面向深度神经网络加速芯片的高效硬件优化策略

张 萌[①] 张经纬*[①] 李国庆[①] 吴瑞霞[①] 曾晓洋[②]

[①](东南大学电子学院国家专用集成电路系统工程技术研究中心 南京 210096)

[②](复旦大学专用集成电路与系统国家重点实验室 上海 200433)

摘 要：轻量级神经网络部署在低功耗平台上的解决方案可有效用于无人机(UAV)检测、自动驾驶等人工智能(AI)、物联网(IOT)领域，但在资源有限情况下，同时兼顾高精度和低延时来构建深度神经网络(DNN)加速器是非常有挑战性的。该文针对此问题提出一系列高效的硬件优化策略，包括构建可堆叠共享计算引擎(PE)以平衡不同卷积中数据重用和内存访问模式的不一致；提出了可调的循环次数和通道增强方法，有效扩展加速器与外部存储器之间的访问带宽，提高DNN浅层网络计算效率；优化了预加载工作流，从整体上提高了异构系统的并行度。经Xilinx Ultra96 V2板卡验证，该文的硬件优化策略有效地改进了iSmart3-SkyNet和SkrSkr-SkyNet类的DNN加速芯片设计。结果显示，优化后的加速器每秒处理78.576帧图像，每幅图像的功耗为0.068 J。

关键词：深度神经网络；目标检测；神经网络加速器；低功耗；硬件优化

中图分类号：TN79.1 **文献标识码**：A **文章编号**：1009-5896(2021)06-1510-08

DOI: 10.11999/JEIT210002

Efficient Hardware Optimization Strategies for Deep Neural Networks Acceleration Chip

ZHANG Meng[①] ZHANG Jingwei[①] LI Guoqing[①]

WU Ruixia[①] ZENG Xiaoyang[②]

[①](*National ASIC Engineering Center*, *School of Electronic Sci. and Eng.*, *Southeast University*, *Nanjing* 210096, *China*)

[②](*National ASIC Key Laboratory*, *Fudan University*, *Shanghai* 200433, *China*)

Abstract: Lightweight neural networks deployed on low-power platforms have proven to be effective solutions for Artificial Intelligence (AI) and Internet Of Things (IOT) domains such as Unmanned Aerial Vehicle (UAV) detection and unmanned driving. However, in the case of limited resources, it is very challenging to build Deep Neural Networks (DNN) accelerator with both high precision and low delay. In this paper, a series of efficient hardware optimization strategies are proposed, including stackable shared Processing Engine (PE) to balance the inconsistency of data reuse and memory access patterns in different convolutions; Regulable loop parallelism and channel augmentation are proposed to increase effectively the access bandwidth between accelerator and external memory. It also improve the efficiency of DNN shallow layers computing; Pre-Workflow is applied to improve the overall parallelism of heterogeneous systems. Verified by Xilinx Ultra96 V2 board, the hardware optimization strategies in this paper improve effectively the design of DNN acceleration chips like iSmart3-SkyNet and SkrSkr-SkyNet. The results show that the optimized accelerator processes 78.576 frames per second, and the power consumption of each picture is 0.068 Joules.

Key words: Deep Neural Networks (DNN); Object detection; Neural network accelerator; Low power consumption; Hardware optimization

收稿日期：2021-01-04；改回日期：2021-04-21；网络出版：2021-04-29

*通信作者： 张经纬 zhangjingwei@seu.edu.cn

基金项目：国家重点研发计划(2018YFB2202703)，江苏省自然科学基金(BK20201145)

Foundation Items: The National Key R&D Program of China(2018YFB2202703), Jiangsu Province of Natural Science and Technology(BK20201145)

1　引言

人工智能算法的理论研究相比于几年前取得了突出的进步。因此，一部分研究人员开始将精力投在把人工智能算法特别是深度神经网络(Deep Neural Networks, DNN)部署在各种硬件平台上，包括中央处理器(Central Processing Unit, CPU)、图形处理器(Graphics Processing Unit, GPU)、现场可编程门阵列(Field Programmable Gate Array, FPGA)、专用集成电路(Application Specific Integrated Circuit, ASIC)上[1,2]。这些基于硬件平台而专门构建出来的神经网络加速器被证明在应用于图像分类[3]、目标检测[4,5]和语音识别等方面有着很好的效果，而像无人机检测、自动驾驶这类的强实时性的人工智能(Artificial Intelligence, AI)、物联网应用不仅对网络识别精度有较高要求，而且对网络推理速度以及硬件平台功耗同样是严峻的挑战。相比较基于性能优先的CPU/GPU这类通用处理器很难满足功耗约束，FPGA和ASIC具有高能效、低功耗的特性成为当前边缘AI应用方案的首选[6]。

但是想要基于低功耗硬件平台，构建出符合要求的DNN加速器是非常有挑战性的[7,8]。要求开发人员不仅了解AI算法，更加需要包括：(1)专用于低功耗平台的DNN体系结构；(2)高效的内存管理方案；(3)并行可靠的工作流。整个任务严重依赖设计者跨越机器学习和集成电路设计的专业知识。设计过程中如果只实现AI算法却不考虑硬件优化策略，就很难满足边缘计算方案低功耗和高实时性的严格要求。因此硬件优化策略具有较高的普适性和重要的研究价值。

为了解决以上诸多问题，本文提出用于DNN加速芯片设计的一系列高效率优化策略，并在FPGA上验证性能，包括可堆叠共享计算引擎(Process Engine, PE)、可调的循环次数、通道增强和预加载工作流优化方法，利用有限的低功耗硬件资源获得最好的性价比。本文所提硬件优化策略：(1)基于SkyNet网络模型应用，设计出SEUer A型加速器，其精度和功耗都要优于iSmart3-SkyNet (the 1st place in the DAC'2019-SDC) [9]；(2)基于SkrSkr-SkyNet(the 2nd place in the DAC'2020-SDC) [10]应用优化出了SEUer B型加速器，最终分数会超过UltraNet(the 1st place in the DAC'2020-SDC)。本文的主要贡献如下：

(1)提出一种可堆叠共享PE用于组装成并行度灵活的卷积计算模块，可以独立且高效率地完成逐点卷积(PointWise Convolution, PWC)计算和深度可分离卷积(DepthWise Convolution, DWC)计算，并且采用了行缓冲区结构解决PWC单次读取特征图的限制。

(2)针对可堆叠共享PE提出了循环次数可调，可根据网络层数自动优化计算模块中循环次数，提高计算的效率。

(3)通道增强的方法不仅可以大幅度增强图片输入层的并行度，还可以增加与外部存储器的通信带宽。

(4)预加载工作流可以更好地协同处理系统(Processing System, PS)与可编程逻辑(Programmable Logic, PL)，从而加速整个系统，降低系统的延迟和功耗。

本文其余安排如下：第2节介绍专用于低功耗平台的SkyNet的网络结构以及存在的问题，然后使用Roofline模型[11]分析基于SkyNet网络设计出的iSmart3-SkyNet加速器的性能瓶颈；第3节针对这些问题给出对应的优化策略；第4节将讨论我们为低功耗平台提出的优化策略的实验结果；第5节给出结论。

2　SkyNet网络模型及其加速器问题分析

2.1　专注于低功耗目标检测的SkyNet模型

SkyNet是一个冠军模型，它被设计出来专门用于嵌入式边缘设备的目标检测和追踪任务[9]。可以为低功耗嵌入式系统提供可靠的推理精度的同时满足实时响应这类的低延时要求。SkyNet采用自下而上的DNN设计方法[12]，将3×3深度卷积层、1×1点卷积层、批量归一化的组合层和ReLU6组合成捆绑包(bundle)。将捆绑包反复堆叠形成如表1所示的网络结构。表1所示的超参数(包括网络通道数和池化层的位置)是通过使用实际硬件进行神经网络搜索而获得的最佳解决方案。从硬件执行效率角度考虑，SkyNet网络比自上而下设计的卷积神经网络(Convolutional Neural Network, CNN)更有效[13]。

CNN网络的图像输入通道一般为3个光学三原色(Red Green Blue, RGB)或4个三原色加不透明参数(Red Green Blue Alpha, RGBA)，因此很难在固定计算通道的PE上提高浅层网络的计算效率[14,15]。例如，表1最后两列反映了iSmart3-SkyNet理论计算量占比跟实际硬件延时占比差异，在第1个捆绑包中，虽然SkyNet的输入层是3，但在理论计算中第1层的PWC的输入层却是32，DWC的输出层却是64而不是48。因为加速器中计算模块输入输出通道固定，iSmart-SkyNet的输入并行度为16，输出并行度为32。在计算第1层时，即便通道当中存在无效数据，它在输入输出通

表 1 SkyNet的体系结构和每个捆绑包的推理速度表格

捆绑包	层数	输入尺寸	操作类型	计算量、计算量占比(%)	延迟占比(%)
#1	1	3×160×320	DW-Conv3	119.61M, 20.6	**33.90**
	2	3×160×320	PW-Conv1		
	3	48×160×320	POOLING		
#2	4	48×80×160	DW-Conv3	86.02M, 14.42	16.54
	5	48×80×160	PW-Conv1		
	6	96×80×160	POOLING		
#3	7	96×40×80	DW-Conv3	61.75M, 10.36	6.23
	8	96×40×80	PW-Conv1		
	9	192×40×80	POOLING		
#4	10	192×20×40	DW-Conv3	60.36M, 10.13	4.92
	11	192×20×40	PW-Conv1		
#5	12	384×20×40	DW-Conv3	160.05M, 26.85	12.43
	13	384×20×40	PW-Conv1		
	–	合并第9层输出			
#6	14	1280×20×40	[旁路] DW-Conv3	107.52M, 18.04	20.08
	15	1280×20×40	PW-Conv1		
#7	16	96×20×40	PW-Conv1	0.77M, 0.14	0.10
–	17	10×20×40	计算回归框		0.16
CPU	–	–	–	–	5.64

道维度依旧各自循环两次，也因此在表格中第1个、第2个捆绑包的计算量之和为34.48%；而实际加上异构计算平台的计算耗时，仅仅第1个捆绑包的延时就占了整体时间的33.90%，实际延时相比较计算量，增加了69%。在这种情况下，第1个、第2个捆绑包的延时占到整个系统延时的1/2。因此考虑到带宽限制等诸多现实因素，提高浅层网络的计算效率，使得延时与计算量一致是非常困难的。

2.2 Roofline模型

计算和通信是系统吞吐量优化中的两个主要限制。加州大学洛杉矶分校丛京生教授团队[11]的Roofline性能模型，反映了系统性能与片外内存流量的关系，更重要的是，突出了硬件平台的峰值性能与神经网络模型之间的相关性。式(1)描述了Roofline模型，P_{P}是系统中所有可用计算资源提供的浮点吞吐量，也称为计算屋顶。B是内存带宽(Memory Bandwidth, BW)，而C_{C}代表计算与通信之比(Computing To Communication, CTC)是每个片上存储器流量的操作。加速器的实际计算性能不能超过计算屋顶和计算与通信比率×内存带宽的最小值。在第1种情况下，可达到的性能受到处理器可提供的最大计算资源限制；第2种情况则可达到的性能受到内存带宽的限制，并且无法充分利用计算资源

$$\text{可获得的吞吐量} = \mathrm{MIN}(P_{\mathrm{P}}, B \times C_{\mathrm{C}}) \tag{1}$$

对于不同类型卷积以及不同循环展开的相同卷积，加速器计算通信比公式都是不一样的。对于深度可分离卷积，计算通信比如式(2)所示

$$\begin{aligned}\text{DWC CTC.} &= \frac{\text{总操作数}}{\text{总执行周期}} \\ &= \frac{2 \times R \times C \times M \times N \times K}{\alpha_i \times \beta_i + \alpha_w \times \beta_w + \alpha_o \times \beta_o}\end{aligned} \tag{2}$$

其中，M, N分别为输入通道和输出通道，输出特征图尺寸为R×C，卷积核大小为K，卷积步长为S。$\alpha_i, \alpha_w, \alpha_o$和$\beta_i, \beta_w, \beta_o$的定义在式(3)中。[$T_m$, T_n, T_r, T_c]分别是输入、输出通道和输出特征图行和列的分块大小。可以理解为对应循环的并行度

$$\left.\begin{aligned} a_i &= a_w = a_o = \frac{N}{T_n} \times \frac{R}{T_r} \times \frac{C}{T_c} \\ \beta_i &= T_m(ST_r + K - S)(ST_c + K - S) \\ \beta_w &= T_m K^2 \\ \beta_o &= T_m T_r T_c \end{aligned}\right\} \tag{3}$$

对于逐点卷积的计算通信比如式(4)和式(5)所示，其中出现的参数量的含义与式(2)和式(3)的一致

$$\begin{aligned}\text{PWC CTC.} &= \frac{\text{总操作数}}{\text{总执行周期}} \\ &= \frac{2 \times R \times C \times M \times N}{\alpha_i \times \beta_i + \alpha_w \times \beta_w + \alpha_o \times \beta_o}\end{aligned} \tag{4}$$

$$\left.\begin{aligned} a_i &= a_w = \frac{M}{T_m} \times \frac{N}{T_n} \times \frac{R}{T_r} \times \frac{C}{T_c} \\ a_o &= \frac{N}{T_n} \times \frac{R}{T_r} \times \frac{C}{T_c} \\ \beta_i &= T_m(ST_r + K - S)(ST_c + K - S) \\ \beta_w &= T_m T_n K^2 \\ \beta_o &= T_m T_r T_c \end{aligned}\right\} \tag{5}$$

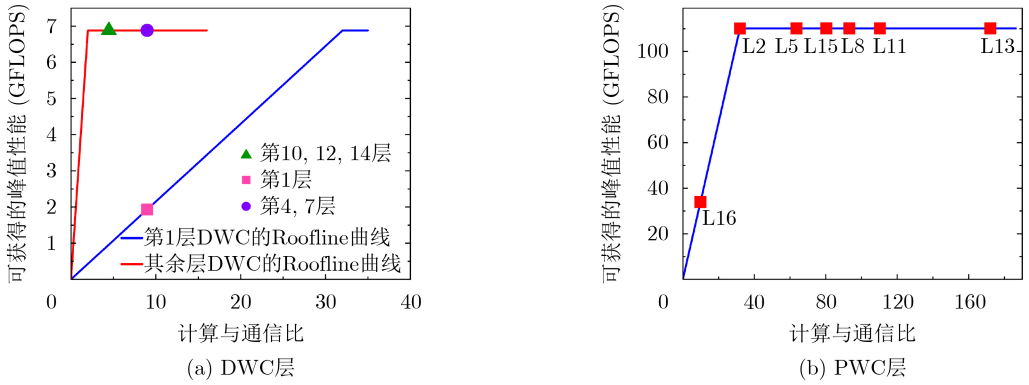
在iSmart3-SkyNet中，DWC和PWC分别由不同的计算引擎计算。DWC和PWC的Roofline模型的说明如图1所示。对于DWC的Roofline模型的图示如图1(a)所示，第1层可获得的性能非常低，这受到带宽的限制。因此，捆绑包#1的等待时间百分比大于捆绑包#1的浮点操作次数(Floating Point of OPeration, FLOP)。此外，计算上限仅约为6.88 GFLOPS，这受计算资源的限制，因为PWC计算引擎占用了最多的计算资源。图1(b)显示了PWC层的Roofline模型，所有层的可达到的性能都达到了计算上限110 GFLOPS。但是，计算能力不高。此外，DWC和PWC由不同的计算引擎计算。换句话说，当PWC引擎工作时，DWC引擎是自由的，反之亦然。计算资源的利用率不高。在第1层中，输入通道为3，但是iSmart3-SkyNet计算32个通道。冗余计算会增加推理延迟。这些问题在其他用于轻量级目标检测网络的加速器中也存在。为了解决这些问题，提出了几种方法，这些方法在第3节介绍。

3　优化策略

本节首先介绍高效优化策略下的加速器总体架构，该架构可专用于低功耗平台。以专用于低功耗平台的网络SkyNet为例，经问题分析，可堆叠共享PE、通道增强、循环次数可调、预加载工作流优化方法依次被提出。这样，基于低功耗平台有限的硬件资源，可充分发挥并行计算优势，实现更加高效的边缘计算。

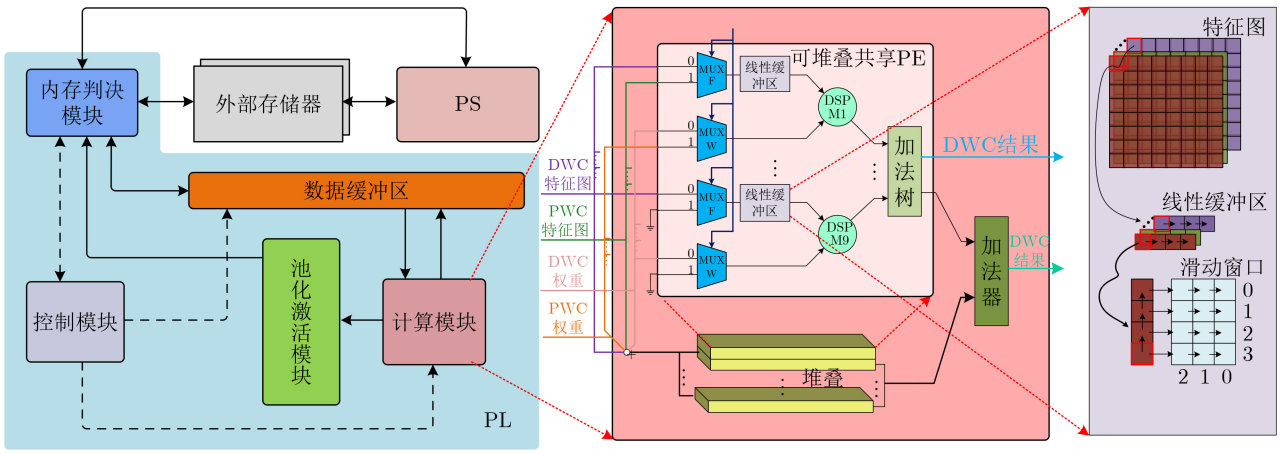
3.1　系统整体结构

系统的整体架构如图2所示，架构基于异构结构，包含了可编程逻辑(Progarmmable Logic, PL)和高性能处理系统(Processing System, PS)。PL由4个模块组成：内存判决模块通过高性能总线接口(Advanced eXtensible Interface, AXI)直接访问外部存储器以及调度双倍的数据缓冲区形成乒乓



(a) DWC层　　(b) PWC层

图 1 iSmart3-SkyNet加速器上的SkyNet Roofline模型分析



(a) 系统整体结构　　(b) 计算模块结构　　(c) 线性缓冲区结构

图 2 系统-计算模块-线性缓冲区结构示意图

结构，最大化利用片上存储器；多个可堆叠共享计算引擎(Process Engine, PE)堆叠成为计算模块来处理PWC和DWC计算；池化激活模块完成池化与激活计算；控制模块被设计用来控制PL逻辑和时序，保证每个模块在正确的时刻被正确地重用。PS采用了多进程优化，以提高图像预处理和后处理的计算速度。

加速器开始工作时，图像和权重被存储在外部存储器中。PS做图像预处理的同时，向PL发出启动信号。PL中的控制模块接收启动信号后，控制内存判决模块通过AXI4接口从外部存储器中读取权重和图像并存储到相应的缓冲区中，接着控制模块按照网络模型顺序依次重复调用计算模块和池化激活模块，最后内存判决模块将结果输出给PS做后处理完成所有任务。

3.2 可堆叠共享PE

由于有限的硬件资源，并且PWC计算引擎占用了大多数计算资源[16]，因此DWC计算引擎的峰值性能非常低。本文提出一种可堆叠的共享PE，以提高DWC计算引擎的峰值性能。几个这样的PE被组装并堆叠到一个大型计算模块中。PWC和DWC操作都共享此计算模块。这意味着在整个任务中只有一个这样的计算模块就可以完成PWC和3×3 的DWC。

如图2(b)所示，共享PE由9个DSP组成，因此可以在1个时钟周期内完成3×3 DWC计算。受双端口块随机存取存储器(dual-port Block Random Access Memory, BRAM)功能的限制，1个时钟周期内最多可以读取两个像素。为了解决这个问题，共享PE利用BRAM和DWC计算引擎之间的行缓冲区。如图2(c)所示，红色特征图的像素被存储在行缓冲区中，而滑动窗口的像素被用于当前计算。在每个时钟周期，窗口中的像素都会移动，新的像素会从行缓冲区中移入，而旧的像素会从窗口中弹出，以便可以展开内核，并且可以映射特征图的9个像素1个时钟周期内即可轻松获得。

对于SkyNet，选择32个共享PE堆叠到计算模块中，并且在计算PWC时，将展开18个输入通道和16个输出通道。16/18输入通道用于传输数据，其他两个通道的值为0。在计算DWC时，所有通道都用于传输数据，并且计算速度将比访问外部存储器更快，并且会触及加速器的存储墙。

3.3 循环次数可调

从表1可以看出，SkyNet的前两层实际耗时比理论上花费更多的时间。浅网络层的通道很少，图像输入通道通常只有3(RGB)或4(RGBA)。具有固定输入和输出通道尺寸的计算引擎将在处理浅层网络时导致某些通道的计算资源空闲，并且其他循环将导致额外的系统延迟。具有不变循环计数的计算模块不能完美地应用于浅层网络。因此，本文提出循环计数可调节的优化方法。

可调节的循环计数方法建立在可堆叠共享PE基础上。通过提供的参数变量[输入通道并行度(Channel Input, CI)]，可以自由地堆叠到计算模块中，它根据不同的层自动调节循环计数，以达到最小功耗和最低延迟。例如，iSmart3-SkyNet的输入通道并行度为16。在计算PWC的第1层时，在输入通道尺寸上循环了两次，而第2个循环为无效计算。最终iSmart3-SkyNet第1层中超过90%的计算资源处于空闲状态，这也导致更长的延迟和更多的功耗。如果iSmart3-SkyNet使用可调节的循环计数优化，则可以基于不同的层调节循环计数。在计算第1个PWC层时，输入通道尺寸将仅循环1次，而不是原始的两倍。即使图像输入通道尺寸仅循环1次，仍然有13/16 DSP处于空闲状态，第1个PWC层的这种无用计算可以通过3.4节描述的通道增强优化来解决。

3.4 通道增强

由表1分析，相比较于第2层网络，第1层网络硬件延时远远大于计算量，占整个系统延时的1/3。这是因为在像SkyNet这样的CNN中，第1层图片的输入通道只有3个(RGB)。即使使用了循环次数可调优化，iSmart3-SkyNet的第1层PWC的通道利用率并不高，DSP的利用效率仅为3/16。为了解决这一问题，通道增强优化被提出。方法如图3所示，先将特征图以像素单位裁剪，并以RGB的顺序在通道维度上重新排列。从低位开始一直排列到高位。下一轮从上一轮结束的像素开始，重复以上操作直至最后一个像素点。如此使得处理后的特征图的通道展宽，提高了第1层的计算并行性。

本文提出的通道增强不同于传统通道增强技术[17]。通道增强优化不仅提高了图片输入层的计算

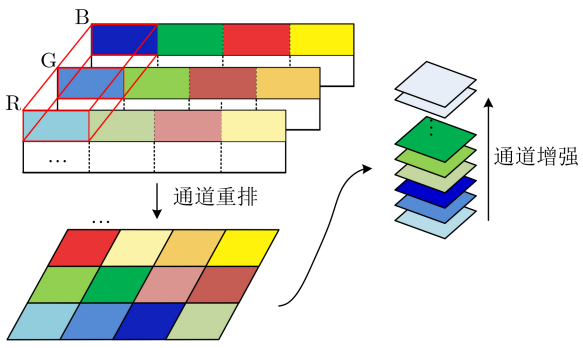


图 3 通道增强流程说明图

并行性，还大大提高了与外部存储器的传输带宽。例如在iSmart3-SkyNet中，传输图像的AXI4的数据位宽只有8位，在Skrskr-SkyNet中传输图像的AXI4的数据位宽度为32位，但只有24位有效。而在使用了通道增强优化的SkrSkr-SkyNet，用于传输图像的AXI4总线数据宽度为128位并且没有原先的无效数据。因此传输图像的带宽大大提升，访问存储器的次数减少，系统延时和功耗也随之减少。

3.5 预加载工作流

当前可用于计算CNN的工作流程如下：获取图像(picture)、对该图像进行预处理、将处理后的图像复制到DDR存储器进行存储、在加速器上计算DNN模型以及后处理，如图4(a)所示，尽管DNN是在PL上并行计算的，但其工作受到串行工作流程的限制，并且必须等待上一幅图像的后处理完成，才能对DNN进行预处理。将处理后的图像复制到DDR内存中，然后才能开始工作。此串行工作流程效率低下。一般的并行工作流程如图4(b)所示。与串行工作流程相比，在并行工作流程中，PL正在执行DNN计算，而PS也在预处理下一张图片。但是，PL仍处于空闲状态。因此，提出了预加载工作流程。如图4(c)所示，在计算图像的第1层之后，PS将预处理的下一个图像复制到DDR以替换旧图像。由于PL几乎始终处于工作状态，因此预加载工作流程的效率高于图4(b)。

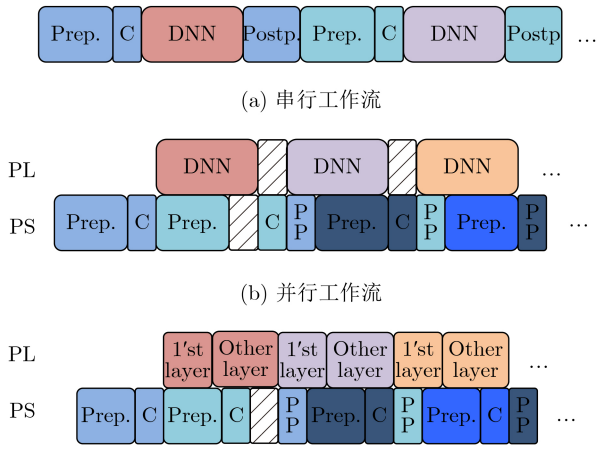


图 4 3种工作流比较图

值得一提的是，其中PS上的预处理速度一般都要快于PL上DNN的计算速度。而要想做到这一点，需要对PS做多进程优化。以多个进程对摄像头采集的照片同时预处理，然后将图像放入队列并等待存入DDR中。PL在计算完DNN之后会将结果存入另一个队列，PS也会以多进程优化确保后处理时间快于DNN第1层的计算时间。如此一来，系统的关键路径在PL处，且PL的效率接近100%。

4 实验结果分析

根据第3节提出的优化方法优化iSmart3-SkyNet加速器后，优化后的DWC层的Roofline模型如图5(a)所示。可以看出，与原始Roofline模型相比，第1层DWC的性能不再比其他DWC层的性能更弱，因为通道增强优化增加了带宽。由于共享计算资源，DWC的计算能力从原来的6.88 GFLOPS增加到172.8 GFLOPS。它使所有DWC层从计算限制变为内存限制，使得可达到的性能得到了显著提高。PWC的Roofline模型如图5(b)所示。优化的PWC计算屋顶从原来的110 GFLOPS增加到307.2 GFLOPS。计算屋顶的增加导致该层的一部分从计算限制更改为内存限制，使得每层可获得的性能都比iSmart3-SkyNet高。

为了更好地说明本文优化方法在低功耗目标检测方面的性能改进，分别选择了iSmart3-SkyNet和SkrSkr-SkyNet作为基准模型。本文iSmart3-SkyNet和Skrskr-SkyNet中提出的方法的有效性如图6所示，其中它们的性能(包括延迟和能量)被标准化为1。与iSmart3-SkyNet和SkrSkr-SkyNet相

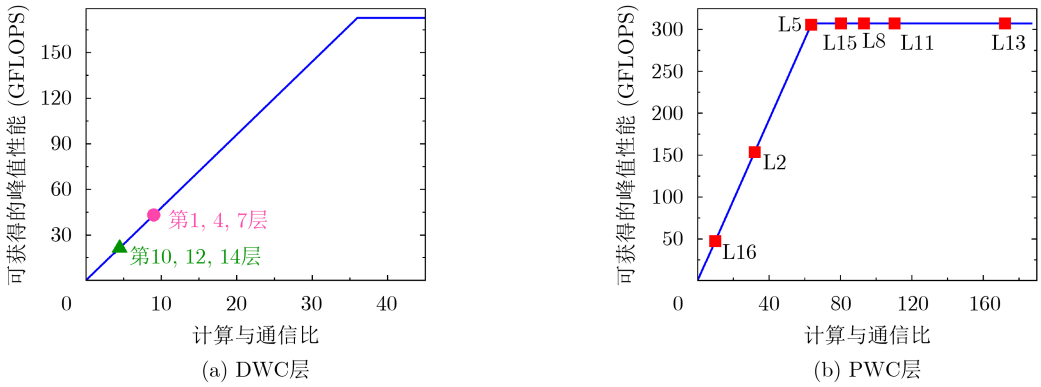


图 5 优化后加速器上的SkyNet Roofline模型分析

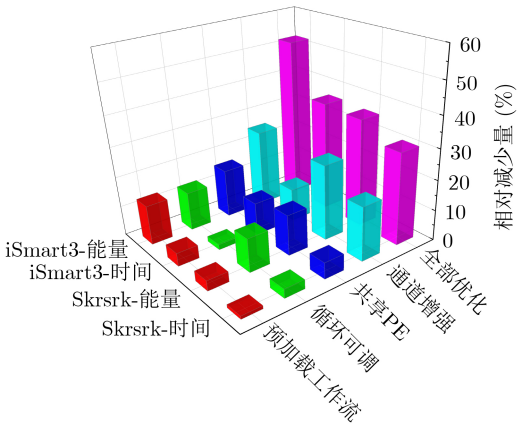


图 6 iSmart3和Skrskr加速优化前后性能对比

比，使用预加载的工作流方法处理1000幅图像，分别可以将运行时间减少约4.44%和1.11%，并将能耗分别减少13.27%和3.45%。当处理的图像数量增加时，预加载工作流程可以减少更多的运行时间和能耗。对于iSmart3和Skrskr，可调节循环次数方法的运行时间分别减少了1.78%和3.10%，能耗分别减少了12.92%和11.29%。考虑到可调节循环计数仅应用于前两个PWC层，因此更改为合适的网络模型将获得更显著的效果。对于iSmart3，可堆叠共享PE减少了10.87%的运行时间和16.07%的能耗。对于Skrskr，可堆叠共享PE减少了5.49%的运行时间和13.55%的能量。由于内存的限制，运行时间的减少并不重要。但是，由于减少了计算资源的消耗，因此能源的节省非常重要。通道增强方法具有最大的改进，可以减少Skrskr和iSmart3的运行时间10.85%和17.98%。它们的功耗降低了约1/4。通道增强不仅可以提高硬件资源的利用率，而且可以大大提高加载图像的带宽。将所有优化方法应用于SkrSkr，运行时间缩短了30.29%，能耗降低了35.49%。iSmart3的改进更加明显，运行时间缩短了35.98%，能耗降低了52.06%。

与之前的DAC-SDC目标检测加速器相比，结果如表2所示。与iSmart3- SkyNet相比，Skrskr-SkyNet使用了更有效的量化，这也使得Skrskr的GOPS/W值为iSmart3的2.24倍，准确率也因为DWC使用了线性缓冲区从0.716增加到0.731。根据第3节所述的优化策略，可以在不改变量化方法的情况下对每个加速器进行进一步优化。基于iSmart3改进的SEUer A型不仅由于行缓冲区使精度提升到0.724，GOPS/W和Energy/Picture还分别优化了1.85倍和2.14倍。基于Skrskr改进的SEUer B型帧率从52.429提高到78.576，功耗降低了近1/2。此外，SEUer B型的GOPS/W和Energy/Picture分别比Skrskr增加了1.5倍和1.9倍。

表 2 优化策略效果对比

加速器	iSmart3 [9]	SEUer A	Skrskr [10]	SEUer B
网络模型	SkyNet	SkyNet	SkyNet	SkyNet
量化精度	A9/W11	A9/W11	A8/W6	A8/W6
硬件平台	Ultra96V2	Ultra96V2	Ultra96V2	Ultra96V2
准确率(DJI)	0.716	0.724	0.731	0.731
时钟频率(MHz)	215	215	300	300
DSP数量	329	287	360	360
LUT数量(k)	54	54	56	46
FF数量(k)	60	70	68	51
帧率(fps)	25.05	37.393	52.429	78.576
GOPS/W	3.21	5.95	7.22	11.19
Energy/Pic.(J)	0.289	0.135	0.129	0.068

5 结束语

本文利用Roofline模型对深度可分离卷积进行分析，然后面向低功耗目标检测提出了DNN加速芯片的优化策略，包括可堆叠共享PE、可调节循环次数、通道增强和预加载工作流等优化措施，通过选择iSmart3-SkyNet和Skrskr-SkyNet作为基础网络，FPGA验证效果显著。对于iSmart3-SkyNet，本文提出的方法可以增加精度并提高能效；对于Skrskr-SkyNet，本文基于提出的优化策略所设计的加速器以78.576 fps的速度和0.068 J/图像的速度进行计算，性能超过2020年DAC低功耗目标检测国际顶尖竞赛第1名UltraNet，目前在此类目标检测加速器设计领域处于较为领先地位。

参 考 文 献

[1] 王巍, 周凯利, 王伊昌, 等. 基于快速滤波算法的卷积神经网络加速器设计[J]. 电子与信息学报, 2019, 41(11): 2578–2584. doi: 10.11999/JEIT190037.
WANG Wei, ZHOU Kaili, WANG Yichang, *et al*. Design of convolutional neural networks accelerator based on fast filter algorithm[J]. *Journal of Electronics & Information Technology*, 2019, 41(11): 2578–2584. doi: 10.11999/JEIT190037.

[2] ZHANG Xiaofan, WANG Junsong, ZHU Chao, *et al*. DNNBuilder: An automated tool for building high-performance DNN hardware accelerators for FPGAs[C]. 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), San Diego, USA, 2018: 1–8.

[3] LI Huimin, FAN Xitian, JIAO Li, *et al*. A high performance FPGA-based accelerator for large-scale convolutional neural networks[C]. The 26th International Conference on Field Programmable Logic and Applications (FPL), Lausanne, Switzerland, 2016: 1–9.

[4] REDMON J, DIVVALA S, GIRSHICK R, *et al*. You only

look once: Unified, real-time object detection[C]. 2016 IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, USA, 2016: 779–788.

[5] REN Shaoqing, HE Kaiming, GIRSHICK R, *et al*. Faster R-CNN: Towards real-time object detection with region proposal networks[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017, 39(6): 1137–1149. doi: 10.1109/TPAMI.2016.2577031.

[6] TAN Mingxing, PANG Ruoming, and LE Q V. EfficientDet: Scalable and efficient object detection[C]. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, USA, 2020: 10781–10790.

[7] YU Yunxuan, WU Chen, ZHAO Tiandong, *et al*. OPU: An FPGA-based overlay processor for convolutional neural networks[J]. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, 2020, 28(1): 35–47. doi: 10.1109/TVLSI.2019.2939726.

[8] YU Yunxuan, ZHAO Tiandong, WANG Kun, *et al*. Light-OPU: An FPGA-based overlay processor for lightweight convolutional neural networks[C]. 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Seaside, USA, 2020: 122–132.

[9] ZHANG Xiaofan, LU Haoming, HAO Cong, *et al*. SkyNet: A hardware-efficient method for object detection and tracking on embedded systems[J]. arXiv: 1909.09709, 2019.

[10] JIANG W, LIU X, SUN H, *et al*. Skrskr: Dacsdc. 2020 2nd place winner in fpga track[EB/OL]. https://github.com/jiangwx/SkrSkr/, 2020.

[11] ZHANG Chen, LI Peng, SUN Guangyu, *et al*. Optimizing FPGA-based accelerator design for deep convolutional neural networks[C]. 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, Monterey, USA, 2015: 161–170.

[12] HAO Cong, ZHANG Xiaofan, LI Yuhong, *et al*. FPGA/DNN Co-Design: An efficient design methodology for 1ot intelligence on the edge[C]. The 56th ACM/IEEE Design Automation Conference (DAC), Las Vegas, USA, 2019: 1–6.

[13] MOTAMEDI M, GYSEL P, AKELLA V, *et al*. Design space exploration of FPGA-based deep convolutional neural networks[C]. The 21st Asia and South Pacific Design Automation Conference (ASP-DAC), Macao, China, 2016: 575–580.

[14] FAN Hongxiang, LIU Shuanglong, FERIANC M, *et al*. A real-time object detection accelerator with compressed SSDLite on FPGA[C]. 2018 International Conference on Field-Programmable Technology (FPT), Naha, Japan, 2018: 14–21.

[15] LI Fanrong, MO Zitao, WANG Peisong, *et al*. A system-level solution for low-power object detection[C]. 2019 IEEE/CVF International Conference on Computer Vision Workshops, Seoul, Korea (South), 2019: 2461–2468.

[16] DONG Zhen, WANG Dequan, HUANG Qijing, *et al*. CoDeNet: Efficient deployment of input-adaptive object detection on embedded FPGAs[J]. arXiv: 2006.08357, 2020.

[17] WU Di, ZHANG Yu, JIA Xijie, *et al*. A high-performance CNN processor based on FPGA for MobileNets[C]. The 29th International Conference on Field Programmable Logic and Applications (FPL), Barcelona, Spain, 2019: 136–143.

张　萌：男，1964年生，研究员，研究方向为数字信号处理、深度学习算法及硬件加速.

张经纬：男，1997年生，硕士生，研究方向为深度学习硬件加速器设计.

李国庆：男，1991年生，博士生，研究方向为计算机视觉和深度学习硬件加速器设计.

吴瑞霞：女，1996年生，硕士生，研究方向为深度学习算法.

曾晓洋：男，1972年生，教授，研究方向为高能效系统芯片(SoC).

责任编辑：余　蓉