第 1 章　外文资料的书面翻译

PCONV：移动设备上实时运行的 DNN 权重剪枝中缺少但值得拥有的稀疏性

摘要

深层神经网络（Deep Neural Network，DNN）上的模型压缩技术已被广泛认为是在各种平台上实现加速的有效途径，而 DNN 权重剪枝是一种简单而有效的方法。当前有两种主流的剪枝方法，分别代表了剪枝规整性的两个极端：非结构化、细粒度的剪枝可以实现较高的稀疏性和准确性，但对硬件不太友好；结构化、粗粒度的剪枝利用了硬件高效的结构，但在剪枝率较高时准确性会降低。在本文中，我们介绍了 *PCONV*，其拥有一种新的稀疏性维度—— 粗粒度结构内的细粒度修剪模式。*PCONV* 包括两种类型的稀疏性：卷积核内部剪枝生成的稀疏卷积模式（Sparse Convolution Patterns，SCP）和卷积核之间剪枝生成的关联稀疏性。本质上来说，SCP 由于其特殊的视觉性质而提升了准确性，而关联稀疏性在提高剪枝率的同时平衡了滤波器的计算负载。为了部署 *PCONV*，我们开发了一种新颖的编译器辅助的 DNN 推理框架，并实时执行 *PCONV* 模型且不会影响准确性，这是先前工作无法实现的。我们的实验结果表明，*PCONV* 优于 TensorFlow-Lite，TVM 和 Alibaba Mobile Neural Network 这三种最先进的端到端 DNN 框架，其加速比分别达 39.2 倍，11.4 倍和 6.3 倍，且没有精度损失。移动设备上也可以实现大规模 DNN 的实时推断。

1.1　引言

深度神经网络（DNN）由于其高准确性、出色的可扩展性和自适应性Goodfellow et al (2016) 已成为机器学习应用中的基本要素和核心推动力。经过良好训练的 DNN 可以部署为多种任务的推理系统，如图像分类Krizhevsky, Sutskever, and Hinton (2012)、目标检测Ren et al (2015) 和自然语言处理Hinton, Deng, and Yu (2012)。然而，最先进的 DNN 模型，例如 VGG-16Simonyan and Zisserman (2014)、ResNet-50He et al (2016) 和 MobileNetHoward et al (2017)，涉及到高密度的计算和大量的内存开销，这使得在当前移动平台上实时运行推理系统遇到很大挑战。

近一段时间，高端移动平台正迅速取代台式机和笔记本电脑，成为可穿戴设备、视频流、无人驾驶、智能医疗设备等大量 DNN 应用的主要计算设备Philipp, Durr, and Rothermel (2011)Lane et al (2015)Boticki and So (2010)。开发实时 DNN 推理系统是人们所需要的，但仍然收到移动平台嵌入式处理器上有限的计算资源的限制。许多端到端移动 DNN 加速框架已经被开发出来，例如 TVMChen et al (2018)，TensorFlow-Lite（TFLite）Ten和阿里巴巴移动神经网络（MNN）Ali。但是，大规模 DNN 的推理时间（例如，在 Adreno 640 GPU 上使用 TVM 运行 VGG-16 网络需要 242ms 推理时间）仍然远远不能满足实时的要求。

为了减轻 DNN 庞大的计算量带来的挑战并实现实时推理的目标，有必要考虑算法层面的创新。人们研究了多种 DNN 模型压缩技术，其中 *权重剪枝*Han, Mao, and Dally (2015)Mao et al (2017) Dai, Yin, and Jha (2017)Wen et al (2016)He, Zhang, and Sun (2017) 可以显著地减小模型大小。关于*非结构化*的权重剪枝（细粒度）的早期工作Han, Mao, and Dally (2015) 在神经网络的任意位置修剪权重，最终得到稀疏的模型，并以压缩稀疏列（CSC）格式存储。压缩权重表示中的索引会在高度并行的架构上导致停顿或复杂的工作量，因此导致网络处理的吞吐量下降Han, Mao, and Dally (2015)Wen et al (2016)。而另一方面，*结构化*的权重剪枝Wen et al (2016)（粗粒度）对硬件更加友好。通过利用滤波器剪枝和通道剪枝，剪枝后的模型的形状更加规则，从而消除了权值索引的存储需求。但是人们发现，与非结构化的稀疏性相比，结构化剪枝对准确性的损害更大。

当前的迫切需要是找到一种新的粒度等级，既能满足搞准确性的要求，又能满足 DNN 模型结构的规则性。我们通过观察发现，非结构化的剪枝和结构化的剪枝在整个设计空间中走向了两个极端。两个缺失的关键是：(i) 找到一种新的、中等程度的稀疏等级，可以充分平衡细粒度模型的高精度和粗粒度模型的高规则性；(ii) 找到与之关联的（算法-编译器-硬件）优化框架，其可以无缝地弥合硬件效率与新的稀疏等级之间的隔阂。为了解决上述问题，本文提出了 *PCONV*，其包括：(a) 一个新的稀疏等级，它同时充分地利用了卷积内和卷积间的稀疏性，同时展现出了高精确度和高规则性，并揭示了以前设计空间中*未知的*这一点; (b) 一个*编译器辅助的 DNN 推理框架*，其充分利用了新的稀疏等级，并在移动设备上实现了实时 DNN 加速。

在 *PCONV* 中，我们将卷积核内部的剪枝称为*模式剪枝*，将卷积核之间的剪枝称为*关联剪枝*。对于模式剪枝，我们在每个卷积核中修剪固定数量的权值。不同于非结构化权值剪枝，模式剪枝在每个卷积核中得到相同的稀疏率和限定数目

的模式形状。本质上说，我们的设计模式与计算机视觉中的关键卷积滤波器概念有关，例如用于平滑的高斯滤波器，用于平滑和锐化的拉普拉斯-高斯滤波器。对于关联剪枝，关键一点是切断特定输入通道和输出通道的关联，这等价于移除相应的卷积核，让滤波器的“长度”比原模型更短。通过关联剪枝，我们进一步提高了压缩率，并提供更大的 DNN 压缩潜力，同时在滤波器维度上保持 DNN 计算量的平衡。模式剪枝和关联剪枝可以再算法层面组合在一起，并在统一的编译器辅助的加速框架下加速。对于我们先进的编译器辅助的 *DNN* 推理框架，我们使用可执行代码生成，其将 DNN 模型转换为计算图，并进行了多种优化，包括高层、细粒度的 DNN 分层信息提取，滤波器核重排列以及负载冗余的消除。所有的设计优化都是通用的，同时适用于 CPU 和 GPU。

我们证明了模式剪枝能持续提高模型的准确性。当其与关联剪枝相结合时，结果仍然优于当前包括结构化和非结构化剪枝的 DNN 剪枝方法。在“准确性分析”一节中，我们展示了 *PCONV* 是当前剪枝加速工作中最理想的稀疏性。我们还将 *PCONV* 模型部署在我们的编译器辅助的移动加速框架上，并使用三个被广泛应用的神经网络（VGG-16、ResNet-50 和 MobileNet-v2），两个基准数据集（ImageNet 和 CiFAR-10），与移动端 CPU 和 GPU 上表现最好的三个框架（Tensorflow Lite，TVM 和 MNN）进行了比较。运行结果表明 *PCONV* 达到了 39.2 倍的加速，并且没有任何精度损失。使用 Adreno 640 嵌入式 GPU，*PCONV* 在 VGG-16 网络和 ImageNet 数据集上达到了前所未有的 19.1ms 的推理时间。据我们所知，这是首次在移动设备上实时运行如此具有代表性的大规模 DNN。

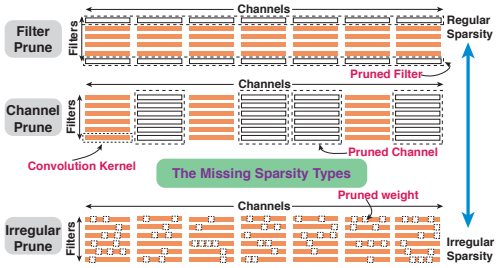

图 1　不同剪枝维度

1.2 背景

1.2.1 DNN 模型压缩

DNN 模型压缩是一种减少原始模型冗余的很有希望的方法。它的目标是，计算图中的权重越少，模型的推理时间就越短。权重剪枝就像外科医生那样移除原来多余的神经元或突触。如图 1 所示，权重剪枝的两种主要方法是笼统的、非结构化的剪枝和结构化的剪枝，它们分别得到不规则的和规则的压缩 DNN 模型。

非结构化剪枝：早期的工作是Han, Mao, and Dally (2015)，其中使用了一种迭代的启发式方法，但模型压缩率有限且不均匀。非结构化剪枝被Zhang et al (2018) 和Ren et al (2019) 进一步发展，他们借助了强大的 ADMMBoyd et al (2011) 优化框架，达到了很高的权值减少率和很有前景的准确性。然而，对于编译器和代码优化而言，卷积核内不规则的权值分布需要大量的控制流指令，这降低了指令级的并行度。同时，不同滤波器的核具有不同的计算量，这在滤波器进行多线程计算时加重了线程级并行性。更进一步而言，不规则的内存访问也导致了存储器较低的访问性能，并增加了运行开销。

结构化剪枝：这种方法被提出用于解决非结构化剪枝带来的索引开销和不平衡工作量。在Wen et al (2016) 和He, Zhang, and Sun (2017) 等人的先驱工作中，结构化权值剪枝可以生成结构化的、更小的权值矩阵，并消除了权值索引的开销，在 CPU 和 GPU 运行时达到了更高的加速。然而当剪枝率上升时，这种方法会遭到准确率下降的困扰。

1.2.2 计算机视觉中的模式

卷积运算在图像处理、信号处理、概率理论和计算机视觉等不同研究领域中已经存在了很长一段时间。在这项工作中，我们着眼于研究传统的图像处理和最先进的卷积神经网络在使用卷积时的关系。在图像处理中，卷积算子是人们根据各种模式的特定特征，利用先验知识手工设计的。另一方面，在卷积神经网络中，卷积核随机初始化，然后在大型数据集上使用基于梯度的学习算法进行权值更新。

Mairal et al (2014) 提出了一种名为卷积核网络（Convolutional Kernel Networks，CKN），其准确率低于当前的 DNN，因此使用受到限制。Zhang (2019) 提出在池化之前将模糊滤波应用于 DNN 以保持平移等效性。之前有限的将传统视觉滤波应用于 DNN 的工作需要改变网络结构，且不着重于权值剪枝或模型加速，因此与 *PCONV* 不同。

1.2.3 移动平台上的 DNN 加速框架

近一段时间，来自学术界和工业界的研究人员们研究了 DNN 移动平台上的推理加速框架，其中包括 TFLiteTen，TVMChen et al (2018)，Alibaba Mobile Neural Network (MNN)Ali，DeepCacheXu et al (2018) 和 DeepSenseYao et al (2017)。这些工作没有依赖于模型压缩技术，且性能仍远没有达到实时的要求。还有其他利用模型稀疏性来加速 DNN 推理的研究，例如Liu et al (2015), SCNNParashar et al (2017)，但它们要么不针对移动平台（需要新的硬件），要么不考虑压缩率和准确性之间的平衡，因此与我们的工作有不同的挑战。

1.3 动机

在当前 DNN 模型压缩和加速的研究基础上，我们分析并重新思考了整个设计空间，并受到以下三点的启发：

同时实现高模型准确性和剪枝规则性。在非结构化剪枝中，任意权值都可以被修剪。这种剪枝具有最大的灵活性，因此也达到了高准确性和高剪枝率。但它对硬件并不友好。另一方面，结构化的剪枝可以生成硬件友好的模型，但剪枝方法缺乏灵活性并会导致准确率下降。我们的动机是最好地利用上述两种稀疏性。为达到这个目的，我们引入了一种新的维度，即基于模式的稀疏性，达到了一个之前未知的设计点，同时具有高准确性和结构规则性。

由图像增强启发的新型卷积模式。当前的 DNN 权值剪枝方法源于消除冗余信息（权重）且不损害准确性的动机。另一方面，这些剪枝方法很少将剪枝作为一种特殊的二元卷积运算看待，更不用说利用其相关的特性了。沿着这种思路，我们发现稀疏卷积模式由于其特殊的视觉特性而具有增强图像质量的潜力。基于稀疏卷积模式可能增强图像质量的事实，我们提出了精心设计的来自数学视觉理论的模式。

编译器辅助的 DNN 推理框架。细粒度的剪枝模式实现了更高的准确性，现在的关键问题是如何重新得到与粗粒度剪枝相似（甚至超过）的硬件效率。我们采用一种独特的方法，设计了一种优化过的、编译器辅助的 DNN 推理框架，以消除完全结构化的剪枝和基于模式的剪枝的性能差距。

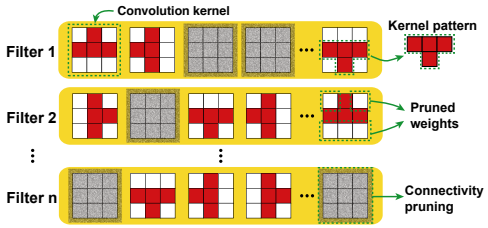

图 2　模式剪枝和关联剪枝

1.4　稀疏卷积模式（SCP）理论

设分辨率 $H \times W$ 的图片可以表示为 $X \in \mathbb{R}^{H \times W \times 3}$。一个 L 层的 DNN 可以表示为一个特征提取器 $\mathcal{F}_L(\mathcal{F}_{L-1}(\cdots \mathcal{F}_1(X)\cdots))$，层编号 $l \in \{1, \cdots, L\}$。在 DNN 内部，每个卷积层可以定义为 $\mathcal{F}_l(X_l) \in \mathbb{R}^{H_l \times W_l \times F_l \times C_l}$，其滤波器核大小为 $H_l \times W_l$，滤波器的个数为 F_l，通道数为 C_l。

除了将剪枝作为一种消除冗余信息的技术外，我们还将其当做一个附加的额外卷积核 P，与原来的卷积核进行逐元素点积。P 被称为稀疏卷积模式（Sparse Convolution Pattern，SCP），其大小为 $H_l \times W_l$，且元素为二进制的值（0 或 1）。根据我们的后续推导，特定的 SCP 很适合数学视觉理论。基于数学上的严格性，我们提出了新的模式剪枝方案，即将 SCP 应用于卷积核上。如图 2 所示，白色块代表每个卷积核中固定数目的被修剪的权值。每个卷积核中剩余的红色块的权值可以为任意值，它们的位置构成了一个特定的 SCP P_i。不同的卷积核可以有不同的 SCP，但 SCP 类型的总数目是有限的。

为了进一步提高剪枝率和 DNN 推理速度，我们可以选择性地切断特定输入通道和输出通道之间的连接，这等价于移除相应的卷积核。这被称为*关联剪枝*。*关联剪枝*如图 2 所示，灰色的卷积核被修剪掉。关联剪枝的基本原理源于在人类视觉系统的启发下进行分层计算时对局部性的需求Yamins and DiCarlo (2016)。它是对模式剪枝的良好补充。两种剪枝方法都可以集成到同样的算法层面解决方案和编译器辅助的移动加速框架中。

1.4.1　卷积算子

在传统的图像处理中，一个卷积算子可以正式地由以下公式定义，其中输出像素值 $g(x, y)$ 是输入像素值 $f(x, y)$ 的加权求和，而 $h(k, l)$ 是权重核的值

$$g(x, y) = \sum_{k,l} f(x + k, y + l)h(k, l) \tag{1}$$

该公式可转换为

$$g(x,y)=\sum_{k,l} f(k,l)h(x-k,y-l) \tag{2}$$

于是我们得到卷积算子的表示符：

$$g=f*h \tag{3}$$

卷积是线性平移不变（linear shift-invariant，LSI）运算，满足交换律、叠加原理和平移不变性。此外，根据 Fubini 定理，卷积具有结合律。

1.4.2 稀疏卷积模式（SCP）设计

我们设计的 SCP 可以转化为一系列可操纵的滤波器Freeman and Adelson (1991)，即高斯滤波器和高斯-拉普拉斯滤波器，它们在数学视觉理论中起到图像平滑、图像边缘化或图像锐化的作用。

高斯滤波器：考虑一个二维的高斯滤波器 G：

$$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4}$$

x 和 y 是输入坐标，σ 是高斯分布的标准差。一般而言，高斯滤波器实现图像平滑。可以先用单位区域的高斯滤波器处理输入图片，然后使用其他可操纵的滤波器，以得到更加复杂的滤波器。

高斯-拉普拉斯滤波器：拉普拉斯算子是二阶微分算子。根据结合律，先使用高斯滤波器平滑图片，再使用拉普拉斯算子，等价于用拉普拉斯-高斯（Laplacian of Gaussian，LoG）算子对图像进行卷积：

$$\nabla^2 G(x,y,\sigma)=\left(\frac{x^2+y^2}{\sigma^4}-\frac{2}{\sigma^2}\right)G(x,y,\sigma) \tag{5}$$

LoG 滤波器是一种带通滤波器，可消除高频和低频噪声。LoG 具有优雅的数学性质，并适用于包括图像增强，边缘检测和立体声匹配在内的多种应用。

泰勒级数展开被用于确定 3×3 大小的 LoG 滤波器的近似值。我们首先考虑一维情况。一维高斯滤波器 $G(x)$ 的泰勒级数展开式为：

$$G(x+h)=G(x)+hG'(x)+\frac{1}{2}h^2G''(x)+\frac{1}{3!}h^3G'''(x)+O\left(h^4\right) \tag{6}$$

$$G(x-h)=G(x)-hG'(x)+\frac{1}{2}h^2G''(x)-\frac{1}{3!}h^3G'''(x)+O\left(h^4\right) \tag{7}$$

将式 (6) 和式 (7) 相加，我们得到

$$G(x+h)+G(x-h)=2G(x)+h^2G''(x)+O\left(h^4\right) \tag{8}$$

高斯算子的二阶导数 $G''(x)$ 等价于 LoG $\nabla^2 G(x)$。式 (8) 可以进一步转化为

$$\frac{G(x-h)-2G(x)+G(x+h)}{h^2}=\nabla^2 G(x)+O\left(h^2\right) \tag{9}$$

对 LoG $\nabla^2 G(x)$ 使用中心差估计，我们得到 LoG 滤波器的一维估计值 $[1\ -2\ 1]$。接下来我们通过 $[1\ -2\ 1]$ 和 $\begin{bmatrix}1\\-2\\1\end{bmatrix}$ 的卷积得到 LoG 滤波器的二维估计值，结果为 $\begin{bmatrix}-1&2&-1\\2&-4&2\\-1&2&-1\end{bmatrix}$。根据二阶微分的性质：

$$\nabla^2 G(x,y)=G_{xx}(x,y)+G_{yy}(x,y) \tag{10}$$

和式 (9)，我们有

$$G_{xx}(x,y)+G_{yy}(x,y)=\left([1\ -2\ 1]+\begin{bmatrix}1\\-2\\1\end{bmatrix}\right)*G(x,y) \tag{11}$$

在式 (11) 的基础上，我们得到 LoG 的另一个近似值 $\begin{bmatrix}0&1&0\\1&-4&1\\0&1&0\end{bmatrix}$。

根据中心极限定理，两个高斯函数的卷积仍是高斯函数，且新高斯函数的方差为原来两个高斯函数方差的和。因此，我们将上面 LoG 的两种近似值卷积，然后归一化，得到增强高斯-拉普拉斯（Enhanced Laplacian of Gaussian，ELoG）滤波器 $\begin{bmatrix}0&1&0\\1&8&1\\0&1&0\end{bmatrix}$。

Siyuan, Raef, and Mikhail (2018) 证明了（多层）DNN 上下文中插值的收敛性，因此我们利用插值概率密度估计来进一步近似。在 ELoG 滤波器中 1 出现的位置，我们有 $(1-p)$ 的概率将其覆盖成 0。因为我们对 n 个卷积层都卷积了 SCP，这种随机覆盖操作可以视为 SCP 的分布插值。在连续概率空间中，将 SCP 插值到卷积函数是一个特定的概率密度函数（Probability Density Function，PDF），因此 SCP 插值的影响是将差值的概率期望累积到 n 个卷积层中。此外，卷积函数被归一化，因此我们在下面的式子中提出系数 p。

$$\underbrace{\begin{bmatrix}0&1&0\\1&1&1\\0&0&0\end{bmatrix}\cdots\begin{bmatrix}0&1&0\\1&1&0\\0&1&0\end{bmatrix}\cdots\begin{bmatrix}0&0&0\\1&1&1\\0&1&0\end{bmatrix}\cdots\begin{bmatrix}0&1&0\\0&1&1\\0&1&0\end{bmatrix}}_{\text{n times interpolation}}=\begin{bmatrix}0&p&0\\p&1&p\\0&p&0\end{bmatrix}^n=\left[p\begin{bmatrix}0&1&0\\1&1/p&1\\0&1&0\end{bmatrix}\right]^n \tag{12}$$

式 (12) 左侧展示了四个 SCP。为了获得 ELoG 滤波器的最佳近似，我们设置 $p=0.75$ 和 $n=8$，则所需的滤波器就等价于将这四个 SCP 插值 8 次。系数 p 在归一化后没有影响。

上界：根据C.Blakemore and Campbell (1969)，使用 LoG 滤波器的最佳次数是 6 次，上界是 10 次。因此所需的式 (12) 中插值 SCP 的次数约为 24 次，最大次数约为 55 次。这个上界涵盖了当前大部分高效的 DNN，甚至是有 50 个卷积层、卷积核大小为 3×3 的 ResNet-152。

式 (12) 中的四个 SCP 通过插值组成了 ELoG 滤波器。因此，设计的 SCP 继承了 LoG 滤波器的降噪和锐化特性。在下面的章节，我们将 DNN 的中间结果可视化，以解释和验证我们设计的 SCP 的优越性。

1.4.3 可视化和解释

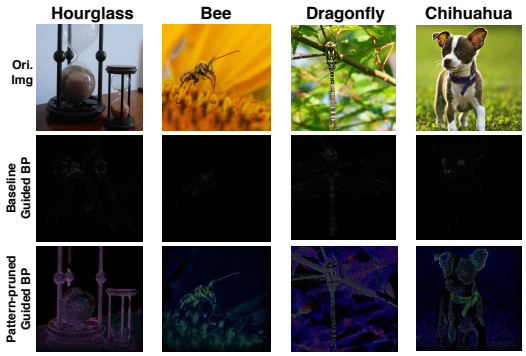

图 3　原 VGG-16 模型和经过模式剪枝的 VGG-16 模型通过定向反向传播得到的可视化中间结果（梯度图片的显著性图）

单个 DNN 决策的解释已经被人们探索过，像生成信息热图如 CAM 和 grad-CAMSelvaraju et al (2017)，以及通过基于最终预测的定向反向传播（BP）Springenberg and Alexey Dosovitskiy (2015)。通过定向反向传播，我们可以可视化 DNN 学习到的内容。图 3 展示了在原 DNN 模型上使用 SCP（模式剪枝）的可视化结果。我们从 ImageNet 数据集中选取了 4 张图片，分别是“沙漏”、“密封”、“蜻蜓”、和“奇瓦瓦狗”，然后使用定向反向传播从每个目标类标签传播回来并得到梯度图像。最终，我们得到了梯度图像的显著性图。相比于原 VGG-16 模型，模式剪枝后的 VGG-16 模型获取了输入图片的更多细节，且噪声更少。

我们得出结论，通过使用我们设计的 SCP，模式剪枝可以增强 DNN 的图像处理能力，这可能会提高 DNN 推理的准确性。

1.4.4 准确性分析

在我们之前的推导中，我们已经决定将（四种）SCP 作为我们的模式集合。我们的算法层面解决方案可以从预训练的 DNN 模型开始，或者从头训练模型。为

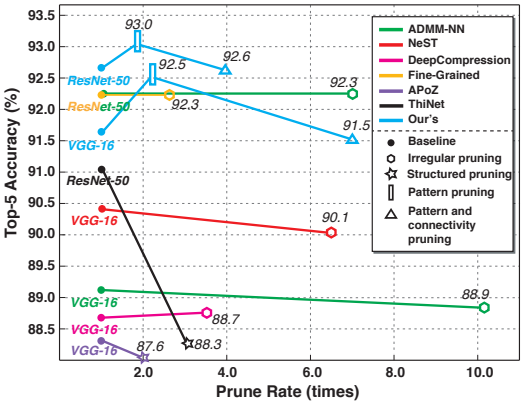

图 4　我们的模式剪枝和关联剪枝在 VGG-16 和 ImageNet 数据集上与 ADMM-NNRen et al (2019)、NeSTDai, Yin, and Jha (2017)、Deep CompressionHan, Mao, and Dally (2015)、Fine-grained pruningMao et al (2017)、APoZHu et al (2016) 和 ThiNetLuo, Wu, and Lin (2017) 的对比

了生成 PCONV 模型，我们需要将 SCP 分配给每个卷积核（模式剪枝），或修剪特定的卷积核（关联剪枝），并训练活跃的（未被修剪的）权值。为了实现这个目标，我们扩展了 (Ren et al. 2019) 中的 ADMM-NN 框架来生成模式剪枝和关联剪枝后的模型。

准确性结果在图 4 中展示。从在许多情况下比其他工作有更高准确率的基线开始，我们得出第一个结论：*将我们设计的 SCP 应用到每个卷积核上时会提高准确率*。在 ImageNet 数据集上，通过将 SCP 应用到每个卷积核上，模式剪枝将 VGG-16 的 top-5 准确率从 91.7% 提升到 92.5%，将 ResNet-50 从 92.7% 提升到 93.0%。准确率的提升得益于我们设计的 SCP 增强了图像处理能力。

非结构化剪枝、结构化剪枝和 *PCONV* 的剪枝 vs 准确率。结合关联剪枝，*PCONV* 达到了更高的压缩率，并且不会影响准确性。通过与其他剪枝方法如非结构化剪枝和结构化剪枝相比，我们得出结论：(i) 与非结构化剪枝相比，*PCONV* 达到了更高的准确率和更高的压缩率，且很接近 ADMM-NN 的结果；(ii) 与结构化剪枝相比，在同样的压缩率下，*PCONV* 达到了更高的准确率，并可以在结构上修剪更多权值而不影响准确率。对于不同稀疏性和压缩率的详细的对比在图 4 中展示。

1.5　编译器辅助的 DNN 推理框架

在本节中，我们提出了新的针对移动设备的编译器辅助的 DNN 推理加速框架。受 *PCONV* 模型灵活性和规则性这两个优点的启发，我们的编译器辅助平台

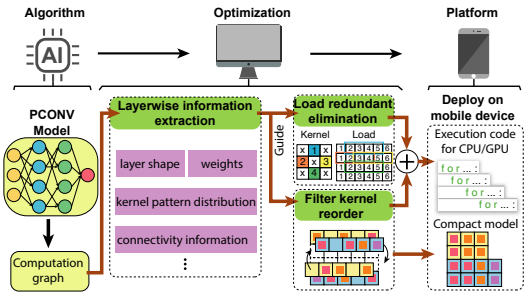

图 5　*PCONV* 的加速框架。从算法层面的设计到平台层面的实现。

独特地实现了优化代码生成，以保证端到端的执行效率。由于 DNN 的计算范例是按层执行的，我们可以将一个 DNN 模型转化为计算图，其可以用 C++（CPU 运行）或 OpenCL（GPU 运行）实现。代码生成过程包括图 5 所示的三个步骤：(i) 层级信息提取；(ii) 滤波器核重排列；(iii) 消除加载冗余。

层级信息提取是一个模型分析过程。特别的是，它分析了详细的卷积核模式和与连接有关的信息。关键信息，如模式的分布、模式的顺序和输入输出通道通过卷积核的连接等，被编译器用来执行步骤 (ii) 和 (iii) 中的优化。

滤波器核重排列被设计用于在指令级和线程级达到最高的并行性。当一个 *PCONV* 模型被训练时，所有卷积核的模式和连接都是已知的，即计算图在模型被部署和推理前就已经确定了。所有这些模式信息是在层级信息中提取的，并被用于滤波器核的重排列，以便 (i) 将具有相似内核的滤波器放在一起以提升线程间并行性，且 (ii) 将同一过滤器内的相同内核放在一起以提升线程内并行性。图 6 展示了滤波器核重排列的两个关键步骤：(i) 将相似的滤波器彼此相邻组织；(ii) 在每个滤波器内，将具有相同模式的内核组织到一起。结果，生成的执行代码消除了大量的执行分支，这意味着更高的指令级并行性；同时，相似的滤波器组提升了运行的相似性，导致良好的加载平衡，达到了更好的线程级并行性。

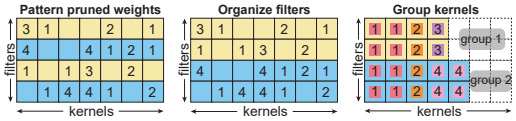

图 6　滤波器核重排列的步骤：每个方格代表一个卷积核；数字代表卷积核的特定模式类型。

消除加载冗余解决了会增大内存开销的不规则的内存访问。在 DNN 执行时，输入输出的数据访问由卷积核的（非 0 元素）模式决定。因此，利用每个卷积核的模式信息，我们可以生成数据访问代码并在 DNN 运行时动态调用它们。数据访问代码包含所有卷积核级别的计算信息，因此可以直接访问与模式内核中的非

0 元素相关联的输入数据。经过步骤 (i) 和 (ii)，模式以结构化的方式分布式存储，这减少了数据访问代码的调用频率，降低了内存开销。

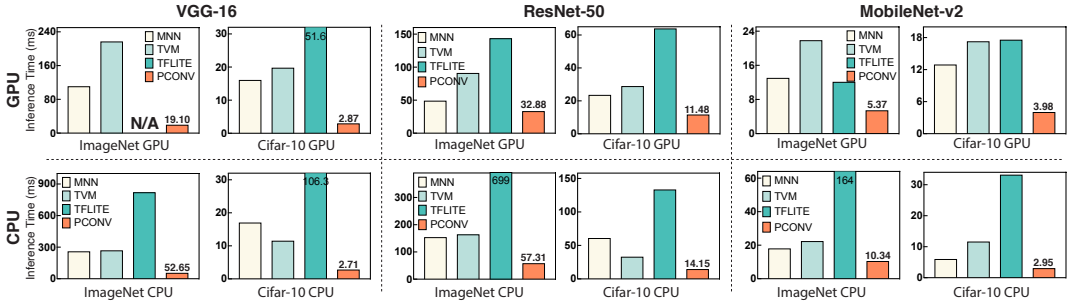

图 7　不同网络结构在 Cifar-10 和 ImageNet 图片上的移动 CPU/GPU 推理时间。

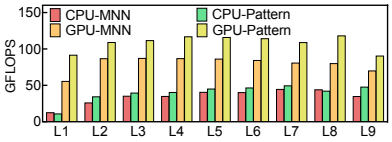

图 8　MNN 和 *PCONV* 在设备上的 GFLOPS 性能评估

1.6　实验结果

在本节中，通过我们部署的 *PCONV* 模型，我们评估了我们编译器辅助框架的运行效率。我们所有的评估模型是通过附录材料中的 ADMM 剪枝算法生成的，并使用 PyTorch 在一台有 8 个 NVIDIA RTX-2080Ti GPU 的服务器上训练。

1.6.1　方法

为了显示 *PCONV* 在移动设备上的加速，我们将其与三个最先进的 DNN 推理加速框架 TFLiteTen、TVMChen et al (2018) 和 MNNAli进行了比较。我们的实验在一台带有高通骁龙 855 移动平台的三星 Galaxy S10 手机上进行，其带有一个高通 Kryo 485 Octa-core CPU 和一个高通 Adreno 640 GPU。

在我们的实验中，我们生成的 *PCONV* 模型基于三个应用广泛的神经网络结构: VGG-16Simonyan and Zisserman (2014), ResNet-50He et al (2016) 和 MobileNet-v2Howard et al (2017)。由于卷积是 DNN 计算中最耗时的步骤（占超过 95% 的总推理时间），我们对上面网络结构的评估集中在卷积层的性能上。为了清晰地展示 *PCONV* 如何提升移动性能，设备层面的评估从三方面显示：(i) 运行时间，(ii) 设备上的 GFLOPS 性能和 (iii) 模式数量如何影响性能。

1.6.2 性能评估

在节中，我们从上面讨论的三个方面展示了我们在移动设备上的评估结果。为了展示出 *PCONV* 具有移动设备上具有最佳的加速性能，我们的比较基线，即 TFLite、TVM 和 MNN 使用了全部的优化设置（如打开了 Winograd 优化）。

运行时间。图 7 展示了 *PCONV* 在我们编译器辅助的 DNN 推理框架上的移动 CPU、GPU 性能。在 CPU 上，*PCONV* 相比 TFLite 达到了 9.4 倍到 39.2 倍加速，相比 TVM 达到了 2.2 倍到 5.1 倍的加速，以及相比 MNN 达到 1.7 倍到 6.3 倍加速。在 GPU 上，*PCONV* 相比 TFLite 达到了 2.2 倍到 18.0 倍加速，相比 TVM 达到 2.5 倍到 11.4 倍加速，相比 MNN 达到了 1.5 倍到 5.8 倍加速。对于最大的 DNN（VGG-16）和最大的数据集（ImageNet），我们的框架在 GPU 上在 19.1ms 的时间内完成了计算，这达到了实时的要求（通常是每秒 30 帧，或每帧 33ms）。

设备上的 GFLOPS 性能。从前面的比较结果我们可以看出 MNN 比 TVM 和 TFLite 有更好的表现。为了证明 *PCONV* 在移动设备上有更好的吞吐量，我们比较了 *PCONV* 和 MNN 在 CPU 和 GPU 上运行的 GFLOPS。图 8 展示了 *PCONV* 和 MNN 比较的层级 GFLOPS 性能。我们从 VGG-16 网络的 13 个卷积层中选择了 9 层，代表了 9 个不同的层和 9 种不同的层大小。另外 4 层在图 8 中忽略了，因为它们重复层大小会导致重复的 GFLOPS 结果。从结果中我们可以看到，*PCONV* 在 CPU 和 GPU 上的吞吐量都优于 MNN。

模式数量 vs 性能。为了确定不同模式的数目如何影响运行效率，我们在一个卷积核内设计了一些除我们设计的 SCP 之外的有 4 个非零元素的随机的模式。表 1 和表 2 展示了不同模式数量的 VGG-16 网络在 Cifar-10 和 ImageNet 数据上的准确率和运行时间。结果表明，随着模式数量增加，准确率与之没有直接关系，但运行性能下降很快，特别是在 ImageNet 数据集上。模式数量 vs 性能的结果表明，我们设计的 SCP 可以获得理想的性能，且精度损失可以忽略。

表 1　模式数量 vs 性能。在 VGG-16 和 Cifar-10 数据集上，对使用了模式稀疏性（2.25×）和关联稀疏性（8.8×）的模型进行测试。表中展示的是 Top-1 准确率。

Dataset	Pattern#	Acc. (%)	Acc. loss (%)	Device	Speed (ms)
Cifar-10	**4**	93.8	-0.3	CPU	2.7
				GPU	2.9
	8	93.7	-0.2	CPU	2.9
				GPU	3.0
	12	93.8	-0.3	CPU	3.1
				GPU	3.3

表 2　模式数量 vs 性能。在 VGG-16 和 ImageNet 数据集上，对使用了模式稀疏性（2.25×）和关联稀疏性（3.1×）的模型进行测试。表中展示的是 Top-5 准确率。

Dataset	Pattern#	Acc. (%)	Acc. loss (%)	Device	Speed (ms)
ImageNet	**4**	91.5	0.2	CPU	52.7
				GPU	19.1
	8	91.6	0.1	CPU	58.9
				GPU	22.0
	12	91.6	0.1	CPU	105.2
				GPU	32.1

1.7　结论

本文介绍了 PCONV，一种 DNN 权值剪枝中理想的稀疏类型，其可带来移动设备上的加速，从而实现实时移动推理。PCONV 继承了非结构性剪枝的高度灵活性，这有助于达到更高的准确率和压缩率；并且保持了像结构化剪枝那样高度规则的权值结构，这使其对硬件更加友好，具有优化的内存访问，平衡的工作负载和计算并行性等。为了展示 PCONV 在移动设备上的实时性能，我们设计了一个编译器辅助的 DNN 推理框架，其可以充分利用 PCONV 的结构特性，在有代表性的大型 DNN 上达到了很高的推理速度。

原文参考文献

https://github.com/alibaba/MNN.

Boticki, I., and So, H.-J. 2010. Quiet captures: A tool for capturing the evidence of seamless learning with mobile devices. In *International Conference of the Learning Sciences-Volume 1*.

Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; and Eckstein, J. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* 3(1):1–122.

C.Blakemore, and Campbell, F. W. 1969. On the existence of neurones in the human visual system selectively sensitive to the orientation and size of retinal images. In *The Journal of Physiology*. The Physiological Society.

Chen, T.; Moreau, T.; Jiang, Z.; Zheng, L.; Yan, E.; Shen, H.; Cowan, M.; Wang, L.; Hu, Y.; Ceze, L.; et al. 2018. TVM: An automated end-to-end optimizing compiler for deep learning. In *OSDI*.

Dai, X.; Yin, H.; and Jha, N. K. 2017. Nest: a neural network synthesis tool based on a grow-and-prune paradigm. *arXiv preprint arXiv:1711.02017*.

Freeman, W., and Adelson, E. 1991. The design and use of steerable filters. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 13, 891–906. IEEE.

Goodfellow, I.; Bengio, Y.; Courville, A.; and Bengio, Y. 2016. *Deep learning*, volume 1. MIT press Cambridge.

Han, S.; Mao, H.; and Dally, W. J. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

He, Y.; Zhang, X.; and Sun, J. 2017. Channel pruning for accelerating very deep neural networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 1398–1406. IEEE.

Hinton, G.; Deng, L.; and Yu, D. e. a. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Hu, H.; Peng, R.; Tai, Y.-W.; and Tang, C.-K. 2016. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NeurIPS*.

Lane, N. D.; Bhattacharya, S.; Georgiev, P.; Forlivesi, C.; and Kawsar, F. 2015. An early resource characterization of deep learning on wearables, smartphones and internet-of-things devices. In *International workshop on IOT towards applications*.

Liu, B.; Wang, M.; Foroosh, H.; Tappen, M.; and Pensky, M. 2015. Sparse convolutional neural networks. In *CVPR*, 806–814.

Luo, J.-H.; Wu, J.; and Lin, W. 2017. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE international conference on computer vision*, 5058–5066.

Mairal, J.; Koniusz, P.; Harchaoui, Z.; and Schmid, C. 2014. Convolutional kernel networks. In *NeurIPS*.

Mao, H.; Han, S.; Pool, J.; Li, W.; Liu, X.; Wang, Y.; and Dally, W. J. 2017. Exploring the regularity of sparse structure in convolutional neural networks. *arXiv preprint arXiv:1705.08922*.

Parashar, A.; Rhu, M.; Mukkara, A.; Puglielli, A.; Venkatesan, R.; Khailany, B.; Emer, J.; Keckler, S. W.; and Dally, W. J. 2017. Scnn: An accelerator for compressed-sparse convolutional neural networks. In *ISCA*.

Philipp, D.; Durr, F.; and Rothermel, K. 2011. A sensor network abstraction for flexible public sensing systems. In *2011 IEEE Eighth International Conference on Mobile Ad-Hoc and Sensor Systems*, 460–469. IEEE.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

Ren, A.; Zhang, T.; Ye, S.; Xu, W.; Qian, X.; Lin, X.; and Wang, Y. 2019. Admm-nn: an algorithm-hardware co-design framework of dnns using alternating direction methods of multipliers. In *ASPLOS*.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Computer Vision (ICCV), 2019 IEEE International Conference on*. IEEE.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Siyuan, M.; Raef, B.; and Mikhail, B. 2018. The power of interpolation: Understanding the effectiveness of sgd in modern over-parametrized learning. In *2018 International Conference on Machine Learning (ICML)*. ACM/IEEE.

Springenberg, J. T., and Alexey Dosovitskiy, T. B. a. R. 2015. Striving for simplicity: The all convolutional net. In *ICLR-2015 workshop track*.

https://www.tensorflow.org/mobile/tflite/.

Wen, W.; Wu, C.; Wang, Y.; Chen, Y.; and Li, H. 2016. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, 2074–2082.

Xu, M.; Zhu, M.; Liu, Y.; Lin, F. X.; and Liu, X. 2018. Deepcache: Principled cache for mobile deep vision. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, 129–144. ACM.

Yamins, D. L., and DiCarlo, J. J. 2016. Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience* 19(3):356.

Yao, S.; Hu, S.; Zhao, Y.; Zhang, A.; and Abdelzaher, T. 2017. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*.

Zhang, T.; Ye, S.; Zhang, K.; Tang, J.; Wen, W.; Fardad, M.; and Wang, Y. 2018. A systematic dnn weight pruning framework using alternating direction method of multipliers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 184–199.

Zhang, R. 2019. Making convolutional networks shift-invariant again. In *ICML*.