



## 基于FPGA的稀疏化卷积神经网络加速器

狄新凯<sup>1,2</sup>, 杨海钢<sup>1,2</sup>

(1. 中国科学院空天信息创新研究院, 北京 100094; 2. 中国科学院大学, 北京 100049)

**摘要:** 为消除卷积神经网络前向计算过程中因模型参数的稀疏性而出现的无效运算, 基于现场可编程门阵列(FPGA)设计针对稀疏化神经网络模型的数据流及并行加速器。通过专用逻辑模块在输入通道方向上筛选出特征图矩阵和卷积滤波器矩阵中的非零点, 将有效数据传递给由数字信号处理器组成的阵列做乘累加操作。在此基础上, 对所有相关的中间结果经加法树获得最终输出特征图点, 同时在特征图宽度、高度和输出通道方向上做粗粒度并行并寻找最佳的设计参数。在Xilinx器件上进行实验验证, 结果表明, 该设计实现VGG16卷积层综合性能达到678.2 GOPS, 性能功耗比为69.45 GOPS/W, 其性能与功耗指标较基于FPGA的稠密网络加速器和稀疏网络加速器有较大提升。

**关键词:** 卷积神经网络; 稀疏性; 现场可编程门阵列; 并行加速器; 数字信号处理器; 加法树

开放科学(资源服务)标志码(OSID):



中文引用格式: 狄新凯, 杨海钢. 基于FPGA的稀疏化卷积神经网络加速器[J]. 计算机工程, 2021, 47(7): 189-195, 204.

英文引用格式: DI X K, YANG H G. FPGA-based accelerator for sparse convolutional neutral network[J]. Computer Engineering, 2021, 47(7): 189-195, 204.

## FPGA-based Accelerator for Sparse Convolutional Neutral Network

DI Xinkai<sup>1,2</sup>, YANG Haigang<sup>1,2</sup>

(1. Aerospace Information Research Institute, Chinese Academy of Sciences, Beijing 100094, China;

2. University of Chinese Academy of Sciences, Beijing 100049, China)

**[Abstract]** In order to eliminate the invalid operations caused by the sparsity of the model parameters in the forward process of the Convolution Neural Network (CNN), a dataflow and parallel accelerator system for the sparse neural network are designed based on the Field Programmable Gate array (FPGA). By using a dedicated logic module, the non-zero elements in the feature map matrices and the convolution filter matrices are picked up. Then the valid data is transferred to the array consisting of Digital Signal Processor (DSP) for multiply-accumulate operations. On this basis, all relevant intermediate results are transferred to the adder tree to generate the final output feature map. Meanwhile, the coarse-grained parallelism is implemented along the width, height and output channel of the feature maps, and the optimal design parameters are searched for. Experiments are carried out based on Xilinx FPGAs for verification, and the results show that the design enables the sparse convolution layer in VGG to deliver performance of 678.2 GOPS and energy efficiency of 69.45 GOPS/W, displaying a considerable improvement of performance and energy efficiency compared with FPGA-based accelerators for the dense and sparse networks.

**[Key words]** Convolutional Neural Network (CNN); sparsity; Field Programmable Gate Array (FPGA); parallel accelerator; Digital Signal Processor (DSP); adder tree

DOI: 10.19678/j.issn.1000-3428.0058640

### 0 概述

近年来, 卷积神经网络(Convolutional Neural Network, CNN)模型<sup>[1-2]</sup>在计算机视觉、自动驾驶、自然语言处理等任务场景中得到广泛应用, 但CNN拓扑结构复杂、计算密集且参数规模大, 这对硬件系统

的计算能力造成极大的压力。研究人员通过分析CNN算法特性, 发现算法模型的权重参数和特征图参数均具有稀疏化特征。稀疏化特征说明算法在前向运算的过程中会出现大量“0”参与的无效运算操

**基金项目:** 国家自然科学基金(61876172); 北京市科委重大科研计划项目(Z171100000117019)。

**作者简介:** 狄新凯(1992—), 男, 博士研究生, 主研方向为高性能计算、计算机体系结构; 杨海钢, 研究员、博士、博士生导师。

**收稿日期:** 2020-06-15 **修回日期:** 2020-08-08 **E-mail:** dixinkai15@mails.ucas.edu.cn

作。如果在计算系统中只运行非“0”元素参与的有效操作,那么算法模型的计算量就能大幅减少。然而,传统针对CNN的并行加速器<sup>[3-5]</sup>或数据流优化<sup>[6]</sup>通常只适用于稠密矩阵运算,并不能在实际意义上减少操作数。因此,需要针对稀疏性做优化处理,以探索稀疏神经网络硬件加速器。

对于稀疏化的神经网络加速问题,学术界已有部分研究。文献[7-8]考虑权重数据的稀疏特征,所提方法的核心思想均是用特定的压缩格式来存储和传输权重,同时在电路中加入索引模块,利用压缩格式中的索引信息定位与非零权重相对应的特征值,进而只计算有效权重参与的运算操作。文献[9-10]考虑特征图数据的稀疏特征,所提方法的核心思想是在计算单元中加入判定模块,如果遇到值为0的特征图点,则跳过该次运算。此外,也有相关研究同时利用特征图稀疏性和权重稀疏性来避免无效运算,如文献[11]利用笛卡尔乘作为基本的操作来实现卷积,然而在该计算流程中,每个计算单元都需要设计复杂的坐标索引模块,并且中间结果较多,对缓存造成了很大压力。文献[12-13]基于ASIC为特征图和权重建立二进制掩码索引,然后通过对两者做“与”(AND)操作,选出有效值进行运算,但是此类方案需要额外线上动态地对特征值建立索引信息。文献[14-15]方法则是针对因特定结构化剪枝方式而造成的有规律的稀疏特征。此外,学术界大量工作集中在基于ASIC对稀疏化全连接层做加速,关于稀疏化的卷积层加速研究较少。

本文研究稀疏神经网络卷积层的计算数据流,基于现场可编程门阵列(Field Programmable Gate Array, FPGA)器件设计新的加速器硬件系统。根据特征图数据和权重数据的数据特点,设计逻辑处理模块从两者中筛选出有效的数据参与运算。在此基础上,利用FPGA器件特点研究逻辑处理模块并行操作的整体架构,并根据器件参数和模型参数的约束探索最佳设计参数。通过在Xilinx软硬件平台上实现具体的稀疏化VGG模型,验证本文方案的有效性和优越性。

## 1 卷积神经网络及其稀疏性分析

### 1.1 卷积神经网络

卷积神经网络是一种前馈网络,包含由卷积层和子采样层构成的特征抽取器以及由全连接层组成的分类器。在卷积层中,有多个卷积滤波器(卷积核)扫过该层的特征图像做卷积操作得到下一级特征图,如式(1)所示,卷积滤波器中的参数即为参与

运算的权重。式(1)的基本变量和参数在表1中做详细说明。

$$\begin{aligned} & \forall \{b, n, u, v\} \in [1, B] * [1, N] * [1, V] * [1, U], \\ & Y^{\text{CONV}}[b, n, u, v] = \\ & \sum_{c=1}^C \sum_{j=1}^J \sum_{k=1}^K (X^{\text{CONV}}[b, c, v+j, u+k] * F[n, c, j, k]) + \beta^{\text{CONV}}[n] \end{aligned} \quad (1)$$

表1 卷积层参数定义

参数	定义
$X$	输入特征图 ( $B \times C \times H \times W$ )
$Y$	输出特征图 ( $B \times N \times V \times U$ )
$F$	卷积滤波器 ( $N \times C \times J \times K$ )
$\beta$	线性偏置 ( $N \times 1$ )
$B$	输入特征图的批次
$W/H/C$	输入特征图的宽度/高度/深度
$U/V/N$	输出特征图的宽度/高度/深度
$K/J$	卷积核的尺寸 (通常 $K=J$ )

实际上,子采样层和全连接层也可以看作是特殊的卷积层统一到式(1)的表述中,例如全连接层就是卷积滤波器大小等于上层特征图大小的卷积运算,每个卷积滤波器做卷积运算的结果对应全连接层的一个神经元。

### 1.2 算法模型参数的稀疏性分析

在本文中,CNN网络的稀疏度定义为卷积滤波器矩阵以及特征图矩阵中零值的比例。一方面,利用特征图和权重参数的稀疏化特性可降低参数规模,进而减小系统存储压力;另一方面,设计硬件电路避免稀疏矩阵中“0”参与的运算,可降低运算代价,进而提高实时性。

神经网络中特征图和权重的稀疏性分析如图1所示。权重的稀疏性主要来源于剪枝优化技术,该技术被证实可以在保证神经网络精度的前提下有效地压缩模型。以文献[16-18]为代表的研究发现,训练时根据特定的规则将权重参数中不重要的值剪除(即置为0),不会损坏模型的识别精度。

此外,特征图的稀疏性主要来源于激活函数ReLU,因为ReLU将卷积操作的中间结果中非正值均置为0。以VGG16网络为例,文献[17]实验结果表明,卷积层中特征图0值点的比例最多可以达到80%,平均比例也在50%以上。与权重稀疏性不同,特征图的0值点都是运行时获取且依赖于输入图像数据,而前者在训练后便成为可线下分析的静态数据。

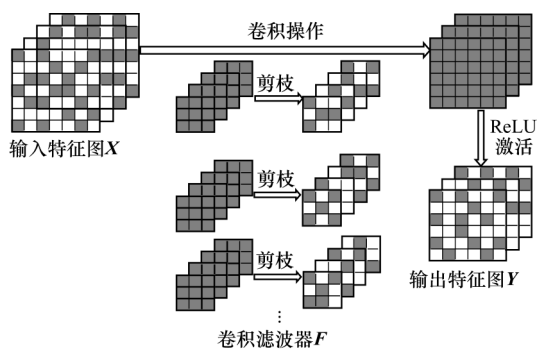


图1 权重剪枝优化及激活产生的稀疏性分析

Fig.1 Analysis of sparsity by weight pruning optimization and activation

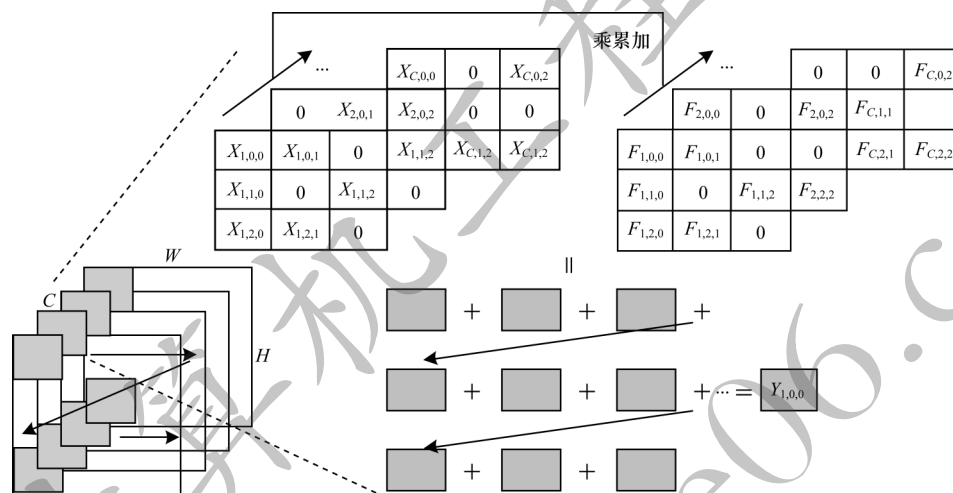


图2 卷积操作简化示意图

Fig.2 Simplified schematic diagram of convolution operation

在数据稀疏的情况下,  $K \times K$  个逻辑模块负责筛选出输入特征图和权重均为有效值的位置, 然后传输给乘累加器 (Multiplication Accumulator, MACC)。在基于FPGA的设计中, 通常采用片上DSP来完成这项工作。图3为  $K=3$  时  $3 \times 3$  个通道间向量-向量乘计算操作, 其中每个方格代表1对有效的特征图和权重数据, 需要做1次有效计算。在图3中, 假设每个逻辑模块后接了2个DSP, 则这组运算占用5个cycle时间。

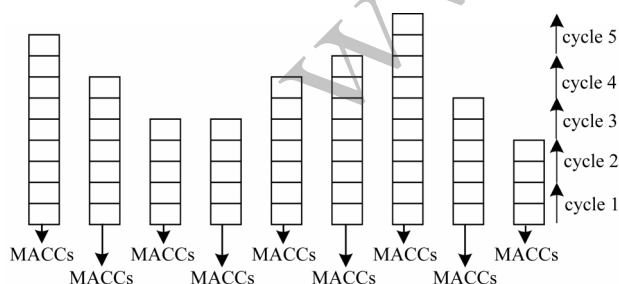


图3 通道间向量-向量乘计算操作示意图

Fig.3 Schematic diagram of vector-vector multiplication calculation operation among channels

## 2 加速器实现方案

### 2.1 卷积计算流程与权重压缩

在本文的设计中, 神经网络卷积操作计算流程简化示意图如图2所示, 共有  $N$  组  $C \times K \times K$  的卷积核在  $C \times W \times H$  的特征图上滑动 (示意图中仅展示一组)。卷积层参数有输入特征图 ( $X_{c,w,h}$ ) 和对应的权重滤波器 ( $F_{c,w,h}$ ) 以及输出 ( $Y_{n,w,h}$ )。在每组运算中, 先做  $C$  通道间向量-向量乘, 得到  $K \times K$  个中间结果后再进行累加, 产生最终输出特征图点。

对权重训练和剪枝后, 使其成为固定的数据, 在此基础上, 可以在线下对权重进行分析并以压缩格式存储和传输。如图4所示, 采用二维掩码 Mask 的格式对权重进行索引, 每组卷积核共有  $K \times K$  个  $\text{lb } C$  位宽的权重索引数据。特征图数据是动态数据, 需要线上实时产生, 因此, 处理方式方式为线上实时判断。

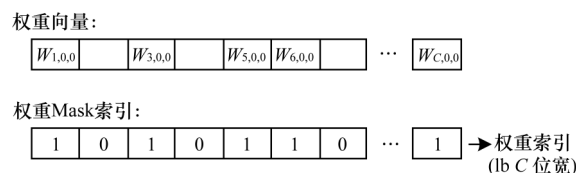


图4 权重数据及其Mask索引

Fig.4 Weight data and its Mask index

### 2.2 加速器整体架构设计

本文基于FPGA器件设计稀疏神经网络加速器, 其整体架构如图5所示。

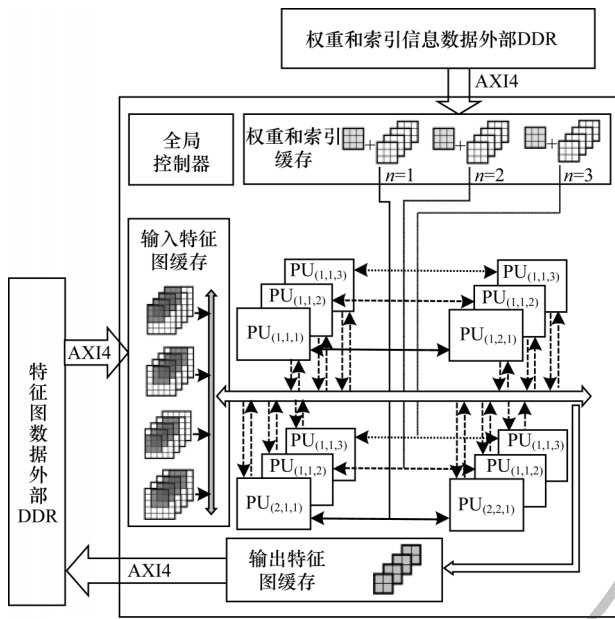


图5 稀疏神经网络加速器整体架构

Fig.5 Overall architecture of sparse CNN accelerator

在图5中,特征图和权重通过FPGA的AXI4接口从片外传入片上缓存,然后片上缓存再将数据传入处理模块(Processing Unit, PU)。为利用FPGA的并行特性,多个PU在输入特征图的横向尺寸 $W$ 、纵向尺寸 $H$ 和输出通道 $N$ 方向做并行操作,因此,共有 $P_w \times P_H \times P_N$ 个PU单元参与运算。每个处理模块承担 $C$ 通道 $K \times K$ 尺寸的输入特征图与其对应的权重做卷积以及通道间累加的计算工作,从而产生某一输出通道的输出特征图点。由于输入特征图和权重的稀疏特性,PU会主要通过逻辑单元来提取出有效的值参与运算。PU单元得到的结果通过输出特征图缓存,并且同样通过AXI4接口输出到片外。

### 2.3 处理单元设计

每个PU需要进行有效数据提取和计算工作,图6为处理单元PU的具体结构。

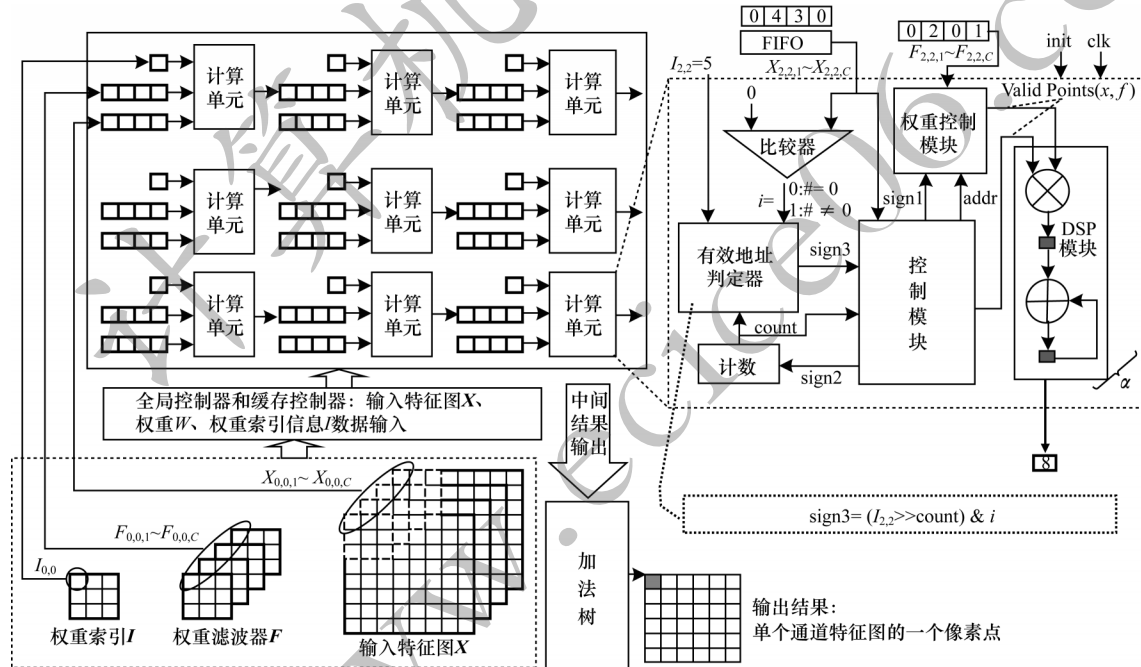


图6 处理单元的具体结构

Fig.6 Detailed structure of the processing unit

如上文所述,在本文设计的加速器中,每个PU单元实现 $C \times K \times K$ 个乘加运算。首先,每个 $K \times K$ 尺寸的输入特征与其对应权重的运算操作分到 $K \times K$ 个计算单元中;然后,每个计算单元做 $C$ 通道乘累加操作得到中间输出;最后,将所有中间输出通过加法树得到最终输出结果。

计算单元的设计是稀疏神经网络加速的核心,需要达到选取有效数据参与运算的目标。图6右半

部分为计算单元的结构图。对于特征图向量,由于通过比较器判断是否为有效值(非零值),因此输出值 $i$ 为0或者为1。本文设计未采用文献[13]为特征图数据建立二维掩码索引的方案,因为建立索引的工作除了必要的有效值判断,还要对索引数据进行存储、传输等操作。 $C$ 通道的权重Mask索引可以使用 $\text{lb } C$ 位宽的定点数 $I_{k,k}$ 来表示(右侧图中使用的索引为 $I_{2,2}$ )。每判断一个特征图点,会有局部计数器



来输出计数 count。有效地址判定器模块获取  $i$ 、 $I_{2,2}$  和 count 值并执行逻辑运算  $\text{sign3}=(I_{2,2}>>\text{count}) \& i$ 。该运算的功能是判断权重和特征图点是否同时为有效值。逻辑计算的结果反馈给控制单元,控制单元会提取有效的输入值给后端的乘累加单元。在 FPGA 器件中,乘累加单元采用片上 DSP 资源来实现。计算单元的结果会输出到加法树做最后的累加。

需要注意的是,做计算单元内部的并行,计算单元后端可以占用多个 DSP 模块,假设为  $\alpha$  个。因此, DSP 资源的占用量  $D_{\text{DSP}}$  可以根据以上讨论做预估,如式(2)所示:

$$D_{\text{DSP}}=(P_W \times P_H \times P_N) \times (K \times K) \times \alpha \quad (2)$$

其中:  $P_W \times P_H \times P_N$  是指 PU 阵列中 PU 的个数;  $K \times K$  是指 PU 内部计算单元的个数,该值是固定的且这些计算单元中的 DSP 除了计算输入通道方向最后几个数,在多数情况下不会出现“闲置”状态;  $\alpha$  值与模型数据的稀疏特征有关,该值的获取将在下文进行讨论。

## 2.4 片上缓存与 BRAM 存储块划分

片上缓存采用的是 FPGA 片上 BRAM 资源,主要考虑存储块划分和数据复用这 2 点。存储块划分借鉴了文献[19],核心思想是尽量将计算过程中需要并行获取的参数值放到不同的 BRAM Bank 中。表 2 是根据该思想对特征图矩阵和权重矩阵在各个维度上的划分情况,表中的数字是指该维度下需要并行获取的数据个数。例如输入特征图缓存的宽度方向,PU 每次需要同时获取  $(P_W+K-1)$  个数参与运算,那么该维度的数据可以平均分配到  $(P_W+K-1)$  个 BRAM 单元中。

表 2 BRAM 存储器划分情况

Table 2 Partition of BRAM memory

缓存	宽度方向	高度方向	输入深度	输出深度
输入特征图	$(P_W+K-1)$	$(P_H+K-1)$	$\alpha$	—
输出特征图	$P_W$	$P_H$	—	$P_N$
权重	$K$	$K$	$\alpha$	$P_N$

数据复用主要是为了减轻片上-片外传输的带宽压力,在满足存储块上述划分规则的情况下将尽量多的数据缓存在片上。以权重参数为例,片上 PU 阵列每批次运算需要  $K \times K \times P_N \times \alpha$  个 BRAM 单元,

当该批次运算计算完成后,依然希望这些数存储在片上,以便于在输入特征图窗口滑动到下个位置时再复用这些数据。因此,添加一个与  $K \times K \times P_N \times \alpha$  相乘的参数  $\chi_{\text{weight}}$  来表征复用情况。

综上所述,根据表 2 以及数据复用的考虑预估出片上 BRAM 资源的占用情况,如式(3)所示:

$$O_{\text{BRAM}}=\chi_{\text{in}} \times \left[ (P_W+K-1) \times (P_H+K-1) \times \alpha \right] + (\chi_{\text{in}} \times \chi_{\text{weight}}) \times \left[ (P_W) \times (P_H) \times P_N \right] + \chi_{\text{weight}} \times \left[ K \times K \times P_N \times \alpha \right] \quad (3)$$

需要注意的是,式(3)中输出特征图的复用参数  $\chi_{\text{out}}$  是根据  $\chi_{\text{in}}$  和  $\chi_{\text{weight}}$  来确定的。

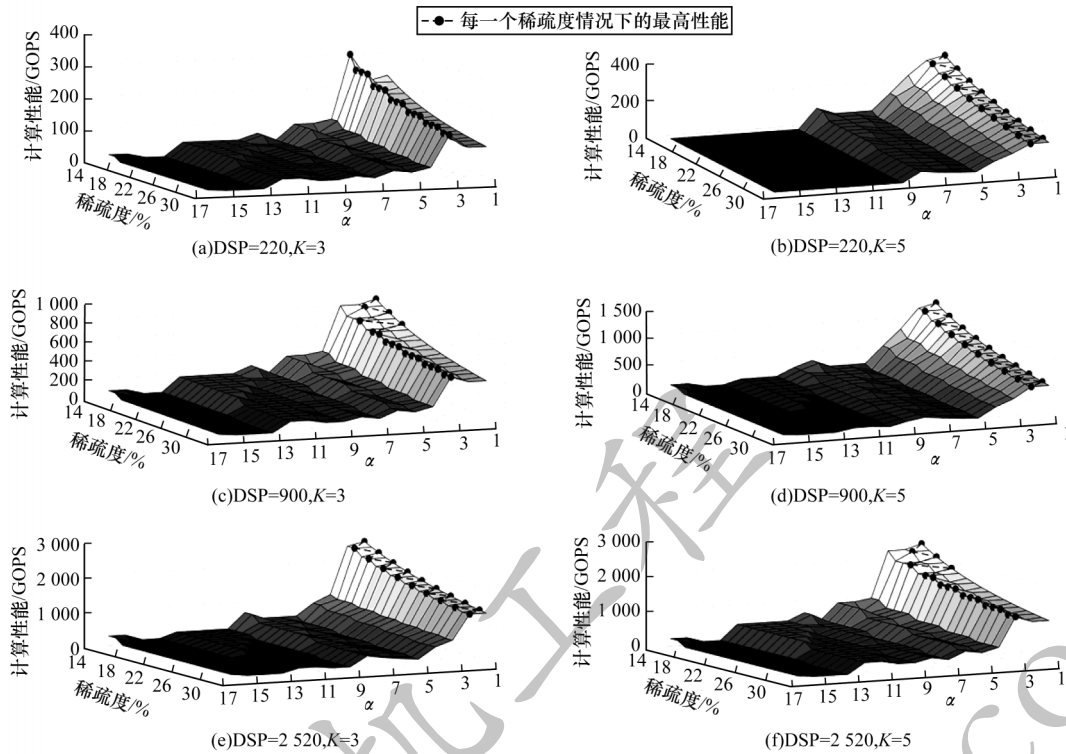
## 2.5 设计参数分析

本文加速器系统的设计参数有  $P_W$ 、 $P_H$ 、 $P_N$  和  $\alpha$ , 这些参数都与并行操作相关。由 2.2 节和 2.3 节可知,加速系统共有  $(P_W \times P_H \times P_N) \times (K \times K)$  个计算单元并行操作。 $\alpha$  的值则与算法模型参数的稀疏化特征有关:  $\alpha$  越大,并行操作的计算单元之间需要等待的时间会越短,但是 DSP 模块的闲置越严重;  $\alpha$  越小则相反。可以通过小规模实验实现不同  $\alpha$  参数下的处理模块来确定最佳的设计参数。假设某卷积层参数为  $\langle W=224, H=224, C=64, N=64 \rangle$ , 本文在具有不同数量 DSP 的 3 个典型的 Xilinx 器件上实现单个 PU, 可获取的计算性能(GOPS)的计算如式(4)所示:

$$P_{\text{Perform}} = \frac{2 \times (N \times C \times W \times H \times K^2)}{\left( \left\lceil \frac{W}{P_W} \right\rceil \times \left\lceil \frac{H}{P_H} \right\rceil \times \left\lceil \frac{N}{P_N} \right\rceil \times \left\lceil \frac{\lambda C}{\alpha} \right\rceil \times I_a + L_a \right) \times \left\lceil \frac{1}{f} \right\rceil} \quad (4)$$

其中:  $\lambda$  表示在  $C$  方向上权重和输入特征图的稀疏程度,实践中一般为 20%~50%;  $I$  和  $L$  分别表示对 PU 做流水线优化后的启动间隔(Initiation Interval, III)和启动延迟(Initiation Latency, IL),它们均通过实验获得;  $f$  表示频率,设置为 200 MHz。

初步实验结果如图 7 所示,可以看出,在不同的稀疏度情况下,PU 中的每个计算单元  $\alpha$  取 1 或者 3 可以达到最佳的性能。需要注意的是,对于不同的器件和不同算法模型,以及同一算法模型中的不同层,均可使用该方法预估最佳性能。

图7 不同 $\alpha$ 参数下处理单元的计算性能Fig.7 Computational performance of processing unit under different  $\alpha$  parameters

### 3 实验与结果分析

#### 3.1 实验设置

本文实验在GPU平台训练了VGG16<sup>[20]</sup>网络并使用文献[16]的方法进行剪枝和再训练,最终采用的神经网络卷积层参数稀疏情况如表3所示。在Xilinx硬软件环境下实现卷积层前向推理的加速,硬件平台为Xilinx Zynq XC7Z045,软件环境为Vivado HLx (v2018.3)。对于功耗,本文使用Xilinx的Xpower工具给出评估结果。根据前文中的分析,设计参数设置为: $P_w=2, P_H=2, P_N=8, \alpha=3, \chi_{in}=2, Z_{weight}=2$ 。

表3 VGG16卷积层稀疏情况

Table 3 Sparsity of convolutional layer of VGG16 %

网络层	稀疏度		有效运算占用率
	权重	输入特征图	
Conv1_2	50	100	50
Conv1_2	36	35	22
Conv2_1	31	32	19
Conv2_2	29	36	19
Conv3_1	31	37	18
Conv3_2	33	37	21
Conv3_3	32	36	20
Conv4_1	31	33	18
Conv4_2	30	33	17
Conv4_3	30	34	17
Conv5_1	29	35	18
Conv5_2	29	33	18
Conv5_3	30	32	17

#### 3.2 结果分析

表4统计了最终的实验结果中FPGA内部资源的使用情况。从中可以看出:LUT占用量达到了93%,这是因为PU为了处理稀疏的数据而增加了逻辑单元模块,这些逻辑单元模块主要都是由LUT来实现;同时DSP的数目达到了占比最多,这是因为硬件单元充分考虑了并行。

表4 实现VGG16 FPGA的内部资源使用情况

Table 4 Statistics of FPGA resource usage when implementing VGG16

模块	总使用量	资源可利用量	利用率/%
LUT	204 928	218 600	93.0
FF	95 776	437 200	21.9
BRAM_18K	924	1 090	84.7
DSP48E	864	900	96.0

计算性能和功耗比实验结果如表5所示,其中,本文设计的加速器对卷积层加速的总体性能为678.2 GOPS,性能功耗比为69.45 GOPS/W。此外,表5也展示了本文方案与其他方案的比较结果,从中可以得出以下结论:

1)与稠密神经网络的FPGA加速器的对比:与使用相同器件的文献[3]方案相比,本文计算性能和性能功耗比分别达到了它的2.9倍和2.8倍;与文

献[4,6]方案相比,本文DSP的使用量约为它们的60%,但却在计算性能指标上表现更优秀;与文献[5]方案相比,按每个DSP可以达到的GOPS,计算性能基本一致,但是本文方案的性能功耗比较文献[5]方案低。

2)与稀疏神经网络的FPGA加速器的对比:文献[7,14]均使用了ZCU102平台,本文DSP使用量分别是它们的75%和64%,计算性能指标上本文与它们相比分别有2.1和1.3倍的提升;在性能功耗比上则分别有5.3和2.3倍的提升。

表5 性能与功耗指标实验结果及对比

Table 5 Performance and energy efficiency experiment results and comparison

方案	加速网络类型	器件	工作频率/kHz	数据精度	DSP资源使用情况	性能/GOPS	功耗/W	性能功耗比/(GOPS·W <sup>-1</sup> )
文献[3]方案	稠密	Zynq XC7Z045	100	16 bit fixed	824(91.6%)	229.50	9.40	24.42
文献[4]方案	稠密	Virtex-7 VX690T	150	16 bit fixed	1 376(38%)	570.00	25.00	22.80
文献[5]方案	稠密	Cyclone V SCGX	100	16 bit fixed	342(100%)	317.86	9.71	32.73
文献[6]方案	稠密	Arria-10 GX 1150	150	16 bit fixed	1 518(100%)	645.25	—	—
文献[7]方案	稀疏	Zynq ZCU102	200	16 bit fixed	1 144(45%)	309.00	23.60	13.09
文献[14]方案	稀疏	Zynq ZCU102	200	16 bit fixed	1 350(53%)	495.40	15.40	29.83
本文方案	稀疏	Zynq XC7Z045	150	16 bit fixed	864(96%)	678.20	9.77	69.45

#### 4 结束语

本文提出一种针对神经网络模型的稀疏化卷积层FPGA硬件加速器。设计稀疏数据处理单元避免无效计算,同时通过提高多个处理单元的并行度和探索最佳设计参数提高计算性能。在Xilinx平台上的实验结果表明,本文设计能够有效提高稀疏化卷积层的实现性能和性能功耗比。下一步将对稀疏神经网络做软硬件协同优化:在软件方面,在剪枝过程就预先考虑硬件并行的需求,做面向硬件的平衡剪枝;在硬件方面,使用已有的并行、流水线等技术,将平衡剪枝后模型映射到FPGA上。

#### 参考文献

- [1] LECUN Y, BENGIO Y, HINTON G. Deep learning[J]. Nature, 2015, 521(7553): 436-444.
- [2] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet classification with deep convolutional neural networks[C]//Proceedings of the 25th International Conference on Neural Information Processing Systems. New York, USA: Curran Associates Inc., 2012.
- [3] XIAO Q C, LIANG Y, LU L Q, et al. Exploring heterogeneous algorithms for accelerating deep convolutional neural networks on FPGAs[C]//Proceedings of the 54th Annual Design Automation Conference. New York, USA: ACM Press, 2012: 1-6.
- [4] SHEN J Z, HUANG Y, WANG Z L, et al. Towards a uniform template-based architecture for accelerating 2D and 3D CNNs on FPGA[C]//Proceedings of 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. New York, USA: ACM Press, 2018: 97-106.
- [5] 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.

- [6] MA Y F, CAO Y, SARMA V, et al. Optimizing loop operation and dataflow in FPGA acceleration of deep convolutional neural networks[C]//Proceedings of 2018 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. San Diego, USA: [s. n.], 2018: 45-54.
- [7] LU L Q, XIE J M, HUANG R R, et al. An efficient hardware accelerator for sparse convolutional neural networks on FPGAs[C]//Proceedings of the 27th Annual International Symposium on Field-Programmable Custom Computing Machines. Washington D. C., USA: IEEE Press, 2019: 17-25.
- [8] ZHANG S J, DU Z D, ZHANG L, et al. Cambricon-X: an accelerator for sparse neural networks[C]//Proceedings of 2016 IEEE International Symposium on Microarchitecture. Washington D. C., USA: IEEE Press, 2016: 1-5.
- [9] AIMAR A, MOSTAFA H, CALABRESE E, et al. NullHop: a flexible convolutional neural network accelerator based on sparse representations of feature maps[J]. IEEE Transactions on Neural Networks and Learning Systems, 2018, 30(3): 644-656.
- [10] 刘勤让, 刘崇阳. 利用参数稀疏性的卷积神经网络计算优化及其FPGA加速器设计[J]. 电子与信息学报, 2018, 40(6): 1368-1374.
- [11] LIU Q R, LIU C Y. Calculation optimization for convolutional neural networks and FPGA-based accelerator design using the parameters sparsity[J]. Journal of Electronics and Information Technology, 2018, 40(6): 1368-1374. (in Chinese)
- [12] PARASHAR A, RHU M, MUKKARA A, et al. SCNN: an accelerator for compressed-sparse convolutional neural networks[J]. Computer Architecture News, 2017, 45(2): 27-40.
- [13] 周圣元, 杜子东, 陈云霁. 稀疏神经网络加速器设计[J]. 高技术通讯, 2019, 29(3): 24-33.
- [14] ZHOU S Y, DU Z D, CHEN Y J. Design of sparse neural network accelerator[J]. Chinese High Technology Letters, 2019, 29(3): 24-33. (in Chinese)

(下转第204页)

(上接第 195 页)

- [13] KIM D, AHN J, YOO S. ZeNA: zero-aware neural network accelerator[J]. IEEE Design & Test, 2017, 35(1): 39-46.
- [14] ZHU C Y, HUANG K J, YANG S Y, et al. An efficient hardware accelerator for structured sparse convolutional neural networks on FPGAs[EB/OL]. (2020-01-07)[2020-06-10]. <https://arxiv.org/pdf/2001.01955.pdf>.
- [15] ZHOU X D, DU Z D, GUO Q, et al. Cambricon-S: addressing irregularity in sparse neural networks through a cooperative software/hardware approach[C]//Proceedings of the 51st Annual IEEE/ACM International Symposium on Microarchitecture. Washington D. C., USA: IEEE Press, 2018: 1-5.
- [16] HAN S, MAO H Z, DALLY W J. Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding[C]//Proceedings of the 4th International Conference on Learning Representations. San Juan, Puerto Rico: [s. n.], 2016: 1-5.
- [17] HAN S, JEFF P, TRAN J, et al. Learning both weights and connections for efficient neural network[C]//Proceedings of NIPS' 15. Montreal, Canada: Neural Information Processing Systems Foundation, Inc., 2015: 1-5.
- [18] ANWAR S, HWANG K, SUNG W. Structured pruning of deep convolutional neural networks[J]. ACM Journal on Emerging Technologies in Computing Systems, 2017, 13(3): 1-18.
- [19] LAING Y, LU L Q, XIAO Q C, et al. Evaluating fast algorithms for convolutional neural networks on FPGAs[C]//Proceedings of the 25th Annual International Symposium on Field-Programmable Custom Computing Machines. Washington D. C., USA: IEEE Press, 2019: 1-5.
- [20] KAREN S, ANDREW Z. Very deep convolutional networks for large-scale image recognition[EB/OL]. (2015-04-10)[2020-06-10]. <https://arxiv.org/pdf/1409.1556.pdf>.

编辑 金胡考