·雷达系统与技术·

DOI:10.16592/ j. cnki. 1004-7859. 20240229001

基于FPGA的SAR图像目标检测加速器设计

汤 亮*,王小华,陈立福

(长沙理工大学 电气与信息工程学院, 湖南 长沙 410114)

摘要:主流的基于中央处理器(CPU)和图形处理器(GPU)的合成孔径雷达(SAR)图像目标检测算法,存在模型大、计算复杂度高、并行度低和功耗高等缺点,不适合部署在卫星和无人机等资源有限的平台上。文中在综合考虑板卡资源、功耗、推理速度和精度的条件下,设计了一种基于现场可编程门阵列(FPGA)的SAR图像目标检测加速器。该加速器采用的网络模型为优化后的YOLOv4-tiny,模型通过16位定点数优化数据位宽并加入空洞卷积来替换标准卷积,从而缩减了网络模型及参数,以便于部署在资源受限的FPGA上;在FPGA卷积层的实现中,采用了多重循环展开并行和循环分块并行的方法来加速卷积运算。实验结果表明,优化的算法在FPGA上获得了15.24 GOPS的吞吐量,每张图像识别速度为256 ms,介于CPU与GPU之间,但是由于FPGA硬件功耗仅为3.06 W,所以所提算法的能效比分别达到了CPU和GPU的18.4倍和7.3倍。

关键词:现场可编程门阵列;合成孔径雷达;硬件加速器;YOLOv4-tiny网络;目标检测

中图分类号:TN957.52 **文献标志码**:A **文章编号**:1004-7859(2025)06-0030-09

引用格式:汤 亮, 王小华, 陈立福. 基于FPGA的SAR图像目标检测加速器设计[J]. 现代雷达, 2025, 47(6): 30-38.
TANG Liang, WANG Xiaohua, CHEN Lifu. Design of SAR image target detection accelerator based on FPGA[J]. Modern Radar, 2025, 47(6): 30-38.

Design of SAR Image Target Detection Accelerator Based on FPGA

TANG Liang*, WANG Xiaohua, CHEN Lifu

(School of Electronical and Information Engineering, Changsha University of Science and Technology, Changsha Hunan 410114, China)

Abstract: The mainstream synthetic aperture radar (SAR) image target detection algorithms based on central processing unit (CPU) and graphics processing unit (GPU) have disadvantages such as large model size, high computational complexity, low parallelism, and high power consumption, and are not suitable for deployment on resource limited platforms such as satellites and unmanned aerial vehicles. A SAR image target detection accelerator based on field programmable gate array (FPGA) is designed in this paper, taking into account the board resources, power consumption, inference speed and accuracy. The network model adopted by the accelerator is an optimized YOLOv4-tiny architecture, which optimizes the data bit width with a 16-bit fixed-point quantization and adds dilated convolutions to replace standard convolutions, thereby reducing the network model and parameters for deployment on resource limited FPGA. In the implementation of FPGA convolutional layers, multiple loops unrolling and loop blocks parallelism methods are used to accelerate convolution operations. The experimental results show that the optimized algorithm achieved a throughput of 15.24 GOPS on FPGA, with a recognition speed of 256 ms per image, which is between CPU and GPU. However, due to the FPGA hardware power consumption of only 3.06 W, the energy efficiency ratios of the proposed algorithm reach 18.4 times and 7.3 times that of CPU and GPU respectively.

key words: field programmable gate array(FPGA); synthetic aperture radar(SAR); hardware accelerator; YOLOv4-tiny network; object detection

0 引 言

近些年来,随着科技的持续进步,合成孔径雷达(SAR)图像处理在军事侦察和地质勘探等领域的应用变得日益广泛。同时,人工智能技术的迅猛发展也带动着深度学习在SAR图像目标检测任务中的广泛应用[1]。目前,在中央处理器(CPU)和图形处理器(GPU)上部署的基于深度学习的SAR图像目标检测算法成为了主流。然而,尽管基于深度学习的SAR图像目标检测算法在CPU和GPU上的部署已经取得了显著的成果[2],但在星载和机载等资源受限场景下仍面临多重挑战:复杂网络设计为追求精度导致模型参数量庞大且存储开销激增,深层架构与密集计算推高了算力需求,而并行优化不足则致使多核计算效能未充分释放,系统功耗过高,严重制约其在轻量化平台的实际部署与应用效能。因此,

基金项目:国家自然科学基金资助项目(41201468)
收稿日期:2024-02-29 **修订日期**:2024-04-30

使用具有高处理速度、高并行能力和超低功耗的现场可编程门阵列(FPGA)来实现对 SAR 图像的目标检测具有重要的意义[3-4]。

在过去的研究中，有一些基于 FPGA 的 SAR 图像目标检测算法以及系统已经被提出并实现。例如，文献[5]提出的基于 YOLOv2 的加速器，文献[6]提出的基于 YOLOv3 的加速器，文献[7]提出的基于图神经网络的加速器，文献[8]提出的基于 Tiny-YOLO 模型的加速器。这些加速器利用了 FPGA 的高并行计算能力和灵活的可编程性，有效地减少了 SAR 图像目标检测所需要的推理时间。但是，以上方法还存在许多不足的地方，如 FPGA 部署的网络模型太大、消耗的 FPGA 资源太多、识别准确率太低等问题。

针对这些问题，本文提出了一种基于 YOLOv4-tiny[9] 的 SAR 图像目标检测方法，并创新性地将该模型部署于低功耗 FPGA 硬件平台。在模型优化方面，为了降低模型所需资源，本文引入了空洞卷积[10]对网络进行层级化精简，构建了稀疏化特征提取通道以降低计算复杂度。为实现 FPGA 硬件的高效部署，本文采用了 16 位定点数量化的方法[11]，替换了原本网络使用的 32 位数浮点数量化，显著减少了 FPGA 部署所需的逻辑单元和存储资源的消耗。在硬件部署方面，本文采用了 FPGA+高级精简指令集架构处理器(ARM)的架构，在 FPGA 端部署采用了多重循环展开并行和循环分块并行的卷积核模块，在 ARM 端部署计算量较少的初始化、图像预处理和 YOLO 检测后处理等模块。基于公开 SAR 舰船数据集的实验表明，本文提出的加速器，推理时间低于对比的其他 FPGA 加速器，其能效比分别是 CPU 和 GPU 的 18.4 倍和 7.3 倍。

1　YOLOv4-tiny 网络及模型压缩

YOLOv4-tiny 是 YOLOv4 的一个优化版本，专为减少模型参数和计算量而设计，旨在提高模型的运行速度和实用性。相较于原始的 YOLOv4，YOLOv4-tiny 具有更简化的卷积层结构和更少的参数数量。这使得它在推理阶段表现得更为迅速，尤其适合在资源有限的嵌入式设备和实时应用中使用。

1.1　YOLOv4-tiny 网络

YOLOv4-tiny 采用 CSPDarknet53-tiny 网络作为骨干网络，而非 YOLOv4 方法所采用的 CSPDarknet53 网络。与传统残差网络中的 ResBlock 模块不同，CSPDarknet53-tiny 网络采用了 CSPBlock 模块。该模块将特征图分成两部分，并通过交叉阶段残差边将它们组合起来，使梯度流能够在两个不同的网络路径中传播，从而增加梯度信息的相关差异。与 ResBlock 模块相比，CSPBlock 模块能够提高卷积网络的学习能力。尽管这会使计算量增加 10%~20%，但它能够提高准确性。此外，为了减少计算量，YOLOv4-tiny 还删除了 CSPBlock 模块中计算量较高的空间金字塔池化模块。YOLOv4-tiny 网络结构如图 1 所示。

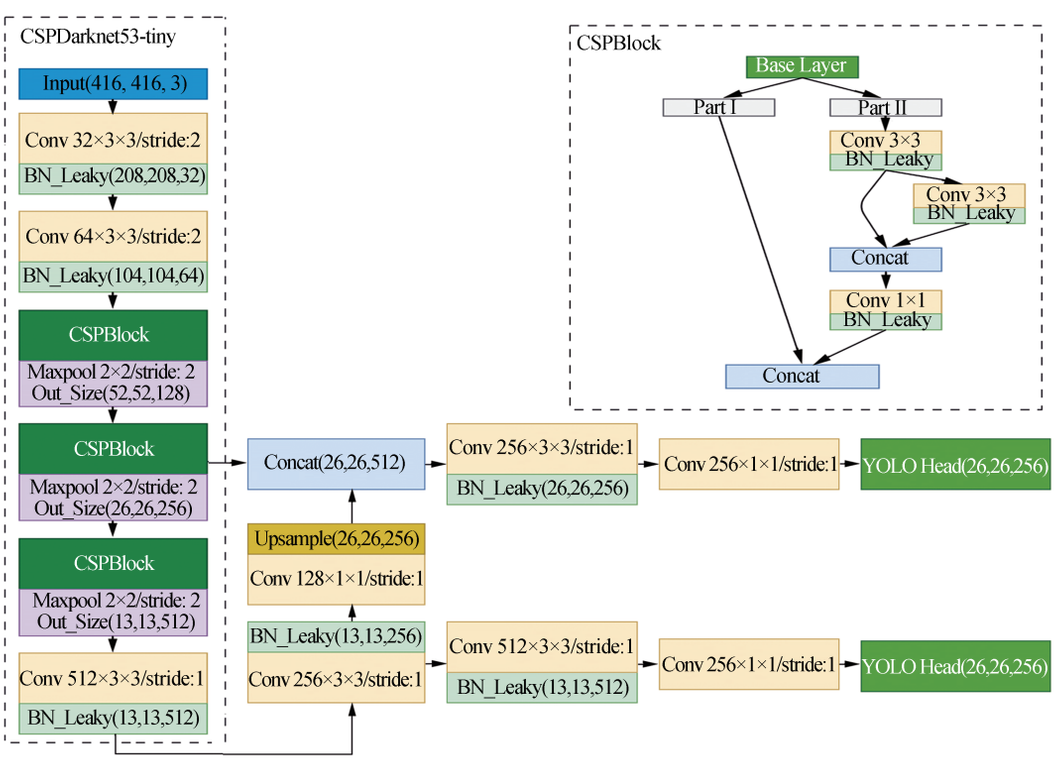


图 1　YOLOv4-tiny 网络结构

Fig. 1　YOLOv4-tiny network architecture

YOLOv4-tiny 和 YOLOv4 在激活函数方面也有所不同。YOLOv4-tiny 为了进一步简化计算过程,在 CSPDarknet53-tiny 模块中放弃了原来的 Mish 函数作为激活函数,而采用了 LeakyReLU 函数作为激活函数。函数如式(1)所示,相对于 Mish 激活函数,LeakyReLU 是一种更为传统的激活函数,它在负半轴上给予一个较小的斜率,使得负数部分也能有较小的输出,从而避免了神经元死亡等问题。这两种激活函数各有优缺点,在不同的应用场景下存在着不同的表现。

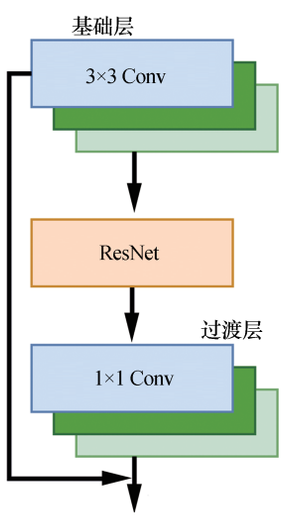
$$y_i = \begin{cases} x_i, & x_i \geqslant 0 \\ \dfrac{x_i}{a_i}, & x_i < 0 \end{cases} \tag{1}$$

1.2　模型压缩

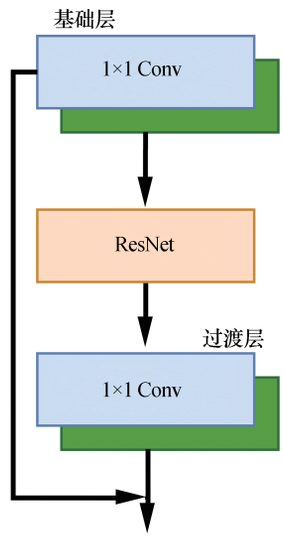
模型压缩是指将庞大而复杂的预训练模型转化为精简的小模型。业界主流的模型压缩方法有:知识蒸馏、轻量化模型架构(也叫紧凑的型设计)、剪枝和量化。为了便于在资源受限的 FPGA 设备上部署该模型,本文主要使用结构化剪枝和 16 位定点数量化对 YOLOv4-tiny 模型进行压缩与优化。

1.2.1　结构化剪枝

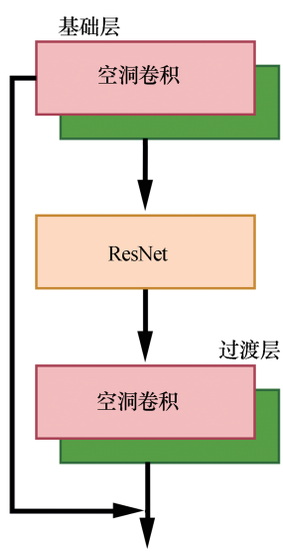
针对 YOLOv4-tiny 模型中部分卷积层使用了较大参数量的 512 个通道的卷积核,本文使用了结构化剪枝的方法来减少其参数量。通过使其 512 通道的卷积核直接减半,并将第三个 CSPBlock 模块的 3×3 卷积核替换成 1×1 卷积核,使得该模型计算量相较原来减少了 30%。然而,随着计算量的减少,计算精度也急剧减小,为了平衡二者之间的关系,本文把 1×1 卷积更换成了具有更高感受野的空洞卷积。具体改进 CSPBlock 模块结构的流程如下图 2a)~图 2c)所示。



a) 原 CSPResNet 结构
a) Original CSPResNet architecture



b) 改进 CSPResNet 结构
b) Modified CSPResNet architecture



c) 本文 CSPResNet 结构
c) Proposed CSPResNet architecture

图 2　改进 CSPBlock 模块结构流程
Fig. 2　Improving the CSPBlock architecture

1.2.2　16 位定点数量化

为了 YOLOv4-tiny 模型能更好地部署在 FPGA 上,本文网络采取了 16 位定点数量化,其中包含 7 位整数、8 位小数、1 位符号位。该方法不仅减少了存储需求,而且加速了计算的速度,降低了功耗。经实验表明,将 32 位浮点数权重和偏置换成 16 位定点数可以将整体模型减小 30%,从而使得对 FPGA 各硬件资源的需求大幅下降。

本文采取的浮点数与定点数量化之间的换算式如下

$$r = S(q - Z) \tag{2}$$

$$q = \text{round}\left(\frac{r}{S} + Z\right) \tag{3}$$

式中:r 为输入的浮点数;q 为量化后的定点数;Z 为实

数 0 经过量化后得到的对应整数；S 为浮点数和定点数的收缩关系。S 与 Z 的计算式分别如下

$$S=\frac{r_{\max}-r_{\min}}{q_{\max}-q_{\min}} \tag{4}$$

$$Z=\text{round}\left(q_{\max}-\frac{r_{\max}}{S}\right) \tag{5}$$

式中：$r_{\max}$、$r_{\min}$ 分别为 r 的最大与最小值；$q_{\max}$、$q_{\min}$ 分别为 q 的最大与最小值。从式(5)可以清晰地看出，当浮点数为 0 时，其对应的定点数 Z 也表示 0。这表明在转化过程中不存在精度损失，浮点数和定点数之间保持着精确的对应关系。

因为卷积网络中的卷积层、批归一化层和线性激活层本质上都是矩阵乘法，所以本文主要处理把浮点数矩阵的运算换成定点数矩阵的运算。具体步骤如下：

(1)如式(6)所示，设 $\boldsymbol{r}_1$、$\boldsymbol{r}_2$ 均为 N 阶的浮点数矩阵，i、j、k 分别表示矩阵 $\boldsymbol{r}_1$、$\boldsymbol{r}_2$ 的行列信息，$\boldsymbol{r}_3$ 为 $\boldsymbol{r}_1$、$\boldsymbol{r}_2$ 矩阵相乘之后的矩阵

$$\boldsymbol{r}_3^{i,k}=\sum_{j=1}^{N}\boldsymbol{r}_1^{i,j}\boldsymbol{r}_2^{j,k} \tag{6}$$

(2)设 S_1、Z_1 是矩阵 $\boldsymbol{r}_1$ 对应的 S 和 Z，S_2、Z_2、S_3、Z_3 同理，将各 S、Z 代入式(6)可得到式(7)，再将其整理并化简可以得到式(8)

$$S_3(q_3^{i,k}-Z_3)=\sum_{j=1}^{N}S_1(q_1^{i,j}-Z_1)S_2(q_2^{j,k}-Z_2) \tag{7}$$

$$q_3^{i,k}=\frac{S_1S_2}{S_3}\sum_{j=1}^{N}(q_1^{i,j}-Z_1)(q_2^{j,k}-Z_2)+Z_3 \tag{8}$$

其中，除了 S_1S_2/S_3 仍为浮点数，其他运算都已经转换为定点数。引入 R，使其代表 S_1S_2/S_3，其值为(0,1)之间的实数，可以表示成 $R=2^{-n}R_0$，其中 R_0 是一个定点实数。值得注意的是，定点数并不一定是整数，而是小数点的位置固定的实数。因此，如果存在 $R=2^{-n}R_0$，就可以通过 R_0 的位移操作来实现 $2^{-n}R_0$。

这样，整个推理过程就都在定点运算上完成了。通过上述推导过程，不难发现，量化后模型的推理已经完全转换为定点运算了。

2　加速器的实现方案

2.1　系统总体设计

本文设计采用 ZYNQ7020 FPGA 芯片来实现 SAR 图像目标检测加速器。考虑到芯片板载资源有限，本文设计采用了常见的部件级加速的方法。加速器算法部署分为了可编程逻辑(PL)和处理系统(PS)两部分，其中 PL 端为 FPGA 芯片 Artix-7，PS 端为双核 ARM Cortex-A9。根据对 YOLOv4-tiny 网络结构的分析，它的计算主要集中在 3×3 标准卷积、逐点卷积、空洞卷积、上采样、池化和连接(Concat)操作。其中标准卷积、逐点卷积、空洞卷积和池化占网络总体计算量的 96%以上，因此要在 FPGA 端重点加速。上采样和 Concat 操作运算的计算量相对于整个网络来说是微乎其微的，因此在 FPGA 上简单实现即可。PS 端主要处理计算量较少但较关键的任务，例如各模块的初始化、输入图像预处理和 YOLO 检测后处理，通过复用三种卷积 IP 加速器和池化 IP 加速器完成整个算法网络的前向推理。

硬件加速器整体设计框架如图 3 所示。其中，片内资源主要由块随机存取存储器(BRAM)构成，其是 FPGA 的重要存储单元，负责存储输入和输出数据；控制寄存器负责配置和控制各个模块的数据交换；双倍数据率同步动态随机存储器(DDR)控制器负责控制 DDR 存储网络的所有中间数据和最终计算结果。由于训练好的 YOLOv4-tiny 模型权重和输入检测的 SAR 图像占用空间较大，因此使用了安全数字(SD)卡作为初始存储设备，然后再将数据读取到片内存储上。

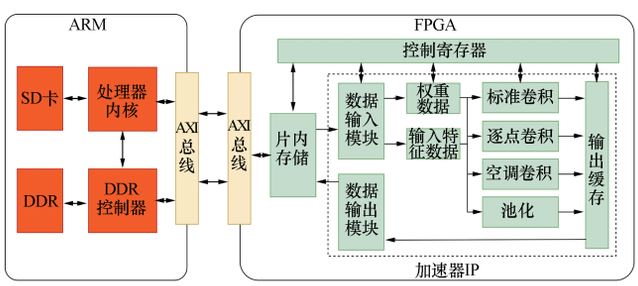


图 3　硬件加速器整体架构设计

Fig. 3　Architectural design of the hardware accelerator

2.2　卷积加速器设计

本文使用的改进版 YOLOv4-tiny 算法主要涉及三种不同类型的卷积操作，其中包括标准卷积、逐点卷积和空洞卷积。这些卷积操作的本质是对输入特征图和卷积核权重进行三维的乘法累加运算(MAC)。通常情况下，卷积操作通过四层嵌套循环来实现，其卷积计算伪代码如下所示：

```
for(mo=0;mo<Mof;mo++) #循环 4
  for(y=0;y<Moy;y++) #循环 3
  for(x=0;x<Mox;x++) #循环 3
   for(mi=0;mi<Mif;mi++) #循环 2
    for(ky=0;ky<Mky;ky++) #循环 1
     for(kx=0;kx<Mkx;kx++) #循环 1
     pix(mo,x,y)+=pix(mi,S*x+kx,S*y+ky)*
     W(mi,mo,kx,ky)
     pix(mo,x,y)= pix(mo,x,y)+bias(mo)
```

其中，S 为步幅，W 为权重，$bias$ 为偏置。为了更

有效地处理这些卷积运算,本文采用两种循环优化技巧,即多重循环展开并行和循环分块并行。多重循环展开决定了这些卷积循环的并行结构,从而影响了所需的寄存器和特定处理单元(PE)的大小;循环分块决定了片上缓冲区的容量需求,它将循环划分为多个块,这些执行块的数据从外部存储器中读取并存储在片上缓冲区中。合理使用这些优化技巧有助于提高卷积操作的效率和性能。

2.2.1　多重循环展开并行

卷积四层循环多维度展开如图 4 所示,其使用了多个维度来描述一个给定的卷积中每个卷积层的特征图和卷积核映射的大小关系。其中,(Mkx,Mky)表示卷积核的尺寸;(Mix,Miy)和(Mox,Moy)分别表示输入特征图和输出特征图的宽度和高度;Mif 和 Mof 分别表示输入和输出特征图的数量;(Pkx,Pky)、(Pox,Poy)分别表示循环展开中输入和输出需要并行计算像素的宽度和高度;Pif 和 Pof 分别表示像素输入和输出特征图并行计算的数量。这些硬件设计中的循环展开的参数将直接影响加速效果和硬件资源的使用情况。

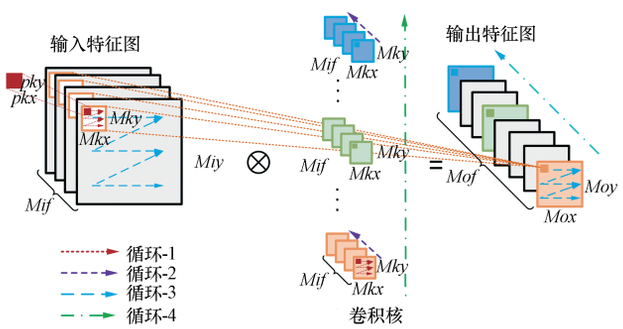
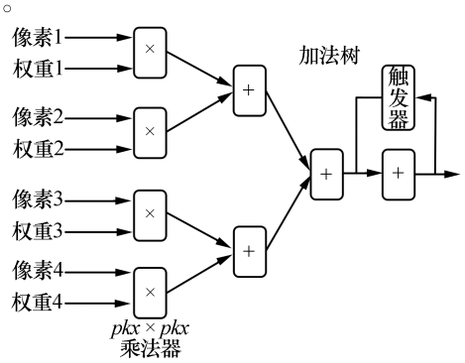


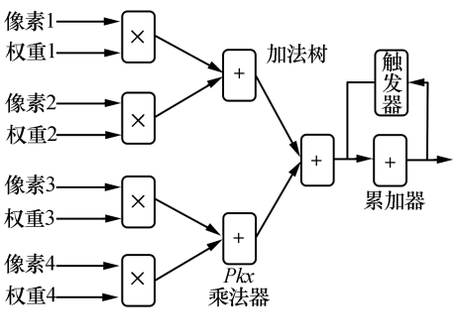
图 4　卷积四层循环多维度展开图

Fig. 4　Multi-dimensional unfolding of four-layer convolutional loop

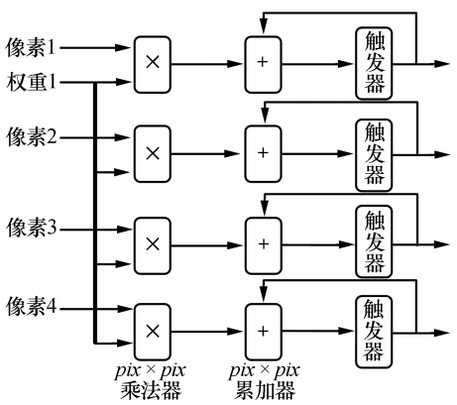
各循环展开并行计算架构如图 5 所示。通过展开卷积的不同循环,可以实现不同计算的并行化,这将直接影响用于卷积操作的 PE 阵列的最佳架构。因为不同的展开方式直接影响数据的重用机会和内存的访问模式,所以选择最优的循环架构可以确保卷积操作在硬件上能够以最佳性能运行,从而达到加速卷积运算的目的。



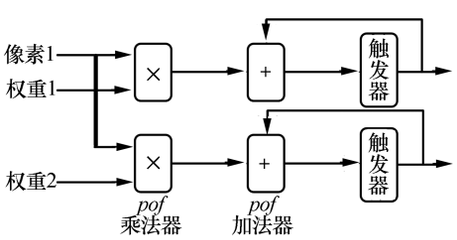
a) 循环-1
a) Loop-1



b) 循环-2
b) Loop-2



c) 循环-3
c) Loop-3



d) 循环-4
d) Loop-4

图 5　各循环展开并行计算架构图

Fig. 5　Parallel computing architecture of loop unfolding

各循环展开具体如下:

(1)循环-1 展开:如图 5a)所示,该循环每个周期都需要计算来自相同特征图和卷积核映射不同(x,y)位置的 $Pkx \times Pky$ 个像素和权重的内积。为了实现这个计算,本文使用一个带有 $Pkx \times Pky$ 个并行乘法的扇入加法树,将它们相加。同时,使用累加器将加法树的输出与之前的部分和相加起来,以完成内积计算。

(2)循环-2 展开:如图 5b)所示,该循环每个周期都需要计算来自不同特征图和卷积核映射的不同的 Pif 个像素和权重,这些像素和权重位于相同的(x,y)位置,用于内积的计算。内积操作与循环-1 展开具有相同的计算结构,但该循环只需要 Pif 个并行乘法的扇入加法树。

(3)循环-3 展开:如图 5c)所示,该循环每个周期都需要计算来自相同特征图中不同(x,y)位置的 $Pix \times$

Piy 个像素与相同的权重相乘，因此，这个权重可以被重复使用 *Pix*×*Piy* 次。其计算架构可以使用 *Pix*×*Piy* 个累加器来串行累加乘法器的输出，而无需使用扇入加法树。

(4)循环-4 展开：如图 5d)所示，该循环每个周期都需要计算单个像素与相同(x，y)位置但来自不同的卷积核映射的 *Pof* 个权重的相乘，这个像素将被重复使用 *Pof* 次。其计算架构与使用 *Pof* 个乘法器和累加器的循环-3 展开是相同的。

上述四个卷积循环的各展开变量的值共同决定了总并行 MAC 操作的数量，将所需要的硬件乘法器的数量记为 *Pm*，可得式(9)

$$Pm = Pkx \times Pky \times Pif \times Pix \times Piy \times Pof \quad (9)$$

2.2.2　循环分块并行

由于 FPGA 的片内存储器容量不足以存储 SAR 图像在卷积过程中产生的所有数据，因此必须合理地使用外部存储器来存储每一层的权重和中间像素卷积产生的结果。为了解决片内存储不足的问题，采用了循环分块的方法将整个数据分成多个块，使这些块的数据可以容纳在片内存储区中。循环分块如图 6 所示，其中(*Dix*，*Diy*)、*Dif*、(*Dox*，*Doy*)和 *Dof* 分别表示输入分块的尺寸、输入分块特征图数量、输出分块尺寸和输入分块特征图数量。这些维度和变量的约束条件为 $1 \leqslant P^* \leqslant D^* \leqslant M^*$，其中 M^*、D^* 和 P^* 分别表示 M、D 和 P 作为前缀的任何维度或变量。这里本文通过适当调整这些参数来改变循环分块的大小，并增加数据的局部性以减少片外内存的访问次数，从而降低延迟和功耗，达到加速的目的。循环分块决定了所需的硬件内存缓冲区大小，具体来说，输入像素缓冲区的大小为 *Dix*×*Diy*×*Dif*×像素数据宽度，权重缓冲区的大小为 *Dkx*×*Dky*×*Dif*×*Dof*×权重数据宽度，输出像素缓冲区的大小为 *Dox*×*Doy*×*Dof*×像素数据宽度。

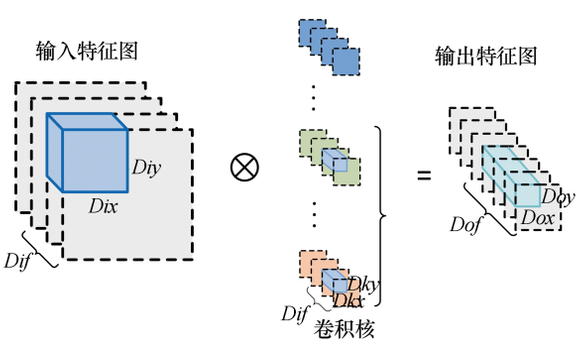


图 6　循环分块图
Fig. 6　Loop tiling diagram

2.3　池化加速器设计

池化运算在卷积神经网络中扮演着至关重要的角色，其主要任务是减小数据维度、剔除冗余信息并紧缩特征图，从而帮助网络更好地学习和理解图像特征。与卷积层不同，池化层不包含权重参数，计算负担较轻。因此，相对于卷积层，设计池化知识产权核(IP 核)相对简单。通常的池化操作包括平均池化和最大池化，YOLOv4-tiny 算法使用的是最大池化。类似卷积操作，池化也采用滑动窗口的方式，从特征图中提取相应位置的最大值，这些最大值构成了新的输出特征图。

池化架构展开如图 7 所示，在该架构中，蓝色框代表的 2×2 池化核在特征图上进行滑动操作，并将相应窗口的四个数据分别作为输入，输入到池化架构中。其中，输入 1 与输入 2 相比较，得到最大值 1，输入 3 与输入 4 相比较，得到最大值 2；其后，通过比较最大值 1 和最大值 2 来获得最终最大值。在池化 IP 核的设计中，由于池化计算相对简单，本设计采用开发软件中自带的高层次综合指令来进行流水线优化。这种设计指令可以有效地利用 FPGA 的并行性，提高了整体系统的性能和效率。

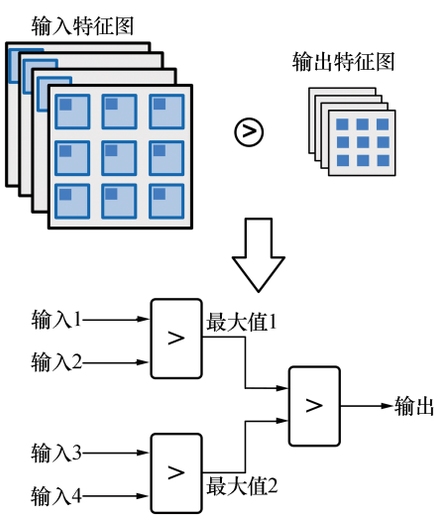


图 7　池化架构展开图
Fig. 7　Pooling architecture unfolding

3　实验及比较

3.1　实验环境

本文所提算法在计算机平台和 FPGA 平台上的实验环境如表 1 所示。

表 1　实验环境
Tab. 1　Experimental environment

描述	环境
操作系统	Windows10
CPU	i5-11400F
显卡	RTX3060
内存	32 GB
Python 版本	Python3. 6
FPGA 型号	ZYNQ-7020
Vivado 版本	Vivado2019. 2

3.2　计算机平台实验与讨论

3.2.1　SAR 舰船数据集

本文采用的是电子科技大学在 2020 年 1 月发布的高分辨率 SAR 图像数据集（HRSID）[12]。HRSID 是高分辨率 SAR 图像中用于船舶检测、语义分割和实例分割任务的数据集。该数据集共包含 5 604 张尺寸为 800×800 的 SAR 图像和 16 951 个舰船实例；图像的分辨率为 0.5 m、1.0 m 和 3.0 m，涵盖远海和近港口等不同场景和多种极化生成的图像。

3.2.2　算法模型训练和结果

在模型的训练中，HRSID 舰船数据集以 8∶1∶1 分为训练集、验证集和测试集。其中训练集为随机打乱的 4 484 张图像，验证集与测试集都为 560 张图像。训练采用余弦退火法，最大学习率为 0.020 0，最小学习率为 0.000 2，训练周期设置为 200。

平均准确率（mAP）阈值为 0.5 时随训练周期变化的曲线如图 8 所示。从图中可知 mAP 在前 25 个周期急剧上升，之后缓慢上升，并在 100 个周期后趋于平滑，最后 mAP 稳定在 87.2%。以上说明该算法的收敛速度快，训练后期波动较小且训练过程相对稳定，训练效果好。训练结束后，将训练好的算法模型的权重参数保存到 SD 卡中，用于进一步的 FPGA 平台的部署和实验。

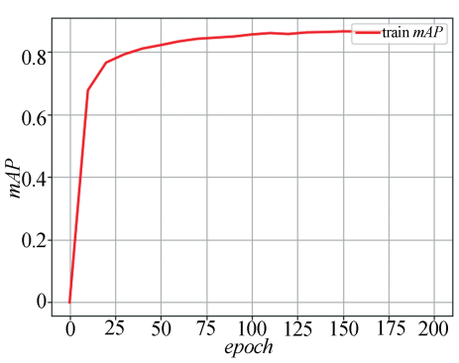
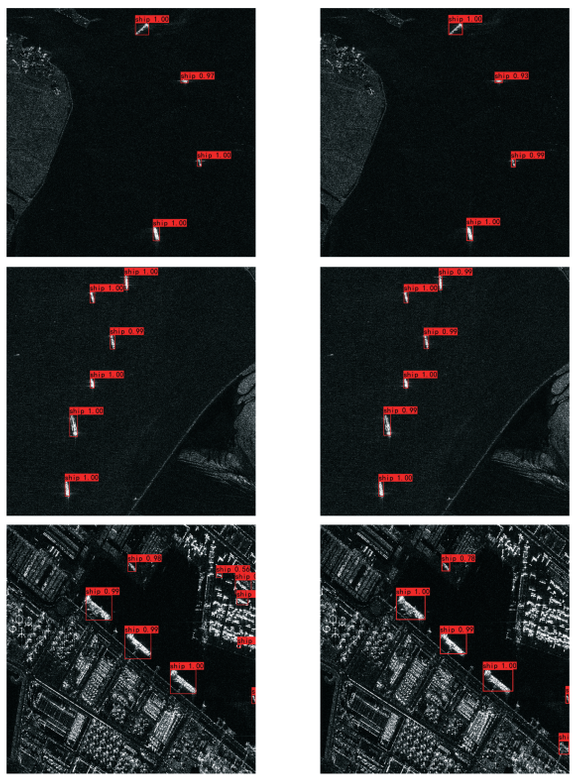


图 8　*mAP* 周期变化曲线
Fig. 8　Epoch-wise *mAP* progression curve

3.2.3　SAR 图像在不同平台识别结果对比

由于 CPU 和 GPU 均采用 32 位数据精度，所以 SAR 图像在 CPU 和 GPU 上识别的 mAP 是相同的。FPGA 采用的为 16 位数据精度，部署的为优化后的模型。如图 9 所示，其中图 9a）与图 9b）分别为 CPU 或 GPU 和 FPGA 的识别结果图。从图中各置信度可以看出，CPU 或 GPU 在检测简单背景时的总体精度是高于 FPGA 的，但 FPGA 在识别以复杂的港口为背景的 SAR 图像时，没有出现像 CPU 或 GPU 误检目标的情况。



a）CPU 或 GPU　　b）FPGA
a）CPU or GPU　　b）FPGA

图 9　CPU 或 GPU 与 FPGA 识别结果对比
Fig. 9　Comparison of recognition results between CPU or GPU and FPGA

3.3　FPGA 平台实验与讨论

3.3.1　FPGA 硬件资源利用率

本文所使用的 FPGA 型号是 ZYNQ7020，其主要资源包括 53 200 个查找表（LUT）、106 400 个触发器（FF）、140 个 BRAM、220 个数字信号处理单元（DSP48E）。FPGA 资源的利用率越高，意味着 FPGA 性能越优越，设计也更出色。表 2 展示了本文加速器中各资源的利用情况，其中具有较高利用率的 LUT 和 BRAM 为加速器提供了大量数据存储和传输的介质，这有利于实现数据的高效处理和高吞吐量，提高了加速器的整体性能。DSP48E 作为完成加法和乘法运算的主要计算资源，其利用率高达 100%，这表明卷积和其他乘法运算的并行度已经接近极限；而用于存储小数据和状态的 LUT 存储器（LUTRAM）以及 FF，其资源的利用率就降低了。

表 2　FPGA 资源的利用
Tab. 2　FPGA resource utilization

资源	利用数量/个	可用数量/个	利用率/%
LUT	40 234	53 200	76
BRAM	118	140	84
LUTRAM	7 853	17 400	45
DSP48E	220	220	100
FF	64 903	106 400	61

3.3.2　FPGA 性能分析与对比

为了验证论文中算法在 FPGA 平台上的性能，本文分别将算法运行在 FPGA 平台上、CPU 以及 GPU 平台上，进行了实验比较。通过实验获得的算法在不同平台上性能的比较结果如表 3 所示。在 FPGA 平台上，识别一张 SAR 图像所需时间为 256 ms，是在 CPU 平台上所需要时间的三分之一，并少于在 GPU 平台上所需时间。CPU 和 GPU 均采用 32 位数据精度，因此它们的 mAP 是相同的。然而，FPGA 采用的是 16 位定点数量化，因此 mAP 相对于 CPU 和 GPU 下降了 1.5%，但仍在合理范围内。衡量硬件计算性能的是吞吐量，该参数表示单位时间内能够处理的数据量。尽管 FPGA 的吞吐量低于 CPU 和 GPU，但由于功耗只有 3.06 W，其能效比分别是 CPU 和 GPU 的 18.4 倍和 7.3 倍。

表 3　所提算法在不同平台上运行的性能对比

Tab. 3　Performances comparison of proposed algorithm running on different platforms

实验平台	数据精度/bit	mAP/%	推理时间/ms	功耗/W	吞吐量/GOPS	能效比/(GOPS·W^{-1})
CPU	32	87.2	723	65.00	17.63	0.27
GPU	32	87.2	58	170.00	116.86	0.68
FPGA	16	85.7	256	3.06	15.24	4.98

目前，有许多研究致力于在 FPGA 平台上实现 YOLO 系列算法，表 4 将本文提出的加速器与其他文献提出的加速器进行了比较。

表 4　所提算法与不同文献算法的性能对比

Tab. 4　Performances comparison of the proposed algorithm and the algorithms in different literatures

文献	硬件平台	网络模型	数据精度/bit	主频/MHz	推理时间/ms	功耗/W	吞吐量/GOPS	能效比/(GOPS·W^{-1})
文献[13]	ZYNQ7020	YOLOv3-tiny	16	100	532	3.360	10.45	3.11
文献[14]	Ultra96V2	YOLOv3-tiny	8	250	121	4.260	31.50	7.40
文献[15]	ZYNQ7020	YOLOv4-tiny	16	180	1 040	2.800	6.53	2.33
文献[16]	ZYNQ7020	YOLOv4-tiny	16	100	18 025	2.384	—	—
文献[17]	ZYNQ7020	YOLOv4-tiny	16	130	383	2.750	18.32	6.66
本文	ZYNQ7020	YOLOv4-tiny	16	150	264	3.060	15.24	4.98

与文献[13]相比，在使用相同的硬件平台和不同的网络模型下，本文设计的推理时间提升了 268 ms，吞吐量提高了 4.79 GOPS；与文献[14]相比，在使用了不同的硬件平台、网络模型和数据精度的情况下，本文设计虽然在推理时间与性能上有所降低，但是使用的硬件平台资源是文献[14]的三分之一，数据精度则是其的一倍；在使用相同的硬件平台，本文算法相对于文献[15]与文献[16]，在除主频不同而其他均保持一致的情况下，推理时间与吞吐量都有了大幅提升；与文献[17]相比，虽然本文设计吞吐量降低了 3.08 Gops，但是在推理时间上提升了 119 ms。综上所述，本文提出的加速器性能优于对比的其他 FPGA 加速器。

4　结束语

综上所述，本文提出了一种基于 YOLOv4-tiny 的 SAR 图像目标检测方法，并成功将其部署在低功耗 FPGA 硬件平台上。通过引入空洞卷积和 16 位定点数量化的优化方法，成功降低了模型所需的硬件资源，使得算法适合部署在资源有限的平台上。在硬件部署方面，采用了 FPGA+ARM 的架构，充分利用了 FPGA 的高并行计算能力和灵活的可编程性。实验结果表明，所设计的加速器在保持较高识别精度的同时，实现了较快的推理速度，并且硬件功耗远低于 CPU 和 GPU。这表明了该设计在实际应用中具有重要的意义，特别是对于卫星和无人机等资源有限的平台。

然而，本文的方法还有一些局限性，例如在处理大规模数据集时可能会面临挑战。未来，将继续优化算法性能，进一步提高识别准确率与推理速度，并探索更多适用于 FPGA 的模型优化和部署策略。

参考文献(References)

[1] 冯博迪，杨海涛，李高源，等. 神经网络在 SAR 图像目标识别中的研究综述[J]. 兵器装备工程学报，2021，42(10)：15-22.
FENG Bodi，YANG Haitao，LI Gaoyuan，et al. Research summary of convolutional neural network in SAR image target recognition[J]. Journal of Ordnance Equipment Engineering，2021，42(10)：15-22.

[2] 曾祥书，王　强，黄一飞，等. 基于改进 YOLOv3 网络的 SAR 图像舰船目标检测方法[J]. 现代雷达，2023，45(2)：31-38.
ZENG Xiangshu，WANG Qiang，HUANG Yifei，et al. Ship target detection in SAR images based on improved YOLOv3[J]. Modern Radar，2023，45(2)：31-38.

[3] ZHANG N，WEI X，CHEN H，et al. FPGA implementation for CNN-based optical remote sensing object detection[J]. Electronics，2021，10(3)：282.

[4] YU Y X，WU C，ZHAO T D，et al. OPU：An FPGA-based overlay processor for convolutional neural networks[J]. IEEE Transactions on Very Large Scale Integration (VLSI)

Systems, 2019, 28(1): 35-47.

[5] 范家赫. 面向 SAR 舰船检测的深度学习算法轻量化及 FPGA 加速器设计[D]. 西安: 西安电子科技大学, 2020.
FAN Jiahe. Deep learning algorithm lightweight for SAR ship detection and corresponding FPGA accelerator design [D]. Xi'an: Xidian University, 2020.

[6] HU L, CHEN J. Ship target detection in SAR images based on FPGA[C]// 2022 3rd China International SAR Symposium. Shanghai: IEEE Press, 2022: 1-4.

[7] ZHANG B Y, KANNAN R, PRASANNA V, et al. Accurate, low-latency, efficient SAR automatic target recognition on FPGA[C]// 2022 32nd International Conference on Field-programmable Logic and Applications. Belfast, United Kingdom: IEEE Press, 2022: 1-8.

[8] YANG X, ZHUANG C, FENG W Q, et al. FPGA implementation of a deep learning acceleration core architecture for image target detection[J]. Applied Sciences, 2023, 13(7): 1-26.

[9] WANG L M, ZHOU K, CHU A L, et al. An improved light-weight traffic sign recognition algorithm based on YOLOv4-tiny[J]. IEEE Access, 2021, 9: 124963-124971.

[10] YU F, KOLTUN V, FUNKHOUSER T. et al. Dilated residual networks[C]// 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, HI: IEEE Press, 2017: 472-480.

[11] JACOB B, KLIGYS S, CHEN B, et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference[C]// 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT: IEEE Press, 2018: 2704-2713.

[12] WEI S J, ZENG X F, QU Q Z, et al. HRSID: A high-resolution SAR images dataset for ship detection and instance segmentation[J]. IEEE Access, 2020, 8: 120234-120254.

[13] YU Z W, BOUGANIS C S. A parameterisable FPGA-tailored architecture for YOLOv3-tiny[C]// 16th International Symposium on Applied Reconfigurable Computing. Toledo, Spain: Springer Verlag, 2020: 330-344.

[14] ADIONO T, PUTRA A, SUTISNA N, et al. Low latency YOLOv3-tiny accelerator for low-cost FPGA using general matrix multiplication principle[J]. IEEE Access, 2016, 4: 1-24.

[15] 曹远杰, 高瑜翔, 杜鑫昌, 等. 基于改进 YOLOv4-Tiny 的 FPGA 加速方法[J]. 无线电工程, 2022, 52(4): 604-611.
CAO Yuanjie, GAO Yuxiang, DU Xinchang, et al. FPGA acceleration method based on improved YOLOv4-tiny[J]. Radio Engineering, 2022, 52(4): 604-611.

[16] LI P, CHE C. Mapping YOLOv4-tiny on FPGA-based DNN accelerator by using dynamic fixed-point method[C]// 2021 12th International Symposium on Parallel Architectures, Algorithms and Programming. Xi'an: IEEE Press, 2021: 125-129.

[17] ZHAO S J, GAO S S, WANG R G, et al. Acceleration and implementation of convolutional neural networks based on FPGA[J]. Digital Signal Processing, 2023, 141: 104188.

作者简介:

汤 亮 男,1999 年生,硕士研究生,研究方向为 FPGA 加速器、SAR 图像目标检测;

王小华 男,1968 年生,博士,教授,研究方向为数字滤波器设计、电力系统谐波分析、信号检测与信号处理、神经网络理论与应用;

陈立福 男,1979 年生,博士,副教授,研究方向为 SAR 图像目标检测、分割、深度学习。