软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2021,32(11):3468−3481 [doi: 10.13328/j.cnki.jos.006057]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

自动化张量分解加速卷积神经网络*

宋冰冰[1], 张 浩[2,3], 吴子锋[2,3], 刘俊晖[2,3], 梁 宇[2,3], 周 维[2,3]

[1](云南大学 信息学院,云南 昆明 650504)

[2](云南大学 软件学院,云南 昆明 650504)

[3](云南大学 跨境网络空间安全工程研究中心,云南 昆明 650504)

通讯作者: 周维, E-mail: zwei@ynu.edu.cn

摘 要: 近年来,卷积神经网络(CNN)展现了强大的性能,被广泛应用到了众多领域.由于 CNN 参数数量庞大,且存储和计算能力需求高,其难以部署在资源受限设备上.因此,对 CNN 的压缩和加速成为一个迫切需要解决的问题.随着自动化机器学习(AutoML)的研究与发展,AutoML 对神经网络发展产生了深远的影响.受此启发,提出了基于参数估计和基于遗传算法的两种自动化加速卷积神经网络算法.该算法能够在给定精度损失范围内自动计算出最优的 CNN 加速模型,有效地解决了张量分解中,人工选择秩带来的误差问题,能够有效地提升 CNN 的压缩和加速效果.通过在 MNIST 和 CIFAR-10 数据集上的严格测试,与原网络相比,在 MNIST 数据集上准确率稍微下降了 0.35%,模型的运行时间获得了 4.1 倍的大幅提升;在 CIFAR-10 数据集上,准确率稍微下降了 5.13%,模型的运行时间获得了 0.8 倍的大幅提升.

关键词: 张量分解;卷积神经网络;自动化机器学习;神经网络压缩;神经网络加速

中图法分类号: TP183

中文引用格式: 宋冰冰,张浩,吴子锋,刘俊晖,梁宇,周维.自动化张量分解加速卷积神经网络.软件学报,2021,32(11):3468−3481. http://www.jos.org.cn/1000-9825/6057.htm

英文引用格式: Song BB, Zhang H, Wu ZF, Liu JH, Liang Y, Zhou W. Automated tensor decomposition to accelerate convolutional neural networks. Ruan Jian Xue Bao/Journal of Software, 2021,32(11):3468−3481 (in Chinese). http://www.jos.org.cn/1000-9825/6057.htm

Automated Tensor Decomposition to Accelerate Convolutional Neural Networks

SONG Bing-Bing[1], ZHANG Hao[2,3], WU Zi-Feng[2,3], LIU Jun-Hui[2,3], LIANG Yu[2,3], ZHOU Wei[2,3]

[1](School of Information Science and Engineering, Yunnan University, Kunming 650504, China)

[2](National Pilot School of Software, Yunnan University, Kunming 650504, China)

[3](Engineering Research Center of Cyberspace, Yunnan University, Kunming 650504, China)

Abstract: Recently, convolutional neural network (CNN) have demonstrated strong performance and are widely used in many fields. Due to the large number of CNN parameters and high storage and computing power requirements, it is difficult to deploy on resource-constrained devices. Therefore, compression and acceleration of CNN models have become an urgent problem to be solved. With the research and development of automatic machine learning (AutoML), AutoML has profoundly impacted the development of neural networks. Inspired by this, this study proposes two automated accelerated CNN algorithms based on parameter estimation and genetic algorithms, which can calculate the optimal accelerated CNN model within a given accuracy loss range, effectively solving the error caused by artificially selected rank in tensor decomposition. It can effectively improve the compression and acceleration effects of the convolutional neural network. By rigorous testing on the MNIST and CIFAR-10 data sets, the accuracy rate on the MNIST dataset is

* 基金项目: 国家自然科学基金(61762089, 61863036, 61663047)
Foundation item: National Natural Science Foundation of China (61762089, 61863036, 61663047)
收稿时间: 2019-11-01; 修改时间: 2020-02-05; 采用时间: 2020-04-16

© 中国科学院软件研究所 http://www.jos.org.cn

slightly reduced by 0.35% compared to the original network, and the running time of the model is greatly reduced by 4.1 times, the accuracy rate on the CIFAR-10 dataset dropped slightly by 5.13%, and the running time of the model was greatly decreased by 0.8 times.

Key words: tensor decomposition; convolutional neural network; automatic machine learning; neural network compression; neural network acceleration

人工智能领域近年来成为全世界的研究和应用热点.作为人工智能的一个重要研究分支,自 LeCun 等人[1]在1980年提出完备的卷积神经网络(convolutional neural network,简称CNN)以来,CNN获得了迅速的发展,并在计算机视觉[2]、自然语言处理、语音识别和医疗影像处理等领域取得了一系列显著的研究成果[3].

卷积结构是 CNN 的核心部件,通过这种巧妙的设计,卷积结构可以显著降低原始神经网络的计算复杂度,并减少训练必须的参数量.然而即便如此,现在常用的 CNN 网络中动辄几十万、上百万的参数量仍然十分庞大.随着任务的复杂程度越高,参数量还会进一步增加.这就造成 CNN 对计算量和计算速度的需求也随之日益增长,传统的 CPU 以及一些低端的 GPU 计算能力很难满足 CNN 对计算资源的需求.另一方面,近些年,随着智能终端的急速增长,CNN 部署在小型设备(例如个人笔记本、手机等设备)备受关注,然而 CNN 庞大的计算量需要消耗大量的计算资源,尤其在资源相对紧张的小型设备中,完成训练和预测任务尤为受限.因此,如何优化 CNN、提高资源利用率,成为迫切需要解决的问题[4].

针对这个难题,研究人员相继提出了参数修剪和共享、张量分解、知识蒸馏[5]等 CNN 优化方法.研究表明,CNN 模型参数存在很大的冗余[6].在张量分解加速与压缩加速 CNN 中,张量分解的秩选择对神经网络的压缩至关重要.近年来,随着 AutoML 方法的研究逐渐成熟,为张量分解中秩的选择提供了有效的参考方案.

本文从 AutoML 角度出发,提出了两种自动化张量分解加速 CNN 的算法,两种算法分别是基于参数估计的自动化加速卷积神经网络和基于遗传算法的自动化加速卷积神经网络,并能够在给定的精度损失范围内求出最优的 CNN 加速模型,解决了人工选择秩导致的繁杂工程量以及无法选取最优方案的问题.

1 相关工作

- 张量分解的应用领域

张量分解在计算机应用领域扮演着重要的角色,已经被运用在谱分析、非线性建模、通信与雷达处理、图像处理、网络检索、对数据进行降维、去噪或数据去重等领域.Welling 等人[7]提出一种定点算法,将非负矩阵分解扩展到任意阶的张量分解,对图像数据进行分解降维;Kim 等人[8]基于 Tucker 分解提出一种非负的 Tucker 分解,并应用到计算机视觉;Hazan 等人[9]在人脸识别中提出一种对局部特征分解的三阶张量分解算法.另外,在信号处理研究领域,Nion 等人[10]将 PARAFAC 分解应用在 MIMO 雷达中的多维谐波参数估计问题;Benetos 等人[11]使用非负矩阵分解方法设计了一个自动分类算法,对乐器声音进行分类.在推荐系统领域,Chen 等人[12]首次将张量分解(Tucker)应用于网页搜索,并提出了 CubeSVD 算法,大幅提升了网页搜索的性能;Rendle 等人[13]针对标签推荐提出一种新的数据解释方案,并利用张量分解(Tucker)来优化对级排序学习模型,降低了运行时间的同时,提升了标签推荐的质量;Xiong 等人[14]将张量用于建模时序销售数据,并提出了贝叶斯概率张量分解(CP),将贝叶斯概率模型引入张量分解,通过马尔可夫链蒙特卡洛方法进行求解.在基于上下文感知的推荐中,Karatzoglou 等人[15]将张量分解(Tucker)应用于显示反馈的推荐数据,在多个数据集上取得了较低的 MAE 值.Shi 等人[16]则将张量分解(CP)应用于隐式反馈的推荐数据上,采用优化 MAP 的方式训练列表级排序学习模型.

- 神经网络加速的研究

最优化脑损失(optimal brain damage)算法[17]和最优化脑手术(optimal brain surgeon)[18]剪枝策略的提出,开创了参数剪枝加速神经网络的研究领域.在之后的研究中,Srinivas 等人[19]直接构建并排序权重的显著性矩阵来删除不显著的权值节点;Han 等人[20,21]提出一种基于低值连接的删除策略;Lebedev 等人[22]在损失函数中加入结构化的稀疏项,利用结构化稀疏的函数删除小于设定阈值的滤波器.NVIDIA 公司的 Molchanov 等人[23]提出一种全局搜索显著性滤波的策略.在 2014 年,Gong 等人[24]使用向量量化技术对参数空间内对神经网络的权值进行量化.Gupta 等人[25]将权值量化到 16bit 大小.而后又出现了三元网络量化[26]和二值量化[27].在 2016 年,

© 中国科学院软件研究所 http://www.jos.org.cn

Chollet[28]在 Inception-V3 基础上提出了 Xception,卷积核更紧凑,提高了网络的计算效率.Google 公司的 Howard 等人[4]利用计算和存储更小的深度分割卷积替换原始的卷积滤波.Zhang 等人[29]使用组卷积和通道重排设计紧凑神经网络模型.在 2015 年,Hinton 等人[30]首次提出了知识蒸馏的压缩方法,利用复杂的教师网络学习先验知识来指导小型的学生网络.

- 张量分解加速神经网络的研究

Denil 等人[6]发现,在神经网络中,模型参数有很大的冗余性.并使用一部分的特征参数来预测剩余的参数,证明了神经网络的可压缩性.Rigamonti 等人[31]提出了不可分离滤波器近似为可分离的线性组合的方法.Denton 等人[32]根据卷积滤波中存在的冗余性,使用低秩近似和滤波聚类的方法减少所需的计算量,来加速卷积网络评估速度.Jaderberg 等人[33]将卷积滤波使用水平滤波和垂直滤波来替换,达到去除神经网络冗余性以及加速神经网络的目的.Lebedev 等人[34]提出对卷积滤波使用 CP 分解去除模型冗余性,以此加速 CNN.Kim 等人[35]对卷积滤波使用 Tucker 分解,并使用变分贝叶斯来选择秩.Girshick[36]在目标检测神经网络中使用了 SVD 分解方法加速全连接层.Tai 等人[37]提出引入批量正则化,从头训练低秩约束的 CNN.温伟等人[38]提出了强力正则化项,使更多的卷积核分布在低秩空间.

- AutoML

Automated machine learning(AutoML)的目的是自动地发掘并构造相关的特征,使得模型可以有最优的表现[39].与传统的机器学习相比,AutoML 可以极大地减少人工调整网络中超参数的工作量,并获得更高的准确率和泛化性能.AutoML 研究通过强化学习[40]、进化算法[41]以及梯度方法[42]等来做神经网络架构和超参数优化.He 等人[43]使用强化学习的方法自动化压缩加速神经网络.Liu 等人[44]提出一种新颖的元学习方法,用于对非常深的神经网络进行自动通道修剪.

受上述工作的启发,本文结合 AutoML 和张量分解加速神经网络设计了两种自动化张量分解的加速神经网络方法.该方法能够在 CNN 加速中自动化压缩加速 CNN,并得到最优性能的加速压缩模型,避免了人为因素对性能的影响.

2 基于张量分解的神经网络加速

现有主流深度学习平台 CNN 中的卷积操作均是以矩阵方式计算完成的,而卷积核的大小均通过人工凭借经验指定,这就难免网络计算过程中存在冗余的参数.在张量分解加速神经网络中,主要使用 CP 分解[34]与 Tucker 分解[35]加速卷积层.总体来说,可以分为两步:使用张量分解的方法分解并替换原始神经网络权重;通过反向传播的微调方法降低分解对神经网络精度的影响.

2.1 CP分解加速卷积层

CNN 中,卷积操作在整个网络中时间消耗占比较大,对卷积层分解能有效提升整个网络的运行效率.假设输入数据 X 为 3 阶张量,尺寸大小是 $I\times J\times S$,使用卷积核 K 进行卷积操作,最终输出张量 Y,卷积操作如公式(1)所示.

$$Y_{(x,y,t)}=\sum_{i=x-\delta}^{x+\delta}\sum_{j=y-\delta}^{y+\delta}\sum_{s=1}^{S}K_{(i-x+\alpha,j-y+\alpha,s,t)}X_{(i,j,s)} \tag{1}$$

卷积核 K 是大小为 $d\times d\times S\times T$ 的四维张量,维度 d 和 d 是空间维度,对应输入张量中的宽和高(I 和 J)卷积,S 是输入维度大小,T 是输出维度大小,其中,$\alpha=(d-1)\div 2$.四阶张量的 CP 分解如图 1 所示.

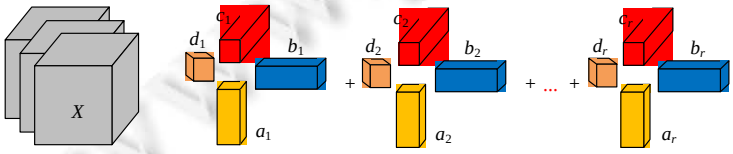


Fig.1 CP decomposition of four-order tensors

图 1 四阶张量的 CP 分解

© 中国科学院软件研究所 http://www.jos.org.cn

对卷积核 K 进行 CP 分解,如公式(2)所示.

$$K_{(i,j,s,t)}=\sum_{r=1}^{R}K^{x}_{(i-x+\alpha,r)}K^{y}_{(j-y+\alpha,r)}K^{s}_{(s,r)}K^{t}_{(t,r)} \tag{2}$$

K^x,K^y,K^s,K^t 表示 CP 分解后因子矩阵,张量(矩阵)大小分别是 $d_1\times R$,$d_2\times R$,$S\times R$,$T\times R$.将公式(2)代入卷积操作公式(1)中,得到 CP 分解替换原卷积公式(3).

$$Y_{(x,y,t)}=\sum_{r=1}^{R}K^{t}_{(t,r)}\left(\sum_{i=x-\delta}^{x+\delta}K^{x}_{(i-x+\alpha,r)}\left(\sum_{j=y-\delta}^{y+\delta}K^{y}_{(j-y+\alpha,r)}\left(\sum_{s=1}^{S}K^{s}_{(s,r)}X_{(i,j,s)}\right)\right)\right) \tag{3}$$

通过公式(3),CP 分解后,4 层小卷积操作如公式(4)~公式(7)所示.

$$Y^{s}_{(i,j,r)}=\sum_{s=1}^{S}K^{s}_{(s,r)}X_{(i,j,s)} \tag{4}$$

$$Y^{sy}_{(i,y,r)}=\sum_{j=y-\delta}^{y+\delta}K^{y}_{(j-y+\alpha,r)}Y^{s}_{(i,j,r)} \tag{5}$$

$$Y^{syx}_{(x,y,r)}=\sum_{j=x-\delta}^{x+\delta}K^{x}_{(i-x+\alpha,r)}Y^{sy}_{(i,y,r)} \tag{6}$$

$$Y_{(x,y,t)}=\sum_{r=1}^{R}K^{t}_{(t,r)}Y^{syx}_{(x,y,r)} \tag{7}$$

CP 分解加速卷积层具体流程如图 2 所示(以分解 1 层卷积操作为例).

- Step 1:对原神经网络的卷积核进行 CP 分解,分解成 4 个维度的因子矩阵形式.
- Step 2:将分解后的因子矩阵增加缺少的维度,组合成新卷积核的参数.
- Step 3:组合成新卷积核的参数赋值给新的卷积核.
- Step 4:使用新生成的 4 层卷积操作来代替原神经网络卷积层操作.

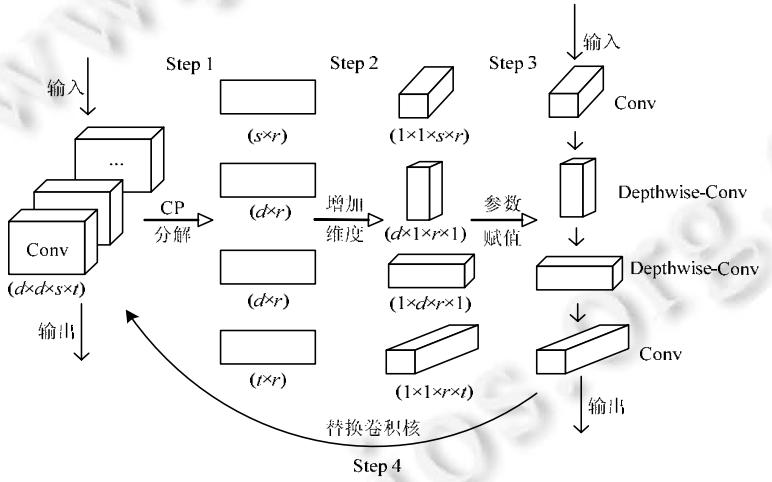


Fig.2 CP decomposition accelerate convolution layer

图 2 CP 分解加速卷积层

替换后的输出近似原输出 Y,如图 3 所示.

- 首先对输入 X 在输入通道上使用卷积核 Ks 卷积,卷积输出 Ys,将通道 S 降为 R,减少了输入通道,加快卷积速度.
- 其次,对通道 S 卷积后的输入 Ys 使用 Ky 卷积,同理再进行 Kx 卷积,卷积后输出 $Ysyx$.不同于原始卷积,原始卷积对空间维度宽高(xy)一起做卷积操作,这里是在通道数不变,分别对高(y)、宽(x)进行卷积.
- 最后,对 $Ysyx$ 使用 Kt 卷积核完成通道数 R 到 T 的卷积,加上初始卷积层的偏执 b.

© 中国科学院软件研究所 http://www.jos.org.cn

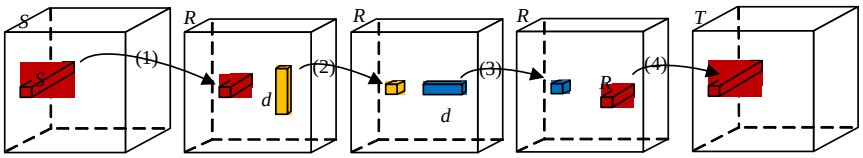


Fig.3 Convolution layer after CP decomposition

图 3 CP 分解后卷积层

使用 CP 分解对卷积核的近似替换,卷积核参数从 $X\times Y\times S\times T$ 变为$(S\times R+X\times R+Y\times R+R\times T)$.当 $R=R_1$ 时,使得 $R_1(S+X+Y+T)=X\times Y\times S\times T$;当 $R<R_1$ 时,使得网络参数降低,加快卷积操作效率.

2.2 Tucker分解加速卷积层

同理,对于 2D-卷积核 K 而言,K 是大小为 $D\times D\times S\times T$ 的四维张量,四阶张量的 CP 分解如图 4 所示.

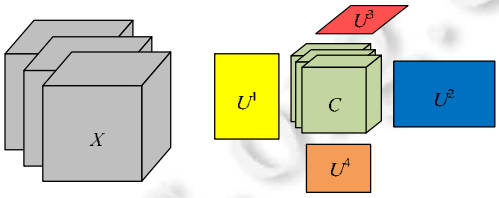


Fig.4 Tucker decomposition of four-order tensors

图 4 四阶张量的 Tucker 分解

对卷积核 K 进行 Tucker 分解,如公式(8)所示.

$$K_{(i,j,s,t)}=\sum_{r_1=1}^{R_1}\sum_{r_2=1}^{R_2}\sum_{r_3=1}^{R_3}\sum_{r_4=1}^{R_4}C_{(r_1,r_2,r_3,r_4)}U^1_{(i-x+\alpha,r_1)}U^2_{(j-y+\alpha,r_2)}U^3_{(s,r_3)}U^4_{(t,r_4)} \tag{8}$$

C 是大小为 $R_1\times R_2\times R_3\times R_4$ 的核张量,$U^1\sim U^4$ 分别是大小 $D\times R_1$,$D\times R_2$,$S\times R_3$,$T\times R_4$ 的因子矩阵.

在对卷积核进行 Tucker 分解时,不一定需要对所有的 mode 都进行分解,我们这里对卷积核分解不做 mode-1 和 mode-2 分解,如公式(9)所示.

$$K_{(i,j,s,t)}=\sum_{r_3=1}^{R_3}\sum_{r_4=1}^{R_4}C_{(i-x+\alpha,j-y+\alpha,r_3,r_4)}U^3_{(s,r_3)}U^4_{(t,r_4)} \tag{9}$$

将 Tucker 分解后的公式(9)代入卷积操作公式(1)中,得到如下 Tucker 分解卷积替换公式(10).

$$Y_{(x,y,t)}=\sum_{r4=1}^{R4}\left(\sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{r_3=1}^{R_3}\left(\sum_{s=1}^{S}U^3_{(s,r_3)}X_{(i,j,s)}\right)C_{(i-x+\alpha,j-y+\alpha,r_3,r_4)}\right)U^4_{(t,r_4)} \tag{10}$$

X 是大小为 $I\times J\times S$ 的输入张量,Y 表示经过卷积后的输出张量.将上式分开来得到如下 3 个步骤.

- Step 1:首先对输入张量 X 的输入通道进行卷积,将输入通道数 S 减少到 R_3,如公式(11)所示.

$$Y^s_{(i,j,r_3)}=\sum_{s=1}^{S}U^3_{(s,r_3)}X_{(i,j,s)} \tag{11}$$

- Step 2:在输入通道数为 R_3,输出通道为 R_4,进行空间卷积(对宽和高卷积),如公式(12)所示,不同于原始卷积在输入通道 S 和输出通道 T 进行卷积:

$$Y^{sxy}_{(x,y,r_3)}=\sum_{i=x-\alpha}^{x+\alpha}\sum_{j=y-\alpha}^{y+\alpha}\sum_{r_3=1}^{R_3}C_{(i-x+\alpha,j-y+\alpha,r_3,r_4)}Y^s_{(i,j,r_3)} \tag{12}$$

- Step 3:最后,将通道数 R_3 卷积输出为 T,使输出通道与初始卷积核的输出通道一致,如公式(13)所示.

$$Y_{(x,y,t)}=\sum_{r_4=1}^{R_4}U^4_{(t,r_4)}Y^{sxy}_{(x,y,r_3)} \tag{13}$$

最后,如果有偏置 b,再加上偏置 b,这样就完成了使用 Tucker 分解的方法替换加速卷积层.

© 中国科学院软件研究所 http://www.jos.org.cn

Tucker 分解加速卷积层流程如图 5 所示.

- Step 1:对原神经网络的卷积核进行 Tucker 分解,分解成一个四阶的 core 张量和两个因子矩阵;
- Step 2:将分解后的因子矩阵增加缺少的维度,组合成新卷积核的参数;
- Step 3:将新卷积核的参数赋值给新的卷积核;
- Step 4:使用新的 3 层小卷积操作来代替原神经网络卷积层操作.

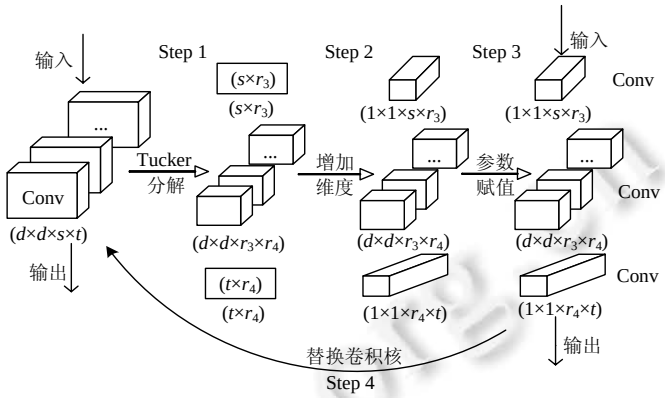


Fig.5 Tucker decomposition accelerate convolution layer

图 5 Tucker 分解加速卷积层

如图 6 所示,有 3 步卷积操作来替换初始的卷积操作:在第 1 步的卷积核中,卷积核大小为 $1\times1\times S\times R_3$,第 2 步的卷积核大小为 $D\times D\times R_3\times R_4$,第 3 步的卷积核大小为 $1\times1\times R_4\times T$.使用 Tucker 分解近似替换后的总卷积核参数为 $S\times R_3+D\times D\times R_3\times R_4+R_4\times T$,初始的卷积核参数为 $D\times D\times S\times T$,当 R_3,R_4 满足 $S\times R_3+D\times D\times R_3\times R_4+R_4\times T<D\times D\times S\times T$ 时,我们就达到了减少卷积核参数的目的,加速 CNN 和减少神经网络模型参数.

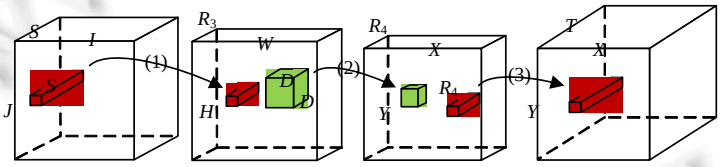


Fig.6 Convolution layer after Tucker decomposition

图 6 Tucker 分解后卷积层

2.3 微调卷积神经网络

在张量分解加速 CNN 中,张量分解是对全连接层和卷积层的近似分解并且会压缩网络模型参数,这样造成了模型的精度下降.因此需要使用训练数据集对张量分解加速的 CNN 进行微调,恢复 CNN 的精度.在本文中,使用反向传播的方法对压缩后的 CNN 进行微调,恢复 CNN 的精度.

3 自动化加速卷积神经网络

在张量分解加速神经网络研究领域,如何选择最优秩(精度下降最少、加速效果最好),一直都是张量分解加速神经网络的关注点.受 AutoML 的研究启发,本文基于张量分解加速 CNN 提出了两种 AutoACNN(自动化加速卷积神经网络 auto acceleration CNN)算法,该算法能在给定压缩后网络模型的容忍精度下(允许加速后模型精度损失多少),求最优加速压缩神经网络模型.

3.1 基于参数估计的自动化加速卷积神经网络

基于参数估计的自动化加速卷积神经网络算法的总体思路是:先估计卷积核的总参数,使其逼近最优加速

© 中国科学院软件研究所 http://www.jos.org.cn

网络模型近似所需的总参数,然后在近似参数区间内分组查找各个卷积核最优秩选择,最终返回满足容忍精度的最优压缩神经网络模型.

如图 7 与算法 1 所示,基于参数估计的自动化加速卷积神经网络算法总体分为 4 步.

- Step 1:相关变量的计算与初始化,包括输入模型的容忍精度、得到原模型的相关参数以及相关变量的计算.
- Step 2:张量分解压缩神经网络.详细见第 2 节基于张量分解的神经网络加速.
- Step 3:参数估计.参数估计是根据模型的容忍精度估计分解后 CNN 的总参数大小范围.在估计参数下确定秩中,首先估计的参数按照原卷积核参数占比多少分配,确定每个层分解后的估计参数,然后根据每个层的估计参数计算不同卷积核的分解秩,最后迭代更新的估计总参数.由公式(14)推出公式(15),得到不同层的分解秩 $Rank[i]$(其中,相关变量在算法步骤中定义):

$$\begin{cases} PE=\sum_{i=1}^{L} S_{[i]}\times Rank_{[i]} \\ \dfrac{Rank_{[1]}}{Num\_W_{[1]}}=\dfrac{Rank_{[2]}}{Num\_W_{[2]}}=...=\dfrac{Rank_{[L]}}{Num\_W_{[L]}}, Rank_{[i]}\in \mathbf{N}^{+} \\ 1\leqslant Rank_{[i]}<\dfrac{Num\_W_{[i]}}{S_{[i]}} \end{cases} \tag{14}$$

$$Rank_{[i]}=\max\left(1,\min\left(\left\lfloor\frac{Num\_W_{[i]}}{S_{[i]}}\right\rfloor,\left\lfloor\frac{PE\times Num\_W_{[i]}}{\sum_{j=1}^{L} Num\_W_{[j]}S_{[j]}}\right\rfloor\right)\right) \tag{15}$$

- Step4:分组查找.在近似参数区间内查找各个卷积核分解需要的秩:在估计的参数下,首先按照分组的思想,将上层卷积核分解时选择的不同秩划分成不同的组,而后对每个组内求下一层卷积核的秩的组合,并有序排列.如果有多层秩选择,继续向下分组.在其他层情况相同下,某一层分解后的参数越多,模型的精度越高.在一个组内的,如果选择大的秩组合不能满足容忍精度,那么小的秩组合也就更不能满足容忍精度.同理,小的秩组合能满足容忍精度,那么大的秩组合也就能满足容忍精度.以两层的分组查找为例,如图 8 所示:如果秩组合[1,*j*]不能满足分解后的模型容忍精度,那么这个组内其他秩组合也就不满足模型容忍精度,跳到下个组进行查找.当秩组合[1,*j*]的模型精度大于模型容忍精度时,存在秩组合[1,*i*]的模型精度大于模型容忍精度,并且秩组合[1,*i*–1]的模型精度小于模型容忍精度,那么秩组合[1,*i*]是组内最优秩组合.当[1,*i*]是第 1 组组内最优秩组合时,那么第 2 组的组内最优秩组合存在于第 2 组的 1~*i* 的区间中,那么第 2 组的查找区间为 1~*i*,以此类推.公式(16)是不同层的秩选择的限制条件,由公式(16)推出公式(17),$Max_Rank(i)$表示第 *i* 层能选择的最大秩,那么从 1 到 $Max_Rank[i]$都是第 *i* 层秩的选择方案:

$$\begin{cases} 1\leqslant Rank_{[i]}<\dfrac{Num\_W_{[i]}}{S_{[i]}} \\ \sum_{i=1}^{L} S_{[i]}\leqslant\sum_{i=1}^{L} Rank_{[i]}\times S_{[i]}\leqslant PE \end{cases}, Rank_{[i]}\in \mathbf{N}^{+} \tag{16}$$

$$Max\_Rank_{[i]}=\begin{cases} \min\left(\dfrac{Num\_W_{[i]}}{S_{[i]}},\max\left(1,\left\lfloor\dfrac{PE}{S_{[i]}}\right\rfloor\right)\right), & i=1 \\ \min\left(\dfrac{Num\_W_{[i]}}{S_{[i]}},\max\left(1,\left\lfloor\dfrac{PE-\sum_{j=1}^{i-1} Rank_{[j]}\times S_{[j]}}{S_{[i]}}\right\rfloor\right)\right), & i>1 \end{cases} \tag{17}$$

© 中国科学院软件研究所 http://www.jos.org.cn

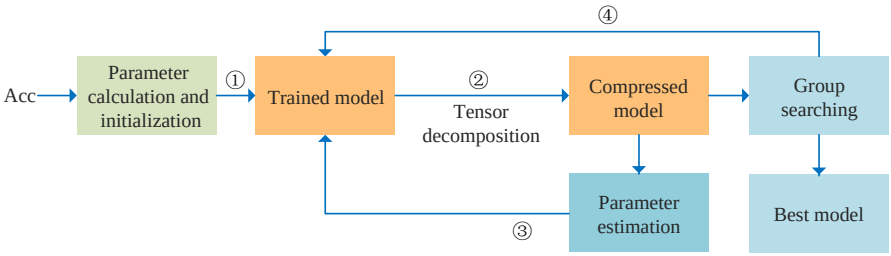


Fig.7 Auto acceleration CNN based on parameter estimation

图 7 基于参数估计的自动化加速 CNN

算法 1. 基于参数估计的自动化加速 CNN 算法.

Input:原模型卷积层参数 $W[i]$,i 表示第几个卷积核;张量分解加速后模型的容忍精度 Acc.

Output:输出最优秩选择方案 $Progm$.

```
L=W.lenth,S[i]=sum(W[i].shape),Num_W(i)=multiply(W[i].shape),PER=[1,sum(Num_W)]
while (PER[1]–PER[0])/sum(S)>2 do
    PE=(PER[1]+PER[0])/2
    Rank[1,…,L]   //由公式(26)计算每层秩
    if model_acc(Rank)>Acc then PER[1]=PE   //model_acc 返回分解后模型的准确率
    else PER[0]=PE
end
Combn[1,…,L]=0   //秩的组合
Group[1]=[1,…,Max_Rank[1]]   //Max_Rank 公式(28)
Level=1
while Level>0 do
    while Level!=L do
        if group[level].lenth==Combn[Level] then
            Level–=1
            break
        else
            Combn[Level]+=1
            Level+=1
            Group[Level]=[1,…,Max_Rank[Level]]   //Max_Rank 公式(28)
            if Level!=L then Combn[Level]=0
    if Level==L then
        time, Best_rank=top_one(Combn,Group)   //查找最优秩以及模型加速时间
        Combn[Level]=Best_rank
        Progm.add([time,Combn.copy])
    end
end
return Progm.min(key=time)
```

© 中国科学院软件研究所 http://www.jos.org.cn

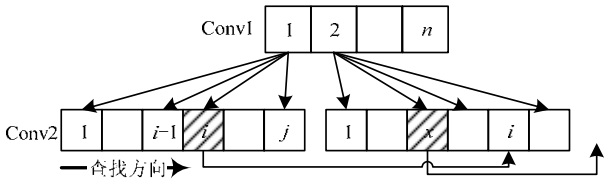


Fig.8 Two-layer convolution rank grouping lookup

图 8 两层卷积秩的分组查找

3.2 基于遗传算法的自动化加速卷积神经网络

在基于参数估计的自动化加速卷积神经网络算法工作中,第 3 步与第 4 步是搜寻最优秩选择并尽最大可能减少搜寻次数.受其工作启发,步骤 1 与步骤 2 不变,步骤 3 与步骤 4 使用遗传算法替换,最终返回满足容忍精度的最优压缩神经网络模型.算法流程图如图 9 所示.

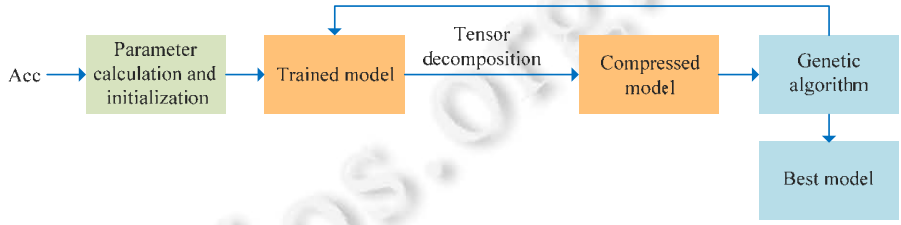


Fig.9 Auto acceleration CNN based on genetic algorithm

图 9 基于遗传算法的自动化加速 CNN

图 9 的步骤 3 遗传算法见算法 2.在 Selection 选择函数中,我们首先淘汰掉准确率低于容忍精度,而后以加速效果为适应度来选择下一代种群,经过迭代遗传,种群的基因表现趋近最优的秩组合.

算法 2. 基于遗传算法的自动化加速 CNN 算法.

Hyper Parameters:种群大小 P;变异数 M;交叉数 S;最大迭代次数 N.

Input:原模型卷积层参数 $W[i]$;张量分解加速后模型的容忍精度 Acc.

Output:输出最优秩选择方案 $Progm$.

1: $C=Constraints(W)$
2: $G_0=Random(P)$, s.t. C
3: **for** i=0:N **do**
4: $time, accuracy=Compressed\_Model(G_i)$
5: $G_i, time, accuracy=Selection(G_i, time, accuracy, Acc)$
6: $G_m=Mutation(G,M)$
7: $G_c=Crossover(G,S)$
8: $G_i=G_m+G_c$
9: **end**
10: $G_{top}=Top\_one(G_N, time, accuracy, Acc)$
11: $Progm=decode\_to\_rank(G_{top})$
12: **return** $Progm$

4 实 验

本实验基础网络选择 LeNet5[1]神经网络结构对 MNIST 数据集进行手写数字识别和对 CIFAR-10 数据集进行图像识别.我们在实验中首先探究了在张量分解时选择的秩对神经网络的影响,而后对本文提出的两个

© 中国科学院软件研究所 http://www.jos.org.cn

AutoACNN 算法、初始卷积神经网络 LeNet5 以及传统的张量分解加速 CNN 的效果对比.

如图 10 所示分别是 MNIST 与 CIFAR-10 两个数据集上的 CNN 模型以及使用张量分解后的卷积神经网络结构.CP 分解和 Tucker 分解分别对应图 6 和图 8 的分解流程图.DW_conv 是深度卷积层.

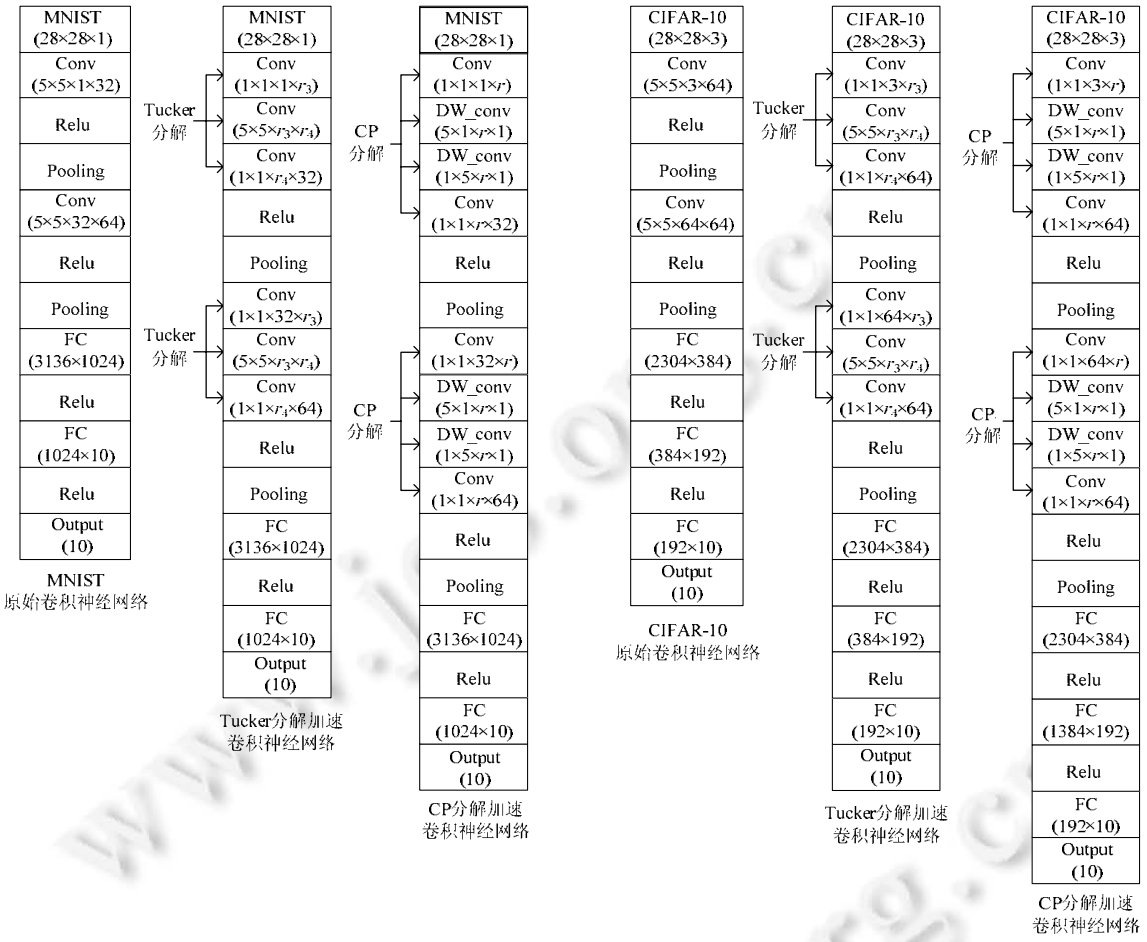


Fig.10 Original network structure and tensor decomposed network structure

图 10 原始网络结构与张量分解后网络结构

张量分解的秩(rank)对 CNN 加速效果的影响如图 11 所示.图 11 展示了不同数据集上,模型的加速效果随分解卷积核的秩的变化测得的实验数据.随着张量分解的秩越大,模型分解后的参数就越多,对模型准确率有一定的提升,但模型训练时间和运行时间也相应增加.因此在张量分解加速 CNN 中,张量分解的秩是基于经验选择,CNN 的加速效果受人为因素影响较大.

实验结果见表 1 和表 2,在 MNIST 数据集上输入容忍精度为 99.88%,在 CIFAR-10 数据集上输入容忍精度为 71%.实验评价指标包含 ACC、Time、Cov Floats 以及 Cov FLOPs:ACC 表示模型的准确率,Time 表示 CNN 运行的时间,Cov Floats 表示卷积层的参数,Cov FLOPs 表示卷积层的浮点数运算量.表中 e 表示科学计数法.实验对比方法包含无压缩、Cp、PE-Cp、Gene-Cp、Tucker、PE-Tucker、Gene-tucker.

- 无压缩表示原始卷积神经网络 LeNet5[1],未使用张量分解压缩.
- CP[34]表示使用 CP 分解的方式加速 CNN,详情见第 2 节,CP 分解加速卷积层.
- Tucker[35]表示使用 Tucker 分解的方式加速 CNN,详情见第 2 节,Tucker 分解加速卷积层.

© 中国科学院软件研究所 http://www.jos.org.cn

- PE-CP 与 PE-Tucker 是本文提出的基于参数估计的自动化加速卷积神经网络方法.
- Gene-CP 与 Gene-Tucker 是本文提出的基于遗传算法的自动化加速卷积神经网络方法.

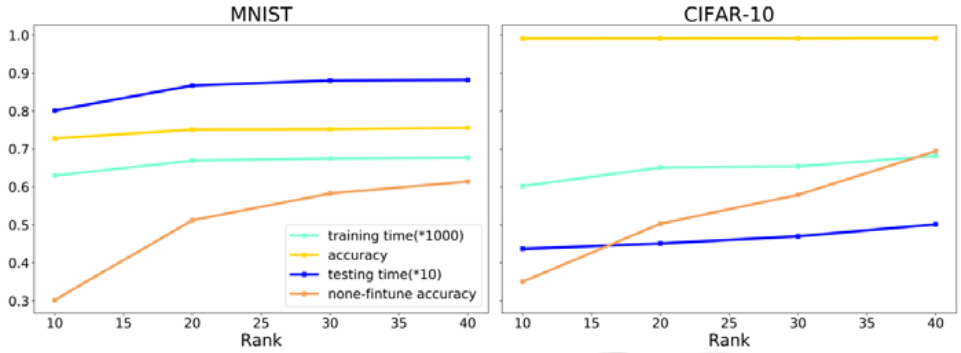


Fig.11 Model accelerated effect varied with rank

图 11 模型加速效果随秩的变化

Table 1 Comparison of different methods on MNSIT

表 1 MNIST 数据集上不同方法对比

压缩方法	ACC (%)	Time	Cov Floats	Cov FLOPs
无压缩	99.26	14.8	52 096	2.13e7
CP	98.93	3.2	1 056	7.54e5
PE-CP	98.91	2.9	758	5.7e5
Gene-CP	98.91	2.9	758	5.7e5
Tucker	98.82	3.9	5 184	1.24e6
PE-Tucker	98.82	3.6	1 651	5.23e5
Gene-tucker	98.88	3.8	2 586	7.18e5

Table 2 Comparison of different methods on CIFAR-10

表 2 CIFAR-10 数据集上不同方法对比

压缩方法	ACC (%)	Time	Cov Floats	Cov FLOPs
无压缩	76.31	8.8	107 328	3.5e7
CP	71.64	5.8	2 278	1.1e6
PE-CP	71.18	4.9	2 063	9.92e5
Gene-CP	71.75	5.0	2 201	1.02e6
Tucker	71.73	5.7	10 880	1.97e6
PE-Tucker	71.32	5.4	5 690	1.21e6
Gene-tucker	71.39	5.5	6 054	1.27e6

实验结果分析:

- 加速效果 Time 的实验结果显示:张量分解能显著提升 CNN 运行时间,达到加速 CNN 的目的.从 CP 分解与 Tucker 分解加速 CNN 的总体实验结果表现,CP 分解一定程度上优于 Tucker 分解.从传统的张量分解与 AutoACNN 相比,AutoACNN 的加速效果表现更好;除此之外,准确率 ACC 表现非常接近甚至更优.
- 从 ACC 与 Time 的实验结果显示:张量分解加速 CNN 能牺牲微量的准确率,获得大幅的时间性能提升.在 Mnist 数据集上最好的实验表现,CNN 精度损失 0.35%,获得 4.1 倍时间性能提升;在 Cifar10 数据集上最好的实验表现,CNN 精度损失 5.13%,获得 0.8 倍时间性能提升.相比于手工设计的 CP 分解与 Tucker 分解,自动化的模型压缩有更好的性能表现.
- 从 Cov Floats 的实验结果显示:张量分解后卷积层参数大幅降低,AutoACNN 压缩效果更优,最好的实验结果能得到 67 倍卷积层参数量压缩.
- 从 Cov FLOPs 的实验结果显示:张量分解后卷积层浮点数计算量大幅降低,AutoACNN 表现更优.最好

© 中国科学院软件研究所 http://www.jos.org.cn

的实验结果能得到37倍的卷积层浮点数计算量降低,浮点数计算量降低,在一定程度上反映出CNN运行时间降低与运算能耗的降低.

通过实验结果分析,本文提出的两种 AutoACNN 算法表现出了更好的加速性能与参数压缩、更少的卷积层浮点数运算量以及更少的精度损失.

5 总 结

本文利用张量分解的方法来加速卷积神经网络,分析了 CP 分解和 Tucker 分解加速卷积层,提出了自动化的张量分解加速 CNN.通过基于 MNIST 数据集和 CIFAR-10 数据集的实验,探究了本文设计的基于参数估计的自动化加速卷积神经网络和基于遗传算法的自动化加速卷积神经网络算法,两种算法能在给定的容忍精度下,自动求出最优加速性能的神经网络模型,解决了人工在选择秩的过程中导致的繁杂工程量以及不一定无法选取最优方案的问题.

通过实验可见:自动化的张量分解来加速和压缩卷积神经网络有着良好的表现,为自动化加速和压缩卷积神经网络提供可靠的解决方案.

References:

[1] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. IEEE Institute of Electrical and Electronics Engineers, 1998,86(11):2278−2324.

[2] Bai C, Huang L, Chen JN, Pan X, Chen SY. Optimization of deep convolutional neural network for large scale image classification. Ruan Jian Xue Bao/Journal of Software, 2018,29(4):1029−1038 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5404.htm [doi: 10.13328/j.cnki.jos.005404]

[3] Zhou FY, Jin LP, Dong J. A review of convolutional neural networks. Chinese Journal of Computers, 2017,40(6):1229−1251 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2017.01229]

[4] Howard AG, Zhu M, Chen B, Kalenichenko D, Wang WJ, Weyand T, Andreetto M, Adam H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.

[5] Ji RR, Lin SH, Chao F. A survey of deep neural network compression and acceleration. Journal of Computer Research and Development, 2018,55(9):1871−1888 (in Chinese with English abstract). [doi: 10.7544/issn1000-1239.2018.20180129]

[6] Denil M, Shakibi B, Dinh L, Ranzato M, DeFreitas N. Predicting parameters in deep learning. In: Proc. of the 2013 MIT Press Conf. on Neural Information Processing Systems (NIPS). 2013. 2148−2156.

[7] Welling M, Weber M. Positive tensor factorization. Pattern Recognition Letters, 2001,22(12):1255−1261. [doi: https://doi.org/10.1016/S0167-8655(01)00070-8]

[8] Kim YD, Choi S. Nonnegative Tucker decomposition. In: Proc. of the 2007 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2007). 2007.

[9] Hazan T, Polak S, Shashua A. Sparse image coding using a 3D non-negative tensor factorization. In: Proc. of the 2005 IEEE Conf. on 10th IEEE Int'l Conf. on Computer Vision (ICCV 2005). 2005. 50−57. [doi: 10.1109/ICCV.2005.228]

[10] Nion D, Sidiropoulos ND. Tensor algebra and multidimensional harmonic retrieval in signal processing for MIMO radar. IEEE Trans. on Signal Processing, 2010,58(11):5693−5705.

[11] Benetos E, Kotropoulos C, Lidy T, Rauber A. Testing supervised classifiers based on non-negative matrix factorization to musical instrument classification. In: Proc. of the 14th European Signal Processing Conf. 2006. 1−5.

[12] Chen Z, Lu Y. CubeSVD: A novel approach to personalized Web search*. In: Proc. of the 14th Int'l World Wide Web Conf. 2005. 382−390. [doi: 10.1145/1060745.1060803]

[13] Rendle S, Marinho LB, Nanopoulos A, Schmidt-Thieme L. Learning optimal ranking with tensor factorization for tag recommendation. In: Proc. of the KDD 2009. 2009. 727−736.

[14] Xiong L, Chen X, Huang T, Schneider J, Carbonell J. Temporal collaborative filtering with Bayesian probabilistic tensor factorization. In: Proc. of the SIAM Int'l Conf. on Data Mining (SDM). 2010. 211−222.

© 中国科学院软件研究所 http://www.jos.org.cn

[15] Karatzoglou A, Amatriain X, Baltrunas L, Oliver N. Multiverse recommendation: *N*-dimensional tensor factorization for context-aware collaborative filtering. In: Proc. of the 4th ACM Conf. on Recommender Systems. ACM, 2010. 79–86.

[16] Shi Y, Karatzoglou A, Baltrunas L, Larson M, Hanjalic A, Oliver N. TFMAP: Optimizing map for top-*N* context-aware recommendation. In: Proc. of the 35th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval. 2012. 155−164.

[17] LeCun Y, Denker JS, Solla S. Optimal Brain Damage. 1990. 598−605.

[18] Hassibi B, Stork DG. Second order derivatives for network pruning: Optimal brain surgeon. In: Proc. of the 1993 MIT Press Conf. on Neural Information Processing Systems (NIPS). 1993. 164−171.

[19] Srinivas S, Babu RV. Data-free parameter pruning for deep neural networks. arXiv preprint arXiv:1507.06149, 2015.

[20] Han S, Mao H, Dally WJ. Deep compression: Compressing deep neural networks with pruning trained quantization and Huffman coding. arXiv preprint arXiv:1510.00149, 2015.

[21] Han S, Pool J, Tran J, Dally W. Learning both weights and connections for efficient neural network. In: Proc. of the 2015 MIT Press Conf. on Neural Information Processing Systems (NIPS). Cambridge: MIT Press, 2015. 1135−1143.

[22] Lebedev V, Lempitsky V. Fast ConvNets using group-wise brain damage. In: Proc. of the 2016 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2016). Piscataway: IEEE, 2016. 2554−2564.

[23] Molchanov P, Tyree S, Karras T, Aila T, Kautz J. Pruning convolutional neural networks for resource efficient inference. arXiv preprint arXiv:1611.06440, 2017.

[24] Gong Y, Liu L, Yang M, Bourdev L. Compressing deep convolutional networks using vector quantization. arXiv preprint arXiv: 1412.6115, 2014.

[25] Gupta S, Agrawal A, Gopalakrishnan K, Narayanan P. Deep learning with limited numerical precision. In: Proc. of the 32nd Int'l Conf. on Machine Learning (ICML). 2015. 1737−1746.

[26] Li F, Zhang B, Liu B. Ternary weight networks. arXiv preprint arXiv:1605.04711, 2016.

[27] Courbariaux M, Bengio Y, David J. BinaryConnect: Training deep neural networks with binary weights during propagations. In: Proc. of the 2015 MIT Press Conf. on Neural Information Processing Systems (NIPS). 2015. 3123−3131.

[28] Chollet F. Xception: Deep learning with depthwise separable convolutions. In: Proc. of the 2017 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 2017. 1800−1807.

[29] Zhang X, Zhou X, Lin M, Sun J. ShuffleNet: An extremely efficient convolutional neural network for mobile devices. In: Proc. of the 2018 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 2018. 6848−6856.

[30] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015.

[31] Roberto R, Sironi A, Lepetit V, Fua P. Learning separable filters. In: Proc. of the 2013 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). 2013. 2754−2761.

[32] Denton E, Zaremba W, Bruna J. Exploiting linear structure within convolutional networks for efficient evaluation. In: Proc. of the 2014 MIT Press Conf. on Neural Information Processing Systems (NIPS). 2014. 1269−1277.

[33] Jaderberg M, Vedaldi A, Zisserman A. Speeding up convolutional neural networks with low rank expansions. arXiv preprint arXiv:1405.3866, 2014.

[34] Lebedev V, Ganin Y, Rakhuba M, Oseledets I, Lempitsky V. Speeding-up convolutional neural networks using fine-tuned CP-decomposition. arXiv preprint arXiv:1412.6553, 2014.

[35] Kim YD, Park E, Yoo S, Choi T, Yang L, Shin DJ. Compression of deep convolutional neural networks for fast and low power mobile applications. Computer Science, 2015,71(2):576−584.

[36] Girshick R. Fast R-CNN. In: Proc. of the IEEE Int'l Conf. on Computer Vision (ICCV). 2015. 1440−1448.

[37] Tai C, Xiao T, Zhang Y, Wang XG, E WN. Convolutional neural networks with low-rank regularization. arXiv preprint arXiv: 1511.06067, 2016.

[38] Wen W, Xu C, Wu C, Wang YD, Chen YR, Li H. Coordinating filters for faster deep neural networks. In: Proc. of the 2017 IEEE Int'l Conf. on Computer Vision (ICCV). 2017. 658−666.

[39] Yao Q, Wang MS, Chen YQ, Dai WY, Li YF, Tu WW, Qiang Y, Yu Y. Taking human out of learning applications: A survey on automated machine learning. arXiv preprint arXiv:1810.13306, 2018.

© 中国科学院软件研究所 http://www.jos.org.cn

[40] Zoph B, Le QV. Neural architecture search with reinforcement learning. In: Proc. of the 5th Int'l Conf. on Learning Representations (ICLR). 2017.

[41] Zhang HL, Kiranyaz S, Gabbouj M. Finding better topologies for deep convolutional neural networks by evolution. arXiv preprint arXiv:1809.03242v1, 2018.

[42] Ma NN, Zhang XY , Zheng HT, Sun J. ShuffleNet V2: Practical guidelines for efficient CNN architecture design. In: Proc. of the European Conf. on Computer Vision (ECCV). 2018. 122−138.

[43] He YH, Lin J, Liu ZJ, Wang HR, Li LJ, Han S. AMC: AutoML for model compression and acceleration on mobile devices. In: Proc. of the European Conf. on Computer Vision (ECCV). 2018. 815−832.

[44] Liu ZC, Mu HY, Zhang XY, Guo ZC, Yang X, Cheng K, Sun J. MetaPruning: Meta learning for automatic neural network channel pruning. In: Proc of the International Conference on Computer Vision ICCV. 2019. 3295−3304.

附中文参考文献:

[2] 白琮,黄玲,陈佳楠,潘翔,陈胜勇.面向大规模图像分类的深度卷积神经网络优化.软件学报,2018,29(4):1029−1038. http://www.jos.org.cn/1000-9825/5404.htm [doi: 10.13328/j.cnki.jos.005404]

[3] 周飞燕,金林鹏,董军.卷积神经网络研究综述.计算机学报,2017,40(6):1229−1251. [doi: 10.11897/SP.J.1016.2017.01229]

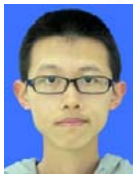
[5] 纪荣嵘,林绍辉,晁飞.深度神经网络压缩与加速综述.计算机研究与发展,2018,55(9):1871−1888. [doi: 10.7544/issn1000-1239.2018.20180129]

宋冰冰(1994－),男,博士生,CCF 学生会员,主要研究领域为深度学习,模型加速.

刘俊晖(1980－),男,博士,讲师,CCF 专业会员,主要研究领域为模型驱动开发,深度学习,计算机视觉.

张浩(1992－),男,硕士,主要研究领域为深度学习,生物信息学.

梁宇(1964－),男,教授,主要研究领域为网络技术,虚拟化,云计算.



吴子锋(1996－),男,硕士生,主要研究领域为模型压缩与分解.

周维(1974－),男,博士,教授,博士生导师,CCF 专业会员,主要研究领域为分布式处理.

© 中国科学院软件研究所 http://www.jos.org.cn