文章编号：1007-2780(2023)11-1521-10

基于FPGA的Winograd算法卷积神经网络加速器设计与实现

牛朝旭[1,2]，孙海江[1,2*]
（1. 中国科学院 长春光学精密机械与物理研究所，吉林 长春 130033；
2. 中国科学院大学，北京 100049）

摘要：为实现卷积神经网络在低功耗、边缘计算等场景中的加速计算，设计了一种基于现场可编程门阵列（FPGA）的Winograd算法卷积神经网络加速器。首先，将图像数据和权重数据量化为8位定点数，并设计了硬件卷积计算过程中的量化流程，提升了数据传输速度和计算速度。接着，设计了输入数据缓存复用模块，将多输入通道数据融合后传输，复用了行重叠数据。然后设计了Winograd流水线卷积模块，实现列数据的组合复用，从而最大化重用了片上数据，降低了片上数据存储的占用和带宽压力。最后将加速器在Xilinx的ZCU104开发板上部署。经过实验验证，加速器的卷积层计算性能达到354.5 GOPS，片上DSP计算效率达到0.69，与相关研究相比，实现了1.6倍以上的提升。该加速器能够以高能效比完成基于VGG-16网络的遥感图像分类任务。

关　键　词：卷积神经网络；现场可编程门阵列；Winograd算法；流水线；并行计算
中图分类号：TP332　　**文献标识码：**A　　**doi：**10.37188/CJLCD.2023-0013

Design and implementation of convolution neural network accelerator for Winograd algorithm based on FPGA

NIU Zhao-xu[1,2], SUN Hai-jiang[1,2*]

(1. *Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun* 130033, *China*;
2. *University of Chinese Academy of Sciences, Beijing* 100049, *China*)

Abstract: In order to realize the acceleration of convolutional neural network in low-power, edge computing and other scenarios, a Winograd algorithm convolutional neural network accelerator based on field programmable gate array (FPGA) is designed. Firstly, the image data and weight data are quantized into 8-bit fixed-point numbers, and the quantization process in the hardware convolution calculation process is designed to improve data transmission speed and calculation speed. Secondly, the input data buffer multiplexing module is designed, which fuses the data of multiple input channels and transmits them, reusing the row overlapping data. Then, the Winograd pipeline convolution module is designed to realize the combined reuse of

收稿日期：2023-01-13；**修订日期：**2023-02-04.
基金项目：吉林省科技发展计划（No. 20200404155YY）
Supported by Jilin Province Science and Technology Development Plan (No. 20200404155YY)
*通信联系人，E-mail：sunhaijing@126.com

column data, so as to maximize the reuse of data on chip and reduce the occupation of data storage on chip and bandwidth pressure. Finally, the accelerator is deployed on the ZCU104 development board of Xilinx. Experimental verification shows that the convolution layer computing performance of accelerator reaches to 354.5 GOPS, and the on-chip DSP computing efficiency reaches to 0.69, which is more than 1.6 times higher than relevant research. The accelerator can complete remote sensing image classification task based on VGG-16 network with high energy efficiency ratio.

Key words: convolution neural network; field programmable gate array; winograd algorithm; assembly line; parallel computing

1 引 言

卷积神经网络被广泛应用于许多深度学习系统中,并在全息图像重建[1]、光学计量[2]和自动驾驶[3]等多种计算机视觉任务中取得了显著的成效。为了达到更高的精度,一些研究引入了多尺度特征增强[4]、弱光照图像增强[5]以及RGB-D特征融合[6]等更为复杂的算法,使得网络计算复杂度和模型规模也更为庞大,进而导致了计算功耗的提升[7]。在一些低功耗的应用场景中,如星上AI计算、遥感图像在轨处理等,常规的硬件平台部署卷积神经网络十分困难[8]:通用的中央处理器(Central Processing Unit,CPU)无法满足卷积神经网络的计算需求;图形处理器(Graphic Processing Units,GPU)功耗太高,无法应用于嵌入式环境;专用集成电路(Application Specific Integrated Circuit,ASIC)成本高昂且通用性差。与之相比,计算并行度高、功耗低、可重复编程的现场可编程门阵列(Field Programmable Gate Array,FPGA)更适合应用于星上智能计算、在轨目标识别等低功耗环境下的卷积神经网络的硬件加速中。

传统的空间卷积算法通过循环展开、并行计算的方法进行加速计算,在早期被人们广泛使用,如脉动阵列[9]、层集群并行映射方法[10]等加速方法被先后提出。随着卷积神经网络的层次变深、卷积核尺寸变小,传统的卷积方法在卷积效率上已经逐渐落后,快速卷积算法显现出它的优势。文献[11]将快速傅里叶变换FFT算法应用于卷积中以加速计算,但此方法只对大尺寸卷积核具有良好的加速效果,否则其转换过程会引入大量补零操作,得不偿失。文献[12]提出在卷积计算中使用Winograd算法降低计算复杂度。Winograd算法是通过对输入特征矩阵和权重矩阵做线性变换后再求哈达玛积,减少了乘法次数,实现了硬件上计算效率的提升。文献[13]提出在FPGA上使用行缓存结构提高Winograd算法切片之间的行数据重用。文献[14]提出双缓冲区5×5流水线卷积方法。但是上述文献并没有充分复用重叠数据,也没有在目前使用广泛的小卷积核网络上充分发挥FPGA低功耗的特性。

本文设计了一种Winograd算法卷积神经网络加速器。首先设计了输入数据缓存复用模块,结合行缓存和列缓存重叠数据,最大化重用了片上数据,减少了频繁数据搬运的开销。针对FPGA并行运算特性,设计多通道并行Winograd卷积运算阵列,并将卷积过程分解为六级流水线,提高了运算效率和吞吐量。为了提升计算速度和数据传输速度,使用权重8位定点数(INT8)量化的方式来压缩模型,数据量减少到1/4。最后针对遥感图像分类数据集修改VGG16网络,将加速器部署至ZCU104平台进行实验验证。实验结果表明,本设计相比其他FPGA设计方案在功耗和计算效率上都有一定的提升。

2 Winograd算法

文献[15]提出的Winograd算法可以用于减少有限脉冲响应(Finite Impulse Response, FIR)滤波器的乘法次数,之后被应用于卷积神经网络加速中以减少乘法数量的方式来提升计算速度。在一维Winograd卷积计算中,设卷积核尺寸为r,卷积结果输出长度为m,则一维Winograd卷积计算公式$F(m,r)$需要的乘法数量为$m+r-1$,而传统卷积为$m\times r$。可以看出,当r和m都大于1时,Winograd卷积乘法数量更少。

以$F(2,3)$为例，用$\boldsymbol{d}=[\,d_0\quad d_1\quad d_2\quad d_3\,]^{\mathrm{T}}$表示输入向量，$g=[\,g_0\quad g_1\quad g_2\,]^{\mathrm{T}}$表示卷积核，$\boldsymbol{r}=[\,r_0\quad r_1\,]^{\mathrm{T}}$表示输出向量，其计算过程可以表示为：

$$\begin{bmatrix} d_0 & d_1 & d_2 \\ d_1 & d_2 & d_3 \end{bmatrix}\begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix}=\begin{bmatrix} r_0 \\ r_1 \end{bmatrix}. \tag{1}$$

因此可知，普通卷积需要6次乘法和4次加法，即：

$$r_0=(d_0g_0)+(d_1g_1)+(d_2g_2), \tag{2}$$

$$r_1=(d_1g_0)+(d_2g_1)+(d_3g_2). \tag{3}$$

而$F(2,3)$的Winograd卷积可写成如下矩阵乘法形式：

$$F(2,3)=\begin{bmatrix} d_0 & d_1 & d_2 \\ d_1 & d_2 & d_3 \end{bmatrix}\begin{bmatrix} g_0 \\ g_1 \\ g_2 \end{bmatrix}=\begin{bmatrix} m_0+m_1+m_2 \\ m_0-m_1-m_2 \end{bmatrix}, \tag{4}$$

其中m_0、m_1、m_2、m_3计算如下：

$$\begin{cases} m_0=(d_0-d_2)g_0 \\ m_1=(d_1+d_2)\dfrac{g_0+g_1+g_2}{2} \\ m_2=(d_2-d_1)\dfrac{g_0-g_1+g_2}{2} \\ m_3=(d_1-d_3)g_2 \end{cases}, \tag{5}$$

其中，输入信号$\boldsymbol{d}$的变换需要4次加法。而对于卷积神经网络推理阶段的卷积核g，其数值是固定的，其变换可以预先计算好进行存储，同时其中的除2操作可以用位移代替，所以计算需求可以忽略。输出项$\boldsymbol{r}$还需要中间项m进行4次乘法和4次加法，所以$F(2,3)$的Winograd乘法数量为4次、加法数量为8次。相较于传统卷积，以加法为代价，节省了33%的乘法数，考虑到在硬件实现的乘法的实现成本远高于加法。因此使用Winograd算法能够提升运算速度。

上述计算可以整理为如下的矩阵形式：

$$\mathrm{Out}=\boldsymbol{A}^{\mathrm{T}}\left[(\boldsymbol{GW})\odot(\boldsymbol{B}^{\mathrm{T}}\mathrm{In})\right]. \tag{6}$$

一维Winograd卷积推广到二维Winograd卷积，可得到如下矩阵形式：

$$\mathrm{Out}=\boldsymbol{A}^{\mathrm{T}}\left[(\boldsymbol{GWG}^{\mathrm{T}})\odot(\boldsymbol{B}^{\mathrm{T}}\mathrm{In}\boldsymbol{B})\right]\boldsymbol{A}, \tag{7}$$

其中$\boldsymbol{W}$和In是输入数据，Out是输出结果。$\boldsymbol{A}^{\mathrm{T}}$、$\boldsymbol{G}$、$\boldsymbol{B}^{\mathrm{T}}$都是常数矩阵，在$F(2^2,3^2)$时为：

$$\boldsymbol{A}^{\mathrm{T}}=\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & -1 & -1 \end{bmatrix}, \tag{8}$$

$$\boldsymbol{G}=\begin{bmatrix} 1 & 0 & 0 \\ \dfrac{1}{2} & \dfrac{1}{2} & \dfrac{1}{2} \\ \dfrac{1}{2} & -\dfrac{1}{2} & \dfrac{1}{2} \\ 0 & 0 & 1 \end{bmatrix}, \tag{9}$$

$$\boldsymbol{B}^{\mathrm{T}}=\begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 1 & 0 & -1 \end{bmatrix}. \tag{10}$$

Winograd卷积计算$F(2^2,3^2)$的过程如图1所示。一次$F(2^2,3^2)$卷积计算可以将乘法数从36次降低为16次，计算效率提升了2.25倍。

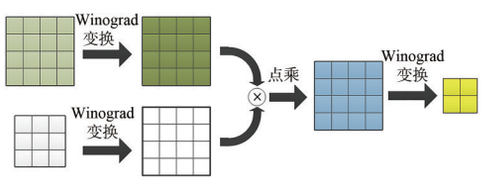

图1　Winograd卷积过程示意图

Fig. 1　Schematic diagram of Winograd convolution process

3　卷积神经网络加速器的构成

卷积神经网络加速器的整体架构如图2所示。因为基于ZYNQ平台，所以分为可编程逻辑(Programmable Logic，PL)端和处理系统(Processing System，PS)端两部分。PL端负责计算卷积神经网络的卷积、池化、全连接等相关层，PS端负责控制PL端的运行以及传输数据。两部分主要通过AXI-DMA进行数据交互，并使用AXI-FIFO起到对数据流缓冲的作用，防止数据接收不及时造成的数据丢失。

PS端读取SD卡中的权重数据和特征数据到DDR内存，通过写寄存器控制当前PL端运行网络的层数。之后开始计算，通过AXI-DMA使用AXI-Stream总线向PL端写入DDR上的数据，缓存至特征数据缓存模块或权重数据缓存模块。在卷积层中，将卷积计算按照并行计算方式循环展开，之后根据计算窗口位置的不同，输出数据到六级流水线卷积层，得到中间计算结果后累加缓存，完成整幅输出通道的计算后经AXI-DMA输出数据到DDR上，作为下一层特征数据存储。依此类推，直至完成网络全部层的计算，得到输出结果。

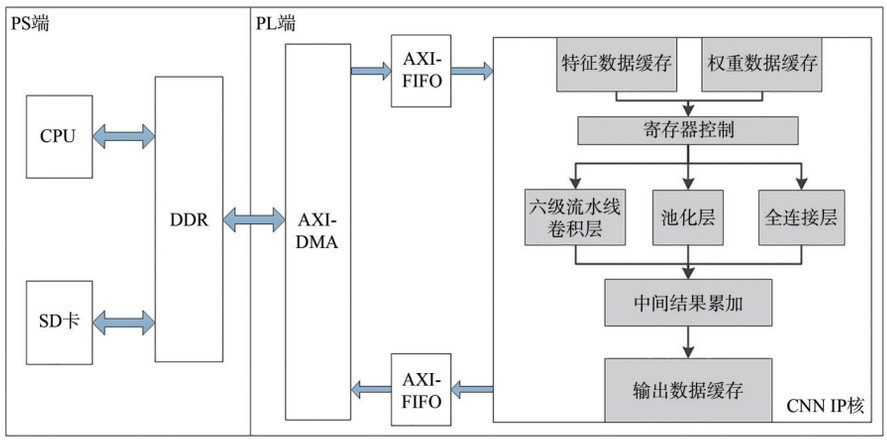

图 2 卷积神经网络加速器硬件架构图

Fig. 2 Hardware architecture diagram of convolutional neural network accelerator

4 卷积神经网络加速器设计

4.1 硬件 8 位定点数量化

32 位浮点数在 FPGA 上进行乘加运算时会消耗更多的片上资源，占用更多的位宽。为了提高数据吞吐速率和加速器计算效率，可以使用量化后的 8 位定点数，在精度损失不大的情况下达到压缩模型的目的。为了更适合 FPGA 硬件电路实现，本文选择了线性对称且逐层的量化方式。

量化的过程就是放缩的过程。将 32 位浮点数的数值范围线性缩小到 8 位定点数的数值范围，就得到了 INT8 的权重数据。其量化公式如式(11)所示：

$$Q(x)=\mathrm{round}\left(\frac{x}{\mathrm{scale}}+\mathrm{zero\_point}\right), \quad (11)$$

其中：x 表示 32 位浮点数据，scale 表示缩放比例，zero_point 表示映射零点偏移，round 表示四舍五入的取整操作。因为采用对称式量化，所以 zero_point 为 0。通过 KL 散度校准训练可以得到最优缩放比例，提升量化后精度。VGG16 网络逐层量化后的部分结果如表 1 所示，可以看到量化后结果全部为整数，且缩放比例均为 2 的整数幂，可以通过移位实现缩放，易于在硬件上实现 8 位定点数量化。

表 1 VGG-16 网络量化后的部分结果

Tab. 1 Partial results after quantification of VGG-16 network

层次	缩放比例	原权重	量化后权重
Conv1-1～Conv3-1	0.003 906 25	−0.553 730 607 032 775 9	−34
Conv3-2～Conv4-3	0.001 953 125	−0.030 606 031 417 846 68	−9
Conv5-1～Conv5-2	0.000 976 562 5	−0.029 665 347 188 711 166	−30
Conv5-3	0.001 953 125	0.027 768 215 164 542 198	14
Fc1	0.000 244 140 625	−0.001 109 445 700 421 929 4	−5
Fc2	0.000 488 281 25	−0.011 262 043 379 247 189	−23
Fc3	0.003 906 25	0.000 411 447 166 698 053 5	0

硬件量化流程如图 3 所示。将 32 位浮点数权重经过训练后量化，可以得到 8 位定点权重，与 8 位定点特征数据卷积后，扩展位宽到 16 位。经过多通道累加，输出数据位宽扩展到 32 位。之后累加上偏置，利用移位量化得到下一层计算所需的 8 位定点特征数据。为减小误差，移位量化时对需要舍去的小数位采用向偶数进位的模式。

4.2 输入数据缓存复用模块

FPGA 的存储资源可以分为片上存储和片外存储两种。片上存储主要是 Block RAM，它存

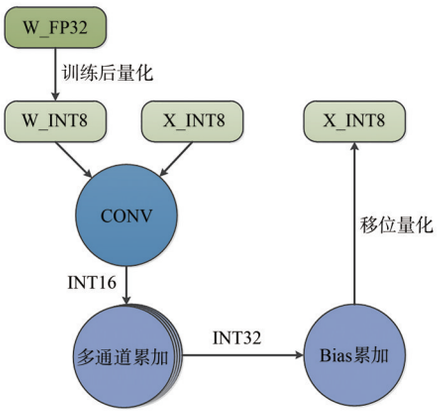

图3　硬件量化流程图

Fig. 3　Diagram of hardware quantification flow

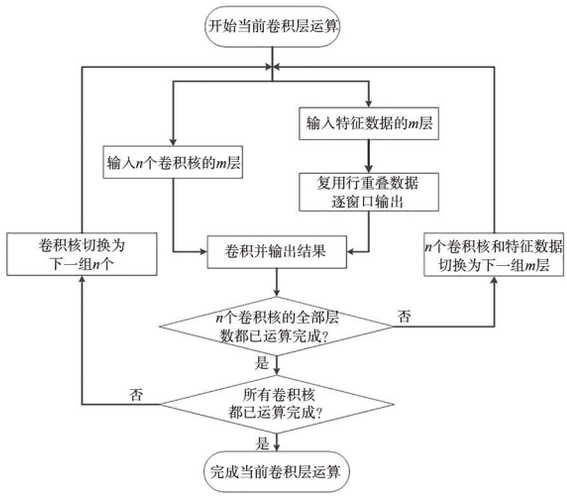

图4　切片卷积流程图

Fig. 4　Slice convolution flow chart

储和读取的速度快，但是容量小，无法将网络权重数据全部保存在片上，只能够将其存于片外存储DDR中，再分批输入到片上进行计算。

卷积计算时一个输出通道的结果需要累加全部输入通道的中间计算结果。为了减少片上缓存的占用，采用输入通道切片的方式处理特征数据。计算完成一组输出通道对应的全部输入通道切片后，再切换到下一组输出通道。

切片卷积的整体流程如图4所示。开始某一卷积层计算后，循环完成部分卷积核的切片卷积计算。待部分卷积核的全部层卷积完成后，更换卷积核继续进行循环卷积，直至全部卷积核都已运算完成，结束当前卷积层运算。

为了便于不同输入通道同时计算，使用特征数据融合设计。原输入顺序是输入第一通道后再输入下一通道，这种方式需要将全部数据输入完成后才可开始计算，占用较多的存储空间。而经过数据融合，可以将同一位置的4个输入通道数据组合传输，如图5所示。这种设计可以充分利用输入数据位宽，在得到中间结果后，不需要

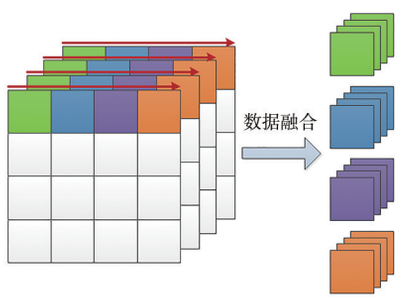

图5　特征数据融合设计图

Fig. 5　Design drawing of feature data fusion

暂存中间数据就可以完成结果累加，减少了片上缓存的占用。

在输入数据缓存复用模块中，因为Winograd卷积的特殊性，输出窗口大小为4×4，步长为2。这就导致相邻窗口间有步长为2的数据重叠。为了复用行重叠数据，本文设计了循环复用的输入数据缓存模块，如图6所示。设置6个Block RAM，每个存储一行特征数据。数据以128 bit位宽顺

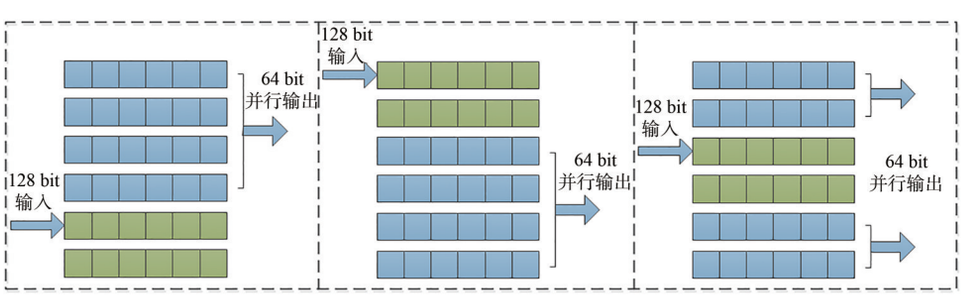

图6　输入数据缓存复用模块示意图

Fig. 6　Schematic diagram of input data buffer reuse module

序写入，当写入完成前4个Block RAM后，在写入第5个Block RAM的同时，并行输出前4个Block RAM。当Block RAM数据全部输出后，第5、6个Block RAM也完成写入，此时在写入第1个Block RAM的同时，复用第3、4个Block RAM数据，并行输出后4个Block RAM。依此交替，3个状态构成循环的一个周期。在降低片上存储占用的同时，实现了串并转换和行数据复用。

4.3 Winograd流水线卷积模块

卷积计算存在内部的并行性，分别是行并行、列并行、输入通道并行、输出通道并行和核内并行。从算法层面看，这5种并行方式可以任意组合先后顺序。但从硬件实现的角度，合理的并行计算方式可以节省片上资源，提高计算效率。经分析，本文选择使用4输入通道并行、8输出通道并行、核内16并行度的方式进行卷积计算。

Winograd卷积需要逐步计算完成，采用流水线的设计方式可以提高计算效率。卷积模块共分为六级流水线，其结构如图7所示。

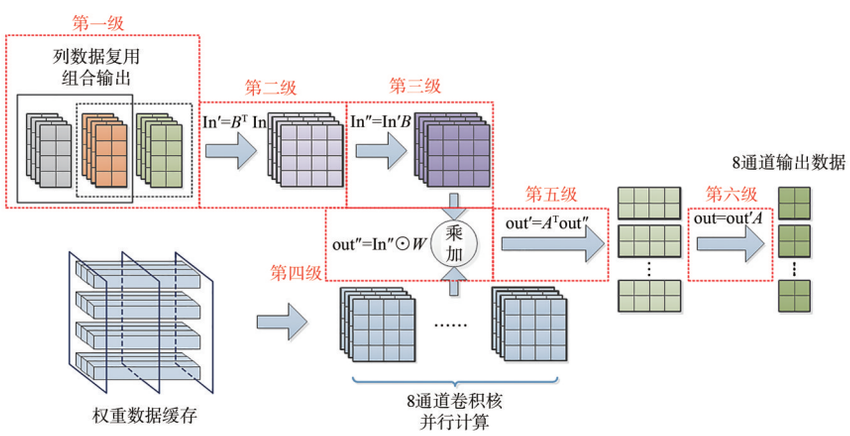

图7 六级流水线卷积设计图

Fig. 7 Convolution design drawing of six stage assembly line

第一级是列数据复用组合输出。模块接收并暂存输入数据缓存模块输出的4×2数据，即图中的灰色数据块。等下一周期橙色数据块输入后，组合为4×4窗口大小输出到第二级，便于下一级进行Winograd矩阵变换。暂存数据也替换为橙色数据块。下一周期绿色数据块输入后再次组合输出，如此循环往复。缓存的4行2列数据每周期切换，实现了相邻窗口之间的列重叠数据复用，提高了数据利用效率。

第二级和第三级是矩阵变换。特征矩阵和权重矩阵都需要进行变换。由于权重是预先训练完成的，可以提前变换后保存在DDR上直接输入，降低片上资源的占用。因此只需要对特征矩阵乘上常数矩阵进行变换。

第四级是多通道并行乘加。多输入通道和多输出通道可以并行相乘，这也是计算最为集中的环节。不同输入通道的中间结果累加后才可以得到输出数据，为了优化时序，使用加法树的方式进行累加。

最后经过第五级和第六级矩阵变换之后，就可以得到8个输出通道的2×2输出数据。作为中间结果输出到缓存模块暂存。

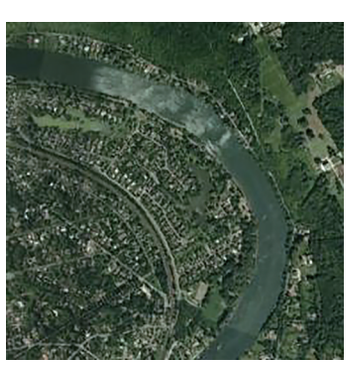

图8 河流图的遥感图像

Fig. 8 Remote sensing image of river map

以VGG-16网络的第一层卷积为例，当输入图像为图8时，可以得到输出前8个输出通道结果以灰度图展示如图9。

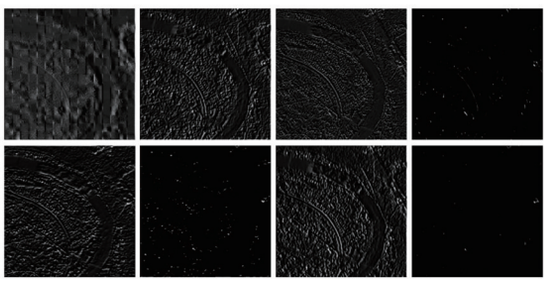

图9　卷积输出灰度图

Fig. 9　Grayscale image of convolution output

4.4　数据累加输出模块

数据累加输出模块（图10）获取到卷积模块的输出数据后，按照输出通道不同，暂存在缓存A的不同Block RAM中。当存储完一组卷积核的一组输出结果后，在下一组结果输入前依次读出，一起输入加法树累加后再次存入Block RAM，覆盖之前的结果。当这组卷积核的全部结果计算完成后，切换缓存，使缓存B继续存储结果，缓存A则输出所有计算结果到DDR中。这种设计可以在不中断数据输入和计算的同时输出结果，减少了数据输出时的阻塞时间。

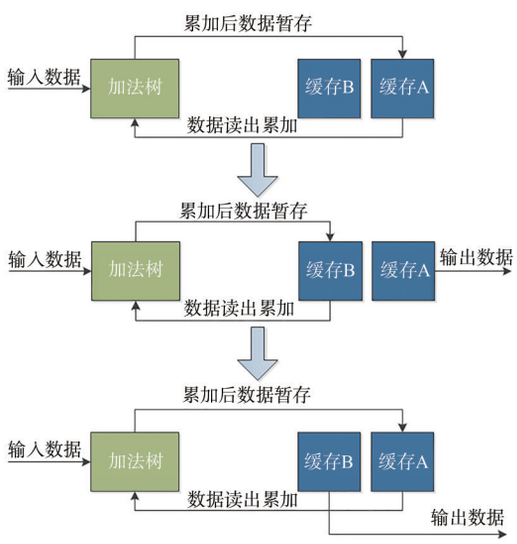

图10　数据累加输出模块状态图

Fig. 10　Status diagram of data accumulation output module

在卷积计算之后，裁剪了边缘处的数据，输出特征图像的尺寸会减小。随着网络层数的加深，如果不做处理，特征图像尺寸会越来越小，造成信息的丢失。为了防止这一现象，在一层数据输出时进行padding填充，即对特征图像周围补0。为了减少数据传输时间、节省存储资源，在设计中省略了第一行和最后一行全部数据为0的值，只在中间行的首尾列补0，在特征数据窗口读取时再补首末行的0。

5　实验结果及分析

5.1　实验环境

本文的实验平台为Xilinx公司的ZCU104开发板，芯片型号为XCZU7EV-2FFVC1156。PS端片外存储为2 GB大小的DDR4，PL端片上存储为38 Mb的Block RAM和Ultra RAM，可以满足实验需求。在RTL代码编写完成后，使用Vivado 2018.3进行仿真测试。在综合实现完成后，编写SDK程序，烧录上板进行实验。

本文基于经典的VGG-16网络进行了改进。VGG-16网络具有提取特征能力强，结构清晰简单、易于修改的特点，且其卷积核尺寸小，全部为3×3，更便于Winograd算法的应用。但是VGG-16网络主要针对ImageNet ILSVRC2012数据集进行1 000种标签的分类，为了将其适用于遥感图像分类的任务中，使用NWPU-RESISC45数据集对其进行迁移学习，并修改最后的全连接层输出为45，对应数据集的飞机、机场、棒球场、篮球场、沙滩等45种标签。修改后的VGG-16网络结构如表2所示。

表2　修改后的VGG-16的网络结构

Tab. 2　Revised VGG-16 network structure table

层次	输入尺寸	输出尺寸	操作数/G
Conv1-1	224×224×3	224×224×64	0.173 4
Conv1-2	224×224×64	224×224×64	3.699 3
Conv2-1	112×112×64	112×112×128	1.849 7
Conv2-2	112×112×128	112×112×128	3.699 3
Conv3-1	56×56×128	56×56×256	1.849 7
Conv3-2	56×56×256	56×56×256	3.699 3
Conv3-3	56×56×256	56×56×256	3.699 3
Conv4-1	28×28×256	28×28×512	1.849 7
Conv4-2	28×28×512	28×28×512	3.699 3
Conv4-3	28×28×512	28×28×512	3.699 3
Conv5-1	14×14×512	14×14×512	0.924 8
Conv5-2	14×14×512	14×14×512	0.924 8
Conv5-3	14×14×512	14×14×512	0.924 8
Fc1	7×7×512	1×1×4 096	0.205 5
Fc2	1×1×4 096	1×1×4 096	0.033 6
Fc3	1×1×4 096	1×1×45	0.000 03
累计	—	—	30.931 83

5.2 性能分析

设定加速器时钟频率为 200 MHz，经综合实现后硬件资源占用情况如表 3 所示。其中 LUT 表示片上所有查找表；LUTRAM 表示作为存储资源使用的查找表；BRAM 和 URAM 表示片上的专用块 RAM 存储器；DSP 则表示片上运算单元，主要用于乘法运算当中。

表 3 硬件资源占用情况

Tab. 3 Hardware resource usage

片上资源	总资源数	占用资源数	比例/%
LUT	230 400	152 943	66.38
LUTRAM	101 760	68 158	66.98
FF	460 800	88 296	19.16
BRAM	312	173	55.45
URAM	96	96	100
DSP	1 728	514	29.75

可以看出，加速器使用 DSP 资源比例相较 LUT 和 BRAM 等资源比例较低，原因在于数据带宽限制了数据传输速度。如果通过提高 DSP 使用量的方式继续提升加速效果，可以更换硬件资源更丰富、带宽更大的开发板平台或者使用 PL 端的 DDR 存储图像和权重数据。在时序稳定的前提下，也可以通过提高时钟频率来获得更快的加速效果。

实验时，首先读取 SD 卡中的权重数据和图像数据到 DDR 内存上，读取后的图像数据如图 11 所示，位于 DDR 内存的 0×1 000 000 地址处。

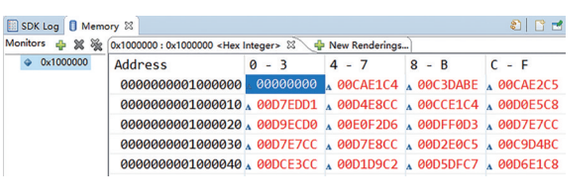

图 11 图像数据存入 DDR 内存展示图

Fig. 11 Display diagram of image data stored in DDR memory

硬件实验平台如图 12 所示。读取图 8 所示图像后完成计算，通过串口输出到上位机得到图像分类的标签与分类概率，如图 13 所示。

在 NWPU-RESISC45 数据集上验证后得到 FPGA 实现遥感图像分类的 TOP-1 准确率，与 GPU 实现的准确率对比如表 4 所示。在精度损失不超过 1% 的情况下，将网络大小压缩为原大小的 28.6%，效果显著。

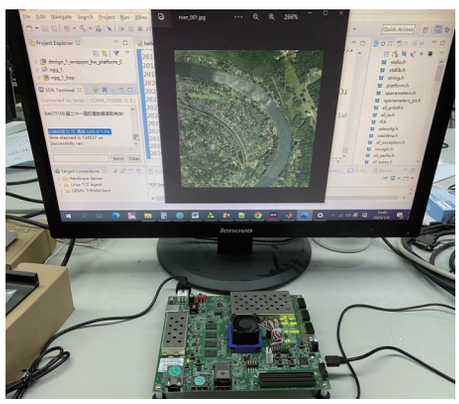

图 12 ZCU104 硬件平台测试图

Fig. 12 ZCU104 hardware platform test diagram

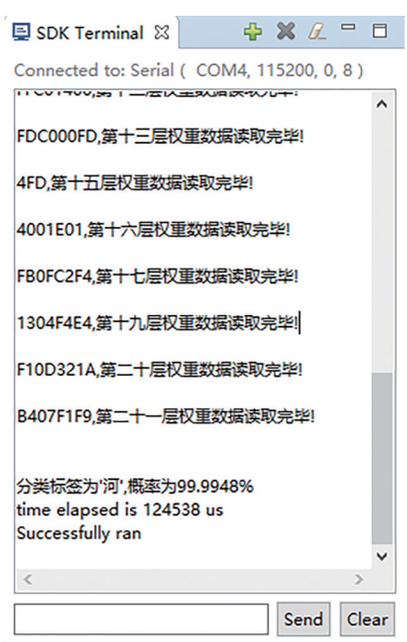

图 13 串口输出结果图

Fig. 13 Serial port output result diagram

表 4 硬件实现后网络精度对比

Tab. 4 Comparison of network accuracy after hardware implementation

网络	大小/MB	TOP-1 acc	损失
VGG-16(GPU)	527	79.4%	
VGG-16(FPGA)	151	78.6%	0.8%

将加速器与其他方案的实验结果进行对比，如表 5 所示。由于不同的设计方案采用的 FPGA 平台不同，因此将计算效率和能效作为主要性能指标进行分析。本文提出的加速器与文献[16]的

表5　与现有FPGA加速方案的对比

Tab. 5　Comparison with existing FPGA acceleration schemes

VGG16加速	文献[18]	文献[17]	文献[16]	本文
平台	XC7VX485T	Zynq XC7Z045	Ultrascale KU060	ZCU104
时钟/MHz	100	150	200	200
数据精度	8 bit	16 bit	16 bit	8 bit
DSP	1 796	780	1 058	514
卷积层计算性能/GOPS	758.19	187.8	310	354.5
计算效率/GOPS(DSP)	0.422	0.241	0.293	0.69
功耗/W	17.8	9.63	25	9.04
能效/(GOPS·W^{-1})	42.59	19.5	12.4	39.21

Caffeine结构、文献[17]的Angel-Eye结构相比，使用了更少的硬件DSP资源，实现了更高的计算吞吐量。与文献[18]中提出的将乘加树与脉动阵列相结合的乘加阵列相比，虽然由于硬件规模不同，在卷积层计算性能方面存在差距，但是本设计在能效比上较为接近，且DSP计算效率提升为1.635倍。

6　结　　论

本文提出了一种基于FPGA的Winograd算法卷积神经网络加速器。在卷积算法方面，采用Winograd算法减少乘法运算量，并设计了输入数据缓存复用和流水线相结合的方式，充分复用了行列间的重叠数据，提高了传输效率。使用8位定点数对权重和数据量化，提高数据吞吐速率和加速器计算效率。经过六级流水线并行卷积计算之后，得到的数据分组循环累加，降低了片上缓存的占用。在ZCU104开发板上的实验表明，加速器卷积层计算性能达到了354.5 GOPS，片上DSP计算效率达到0.69，与相关研究相比，实现了1.6倍以上的提升。本设计在计算效率上优于其他FPGA加速器设计方案，能够以较高能效完成遥感图像分类任务中的硬件加速计算。

参　考　文　献：

[1] CHEN H L, HUANG L Z, LIU T R, *et al*. Fourier imager network (FIN): a deep neural network for hologram reconstruction with superior external generalization [J]. *Light: Science & Applications*, 2022, 11(1): 254.

[2] ZUO C, QIAN J M, FENG S J, *et al*. Deep learning in optical metrology: a review [J]. *Light: Science & Applications*, 2022, 11(1): 39.

[3] LUO W J, SCHWING A G, URTASUN R. Efficient deep learning for stereo matching [C]//*Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*. Las Vegas: IEEE, 2016: 5695-5703.

[4] 邓箴，王一斌，刘立波. 视觉注意机制的注意残差稠密神经网络弱光照图像增强[J]. 液晶与显示，2021，36(11)：1463-1473.
DENG Z, WANG Y B, LIU L B. Attentive residual dense network of visual attention mechanism for weakly illuminated image enhancement [J]. *Chinese Journal of Liquid Crystals and Displays*, 2021, 36(11): 1463-1473. (in Chinese)

[5] 李珣，李林鹏，LAZOVIK A，等. 基于改进双流卷积递归神经网络的RGB-D物体识别方法[J]. 光电工程，2021，48(2)：200069.
LI X, LI L P, LAZOVIK A, *et al*. RGB-D object recognition algorithm based on improved double stream convolution recursive neural network [J]. *Opto-Electronic Engineering*, 2021, 48(2): 200069. (in Chinese)

[6] 周泰汉，王振. 基于多尺度特征增强卷积神经网络遥感目标检测算法[J]. 电光与控制，2022，29(11)：74-81.
ZHOU Q H, WANG Z. A remote sensing target detection algorithm based on multi-scale feature enhancement CNNs [J]. *Electronics Optics & Control*, 2022, 29(11): 74-81. (in Chinese)

[7] 卢丽强,郑思泽,肖倾城,等. 面向卷积神经网络的FPGA设计[J]. 中国科学:信息科学,2019,49(3):277-294.
LU L Q, ZHENG S Z, XIAO Q C, *et al*. Accelerating convolutional neural networks on FPGAs [J]. *Scientia Sinica Informationis*, 2019, 49(3): 277-294. (in Chinese)

[8] 张军阳,王慧丽,郭阳,等. 深度学习相关研究综述[J]. 计算机应用研究,2018,35(7):1921-1928, 1936.
ZHANG J Y, WANG H L, GUO Y, *et al*. Review of deep learning [J]. *Application Research of Computers*, 2018, 35(7): 1921-1928, 1936. (in Chinese)

[9] WEI X C, YU C H, ZHANG P, *et al*. Automated systolic array architecture synthesis for high throughput CNN inference on FPGAs [C]//*Proceedings of the* 54*th ACM/EDAC/IEEE Design Automation Conference*. Austin: IEEE, 2017: 1-6.

[10] LIN X H, YIN S Y, TU F B, *et al*. LCP: A layer clusters paralleling mapping method for accelerating inception and residual networks on FPGA [C]//*Proceedings of* 2018 55*th ACM/ESDA/IEEE Design Automation Conference*. San Francisco: IEEE, 2018: 1-6.

[11] ZHANG C, PRASANNA V. Frequency domain acceleration of convolutional neural networks on CPU-FPGA shared memory system [C]//*Proceedings of* 2017 *ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*. Monterey: ACM, 2017.

[12] LAVIN A, GRAY S. Fast algorithms for convolutional neural networks [C]//*Proceedings of* 2016 *IEEE Conference on Computer Vision and Pattern Recognition*. Las Vegas: IEEE, 2016: 4013-4021.

[13] LIANG Y, LU L Q, XIAO Q C, *et al*. Evaluating fast algorithms for convolutional neural networks on FPGAs [J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020, 39(4): 857-870.

[14] 黄程程,董胄霄,李钊. 基于二维Winograd算法的深流水线5×5卷积方法[J]. 计算机应用,2021,41(8):2258-2264.
HUANG C C, DONG X X, LI Z. Deep pipeline 5×5 convolution method based on two-dimensional Winograd algorithm [J]. *Journal of Computer Applications*, 2021, 41(8): 2258-2264. (in Chinese)

[15] WINOGRAD S. *Arithmetic Complexity of Computations* [M]. Philadelphia: Society for Industrial and Applied Mathematics, 1980: 18-23.

[16] ZHANG C, FANG Z M, ZHOU P P, *et al*. Caffeine: towards uniformed representation and acceleration for deep convolutional neural networks [C]//*Proceedings of* 2016 *IEEE/ACM International Conference on Computer-Aided Design*. Austin: IEEE, 2016.

[17] GUO K Y, SUI L Z, QIU J T, *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*, 2018, 37(1): 35-47.

[18] 梅志伟. 卷积神经网络加速模块设计与FPGA实现[D]. 杭州:浙江大学,2020.
MEI Z W. Design and FPGA implementation of convolutional neutral network acceleration module [D]. Hangzhou: Zhejiang University, 2020. (in Chinese)

作者简介:

牛朝旭(1998—),男,山东菏泽人,硕士研究生,2020年于东北电力大学获得学士学位,主要从事CNN加速器设计方面的研究。E-mail:niuzhaoxu20@mails.ucas.edu.cn

孙海江(1980—),男,吉林辉南人,博士,研究员,2012年于中国科学院长春光学精密机械与物理研究所获得博士学位,主要从事目标识别与跟踪技术及高清视频图像增强显示方面的研究。E-mail:sunhaijing@126.com