文章编号：2095-4980(2024)10-1142-12

基于FPGA的卷积神经网络加速器现状研究

张　坤，高　博，冀亚玮，谢宗甫，高　飞，李宇东

(战略支援部队信息工程大学 信息系统工程学院，河南 郑州 450000)

摘　要：近年来，得益于计算机运算能力的提升和互联网所产生的大量数据，深度学习(DL)技术取得了快速发展，其中最显著的卷积神经网络(CNN)在图像识别、目标检测、自然语言处理等领域已经成功实现商用。然而随着网络层数越来越深，对计算能力和内存需求急剧上升，如何对卷积神经网络进行加速并在硬件加速器上部署的问题逐渐成为学术界研究的热点。从现场可编程门阵列(FPGA)开发神经网络的优势出发，介绍了FPGA的多种开发方式，详细论述了部署和加速卷积神经网络的各种优化策略，以及采用不同优化策略的FPGA卷积神经网络加速器的性能表现。最后，展望了FPGA卷积神经网络加速器的未来发展方向。

关键词：卷积神经网络；FPGA加速器；网络压缩；卷积算法；脉动阵列

中图分类号：TN914.42　　**文献标志码：**A　　**doi：**10.11805/TKYDA2022213

A survey of convolutional neural network accelerator based on FPGA

ZHANG Kun，GAO Bo，JI Yawei，XIE Zongfu，GAO Fei，LI Yudong

(School of Information Systems Engineering，Information Engineering University，Zhengzhou Henan 450000，China)

Abstract：In recent years, thanks to the enhancement of computing power of computers and the vast amount of data generated by the internet, Deep Learning(DL) technology has achieved rapid development. Among them, the most notable Convolutional Neural Networks(CNN) have successfully been commercialized in fields such as image recognition, object detection, and natural language processing. However, as the network layers become deeper, the demand for computing power and memory has risen sharply. How to accelerate convolutional neural networks and deploy them on hardware accelerators has gradually become a hot topic in academic research. Starting from the advantages of developing neural networks with Field-Programmable Gate Arrays(FPGA), various development methods of FPGA are introduced, various optimization strategies for deploying and accelerating convolutional neural networks are discussed in detail, and the performance of FPGA convolutional neural network accelerators using different optimization strategies is presented. Finally, the future development direction of FPGA convolutional neural network accelerators is expected.

Keywords：Convolutional Neural Networks；FPGA accelerator；network compression；convolutional algorithm；systolic array

深度学习(DL)，特别是卷积神经网络(CNN)的快速发展，在识别精确度上的显著提升，促进了目标检测、人脸识别、语音识别、自动驾驶等应用场景的实现，未来将会深刻改变人们的生活方式，推动社会经济快速发展。然而，随着模型越来越深，计算量越来越大，以及对实时性、低功率等应用场景的需求，使得神经网络加速问题日益凸显。

神经网络包括训练和推理两个阶段，训练阶段对数据精确度要求较高，通常采用浮点数据类型。目前，有TensorFlow、PyTorch、Caffe等深度学习库可以很方便地利用图形处理器(Graphics Processing Unit，GPU)的算力加速训练过程。而推理阶段通常具有实时性、低功耗等现实需求，现场可编程门阵列(FPGA)为实现神经网络前向推理加速提供了有吸引力的解决方案，因为FPGA平台具有灵活编程、数据流驱动、高能效、低功耗等特点，可以通过硬件优化映射任何给定的神经网络算法，针对快速更迭的卷积神经网络具有优势，近些年来逐渐成为学术界研究的热点。本文主要讨论FPGA在网络推理阶段的加速研究。

收稿日期：2022-10-14；**修回日期：**2022-10-22

CNN理论最早在20世纪60年代由David Hubel和Torsten Wiesel提出，他们受到猫的大脑视觉神经传递信息启发，提出了感受野[1]的概念。之后，陆续有研究者发表CNN相关研究工作。1998年，Yann Lecun提出了LeNet-5模型[2]，采用反向传播(Back Propagation，BP)算法作为目标函数进行训练，该模型对小数据集如手写字等取得了不错的识别效果，然而对大规模数据的识别效果不佳。受限于当时计算能力和数据量的不足，CNN相比于传统算法在精确度上并没有明显优势，所以没有引起人们的重视。直到2012年AlexNet模型[3]在ILSVRC2012挑战赛取得冠军，越来越多的CNN相关研究才逐渐被提出，此后VGGNet[4]、ResNet[5]等新兴神经网络结构的识别精确度越来越高。然而数据量大、计算密集和频繁访存等特点，使得其在嵌入式平台部署难以实现。例如浅层神经网络AlexNet就有超过6 000万的参数量，需要大约250M容量来存储权重，识别一张图像需要4.6亿次的操作。

近几年，各种轻量级神经网络陆续被提出，在不降低网络准确度的同时，大大降低了计算量和参数量，如SqueezeNet[6]、MobileNet系列[6-9]、ShuffleNet系列[10-11]等，给神经网络在嵌入式等资源受限设备的部署提供了便利，但如何在FPGA上快速实现神经网络加速仍旧是一个亟待解决的问题。CNN中包含着大量的乘加运算，而通用的中央处理器(Central Processing Unit，CPU)指令顺序执行和共享内存的特点，导致网络运算效率很低。于是，业界迫切需要研发出适用于神经网络的硬件加速器。数字信号处理(Digital Signal Processing，DSP)是针对数字信号处理任务而设计的高速专用芯片，本质上DSP采用的是与通用处理器类似的哈佛总线结构，具有指令顺序执行的特点，因此基于DSP的神经网络加速研究并不多见。多核GPU架构的单指令流多数据流(Single Instruction Multiple Data，SIMD)并行处理能力使得它非常适合处理高密度的浮点数矩阵计算，但同时高密度浮点运算导致的高功耗、散热等问题，使其难以适应嵌入式、边缘计算等场景的应用。专用集成电路(Application Specific Integrated Circuit，ASIC)具有更高的能量效率，但不能灵活地将各种CNN算法映射到电路中，难以适应快速更迭的CNN网络模型的发展，且一次性研发成本过高。与其他芯片结构相比，FPGA有着独特的可编程逻辑阵列，从而可以实现大规模的硬件并行和定制化的深度流水线，并带来极高的吞吐量、极低的功耗，在神经网络加速器的应用上具有优势。

1　FPGA及其开发方式

FPGA是一种半定制硬件电路，可以通过编程的方式改变其中的查找表(Look Up Table，LUT)，从而实现各种电路逻辑。FPGA通常包括可编程逻辑单元、可编程IO、布线资源等，为了灵活布线，在可编程连线的某些位置放置了可编程开关矩阵。由于集成工艺的不断发展，目前FPGA还包括可配置的内存模块和DSP(一般由专用乘法器构建)等，FPGA的基本结构示意图如图1所示。

自1984年，赛灵思(Xilinx)推出第一款商用FPGA-XC2064以来，FPGA已经发展了30多年，随着制作工艺的不断提升，目前最大的FPGA已经成为包含数百亿只晶体管的庞大可编程逻辑电路。面对高度复杂的硬件逻辑电路，如何高效编程完成各领域算法的部署，一直是FPGA供应商着力解决的重点问题。接下来，将详细介绍几种具有代表性的FPGA开发方式。

1.1　寄存器传输级(RTL)开发

寄存器传输级(Register Transfer Level，RTL)开发描述了时钟精确控制下寄存器到寄存器之间的逻辑功能，采用的是层级较低的硬件描述语言(Hardware Description Language，HDL)。硬件工程师利用这种语言可以至上而下采用模块化设计的方法，通过类似搭积木的方式逐层构建出复杂的数字系统。

基于RTL的开发方式更贴近底层硬件，经过专业设计后的硬件电路执行效率高，但需要开发者对算法结构和FPGA架构都有很深入的了解，且设计过程相对繁琐，一定程度上限制了软件、算法工程师在FPGA上设计实现各种CNN模型。所以学术界和业界迫切需要采用高层次的开发模式来提高FPGA的易用性和开发效率。

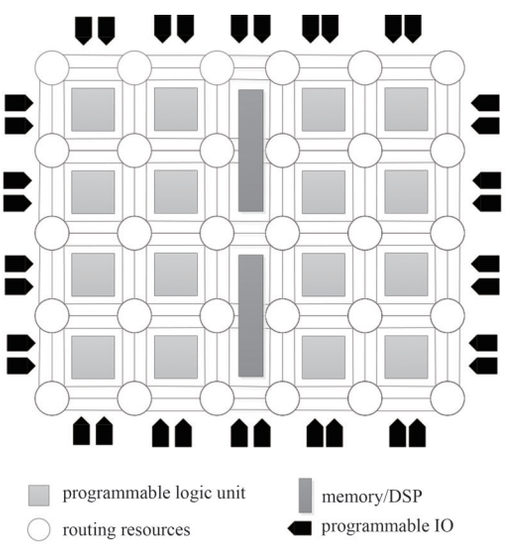

Fig.1 Basic structure of FPGA
图1 FPGA的基本结构

1.2 高层次综合(HLS)开发

高层次综合(High-Level Synthesis，HLS)通常是指将C、C++、OpenCL等具有较高抽象度的编程语言描述的逻辑结构，通过编译工具自动转换成诸如Verilog、VHDL、System Verilog等低抽象级语言描述的电路模型的过程。

通过高层次综合的方法能够屏蔽FPGA的底层逻辑和实现细节，使得软件和算法工程师不用关注底层逻辑，将精力放在算法实现上，快速完成设计迭代。使用像C、C++等高层语言对硬件系统建模，可以将代码密度压缩到原来的10%~14%[12]，这极大地减小了设计复杂度。HLS的开发流程如图2所示。

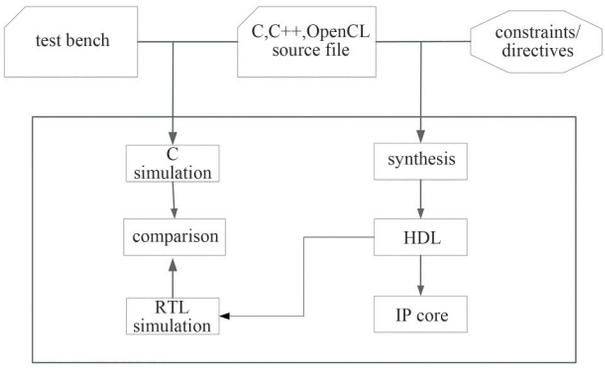

Fig.2 Diagram of High-Level Synthesis process
图2 高层次综合流程示意图

1.3 自动化设计工具链

CPU、GPU的软件编译器经过几十年的发展已经相对成熟，借助Caffe、TensorFlow、PyTorch等深度学习库都能够轻松地完成神经网络的部署。然而针对FPGA的自动化映射工具还远远不够成熟，不能通过简单调用编译器完成从网络模型到硬件的部署。

目前基于FPGA的自动化设计工具链主要分为两种。一种是使用知识产权(Intellectual Property，IP)核组建专用电路的实现方案，如Caffeine[13]、fpgaConvNet[14]、FINN[15]、hls4ml[16]等。该方案从IP库中选取可配置的IP核(由HDL或者HLS方式编写)，并将它们连接起来，以生成神经网络的专用硬件结构。这种方式的缺点是粒度较大，综合、生成比特流时间较长，且缺乏通用性。第二种方式是基于可编程覆盖层[17]的方法，业界的自动化编译工具Vitis AI[18]、VTA[19]、Matlab DLP[20]等都采用基于覆盖层的设计方法，它们能够将Caffe、TensorFLow、PyTorch等框架中神经网络模型经过编译器后，生成二进制比特流，直接部署到FPGA中运行。但是无论是支持的网络类别还是硬件型号，都仍然远远不够成熟或通用，其编译技术未来仍然需要大量的改进，以实现对目标硬件体系结构和神经网络模型的广泛支持。

1.4 开发方式对比分析

虽然采用上述3种开发方式都可以完成神经网络加速器的设计，但是其开发效率和实现的效果却有很大的差异，各种开发方式都有其优势和劣势，适用人群也各有不同。表1总结了RTL、HLS和自动化编译工具3种开发方式，对比分析了它们的编程方式、优缺点以及适用对象。

表1 开发方式对比分析

Table1 Comparison of different development methods

development mode	programming language	advantages	disadvantages	applicable objects
RTL	HDL	close to the bottom layer, fine-grained modeling can be carried out, and the circuit execution efficiency is high	difficult to develop and long design cycle	hardware engineer
HLS	C, C++, System-C, OpenCL	using high-level programming languages, shielding hardware details, and short design cycles	it requires understanding the basic FPGA structure and design constraints, and has a large implementation granularity	software engineer
automated tool chain	C++, Python etc.	a convenient and efficient end-to-end design process that is favored by large companies and developing rapidly	the support for board cards and network models is not comprehensive enough and not yet mature	AI algorithm engineer

2 加速器优化策略

加速器的设计本质上是特定领域算法在硬件上部署和优化的问题。在进行FPGA加速设计时，通常采用简化算法模型和优化硬件结构的方式实现算法在硬件平台的高效运行。基于此，本节从网络模型优化、卷积算法优化以及硬件实现优化3个方面进行阐述。

2.1 网络模型优化

近几年，随着网络深度的增加，模型参数量和运算量日益庞大，限制了神经网络的应用场景。对网络模型进行优化压缩已经成为实际场景中部署神经网络的必要手段。模型压缩的优势在于减少硬件实现的计算和存储负担，神经网络算法的健壮性使得适当压缩模型带来的精确度损失可以忽略不计。根据不同压缩和加速方法的性质，网络模型优化压缩方法主要分为量化、剪枝、轻量型网络3种。

2.1.1 量化

量化是一种减少神经网络模型数据位宽的压缩技术。在训练CNN模型阶段为了寻求较高的模型精确度，通常采用单精度浮点数表示。然而推理应用阶段通常具有低延迟、低功耗等需求，可以采用低位宽的数据表示，以减少在硬件上的存储需求和读写数据所带来的能耗，同时确保网络性能不出现明显下降。

相比CPU、GPU中采用的固定位宽(以8 bit为边界)数据进行运算，FPGA的一个显著优势是可以在运算时采用任意位宽的数据，从而使数据位宽的选择更加灵活。文献[21]基于AlexNet和VGG模型探索权重数据位宽对网络准确率的影响，当权重位宽低于8 bit时，网络准确率会急剧下降。通过对比观察，最终将卷积层权重参数、全连接层参数、中间特征图数据位宽分别设定为8 bit、10 bit、16 bit，在Altera Stratix-V FPGA平台上实现的模型准确率损失相比于浮点单精度权重小于1%。文献[22]采用谷歌提出的Int8量化感知训练方法，所有参数都采用Int8类型数据，在Xilinx Virtex-7 VC707开发板上加速MoblileNetV2网络，数据集采用东北大学发布的热轧带钢表面缺陷数据集，网络准确率为93.89%，数据吞吐量达到了170.06 GOPS。文献[23]将网络的输入和权重从32 bit压缩到了1 bit，卷积运算简化为同或操作，累加操作通过位计数实现，极大地减少了对DSP资源的需求，在Stratix-V FPGA平台上实现了基于Cifar-10数据集的CNN，网络准确率相比于浮点模型精确度下降在3%以内，数据吞吐量达到了9396.41 GOPS，推理速度显著快于以前的设计，然而实验结果表明，激进的二值量化策略对于较大网络模型(如AlexNet，采用224×224×3作为输入)会导致严重的性能衰退，准确率下降超过10%。文献[24]通过聚簇或共享权重的方法将训练权重进行划分，将聚簇中心存储在查找表中，并通过重新训练数据集微调聚簇中心的值，采用该方法将权重位宽从32 bit减少到5 bit，而网络精确度没有损失。

量化是网络模型硬件部署中最常用的优化策略之一，目前，主流的神经网络设计框架如PyTorch、TensorFlow等都有成熟的网络量化方案，可以快捷地完成网络模型量化。量化的好处主要有提升计算吞吐量，降低传输带宽，占用更少内存容量等，实现的难点主要在于在硬件上部署时如何达到如算法所展示的性能提升。

2.1.2 剪枝

卷积神经网络存在着冗余，即使通过剪枝将某些不重要的连接上的权重设置为0，也能够表达出模型特征。在剪枝时，剪枝阈值的选择是设计的关键，剪枝过大会影响模型的准确表达，而恰当的剪枝则对精确度损失较小甚至没有损失，所以剪枝过程需要在压缩模型尺寸和保持精确度之间做出权衡。剪枝前后的模型如图3所示。

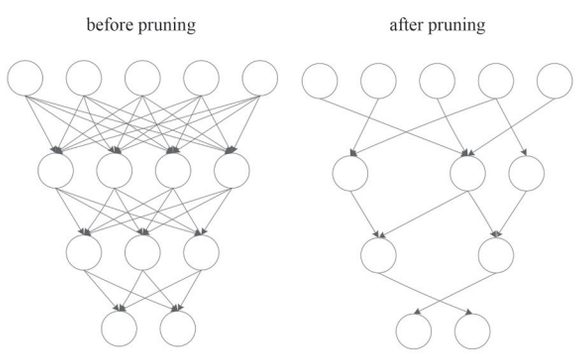

Fig.3 Schematic diagram of network pruning
图3 网络剪枝示意图

近年来，大量的卷积网络模型采用线性整流函数(Rectified Linear Unit，ReLU)作为激活函数。研究表明[25]，ReLU激活函数使得一些经典的网络(如AlexNet，VGG)中大约50%的神经元的值为0。文献[26]在保持准确率的条件下，剪枝掉一些不重要的权重，在AlexNet模型上实现了85%的稀疏度，在ImagNet数据集上的准确率下降仅有大约1%。HAN等人[24]引入参数剪枝的概念，将模型中的近零参数剔除，对剪枝后的稀疏网络重训练，并将临近区间的参数量化为同一数值，采用哈夫曼编码对参数进行索引，实验表明，在去除89%的权重参数的情况下，推理性能几乎不受影响。

细粒度剪枝的缺陷在于引入了稀疏性问题，计算数据的路径不再规则，导致在FPGA上进行并行流水化时，各个数据路径上负载不平衡。采用逐层、逐通道等结构化剪枝的方法可以减少因剪枝导致的稀疏性问题，文献[27]提出了一种块剪枝方法，通过将一段连续的权重取平均值(而不是单个权重)作为是否剪枝的衡量标准，并采用了如图4所示负载均衡方法，在XC7Z020 FPGA上实现了322 GOPS的吞吐量。文献[28]提出了一种硬件友好型剪枝量化(Hardware Friendly Pruning-Quantization，HFPQ)方法，同时采用了分层剪枝和增量量化的方法压缩网络模型，该方法将VGG、ResNet和GooglNet压缩了30倍，同时计算量减少了85%以上，而网络准确率没有明显下降。

2.1.3 轻量化网络

剪枝和量化都是在高性能的网络模型基础上降低计算复杂度，而轻量化网络专注于设计高效的网络模块，如全局平均池化、深度可分离卷积、分组卷积等。这些网络结构具有参数量和计算量小、占用内存少、运算速度快等优点，可以在大幅压缩模型参数的情况下，仍然保持较高的精确度。轻量化网络模型有利于减少计算和带宽压力，加快推理速度，是卷积神经网络未来的一种发展方向。但由于轻量化网络为了保持准确率，模型尺寸(包括深度和宽度)不断增加，对硬件设计提出了新的挑战。

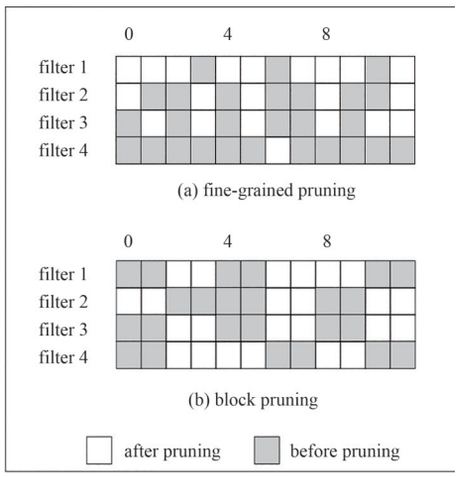

Fig.4 A comparison of different network pruning methods
图4 网络剪枝方式对比图

BAI 等[29]针对 MobileNetV2 和 Xception 中采用的深度可分离卷积在FPGA上的部署问题展开研究，设计了通用的矩阵乘法引擎，采用了乒乓缓冲、数据优化布局等策略，实现的加速器对 ImageNet 数据集图片进行分类，运算性能达到了 170.6 GOPS。HUANG 等[30]基于 ShuffleNetV2 模型设计了动态可变形卷积网络 CoDeNet，量化采用 4 bit 权重和 8 bit 激活，参数量仅为 2.9 M(为 Tiny YOLO 的 1/20.9)，在FPGA上执行目标检测任务，精确度却比 Tiny YOLO 高 10%。

2.2 卷积算法优化

CNN 中的卷积层是计算密集型的，在 LeNet-5、AlexNet、VGG16 等经典卷积神经网络中，卷积运算占据了执行时间的 90% 以上[31]。通常用多层嵌套循环内的乘累加运算来表征卷积核在特征图上滑动的过程，这种空间卷积算法描述非常直观，且容易实现，但计算效率却不高。为了降低卷积运算的复杂度，提高吞吐性能，下面 3 种卷积算法的优化方法常被应用于FPGA加速器。

2.2.1 快速傅里叶变换算法

快速傅里叶变换(Fast Fourier Transform，FFT)是在信号处理中经常使用的方法，使用FFT能够将时域中的卷积操作转换到频域的点乘运算。文献[32]指出，在卷积核尺寸较大(大于5)时，采用频域卷积算法可以有效降低算法复杂度。FFT算法加速卷积首先需要将输入特征图 X 和对应的卷积核 W 分别做FFT变换(其中权重的频域转换过程可以离线完成)，然后在频域做矩阵点乘，最后再通过逆快速傅里叶变换(Inverse Fast Fourier Transform，IFFT)得到输出特征图 Y。式(1)给出了基于FFT算法的卷积运算表达式：

$$Y=IFFT\left(FFT(X)\odot FFT(W)\right) \tag{1}$$

很多加速器设计都采用了频域卷积实现算法。KO 等[33]在文献中提出了一种基于FFT的CNN模型架构，可用于训练和推理过程。在该设计中，训练过程也是在频域中完成的，由于反向传播计算梯度时卷积核尺寸较大，因此加速效果非常明显。ZHANG 等在文献[34]中提出了在FFT的基础上采用 Overlap-and-Add 的优化方法，进一步减少了计算量，在FPGA平台上实现的VGG16、AlexNet和GoogLeNet模型分别达到了123.48 GFLOPS、83.00 GOPS 和 96.60 GOPS 的吞吐量。ZENG 等[35]针对 Overlap-and-Add 存在的问题进行改进，提出了 Concatenate-and-Pad 方法，并采用了一种频域循环平铺算法，所实现的加速器在运行 AlexNet、VGG16 网络模型的推理任务时的吞吐率性能分别达到 780.6 GOPS 和 669.1 GOPS。

FFT 算法将卷积运算转换到频域进行计算，但在卷积核尺寸较大时效果才明显，而 CNN 的发展趋势是采用 3×3、1×1 等小卷积核，因此对于采用小卷积核的 CNN 加速效率并不高。

2.2.2 Winograd算法

Winograd 算法[36]由 Shmuel Winograd 于 1980 年提出，原本是用于信号处理中减少计算量的一种方法。Winograd 算法在减少乘法次数的同时增加了加法运算的次数，考虑到硬件中加法器和乘法器的硬件相对复杂程度，以增加加法次数为代价减少乘法次数是可行的。利用 Winograd 算法计算卷积时，图像是按照一个个数据块来处理的，计算过程与 FFT 算法类似，将输入特征图数据块和卷积核转换到 Winograd 域，点乘后的结果也需要做相应的反变换，生成输出特征图数据块。

Podili 等[37]在加速器实现中采用了 Winograd 算法，并提出了一种新的数据排序，以减少 Winograd 算法带来的巨大内存带宽需求，实现的 VGG16 网络模型的吞吐量达到了 229.2 GOPS。SHEN 等[38]针对 VGG16 和 C3D 这两个

2D和3D CNN模型提出了一种基于Winograd算法的统一架构，采用OpenCL编程语言开发，在Xilinx VC709开发板上VGG模型和C3D的吞吐量分别达到了570 GOPS和430.7 GOPS。Aydonat等在文献[39]中应用Intel OpenCL工具链在Arria10 FPGA平台上映射一维Winograd算法，并采用循环展开和分块策略，设计实现的加速器吞吐量高达1.38 TOPS。LU等[40]提出了Winograd卷积的性能模型，以预测硬件设计的资源利用率，采用数据复用和流水线优化设计，该加速器在推理任务中的吞吐量高达2.94 TOPS。

与FFT算法相比，Winograd算法具有更低的计算复杂度，但转换代价也更高，这是因为随着Winograd算法参数量增大，常数矩阵中的值会骤增[41]，因此，如何有效地使用数据重用方法，减少冗余的片外访存是Winograd卷积加速器的研究重点。

2.2.3 GEMM算法

GPU和DSP常常采用通用矩阵乘法(General Matrix Multiplication，GEMM)的方式实现CNN，并借助OpenBLAS、cuBLAS等矩阵运算库和CUBA编程工具加速，实现对卷积神经网络运算的高效执行。在FPGA中采用GEMM算法的优势是将卷积运算转换为易于实现的矩阵乘法，能够灵活适应不同的网络模型和结构，并可以通过配置控制加速器的并行度。

为了提高加速器的吞吐量，同时让加速器可以兼容其他CNN模型，文献[21]中将卷积运算转换为矩阵乘法，采用OpenCL方式设计了推理加速器，实现了117.8 GOPS的吞吐量性能。文献[42]设计了由基于DSP和LUT的通用矩阵乘法计算模块组成的加速器，有统一的指令集体系结构和缓冲区，以异构方式形成了整个计算系统，实验中，将MobileNetV3部署到XC7Z045平台上实现了364 GOPS的吞吐量。

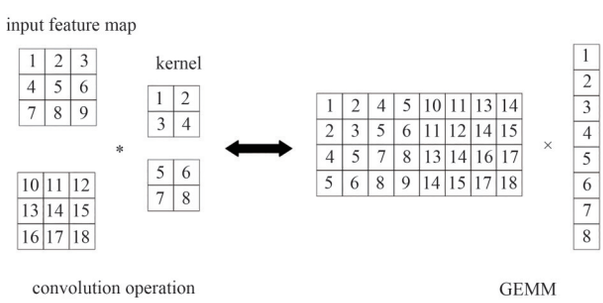

Fig.5 Converting convolution operation to GEMM operation
图5 将卷积操作转化为GEMM

将步长为1的卷积操作转化为GEMM的过程如图5所示。矩阵乘法不会减少运算量，且由于通常卷积滑动步长小于卷积核大小，在将输入特征图展开时会存在区域重叠的情况，使得展开后的矩阵元素个数多于原特征图的元素个数，额外占用了存储资源。针对这一问题，文献[43]中指出一个$K_h \times K_w$卷积核可以通过$K_h K_w$个1×1卷积核来替代，而采用1×1卷积核时不会额外占用存储空间，设计了脉动阵列来加速矩阵乘法运算，在加速器上部署MobileNetV1和Inception V4网络执行推理任务都达到了3.5 TOPS的吞吐量。

2.2.4 卷积优化算法性能对比

表2是近几年采用卷积优化算法在FPGA上的实现，可以看出Winograd算法使用得更加广泛，这源于Winograd算法比FFT算法的计算复杂度更低[41]。从趋势上看，采用的网络从AlexNet、VGGNet等经典网络模型逐渐向轻量化的MobileNet系列模型过渡，量化位宽的选择也越来越低。加速器的吞吐量与DSP资源的使用量总体上成正比，其中文献[40]采用Winograd算法实现的加速器的DSP资源效率为1.21 GOP·s^{-1}·DSP^{-1}，达到最高资源效率。

表2 优化算法在FPGA上实现对比

Table2 Comparison of optimization algorithm implementation on FPGA

item	FFT		Winograd			GEMM	
	[34]	[44]	[39]	[40]	[45]	[43]	[42]
year	2017	2020	2017	2017	2022	2020	2021
device	Strati5–V QPI	Alveo U200	Arria10 GX1150	ZCU102	XC7K325T	XCVU9P	XC7Z045
network	VGG	VGG	AlexNet	VGG	VGG	MobileNet V1	MobileNet V2
datatype	fp32	fix16	fp16	fix16	fix8	int8	Flexible
frequeny/MHz	200	200	303	200	150	300	100
DSP	224	2 680	1 576	2 520	840	5149	900
LUT/K	201	230	246	600	—	246	193
throughput/GOPS	123	1 699	1 382	3 045	441	3651	364

2.3 硬件实现优化

本节讨论在基于FPGA的卷积神经网络加速器设计中常用的硬件优化策略，包括循环优化、脉动阵列、完全

片上映射和多层融合等。文献[46]指出利用FPGA加速卷积神经网络性能的瓶颈主要体现在两个方面：计算量和数据传输带宽。循环展开和脉动阵列常用于提高计算的并行度，完全片上映射和多层融合降低了片外数据交换带来的时间和能量开销。

2.3.1 循环优化

卷积运算可以表示为6层嵌套循环的形式，在最内层完成卷积的乘累加运算，如图6所示。若循环操作使用最少的硬件资源以时分复用的方式串行执行，则计算效率较低。在优化神经网络加速器时，通常将没有数据依赖关系的循环展开，以提高数据并行执行能力。CNN加速的关键在于将嵌套循环展开并行计算，然而有限的硬件资源限制了并行计算的程度，同时由于数据之间的依赖关系，导致部分维度展开效果不理想。

```
for (r=0; r<R; r++)                          //feature row
for (c=0; c<C; c++)                          //feature column
for (o=0; o<O; o++)                          //output feature
for (i=0; i<I; i++)                          //input feature
for (h=0; h<K; h++)                          //kernel weight
for (w=0; w<K; w++)                          //kernel height
        OUT[o][r][c]+=W[i][i][h][w]*
                IN[i][r+h]=[c+w];
```

Fig.6 Convolutional layer code
图6 卷积层代码

硬件设计中，对于卷积循环的优化方法主要有3种，分别是循环展开、循环交换和循环分块。循环展开决定了硬件加速的并行度，由于卷积运算主要是数组乘加运算，所以展开程度受到FPGA片上DSP和块随机存取存储器(Block Random Access Memory，BRAM)等资源数量限制；循环交换会改变运算先后顺序，可以将需要展开的维度放在内层嵌套，也可用于消除数据依赖；循环分块可以将大特征图划分为小特征图，解决FPGA片上资源有限导致无法计算和存储完整特征图的问题，采用小特征图还能够使不同网络层复用相同的硬件加速模块。

文献[47]将输入通道、输出通道并行展开，设计了经典的乘加树结构加速卷积运算，同时进行流水线操作，并设计了线性缓存器结构以确保卷积核的数据同时读取，实现的加速器性能达到了317.86 GOPS。Ma等[48]深入分析了3种卷积优化方法，并且使用设计变量量化分析加速方案，提出了一种最小化数据移动的加速策略，并设计了对应的硬件结构，在Altera Arria10 GX1150 FPGA上实现的VGG16网络达到了645.25 GOPS的吞吐量和47.97 ms的端到端延迟。文献[49]采用循环展开、分块、交换等策略，得到了能够增加数据复用、减少计算量和片外访存的最佳硬件结构。实验表明，在Intel Stratix V和Arria10 FPGA上部署VGG16模型的总吞吐量分别达到348 GOPS和715 GOPS。

2.3.2 脉动阵列

当硬件资源利用率较高时，FPGA的时序约束越来越难以满足，导致加速器运行频率较低，影响了性能提升。其中一个重要原因是当硬件资源利用率高时，FPGA的布局布线越来越困难，关键路径的时延影响了加速器的运行频率提升。脉动阵列架构采用深度流水化的处理单元(Processing Element，PE)间的短数据通路替代原来长广播、多扇入扇出的数据通路，能够在充分利用片上资源的同时保证较高的运行频率，典型的脉动阵列结构如图7所示。

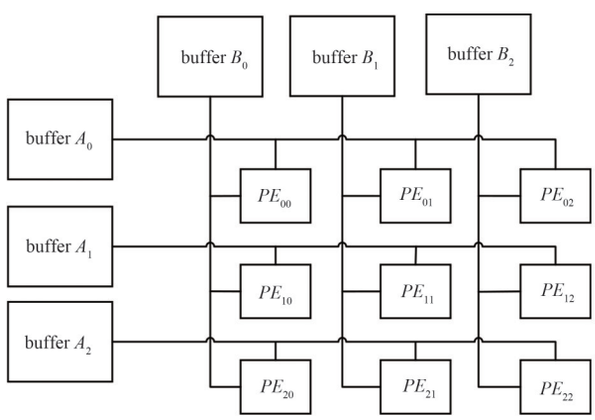

Fig.7 Diagram of systolic array
图7 脉动阵列结构示意图

文献[50]针对FPGA算力难以充分利用的问题，采用脉动阵列架构加速CNN运算，以最大化系统吞吐量为目标，将卷积运算6层嵌套循环中的3个维度映射到脉动阵列上实现并行加速，通过设计空间探索找出具有最高吞吐量的设计选择，实验结果表明，将AlexNet和VGG16部署在Intel Arria10 GX1150 FPGA分别实现了360.4 GOPS和460.5 GOPS的吞吐量。文献[51]采用类似的脉动阵列架构以确保在FPGA上实现高质量的布局、布线，同时PE引入了对剪枝后稀疏权重网络的支持，在Intel Arria10 GX1150 FPGA部署剪枝后的VGG16和ResNet-50实现了2 260 GOPS和1 210 GOPS的吞吐量。

2.3.3 完全片上映射

由于FPGA硬件资源限制，加速器大多采用可配置的通用PE阵列，神经网络的各层依次在PE阵列中执行。采用这种方式时，中间特征图存储在外部存储器中，增加了片外访存压力，影响数据的吞吐量提升。研究表明[52]，神经网络加速器的片外访存操作比参数计算的功耗更高，因此，减少加速器的片外访存是提高吞吐量和

能量效率的关键。

加速器设计的另一种思路是将神经网络完全映射到片上[53−54]，为每一层实现特定的硬件模块，可以针对不同层进行优化，这些层以流水线的方式执行。LI 等[55]针对大规模卷积神经网络计算和访存密集带来的硬件部署问题，提出了一种将所有层都部署到片上的方法，各层之间采取流水线结构以提高数据吞吐量，中间数据存储在片上缓存，从而避免了片外存储和重新加载中间特征图，只将权重数据存放在片外存储器中，并采用乒乓缓冲策略，将 AlexNet 模型部署到 Xilinx VC709 平台实现了 565.94 GOPS 的峰值性能和 391 FPS 的帧率。文献[56]设计了一个编译器，根据 TensorFlow 计算图作为输入，为每一层都实现相应的硬件处理单元，在 Intel Stratix 10 2800 FPGA 上实现了稀疏化的 Resnet−50，吞吐量是英伟达的 V100 GPU 的 4 倍。

这种将神经网络完全映射到片上的架构减少了外部存储的访问，但需要大量的片上内存资源来存储中间特征图和权重，对于较大的 CNN 模型可能会很容易耗尽所有可用的片上存储器，因此往往仅限于部署小型的神经网络。

2.3.4 多层融合

通常设计中，CNN 在加速器上逐层执行，上一层卷积计算结束后的输出特征图需要存放到片外存储器，下一层运算时再从片外存储器中读取数据作为输入特征图，导致了对片外存储器的频繁读写。多层融合的思想是将第一个输入层的数据读取到片上，然后多个卷积层在片上一起计算，仅将最后一层的输出特征图送到片外存储器。其目的是减少大型神经网络的中间数据在片外存储器上的频繁读写。

多层融合技术还可以利用相邻层之间的数据复用，进一步减少了对片外存储的访问。如图 8 所示，对于卷积运算，层 3 中的一个像素依赖于层 2 中的较小区域(例如卷积核大小)，这又取决于层 1 中的较大区域，形似一个金字塔结构。在层融合设计时，相邻像素的金字塔相互重叠，存在数据复用机会，在计算层 3 的相邻像素时，只需载入层 1 中分块右侧的一个新列(同时丢弃旧分块的最左侧列)即可。

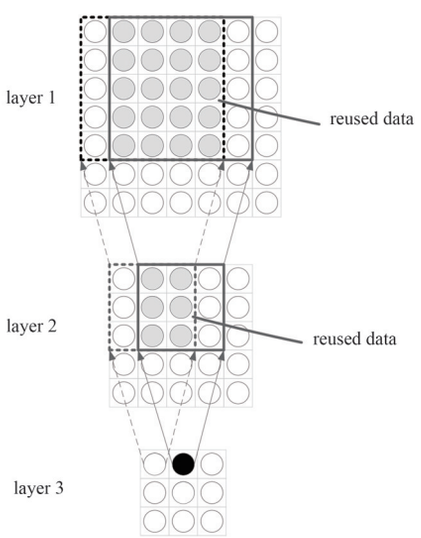

Fig.8 Diagram of layer fusion

图 8　层融合示意图

Alwani 等[57]提出 CNN 中多个卷积层之间数据存在金字塔型的依赖关系，将上层的输出特征直接作为下一层的输入特征，而不是先将上层输出传送到片外存储，从而能够显著减少访问片外存储的数据量。在 VGG19 网络模型的多层融合实现中，将所需的总数据量从 77MB 减少到 3.6MB(减少 95%)，只需额外的 362KB 片上存储即可，大大减少了片外访存需求。XIAO 等[58]为了减少加速器中片外访存带宽的需求，采用了多层融合的设计思想，并采用线性缓冲器结构读取输入特征图，充分利用数据复用，节省了片外访存的时间和功耗，提高了系统性能。

2.4 优化策略总结

FPGA 加速器的设计涉及到开发方式，算法优化策略，硬件实现优化策略等多方面的选择和权衡。网络模型优化和卷积算法优化都属于算法层面的优化策略，其目的在于转换模型算法和降低计算复杂度。而硬件实现优化与具体的硬件结构密切相关，需要考虑不同数据通路的时延和功耗等问题。表 3 总结了 FPGA 加速器的优化策略，对各优化策略的方法、优化目标以及局限性进行了简要描述。

加速器设计时往往需要综合使用各种优化策略，从而实现最优的实现效果。如文献[58]首先通过剪枝压缩网络模型，又采用 Winograd 算法加速卷积运算，同时还设计了多层融合架构降低片外访存导致的数据传输带宽需求。文献[59]选择针对轻量化网络模型 MobileNetv1 设计加速器，采用了剪枝、量化等方法压缩网络模型，最后将设计好的网络完全映射到片上，不需要在片外存储器中存储参数，提高了加速器的总体性能。

3　未来发展方向

3.1 训练加速器

目前神经网络加速器设计主要是针对推理任务，用于训练任务的加速器还比较少。卷积神经网络的训练任务对计算精确度的要求较高，且涉及到梯度计算和权重更新，需要反复的迭代运算，因此计算复杂度也更高。由于卷积神经网络的训练和推理采用相似的计算模式，未来，采用 FPGA 加速卷积神经网路训练可能会是研究热点之一。

表3 FPGA加速器优化策略总结

Table3 Summary of optimizing strategies on FPGA accelerator

type	strategy	method	optimization	limitations
network model optimization	quantization	reduce the bit width of weights or activation data	reduce storage requirements	binary quantized networks suffer from significant accuracy degradation
	pruning	set unimportant weights or activations to zero	compress the network model to accelerate computation	introduce sparsity
	lightweight network	use efficient computation modules to compress the model's computation	reduce the number of parameters and computation	increase the number of network layers
convolution algorithm optimization	FFT	convert convolution operations to frequency domain multiplication operations	reduce computational complexity	the effect is not obvious for small convolutional kernels
	Winograd	convert convolution operations to Winograd algorithm	reduce computational complexity	high demand for storage bandwidth
	GEMM	convert convolution operations to matrix multiplication	easy to implement and increase versatility	occupies additional storage resources
hardware implementation optimization	loop optimization	unroll, swap, and block the six-level nested loops	fully utilize hardware resources for parallel computing	large design space requiring modeling and analysis
	systolic array	employ a deeply pipelined short data path	optimize timing constraints to increase operating frequency	difficult to design
	fully on-chip, mapping	implement specific hardware modules for each layer of the network	reduce off-chip data movement	only suitable for small network models
	multi-layer fusion	complete multi-layer convolution operations within the FPGA	reduce off-chip data movement and increase data reuse	large design space

3.2 基于查找表的加速器设计

以往的CNN加速器设计中，通常会使用FPGA中的DSP资源进行乘法运算，而对查找表(LUT)资源并未充分利用。在二值神经网络运算中，将乘加运算转化为了同或运算，用丰富的LUT取代紧张的DSP乘法器资源，可以大大提高面积效率。二值神经网络和LUT结构的结合给FPGA加速神经网络提供了新的思路，如何提高推理精确度将成为相关研究的关键。

3.3 加速器中断技术

当把CNN部署到操作系统中时，有时会涉及到多个神经网络任务同时使用CNN硬件加速器的情况，这时就要通过调度来中断低优先级的任务，让高优先级的任务先执行。由于神经网络加速器在执行中会涉及巨大的参数量，所以中断的关键在于中断位置的选择，理想的中断位置应该确保较短的响应时间和较小的数据量存储。目前，针对神经网络加速器中断技术的研究较少，相信随着CNN的大量应用，相关技术也会不断取得突破。

4 结论

针对大规模FPGA神经网络加速应用，学术界在自动化部署、模型压缩、快速算法、并行计算、数据复用等方面取得了显著成果。FPGA加速器的运算吞吐量、功耗与CPU、GPU相比具有优势，目前互联网知名企业如微软、百度等的云数据中心都在大规模部署FPGA加速器，而在边缘设备中，FPGA也可以完成神经网络模型的加速推理。

人工智能技术的加速应用已经成为FPGA发展的战略方向之一。深度神经网络算法的发展日新月异，FPGA在硬件加速上具有独特的优势；但同时，FPGA及其开发方式也需要不断更新和进步，适应层出不穷的新算法、新应用和新要求，才能在人工智能时代创造新的辉煌。

参考文献：

[1] HUBEL D H,WIESEL T N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex[J]. The Journal of Physiology, 1962,160(1):106-154. doi:10.1113/jphysiol.1962.sp006837.

[2] LECUN Y,BOTTOU L,BENGIO Y,et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998,86(11):2278-2324. doi:10.1109/5.726791.

[3] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017,60(6):84-90. doi:10.1145/3065386.

[4] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[EB/OL]. (2014-09-04) [2022-04-14]. https://arxiv.org/abs/1409.1556.

[5] HE Kaiming,ZHANG Xiangyu,REN Shaoqing, et al. Deep residual learning for image recognition[C]// 2016 IEEE Conference on Computer Vision and Pattern Recognition(CVPR). Las Vegas:IEEE, 2016:770-778. doi:10.1109/CVPR.2016.90.

[6] IANDOLA F N, HAN S, MOSKEWICZ M W, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and<0.5 MB model size[EB/OL]. (2016-02-24) [2022-04-14]. https://arxiv.org/abs/1602.07360.

[7] HOWARD A G,ZHU M L,CHEN B,et al. MobileNets: efficient convolutional neural networks for mobile vision applications[EB/OL]. (2017-04-17) [2022-04-14]. https://arxiv.org/abs/1704.04861.

[8] SANDLER M, HOWARD A, ZHU M L, et al. MobileNetV2: inverted residuals and linear bottlenecks[C]// 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City: IEEE, 2018: 4510-4520. doi: 10.1109/CVPR. 2018. 00474.

[9] HOWARD A, SANDLER M, CHEN B, et al. Searching for MobileNetV3[C]// 2019 IEEE/CVF International Conference on Computer Vision(ICCV). Seoul:IEEE, 2019:1314-1324. doi:10.1109/ICCV.2019.00140.

[10] ZHANG Xiangyu,ZHOU Xinyu,LIN Mengxiao,et al. ShuffleNet: an extremely efficient convolutional neural network for mobile devices[C]// 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City:IEEE, 2018:6848-6856. doi:10.1109/CVPR.2018.00716.

[11] MA Ningning, ZHANG Xiangyu, ZHENG Haitao, et al. ShuffleNet V2: practical guidelines for efficient CNN architecture [C]// Computer Vision-ECCV 2018. Cham:Springer International Publishing, 2018:122-138. doi:10.1007/978-3-030-01264-9_8.

[12] CONG J, LIU B, NEUENDORFFER S, et al. High-level synthesis for FPGAs: from prototyping to deployment[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2011, 30(4): 473-491. doi: 10.1109/TCAD. 2011. 2110592.

[13] ZHANG Chen, FANG Zhenman, ZHOU Peipei, et al. Caffeine: towards uniformed representation and acceleration for deep convolutional neural networks[C]// 2016 IEEE/ACM International Conference on Computer-Aided Design(ICCAD). Austin: IEEE, 2016:1-8. doi:10.1145/2966986.2967011.

[14] VENIERIS S I, BOUGANIS C S. fpgaConvNet: a framework for mapping convolutional neural networks on FPGAs[C]// 2016 IEEE the 24th Annual International Symposium on Field-Programmable Custom Computing Machines(FCCM). Washington: IEEE, 2016:40-47. doi:10.1109/FCCM.2016.22.

[15] UMUROGLU Y,FRASER N J,GAMBARDELLA G,et al. FINN:a framework for fast,scalable binarized neural network inference[C]// Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey: Association for Computing Machinery, 2017:65-74. doi:10.1145/3020078.3021744.

[16] DUARTE J, HAN S, HARRIS P, et al. Fast inference of deep neural networks in FPGAs for particle physics[J]. Journal of Instrumentation, 2018(13):P07027. doi:10.1088/1748-0221/13/07/P07027.

[17] PLAGWITZ P,HANNIG F,STRÖBEL M,et al. A safari through FPGA-based neural network compilation and design automation flows[C]// 2021 IEEE the 29th Annual International Symposium on Field-Programmable Custom Computing Machines(FCCM). Orlando:IEEE, 2021:10-19. doi:10.1109/FCCM51124.2021.00010.

[18] AMD. Vitis AI[EB/OL]. [2022-04-14]. https://china.xilinx.com/products/design-tools/vitis/vitis-ai.html.

[19] MOREAU T, CHEN Tianqi, VEGA L, et al. VTA: an open hardware-software stack for deep learning[EB/OL]. (2018-07-11) [2022-04-14]. https://arxiv.org/abs/1807.04188v2.

[20] MathWorks. Deep learning processor IP core[EB/OL]. [2022-04-14]. https://www. mathworks. cn/help/deep-learning-hdl/ug/deep-learning-processor-ip-core.html.

[21] SUDA N,CHANDRA V,DASIKA G,et al. Throughput-optimized OpenCL-based FPGA accelerator for large-scale convolutional neural networks[C]// Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey:Association for Computing Machinery, 2016:16-25. doi:10.1145/2847263.2847276.

[22] 孙小坚,林瑞全,方子卿,等. 基于FPGA加速的低功耗的MobileNetV2网络识别系统[J]. 计算机测量与控制, 2023,31(5): 221-227,234. (SUN Xiaojian,LIN Ruiquan,FANG Ziqing,et al. Low-power mobileNetV2 network identification system based on FPGA acceleration[J]. Computer Measurement & Control, 2023,31(5):221-227,234.) doi:10.16526/j.cnki.11-4762/tp.2023.05.033.

[23] LIANG Shuang, YIN Shouyi, LIU Leibo, et al. FP-BNN: binarized neural network on FPGA[J]. Neurocomputing, 2018(275): 1072-1086. doi:10.1016/j.neucom.2017.09.046.

[24] HAN S, MAO H Z, DALLY W J. Deep compression: compressing deep neural networks with pruning, trained quantization and

Huffman coding[EB/OL]. (2015-10-01) [2022-04-14]. https://arxiv.org/abs/1510.00149.

[25] ALBERICIO J,JUDD P,HETHERINGTON T,et al. Cnvlutin: ineffectual-neuron-free deep neural network computing[J]. ACM SIGARCH Computer Architecture News, 2016,44(3):1-13. doi:10.1145/3007787.3001138.

[26] NURVITADHI E,VENKATESH G,SIM J,et al. Can FPGAs beat GPUs in accelerating next-generation deep neural networks [C]// Proceedings of the 2017 ACM/SIGDA international symposium on field-programmable gate arrays. Monterey:Association for Computing Machinery, 2017:5-14. doi:10.1145/3020078.3021740.

[27] VÉSTIAS M. Efficient design of pruned convolutional neural networks on FPGA[J]. Journal of Signal Processing Systems, 2021, 93(5):531-544. doi:10.1007/s11265-020-01606-2.

[28] FAN Yingbo,PANG Wei,LU Shengli. HFPQ: deep neural network compression by hardware-friendly pruning-quantization[J]. Applied Intelligence, 2021,51(10):7016-7028. doi:10.1007/s10489-020-01968-x.

[29] BAI Lin, ZHAO Yiming, HUANG Xinming. A CNN accelerator on FPGA using depthwise separable convolution[J]. IEEE Transactions on Circuits and Systems II—Express Briefs, 2018,65(10):1415-1419. doi:10.1109/TCSII.2018.2865896.

[30] HUANG Qijing, WANG Dequan, DONG Zhen, et al. CoDeNet: efficient deployment of Input-Adaptive object detection on embedded FPGAs[C]// The 2021 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Virtual Event: Association for Computing Machinery, 2021:206-216. doi:10.1145/3431920.3439295.

[31] 蒋璨. 一种关于卷积神经网络的加速结构[D]. 成都:电子科技大学, 2020. (JIANG Can. An acceleration structure of convolutional neural network[D]. Chengdu,China:University of Electronic Science and Technology of China, 2020.)

[32] 刘志强. 基于FPGA的卷积神经网络加速器关键技术研究[D]. 长沙:国防科技大学, 2019. (LIU Zhiqiang. Research on FPGA-based accelerator design for convolutional neural networks[D]. Changsha, China: National University of Defense Technology, 2019.) doi:10.27052/d.cnki.gzjgu.2019.000033.

[33] KO J H,MUDASSAR B,NA T,et al. Design of an energy-efficient accelerator for training of convolutional neural networks using frequency-domain computation[C]// 2017 the 54th ACM/EDAC/IEEE Design Automation Conference(DAC). Austin:IEEE, 2017: 1-6. doi:10.1145/3061639.3062228.

[34] ZHANG Chi,PRASANNA V. Frequency domain acceleration of convolutional neural networks on CPU-FPGA shared memory system[C]// Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey: Association for Computing Machinery, 2017:35-44. doi:10.1145/3020078.3021727.

[35] ZENG Hanqing,CHEN Ren,ZHANG Chi,et al. A framework for generating high throughput CNN implementations on FPGAs[C]// Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey:Association for Computing Machinery, 2018:117-126. doi:10.1145/3174243.3174265.

[36] WINOGRAD S. Arithmetic complexity of computations Philadelphia[M]. Philadelphia: Society for Industrial and Applied Mathematics, 1980.

[37] PODILI A,ZHANG Chi,PRASANNA V. Fast and efficient implementation of Convolutional Neural Networks on FPGA[C]// 2017 IEEE the 28th International Conference on Application-specific Systems, Architectures and Processors(ASAP). Seattle:IEEE, 2017: 11-18. doi:10.1109/ASAP.2017.7995253.

[38] SHEN Junzhong,HUANG You,WANG Zelong,et al. Towards a uniform template-based architecture for accelerating 2D and 3D CNNs on FPGA[C]// Proceedings of the 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey:Association for Computing Machinery, 2018:97-106. doi:10.1145/3174243.3174257.

[39] AYDONAT U,O'CONNELL S,CAPALIJA D,et al. An OpenCL™ deep learning accelerator on arria 10[C]// Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey: Association for Computing Machinery, 2017:55-64. doi:10.1145/3020078.3021738.

[40] LU Liqiang, LIANG Yun, XIAO Qingcheng, et al. Evaluating fast algorithms for convolutional neural networks on FPGAs[C]// 2017 IEEE the 25th Annual International Symposium on Field-Programmable Custom Computing Machines(FCCM). Napa: IEEE, 2017:101-108. doi:10.1109/FCCM.2017.64.

[41] 卢丽强,郑思泽,肖倾城,等. 面向卷积神经网络的FPGA设计[J]. 中国科学:信息科学, 2019,49(3):277-294. (LU Liqiang, ZHENG Size, XIAO Qingcheng, et al. Accelerating convolutional neural networks on FPGAs[J]. Science China Information Sciences, 2019,49(3):277-294.)

[42] GONG Yu,XU Zhihan,HE Zhezhi,et al. N3H-Core: neuron-designed neural network accelerator via FPGA-based heterogeneous computing cores[C]// Proceedings of the 2022 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Virtual Event:Association for Computing Machinery, 2022:112-122. doi:10.1145/3490422.3502367.

[43] ZHANG Wentai, JIANG Ming, LUO Guojie. Evaluating low-memory GEMMs for convolutional neural network inference on

FPGAs[C]// 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines(FCCM). Fayetteville:IEEE, 2020:28-32. doi:10.1109/FCCM48280.2020.00013.

[44] NIU Yue,KANNAN R,SRIVASTAVA A,et al. Reuse kernels or activations? a flexible dataflow for low-latency spectral CNN acceleration[C]// Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Seaside: Association for Computing Machinery, 2020:266-276. doi:10.1145/3373087.3375302.

[45] HUANG Chengcheng,DONG Xiaoxiao,LI Zhao,et al. Efficient stride 2 winograd convolution method using unified transformation matrices on FPGA[C]// 2021 International Conference on Field-Programmable Technology(ICFPT). Auckland:IEEE, 2021:1-9. doi:10.1109/ICFPT52863.2021.9609907.

[46] 吴艳霞,梁楷,刘颖,等. 深度学习FPGA加速器的进展与趋势[J]. 计算机学报, 2019,42(11):2461-2480. (WU Yanxia,LIANG Kai,LIU Ying,et al. The progress and trends of FPGA-based accelerators in deep learning[J]. Chinese Journal of Computers, 2019,42(11):2461-2480.) doi:10.11897/SP.J.1016.2019.02461.

[47] 秦华标,曹钦平. 基于FPGA的卷积神经网络硬件加速器设计[J]. 电子与信息学报, 2019,41(11):2599-2605. (QIN Huabiao, CAO Qinping. Design of convolutional neural networks hardware acceleration based on FPGA[J]. Journal of Electronics & Information Technology, 2019,41(11):2599-2605.) doi:10.11999/JEIT190058.

[48] MA Yufei,CAO Yu,VRUDHULA S,et al. Optimizing loop operation and dataflow in FPGA acceleration of deep convolutional neural networks[C]// Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey:Association for Computing Machinery, 2017:45-54. doi:10.1145/3020078.3021736.

[49] MA Y F,CAO Y,VRUDHULA S,et al. Optimizing the convolution operation to accelerate deep neural networks on FPGA[J]. IEEE Transactions on Very Large Scale Integration(VLSI) Systems, 2018,26(7):1354-1367. doi:10.1109/TVLSI.2018.2815603.

[50] WEI X C,YU C H,ZHANG P,et al. Automated systolic array architecture synthesis for high throughput CNN inference on FPGAs [C]// Proceedings of the 54th Annual Design Automation Conference 2017. Austin:Association for Computing Machinery, 2017: 29. doi:10.1145/3061639.3062207.

[51] LIU L Q,BROWN S. Leveraging fine-grained structured sparsity for CNN inference on systolic array architectures[C]// 2021 the 31st International Conference on Field-Programmable Logic and Applications(FPL). Dresden: IEEE, 2021: 301-305. doi: 10.1109/FPL53798.2021.00060.

[52] HOROWITZ M. 1.1 Computing's energy problem (and what we can do about it)[C]// 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC). San Francisco, CA, USA: IEEE, 2014: 10-14. doi: 10.1109/ISSCC.2014.6757323.

[53] LIU Zhiqiang, DOU Yong, JIANG Jingfei, et al. Throughput-Optimized FPGA accelerator for deep convolutional neural networks[J]. ACM Transactions on Reconfigurable Technology and Systems, 2017,10(3):17. doi:10.1145/3079758.

[54] NGUYEN D T,NGUYEN T N,KIM H,et al. A high-throughput and power-efficient FPGA implementation of YOLO CNN for object detection[J]. IEEE Transactions on Very Large Scale Integration(VLSI) Systems, 2019, 27(8): 1861-1873. doi: 10.1109/TVLSI.2019.2905242.

[55] LI Huimin,FAN Xitian,JIAO Li,et al. A high performance FPGA-based accelerator for large-scale convolutional neural networks [C]// 2016 the 26th International Conference on Field Programmable Logic and Applications(FPL). Lausanne:IEEE, 2016:1-9. doi:10.1109/FPL.2016.7577308.

[56] HALL M,BETZ V. HPIPE: heterogeneous layer-pipelined and sparse-aware CNN inference for FPGAs[C]// The 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays(FPGA'20). New York: Association for Computing Machinery, 2020:320. doi:10.1145/3373087.3375380.

[57] ALWANI M,CHEN H,FERDMAN M,et al. Fused-layer CNN accelerators[C]// 2016 the 49th Annual IEEE/ACM International Symposium on Microarchitecture(MICRO). Taipei,Taiwan,China:IEEE, 2016:1-12. doi:10.1109/MICRO.2016.7783725.

[58] XIAO Qingcheng,LIANG Yun,LU Liqiang,et al. Exploring heterogeneous algorithms for accelerating deep convolutional neural networks on FPGAs[C]// 2017 the 54th ACM/EDAC/IEEE Design Automation Conference(DAC). Austin:IEEE, 2017:1-6. doi: 10.1145/3061639.3062244.

[59] MENG J,VENKATARAMANAIAH S K,ZHOU C T,et al. FixyFPGA:efficient FPGA accelerator for deep neural networks with high Element-Wise sparsity and without external memory access[C]// 2021 the 31st International Conference on Field-Programmable Logic and Applications(FPL). Dresden:IEEE, 2021:9-16. doi:10.1109/FPL53798.2021.00010.

（下转第1167页）