

文章编号: 1671-7449(2026)02-0267-08

J. Test Measur. Technol., 2026, 40(2): 267-274.

面向YOLOv4-Tiny的FPGA分块硬件加速与实现

南洋, 逯暄*, 张婷

(山西大学 物理工程学院, 山西 太原 030006)

摘要: 针对YOLOv4-Tiny算法在FPGA上运行耗时长及功耗大的问题, 提出了一种分块优化的硬件加速方法。该方法将特征图等分为多个小块, 在每小块的推理过程中利用乒乓缓存机制实现数据预取与计算的流水化调度, 同时通过流水线设计提升计算吞吐量, 最终, 在ZYNQ7020 FPGA上实现了YOLOv4-Tiny卷积神经网络的加速。最后, 进行了硬件加速前后的对比实验和道路场景测试实验, 结果显示, 该方法在ZYNQ7020 FPGA上的单帧图像推理时间为0.411 s, 平均功耗为3.021 W, 综合性能优于现有的FPGA加速平台, 验证了该方法可以有效提高系统的数据传输效率和带宽利用率。

关键词: YOLOv4-Tiny; 硬件加速; 目标检测; 现场可编程门阵列

中图分类号: TP391

文献标识码: A

DOI: 10.62756/csjs.1671-7449.2026013

引用格式: 南洋, 逯暄, 张婷. 面向YOLOv4-Tiny的FPGA分块硬件加速与实现[J]. 测试技术学报, 2026, 40(2): 267-274.

NAN Yang, LU Xuan, ZHANG Ting. FPGA-based hardware accelerator implementation for YOLOv4-tiny with input feature map block partitioning[J]. Journal of Test and Measurement Technology, 2026, 40(2): 267-274.

FPGA-Based Hardware Accelerator Implementation for YOLOv4-Tiny with Input Feature Map Block Partitioning

NAN Yang, LU Xuan*, ZHANG Ting

(College of Physics and Electronic Engineering, Shanxi University, Taiyuan 030006, China)

Abstract: To address the issues of long execution time and high power consumption when running the YOLOv4-Tiny algorithm on an FPGA, a block-wise optimization hardware acceleration method was proposed. In this method, the feature map was divided equally into multiple small blocks. During the inference process of each block, a ping-pong buffering mechanism was utilized to achieve pipelined scheduling of data prefetching and computation. Concurrently, throughput was enhanced through pipeline design. The acceleration of the YOLOv4-Tiny convolutional neural network was ultimately implemented on a ZYNQ7020 FPGA. Finally, comparative experiments before and after hardware acceleration, along with road scene testing experiments, were conducted. The experimental results showed that the inference time for a single frame on the ZYNQ7020 FPGA was 0.411 s, with an average power consumption of 3.021 W. The overall performance was superior to existing FPGA acceleration platforms. This result demonstrates that the proposed method effectively improves the system's data transmission efficiency and bandwidth utilization.

Key words: YOLOv4-Tiny; hardware acceleration; object detection; field-programmable gate array(FPGA)

【收稿日期】2025-03-05

【基金项目】山西省基础研究计划资助项目(202203021222016)

【作者简介】南洋(1998—), 男, 硕士生, 主要从事FPGA智能硬件研究。E-mail: m13233229761@163.com。

*【通信作者】逯暄(1987—), 女, 副教授, 博士, 主要从事信号处理、人工智能算法研究。E-mail: xuanlu@sxu.edu.cn。

0 引言

近年来,基于卷积神经网络的模型算法在目标检测、图像分类、语音识别等领域都有着重要应用^[1-3]。其中,YOLO算法在精度和速度之间实现了良好的平衡,成为目标检测领域常用的算法之一。然而,随着YOLO网络的不断更新迭代,其模型规模和参数数量也在增加。在使用图形处理器进行部署时^[4-5],需要较高的功耗才能实现最大限度的推理速度和精度,限制了其在资源受限的边缘设备上的使用。相比之下,专用集成电路虽然性能优越,但存在开发周期长、灵活性低等问题。现场可编程门阵列(Field-Programmable Gate Array, FPGA)能够灵活地进行设计和修改,可以更好地适应网络模型的升级、调整和部署,顺应了深度学习快速演进的趋势,凭借其强大的可重构性,更适合用于卷积神经网络的加速与部署^[6-8]。

尽管FPGA在深度学习硬件加速领域具有较低功耗和高性能的优势,但实现高效部署仍面临诸多挑战,例如模型大小与精度的权衡、推理速度的优化以及计算单元在FPGA上的有效映射。学术界已有不少研究针对这些问题展开探索^[9-10]。Zhang等^[11]提出了一种针对通信密集型应用的方案,通过增加突发传输长度来更有效地利用带宽;Yu等^[12]采用Winograd快速算法,通过FPGA加速卷积计算,虽然能降低片上资源消耗,但该算法只适用于小卷积核网络,对于大卷积核网络的加速效果有限;梅志伟等^[13]提出了一种结合乘加树和脉动阵列的乘加阵列方案来加速卷积层计算,这种方法虽能节省硬件资源并提升计算性能,但未能有效利用数字信号处理(Digital Signal Processor, DSP)单元资源;Liu等^[14]将Winograd算法引入FPGA,并提出了将其与脉动阵列结合的架构,然而该架构依赖复杂的内存管理系统,且Winograd算法的局限性限制了其应用范围。因此,未来的研究应更加注重提升FPGA加速器的通用性和灵活性。Zhao等^[15]提出了一种基于输入输出通道并行的组合策略,结合多通道数据传输,通过提升并行度提高系统带宽利用率,以此实现传统卷积计算的加速,然而该设计在数据处理的优化方面存在不足,直接制约了推理时间的进一步缩短。然而该设计在数据搬运优化方面仍有不足,尤其是在从片外双倍速率同步动态随机存储

器(Double Data Rate Synchronous Dynamic Random Access Memory, DDR)读取数据时,仍然存在较大的传输延迟问题。李雷等^[16]通过简化模型并采用双缓冲策略,对卷积模块和池化模型进行了高效并行的设计,但模块间的数据传输可能会引入额外的延迟,进而影响整体性能。姜明飞等^[17]提出了基于双缓冲脉动阵列的硬件加速器方案,通过构建脉动阵列结构有效提升了传统卷积运算的效率。然而,其实验所选数据集的规模和复杂度相对较低,该方法在处理更复杂的模型或更大规模的数据集时可能面临资源瓶颈问题。

据此,本文提出了一种面向YOLOv4-Tiny的FPGA分块硬件加速优化方法,旨在提升FPGA在卷积运算中的资源利用率和单帧推理效率。该方法将输入特征图分成小块,使得每次计算只需加载小块数据到缓存,从而降低了频繁访问外部存储所带来的延迟。在每小块的推理过程中,通过结合多重乒乓操作和高度并行的流水线设计,进一步提升了硬件资源的利用率和计算速度。本文所提方法优化了YOLOv4-Tiny网络在边缘硬件FPGA平台上的计算速度、性能和资源利用率等关键性能指标。

1 相关理论

1.1 YOLOv4-Tiny网络模型

YOLOv4-Tiny是YOLO系列模型^[18]的一个轻量化版本,相比于标准YOLOv4,YOLOv4-Tiny在保持较高性能的同时,通过简化网络结构和减少参数量,降低了计算开销和内存需求。这使得YOLOv4-Tiny在边缘计算设备、移动设备和嵌入式系统等对计算资源要求较低的环境中具有较大优势。YOLOv4-Tiny模型主要由主干网络和头部网络两部分组成。

主干网络由主干特征提取网络和特征金字塔网络两部分组成,如图1所示。其中,主干特征提取网络采用简化的CSPDarknet53结构,包含3个标准卷积激活层与3个残差单元,并在构建特征金字塔网络时合并了有效特征层。标准卷积激活层(DarknetConv2D_BN_Leaky)由卷积层(DarknetConv2D)、批量归一化(Batch Normalization)和LeakyReLU激活函数三部分组成,该模块首先通过卷积运算提取空间特征,接着使用批量归一化对特征分布进行标准化处理以加速训练并

提升模型稳定性,最后,通过 LeakyReLU 激活函

数引入非线性来增强模型表达能力。

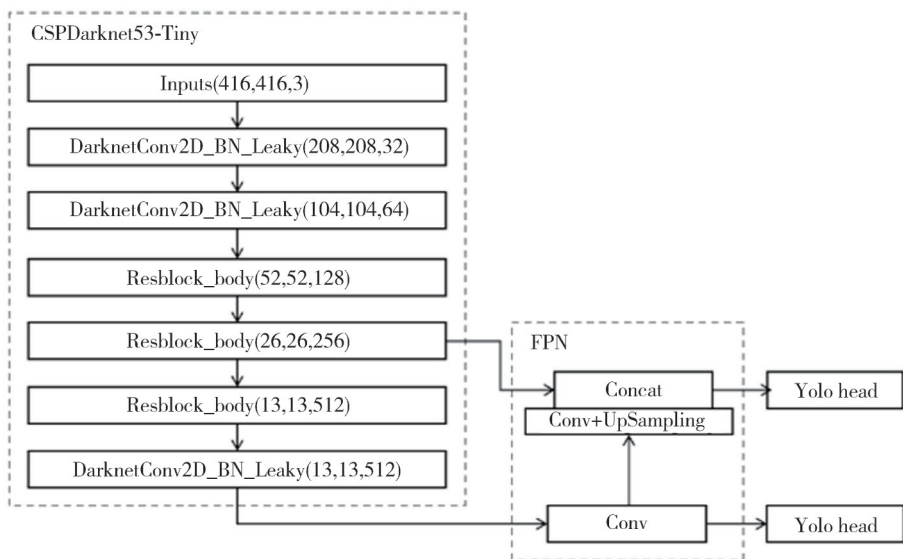


图 1 YOLOV4-Tiny 网络架构

Fig. 1 Architecture of YOLOV4-Tiny network

残差单元(Resblock)的结构如图 2 所示,它由 4 个连续的基本 Conv 块和 1 个 Maxpool 块构成。通过拆分和重新组合数据结构,残差单元能够使网络学习到更深层次的特征,同时缓解深度学习中常见的梯度消失问题,从而提升目标检测的精度和效率。输入的特征图经过这些层的处理后,能够完成对输入特征的初步提取和处理。

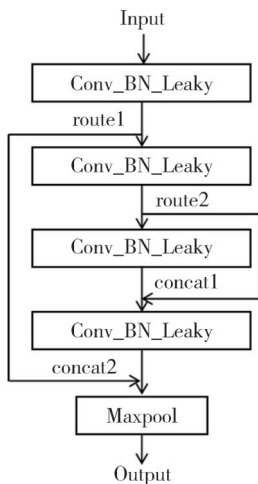


图 2 残差网络

Fig. 2 Residual network

特征金字塔网络中上采样(UpSample)模块能够结合不同尺度的特征信息,从而丰富检测区域的细节,增强模型对不同尺寸目标的检测能力。该网络对输入特征进行处理后,输出的两个特征图通过残差连接和卷积操作进一步整合,最终形成头部网络部分,生成用于预测边界框的特征层。

1.2 基于 HLS 的硬件加速原理

高层次综合(High-Level Synthesis, HLS)工具能够将高级语言(如 C、C++ 或 SystemC)描述的算法自动转换为寄存器传输级实现,直接对 Xilinx 系列的 FPGA 进行编程,大大减少了设计者使用传统寄存器传输级描述进行 FPGA 开发的时间,提高了 FPGA 的设计效率。同时, HLS 代码更易于编写和优化调整。此外, HLS 设计还具有良好的可移植性,使设计者能够轻松适配不同的硬件平台,而无需重新进行底层硬件描述。

2 硬件系统设计

2.1 分块流水化设计

在多数应用情境中,边缘设备的存储空间十分有限,因此,如何高效利用存储资源变得尤为重要。分块流水化设计将神经网络的计算流程分为多个可并行执行的小块。每次计算前只需加载小块数据到缓存,减少了频繁访问外部存储的延迟。同时,通过在并行执行过程中引入流水线设计, FPGA 能够同时处理多个数据流,避免了数据依赖性对计算速度的限制。在 FPGA 上部署卷积神经网络时,神经网络需要大量的卷积操作提取特征,卷积运算通常占总体计算量的 90%^[19]以上,由于 YOLOv4-Tiny 采用轻量化的结构,相较于完整的 YOLOv4,其计算量有所降低,但卷积

运算仍然是核心瓶颈,占据绝大部分硬件资源与计算时间。因此,优化卷积操作是提升FPGA部署性能的关键点。

以网络结构中第一次卷积操作为例,本文采用的分块流水化设计的具体步骤如下:

首先,对输入的RGB特征图进行零填充与分块,如图3所示。零填充是确保输入和输出的尺寸一致,避免边缘信息丢失,使网络能够更好地学习图像的全局特征。然后,将零填充后的特征图等分为 27×27 大小的子特征图。为了避免分块后子特征图边缘的有效数据在卷积操作中丢失,相邻的子特征图会重复两个像素。为了适配后续卷积层输入通道数均为4的倍数的硬件优化设计,本方案将并行度设置为4。经过特征图分块流程后,输入数据被规范化为多个 $4 \times 27 \times 27$ 维度的子特征图。针对初始RGB特征图仅有3个通道的特性,在首次分块处理时对第4个通道的所有数据赋值为0,以避免第4个通道对第一次卷积运算产生影响。

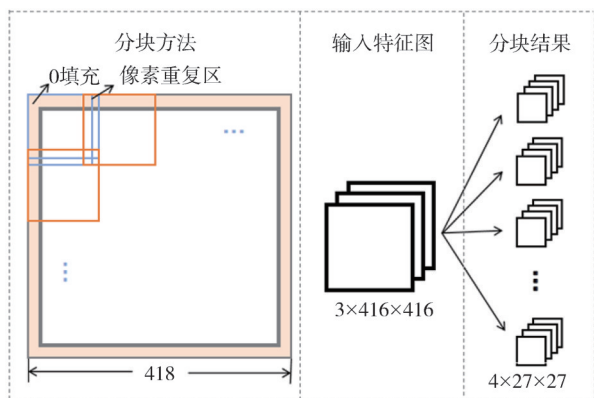


图3 分块设计

Fig. 3 Block design

然后,进行 $4 \times 27 \times 27$ 大小的子特征图与 $4 \times 3 \times 3$ 卷积核的流水化卷积操作。由于卷积操作由重复的乘加运算组成,具有高度的计算规律性,可通过HLS的PIPELINE优化指令,将卷积运算中的嵌套循环结构重构为深度并行的流水线架构:将原本串行的乘累加运算拆解为多级流水阶段,确保每个时钟周期都能启动一组新的计算任务,从而最大化计算吞吐量,这种架构适合FPGA的硬件特性,能够充分发挥其并行计算优势;同时,通过将具有重复特性的子特征图与权重存储于片上存储,可以减少外部存储访问的延迟;此外,FPGA可配置多个独立计算单元并行处理不同数据块,形成空间-时间双重并行架构,在空间

维度上实现通道并行和数据块并行处理,在时间维度上采用深度流水线技术使数据加载、计算和存储操作重叠执行,通过多维度协同优化进一步提高计算吞吐量。设置输出通道数为32,卷积流程如图4所示。

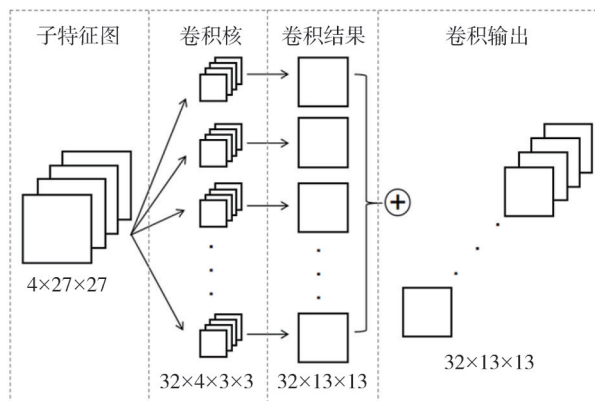


图4 卷积流程

Fig. 4 Convolution process

通过上述步骤,本文的分块流水化设计能够有效利用FPGA的并行处理能力,降低卷积神经网络的推理功耗并提高计算吞吐量,使得YOLOv4-Tiny网络能够更好地部署在对实时性要求较高的FPGA上。

2.2 双缓存结构

在FPGA上部署卷积神经网络时,由于硬件资源有限,数据无法全部缓存到片上缓冲区。为了减少数据缓存对推理速度的影响,本文采用乒乓操作来优化推理流程。

乒乓操作可以通过双缓存切换机制实现计算与数据传输并行化,具体而言,通过配置两个独立的缓存区,当计算单元处理一个缓存区的数据时,同时可以将外部数据加载到另一缓存区,通过交替切换缓存区实现数据加载与计算的时间重叠。通常情况下,计算所需的时间大于加载时间,因此,这种设计能够覆盖加载时间,实现连续且并行的数据处理流程,提升数据传输和处理的速率,进一步优化系统性能。通过对卷积计算、存储卷积结果以及卷积计算过程中的加载数据和计算卷积块进行双缓存设计,FPGA能够在数据传输和计算处理之间实现并行操作,从而提高神经网络推理速度,减少整体延迟,并提高资源利用率。如图5所示,“Load_in”表示加载特征图输入数据,“Load_w”表示加载权重数据,“Compute Block”表示计算子特征图。

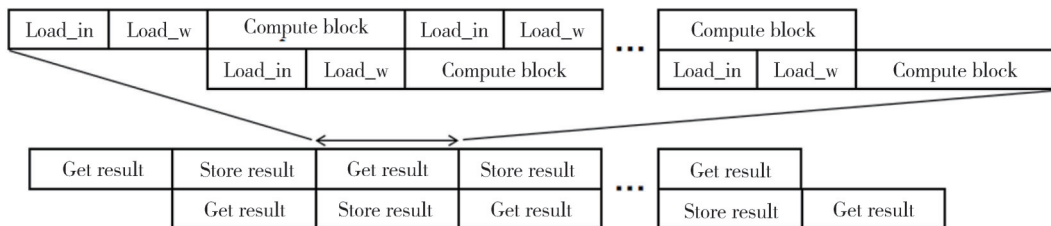


图 5 双缓存结构

Fig. 5 Double buffering structure

3 实验验证与评估

3.1 实验环境

本实验采用 Xilinx 公司推出的 ZYNQ7020 开发平台作为硬件基础,该开发板的核心主控芯片型号为 XC7Z020-CLG400-2,由处理系统和可编程逻辑两部分组成,其中,FPGA 上查找表(Look-Up Table, LUT)、DSP、块随机存取存储器(Block Random Access Memory, BRAM)和触发器(Flip-Flop, FF)的可用资源数分别为 53 200、220、280 和 106 400。此外,开发板上还集成了一个 ARM 双核处理器和两片 DDR3 内存,如图 6 所示。



图 6 FPGA 实验环境

Fig. 6 FPGA experimental environment

实验采用的神经网络模型为 YOLOv4-Tiny,数据集为 COCO 数据集,网络参数经过了 16 位定点量化。使用 Vivado HLS 2019.2 设计卷积 IP 核,采用 Vivado 2019.2 进行综合布局布线,并利用 Xilinx Vitis IDE 2019.2 进行软件开发,在软件开发阶段用 Xilinx 提供的高精度计时函数来记录网络推理时间。

3.2 目标检测实验

为了验证本文提出的分块加速优化方法仅对推理过程进行加速而不影响推理结果的准确性,设计了一组去掉本文流水线设计而保持其他操作不变的无流水线设计对比实验。实验分为两部分:一是使用分块流水线设计对检测网络进行加速;二是去掉 FPGA 的并行流水线设计,使检测网络串行执行,作为对比实验。实验检测结果如图 7 所示。



(a) 无加速检测效果图



(b) 加速检测效果图

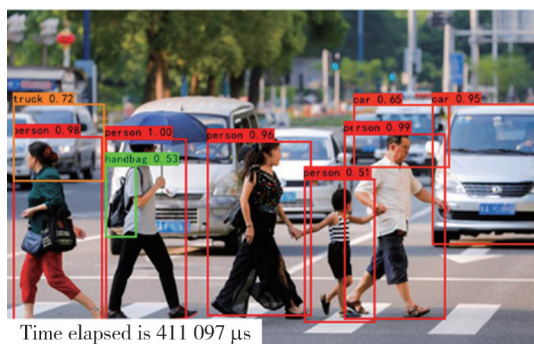
图 7 效果对比实验

Fig. 7 Effect comparison experiment

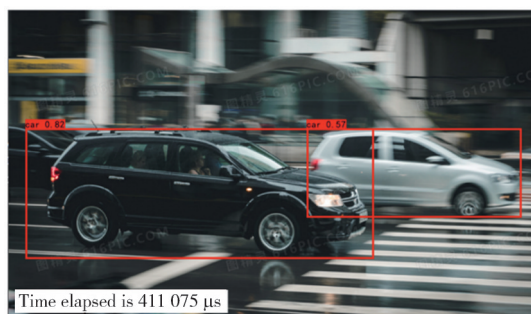
实验对比了优化前后的检测结果准确率和检测时间。图中检测框左上角显示了目标物体的检测结果,以图 7(a) 中最大的检测框为例,信息表明网络推断该框内物体为公共汽车的概率为 94%;左下角的信息则显示了网络推理整张图片所耗费的时间。

通过对比图 7(a) 和图 7(b) 可以发现, 检测网络加速后, 图中的检测框、物体类别及其概率信息均保持一致, 仅推理时间有所减少。这表明本文采用的分块加速优化方法仅缩短了网络推理时间, 而不会影响目标检测的准确性。

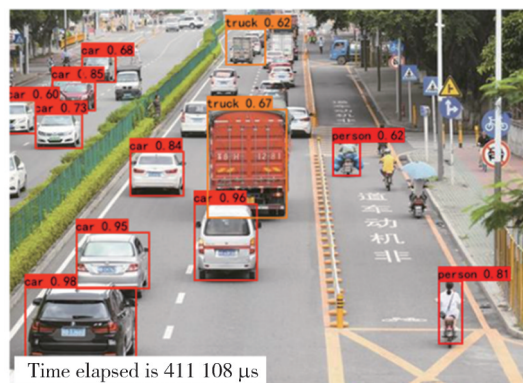
为了验证本文提出的加速方法在实际驾驶环境中的性能, 模拟了道路物体检测场景, 分别从正面、侧面和后方 3 个视角对车辆目标以及行人进行实时检测。实验结果如图 8 所示, 展示了检测网络在不同视角下的检测效果。通过多角度、多目标的测试, 能够更好地评估加速后的检测网络在实际驾驶场景中的鲁棒性和实用性, 为智能驾驶系统的部署提供可靠的技术支持。



(a) 从正面对行人车辆的加速检测



(b) 从侧面对车辆的加速检测



(c) 从后面对行车辆的加速检测

图 8 情景测试实验

Fig. 8 Scenario testing experiment

从图 8(a) 中可以看出, 经过加速优化的目标检测网络能够准确识别行人目标, 检测框位置精确, 置信度高, 展现了推理网络对行人目标的高效检测能力; 从图 8(b) 中可以看出, 网络对行驶中的车辆目标同样表现出色, 能够快速且准确地定位并识别车辆, 即使在比较模糊的图像中也能保持稳定的检测效果; 而在图 8(c) 中, 尽管由于远处目标存在重叠而导致部分目标出现漏检, 但近距离目标的检测结果依然准确可靠, 体现了网络在复杂场景下的鲁棒性。

综合 3 种场景的测试结果, 经过分块加速优化后的目标检测网络在准确性、实时性和稳定性方面均表现出色, 能够有效保障车辆行驶过程中的安全性, 为无人驾驶领域的目标检测技术提供了重要的实践参考和技术支持。

3.3 性能分析

为研究分块加速优化方法对硬件资源消耗及其性能的影响, 实验设计分为两部分: 基于分块乒乓操作的非流水线设计和基于分块乒乓操作的并行流水线设计。在 HLS 中分别为两种实验设计了各自的 IP 核, 软件开发阶段不做改变。实验可编程逻辑端的时钟频率为 120 MHz, 处理系统端的 ARM 时钟频率为 667 MHz。

本文将从硬件资源消耗、模型推理时间、推理功耗和处理器运算能力(GOPS)4 个角度对实验结果进行分析。其中, 1 GOPS 代表处理器每秒可进行 10 亿次操作, 计算公式如式(1)所示。

$$GOPS = \frac{a}{T_s} \times 10^{-9}, \quad (1)$$

式中: T_s 为模型单次推理时间; a 为本文所设计的 YOLOv4-Tiny 网络加速器完成单次推理所需的总操作数。1 GOPS/W 表示每瓦每秒十亿次运算, 是结合功耗的性能指标, 用来衡量 FPGA 的能效比。

相关实验数据如表 1 所示。结果显示, 引入流水线设计后, LUT 和 FF 等硬件资源的消耗有所增加, 尤其是 DSP 的使用率达到了 100%, 表明流水线设计充分挖掘了 DSP 的计算潜力, 提升了计算效率。对于推理时间, 在非流水线设计中, 模型计算过程完全串行执行, 尽管乒乓操作减少了参数读取时间, 但与串行计算的时间相比可忽略不计; 而在并行流水线设计中, 通过在每小块推理过程中进一步引入流水线技术, 计算任务被分解为多个阶段并行执行, 提高了硬件资源利

用率和计算效率,推理时间缩短。对于推理功率,由于采用流水线设计后硬件资源的使用量增加,推理功耗增加了约 0.7 W,但是相比非流水线设计,引入流水线设计具有更大的处理器运算能力,且能量效率更高。

表 1 实验结果
Tab. 1 Experimental results

相关参数	非流水线设计	流水线设计
LUT 使用率/%	43.29	56.92
FF 使用率/%	25.04	30.74
BRAM 使用率/%	67.50	67.50
DSP 使用率/%	34.09	100.00
推理时间/s	149.40	0.41
推理功耗/W	2.322	3.021
处理器运算能力/GOPS	0.046	16.788
每瓦运算能力/(GOPS·W ⁻¹)	0.014	5.557

总得来说,本文的流水线设计能有效利用硬件资源提升系统的运算能力,对 YOLO 系列模型的部署提供了有价值的参考。由于本文所用的实

验平台硬件资源有限,特别是 DSP 资源有限,若在 DSP 资源更丰富的平台上部署,可以通过增加并行度来进一步提升加速性能。

将本文在 ZYNQ7020 平台上设计的 YOLOv4-Tiny 硬件加速器与其他面向 YOLO 网络的 FPGA 加速器在资源占用和实时性等方面进行对比,结果如表 2 所示。文献[15]采用了基于卷积输入输出并行组合的加速策略,该方案在每个计算周期内需并行处理多个计算任务,依赖 BRAM 存储大量中间数据和权重参数。尽管该设计在功耗控制方面略优于本文方案,但其系统算力要低于本文提出的设计方案。文献[16]所采用的加速方法受限于 DSP 资源,且缓存数据时未能有效利用 BRAM 资源,导致传参功耗增加,同时推理速度较本文设计也有所下降。文献[17]采用软硬件协同优化策略,设计了复杂的脉动阵列结构,提高了 LUT 资源的消耗量,导致其功耗上升。

表 2 与其他文献 FPGA 平台加速器设计的比较

Tab. 2 Comparison with other literature on FPGA platform accelerator design

来源	模型	硬件平台	频率/MHz	LUT 资源数	DSP 资源数	BRAM 资源数	推理 时间/s	推理 功耗/W	运算能力/ GOPS	每瓦运算能力/ (GOPS·W ⁻¹)
文献[15]	YOLOv4-Tiny	PYNQ-Z2	150	40 569	218	491	—	2.980	10.960	2.980
文献[16]	YOLOv4-Tiny	PYNQ-Z2	100	43 639	218	74.50	0.45	2.650	15.330	5.790
文献[17]	YOLOv4-Tiny	KV260	100	69 713	218	76	0.43	3.712	16.047	4.322
本文	YOLOv4-Tiny	ZYNQ7020	120	40 192	220	94.50	0.411	3.021	16.788	5.557

相比之下,本文通过分块并行流水线设计,优化数据缓存和卷积计算,实现了资源的高效利用。同时,通过提升运行频率提高了计算效率,从而达到了更快的推理速度。尽管本文设计因 120 MHz 的运行频率导致功耗高于文献[16],但在 GOPS 性能上更具优势。综上所述,本文在资源消耗、推理速度和功耗之间取得了较好平衡,综合性能优于其他设计。

4 结 论

本文针对 YOLOv4-Tiny 网络在 FPGA 上的部署进行了硬件加速设计。针对卷积操作中高度的重复乘加运算特点,通过分块流水化设计和双缓存机制,有效利用 FPGA 的并行处理能力,提高吞吐量并减少整体延迟,从而提升了神经网络推理的速度,同时降低了推理过程的功耗。在 ZYNQ7020 硬件平台上进行了板级验证,并在多种场景下进行了性能测试,结果表明,本设计在推理速度和功耗方面的综合性能优于现有基于 FPGA 的硬件加速设计,同时能够在多种场景下

及时、准确地检测目标,为 FPGA 平台部署深度学习网络提供了有价值的参考。

利益冲突声明/Conflict of Interests

所有作者声明不存在利益冲突。

All authors declare no relevant conflict of interests.

参考文献

- [1] XU S, CHEN Z, ZHANG H, et al. Improved remote sensing image target detection based on YOLOv7[J]. Optoelectronics Letters, 2024, 20(4): 234-242.
- [2] SONG H, WEI C, YONG Z. Efficient knowledge distillation for remote sensing image classification: a CNN-based approach [J]. International Journal of Web Information Systems, 2024, 20(2): 129-158.
- [3] TOMAR H, DESHWAL D, TRIVEDI N. Convolutional neural network based language identification system: a spectrogram based approach [J]. Multimedia Tools and Applications, 2025, 84: 28951-28976.
- [4] WANG B, ZHANG F, ZHAO Y. LCH: fast RGB-D salient object detection on CPU via lightweight convo-

- lutional network with hybrid knowledge distillation[J]. The Visual Computer, 2024, 40(3): 1997-2014.
- [5] GUERROUJ F Z, RODRÍGUEZ FLÓREZ S, EL OUARDI A, et al. Optimizing convolution operations for YOLOv4-based object detection on GPU[J]. ITM Web of Conferences, 2024, 69: 04008.
- [6] ZHAO S, GAO S, WANG R, et al. Acceleration and implementation of convolutional neural networks based on FPGA[J]. Digital Signal Processing, 2023, 141: 104188.
- [7] NGUYEN D D, NGUYEN D T, LE M T, et al. FPGA-SoC implementation of YOLOv4 for flying-object detection[J]. Journal of Real-Time Image Processing, 2024, 21(3): 63.
- [8] WANG F, ZHANG M, WANG X, et al. Deep learning for edge computing applications: a state-of-the-art survey[J]. IEEE Access, 2020, 8: 58322-58336.
- [9] MOHAN N, HOSNI A, ATEF M. Neural networks implementations on FPGA for biomedical applications: a review[J]. SN Computer Science, 2024, 5(8): 1004.
- [10] LI W, HU H. FPGA-based object detection acceleration architecture design[J]. Journal of Physics: Conference Series, 2022, 2405(1): 012011.
- [11] ZHANG C, FANG Z, ZHOU P, et al. Caffeine: towards uniformed representation and acceleration for deep convolutional neural networks[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2019, 38(11): 2072-2085.
- [12] YU J, HU Y, NING X, et al. Instruction driven cross-layer CNN accelerator with winograd transformation on FPGA[C]//2017 International Conference on Field Programmable Technology (ICFPT), IEEE, 2017: 227-230.
- [13] 梅志伟, 王维东. 基于FPGA的卷积神经网络加速模块设计[J]. 南京大学学报(自然科学), 2020, 56(4): 581-590.
- MEI Zhiwei, WANG Weidong. Design of convolutional neural network acceleration module based on FPGA[J]. Journal of Nanjing University (Natural Science), 2020, 56(4): 581-590. (in Chinese)
- [14] LIU X, CHEN Y, HAO C, et al. WinoCNN: kernel sharing winograd systolic array for efficient convolutional neural network acceleration on FPGAs [C]//2021 IEEE 32nd International Conference on Application-Specific Systems, Architectures and Processors (ASAP), IEEE, 2021: 258-265.
- [15] 赵洋, 靳永强, 王艺钢. YOLOv4-tiny模型在边缘计算平台的加速设计[J]. 物联网技术, 2024, 14(1): 93-97.
- ZHAO Yang, JIN Yongqiang, WANG Yigang. Accelerated design of YOLOv4-tiny model in edge computing platform[J]. Internet of Things Technologies, 2024, 14(1): 93-97. (in Chinese)
- [16] 李雷, 黎远松, 石睿. 基于FPGA的YOLOv4-tiny网络的硬件加速与实现[J/OL]. 重庆工商大学学报(自然科学版), 2024. (2024-07-30). <https://link.cnki.net/urlid/50.1155N.20240730.1300.003>.
- LI Lei, LI Yuansong, SHI Rui. Hardware acceleration and implementation of YOLOv4-tiny network based on FPGA[J/OL]. Journal of Chongqing Technology and Business University (Natural Science Edition), 2024. (2024-07-30). <https://link.cnki.net/urlid/50.1155N.20240730.1300.003>. (in Chinese)
- [17] 姜明飞, 冯凤阳, 冯赞, 等. 基于YOLOv4-Tiny的硬件加速系统的设计与实现[J]. 电脑知识与技术, 2024, 20(10): 11-14.
- JIANG Mingfei, FENG Fengyang, FENG Yun, et al. Design and implementation of hardware acceleration system based on YOLOv4-tiny [J]. Computer Knowledge and Technology, 2024, 20(10): 11-14. (in Chinese)
- [18] 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 (CVPR), IEEE, 2016: 779-788.
- [19] GONG Y, LIU L, YANG M, et al. Compressing deep convolutional networks using vector quantization [EB/OL]. [2025-03-05]. <https://doi.org/10.48550/arXiv.1412.6115>.