.2	16	0.58
ShuffleNet(1.5)	2.9	11	0.29
ShuffleNet(×2)	4.4	16.7	0.52
MobileNetV2	3.4	12.9	0.3

为了在资源受限的设备上部署深度神经网络, 轻量化模型设计的思路应运而生, 随即产生了 MobileNet 网络[8]. MobileNet 最大的特点是采用了深度可分离卷积的独特设计, 将普通卷积拆分为深度卷积和点卷积两步. 深度可分离卷积相比于标准卷积, 在保持精度几乎不变的情况下, 参数量和计算量都大大减小. 沿着采用深度可分离卷积的思路, 相继衍生出 MobileNets-v2[9]、ShuffleNet-v1[10] 和 ShuffleNet-v2[11] 等轻量化网络模型. 虽然 MobileNet 网络结构相比 VGG19 等网络已经减小了很多, 但在移植到移动端人工智能应用中时仍然会消耗大量计算资源, 而且 MobileNet 中依然存在稀疏性, 还有继续压缩的空间.

1.2 模型压缩工具与硬件部署

随着神经网络模型压缩方法的发展, 已经孕育出一系列承载最新成果的压缩方法工具包, 表 2 列举了一些常用的压缩方法工具包. 其中, Distiller、Pocketflow、PaddleSlim 均提供多种参数剪枝方法、量化方法、知识蒸馏 (Knowledge Distillation, KD) 方法的支持, 并且提供自动化模型压缩算法 AMC 的实现. Distiller 工具包复现了基本的幅度剪枝算法以及敏感度剪枝等多种近年来新兴的剪枝算法[12–14], 涵盖适用于 RNN 的剪枝算法[15] 和面向 CNN 的算法, 此外该工具包还集成了对称线性量化等几种量化算法. PocketFlow 工具包除了腾讯自研的鉴别力感知的通道剪枝算法[16] 外, 还提供了深鉴科技[17]、谷歌公司研发的剪枝算法[18] 的复现.

结构化压缩方法在上述压缩工具包中得到更多应用的因素之一, 是模型压缩方法在硬件平台的部署会受到矩阵稀疏性粒度的影响. 如图 1 所示, 结构化压缩方法的稀疏性粗粒度可分为滤波器级、通道级和向量级, 非结构化压缩方法的稀疏性细粒度为元素级. 虽然非结构化压缩方法可取得高压缩率以及高准确率, 但

非结构化压缩后的权重矩阵或特征图矩阵中非零值的位置是不规则的, 这为有效地支持硬件中稀疏矩阵的存储与计算造成困难. 在不同的硬件平台中稀疏矩阵的处理需要调用特定的运算库来加速, 在 GPU 上稀疏矩阵计算需要调用 cuSPARSE 库, 在 CPU 上稀疏矩阵计算稀疏需要 mkl_sparse 之类的库去优化计算. 此外, 神经网络的稀疏矩阵能够以压缩稀疏行 (CSR) 和压缩稀疏列 (CSC) 两种方式存储在压缩格式[19]. 结构化压缩剪枝后的矩阵中非零值的位置是规则的, 而且稀疏矩阵的 CSR 格式中粗粒度稀疏性可以节省索引的存储开销, 易于硬件部署的实施.

表 2 现有神经网络压缩工具包

工具包	发布者	支持语言	支持框架	支持方法
NNI	微软	Python	Pytorch, MXNet, Tensorflow, Caffe2	剪枝和量化
Tensorflow Lite	谷歌	Java, Swift, Objective-C, C++, Python	Tensorflow	剪枝和量化
Distiller	Intel	Python	Pythorch	剪枝、量化、知识蒸馏
DNNDK	xilinx	C/C++	Caffe\ Tensorflow	剪枝和量化
PaddleSlim	百度	Python, C++	—	剪枝、量化、知识蒸馏
Pocketflow	腾讯	Python	Tensorflow	剪枝、量化、知识蒸馏

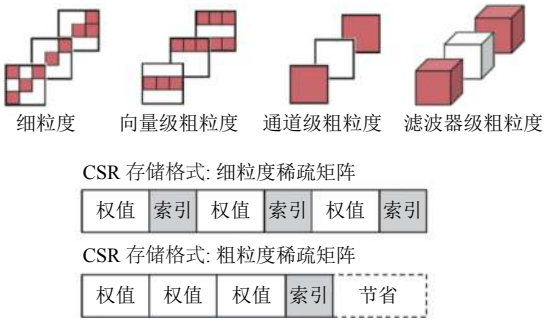

图 1 矩阵稀疏性的粒度与稀疏矩阵存储格式

为了高效的支持压缩后模型的硬件部署, 软硬件结合的压缩方法设计已成为当前发展的趋势之一, 已有诸多设计专用硬件处理架构的研究被发表. 为了将模型压缩方法的代表性算法 Deep compression[20] 部署到硬件平台, 文献 [21] 设计了高效的推理引擎 EIE, 比 CPU、GPU 和 Mobile GPU 的运行速度分别快189×、13×和307×; 文献 [22] 设计了专用硬件处理架构 ESE, 在进行压缩剪枝时进行多核并行的负载均衡, 进一步加快神经网络的推理速度.

2 神经网络模型压缩方法

本节将逐一介绍各类模型压缩方法的代表性算法与优缺点, 内容涵盖参数剪枝、低秩分解、参数量化和知识蒸馏 4 类主流压缩方法.

表 3 列举了 4 类模型压缩方法各自的特点. (1) 4 类压缩方法均适用于卷积层和全连接层. (2) 由于各类方法的压缩机制不同, 预训练模型对不同压缩方法的必要性不同, 其中剪枝方法对预训练模型的依赖性更高. (3) 传统的剪枝、量化与低秩分解算法需要在压缩后微调网络来补偿网络的精度损失, 而最新的进展中已出现不需要重训练的压缩方式, 大大减少计算成本. 例如 Tensorflow 中提供训练后量化方案, 在不重新训练模型的前提下, 只通过量化网络权重和输出激活图来压缩模型, 就能够达到与浮点型网络相接近的精度. (4) 剪枝方法的优势在于其精度损失小, 能够与其余 3 类压缩方法联合应用; 低秩分解的优点是支持端到端的训练, 但其分解操作的计算昂贵; 知识蒸馏可以使模型层级变浅, 降低推理时计算成本, 但其对模型的假设有时过于严格从而限制了应用.

各类压缩方法在特定任务及场景中表现出不同的压缩性能, 在选用压缩方法时可以依据应用需求来选择. 例如知识蒸馏方法适用于小型或者中型数据集上的应用, 由于压缩后的学生模型可以从教师模型中提取知识, 在数据集不大时, 也能取得鲁棒的性能; 剪枝和量化则更适合于要求模型表现稳定的应用场景或内存有限的设备, 因为这两种方法具有合理的压缩比, 精度损失小, 也能减小计算中内存使用量.

2.1 剪枝方法

剪枝方法依据一定标准来衡量网络结构的重要性, 通过移除不重要的网络结构来降低计算量和权重数量, 加速推理. 以基于稀疏约束的剪枝方法[14] 为代表, 在网络的优化目标中加入权重的稀疏正则项, 使得训练时网络的部分权重趋向于 0 值, 再将这些 0 值清除以实现剪枝. 最简单直接的衡量重要性的指标是权重的幅值. 文献 [14] 利用批归一化层的缩放因子γ来高效鉴别与裁剪不重要的通道, 并在损失函数中增加一个关于γ的正则项作约束.

表 3 各类神经网络压缩方法总结

方法	适用范围	重训练	基于预训练模型/从零开始训练	压缩机制
剪枝	卷积层/全连接层	需要	需要预训练模型	移除预训练模型的冗余权值, 减少模型参数量.
量化	卷积层/全连接层	需要	均可	不改变模型结构, 通过权重共享或权值精简实现.
低秩分解	卷积层/全连接层	需要	均可	将大尺寸参数张量分解成多个小尺寸张量的乘积, 减少计算量.
知识蒸馏	卷积层/全连接层	不需要	从零开始训练	复杂教师网络的软目标指导精简学生网络的训练, 实现知识迁移.

包括文献 [14] 在内的传统剪枝方法需要对压缩后网络模型进行微调来补偿压缩造成的准确率损失, 而微调既耗时又耗资源. 文献 [17] 提出了推理时剪枝方式, 在前向推理过程中进行压缩处理, 在剪枝后不再微调网络. 该文算法在进行通道裁剪之后, 直接通过最小均方误差得到特征重建误差最小化的新网络参数, 因而不需要再微调网络来恢复精度[17].

传统剪枝方法直接丢弃被裁减的网络结构, 使得网络容量随算法的迭代不断减少, 而且错误的裁剪所造成的精度损失无法通过微调弥补. 针对这一问题衍生出的动态剪枝算法, 保证被裁剪掉的权重在后续训练过程中仍会更新, 能够动态恢复裁剪部分或者对网络进行扩充. 文献 [23] 提出 SFP (Soft Filter Pruning) 方法, 在训练的每次迭代后进行滤波器剪枝, 并在下一次迭代中继续更新被裁剪部分的梯度.

由于神经网络中各层的稀疏性不同, 剪枝方法需要以预定义或自动设定的方式为每层设置适合的压缩比, 以减小压缩造成的准确率损失. 有研究工作提出渐变式压缩比设定方法[18], 预设每层压缩比与算法迭代次数的函数关系式, 逐渐地调整压缩比至目标压缩率. 现有方法更倾向于制定策略在剪枝过程中自动设定压缩比, 例如 AMC (AutoML for Model Compression) 方法[24], 根据不同需求 (如保证精度或限制计算量), 应用强化学习来学习每层最优的压缩比, 再通过基于幅度的通道剪枝压缩网络模型.

AMC 方法将自动机器学习引入剪枝方法, 减轻了人工调节神经网络超参数的压力, 但其剪枝操作是逐层执行的. 现有剪枝方法大多忽略层间的关联性, 逐层移除不重要的权重来压缩神经网络. 而最近的研究工作发现剪枝的本质是识别约束下最优的压缩网络结构[25], 而不是筛选每层中重要的权重. 已有剪枝方法[26] 借鉴神经网络架构搜索的算法来获取最优的压缩网络结构, 但在搜索空间设计和性能评估加速方面的优化是开放性问题.

2.2 低秩分解

低秩分解又称低秩近似, 利用卷积神经网络的参数张量和激活张量低秩且稀疏的特点, 将大尺寸张量分解成多个小尺寸张量的乘积, 即用若干个小张量对原张量进行估计, 减少推理时计算量[27]. 常见的张量分解有奇异值分解 (SVD)、Tucker 分解和 Canonical Polyadic (CP) 分解[28] 等, 这些分解方法能使压缩后模型在较少参数下保持高精度, 例如 CP 分解法[28] 在 AlexNet 上实现了 4 倍速度提升而只损失了 1% 的精度, SVD 分解后的 NIN[29] 在 CIFAR-10 数据集上达到的精度比原始 NIN 高 1%.

低秩分解方法在全连接层和卷积层的性能表现不同, 对全连接层的压缩效果更好. 针对全连接层低秩分解的研究[30], 仅对卷积网络最后一层全连接层进行分解就将参数减少 30–50%, 训练速度提升 30–50%, 语音识别的精度并没有下降.

低秩分解方法中, 保留多少秩关系到压缩后准确率与推理速度的权衡. 但是保留多少秩是不确定的, 文献 [31] 可以通过全连接层 5% 的权重值准确地预测出剩余 95% 的权重值. 对于面向卷积核的低秩分解方法, 秩的估计也是待优化的问题, 保留的秩多能保证高准确率, 但相应地加速效果下降. 文献 [29] 提出基于批归一化 (BN) 训练的 SVD 分解方法, 针对基于非线性最小二乘的 CP 分解法[28] 的最优秩很难求解且最优秩可能不存在的问题进行了优化, 保证最优秩总是存在, 并且能够训练层数大于 30 的深度神经网络. 文献 [32] 设计了基于 tucker 分解的一步式 (One-shot) 全网压缩方法, 首先利用变分贝叶斯矩阵分解进行一步式的秩选择后, 再进行核张量 tucker 分解和模型微调, 并且解决了 1×1卷积在硬件实现层面的问题, 降低了采用 inception 模块的 GoogleNet 网络在压缩后的功耗.

低秩分解后的网络模型在参数量压缩之外表现出两点提升, 一是关于局部最小值寻优[30], 经过低秩分解, 优化过程的寻优方向受限, 迭代次数减少, 寻优更有效率. 二是参数减少有助于降低神经网络过拟合的风险[29], 低秩分解后的网络模型具有更小的测试误差, 对新数

据集的泛化能力更好.

低秩分解方法并没有改变基础的卷积运算, 但由于受到几点缺陷的制约而不易于部署. 一是分解操作的计算成本昂贵. 二是目前的方法逐层执行, 无法进行全局参数压缩[27], 文献 [29] 虽然设计了低秩分解的全局优化器, 但其对网络的分解也是逐层单独进行. 三是分解后模型的收敛需要大量的训练.

2.3 参数量化

参数量化是指在不改变模型结构的情况下, 对模型参数或激活输出进行权值共享或权值精简[33] 的方法. 权值共享是指网络中的多个权重共用一个权值, 权值精简则指用较低位宽精度参数 (如 1-bit, 2-bit, 4-bit, 8-bit) 代替原始浮点型的参数 (如 32-bit).

权值共享常用的算法有 K-means 聚类算法和哈希共享. 文献 [20] 利用 K-means 算法将每一层权重矩阵聚类成若干个簇, 用同一簇的聚类中心值代替该簇的权重值, 因此只需存储每个簇的聚类中心值就能保存完整的模型参数, 以此来压缩模型的存储大小. 文献 [34] 设计了一种 HashNet, 利用哈希函数随机将网络连接权重分组到哈希桶, 每个哈希桶内的网络连接共享相同的权重参数, 该方法可显著减小模型体积, 且对结果精度影响较小.

权值精简方法是指利用低位宽精度参数 (如 8-bit) 代替原始的高位宽精度参数 (如 32-bit) 以达到模型压缩和计算加速的目的, 包括直接量化和模型重训两种模式. 直接量化是指对预训练得到的网络模型, 直接通过量化权重或 (和) 激活输出来缩减模型大小, 加快预测速度. 文献 [35] 设计了一种基于线性映射的直接量化方法, 通过 KL 散度寻找最佳裁剪阈值来计算量化参数以减小量化带来的精度损失. 目前常用的最佳裁剪阈值计算方法有最小均方差 (MMSE)、KL 散度, ACIQ[36] 等. 不同于 KL 的穷举搜索模式, 文献 [36] 提出的 ACIQ 方法计算速度快, 得到的裁剪阈值更优. 相对于直接量化, 模型重训则更复杂, 但它能在模型参数位宽精度更低时 (如 1 bit), 保证模型精度不受影响. 在模型重训中, 研究者主要侧重于训练方案的设计. 文献 [37] 提出了一种渐进式量化模式, 通过训练将浮点型神经网络模型转换为无损的低比特二进制模型, 并通过移位计算实现乘法过程, 方便模型在移动平台的部署和加速. 文献 [38] 提出了一种量化模式, 在前向传播时使用 8-bit 整型计算, 但在后向传播时仍使用 32-bit 浮点型计算损失参数以保证训练精度. 文献 [39] 则从训练最佳裁剪阈值的角度提出了 PACT 量化方法, 该方法将权重和激活都量化为 4-bits, 仍然能保持与全精度 (32-bit) 几乎相近的精度.

综上所述可见, 单一的权值共享方法重点在于对模型参数进行压缩, 无法加速推理端的计算过程, 很少被单独使用. 直接量化因为操作简单且方便部署, 得到了很多硬件厂商的青睐, 但因为精度损失的影响, 单一的直接量化无法做到较低位宽精度 (如 1-bit, 2-bit) 的量化. 模型重训的量化方法则能在保证模型精度的同时, 得到位宽精度更低的模型参数 (如 1-bit), 但该方法训练耗时较长且不易部署. 同时训练方案的设计, 以及如何有效地应用到 CPU、FPGA、ASIC 和 GPU 等硬件来加速训练过程, 也是重训方法的研究重点. 因此如何快速且更低位的对模型进行量化压缩, 同时保持模型精度, 是当前技术研究的核心方向. 如文献 [40] 在 ACIQ 方法[36] 的基础上, 通过对权重采用 K-means 聚类, 激活输出采用逐通道量化, 以及偏置误差补偿的方法来保证低位 (4-bit) 量化模型后的分类精度. 文献 [41] 则提出了一种基于无标签数据训练的网络压缩方法, 能在快速量化的同时保持模型精度. 文献 [42] 则从混合精度的角度出发, 提出了自动化的 HAQ 方法, 该方法能对不同硬件的性能进行自动识别以采用不同的低精度适配不同硬件.

2.4 知识蒸馏

知识蒸馏是指将训练好的复杂模型的“知识”迁移到一个结构简单的网络中, 或者通过简单网络去学习复杂模型的“知识”. Hinton[43] 首次提出了知识蒸馏的概念, 通过引入与教师网络相关的软目标作为总损失函数的一部分, 以引导学生网络的训练, 实现知识迁移的过程.

知识蒸馏的核心在于学生网络如何去学习教师网络以得到教师网络的“知识”. 文献 [44] 提出了一种基于空域注意力的知识迁移模式, 针对 CNN 网络, 将教师网络的注意力信息迁移给学生网络. 文献 [45] 则从学习网络层与层之间关系的角度进行知识蒸馏. 文献 [46] 通过学习教师网络和学生网络的样本间的相似度进行知识蒸馏. 文献 [47] 将知识迁移的过程看作学习教师-学生之间对应特征分布匹配的过程, 采用最大平均差异 MMD (Maximum Mean Discrepancy) 进行优化. 文献 [48] 从 KD 损失函数入手, 将可学习损失函数 GAN

引入到知识蒸馏框架中, 作者认为教师-学生网络就是学生对教师的模仿过程, 因此学生网络可看作一个生成器, 产生对于输入的 logits. 文献 [49] 尝试在大规模分布式计算环境下使用在线蒸馏的方法, 即分布式环境中的每个节点之间都可以互为教师和学生, 并且相互提取内在知识, 以提升其它节点的模型性能.

综上各种蒸馏方法可以看出, 当前很多的知识蒸馏方法都是基于各自的一种“知识”假设模型进行蒸馏, 因此可能存在知识学习的不全面性, 如何设置一种更加自动化的知识蒸馏方法, 是以后研究的重点. 另外随着硬件的发展, 文献 [49] 所提出的大规模分布式在线蒸馏方法也将会是发展趋势之一. 同时知识蒸馏方法与其它各种学习方法的结合也将带来新的发展, 如 GAN, 监督学习, 半监督学习, 以及弱监督学习等.

3 已有方法性能对比

现有文献提出了多种衡量比较其压缩性能的量化准则, 本节将对这些量化准则进行总结, 简述在评测模型压缩方法时常用的数据集, 并且对比分析了代表性压缩方法的压缩性能.

3.1 评价指标

模型压缩算法的评价指标通常涵盖准确率压缩 (或准确率损失)、参数量压缩、推理时延压缩 (或加速比)、MAC 量压缩、能耗压缩、索引空间压缩率. 各个指标的定义与计算如下:

(1) 准确率压缩率 r_A: 原始模型 M 的图像分类准确率A_{original}与压缩后模型 M^*的分类准确率$A_{\text{compressed}}$之比, 即$r_A = A_{\text{original}}/A_{\text{compressed}}$.

(2) 参数量压缩率 r_p: 压缩后模型 M^*的所有参数所占的内存开销$S_{\text{compressed}}$与原始模型 M 的所有参数所占的内存开销S_{original}之比, 即$r_p = S_{\text{compressed}}/S_{\text{original}}$.

(3) 时延压缩率 r_T (加速比): 存在两种定义方式, 一种是平均测试时间即推理时间的压缩比, 另外一种是每次迭代的平均训练时间的压缩比, 同样都是压缩后模型与原模型的时间比, 即$r_T = T_{\text{compressed}}/T_{\text{original}}$.

(4) MAC 量压缩率 r_c: 压缩后模型 M^*中所有的相乘累加操作数量$C_{\text{compressed}}$与原始模型 M 中所有的相乘累加操作数量C_{original}之比, 即$r_c = C_{\text{compressed}}/C_{\text{original}}$.

(5) 能耗压缩率 r_E: 压缩后模型 M^*中进行推理所消耗的能量$E_{\text{compressed}}$与原始模型 M 中进行推理所消耗的能量E_{original}之比, 即$r_E = E_{\text{compressed}}/E_{\text{original}}$.

(6) 索引空间压缩率 r_D: 压缩后模型 M^*中索引空间维度$D_{\text{compressed}}$与原始模型 M 中索引空间维度D_{original}之比, 即$r_D = D_{\text{compressed}}/D_{\text{original}}$.

有关文献 [7] 指出, 单一的评价指标不能够很好的评价压缩模型的性能, 由于这些指标并不是独立、不相关的, 因此其提出对上述所有评价指标进行平均加权, 全面、综合地评价压缩模型的性能. 同时, 该文献建议在不同资源限制的硬件平台中选用压缩方法时, 应考虑多项指标的综合结果作为选择的标准, 除了加权平均的方法之外还可以再进一步研究其他综合考量多项指标的方法.

3.2 常用评测数据集

MNIST、CIFAR 和 ImageNet 数据集是评测模型压缩方法在分类任务中性能的常用数据集[50]. 表 4 列举了 MNIST 数据集和 CIFAR 数据集的类别数和包含的图像数量. 用于小图像分类的 CIFAR 数据集分为 CIFAR-10 和 CIFAR-100 两个版本, CIFAR-100 数据集的 100 个类被分成 20 个超类, 每个图像都带有一个“精细”标签 (小类) 和一个“粗糙”标签 (超类). ImageNet 是一个大尺度图像数据集[6], 包含 1000 个类别彩色图像, 根据 WordNet 层次结构组织而成. ImageNet 可以测试分类任务及目标检测的准确率.

表 4 分类任务中常用数据集[50]

数据集	类别数	训练集规模	测试集规模	图像尺寸
MNIST	10	6×10^4	1×10^4	28×28
CIFAR-10	10	5×10^4	1×10^4	32×32
CIFAR-100	100	5×10^4	1×10^4	32×32
ImageNet	1000	1.2×10^6	1×10^5	—

除了分类数据集外, Pascal VOC 数据集和 MS COCO 数据集是常用的目标检测数据集. Pascal VOC 包含 VOC2007 和 VOC2012 两个版本. Pascal VOC 中 20 个类别图像的标注情况和标注出的对象实例数目如表 5 所示. MS COCO 数据集以场景理解为目标, 从复杂的日常场景中截取图像, 图像中的目标通过精确的分割进行位置的标定, 包含 91 类目标. 与 Pascal VOC 相比, MS COCO 数据集中小尺寸目标多, 单幅图片中目标多, 物体大多非中心分布, 更符合日常环境, 所以 MS COCO 检测难度更大.

表 5 目标检测任务中常用数据集

数据集	类数	训练集规模	对象实例	测试集规模
Pascal VOC 2007	20	5011	12 608	4952
Pascal VOC 2012	20	11 540	27 450	—
MS COCO 2014	91	123 287	1.156×10^6	40 775
MS COCO 2017	91	246 690	1.5×10^6	81 434

3.3 分类任务中的模型压缩

已有研究在移动端测试分析剪枝方法和低秩分解方法的代表性算法的压缩效果, 并分别评估低秩分解方法在全连接 (FC) 层和卷积 (CONV) 层的性能[7], 移动端的运行环境是 Xiaomi RedMi 3S (DRAM: 3 GB, Battery: 4100 mAh, MAC: 691.3 Mflops). 实验结果如表 6 所列举, 包括 12 层 AlexNet 网络在 CIFAR-10 数据集上的测试数据. 其中, 用于评价的剪枝方法是 deep compression[20], 而低秩分解方法选用的代表性算法是基于 SVD 分解的算法[51]. 该研究工作中, 剪枝方法被应用于第一个 FC 层, 减少了 40% 的 MAC 计算量; SVD 低秩分解方法分别作用于第一个 FC 层和第二个卷积层, 各自缩减了 20% 和 40% 的 MAC 计算量.

表 6 AlexNet 中模型压缩方法在移动端的性能对比[7]

评价指标	Baseline	剪枝[20]	低秩分解[7] (FC)	低秩分解[7] (CONV)
准确率Top5	82.12	83.79	77.16	60.32
参数压缩率$1/r_p$	1×	35×	1.23	1.03
时延压缩率$1/r_T$	1×	0.97	0.78	1.4

表 6 中参数压缩率的值等于原始网络的参数量与压缩后网络的参数量之比, 反映了压缩算法所取得的参数量压缩倍数. 由表中数据可见, 对于 AlexNet 剪枝算法取得了 35 倍的参数压缩率. 表中的时延压缩率等于原始网络的推理时延与压缩后网络的推理时延之比, 衡量网络前向处理时间的加速倍数. 这个指标的对比结果表明时延与 MAC 量或内存消耗量无直接关系, 而是由神经网络的计算与存储开销和设备 CPU 的动态使用情况联合影响的. 参数量和 MAC 量的减少, 不一定会带来时延的压缩, 在设计模型压缩算法时, 优化时延压缩率等直接指标比减少 MAC 计算量等间接指标所取得加速效果更好, 这一结论与其他研究工作[52] 的实验结果保持一致.

文献 [27] 还比较了模型压缩方法在 VGG 网络中的压缩效果, 在表 7 的结果中, 剪枝方法可以在获取较低模型准确率损失的同时达到 49 倍的参数压缩率, 高于基于 CP 分解的低秩分解方法的参数压缩率. 此外, 最初的深度神经网络会采用较大尺寸的卷积核, 例如 AlexNet 采用 11×11、5×5、3×3 卷积核, 而随着深度可分离卷积的出现, 越来越多的深度神经网络模型采用小尺寸的卷积核, 例如 ResNet 和 MobileNet 采用的 1×1 卷积核, 低秩分解方法对 1×1 卷积核的压缩无显著效果, 因而对于低秩分解方法的研究文献数呈下降趋势.

表 7 VGG 中模型压缩方法的性能对比[27]

评价指标	剪枝[20]:	低秩分解[27]:
准确率Top5	+0.41%	−0.29%
参数压缩率$1/r_p$	49×	2.75

3.4 识别任务中的模型压缩

现有工作还在移动端对参数剪枝方法、低秩分解等压缩方法在几种识别任务中的性能进行了测评[7], 包括: (1) 任务一, LeNet 在 MNIST 数据集上的数字识别; (2) 任务二, AlexNet 在 CIFAR-10 数据集上的图像识别; (3) 任务三, AlexNet 在 CIFAR-10 数据集上的图像识别; (4) 任务四, LeNet 在 UbiSound 数据集上的语音识别. 在文献 [7] 的实验结果中, 参数剪枝方法在任务一中性能优于低秩分解等方法, 可取得参数压缩率 $r_p = 0.21$, 推理时延压缩率$r_T = 0.44$, MAC 压缩率$r_c = 0.3$; 在任务三中将深度可分离卷积应用于 AlexNet 的轻量化方法呈现了最好的性能, 取得参数压缩率$r_p = 0.32$, 推理时延压缩率$r_T = 0.23$, MAC 压缩率$r_c = 0.13$; 而低秩分解方法在 4 个识别任务中均未取得最佳性能表现.

此外, 文献 [7] 在具有不同资源约束的移动设备端测试了各类压缩方法的性能, 表 8 给出了移动设备端的 DRAM、Cache、MAC 处理速率的设置情况以及性能表现最佳的压缩方法. 测试结果表明, 在 MAC 处理速率最低的 Device 1 上综合性能最佳的是参数剪枝方法 deep compression. 而且该文章指出, 没有一种压缩方法可以同时在精度损失、参数压缩率、时延压缩率、MAC 量压缩率和能量消耗压缩率等 5 个评价指标取得最优, 设计融合多类压缩算法的复杂方法可以集成各类算法的优势, 并突破各类算法的性能提升瓶颈.

表 8 不同移动设备端压缩方法性能对比[7]

设备编号	DRAM (GB)	Cache	MAC处理速率	性能最佳方法
Device 1	3	L1 (64 KB), L2 (2 MB)	691.3 Mflops	剪枝
Device 2	4	L1 (32 KB), L2 (1 MB)	3.87 Gflops	深度可分离卷积
Device 3	4	L1 (38 KB), L2 (2 MB)	2.51 Gflops	低秩分解

4 压缩技术展望

压缩技术是深度神经网络得以迅速发展和广泛应用的助推器, 还存在很多需要解决的问题. 就目前的研究重点来看, 这些问题基本都集中在网络参数上. 这些参数所要处理的大批量数据, 其中往往只有少许的关键特征信息是我们所关心的. 如何从海量的数据中提取出关键信息, 过滤掉冗余数据, 也是深度神经网络压缩技术所要面对的一个难点.

目前虽然各方研究者提出了多种算法和理论, 但是都有一定的适用范围或适用条件, 没有一种方法可以兼顾各种应用的特点. 而深度神经网络本身所能够支持的机器视觉任务种类将越来越多样化, 不再仅仅集中于某一种特定任务. 因此能够集成目标检测、目标跟踪、图像分割等多种任务于一体的模型压缩方法会发展成新的研究热点.

同时, 之前相对独立发展的各种压缩技术也将进行融合, 集成各个压缩方法的优势, 突破单个压缩方法的局限. 另一方面也可以将神经网络结构搜索 NAS 技术、自动调参技术等加入到模型压缩方法中, 实现自动化压缩.

人工智能中的模型压缩技术研究, 其最重要的参考对象就是人类大脑. 随着对人类大脑机理本质的认识逐步深入, 各种类脑芯片将会不断涌现. 人类大脑的自身机能将对神经网络压缩技术的发展产生深远影响, 将会提出效率更高、更为贴近人脑机能特点的压缩理论及算法, 应用于新型的人工智能行业.

5 总结

本文对神经网络压缩技术的进展进行了概述. 在总结深度神经网络的最新发展成果的基础上, 本文详细介绍了参数剪枝、低秩分解、参数量化和知识蒸馏这四种主要的神经网络压缩方法的原理, 并且分析了这四种方法各自的优缺点. 本文对已有的神经网络压缩方法进行了性能上的对比, 介绍了常用的压缩方法评价指标、常用来验证压缩方法性能的经典神经网络模型和数据集, 并总结了在不同移动设备的资源约束下模型压缩方法的性能. 除此之外, 本文还讨论了神经网络压缩加速领域的发展趋势和热点问题, 希望本文的总结工作能为模型压缩方法的研究发展提供一些参考与帮助.

参考文献

1 Lindholm E, Nickolls J, Oberman S, *et al*. NVIDIA tesla: A unified graphics and computing architecture. IEEE Micro, 2008, 28(2): 39–55. [doi: 10.1109/MM.2008.31]

2 Jouppi NP, Young C, Patil N, *et al*. In-datacenter performance analysis of a tensor processing unit. Proceedings of 2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture. Toronto, ON, Cananda. 2017. 1–12.

3 Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. Proceedings of the 25th International Conference on Neural Information Processing Systems. Red Hook, NY, USA. 2012. 1097–1105.

4 Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. Proceedings of the 3rd International Conference on Learning Representations. San Diego, CA, USA. 2015.

5 He KM, Zhang XY, Ren SQ, *et al*. Deep residual learning for image recognition. Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, NV, USA. 2016. 770–778.

6 Deng J, Dong W, Socher R, *et al*. ImageNet: A large-scale hierarchical image database. Proceedings of 2009 IEEE Conference on Computer Vision and Pattern Recognition. Miami, FL, USA. 2009. 248–255.

7 Nan KM, Liu SC, Du JZ, *et al*. Deep model compression for mobile platforms: A survey. Tsinghua Science and Technology, 2019, 24(6): 677–693. [doi: 10.26599/TST.2018.9010103]

8 Howard AG, Zhu ML, Chen B, *et al*. MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv: 1704.04861, 2017.

9 Sandler M, Howard AG, Zhu ML, *et al*. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. arXiv preprint arXiv: 1801.04381V2, 2018.

10 Zhang XY, Zhou XY, Lin MX, *et al*. ShuffleNet: An extremely efficient convolutional neural network for mobile devices. Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA. 2018. 6848–6856.
11 Ma NN, Zhang XY, Zheng HT, *et al*. ShuffleNet V2: Practical guidelines for efficient CNN architecture design. Proceedings of the 15th European Conference on Computer Vision. Munich, Germany. 2018. 122–138.
12 Li H, Kadav A, Durdanovic I, *et al*. Pruning filters for efficient ConvNets. arXiv preprint arXiv: 1608.08710, 2017.
13 Wen W, Wu CP, Wang YD, *et al*. Learning structured sparsity in deep neural networks. Proceedings of the 30th Conference on Neural Information Processing Systems. Barcelona, Spain. 2016.
14 Liu Z, Li JG, Shen ZQ, *et al*. Learning efficient convolutional networks through network slimming. Proceedings of 2017 IEEE International Conference on Computer Vision. Venice, Italy. 2017. 2755–2762.
15 Narang S, Elsen E, Diamos G, *et al*. Exploring sparsity in recurrent neural networks. arXiv preprint arXiv: 1704.05119, 2017.
16 Zhuang ZW, Tan MK, Zhuang BH, *et al*. Discrimination-aware channel pruning for deep neural networks. arXiv preprint arXiv: 1810.11809, 2018.
17 He YH, Zhang XY, Sun J. Channel pruning for accelerating very deep neural networks. Proceedings of 2017 IEEE International Conference on Computer Vision. Venice, Italy. 2017. 1398–1406.
18 Zhu MH, Gupta S. To prune, or not to prune: Exploring the efficacy of pruning for model compression. arXiv preprint arXiv: 1710.01878, 2018.
19 Sze V, Chen YH, Yang TJ, *et al*. Efficient processing of deep neural networks: A tutorial and survey. Proceedings of the IEEE, 2017, 105(12): 2295–2329. [doi: 10.1109/JPROC.2017.2761740]
20 Han S, Mao HZ, Dally WJ. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. arXiv preprint arXiv: 1510.00149, 2016.
21 Han S, Liu XY, Mao HZ, *et al*. EIE: Efficient inference engine on compressed deep neural network. Proceedings of 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture. Seoul, Republic of Korea. 2016. 243–254.
22 Han S, Kang JL, Mao HZ, *et al*. ESE: Efficient speech recognition engine with sparse LSTM on FPGA. Proceedings of the 2017 ACM/SIGDA International Symposium on Field. New York, NY, USA. 2016. 75–84.
23 He Y, Kang GL, Dong XY, *et al*. Soft filter pruning for accelerating deep convolutional neural networks. Proceedings of the 27th International Joint Conference on Artificial Intelligence. Melbourne, Australia. 2018. 2234–2240.
24 He YH, Lin J, Liu ZJ, *et al*. AMC: AutoML for model compression and acceleration on mobile devices. Proceedings of 15th European Conference on Computer Vision. Munich, Germany. 2018. 815–832.
25 Liu Z, Sun MJ, Zhou TH, *et al*. Rethinking the value of network pruning. arXiv preprint arXiv: 1810.05270, 2019.
26 Liu ZC, Mu HY, Zhang XY, *et al*. MetaPruning: Meta learning for automatic neural network channel. Proceedings of 2019 IEEE/CVF International Conference on Computer Vision. Seoul, Republic of Korea. 2019. 3295–3304.
27 Cheng Y, Wang D, Zhou P, *et al*. Model compression and acceleration for deep neural networks: The principles, progress, and challenges. IEEE Signal Processing Magazine, 2018, 35(1): 126–136. [doi: 10.1109/MSP.2017.2765695]
28 Lebedev V, Ganin Y, Rakhuba M, *et al*. Speeding-up convolutional neural networks using fine-tuned CP-decomposition. Proceedings of 3rd International Conference on Learning Representations. San Diego, CA, USA. 2015.
29 Tai C, Xiao T, Zhang Y, *et al*. Convolutional neural networks with low-rank regularization. Proceedings of 4th International Conference on Learning Representations. San Juan, Puerto Rico. 2016.
30 Sainath TN, Kingsbury B, Sindhwani V, *et al*. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. Vancouver, BC, Canada. 2013. 6655–6659.
31 Denil M, Shakibi B, Dinh L, *et al*. Predicting parameters in deep learning. Proceedings of the 26th International Conference on Neural Information Processing System. Red Hook, NY, USA. 2013. 2148–2156.
32 Kim YD, Park E, Yoo S, *et al*. Compression of deep convolutional neural networks for fast and low power mobile applications. Proceedings of 4th International Conference on Learning Representations. San Juan, Puerto Rico. 2016.
33 Krishnamoorthi R. Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv preprint arXiv: 1806.08342, 2018.
34 Chen WL, Wilson J, Tyree S, *et al*. Compressing neural

networks with the hashing trick. Proceedings of the 32nd International Conference on Machine Learning. Lile, France. 2015. 2285–2294.

35 Migacz S. 8-bit inference with TensorRT. http://on-demand.gputechconf.com/gtc/2017/presentation/s7310-8-bit-inference-with-tensorrt.pdf. (2017-05-08).

36 Banner R, Nahshan Y, Hoffer E, *et al*. ACIQ: Analytical clipping for integer quantization of neural networks. arXiv preprint arXiv: 1810.05723v1, 2019.

37 Zhou AJ, Yao AB, Guo YW, *et al*. Incremental network quantization: Towards lossless CNNs with low-precision weights. arXiv preprint arXiv: 1702.03044, 2017.

38 Nagel M, Van Baalen M, Blankevoort T, *et al*. Data-free quantization through weight equalization and bias correction. Proceedings of 2019 IEEE/CVF International Conference on Computer Vision. Seoul, Republic of Korea. 2019. 1325–1334.

39 Choi J, Wang Z, Venkataramani S, *et al*. PACT: Parameterized clipping activation for quantized neural networks. arXiv preprint arXiv: 1805.06085, 2018.

40 Banner R, Nahshan Y, Hoffer E, *et al*. Post-training 4-bit quantization of convolution networks for rapid-deployment. arXiv preprint arXiv: 1810.05723, 2019.

41 He XY, Cheng J. Learning compression from limited unlabeled data. Proceedings of 15th European Conference on Computer Vision. Munich, Germany. 2018. 778–795.

42 Wang K, Liu ZJ, Lin YJ, *et al*. HAQ: Hardware-aware automated quantization with mixed precision. Proceedings of 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach, CA, USA. 2019. 8604–8612.

43 Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv preprint arXiv: 1503.02531, 2014.

44 Zagoruyko S, Komodakis N. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. Proceedings of the 5th International Conference on Learning Representations (ICLR 2017). Toulon, France. 2017.

45 Yim J, Joo D, Bae J, *et al*. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, HI, USA. 2017. 7130–7138.

46 Chen YT, Wang NY, Zhang ZX. DarkRank: Accelerating deep metric learning via cross sample similarities transfer. Proceedings of 32nd AAAI Conference Artificial Intelligence. New Orleans, LA, USA. 2018. 2852–2859.

47 Huang ZH, Wang NY. Like what you like: Knowledge distill via neuron selectivity transfer. arXiv preprint arXiv: 1707.01219, 2017.

48 Xu Z, Hsu YC, Huang JW. Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks. arXiv preprint arXiv: 1709.00513, 2018.

49 Anil R, Pereyra G, Passos A, *et al*. Large scale distributed neural network training through online distillation. arXiv preprint arXiv: 1804.03235, 2018.

50 纪荣嵘, 林绍辉, 晁飞, 等. 深度神经网络压缩与加速综述. 计算机研究与发展, 2018, 55(9): 1871–1888. [doi: 10.7544/issn1000-1239.2018.20180129]

51 Lane ND, Bhattacharya S, Georgiev P, *et al*. DeepX: A software accelerator for low-power deep learning inference on mobile devices. 2016 Proceedings of 15th ACM/IEEE International Conference on Information Processing in Sensor Networks. Vienna, Austria. 2016. 1–12.

52 Yang TJ, Howard AG, Chen B, *et al*. Netadapt: Platform-aware neural network adaptation for mobile applications. Proceedings of 15th European Conference on Computer Vision. Munich, Germany. 2018. 285–300.