高技术通讯 2023 年 第33 卷 第10 期:1060-1067

doi:10.3772/j.issn.1002-0470.2023.10.006

FPGA 实现卷积神经网络加速器①

张立国② 黄文汉③ 金 梅

(燕山大学电气工程学院 秦皇岛 066004)

摘 要 卷积神经网络传统的应用平台是中央处理器(CPU)和图形处理器(GPU),其体积和功耗不能适应轻量化的行业,轻量化的专用集成电路(ASIC)平台专用加速器的开发成本又不能适应愈发复杂和深层次的网络结构。针对上述问题,设计一种基于现场可编程门阵列(FPGA)的卷积神经网络(CNN)加速器,既满足轻量化应用场景,又有低开发成本的特性。设计浮点加法器和浮点乘法器组合成卷积运算的基本运算单元,完成 16 bits 浮点数乘累加操作只需要消耗一个数字信号处理器(DSP)资源;针对 FPGA 运算特性设计了基于 ReLU 函数的激活层模块;设计可调节并行度的各层模块,可根据平台资源在性能、功耗和面积上取得平衡;设计用比较器简化的 SoftMax 模块。实验结果表明,在 100 MHz 工作频率下,峰值算力可达 44.8 GFLOPS,功率仅为 4.51 W。

关键词 现场可编程门阵列(FPGA);卷积神经网络(CNN);硬件加速器;并行度

0 引 言

近年来,深度学习[1]被广泛应用于安全防护[2]、医疗康养[3-5]、自动驾驶[6-8]等领域。卷积神经网络(convolutional neural network, CNN)是深度学习中的一种重要架构。目前,实现卷积神经网络加速器的平台以中央处理器(central processing unit, CPU)、图形处理器(graphics processing unit, GPU)、专用集成电路(application specific integrated circuit, ASIC)和现场可编程门阵列(field programmable gate array, FPGA)为主[9]。其中,CPU 受限于内存带宽,计算 CNN 的大量数据时效率较低;GPU 拥有大量计算核心和高速内存带宽,主要用于加速图像数据的运算,在深度学习训练阶段的优势明显;ASIC[10-12]是可以实现特定算法的加速器,在计算效率、功耗和面积上表现最好;但是,ASIC 只能为既定的算法加速,昂贵的开发成本不能适应愈发复杂和深层次的网络结构。相比之下,FPGA 具有高并行度、低功耗、灵活编程和短开发周期等特点,这些综合优势使 FPGA 相较于其他 3 个平台更适合深度学习的前向推理过程。

本文设计基于 FPGA 的卷积神经网络加速器,神经网络结构为 LeNet-5[13],使用 Verilog 硬件描述语言,数据格式为 IEEE 754 标准的 16 bits 浮点类型。

近年来,已有许多对 FPGA 实现卷积神经网络的相关研究。

文献[14]使用异构计算架构成功实现了目标检测卷积神经网络的硬件加速。方案选择软核 CPU 实例化在 FPGA 上,并通过 CPU 控制数据调度,使 FPGA 能够实现卷积、ReLU 非线性激活、最大值池化等算法的硬件加速。通过按照特定方式排列输入数据,实现了将多维卷积运算转化成一维卷积运算,从而提高了数据并行度。尽管该方案只能为特定神经网络进行加速,但其成功地为硬件加速提

① 国家重点研发计划(2020YFB1711001)资助项目。
② 男,1978 年生,博士,副教授;研究方向:机器视觉,故障诊断,虚拟现实;E-mail:zlgtime@163.com。
③ 通信作者,E-mail:huangwenhan@126.com。
(收稿日期:2023-02-17)

供了一种可行的方案。值得注意的是,一旦神经网络结构发生改变,则需要重新搭建加速系统,无法应对复杂多变的神经网络。

文献[15]提出的卷积硬件计算单元设计具有可扩展性,可以在资源有限的情况下复用卷积计算模块,同时采用对应数据分割的形式提高并行度,从而显著提高了第 1 层卷积的计算速度。然而,该方案在卷积复用的同时会牺牲速度,因此在要求实时性的应用场景中可能缺乏优势。

文献[16]设计了一个 2D 卷积算法,该算法采用较少的寄存器、乘法器、加法器和控制模块,从而节省了大量的硬件开销。该算法使用块浮点(block-floating-point, BFP)算法进行数据处理,数据位宽选择 16 位和 8 位,相比传统的 32 位数据位宽,可以至少降低 50% 的资源开销。但是,这种算法采用较低精度的数据运算方式,因此不可避免地导致识别精度下降。

1 关键技术设计

1.1 浮点运算单元设计

在大多数对卷积神经网络加速器的硬件设计中,采用量化后的 16 位[17]甚至 8 位[18]定点数作为参数。这样的设计虽然可以减少数据存储以及计算量,但是不可避免地损失了精度。于是,本文设计浮点运算单元(processing element, PE),它由浮点加法器和浮点乘法器构成。

浮点运算单元相对于定点运算单元的优势有:

(1)可以表示非常大或非常小的数值,以及小数和分数。

(2)可以提供高精度的计算和表示,因为指数位和尾数位的可变性。

(3)通过硬件的优化,在消耗一定资源的同时,可以减少延时,且达到几乎等同于神经网络最优的精度效果。

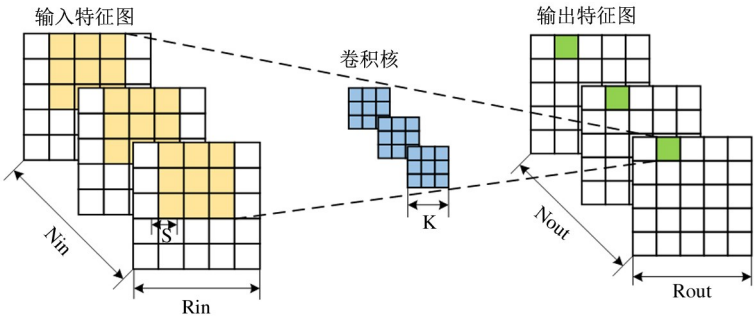
在浮点数加法器硬件电路设计中,为了减少资源开销,设计预移位(pre-shifting)功能:如果 2 个操作数指数不相等时,将较小的数值通过右移操作使指数相等。

在浮点乘法器的设计中,有效数字的乘法运算占运算时间的比重最大,因此,将指数计算与有效数字计算并行,消去了指数计算时间,提高了计算效率。

将浮点加法器串联在浮点乘法器后组成浮点运算单元,并且加入 3 个寄存器用以寄存输入和输出。

1.2 并行度优化设计

卷积神经网络中的三维图像卷积运算是卷积神经网络中的重点。图像的三维卷积运算可以按不同维度进行拆解,从而达到不同程度的并行度。图 1 所示为卷积层多层嵌套循环模型图。




```
for(ti=0;ti<Nin;ti++)
  for(to=0;to<Nout;to++)
    for(ro=0;ro<Rout;ro++)
      for(co=0;co<Cout;co++)
        for(x=0;x<K;x++)
          for(y=0;y<K;y++)
            ofmap[to][ro][co] += w[ti][to][x][y] * ifmap[ti][S*ro+x][S*co+y]
```

图 1 卷积层多层嵌套循环模型图

将图像卷积运算按照不同维度展开可以得到不同的并行度。

(1)卷积核内的并行

卷积核内的并行是指在卷积运算中,对于每一个卷积核内的元素,都可以独立地进行计算,从而加速卷积运算。具体来说,假设卷积核的大小为 $K \times K$,那么在卷积运算的过程中,可以同时计算卷积核内的所有元素,而不需要逐个进行计算。这种并行计算的方式可以有效地提高卷积运算的速度。

(2)特征图内的并行

特征图内的并行是指在卷积神经网络中,一个特征图与一个卷积核的卷积操作相当于将卷积核在特征图上滑动计算卷积,同时参与计算的滑动窗口的个数称为特征图内的并行度。

(3)输出通道间的并行

特征图与一个卷积核进行卷积运算后输出一张二维的特征图作为输出特征图的一个通道。同时参与卷积计算的卷积核个数即为输出通道间的并行度。

设计卷积核内全并行,特征图内的并行度根据例化卷积运算单元调节,输出通道间的并行度通过例化单通道卷积模块个数调节。因此,本文设计的加速器在卷积运算部分可以在各维度进行并行度调节,适应不同网络结构,可重构性强。

2　系统架构设计及具体实现

2.1　加速器系统总架构

本文设计的卷积神经网络硬件加速系统架构如图 2 所示,系统主要由寄存器传输级(register transfer level, RTL)描述的卷积神经网络加速器、直接内存访问模块与处理器组成。卷积神经网络加速器主要负责卷积层、激活函数层、池化层、全连接层和 SoftMax 层的计算。直接内存访问模块承担数据的传输。处理器的功能是通过配置总线为整个系统的任务进行调度以及加速器中寄存器的配置。存储器用于存储输入图像数据、特征图缓存数据和输出结果数据。

实验实现的卷积神经网络结构为 LeNet-5,该卷积神经网络结构的卷积核大小均为 5×5,步长为 1,池化方式采用平均池化,激活函数为 ReLU 函数。输入图像尺寸固定为 32×32。

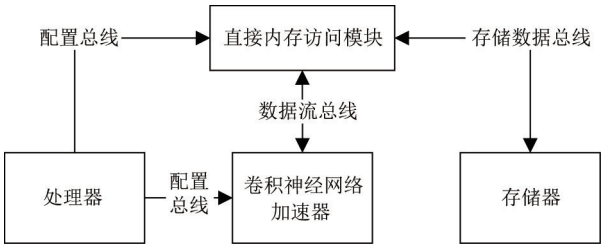


图 2　卷积神经网络加速系统架构

2.2　可调并行度的浮点基本运算模块设计

LeNet-5 网络共有 3 层卷积层,每层卷积层的卷积核种类分别为 6、16、120。

图 3 是浮点加法器结构图,运算数据为 IEEE 754 标准格式的 16 bits 浮点数,2 个输入记为操作数 1 和操作数 2,输出为记为和。按照 IEEE 754 标准格式将 2 个输入操作数拆包为符号位、指数位和尾数位。在操作数 1 或操作数 2 的特殊情况下,和分别为操作数 2 或操作数 1。一般情况下,首先比较 2 个操作数的指数部分,以指数大的数为标准,另一个输入操作数通过右移指数差值个位数使指数相等。其次判断两输入的符号位,若符号位相同,则小

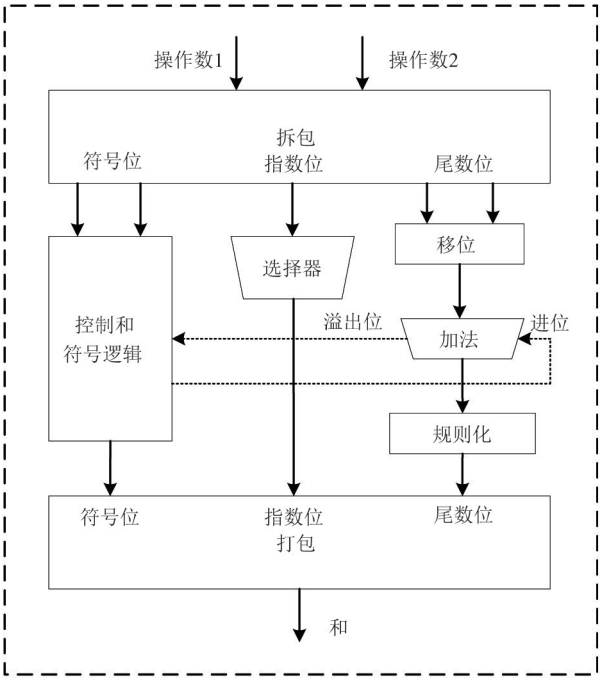


图 3　浮点加法器结构图

数部分直接相加;若符号位不同,则用正数减负数。最后结果打包为 IEEE 754 标准格式输出。

图 4 是浮点乘法器结构图,2 个输入操作数分别记为操作数 1 和操作数 2,输出记为乘积。特殊情况下,其中一个输入为 0 时,乘积为 0。一般情况下,首先用异或判断符号位是否相同,若相同则乘积符号为正;若不同则符号位为负。根据 IEEE 754 标准,乘积的指数为操作数 1 的指数与操作数 2 的指数的和,小数部分直接用一个乘法器相乘,乘积规则化的同时调节指数位。

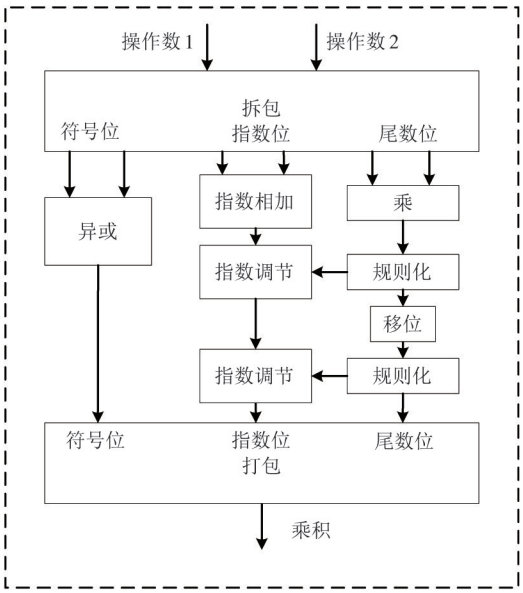


图 4　浮点乘法器结构图

本文设计的浮点乘法器与浮点加法器组合成浮点基本运算单元,如图 5 所示。浮点乘法器输入 2 个 16 bits 的数据,浮点加法器把浮点乘法器的结果与上一次累加的结果相加。

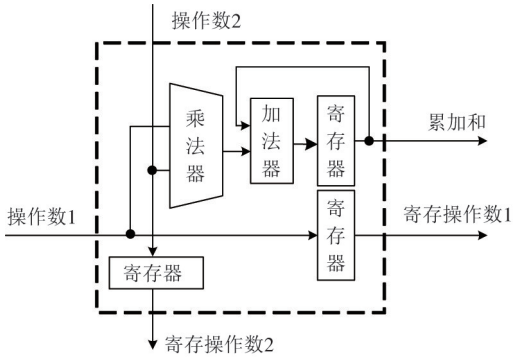


图 5　浮点基本运算单元

设计的卷积运算单元如图 6 所示,其中例化浮点基本运算单元,每个浮点基本运算单元运算需要一个周期,根据卷积核的尺寸可以求出卷积运算单元需要的周期数。

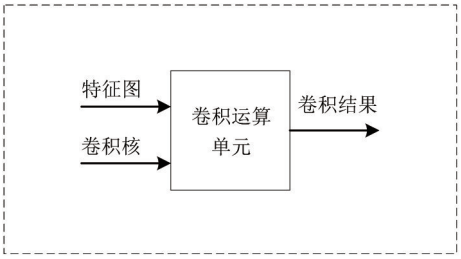


图 6　卷积运算单元

设计的单通道卷积模块并行结构如图 7 所示,其中例化卷积运算单元模块,通过改变例化卷积运算单元模块的数量,可以控制单个卷积核对一个特征图进行卷积运算的并行度。通过滑窗选择模块将输入特征图分割成与卷积核尺寸相同的多个滑窗块,用于同时和卷积核计算。

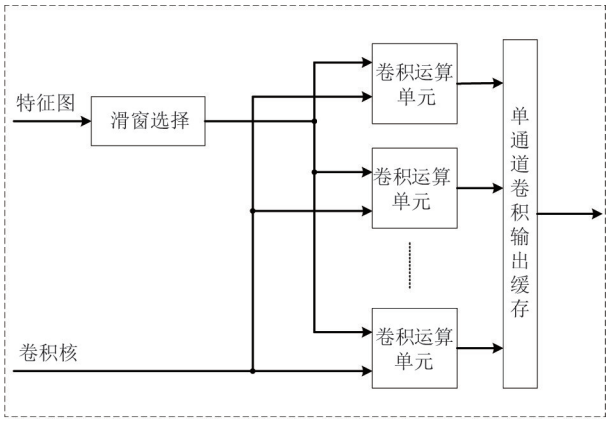


图 7　单通道卷积模块并行结构

本文设计多通道模块并行结构如图 8 所示,通过增加例化单通道卷积模块的数量,可以让特征图和不同卷积核同时进行卷积运算,提高了输出通道的并行度。卷积核选择模块从卷积核堆中选择需要运算的卷积核。

2.3　激活层模块

本设计采用了 ReLU 函数作为激活函数。

ReLU 函数公式如式(1)所示。

$$f(x) = \max(0, x) \tag{1}$$

ReLU 函数图像如图 9 所示。

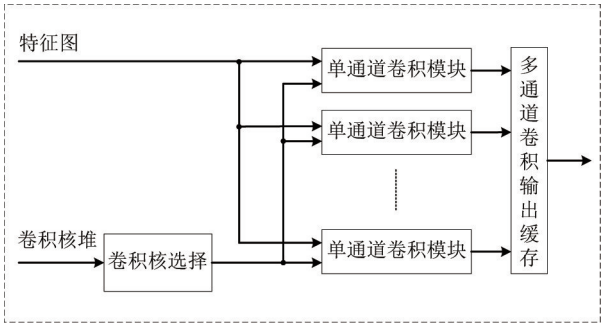


图 8　多通道卷积模块并行结构

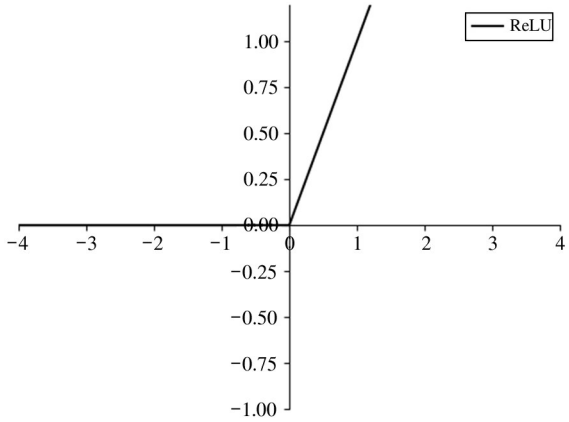


图 9　ReLU 函数图像

相较于 Sigmoid 函数和 tanh 函数，ReLU 函数具有 3 个优势。首先，它更贴近于生物学神经元的工作方式。其次，梯度下降和方向传播的效率更高，这有助于避免梯度爆炸和梯度消失等问题。最后，ReLU 函数的计算较为简单，不需要像其他复杂的激活函数一样进行指数和幂函数运算，降低了资源开销。根据 ReLU 函数，数据与 0 比较即可得到输出值。

2.4　可调并行度的平均池化模块设计

设计平均池化单元结构如图 10 所示，该单元由 3 个浮点加法器和 1 个移位寄存器组成。当执行采样区域尺寸为 2 × 2 的池化运算时，如图所示，将数据 1 和数据 2 相加、数据 3 和数据 4 相加，再将 2 个和相加，加法器 3 的输出结果进入乘法器与常数0.25 相乘得到一个池化采样区的结果。

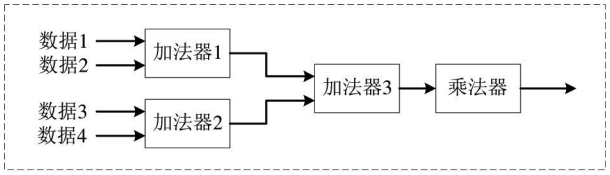


图 10　平均池化单元

设计单通道平均池化模块并行结构如图 11 所示。通过改变实例化平均池化单元的数量，可以调整池化采样的并行度。在资源充足的情况下，可以尽可能地提高并行度，以提高运算效率。以 LeNet-5 网络为例，当处理一张尺寸为 $N \times N$ 的特征图时，最大并行度可以达到 $(N/2) \times (N/2)$。

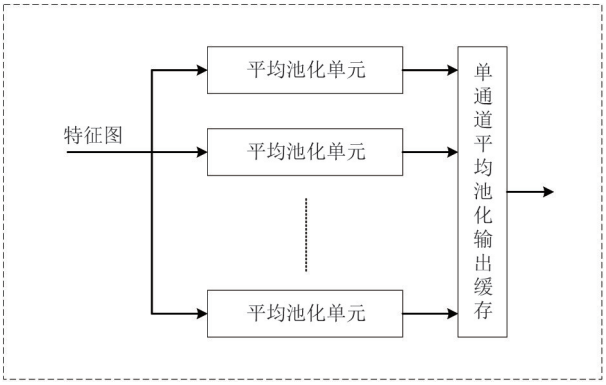


图 11　单通道池化模块并行结构

设计多通道平均池化模块结构如图 12 所示，通过改变例化单通道平均池化模块的数量而改变同时进行池化特征图的层数。最大并行度即为特征图深度。

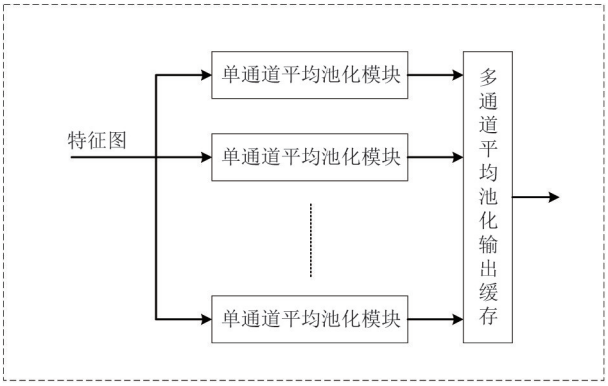


图 12　多通道平均池化模块并行结构

2.5　优化 SoftMax 模块设计以适应 FPGA 运算

LeNet-5 神经网络的全连接层后还有一层 SoftMax 函数，其作用是将求得的分类结果归一化，得到每个分类的概率。而实际应用中，只需要知道概率最大的一个或者几个分类，并不需要关心其概率大小，所以为了减少资源开销和提高运算速度，本设计采用比较器选择出最大的一个分类来代替 SoftMax 模块。

3 实验结果分析

本实验在 Vivado 工具上进行仿真和综合,电路最高工作频率为 100 MHz,在 Xilinx 的 Zynq UltraScale + MPSoC ZCU104 开发板上进行验证。

Xilinx FPGA 开发板资源名称说明:查找表(look-up table,LUT);触发器(flip-flop,FF);数字信号处理器(digital signal processing,DSP)。

以下对各模块的资源分配情况进行统计及分析。

3.1 卷积单元模块

资源使用情况如表 1 所示,该模块使用 205 个 LUT,82 个 FF,1 个 DSP。

由此可以看出,对于 1 次卷积运算(1 块卷积区域和 1 个卷积核的卷积过程),只需要使用 1 个 DSP 就可以实现。

表 1 C 卷积单元模块资源统计表

资源	占用量	资源总量	占用百分比/%
LUT	205	230 400	0.09
FF	82	460 800	0.02
DSP	1	1728	0.06

3.2 单通道卷积模块

资源使用情况如表 2 所列,该模块使用 6137 个 LUT,4296 个 FF,28 个 DSP。

该模块的功能是将输入特征图的一个通道按卷积核尺寸分割成一定数量的待卷积运算的块,其数量可以自由调节。当资源足够,可以增加例化卷积单元的数量提高运算并行度。本实验设置例化卷积单元的数量为 28,即将输入特征图分割成 28 个待卷积运算的块,同时进行 28 个卷积运算。

表 2 单通道卷积模块资源统计表

资源	占用量	资源总量	占用百分比/%
LUT	6137	230 400	2.66
FF	4296	460 800	0.93
DSP	28	1728	1.62

3.3 多通道卷积模块

资源使用情况见表 3,该模块使用 98 192 个 LUT,68 736 个 FF,448 个 DSP。

该模块的功能通过改变例化单通道卷积模块的个数来改变同时输出特征图的通道数。卷积核选择模块从卷积核堆中选择即将进行卷积的卷积核。在资源允许的情况下,该模块可以通过例化更多的单通道卷积模块,实现多个卷积核之间的运算并行度。

表 3 多通道卷积模块资源统计表

资源	占用量	资源总量	占用百分比/%
LUT	98 192	230 400	42.62
FF	68 736	460 800	14.92
DSP	448	1 728	13.19

3.4 结果分析

在 100 MHz 工作频率下,计算输入特征图尺寸为 14 × 14 × 6,与 16 个尺寸为 5 × 5 的卷积核进行卷积运算,例化的卷积单元模块个数为 28,单通道卷积模块个数为 16,此时加速器峰值算力为 44.8 GFLOPS,功率为 4.51 W。

本文的硬件加速器性能与 CPU、GPU 对比如表 4所示,速度显著高于 CPU,功耗低于 GPU 2 个量级。

表 4 CPU、GPU 与硬件加速器性能对比表

平台	CPU (i5 2500K)	GPU (GTX 960)	本文研究 (ARM + FPGA)
频率	3.3 GHz	1.127 GHz	100 MHz
时间/ms	42.6	14.1	21.1
功耗/W	45	235	4.51

本文的加速器性能与其他研究的硬件加速器对比如表 5 所示,文献[19]使用 16 bits 定点数,精度要低于本文的设计,对比文献[20]的设计,本文的能效比要高2.15倍。

4 结 论

根据本文实验部分的结果进行分析可以得出,

表 5 与其他硬件加速器研究比较表

性能	文献[19]	文献[20]	本文
平台	Zynq ZC706	Virtex7 VX485T	ZCU104
量化方式	16 bits fixed	32 bits float	16 bits float
频率/MHz	150	100	100
性能	137.0 GOPS	61.6 GFLOPS	44.8 GFLOPS
功耗/W	9.63	18.61	4.51
能效比/(GOPS/W)	14.22	3.31	7.13

本文设计的卷积运算单元资源开销极少,核心运算单元 DSP 只需要 1 个。并且可以通过调节单通道卷积模块和多通道卷积模块中例化子模块的数量来调节并行度,在不同资源的 FPGA 上平衡面积、功耗和速度。

未来的研究方向如下:卷积核内的并行度对卷积运算的效率提升有重要的影响,可以考虑脉动阵列等方式使计算效率进一步提升;在数据传输上,可以考虑增加 Ping-Pong 以减少数据存取带来的延时。

参考文献

[1] LECUN Y, BENGIO Y, HINTON G. Deep learning[J]. Nature, 2015,521(7553):436-444.

[2] 赵静. 基于卷积神经网络的建筑工程施工安全预警研究[D]. 北京:北京建筑大学管理科学与工程学院, 2020.

[3] 薛迪秀. 基于卷积神经网络的医学图像癌变识别研究[D]. 合肥:中国科学技术大学,2017.

[4] 李超. 基于卷积神经网络的人体行为分析与步态识别研究[D]. 杭州:浙江大学,2019.

[5] 左艳, 黄钢, 聂生东. 深度学习在医学影像智能处理中的应用与挑战[J]. 中国图像图形学报,2021,26(2):305-315.

[6] 韩昕辉. 基于深度学习的无人驾驶场景识别[D]. 中山:中山大学软件工程学院,2017:14-27.

[7] 张新钰, 高洪波, 赵建辉,等. 基于深度学习的自动驾驶技术综述[J]. 清华大学学报,2018,58(4):438-444.

[8] 赵锟. 基于深度卷积神经网络的智能车辆目标检测方法研究[D]. 长沙:国防科学技术大学,2015.

[9] 刘腾达, 朱君文, 张一闻. FPGA 加速深度学习综述[J]. 计算机科学与探索,2021,15(11):1-14.

[10] VAINBRAND D, GINOSAR R. Network-on-chip architectures for neural networks[C]//2010 4th ACM/IEEE International Symposium on Networks-on-Chip. Grenoble: IEEE, 2010:135-144.

[11] REAGEN B, WHATMOUGH P, ADOLF R, et al. Minerva: enabling low-power, highly-accurate deep neural network accelerators[J]. ACM SIGARCH Computer Architecture News, 2016,44(3):267-278.

[12] PRICE M, GLASS J, CHANDRAKASAN A P. A scalable speech recognizer with deep-neural-network acoustic models and voice-activated power gating[C]//2017 IEEE International Solid-State Circuits Conference. Shanghai: IEEE, 2017:244-245.

[13] LECUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998,86(11):2278-2324.

[14] 夏春秋, 陈世淼. 基于 FPGA 的卷积神经网络加速器研究与设计[J]. 电子技术与软件工程, 2022,238(20):170-177.

[15] 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, 2022,44(11):147-154.

[16] LIAN X, LIU Z, SONG Z, et al. High-performance FPGA-based CNN accelerator with block-floating-point arithmetic[J]. IEEE Transactions on Very Large Scale Integration Systems, 2019,27(8):1874-1885.

[17] LI H, FAN X, JIAO L, et al. A high performance FPGA-based accelerator for large-scale convolutional neural networks[C]//2016 26th International Conference on Field Programmable Logic and Applications. Lausanne: IEEE, 2016:1-9.

[18] GUO K, SUI L, QIU J, et al. Angel-eye: a complete design flow for mapping CNN onto embedded FPGA[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2017,37(1):35-47.

[19] QIU J, WANG J, YAO S, et al. Going deeper with embedded FPGA platform for convolutional neural network[C]//Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey: Association for Computing Machinery, 2016:26-35.

[20] ZHANG C, LI P, SUN G, et al. Optimizing FPGA-based accelerator design for deep convolutional neural networks

[C]//Proceedings of the 2015 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey: Association for Computing Machinery, 2015: 161-170.

Implementation of a convolutional neural network on an FPGA

ZHANG Liguo, HUANG Wenhan, JIN Mei

(School of Electrical Engineering, Yanshan University, Qinhuangdao 066004)

Abstract

The traditional application platforms for convolutional neural networks are central processing unit (CPU) and graphics processing unit (GPU), whose size and power consumption cannot be adapted to lightweight industries, and the development cost of lightweight application specific integrated circuit (ASIC) cannot be adapted to increasingly complex and deep network structures. To address the above problems, an convolutional neural network (CNN) hardware accelerator based on field programmable gate array (FPGA) is designed to satisfy both lightweight application scenes and low development cost. Design the floating-point adder and floating-point multiplier to combine into the basic operation unit of convolutional operation, and complete the 16 bits floating-point multiply-accumulate operation only need to consume one digital signal processing (DSP) resource. An activation layer module based on ReLU function is designed for the computing characteristics of FPGA. Designing modules at each layer with adjustable parallelism allows for a balance between performance, power consumption, and area, depending on platform resources. Design of SoftMax modules simplified with comparators. Experimental results show that the peak arithmetic can reach 44.8 GFLOPS at 100 MHz operating frequency with only 4.51 W power.

Key words: field programmable gate array (FPGA), convolutional neural network (CNN), hardware accelerator, parallelism