综述：软硬件协同的深度卷积神经网络剪枝方法

国防科技大学计算机学院 温冬 19023077
310-we-aaa-1@163.com

摘要

深度卷积神经网络在很多任务中取得了优秀的表现。然而卷积神经网络属于计算密集型、访存密集型的任务，对现有的硬件平台提出了很大的挑战。在 CPU 的硬件资源无法高效完成卷积神经网络计算和 GPU 过高的功耗与高昂的价格限制了卷积神经网络应用的背景下，FPGA 和 ASIC 这类可配置硬件平台成为新的选择。通过软硬件协同的深度卷积神经网络剪枝方法，在削减卷积神经网络存储规模与计算规模的同时，优化了其硬件友好特性，极大的扩展了模型的应用场景。

关键词：卷积神经网络、剪枝、软硬件协同

1. 引言

过去五年，卷积神经网络在多个计算机应用领域取得了极大的成功[1]-[3]。以 AlexNet[4]、VGG-16[5]、ResNet[6]等为代表的卷积神经网络在图像识别、人脸识别、目标检测、语义分割和自动驾驶等领域得到了广泛的应用并成为当前最主流、使用效果最好的方法。尽管在很多领域都成为了最佳算法，但卷积神经网络（CNN）的计算同时具备了访存密集和计算密集的特征限制了它在很多场景中的应用。CNN 的出现给传统的计算机硬件平台带来了诸多挑战，其访存密集的特点使得处理器对带宽的要求超出了现有内存接口的限制、其计算密集的特性导致 CNN 的部署将在嵌入式处理器和 CPU 上变得极其困难。上述特点使得 CNN 的训练和推理大多在图形处理器（GPU）上运行。CNN 中 90%的操作可以被规约为矩阵/向量乘累加（MAC）操作，GPU 平台拥有比 CPU 和嵌入式处理器更多的流处理单元和运算器，在处理 MAC 时可以拥有更高的并行度和计算效率，然而 GPU 昂贵的价格、巨大的功耗和芯片面积使得其功耗消费比很低，因此在一些对功耗和成本敏感的场合中，使用 GPU 来加速 CNN 任务是极其困难的事情。对 CNN 进行优化并将其运行在更廉价、更轻量和更快的硬件平台上成为了一个很有必要的工作。

对 CNN 的优化要同时结合其计算密集和访存密集的特点。CNN 往往带有庞大的网络结果和海量的参数，以 ResNet-152[6]神经网络为例，其拥有 150 层以上的卷积层和全连接层结构并拥有多达十亿个参数。如此庞大的网络结果和参数规模使得 CNN 要占用大量的存储空间，并且很难被放入访存速度最快的芯片片上存储器中，因为这些存储器往往昂贵而且空间受限。此外，每个卷积层的都要按照卷积-池化-激活-归一化的顺序执行相应的操作，巨大的网络规模带来了巨大的计算规模，限制了其在硬件上的计算与应用。本质上，CNN 的计算密集和访存密集是同一个问题的两个方面，是由卷积计算模式和 CNN 黑箱性质决定的。因此想要对 CNN 进行优化，就必须从其网络结构或参数特征上进行突破。

谷歌提出的 DenseNet[7]首次针对 CNN 进行了结构上的优化。CNN 虽然拥有海量的参数参与计算，但是大多数参数的绝对值都很小，只有小部分绝对值大的参数才对网络性能和最终结果拥有决定性的影响。基于此现象，谷歌发布了网络结构精炼、参数数

值分布更大的网络 DenseNet。其实验结果证明 DenseNet 凭借更稠密的参数，完全可以使用小得多的网络规模和计算规模达到与更大规模网络相同的效果, 从而让手机设备也可以部署使用 CNN 算法的应用。但随着时间的推移，DenseNet 的实际性能已经难以跟上当下最先进的 CNN 模型，比如 AttentionNet[8]。另一方面，[9]首次提出了知识蒸馏的概念，引入了“教师-学生”模型，利用强化学习的理念来训练出一个结构更小的 CNN 网络。然而知识蒸馏方法在训练阶段需要同时维持对教师模型和学生模型的更新，因此比普通 CNN 花费了更多算力和时间，且强化学习并不能保证一定可以得到一个效果较好的精炼模型。VGG-16[5]和 ResNet[6]的成功证明了 CNN 的网络规模对其性能有着决定性的作用，因此优化 CNN 网络规模、修改其结构的做法会必不可少的带来较大的性能损失。

在不修改 CNN 结果的前提下，对其参数特征进行改动与优化成为一条更可行、代价更小的道路。量化与剪枝是该思路下最常见的两种方法。量化方法通过将原 CNN 中的浮点参数与数据近似化为定点数据来优化计算。相比浮点数据格式，顶点数据格式虽然损失了部分运算精度，但拥有更小的存储空间、更少的计算周期数和计算资源需求量，直接减少了 CNN 模型对访存和计算资源的需求。此外，在神经网络海量参数的黑箱计算模式下，完全可以通过重训练的方法使量化方法几乎不损失任何模型性能[10]。但是量化方法并不能减少中间数据的规模[13]、并不能减少计算次数，因此简单的使用量化方法只适合一些规模较小的模型[11]，在处理规模较大的 CNN 时难以取得好的效果。

在 CNN 模型中，ReLU[12]是最常用的激活函数。在 ReLU 函数的作用下，CNN 的输出会变的越来越稀疏。剪枝方法即利用了 CNN 的稀疏特性，通过对权值的稀疏化：将参数中不符合某些条件的元素设为 0，使得 CNN 计算变成了稀疏矩阵乘稀疏矩阵的传统问题。相较于原来的存储格式，采用 CSC 数据格式存储的稀疏矩阵不仅不会改变任何数值，还能大大节省存储空间，极大的缓解了 CNN 访存密集的问题。此外，稀疏矩阵中大量的 0 元素和 0 计算被跳过，减少了计算次数、计算资源和运行功耗，提高了计算速度，成为现在 CNN 模型在硬件平台上部署的一种主流优化方法。以 CSC 为代表的经典稀疏矩阵格式虽然压缩了数据中的 0 元素并对非零元素进行了编码，但这也导致了在计算时需要对该格式的数据进行额外的解码。这一解码步骤往往包含较多的控制语句与跳转语句，会导致硬件的计算效率降低，因此设计软硬件协同的 CNN 剪枝方法成为现在学术界与工业界的焦点之一。

本文的主要内容为：

1. 介绍经典的 CNN 剪枝方法及其特点；
2. 介绍适合 CNN 剪枝模型运行的硬件平台及其特点；
3. 介绍几种较有新意的软硬件协同剪枝方法；

2. 背景

以 CPU 和 MCU 为代表的硬件平台往往是一种通用的、性能均衡的处理器。这种运算器通过精巧的多发射和乱序处理设计，可以以较高的效率运行经典的、满足各种普通需求的应用程序。现在的高端处理器往往有很大容量的缓存(Cache)、性能很高的分支预测器和很高的分支跳转运算性能，但由于需求和成本的考虑，算术运算部件的数量却很少。CNN 高计算密度的特点使得 CPU 和 MCU 已经很难应对此类需求，CNN 模型逐渐被移动到 GPU 上。

GPU 最初是针对图形、图像、视频流和动画游戏渲染的需求进行设计的。图像视频相关的操作中包含有大量矩阵向量乘计算，而分支跳转与控制指令却少得多[14]。矩阵向量乘操作在算法层面上就具有很高的并行度，因此对这类运算的加速要求硬件具有较

多的计算部件而不需要考虑跳转指令对程序的影响。上述特点导致了 GPU 在设计上具备很高的峰值计算性能和浮点运算部件，拥有完备且性能高度优化的矩阵操作算子库。但增加的计算部件同时带来了惊人的功耗和芯片设计面积。巨大的运行功耗导致 GPU 的设计不得不考虑散热成本，巨大的芯片设计面积也不可避免的导致流片良品率降低。这些因素影响着 GPU 的设计与生产成本。尽管 CNN 模型可以高效的利用 GPU 上的高性能运算器和算法库，但其 GPU 高昂的价格仍然严重制约 CNN 模型的应用。

CPU 和 GPU 在部署 CNN 时都面临挑战，考虑使用其他计算平台就成为新的选择。在部署 CNN 算法时，现场可编辑逻辑部件（FPGA）和专用集成电路（ASIC）成为主流的硬件平台。FPGA 和 ASIC 都可以针对特定算法进行优化设计，通过采用可配置的逻辑设计与模块化的开发方法使得 FPGA 与 ASIC 很适合加速或适配一些在 CPU 和 GPU 上处理效率不高的应用程序。目前，语音识别、碱基对匹配等许多算法已经完成在 FPGA 和 ASIC 上的专用适配并取得了很好的效果[11][15]。此外，CNN 模型，特别是 CNN 剪枝模型，由于其网络参数结构的高度异化或不规则，尤其适合在 FPGA 与 ASIC 这种可配置硬件平台上运行[16]-[20]。

3. 理论

剪枝方法，从其剪枝对象上来看，可以分为结构化剪枝和非结构化剪枝，也被称作粗粒度剪枝和细粒度剪枝。而结构化剪枝从其剪枝单元上来看，又可以分为通道剪枝、滤波器剪枝和形状剪枝。

3.1 非结构化剪枝

非结构化剪枝为最早被提出的一种剪枝方法，他的特点是不受网络结构的约束，对所有结构中的所有参数进行全局约束，如图 1 所示。非结构化剪枝将所有不符合预设调参的参数设置为 0.因为该方法的基本剪枝对象为参数元素，所以非结构化剪枝又称作细粒度剪枝。

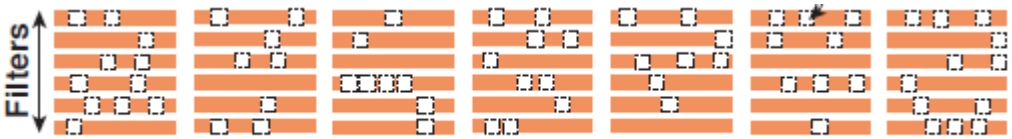

图 1 非结构化剪枝

非结构化剪枝的优势在于很好的保持原 CNN 网络的精度性能。由于其对参数进行了全局约束，因此该方法往往能保持全局重要的参数，进而保持了 CNN 模型性能的稳定。但该方法的缺点也同样明显，由于全局剪枝导致的零元素分布没有规律性，细粒度剪枝后的 CNN 权重的稀疏度分布不均匀、不规则，这种不规则的数据格式会对访存数据通道造成较大的性能影响，因为现在的 DDR 通道往往在访存规则连续时才可以发挥更大性能。此外稀疏度不均会导致硬件平台内的稀疏运算器负载不均衡，由此带来的短板效应会进一步制约硬件资源利用率和运行加速比。

3.2 结构化剪枝

结构化剪枝中的通道剪枝和滤波器剪枝的剪枝对象例较非结构化剪枝粒度更大，往往是一个输入通道或是整个滤波器中的所有权值，因此结构化剪枝又往往被称作粗粒度剪枝。形状剪枝针对的是滤波器中的某些立方块，但最先进的 CNN 模型中卷积核的尺寸往往为 3x3 或 1x1，在这种情况下形状剪枝更接近细粒度剪枝，所以本文对此不作讨论。

结构化剪枝的两种方法如图 2 所示。由于剪掉了整个输入通道或是滤波器，因此粗粒度剪枝剪掉的参数更多，具有更高的压缩比。由图 2 中我们可以看出，该方法剪掉的

都是很规则的模型结构，在访存时不会导致性能下降，因此该方法是对数据通信更加友好的一种方法。此外，规则的剪枝也有利于编译器或是硬件调度方法设计出更加均衡的计算负载流，提高了整体的硬件资源利用率。但是相对于细粒度剪枝，该剪枝方法的粒度较粗，往往会因为权重的位置而剪掉一些具有重要价值的参数元素，带来了更多的精度性能损失。

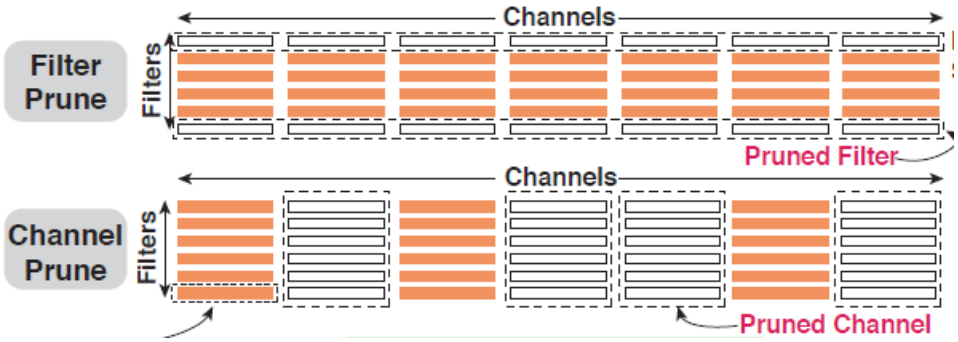

图 2 结构化剪枝方法

由上述内容我们可以看出，不论是结构化剪枝还是非结构化剪枝，都有着各自的优势和短板。学术界和工程界需要更新、更好的方法来解决 CNN 剪枝的软硬件协同问题。

4. 方法

为了解决传统 CNN 剪枝理论中软件性能和硬件性能无法兼得的障碍，[21]-[24]提出了较新颖的软硬件协同 CNN 剪枝方法。

4.1 PCONV

PCONV[21]的主要创新点为 1）充分结合了细粒度剪枝和粗粒度剪枝两种方法的优点，提出了一种折衷的方案：模式剪枝(Pattern Pruning)；2）该方法利用卷积的数学原理精心设计了自己的模式剪枝剪枝方案，在保证了模型硬件友好的前提下，很好的保持了原有 CNN 的模型精度。[25]在 PCONV 的基础上实现了基于 FPGA 的剪枝 CNN 加速器。

4.2 Low-rank Pruning

低秩分解剪枝(Low-rank Pruning)[22]-[23]是一种数学性很强的 CNN 剪枝方法。每个卷积核都是一个固定维度的矩阵，通过对该矩阵进行低秩分解，可以把其分解为若干小矩阵。考虑到神经网络参数的海量性和大部分参数的信息价值较低的性质，低秩分解方法可以有效的保留每个矩阵中的特征值，进而保存下该卷积核中最有剪枝的参数，其他参数即可通过剪枝丢弃。在输入通道维度上将分解过的卷积核排列累加就得到了低秩分解后的滤波器。此时滤波器的尺寸在低秩分解的作用下大大减少，而[22]-[23]中的实验证明，对小尺寸滤波器进行结构化剪枝可以更好的保持精度。因此该方法在使用结构化剪枝得到有利于硬件计算的参数结构后，同样可以拥有关键参数被保留带来的精度优势。

4.3 一种针对 CNN 剪枝的加速器数据流

[24]提出了一种针对 CNN 结构化剪枝方法的数据流。该数据流并没有对 CNN 结构化剪枝方法提出新的贡献，但该数据流可以高效的跳过稀疏数据中的 0 计算并高效的重用数据。事实上，卷积计算的性质决定了数据在 CNN 模型中存在很大的重用空间。该数据流设计了很精巧的数据移动阵列，有很高的数据重用性能，大大减少了访存需求。虽然该方法没有提出新颖的剪枝方法，也没有做到运算部件的负载均衡，但方法具有一定程度上的通用性，仍不失为一种较好的方法。

4.4 其他有特点的 CNN 软硬件协同剪枝方法

[26][27]是该领域中比较经典的工作，均在 ASIC 上进行设计、仿真和验证。同[24]的工作类似，[26]和[27]没有在软件剪枝方法做改进，类似的，他们提出了较完善的 CNN-硬件映射方案。[26]设计了一种输入静态映射方法。而[27]针对 CNN 的输出结构设计了 CNN-加速器的静态映射方案。两种方法都可以用最小的数据流代价来完成 CNN 剪枝网络在特定硬件平台上的部署，但静态映射方案仍然存在硬件利用率不高的问题，尤其是[26]中为了解决静态输入数据选通问题，设计了一个 256 路选通器，不仅消耗了 38%的额外硬件资源，还不能充分的解决负载均衡的问题。

[28]提出了一种比较新颖的资源限制变化场景下的 CNN 剪枝方法。[28]提出了一种增量 CNN 剪枝网络训练方法，先训练出一个压缩率相对较低，但精度最高的剪枝模型。随后渐进增量的提高压缩率、降低剪枝精度以适配不同硬件平台对存储器和计算量的限制。在渐进增量的环境下，精度最高的剪枝模型可以被视作后续所有模型的超集，因此和其他剪枝训练方法比，该方法的精度损失更小。此外[28]还提出了一种剪枝权重存储方案，可以在增加很小存储需求的代价下完成对不同压缩率的参数存储。

[29]主要基于[26]和[27]的静态方法做了改进，将 CNN 中的卷积层和全连接层(FC)分开设计硬件计算模式和硬件部件设计，但由于改动不大，仍可视为静态剪枝模型映射方法。

[30]主要针对 3D 环境下的 CNN 模型进行软硬件协同化剪枝。目前该领域的工作仍然较少。

5. 结论

在 CNN 网络结构规模不断扩大、参数量和计算量不断刷新记录的今天，基于软硬件协同的 CNN 剪枝方法无疑是一种优秀的 CNN 模型落地方案。

参考文献

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 1026–1034.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 580–587.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 779–788.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in Proc. 25th Int. Conf. Neural Inf. Process. Syst., NY, USA, pp. 1097–1105.

[5] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” 2014, arXiv:1409.1556. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770–778.

[7] Deng Z , Jiang Z , Lan R , et al. Image captioning using DenseNet network and adaptive attention[J]. Signal Processing Image Communication, 2020, 85:115836.

[8] Bush G , Shin L M . The Multi-Source Interference Task: an fMRI task that reliably activates the cingulo-frontal-parietal cognitive/attention network.[J]. Nature Protocol, 2006, 1(1):308-313.

[9] Fukuda T , Suzuki M , Kurata G , et al. Efficient Knowledge Distillation from an Ensemble of

Teachers[C]// Interspeech 2017. 2017.
[10] Bethge J , Bartz C , Yang H , et al. MeliusNet: Can Binary Neural Networks Achieve MobileNet-level Accuracy?[J]. arXiv, 2020.
[11] Dong W, Jingfei J, Jinwei X, et al. An Energy-efficient Speech Classification Convolution Neural Network Accelerator Based on FPGA and Quantization
[12] Yarotsky, Dmitry. Error bounds for approximations with deep ReLU networks[J]. Neural networks: the official journal of the International Neural Network Society, 2017, 94:103.
[13] Wang D, Xu K, Jia Q , et al. ABM-SpConv: A Novel Approach to FPGA-Based Acceleration of Convolutional Neural Network Inference[C]// the 56th Annual Design Automation Conference 2019. 2019.
14] Jin X, Chen S, Mao X. Computer-Generated Marbling Textures: A GPU-Based Design System[J]. IEEE Computer Graphics and Applications, 2007, 27(2):78-84.
[15] 邵清. CNN 全连接层 FPGA 硬件实现技术研究[D]. 2019.
[16] Luo T, Liu S , Li L , et al. DaDianNao: A Neural Network Supercomputer[J]. IEEE Transactions on Computers, 2017.
[17] David, Kanter. Google TPU Boosts Machine Learning[J]. Microprocessor report, 2017, 31(5):18-21.
[18] Wang Z, Xu K , Wu S , et al. Sparse-YOLO: Hardware/Software Co-Design of an FPGA Accelerator for YOLOv2[J]. IEEE Access, 2020, PP(99):1-1.
[19] 王巍, 安友伟, 黄展,等. 基于 CNN 的红外图像边缘检测算法的 FPGA 实现[J]. 光子学报, 2012, 041(011):1354-1358.
[20] J.J. Martínez, Toledo F J , J.M. Ferrández. New emulated discrete model of CNN architecture for FPGA and DSP applications[J]. 2003.
[21] Ma X , Guo F M , Niu W , et al. PCONV: The Missing but Desirable Sparsity in DNN Weight Pruning for Real-Time Execution on Mobile Devices[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2020, 34(4):5117-5124.
[22] Chen Z , Chen Z , Lin J , et al. Deep Neural Network Acceleration Based on Low-Rank Approximated Channel Pruning[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2020, 67(4):1232-1244.
[23] Chen Z , Lin J , Liu S , et al. Exploiting Weight-Level Sparsity in Channel Pruning with Low-Rank Approximation[C]// 2019 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2019.
[24] Zhu C , Huang K , Yang S , et al. An Efficient Hardware Accelerator for Structured Sparse Convolutional Neural Networks on FPGAs[J]. 2020.
[25] Li N , Liu L , Wei S , et al. A High-performance Inference Accelerator Exploiting Patterned Sparsity in CNNs[C]// 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE, 2020.
[26] cambroin XZhang S , Du Z , Zhang L , et al. Cambricon-X: An accelerator for sparse neural networks[C]// IEEE/ACM International Symposium on Microarchitecture. ACM, 2016.
[27] Parashar A , Rhu M , Mukkara A , et al. SCNN: An accelerator for compressed-sparse convolutional neural networks[C]// International Symposium. IEEE, 2017:27-40.
[28] Moon S , Byun Y , Park J , et al. Memory-Reduced Network Stacking for Edge-Level CNN Architecture With Structured Weight Pruning[J]. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 2019, PP(99):1-1.

[29] Kang H J . Accelerator-Aware Pruning for Convolutional Neural Networks[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2020, 30(7):2093-2103.
[30] Choi Y K , Cong J . Acceleration of EM-Based 3D CT Reconstruction Using FPGA[J]. IEEE transactions on biomedical circuits and systems, 2016, 10(3):754.