基于 FPGA 的卷积神经网络并行加速设计

龚豪杰[1,2]，周 海[1+]，冯水春[1]

（1. 中国科学院国家空间科学中心 复杂航天系统电子信息技术重点实验室，北京 101499；
2. 中国科学院大学 计算机科学与技术学院，北京 101408）

摘 要：为提升在资源、功耗受限的嵌入式平台上运行的深度卷积网络算法的速度和能效，提出一种基于现场可编程门阵列（FPGA）的卷积并行加速方案。利用卷积层与批归一化（batch normalization，BN）层融合减少计算复杂度；利用数据分片减少片上存储消耗；利用数据复用、并行计算提升运算速度，减少系统硬件开销；利用设计空间探索找到最符合硬件资源约束的计算并行度。实验结果表明，在 100 MHz 的工作频率下，加速器的峰值计算性能可以达到 52.56 GFLOPS，性能是 CPU 的 4.1 倍，能耗仅为 GPU 的 9.9%，与其它 FPGA 方案相比综合性能有一定的提升。

关键词：卷积神经网络；现场可编程门阵列；批归一化；并行计算；数据复用

中图法分类号：TP332 **文献标识号**：A **文章编号**：1000-7024（2022）07-1872-07

doi：10.16208/j.issn1000-7024.2022.07.010

Convolutional neural network parallel acceleration design based on FPGA

GONG Hao-jie[1,2]，ZHOU Hai[1+]，FENG Shui-chun[1]

（1. Key Laboratory of Electronic Information Technology for Complex Aerospace Systems，National Space Science Center，Chinese Academy of Sciences，Beijing 101499，China；2. School of Computer Science and Technology，University of Chinese Academy of Sciences，Beijing 101408，China）

Abstract：To improve the speed and energy efficiency of deep convolutional network algorithms running on embedded platforms with limited resources and power consumption，a convolutional parallel acceleration scheme based on field programmable gate array（FPGA）was proposed. Convolutional layer and batch normalization（BN）layer fusion was used to reduce computational complexity. Data fragmentation was used to reduce on-chip storage consumption. Data multiplexing and parallel calculation were utilized to increase the operation speed and to reduce the system hardware overhead. Design space exploration was used to find the computational parallelism that best met the hardware resource constraints. Experimental results show that at the working frequency of 100 MHz，the peak computing performance of the accelerator can reach 52.56 GFLOPS，which is 4.1 times better than the performance of the CPU and consumes only 9.9% of the GPU. Compared with other FPGA solutions，the overall performance has certain improvement.

Key words：convolution neural network；FPGA；batch normalization；parallel computing；data reuse

0 引 言

随着卷积神经网络[1]算法在目标检测、目标跟踪[2]等领域的发展，卷积神经网络算法正在逐步取代传统算法。针对未来复杂多变的应用环境，空天领域也一直在推进智能化发展，发挥智能技术在空天领域建设中的引领作用[3]，但由于卫星所处环境的特殊性，载荷在功耗、体积、材质等方面都有严格的限制，导致其存储、计算资源的稀缺性，如何将卷积神经网络模型部署到资源受限的嵌入式环境中成为了亟待解决的问题。

FPGA 具有可重构、低功耗、可定制以及高性能等优势[4]，可以高度并行地执行卷积神经网络模型中大量重复的乘加运算，并以较低的功耗完成高精度分类任务[5]。因此，FPGA 在加速卷积神经网络方面具有自己独特的优势[6]。

收稿日期：2021-03-26；**修订日期**：2021-06-01

基金项目：中国科学院青年创新促进会基金项目（E0293401）

作者简介：龚豪杰（1996-），男，湖北武汉人，硕士研究生，研究方向为卷积神经网络加速；+通讯作者：周海（1987-），男，安徽蚌埠人，博士，副研究员，硕士生导师，研究方向为目标探测与识别、遥感图像处理及智能无人系统技术；冯水春（1973-），女，湖北浠水人，硕士生导师，高级工程师，研究方向为目标探测与识别、遥感图像处理。E-mail：zhouhai@nssc.ac.cn

基于 FPGA 的卷积神经网络优化加速是一个复杂的过程，设计方法也多种多样。Wang D 设计流水线卷积计算内核来加速卷积计算[7]；Eyeriss 团队提出 RS 数据流以提高卷积计算的并行性[8]；Liu 等提出了一种大规模并行框架来提升卷积计算的吞吐量[9]；张榜等通过双缓冲和流水线技术对卷积进行优化[10]；Suda 等提出一种系统化的设计空间探索算法以最大化吞吐量[11]。目前的研究虽然提出了多种技术来加速卷积神经网络，但都专注于某一方面的性能优化，并没有完全发挥 FPGA 的并行计算潜能。

本研究提出了一种卷积并行加速系统，利用层融合、数据分片、缓存设计、并行设计等多种优化策略加速卷积计算，与 CPU、GPU 平台以及其它 FPGA 平台设计方案相比综合性能有一定的提升。

1　卷积神经网络与层融合

1.1　卷积神经网络

为了提升卷积神经网络的性能，处理更复杂的任务场景，卷积神经网络的层级逐渐加深，结构也更加复杂。2012 年，AlexNet 奠定了卷积神经网络在图像处理领域的地位，网络层数只有 8 层；2014 年，GoogLeNet 通过不断复用 inception 结构来提升性能，卷积层数达 22 层；2016 年，ResNet[12] 利用残差结构将网络深度提升到 152 层。随着网络深度的增加，其特征表达能力会有一个实质性的突破。现如今，数千层的卷积神经网络已非常常见。

然后，深层网络也带来新的问题，首当其冲的就是训练难度加大，出现梯度消失和梯度爆炸，以及容易过拟合。因此，如今的深层网络通常都会在卷积层后加一层 BN 层，BN 层可以有效加快网络的训练过程，防止梯度消失和梯度爆炸以及过拟合问题。

图 1 为 ResNet（残差网络）的基础残差模块，包含两个主要部分：卷积层和 BN 层。卷积层是图形特征有效的提取器，BN 层则可以解决隐藏层协变量偏移问题，使得非线性变换函数的输入值落入对输入比较敏感的区域，以防止梯度消失，并加快训练过程。

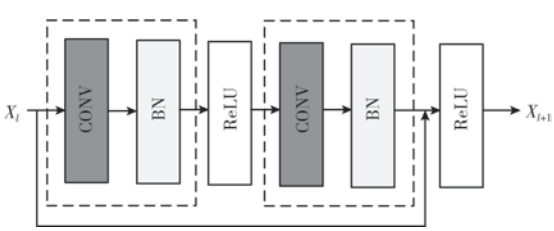

图 1　残差模块

图 2 为卷积层的基本运算——卷积运算。卷积时将卷积核与输入特征图矩阵的局部做卷积运算得到输出特征图，局部大小取决于卷积核的大小，也即感受野的大小。其计算公式如下所示

$$X_{lj} = \sum_{i \in M_j} X_i^{l-1} \cdot W_{ij}^{l} \tag{1}$$

式中：X_{lj} 表示第 l 层网络的第 j 个通道；X_i^{l-1} 为上一层的输入；W_{ij}^{l} 为卷积权值。卷积窗口每完成一次卷积，会向右移动一个步长，移动至行末时再向下移动一个步长并从左侧开始新的一行卷积，直至完成整张特征图的卷积运算。

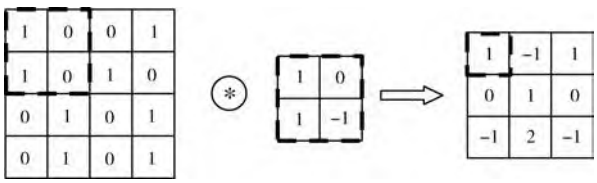

图 2　卷积运算

本文采用的卷积网络参数见表 1。该网络结构包含 6 层卷积层，卷积核为 3×3，移动步长为 1。

表 1　卷积网络配置参数

卷积层	1	2	3	4	5	6
输入通道	3	64	64	128	128	256
输出通道	64	64	128	128	256	256
特征图行	112	56	56	28	28	14
特征图列	112	56	56	28	28	14
核尺寸	3	3	3	3	3	3

1.2　层融合

使用 BN 层是为了解决网络训练过程中的梯度消失和梯度爆炸问题并提高泛化性能，然而在前向推理过程中，使用训练后的数值固定的尺度因子 γ 和偏移因子 β，以及每个通道的平均值 μ 与方差 σ^2 的无偏估计计算 BN 层输出。卷积层与 BN 层在 FPGA 实现时很难共用同一套处理资源，导致 BN 层带来较大的额外资源开销，为了提升网络推理速度和降低资源消耗，可以将 BN 层与卷积层进行融合。BN 层的计算公式如下

$$\hat{x}_i = \gamma \frac{x_i - \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \tag{2}$$

式中：μ 表示输入数据的均值，σ^2 表示输入数据的方差。ε 是分母添加的一个很小的值，防止分母为 0。γ 表示尺度因子，β 表示偏移因子，是模型在训练过程中自动学习到的两个参数。将卷积式（3）带入到 BN 层计算式（2）中并展开得到式（4）

$$Y = W \cdot X + b \tag{3}$$

$$y_i = \gamma \frac{x_i - \mu}{\sqrt{\sigma^2 + \varepsilon}} W + \beta + (b - \mu) \frac{\gamma}{\sqrt{\sigma^2 + \varepsilon}} \tag{4}$$

由于 BN 层本身带有偏移因子，故卷积层的偏置 b 可以省略，因此融合后新卷积层的权重参数 W' 和偏置参数 b' 整理如下

$$W' = \frac{\gamma}{\sqrt{\sigma^2 + \varepsilon}} W \tag{5}$$

$$b' = \beta - \frac{\gamma\mu}{\sqrt{\sigma^2 + \varepsilon}} \tag{6}$$

融合卷积计算公式如下

$$Y' = W'X + b' \tag{7}$$

基于以上推理，将 BN 层在推理阶段完全融入到卷积计算中，而没有任何的精度损失，通过这种方式可以有效降低硬件计算量，减少资源消耗，加快推理过程。

2 基于 FPGA 的硬件设计

2.1 优化层级分析

随着网络的深度加深，结构更加复杂，将导致大量的特征图输入和权重输入，而片上存储器由于资源的限制不能存储所有的输入数据，因此需要较大的外部存储。本系统涉及 3 个层级存储模块：外部存储、片上缓存和寄存器级数据处理单元。外部存储器存储完整的网络模型数据，数据传输延迟是该部分主要的性能瓶颈，通过数据复用减少数据传输；片上存储器存储当前网络层的模型数据，存储资源限制是该部分主要的性能瓶颈，通过数据分片降低资源消耗；寄存器级存储单元存储处于计算状态的数据，计算并行性是该部分主要的性能瓶颈，通过循环展开提升计算并行性。系统数据调度如图 3 所示。

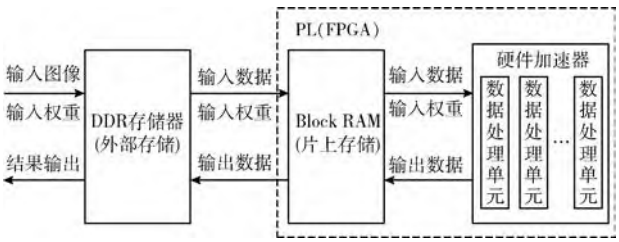

图 3 数据流调度

输入数据流经过处理器端的处理和预加载，将输入图像和权重存入外部存储器 DDR。PL 端通过数据总线从外部存储读入当前卷积层和 BN 层的融合数据到片上缓存，然后输入到硬件加速模块，数据处理单元完成卷积运算，输出数据返回到片上存储，继续返回到外部存储器作为结果输出或作为下一层网络的输入。

2.2 加速器总体架构设计

加速器架构如图 4 所示，主要包括片上缓存、数据处理单元（PE）和控制器。片上缓存通过 AXI 总线与外部存储交流。

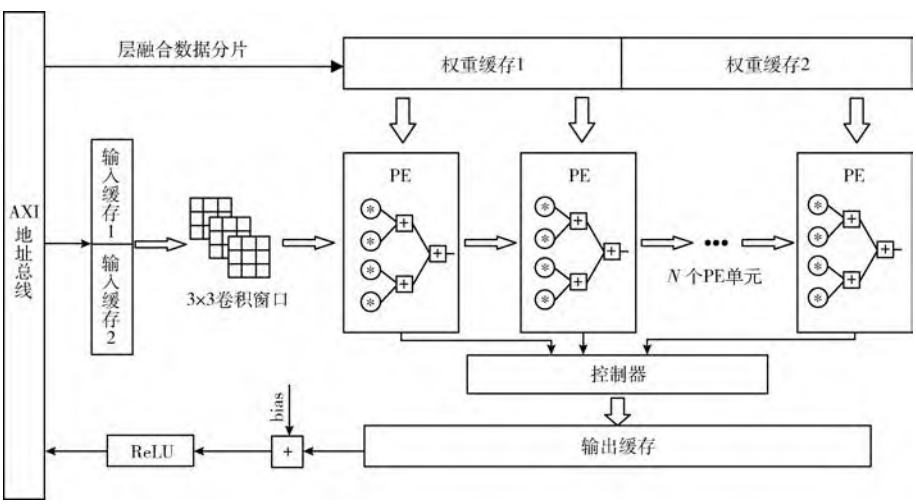

图 4 加速器总体架构

由于卷积神经网络参数量大，FPGA 片上存储资源有限，无法存储所有的参数。数据缓存操作包括数据加载和数据处理两部分，只有完成了数据加载之后才能对其中的数据进行后续处理。为了避免数据传输导致数据处理单元闲置的情况，本文采用双缓存机制，以图中权重缓存为例，当权重缓存 1 进行数据加载时，权重缓存 2 进行数据处理；当权重缓存 2 进行数据加载时，权重缓存 1 则进行数据处理。这种 ping-pong RAM 的工作模式可以保证在整个处理过程中，数据处理单元始终处于工作状态。同理，输入缓存也是如此，数据加载和数据处理完全并行操作，使数据处理模块的计算能力得到最大化利用。

卷积计算主要是权重矩阵与输入特征图进行乘累加运算得到输出特征图。相比于所有通道的输入卷积窗口参与卷积生成最终的输出通道数据，本设计使用部分通道的输入卷积窗口与所有的卷积核卷积生成中间结果，直到所有输入通道计算完成并累加到输出缓存生成最终输出通道数据，可以有效增加输入特征图的数据复用，减少其在 BRAM 和外部存储器之间的重复传输。

加速器处理流程如下，卷积层与 BN 层融合权重通过数据分片设计，输入到权重缓存。输入缓存生成多个通道

的 3×3 卷积窗口，与权重同时输入到 PE 单元进行卷积运算，N 个 PE 单元完全并行处理，控制器计算输出通道的索引值，通过索引值将运算结果输出到相应的输出缓存地址，由于输入特征图和权重进行了分片，计算结果是中间值，需要在输出缓存上进行累加。当所有的输入特征图计算完毕，输出缓存添加偏置，并通过控制器判断是否进行激活处理，激活函数使用 ReLU 函数。

2.3　层融合分片设计

本文采用的层融合分片设计如图 5 所示，矩阵代表完整的层融合输入权重矩阵，其中列数表示输入通道数，行数表示输出通道数，每个灰色框图表示一个 3×3 卷积核，实线框中矩阵即为数据分片后权重矩阵。本文在两个维度上进行数据分片，分别是输入通道 N 和输出通道 M。在后续的卷积并行计算中，N 表示输入通道并行度，M 表示输出通道并行度。输入通道并行和输出通道并行在不同程度上影响计算资源的开销。在两个维度上设置分片参数，提升了数据分片和计算并行度的灵活性，能更充分利用平台的硬件资源。

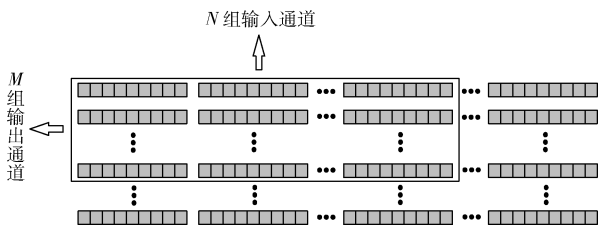

图 5　层融合数据分片

2.4　数据处理单元设计

数据处理单元结构如图 6 所示，每个虚线框代表一个数据处理单元，其内部为 N 个输入通道之间并行运算，需要 N 个不同的卷积核。虚线框之间为 M 个输出通道之间的并行运算，需要 M 组不同的卷积核。因此输入通道 N 和输出通道 M 决定了卷积计算的并行度。当 N 和 M 分别等于当前层的输入特征图数量和输出特征图数量时，卷积并行性达到理论最大值，但也会消耗大量的硬件资源。因此，可以利用设计空间探索，合理分配并行粒度以平衡计算性能和资源消耗。

2.5　数据复用和卷积计算优化

数据复用与卷积优化方案如图 7 所示，输入特征图按行顺序一行一行输入，每个像素值从第一次参与卷积计算到最后一次参与卷积计算，3×3 卷积窗口会走过两个输入特征图行尺寸的距离，为了最大化输入特征值的复用机率，需要 2 倍特征图行尺寸大小的线性缓存存储访问过的特征值，这样当像素第二次进入卷积窗口时可以直接从线性缓存中顺序读取，降低了数据访问延迟。

本文采用 3×3 卷积核，卷积步长为 1。卷积窗口向右

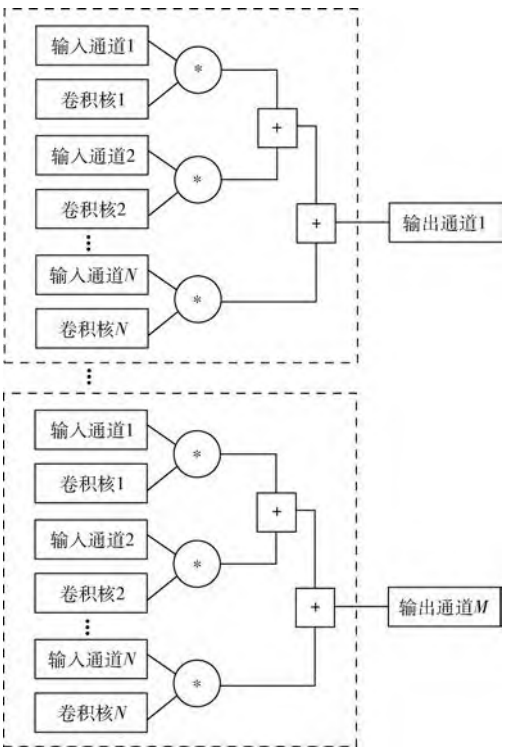

图 6　数据处理单元结构

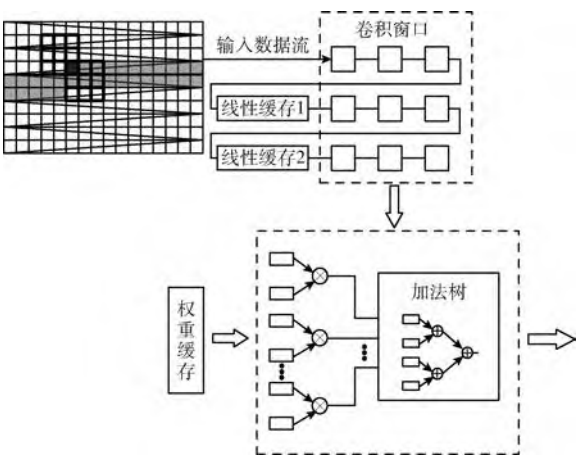

图 7　3×3 卷积计算设计

滑动一个步长，只有最右边的一列需要更新，复用第一列和第二列的输入像素值，数据复用比例达 66.7%。因此，本文采用两个线性缓存，分别是线性缓存 1 和线性缓存 2，每个线性缓存的大小为输入特征图行尺寸大小。每个读数据阶段线性缓存工作流程如下：

（1）输入缓存输入一个数据到卷积窗口；

（2）同时线性缓存 1 按从左到右顺序存储该输入数据；

（3）如果线性缓存 1 数据存满，则从左到右依次更新数据，并将原始数据存入线性缓存 2；

（4）如果线性缓存 2 数据存满，则从左到右依次更新

数据，并丢弃原始数据。

如图 7 所示，线性缓存 2 负责更新卷积窗口第一行，线性缓存 1 负责更新第二行，每次从输入缓存读取的数据则负责更新第三行。卷积窗口的三行数据同时更新，读数据阶段需要两个时钟周期，一个时钟周期读地址，一个时钟周期读取数据，通过完全流水线结构，可以在每个时钟周期生成一个卷积窗口进行后续计算。

生成的卷积窗口与权重同时输入到卷积计算单元，卷积计算单元的乘法运算完全展开，之后输入到加法树，通过乘法阵列-加法树的模式最大化核卷积计算的并行性。

2.6 设计空间探索

由于不同的 FPGA 平台有不同的资源限制，为了更好匹配平台的资源限制，提升硬件资源利用率，本文提出了如表 2 所示的设计空间探索算法。

表 2 设计空间探索伪代码

设计空间探索伪代码
输入：网络参数和硬件资源约束 过程： (1)初始化层内并行度 N，层间并行度 M； (2) for i from 1 to C_{out} (3) for j from 1 to C_{in} (4) 根据当前参数 i 和 j 运行网络模型； (5) if 资源开销满足硬件资源约束 then (6) if 网络性能有提升 then (7) 使用 i 和 j 更新 M 和 N； (8) end if (9) end if (10) end for (11)end for 输出：网络并行度 N 和 M

如伪代码所示，C_{out} 表示输出特征图数量，C_{in} 表示输入特征图数量。算法的输入是当前网络的参数和硬件资源约束，输出是网络的并行度，包括输入通道并行度 N 和输出通道并行度 M。首先，初始化参数 N 和 M，根据当前网络并行度参数运行网络，如果资源开销满足硬件资源约束，则继续判断网络的性能是否有提升，如果有提升则通过当前的并行度参数更新 N 和 M，否则丢弃当前结果进入下一次迭代，直到找到最优的网络并行度参数，N 和 M 确定了计算资源的使用量。通常情况下，卷积网络的通道数会随着网络加深而成倍增长，且本文卷积加速模块在不同卷积层间是复用的，因此在充分利用硬件资源情况下确定的初始卷积层的并行粒度可以沿用到后续网络层中。网络的并行粒度即为 $N\times M$。

3 实验结果

3.1 实验环境

本次实验使用 Xilinx 公司的 ZCU104 开发板作为实验平台，该芯片含有 PL 端 1728 个 DSP48E 单元，38 Mb RAM，以及 PS 端 2 GB DDR4 存储器件，完全可以满足本实验的硬件要求。实验数据位宽设定为浮点 32 位，FPGA 主频设定为 100 MHz。

此外还在 CPU 和 GPU 平台上进行计算性能测试，测试卷积网络模型为表 1 的卷积层 2，CPU 采用的是 Inter Core i5-4210H，主频为 2.90 GHz；GPU 采用的是 NVIDIA GTX1080ti，显存容量为 11 GB，核心频率为 1.582 GHz。

3.2 功能验证

本实验基于 Vivado_hls 2020.1 和 Vivado 2020.1 软件进行开发和仿真，仿真波形如图 8 所示。卷积的输入、权值和偏置都为 32 位浮点数据，输入时钟频率为 100 MHz，时钟波形如图 8 中矩形框 2 所示。图 8 所示波形包括待测加速器的波形信号和 Test Bench Signals，为验证加速器功能正确性，这里关注这两个信号的输出波形，即 ofm 波形。当输出波形的写使能信号 ofm_we0 和片选信号 ofm_ce0 置高位 1 时，输出数据信号 ofm_d0 有效。矩形框 1 为待测加速器的输出数据信号，矩形框 3 为理论输出数据信号，可以看到，加速器的卷积输出结果与理论输出值完全相等，从而验证了卷积加速器的功能的正确性。

图 8 卷积模块仿真波形

3.3　资源使用

本次实验设计经过综合之后，Vivado HLS 2020.1 给出了 FPGA 硬件资源使用情况。根据 2.6 节设计空间探索的结果，本文使用的输入通道并行度参数 $N=8$，输出通道并行度参数 $M=4$，卷积层在浮点 32 位运算下的资源使用情况见表 3。FPGA 工作在 100 MHz 下，由于输出缓存需要存储中间结果，分片后的部分权重和部分输入数据也存储在片上 RAM，所以 BRAM 和 URAM 使用资源较高，分别达到 53%和 46%。乘法运算逻辑全部使用 DSP 资源，使用率达到 83%；此外，LUT 资源使用率也达到了 93%，可以看出本文设计对资源的利用率已经很高。

表 3　FPGA 资源利用率

资源	BRAM	DSP48E	FF	LUT	URAM
已用	334	1449	227 975	215 541	45
可用	624	1728	460 800	230 400	96
占比	53	83	49	93	46

3.4　性能比较

由于不同文献使用的 FPGA 器件和网络结构不同，如果仅采用前向推理的时延和平台功耗，无法有效和公正地对比不同设计架构的优劣。此外，考虑到不同方法的资源利用效率存在差异，卷积运算过程主要是乘加运算。因此，本文提出 DSP 效率作为性能评估依据。DSP 是卷积计算中使用的主要资源，通常情况下加速器的吞吐量与硬件平台的 DSP 数量呈正相关趋势，DSP 效率表示单位 DSP 所能提供的 GFLOPS，可以避免不同硬件平台 DSP 数量差异带来的影响，从而更有效比较不同设计方案的性能优劣。本文同时增加能效比参数以便全面对比不同方法的加速效果。

各卷积层的时延和性能见表 4。时间延迟主要包括数据传输和卷积计算的延时。网络整体的平均性能为 35.45 GFLOPS。

表 4　各卷积层的性能对比

卷积层	复杂度/(GFLOPs)	时间/ms	性能/GFLOPS
1	0.043	8.02	5.36
2	0.231	5.26	43.92
3	0.462	10.21	45.25
4	0.231	5.47	42.23
5	0.462	10.77	42.90
6	0.231	7.10	32.54
总计	1.66	46.83	35.45

通过本次设计的板级测试，得到系统的性能和功耗等数据与其它文献的对比见表 5。文献［13］使用循环展开并行处理和多级流水线加速卷积神经网络，但没有设计片上缓存结构来提升数据访问效率，在 DSP 效率和能效比上远低于本设计。文献［11］利用设计空间探索寻找最优的计算并行性来最大化系统吞吐量，但没有充分设计片上缓存的数据复用方案，DSP 效率虽然高于本文，但是本文的能耗比更高，是其 1.5 倍，且本文使用 32 位浮点数表示，计算精度高于其 16 位定点数。综上所述，本文设计的加速方案在利用了缓存设计、并行设计、数据复用以及设计空间探索等多种优化设计的基础上，具有更高的综合性能。在硬件资源和功耗严格受限的嵌入式平台中，本设计相比表中其它方案，在单位能量上贡献更高的计算量。

表 5　与其它文献方法的对比

方法	平台	时钟频率/MHz	位宽	时间/ms	功耗/W	DSP/个	性能/GFLOPS	DSP 效率/(GFLOPS/个)	能效比/(GFLOPS/W)
文献[13]	XC7Z020	100	32 bits float	270	1.549	220	0.123	5.59e−4	0.079
文献[11]	Stratix-V GSD8	120	(8-16) bits fixed	—	19.1	1963	72.4	0.037	3.79
本文	ZCU104	100	32 bits float	46.83	6.354	1728	35.45	0.021	5.58

在 100 MHz 的工作频率下，数据处理单元处理一次 8×56×56 的输入特征图，通过 4×8×3×3 的卷积核生成 4×56×56 的输出特征图的卷积计算，计算量为 1 806 336 FLOPS，数据处理单元的运行时间为 34.37 μs，卷积计算的峰值性能为 52.56 GFLOPS。

表 6 为本文设计的卷积峰值计算性能与 CPU、GPU 以及文献［14］的对比，计算性能是 CPU 的 4.1 倍，功耗仅为 GPU 的 9.9%。相比文献［14］，本文利用设计空间探索充分挖掘卷积计算的并行性，本文设计的卷积计算性能是其 1.4 倍。

表 6　卷积计算性能的比较

平台	CPU i5-4210H	GPU GTX1080Ti	文献[14]	本文
工作频率	2.9 GHz	1.5 GHz	100 MHz	100 MHz
时间/ms	18	1.34	—	—
功耗/W	13.28	64	—	6.354
性能/GFLOPS	12.83	172.38	42.13	52.56

4　结束语

针对如何提升运行在嵌入式平台的深度卷积神经网络

的速度和能效，本文提出了一种卷积并行加速架构。该架构首先利用卷积层和 BN 层融合算法降低计算复杂度，通过层融合的分片设计，降低片上存储的资源消耗。为了优化内存，本文提出了一种双线性缓存设计，有效提高了数据复用效率，并利用 ping-pong RAM 的缓存方式和数据并行计算提高数据吞吐量。同时，利用设计空间探索寻找最优的计算并行度，从而最大化资源利用率。

实验结果表明，与 CPU、GPU 和其它 FPGA 平台的实现对比，本文提出的方法在性能、能耗以及资源利用率方面具有更高的综合性能，对于在资源和功耗严格受限的嵌入式环境中部署深度卷积网络，实现航天电子系统的智能化处理具有意义。

参考文献：

[1] Samip G，Sarala G，Santosh S. A study on deep learning architecture and their applications [C] //International Conference on Power Electronics，Control and Automation. New Delhi：IEEE，2019：430-435.

[2] Bertinetto L，Valmadre J，Henriques J F，et al. Fully-convolutional siamese networks for object tracking [C] //European Conference on Computer Vision. Amsterdam：Springer，2016：850-865.

[3] WANG Bin，LI Haiyan，WANG Yulin. Future prospects of artificial intelligence technology in the aerospace field [J]. Journal of Command and Control，2020，6 (4)：349-355 (in Chinese). [王彬，李海岩，王玉林. 未来空天领域中的人工智能技术展望 [J]. 指挥与控制学报，2020，6 (4)：349-355.]

[4] Garea A，Heras D，Argüello F. Caffe CNN-based classification of hyperspectral images on GPU [J]. Journal of Supercomputing，2019，75 (3)：1065-1077.

[5] Raghid M，Mazen E，Haitham A. FPGA-based accelerator for deep convolutional neural networks for the SPARK environment [C] //IEEE International Conference on Smart Cloud. New York：IEEE，2016：126-133.

[6] WU Yanxia，LIANG Kai，LIU Ying，et al. The progress and trend of deep learning FPGA accelerators [J]. Chinese Journal of Computers，2019，42 (11)：2461-2480 (in Chinese). [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.]

[7] Wang D，Xu K，Jiang D. PipeCNN：An OpenCL-based open-source FPGA accelerator for convolution neural networks [C] //16th International Conference on Field Programmable Technology. Melbourne：IEEE，2017：279-282.

[8] Chen Y H，Emer J，Sze V. Eyeriss：A spatial architecture for energy-efficient dataflow for convolutional neural networks [J]. ACM SIGARCH Computer Architecture News，2016，44 (3)：367-379.

[9] Liu Z，Dou Y，Jiang J，et al. Throughput-optimized FPGA accelerator for deep convolutional neural networks [J]. ACM Transactions on Reconfigurable Technology and Systems，2017，10 (3)：1-23.

[10] ZHANG Bang，LAI Jinmei. Design and implementation of a convolutional neural network accelerator based on FPGA [J]. Journal of Fudan University (Natural Science Edition)，2018，57 (2)：236-242 (in Chinese). [张榜，来金梅. 一种基于 FPGA 的卷积神经网络加速器的设计与实现 [J]. 复旦学报 (自然科学版)，2018，57 (2)：236-242.]

[11] Naveen S，Vikas C，Ganesh D，et al. Throughput-optimized opencl-based fpga accelerator for large-scale convolutional neural networks [C] //ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. New York：Association for Computer Machinery，2016：16-25.

[12] He K，Zhang X，Ren S，et al. Deep residual learning for image recognition [C] //Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas：IEEE，2016：770-778.

[13] LI Xiaoyan，ZHANG Xin，YAN Xiaobing，et al. Convolutional neural network acceleration system based on FPGA [J]. Journal of Hebei University (Natural Science Edition)，2019，39 (1)：99-105 (in Chinese). [李小燕，张欣，闫小兵，等. 基于 FPGA 的卷积神经网络加速系统 [J]. 河北大学学报 (自然科学版)，2019，39 (1)：99-105.]

[14] ZHAO Shuo，FAN Jun，HE Hu. CNN accelerated SoC system design based on FPGA [J]. Computer Engineering and Design，2020，41 (4)：939-944 (in Chinese). [赵烁，范军，何虎. 基于 FPGA 的 CNN 加速 SoC 系统设计 [J]. 计算机工程与设计，2020，41 (4)：939-944.]