DOI:10.20079/j.issn.1001-893x.230216003

基于 FPGA 的 YOLOv5s 网络高效卷积加速器设计*

刘　谦[1,2],王林林[1],周文勃[1,2]

(1. 中国科学院国家空间科学中心,北京 100190;2. 中国科学院大学 计算机科学与技术学院,北京 100049)

摘　要:为提升在资源受限情况下的嵌入式平台上卷积神经网络(Convolutional Neural Network, CNN)目标识别的资源利用率和能效,提出了一种适用于 YOLOv5s 目标识别网络的现场可编程门阵列(Field Programmable Gate Array, FPGA)共享计算单元的并行卷积加速结构,该结构通过共享 3×3 卷积和 1×1 卷积的计算单元提高了加速器硬件资源利用率。此外,还利用卷积层 BN(Batch Normalization)层融合、模型量化、循环分块以及双缓冲等策略,提高系统计算效率并减少硬件资源开销。实验结果表明,加速器在 200 MHz 的工作频率下,实现的卷积计算峰值性能可达 97.7 GOPS(Giga Operations per Second),其 YOLOv5s 网络的平均计算性可达 78.34 GOPS,与其他 FPGA 加速器方案相比在 DSP 效率、能耗比以及整体性能等方面具有一定的提升。

关键词:卷积神经网络(CNN);目标识别;YOLOv5s;并行卷积加速结构

开放科学(资源服务)标识码(OSID):

微信扫描二维码
听独家语音释文
与作者在线交流
享本刊专属服务

中图分类号:TP391　　**文献标志码**:A　　**文章编号**:1001-893X(2024)03-0366-10

Design of a YOLOv5s Network Efficient Convolution Accelerator Powered by FPGA

LIU Qian[1,2], WANG Linlin[1], ZHOU Wenbo[1,2]

(1. National Space Science Center, Chinese Academy of Sciences, Beijing 100190, China;
2. School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: A parallel convolution acceleration structure of field programmable gate array (FPGA) sharing computing unit suitable for YOLOv5s target recognition network is proposed to improve the hardware resource utilization of the accelerator by sharing computing units of 3×3 convolution and 1×1 convolution. Furthermore, strategies such as batch normalization (BN) layer fusion of convolution layer, model quantization, cyclic blocking, and double buffering are used to improve system computational efficiency and reduce hardware resource overhead. The experimental results show that at a working frequency of 200 MHz, the peak performance of convolution calculation can reach 97.7 GOPS (Giga Operations per Second), and the average calculation performance of the YOLOv5s network can reach 78.34 GOPS, which is better than other FPGA accelerator schemes in DSP efficiency, energy consumption ratio, and overall performance.

Key words: convolutional neural network (CNN); target detection; YOLOv5s; parallel convolution acceleration structure

* **收稿日期**:2023-02-16;**修回日期**:2023-05-11
基金项目:国家重点研发计划(2020YFE0202100)
通信作者:王林林　　Email:wanglinlin@nssc.ac.cn

0　引　言

近年来,卷积神经网络目标识别算法在人脸识别、自动驾驶以及物体识别等民用领域获得了巨大的成功。目前国内航天技术也逐渐向智能化转变,在航天设备上应用智能化的目标检测系统受到越来越多的关注。然而星载硬件计算资源十分匮乏,如何将基于卷积神经网络的目标检测算法部署到资源有限的宇航级芯片上是亟需解决的一项技术难题[1]。

现场可编程门阵列(Field Programmable Gate Array,FPGA)作为航天常用主控器件,具有功耗低、可重构以及并行计算能力强等优点,利用其高度的并行计算能力能够以较低的功耗实现高效的卷积计算,在卷积加速方面能够表现出独特的优势[2]。因此,本文提出采用 FPGA 作为硬件加速单元实现基于卷积神经网络的目标检测加速系统。

目前基于 FPGA 的卷积加速器研究主要集中在 3×3 标准卷积加速的优化上,对于目标检测常用的 YOLO 网络的网络加速优化研究还较少。由于目标检测网络较卷积计算有着更加复杂的结构,因此研究如何实现高效的网络加速更具有意义。在 YOLO 目标检测网络中卷积一般为两种,分别是 1×1 卷积和 3×3 卷积。对于 YOLO 目标检测网络的加速,文献[3-4]中提出的加速结构分别对 1×1 卷积和 3×3 卷积进行了硬件设计,两种卷积计算单元相互独立,计算资源互不共享。由于卷积神经网络目标检测算法每次只对一种卷积进行计算,因此此类结构中总有一部分计算单元会空闲,从而使硬件利用率较低。而文献[5]中提出卷积复用的方法,将 1×1 卷积权重放置在 3×3 卷积核权重第 5 位,剩余 8 位权重置零。该方法用 3×3 卷积直接代替 1×1 卷积,虽然大幅提高了硬件利用率,但是 1×1 卷积计算时每计算 1 次会产生 8 次无效计算和无效数据传输,因此造成了 1×1 卷积计算效率低下。文献[6]中提出了一种策略,将 3×3 卷积转换成 1×1 卷积进行计算,并对 YOLOv5s 网络实现了加速计算,但是该方法却牺牲了部分并行性,计算性能较差。文献[7]采用了基于 Winograd 算法的硬件加速结构完成了对 3×3、5×5 以及 11×11 尺寸的卷积加速,并取得了较好的加速效果,但该方法并不支持 1×1 卷积的计算。由于 YOLOv5 目标检测网络中具有大量的 1×1 卷积计算,因此该方法并不适合用于加速 YOLOv5 目标检测网络。

为解决以上文献中所提结构的缺点,本文提出了一种将 1×1 卷积和 3×3 卷积计算单元复用共享的结构。在大幅提高硬件利用率的同时,既可以计算 1×1 卷积也可以计算 3×3 卷积,同时 1×1 卷积计算时也不会产生无效计算和无效数据传输,解决了文献[3-7]中加速器结构硬件利用率低、计算效率低以及不支持 1×1 卷积计算等问题。此外,该硬件加速器还利用了层融合、多维度并行卷积加速、模型量化降低资源消耗等方法以较低的 FPGA 资源消耗获得了较高的计算能力,实现了卷积神经网络目标检测的高效计算。最后本文以通用目标检测算法 YOLOv5s 算法为基础,并以天体表面陨石坑的识别为主,验证了所提出的加速器结构的有效性。

1　YOLOv5s 目标检测算法

YOLOv5 目标检测算法由 Ultralytics LLC 公司于 2020 年 5 月提出,是目前优秀的目标检测算法之一。通过 YOLOv5 算法能够实现多目标高精度的目标检测,可以解决传统目标检测算法要求检测背景简单以及需要被检测目标特征明显类型单一等问题。YOLOv5 目标检测网络结构共有 5 个主流版本,分别是 YOLOv5n、YOLOv5s、YOLOv5m、YOLOv5l 和 YOLOv5x。

本文综合速度、精度等方面的因素采用了 YOLOv5s,并对 YOLOv5s 算法结构进行了微调:

一是将原算法模型的激活层由 SiLU 替换为 Leaky ReLU。由于 SiLU 激活函数中包括指数、乘法、除法和加法计算,其计算复杂度高,而 Leaky ReLU 激活函数中仅包含一个乘法计算计算复杂度低,因此采用 Leaky ReLU 激活函数能够在一定程度上降低计算量。

二是保留了 YOLOv5s-5.0 中的 FOCUS 结构和 YOLOv5s-6.0 中的 SPPF 结构,SPPF 结构中只采用了一种尺寸 5×5 的 Maxpool,相较于 SPP 结构中 3 种不同尺寸(5×5、9×9、13×13)的 Maxpool 在 FPGA 上仅需要设计一种尺寸 Maxpool,能够简化 FPGA 对 Maxpool 的设计。

以上算法修改均能够有效减少 FPGA 资源使用,且对算法本身影响不大。调整后的 YOLOv5s 网络结构如图 1 所示。

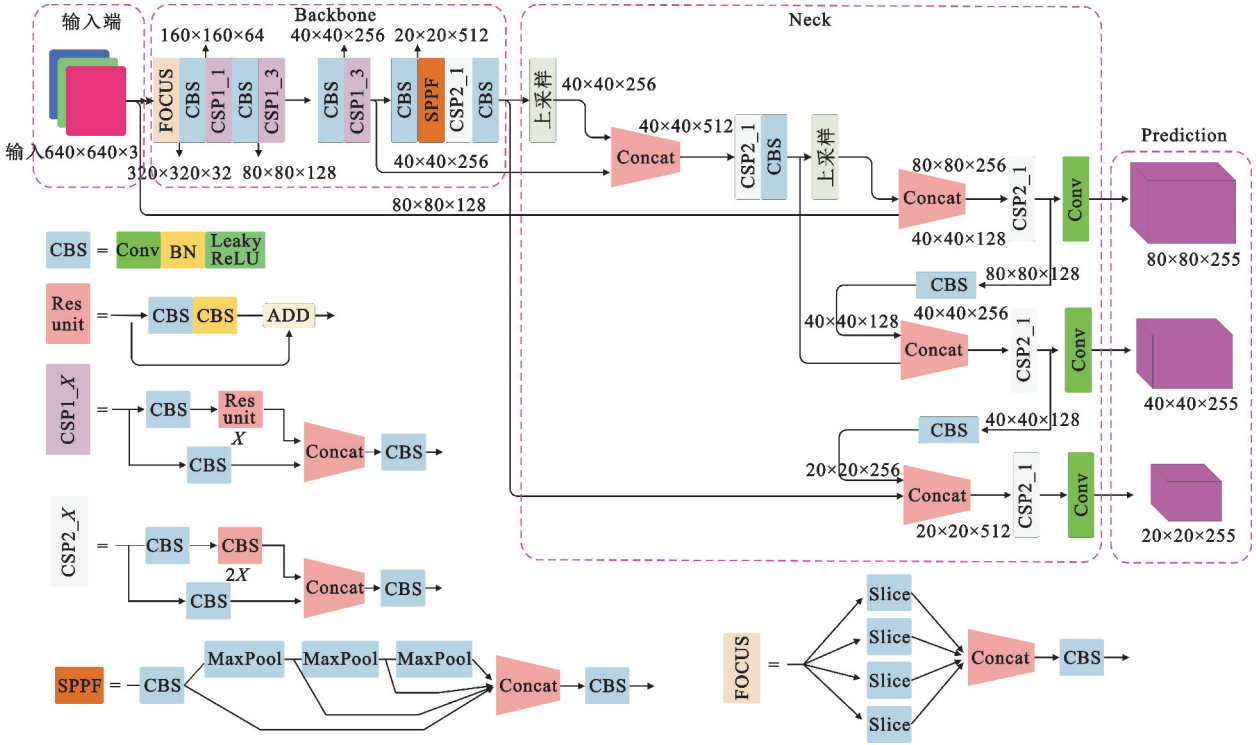

图 1　调整后 YOLOv5s 网络结构

Fig. 1 Network architecture of adjusted YOLO V5s

2　网络模型层融合与量化

2.1　网络模型层融合

YOLOv5s 算法中 CBS 计算过程包括卷积、BN(Batch Normalization)和 Leaky ReLU 三部分。在模型训练过程中 BN 层能够解决梯度消失和梯度爆炸的问题,并可以加快网络的收敛提高训练效率,同时也可以提高网络稳定性。而在前向推理过程中 BN 层却降低了模型的性能,所以在推理过程中可以通过层融合的方式将 BN 层融合到卷积层中,从而达到降低计算量、提高模型性能的效果[8]。卷积层 BN 层融合过程如下:

在 BN 层中 μ 表示输入的均值如式(1)所示,式中 x 表示输入特征图中的元素,m 表示输入特征图中元素的个数。δ^2 为输入的方差,如式(2)所示。

$$\mu=\frac{1}{m}\sum_{i=1}^{m}x_i \tag{1}$$

$$\delta^2=\frac{1}{m}\sum_{i=1}^{m}(x_i-\mu)^2 \tag{2}$$

BN 计算公式如式(3)所示,式中 γ 为尺度因子,β 为偏移因子。尺度因子优化了特征数据分布的宽窄,偏移因子则优化了数据的偏移量,在模型训练过程中这两个参数自动学习,训练结束后数值固定。ε 为一个大于 0 且极小的数值,用于防止方差 δ^2 为 0 的异常情况出现。式(4)为卷积计算公式,其中 W 表示卷积核,X 表示输入特征图,B 表示偏置,Y 表示输出特征图。将式(4)代入到 BN 计算中得到卷积和 BN 计算的总公式(5),式中 Y' 表示卷积和 BN 计算后的输出特征图。

$$y_i=\gamma\frac{x_i-\mu}{\sqrt{\delta^2+\varepsilon}}+\beta \tag{3}$$

$$Y=WX+B \tag{4}$$

$$Y'=\gamma\frac{(WX+B)-\mu}{\sqrt{\delta^2+\varepsilon}}+\beta \tag{5}$$

将式(5)展开可得出新的卷积核 W' 如式(6)所示,新的偏置 B' 如式(7)所示。

$$W'=\frac{\gamma W}{\sqrt{\delta^2+\varepsilon}} \tag{6}$$

$$B'=\frac{\gamma(B-\mu)}{\sqrt{\delta^2+\varepsilon}}+\beta \tag{7}$$

层融合的方法通过提前计算获得新的卷积核 W' 和偏置 B',从而将 BN 层融合到卷积层中,有效降低了模型推理阶段 FPGA 上的计算量,减少 FPGA

片上 DSP48E、LUT 等资源的消耗。同时由于计算步骤减少,所以也能够在一定程度上减小推理延时,提高系统整体性能。

2.2　网络模型量化

在深度学习领域量化是一种常用的能够有效降低存储空间和计算成本的方法[9],可以有效压缩模型大小。在 FPGA 上采用量化方法用低位宽的定点乘法器、加法器代替高位宽的浮点乘法器、加法器可以大幅降低 FPGA 逻辑资源的使用,提高资源紧缺情况下 FPGA 上的计算能力。本文对模型进行了量化,将 32 位单精度浮点转换为 16 位定点数,在 FPGA 中设计 16 位定点乘法器、加法器来完成卷积层的计算。

3　FPGA 硬件加速器设计

3.1　加速器体系架构

根据图 1 中 YOLOv5s 网络结构框图可以看出其内部由基础模块构成,包括卷积 Conv 模块、ADD 模块、Maxpool 模块、Silce 模块、上采样模块和 Concat 模块。其中卷积分为两类,分别为 1×1 卷积和 3×3 标准卷积,在 YOLOv5s 算法中一共了包含 43 层 1×1 卷积和 19 层 3×3 标准卷积,该部分是 YOLOv5s 算法中计算最复杂的部分,需要在 FPGA 设计并行计算单元以达到计算加速效果。本文对 YOLOv5s 算法中计算量较大的模块在 FPGA 上进行了硬件加速设计,包括 Conv1×1 卷积、Conv3×3 卷积、ADD、Maxpool 和上采样模块。

加速器整体架构如图 2 所示,该架构中包括 CPU 和 FPGA 两个部分。其中 CPU 部分为双核 ARM Cortex-A9 MPCore,该部分外挂 DDR3 内存,用于缓存特征图和权重数据以及运算过程中产生的中间数据,外挂 NAND FLASH 用于存储输入的图像文件和权重文件以及存储计算后的输出文件。UART 接口用于监控加速器各个阶段的计算耗时,输出所识别到的陨石坑在图像中的坐标信息。CPU 部分主要进行图像和权重等文件的读取、将图像和权重文件转存到 DDR3、配置 FPGA 端加速器以及处理 YOLOv5s 中简单的计算如 Silce 和 Concat。

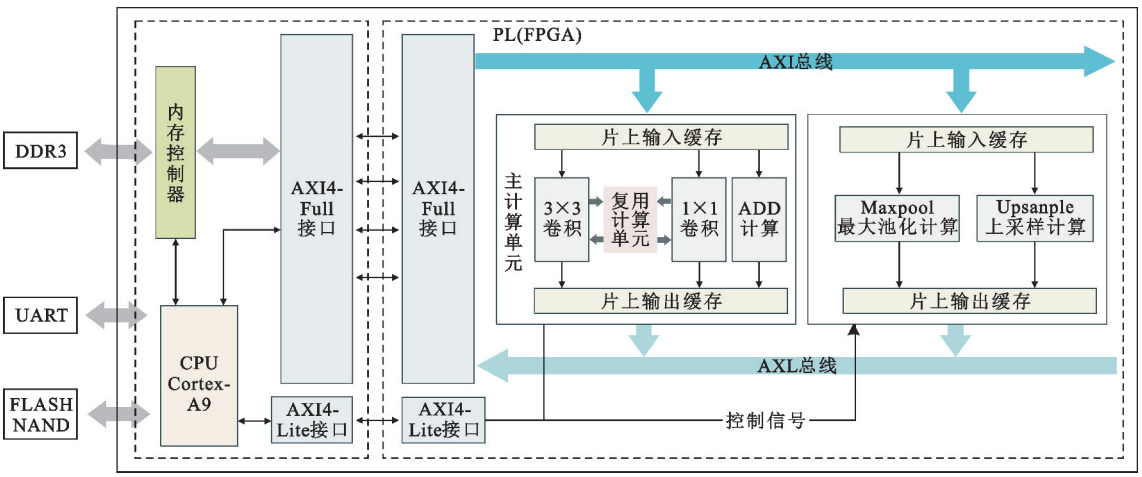

图 2　系统整体架构

Fig. 2 Overall system architecture

FPGA 端设计有主计算单元和辅计算单元两个部分。其中主计算单元包括 3×3 卷积缓存模块、1×1 卷积缓存模块、通用并行计算模块和 ADD 计算模块。在主计算单元中由于涉及的数据量较大且数据传输延时较长,所以在主计算单元中采用双缓存乒乓机制用于掩盖数据传输所消耗的时间。辅计算单元包括 Maxpool 计算模块和上采样计算模块。

CPU 和 FPGA 之间通过 AXI(Advanced eXtensible Interface)总线交互数据,采用了 1 个 AXI-GP 接口和 4 个 AXI-HP 接口。其中 AXI-GP 接口用于 FPGA 和 CPU 之间的控制信号传输,实现加速器的偏移地址配置以及其他参数配置。而 AXI-HP 接口负责计算过程中 DDR3 内存与 FPGA 的高速数据交换,AXI-HP 接口和 DDR 之间具有专有控

制器可以直接实现 FPGA 与内存的高速数据交换，由于不需要经过 CPU 从而大大提高了数据传输效率，适合大数据吞吐量的应用场景。

3.2 存储设计

3.2.1 循环分块策略

由于 FPGA 的片上存储 BRAM（Block Random Access Memory）资源十分有限，而卷积计算过程中的权重、偏置以及特征图会占用大量的存储资源，所以无法直接将整个特征图存储到 FPGA 的片上存储中。因此本文采用对特征图进行循环分块的设计策略，将片外 DDR3 中存储的完整特征图分为数个小块，分批次存储到 FPGA 片上存储 BRAM 中进行后续的卷积计算，解决了 FPGA 片上存储资源不足的问题。循环分块过程如图 3 所示，分块后的小特征图存储在双缓存结构中。

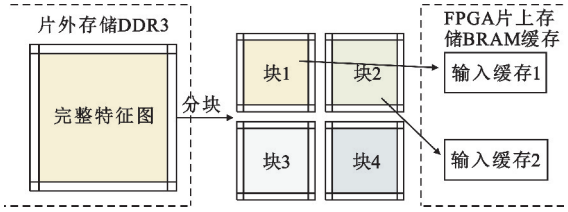

图 3 循环分块过程

Fig. 3 Cyclic chunking process

3.2.2 双缓存与缓存分割

为进一步提高整个系统的吞吐量，掩盖 AXI 总线和内存之间的传输延时，本文设计了一种双缓存机制，使加速器可以在 AXI 总线传输特征图的过程中同时进行卷积计算。双缓存结构如图 4 所示，当计算单元读取缓存 1 中的数据进行卷积计算的过程中，缓存 2 可以同时用于接收 AXI 总线中的数据，而缓存 2 用于计算时，缓存 1 则可以用于接收数据，以此往复。该结构可以使计算单元一直处于工作状态，从而在一定程度上提高整个系统的工作效率。

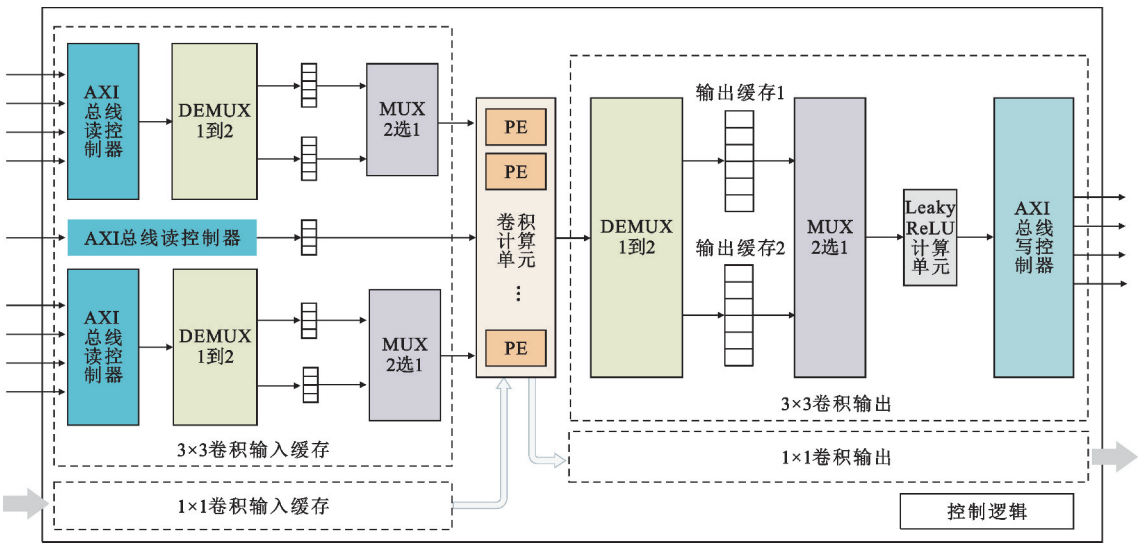

图 4 双缓存结构示意

Fig. 4 Dual cache architecture schematic

此外，由于 FPGA 中的 BRAM 仅能配置为双端口进行读写，读写速率受端口数量的限制。为解决此类问题，本文采用数组分割的方式，将分割后的数组存储在不同的 BRAM 中，用数个小容量缓存代替一个大容量缓存，从而可以实现多个端口同时读写数据提高数据吞吐量，为后续并行计算结构提供充足的数据量。

3.3 卷积计算结构设计

本文中 1×1 卷积和 3×3 卷积采用了复用的计算单元，需要计算单元尽可能兼容两种卷积计算，因此本文设计了输入-输出通道并行的加速结构，该结构中计算单元的硬件结构与卷积核尺寸无关。其计算单元中包含 64 个数据处理单元（Processing Element，PE），PE 由 8 个输入并行数据选择器、4 个乘法器、1 个加法树以及 1 个累加结构构成。每个 PE 单元具有 8 个输入通道其中 1×1 卷积和 3×3 卷积各 4 个通道，通过数据选择器选择其中 4 个通道至后续的运算结构，因此实现了 1×1 卷积和 3×3 卷积的计算资源共享。PE 计算单元结构如图 5 所示。

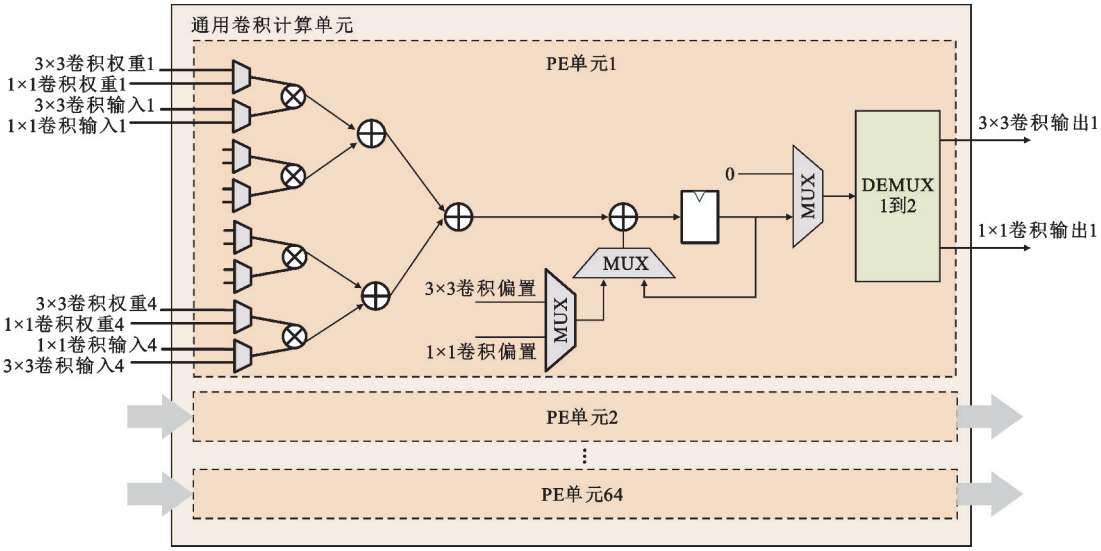

图 5　PE 结构示意

Fig. 5 PE architecture schematic

在 YOLOv5s 算法中的通道数最大达到了 1 024,受限于 FPGA 资源无法将 1 024 个通道全部在 FPGA 上实现并行,所以实际输入的特征图通道数远大于 PE 单元的设计输入通道,因此,输入的特征图只能分批进入 PE 单元中进行计算。此外,对于 3×3 卷积每一个通道的每一个输出数据也都会进行 9 次卷积核内计算,且输出通道的计算结果是分批输入特征图卷积累加的结果,因此在 PE 单元中增加了累加结构用于将不同批次进入 PE 单元的输出数据以及 3×3 卷积核内计算数据进行累加,实现完整的多通道卷积计算。为提高 PE 结构的运行频率和吞吐量,本文在 PE 单元的硬件结构中插入了中间寄存器进行了 6 级流水设计,使加速器可以在高时钟频率下稳定工作。

3.4　Leaky ReLU 激活函数计算单元设计

在 YOLOv5s 中最后一层卷积不需要激活函数,所以由累加器输出的特征图还需要通过数据选择器,选择特征图是否经过激活函数。激活函数根据调整后的算法结构采用 Leaky ReLU,该激活函数较 SiLU 激活函数具有电路结构简单的优点,适合在计算资源紧张的嵌入式环境应用,其计算公式如式(8),本文中 a 采用 0.01。

$$\mathrm{LeakyReLU}(x)=\begin{cases}x, x>0\\ ax, x\leqslant 0\end{cases} \tag{8}$$

由于 Leaky ReLU 激活函数计算简单,计算延时低仅需要 1 个时钟周期,故本文中一共设计了 4 个并行 Leaky ReLU 计算单元对应在 4 个 AXI 输出接口之前,其结构电路如图 6 所示。

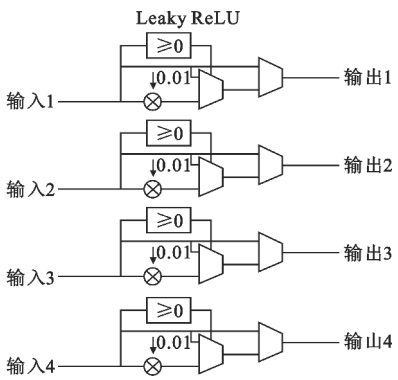

图 6　Leaky ReLU 结构示意

Fig. 6 Leaky ReLU architecture schematic

3.5　ADD 模块设计

ADD 本质是将两个输入层相加,该部分要求输入层的通道数、特征图尺寸完全相等。由于 ADD 单元的计算结构与卷积计算单元有着较大的差异,难以实现与卷积计算共享硬件资源的计算结构,因此本文对 ADD 结构进行了单独的硬件加速设计。ADD 硬件加速模块的结构如图 7 所示。

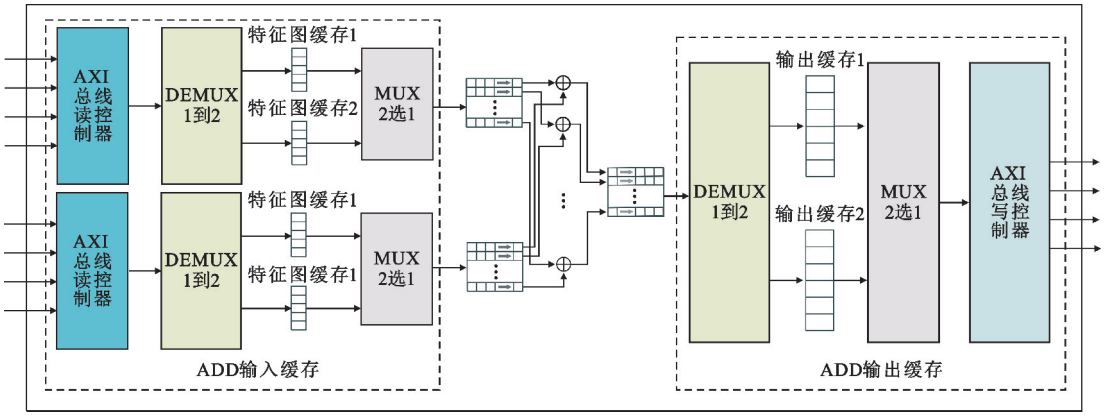

图 7　ADD 结构示意

Fig. 7 ADD architecture schematic

该部分同样采用双缓存结构用于提高吞吐量。为降低寻址难度,该部分采用一维形式分块,将 $n\times n$ 特征图所有的数据按照 Tp 个特征点分批为 $n\times n/Tp$ 次,同时按照 Tn 个通道将输入通道 N 按 N/Tn 次分批处理,将 Tn 个通道的 Tp 个特征点数据一次性由 DDR3 传输到 FPGA 端并缓存到片上 BRAM 中,充分利用 AXI-HP 总线的突发传输模式有效降低传输延时,其中 Tn 为 32,Tp 为 400。其加法计算部分采用三级流水,分别为读数据阶段、加法计算阶段和写数据阶段,用于提高系统工作频率。

3.6　Maxpool 模块设计

Maxpool 在算法中主要起到减小特征图尺寸,控制过拟合的作用[10]。对于最大池化模块 Maxpool 的处理,首先按照 Maxpool 的窗口尺寸将特征图分为 5×5 的小块,然后将 5×5 分块的特征图存储到输入缓存中,之后通过比较器树比较 5×5 特征图中 25 个特征点的大小,取最大值并将该值作为输出特征图中的一个特征点存入到输出缓存中。其处理过程和计算单元结构图如图 8 所示。

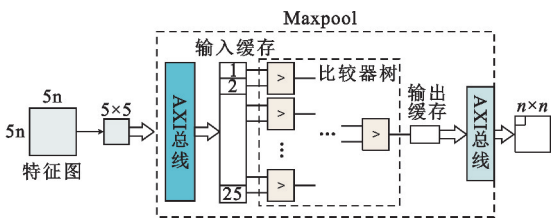

图 8　5×5 Maxpool 结构示意

Fig. 8 5×5 Maxpool architecture schematic

3.7　上采样模块设计

YOLOv5s 中所使用的上采样模块作用是将输入特征图扩大 4 倍,即 1 个输入特征点对应 4 个输出特征点。该部分在 FPGA 上实现的结构较为简单,仅仅涉及缓存的读写,具体操作是将输入缓存中的 1 个特征点取出并转存到输出缓存中对应的 4 个特征点上。

4　实验与分析

4.1　实验环境和平台

本实验在 YOLOv5s 模型训练阶段采用了自制数据集,使用了 124 张来自月球和火星的陨石坑图片,图片尺寸为 640×640,用于验证小样本数据集下目标检测算法的有效性。采用的训练平台硬件环境为 Intel i7-1100 CPU,RTX3080 GPU,16 GB 内存。

在加速效果测试中,为比较不同平台间的差异采用了两种平台,分别为实验平台 1,Xilinx 公司 ZC706 开发板,该开发板采用的主芯片型号为 XC7Z045,XC7Z045 芯片集成双核 ARM Cortex-A9 CPU 和 Kintex-7 FPGA,FPGA 中具备 900 个 DSP48E 单元、218 600 个 LUT 逻辑资源以及 545 个 BRAM36K 存储资源;实验平台 2,微相公司 ZYNQ 开发板,该开发板主芯片为 XC7Z020,XC7Z020 芯片集成双核 ARM Cortex-A9 CPU 和 Artix-7 FPGA,FPGA 中具备 220 个 DSP48E 单元、53 200 个 LUT 逻辑资源以及 140 个 BRAM36K 存储资源。

4.2　实验结果与分析

XC7Z045 芯片中 FPGA 片上资源使用情况如表 1 所示,其中 DSP48E1 一共使用了 65%,主要用于组成卷积计算结构中的乘法器和加法器以及 ADD

模块中的加法器;BRAM36K 使用了 58.07%,主要用于 FPGA 片上对特征图、权重等的缓存,以及构成双缓冲结构等;而 LUT、LUTRAM 以及 FF 等资源主要用于加速器逻辑控制、加速器内部流水线构成等。

表 1　XC7Z045 芯片资源占用率

Tab. 1 XC7Z045 chip resource usage

资源	可用	已用	占比/%
LUT	218 600	83 093	38.01
LUTRAM	70 400	10 204	14.49
FF	437 200	95 792	21.91
BRAM36K	545	316.5	58.07
DSP48E	900	585	65.00

XC7Z020 芯片中 FPGA 片上资源使用情况如表 2 所示,各部分资源的作用与 XC7Z045 相似。

表 2　XC7Z020 芯片资源占用率

Tab. 2 XC7Z020 chip resource usage

资源	可用	已用	占比/%
LUT	53 200	47 307	88.92
LUTRAM	17 400	11 503	66.11
FF	106 400	46 786	43.97
BRAM36K	140	116.5	83.21
DSP48E	220	196	89.09

实验表明,该目标检测系统能够正确读取 FLASH 中的图像和权重文件并进行 YOLOv5s 算法的计算,能够准确识别出图像中的天体陨石坑位置。部分天体陨石坑的识别结果如图 9 所示,矩形框所框出的位置为陨石坑所在位置。表 3 对原始 YOLOv5s 网络和本文中的 YOLOv5s 网络的检测效果做了对比,可见本文中的 YOLOv5s 网络保持了与原始网络效果相近,虽然指标有一定的损失,但是本文网络更加适合在 FPGA 上实现,与文献[6]相比在相同硬件平台下,采用本文中的加速器结构和 YOLOv5s 网络结构获得了数倍的计算性能提升,其利远大于弊。

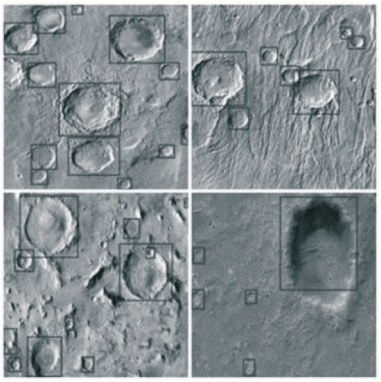

图 9　陨石坑识别结果

Fig. 9 Crater identification results

表 3　与原始网络的识别效果对比

Tab. 3 Recognition results compared with the original network

网络	Precision	Recall	mAP@0.5:0.95	mAP@0.5
原始 YOLOv5s 网络	79.4	79.1	83.5	41.7
本文 YOLOv5s 网络	79.5	76.9	81.7	40.3

通过实验将本文中所提出的 FPGA 加速结构与 RTX2060、树莓派两种平台在功耗、速度以及芯片架构、能效等方面做了比较,结果见表 4。在 FPGA 平台下的能效(单位:J/picture)是 CPU 平台的 11~41 倍,是 GPU 平台的 2.3 倍。通过分析可以得出,FPGA 平台上实现的 YOLOv5s 算法在能效方面较 CPU 和 GPU 平台具有显著优势,能够实现低功耗情景下的高性能计算。

表 4　不同平台间计算速度对比

Tab. 4 Comparison of computing speed between platforms

平台	功耗/W	处理延时/ms	架构	主频/GHz	芯片工艺	能效/(J/picture)
RTX2060	115	32	NVIDIA Turing(GPU)	1.47	12	3.68
树莓派 3B+BCM2837B0	5.4	12 100	ARM Cortex-A53(CPU)	1.4	40	65.34
树莓 4B BCM2711	5.5	3 300	ARM Cortex-A72(CPU)	1.5	28	18.15
ZC706(本文)	7.346	217	Kintex-7(FPGA)	0.2	28	1.59

在加速器方面,将本文中提出的加速结构与其他文献中加速结构在计算性能、能效以及 DSP 利用率等方面进行了比较,结算如表 5 所示。表中 GOPS 表示加速器每秒乘法和加法的总计算次数,

GOPS · W^{-1} 表示能效比,GOPS · DSP^{-1} 表示 DSP 效率。通过对比可以看出,本文所提出的加速器结构在计算能力、能效比、DSP 效率等方面均有着较好的表现,平台 2(XC7Z020 平台)与文献[6]在相同算法、相同硬件平台情况下其计算性能提高了 4.46 倍,能效比提高了 3.69 倍,DSP 利用效率提高了 4.49 倍。此外,平台 1(XC7Z045 平台)与平台 2(XC7Z020 平台)相比,平台 1 实现了更强的计算能力,原因是平台 1 硬件资源较多可以实现更高的并行度,但在能效比和 DSP 效率上两者基本保持一致,受不同平台的影响较小,因此本文中的加速结构可以在不同硬件平台上进行移植,同样可以实现较高的能效和 DSP 利用率。通过以上对比可知,本文所提出的 YOLOv5s 网络 FPGA 加速结构具有资源使用量少、效率高等特点,在相同器件上能够提供更强的计算能力。

表 5 加速性能对比

Tab. 5 Accelerated performance comparison

平台	软件算法	硬件平台	量化方式	工作频率/MHz	DSP 使用量	功耗/W	计算能力/GOPS	能效/(GOPS · W^{-1})	DSP 效率/(GOPS · DSP^{-1})
文献[3]	YOLOv4-tiny	XC7Z020	16 b fixed	180	216/220	2.860	9.24	3.23	0.042 8
文献[6]	YOLOv5s	ZC702 XC7Z020	Float 32	100	197/220	2.508	5.83	2.32	0.029 6
文献[11]	YOLOv1	ZC706 XC7Z045	32 b fixed	200	800/900	1.167	18.82	16.13	0.023 5
文献[12]	YOLOv2-tiny	ZC706 XC7Z045	16~32 b fixed	100	784/900	7.500	41.99	5.60	0.053 6
文献[13]	YOLOv5s	KCU116			1 200/1 824	12.100	44.90	3.71	0.037 4
平台 1(本文)	YOLOv5s	ZC706 XC7Z045	16 b fixed	200	585/900	7.346	78.34 (峰值 97.70)	10.66	0.133 9
平台 2(本文)	YOLOv5s	XC7Z020	16 b fixed	180	196/220	3.039	26.02 (峰值 30.10)	8.56	0.132 8

5 结束语

本文研究了基于 YOLOv5s 目标检测网络的加速设计,并提出了一种适用于 YOLOv5s 网络结构的 FPGA 硬件加速结构,实现了对 YOLOv5s 算法各个阶段的加速工作,完成了对天体陨石坑的目标识别检测。此外,还提出了一种 1×1 卷积和 3×3 卷积硬件计算资源共享的结构,实验表明该结构较其他结构在能效、资源利用效率等方面具有一定的优势,在资源受限的领域具有重要的意义,为日后的航天器智能化提供了新思路。

本文中的加速结构已经在不同的 FPGA 平台上获得了较理想的加速效果,因此下一步研究工作可以聚焦在采用宇航级 FPGA 芯片的验证工作上,实现可以实际应用于航天领域的目标检测系统。

参考文献:

[1] 张鹏程,武文波,李强,等. 面向星载边缘计算的遥感目标检测算法轻量化优化研究[J]. 空间控制技术与应用,2022,48(5):86-94.

[2] 龚豪杰,周海,冯水春. 基于 FPGA 的卷积神经网络并行加速设计[J]. 计算机工程与设计,2022,43(7):1872-1878.

[3] XU S Y,ZHOU Y J,HUANG Y R,et al. YOLOv4-tiny-based coal gangue image recognition and FPGA implementation[J]. Micromachines,2022,13(11):1-19.

[4] ZHANG F X,LI Y Y,YE Z H. Apply Yolov4-tiny on an FPGA-based accelerator of convolutional neural network for object detection [C]//Proceedings of 2022 2nd International Conference on Artificial Intelligence and Industrial Technology Applications. Dali: IOP Publishing,2022:1-8.

[5] 刘华,陶冠男,杨文清. 卷积神经网络并行化设计及 FPGA 实现[J]. 制造业自动化,2022,11(44):147-154.

[6] YU H,LI S Z. A higher performance accelerator for resource-limited FPGA to deploy deeper object detection networks [C]//Proceedings of 2022 IEEE 16th International Conference on Anti-counterfeiting, Security,and Identification. Xiamen:IEEE,2022:1-5.

[7] LAI J,YANG L X,LI D J,et al. Design and optimization of

winograd convolution on array accelerator[J]. Journal of Beijing Institute of Technology,2023,32(1):69-81.

[8] WAI Y J,YUSSOF Z M,SALIM S I,et al. Fixed point implementation of Tiny-Yolo-v2 using OpenCL on FPGA [J]. International Journal of Advanced Computer Science and Applications,2018,9(8):506-512.

[9] 满涛,郭子豪,曲志坚. 卷积神经网络的 FPGA 并行加速设计与实现[J]. 电讯技术,2021,61(11):1438-1445.

[10] NGUYEN D T,NGUYEN T N,KIM H,et al. High-throughput and power-efficient FPGA implementation of YOLO CNN for object detection[J]. IEEE Transactions on Very Large Scale Integration Systems,2019,27(8):1861-1873.

[11] ZHAO R,NIU X,WU Y,et al. Optimizing CNN-based object detection algorithms on embedded FPGA platforms[C]//Proceedings of 2017 13th International Symposium on Applied Reconfigurable Computing. Cham:Springer,2017:255-267.

[12] HUANG H M,LIU Z H,CHEN T S,et al. Design space exploration for YOLO neural network accelerator[J]. Electronics,2020,9(11):1-15.

[13] LIU M S,LUO S Y,KEVIN H,et al. An efficient real-time object detection framework on resource-constricted hardware devices via software and hardware co-design [C]//Proceedings of 2021 IEEE 32nd International Conference on Application-specific Systems,Architectures and Processors. New Jersey:IEEE,2021:77-84.

作者简介:

刘　谦　男,1996 年生于山东日照,硕士研究生,主要研究方向为星载目标检测和卷积硬件加速。

王林林　男,1986 年生于河南商丘,2013 年获博士学位,现为副研究员,主要研究方向为星载设备架构与星载智能处理。

周文勃　男,1999 年生于湖北荆州,博士研究生,主要研究方向为智能光电成像系统仿真与应用。