文章编号:1671-4598(2022)09-0162-08 DOI:10.16526/j.cnki.11-4762/tp.2022.09.025 中图分类号:TP331.2; TN47 文献标识码:A

基于 FPGA 的量化推理 CNN 加速系统研究与设计

何家俊，苏成悦，罗荣芳，施振华，陈堆钰，罗俊丰
（广东工业大学 物理与光电工程学院，广州 510006）

摘要：基于 FPGA 的量化推理设计了 CNN 加速系统；通过对主流的深度神经网络结构的运算特性分析，使用（Density-Based Spatial Clustering of Applications with Noise）DBSCAN 聚类算法截取阈值的 INT8 量化推理方法，融合深度神经网络全连接，减少数据运算位宽和压缩网络大小，在准确率损失很小的情况下有效压缩了网络结构；基于 LeNet-5、VGG-16 与 ResNet-50 的 CNN 网络结构，设计出量化 CNN 加速系统并进行校验；实验结果表明，网络参数和输入特征数据量化精度为 8-bits 时，网络压缩率在 25%的情况下，网络准确率的损失低于 1%；在 Xilinx XC7K325 平台上量化推理 CNN 加速系统的运行频率为 450 MHz，与其他相似类型的加速器比较，其 GOPS 性能提升 2 倍。

关键词：卷积神经网络；量化推理；硬件加速；FPGA；DBSCAN

Research and Design of CNN Acceleration System for Quantitative Reasoning Based on FPGA

HE Jiajun, SU Chenyue, LUO Rongfang, SHI Zhenhua, CHEN Duiyu, LUO Junfeng
(School of Physics and Optoelectronic Engineering, Guangdong University of Technology, Guangzhou 510006, China)

Abstract: Based on the quantitative reasoning of FPGA, the Convolutional Neural Network (CNN) acceleration system is designed. Through the analysis of the operation characteristics of the mainstream deep neural network structure, the INT8 quantitative reasoning method of intercepting the threshold using the density based spatial clustering of applications with noise (DBSCAN) clustering algorithm is used to integrate the full connection of the deep neural network, reduces the data operation bit width and compresses the network size, and effectively compresses the network structure with little loss of accuracy. Based on the CNN network structure of LeNet-5, VGG-16 and ResNet-50, a quantitative CNN acceleration system is designed and verified. The experimental results show that, when the quantization accuracy of network parameters and input characteristic data is 8-bits, the loss of network accuracy is less than 1% as the network compression rate is 25%. On Xilinx xc7k325 platform, the running frequency of CNN acceleration system is 450 MHz. Compared with other similar accelerators, the GOPs performance is improved by 2 times.

Keywords: convolutional neural network (CNN); quantization; hardware acceleration; FPGA; DBSCAN

0 引言

卷积神经网络（CNN）在图像处理分类、目标检测等卷积神经网络应用上有了较大的突破[1]。随着多种深层次卷积神经网络的提出[2]，卷积神经网络的网络结构深度和网络层中参数量以及计算量也随之不断提高。将 CNN 在有限资源的嵌入式设备中计算资源高效得部署成为非常有意义的研究。

许多研究人员在优化神经网络中计算性能[3]，提高外部内存访问效率方面提出了很多 CNN 硬件加速的方案，在实现方面只是考虑到小型网络的加速实现，对于大型网络的实现都缺乏有效的方案[4-6]。且多数网络加速只针对于卷积运算优化，少数网络在加速网络时针对网络全连接的优化[7]，全连接的优化对于网络的参数影响[8]也占据着主要的位置。在推理方面提出了很多量化方案[9]，采用的量化方案多数为定点数量化，在数据跨度比较大的情况下，采集的数据量不足导致损失率增加，降低算法识别率[10]。常用的动态量化推理 INT8 是通过人工设置值域范围，不能有效地确定推理量化范围[11]。取值范围过大，会过度采集量化数据，不能有效地抵抗异常参数数据导致量化结果与原函数偏差过大。取值范围过小，会导致数据采集不完整，也不能有效地量化拟合原始数据。

提出了一种结合 DBSCAN 密度聚类算法[12]的 INT8 动态量化推理算法，根据网络结构特性实现网络结构优化，设计一个基于多个 CNN 计算核心的硬件架构，优化加速器

收稿日期:2022-03-16； **修回日期**:2022-04-14。

作者简介:何家俊(1995-)，男，广东广州人，硕士研究生，工程师，主要从事电子信息技术、FPGA 方向的研究。
通讯作者:苏成悦(1961-)，男，湖南长沙人，博士研究生，教授，主要从事应用物理、应用电子方向的研究。

引用格式:何家俊，苏成悦，罗荣芳，等.基于 FPGA 的量化推理 CNN 加速系统研究与设计[J].计算机测量与控制，2022，30(9)：162-169.

的内存访问速率、资源使用以及卷积神经网络计算效果，并以 LetNet-5，VGG-16 以及 Resnet-50 算法为例部署至 FPGA 平台进行验证。

1　算法分析

1.1　CNN 网络

CNN 是由许多运算模块层组成，有卷积运算的功能模块层，激活数值功能模块层，最大值池化数值功能模块层和数据全连接运算功能模块层。CNN 会由输入图像数据到第一层功能模块层开始，通过运算获得新的多通道图像特征数据并依次进行下一层功能模块层的操作，每层的卷积层可以用下式表达：

$$f_{\text{in}}^{\text{out}} = \sum_{j=1}^{n_{\text{in}}} f_j^{\text{in}} \otimes g_{i,j} + b_i (1 \leq i \leq n_{\text{out}}) \tag{1}$$

其中：j 表示输入特征值，i 表示输出特征值，n_{in} 和 n_{out} 分别表示输入和输出的数量，$g_{i,j}$ 应用于第 j 输入特征值和第 i 输出特征值的卷积核。激活层是选取特征值中大于零的数值，每层的激活计算层可以用下式表达：

$$f^{\text{out}} = \begin{cases} f^{\text{in}} & \text{if } f^{\text{in}} > 0 \\ 0 & \text{if } f^{\text{in}} \leqslant 0 \end{cases} \tag{2}$$

最大值池化层是选取一个区域内最大的值作为输出特征值可以表示为：

$$f_{i,j}^{\text{out}} = \max_{p \times p} \begin{pmatrix} f_{m,n}^{\text{in}} & \cdots & f_{m,n+p-1}^{\text{in}} \\ \vdots & \ddots & \vdots \\ f_{m+p-1,n}^{\text{in}} & \cdots & f_{m+p-1,n+p-1}^{\text{in}} \end{pmatrix} \tag{3}$$

其中：p 是池化层内核的大小。这种非线性向下采样不仅减少了特征数的数量大小和后续计算功能模块的计算量，而且还可以提供一种形式的平移不变性，在不改变数据特性的情况下减少数据量。全连接层可以表示为：

$$f^{\text{out}} = \boldsymbol{W} f^{\text{in}} + b \tag{4}$$

其中：$\boldsymbol{W}$ 是输入输出变化矩阵，b 是偏移参数。在卷积运算后通常会使用激活运算层模块和池化运算模块层去除多余参数降低参数量。经过一次或多次的卷积激活、池化操作，数据以及降低并提取到一定的程度，接下来会使用全连接模块功能层使得数据相互通过权重参数和偏移参数算得结果并将其求和。

上述算法可以看出，卷积神经网络需要大量的乘法和加法运算。通过电路的形式并考虑 FPGA 可并行可串行的特性设计出算法需要做出相对应的分析，其中卷积运算需要考虑卷积核心的移动方法，以及不同深度神经网络算法中有不同的卷积层数、不同的卷积核数、输入通道数和输出通道数。需要设计出合理的数据缓存区提前将数据保存起来再进行卷积操作。在池化层和激活层不需要用到乘法器和加法器，但是用到比较器，且比较的范围不只是在同一行。

文中选用的 3 个网络分别 LetNet-5 网络两个卷积网络以及两个全连接网络的，略深层网络的 13 层卷积网络和 3 层全连接网络的 VGG-16，以及包含 49 层全连接网络和 1 层全连接网络的 Resnet-50。

1.2　网络参数与 MACC 计算量

由图 1 可以看出 LeNet-5 无论是参数量还是计算量都是最少的，而其余两个网络，VGG-16 和 Resnet-50 的 MACC 卷积计算量以及全连接计算量和网络的卷积参数和全连接参数量都不小。图 1 和图 2 分别对网络中主要算子的参数分布和网络中主要的计算量分布进行了统计。

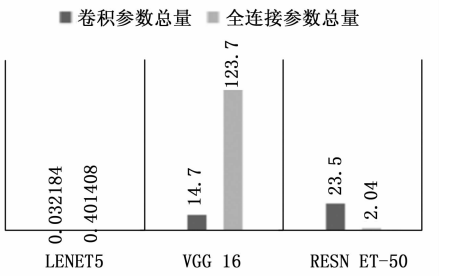

图 1　各网络的参数总量（M）

从图 1 可以看出 VGG-16 卷积算子权重参数少于 Resnet-50 的权重参数，而在全连接计算算子参数的总量中，VGG-16 的算子参数远远多于 Resnet-50 的全部算子参数的总和，而且 LeNet-5 和 VGG-16 网络中全连接参数总量分别是卷积参数总量的 12.5 倍和 8 倍。

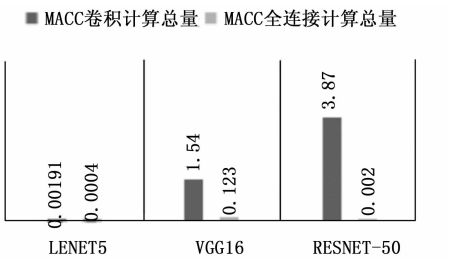

图 2　各网络参数的 MACC 计算总量（GFLOPs）

从图 2 可以看出每个神经网络的计算量都集中在卷积算子层，LeNet-5、VGG-16、Resnet-50 中卷积计算分别为全连接计算的 4.77 倍，1.25 倍和 1 935 倍，所有在卷积层中计算优化对整个 CNN 加速系统的影响占比更大。要做到优化好 CNN 加速系统模块，需要结合优化 VGG-16 这种需要优化大量全连接计算和 ResNet-50 这种需要大量卷积计算的特性，将全连接层可以看作是一个特殊的点乘运算，根据不同神经网络的特性设计一个专用的神经网络加速器尤为重要。

1.3　网络训练推理流程分析

目前大部分 CNN 的研究部署和训练都选择了 GPU 平台，平台对 FP32 浮点型数据计算提供了很好的并行运算开发生态有利于做大数据的图像识别训练，但是 GPU 其高功耗的特性导致其不能成为便携式移动设备的最优选择平台。FPGA 同时拥有并行运算能力以及低功耗特性，并且具有灵活的结构为移动设备部署 CNN 提供了有效的解决方案[13]。CNN 前推过程如图 3 表示。

卷积神经网络需要大量的乘法和加法运算。通过电路的形式并融合 FPGA 拥有并行和串行的特性设计出算法需

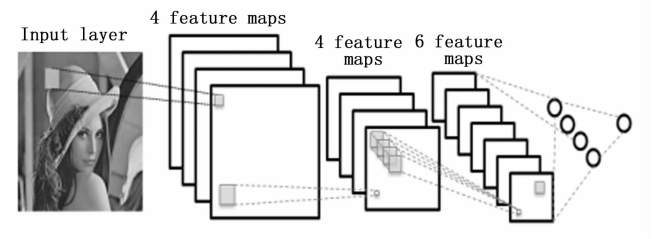

图 3 卷积神经网络前推过程

要做出相对应的分析，卷积运算需要考虑卷积核心的移动方法，以及考虑到不同的深度神经网络算法不同的卷积层数中有不同的卷积核数、输入通道数和输出通道数。需要设计出合理的数据缓存区需要提前将数据保存起来再进行卷积操作。池化层和激活层由比较器组成为主，且比较的范围不只是在同一行，如 2×2 的池化核需要第一行的图像数据与第二行的图像数据作对比，在电路设计上需要设计好缓冲区，保存第一行数据并且通过第二行数据输入后作对比。全连接层则只需要在其他计算完成后得到的数据输出的时候乘上权重和加上偏移值。训练过程如图 4 表示。

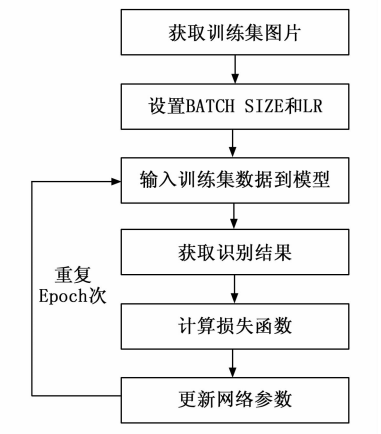

图 4 深度网络训练流程图

深度神经网络训练需要构建训练网络，预设训练次数（EPOCH），打乱数据集，划分训练集和验证集。接下来设置每次训练所抓取的数据样本数量（BATCH SIZE）和学习率（LR），这两个参数影响深度学习的速度和识别率。获取训练集中图像数据，并使用归一化处理。将数据传入到构建好的训练网络，获得识别结果。将结果通过与标识好的结果做一个损失函数计算，再更新网络中的参数。重复第三步当经过 EPOCH 次训练后得到训练参数和识别率。

基于深度卷积神经网络的训练数据量庞大而且只需要训练一次的原因，使用可以高速且具有丰富深度学习开发方案的 GPU 平台成为首选。训练不需要考虑功耗问题，通过 GPU 训练出来的模型参数做处理后，部署到基于 FPGA 平台的 CNN 加速器当中。

1.4 网络优化分析

CNN 加速研究中，常用的方法是通过压缩神经网络模型的方法达到缩小模型中数据体积大小和降低硬件的资源使用的效果，使得算法运算速率增加以及运行算法设备的功耗减少[14]。

通过将数据运算位数删减降低数据精度，有效地提高运算速度和降低功耗。使用在大规模的图像分类中，最为先进的 CNN 模型具有较深的网络层数和大量的神经网络权重参数和偏移参数。大量参数只能存储在外部存储器当中，运行算法时需要从外部参数读取并使用计算，加速性能需要和存储器读写的速度做匹配，在加速性能远大于储存器读写速率的情况下，存储器的读写带宽也就是储存器每次读写数据量的大小以及读写速率成为了 CNN 加速的性能瓶颈[15]。

2 量化与优化

2.1 网络量化

在 FP32 类型的数据下深度神经网络的参数都会集中在一个值域区间[16]，使得 FP32 大部分的数据段都是处于空闲状态，这时可以通过有效截取数据段来使得参数数据压缩，那么就可以得到一个量化的参数模型，如图 5 是 LeNet-5 训练后的参数分布数据图。

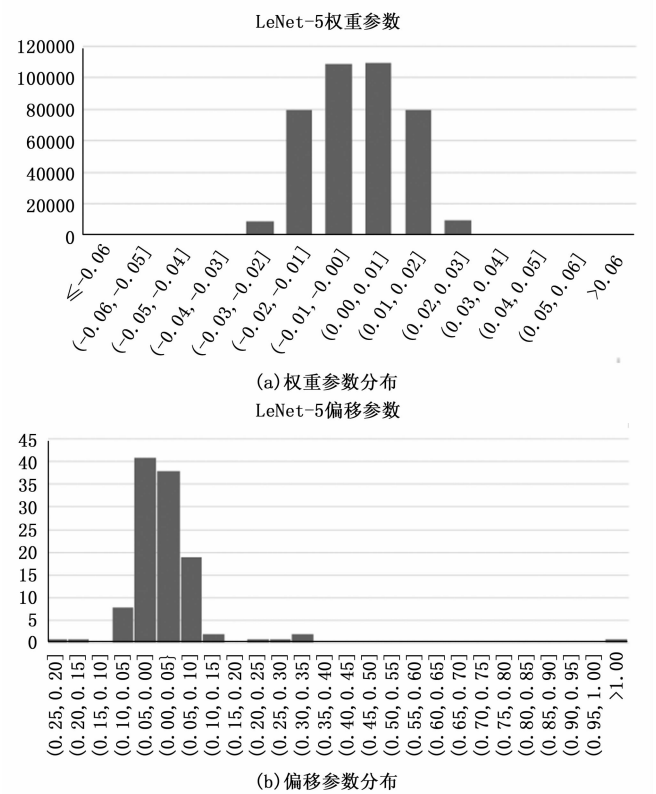

图 5 LeNet-5 参数数据分布

从图 5（a）和图 5（b）可以得到神经网络的权重参数主要分布和偏移量参数主要分布在一定范围，我们可以合理地设置区间为这个综合范围的最大值和最小值映射到 INT8 类型数据的最大值和最小值，其中相差间隔很小的数据会合并映射到一个数值。

在 FP32 转向 INT8 的过程中需要去除小数点后的数，

需要在映射到 8 位数据后进行舍入的操作。当存在少量远大于或小于大量数值参数的异常参数，则异常参数会影响整个量化过程，大量的正常参数才是在图像识别运算中起着重要的权重作用，合并到同一数值当中，减少了两个区域甚至多个区域中数据的可比性。合理选择量化的值域显得尤其重要。

2.1.1　量化 FP32 至 INT8 实现的计算公式

从输入图像数据转换成量化图像数据公式如下：

$$X_q = round\left(\frac{X_f}{s_x} + z_x\right) \tag{5}$$

其中：X_f 为输入图片特征数，X_q 为 X_f 量化后的图片特征数，S_x 为图片特征数量化放缩因子，Z_x 为图片特征数量化偏移值。量化放缩因子是通过输入数据的值域范围和需要量化的范围做一个比例计算，公式如下：

$$s_x = \text{scale}(X,n) = \frac{\max(X_f) - \min(X_f)}{2^n - 1} \tag{6}$$

式（5）Z_x 是通过放缩因子，量化结果以及原数据做一个偏移计算获取，公式如下：

$$z_x = \max(X_q) - \frac{\max(X_f)}{s_x} \tag{7}$$

同理权重的参数量化如同输入图像特征数的量化，给出的 3 个公式如下所示：

$$W_q = round\left(\frac{W_f}{s_w} + z_w\right) \tag{8}$$

$$s_w = scale(W,n) = \frac{\max(W_f) - \min(W_f)}{2^n - 1} \tag{9}$$

$$z_w = \max(W_q) - \frac{\max(W_f)}{s_w} \tag{10}$$

其中：W_f 为输入权重参数，W_q 为 W_f 量化后的权重参数，S_w 为权重参数量化放缩因子，Z_w 为权重参数量量化偏移值。量化计算使用在深度网络训练后得到参数进行量化推理调整。

2.1.2　量化中舍入公式：

量化中的舍入公式使用了随机离散舍入函数方法[17]，我们可以定义为：

$$Round(x) = \begin{cases} \lfloor x \rfloor & 1 - \dfrac{x - \lfloor x \rfloor}{\varepsilon} \\ \lfloor x \rfloor + \varepsilon & \dfrac{x - \lfloor x \rfloor}{\varepsilon} \end{cases} \tag{11}$$

在 CNN 量化中，使用四舍五入的 round 方法会产生一定的误差，在大量的卷积运算会使得这种误差放大，导致图像识别的准确率降低。随机离散舍入函数方法中，采用随机因子，使得函数期望为 x。有助于在量化过程中舍入后减少误差，减少算法损失率。

2.1.3　DBSCAN 聚类算法

DBSCAN 是基于密度的分类算法。算法通过设置中心点直径范围，和中心直径范围内数据量遍历出数据中心点的位置。将与中心相连的点位数据收集到同一数据集中，生成多个数据集。通过删除含有数据少的中心数据团，获取存在大量数据的中心数据团，从而确定量化参数大致分布区域，截取数据的值域区间，提高量化后与量化前的相似度。如图 6 是 LeNet-5 的两个卷积层量化前后对比图。

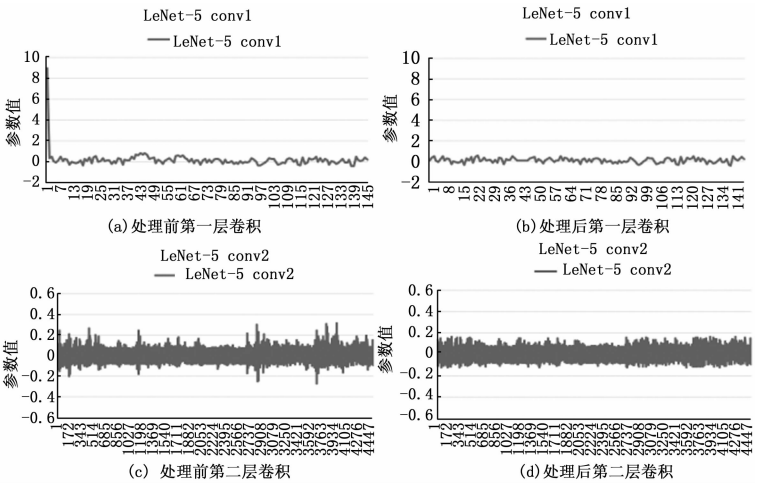

图 6　卷积参数 DBSCAN 聚类算法处理数据图

从图 6（a）以及图 6（c）可以看出在训练后的卷积参数会出现异常数据。异常数据会导致在设置 INT8 推理的阈值过大，大部分的参数分布在 0.1 与 0.2 之间，如图 4（a）出现了最大值 9，若设置 9 为量化最大值则会出现 0.2 到 9 之间存在大部分不占据参数的空间稀释原本参数分布区域的精度参数从而导致量化后结果与原函数的相似度降低。

从图 6（b）以及图 6（d）可以看出经过 DBSCAN 聚类算法处理后相对于处理前的数据减少了很多异常数据，显示聚集密度大得数据集，可以直接截取数据集的最大值和最小值通过公式（6）计算出量化放缩因子，极大地去除异常数据，可以通过算法处理使得数据参数有效得并量化贴合，可以明显有效得去除了数据毛刺，有助于数据量化。

2.2　全连接优化计算公式

LeNet-5 和 VGG-16 等深度神经网络中会存在多个全连接，全连接可以看作是两个矩阵的相乘，将第一层全连接的权重参数等价于 $\boldsymbol{W}$，第一层偏移量等价于 $\boldsymbol{B}$ 得到公式如下：

$$\boldsymbol{W} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{nm} \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \boldsymbol{X} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots\ x_n \end{bmatrix} \tag{12}$$

第一层全连接层的结果等价于 $\boldsymbol{H}$ 可以得到：

$$\boldsymbol{H} = \begin{bmatrix} h_1 & h_2 & \cdots & h_m \end{bmatrix}^T = \boldsymbol{W} * \boldsymbol{X} + \boldsymbol{B} \tag{13}$$

其中：n 为全连接输入数，m 为输出通道数，h 为全连接结果，x 为输入特征值，w 为权重参数，b 为偏移参数。同理可将第二层全连接的权重参数等价于 $\boldsymbol{U}$，第二层全连接的偏移参数等价于 $\boldsymbol{D}$：

$$\boldsymbol{U} = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1m} \\ u_{21} & u_{22} & \cdots & u_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ u_{k1} & u_{k2} & \cdots & u_{km} \end{bmatrix}, \boldsymbol{D} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_k \end{bmatrix} \tag{14}$$

得到第二层全连接最后结果 $\boldsymbol{FC}$，公式如下：

$$\boldsymbol{FC}=\begin{bmatrix}fc_1 & fc_2 & \cdots & fc_k\end{bmatrix}^T=\boldsymbol{U}*\boldsymbol{H}+\boldsymbol{D} \tag{15}$$

其中：k 为输出通道数，fc 为全连接结果。若用式（13）的全连接带入到式（15）中的 $\boldsymbol{H}$ 中，可以将（13）、（15）两式融合成一式：

$$\boldsymbol{FC}=\boldsymbol{U}*(\boldsymbol{W}*\boldsymbol{X}+\boldsymbol{B})+\boldsymbol{D} \tag{16}$$

将（16）中括号拆开，使第二层全连接权重参数与第一层全连接权重参数结合成为总的全连接权重参数，使第二层全连接参数结合第一层全连接偏移参数再加上第二层全连接偏移参数得到总的全连接的偏移参数，新成立的全连接权重矩阵和偏移矩阵分别是：

$$\boldsymbol{W}_n=\boldsymbol{U}*\boldsymbol{W} \tag{17}$$

$$\boldsymbol{B}_n=\boldsymbol{U}*\boldsymbol{B}+\boldsymbol{D} \tag{18}$$

融合后全连接可以简单描述为：

$$\boldsymbol{F}=\boldsymbol{W}_n*\boldsymbol{X}+\boldsymbol{B}_n \tag{19}$$

选用 LeNet-5 和 VGG-16 全连接融合效果对比如表 1 所示。

表 1 全连接层融合对比效果

网络	全连接 MACC 计算总量	全连接参数总量
LeNet-5	40 800	41 004
LeNet-5(FC 融合)	21 504	21 588
VGG-16	123 633 664	123 642 856
VGG-16(FC 融合)	4 096 000	4 097 000

可以看出 LeNet-5 和 VGG-16 在全连接融合后 MACC 计算总量的压缩率分别为 52.9%和 3.3%，全连接参数总量的压缩率分别为 52.6%和 3.3%。

拥有少量全连接参数和计算量的 LeNet-5 网络拥有很好的压缩效果。针对 VGG-16 全连接计算量占据大量参数和计算的特性，VGG-16 在向前推算的参数量和计算量在全连接层融合压缩起到了明显的效果。

3 CNN 推理加速器模块设计

使用 GPU 平台进行 Fashion MNIST（FM）和 CIFAR-10（CR）的数据集的训练并获取各网络模型参数，将参数进行量化数据分析和融合全连接参数等操作得到最后使用到 FPGA 平台上 CNN 加速系统中。

基于以上量化理论，CNN 加速系统需要考虑 CNN 网络在卷积运算的运算速率。CNN 卷积网络数据在并行运算中表现较好，在设计加速电路中需要运用并行的电路设计。

通过对网络的分析，系统同时需要多个模块组合并按顺序执行计算。运用多个计算核心的方法组合加速系统，通过控制系统控制执行顺序可以灵活地调用硬件资源完成 CNN 深度加速。FPGA 设计的 CNN 推理加速器模块设计如图 7 所示。

加速器主要包含了片上系统控制中心模块、内存缓存模块、深度神经网络计算单元核心模块、单元核心控制仲裁模块和输入输出模块。使用 C++代码实现深度神经网络的结构，通过系统控制中心模块调度内存缓存模块，将权重数据和偏移数据与图像数据以及神经网络单元核心的参数通过 DMA 传输进入深度神经网络计算模块中进行前推计算，最后输出数据返回到系统控制中心，完成整个深度神经网络加速过程。

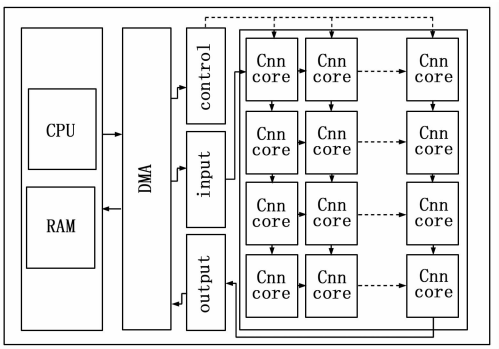

图 7 深度神经网络推理加速器

3.1 内存缓存与数据传输

片上内存主要保存了当前神经网络的权重数据和偏移数据，数据传入神经网络模块传输设计如图 8 所示。

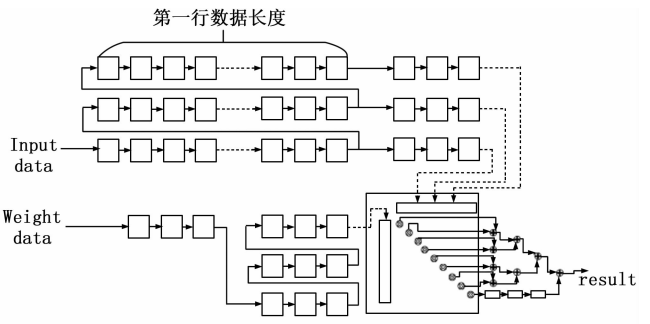

图 8 神经网络模块传输图

考虑到卷积操作多数是以 3×3 的卷积核结构对特征图进行滑动操作，固神经网络模块数据首先读取数据的前三行，新的数据行替换旧的数据行。第一次卷积需要等待三次数据的读取，后续数据可以直接替换旧数据进行卷积，当此次卷积实现了输入特征数据面积数且没有新的数据输入时完成卷积并输出空闲状态。

上述电路实现卷积的运算读取内存中数据[18]，每次时钟只需要读取一个数据量即可，并使用了流水线设计实现了加法树和乘法的功能优化了内存读写效率和在很短的等待时间内执行多条运行命令，提高了卷积计算的速度。

3.2 深度神经网络计算单元核心模块

计算单元核心模块是加速器的主要模块，其中有量化模块（Quantization）、控制模块（CORE CONTORL）、深度神经网络算子等计算模块，其结构如图 9 所示。

量化模块（Quantization）中随机数使用的是 LFSR 随机数产生电路生成随机因子作用于离散舍入计算得到量化放缩因子。模块输入数据后经过量化处理，将 FP32 数据乘以量化放缩因子并加上量化偏移值，得到最后的 INT8 数据，在卷积和全连接的计算中使用 INT8 计算，不需要使用

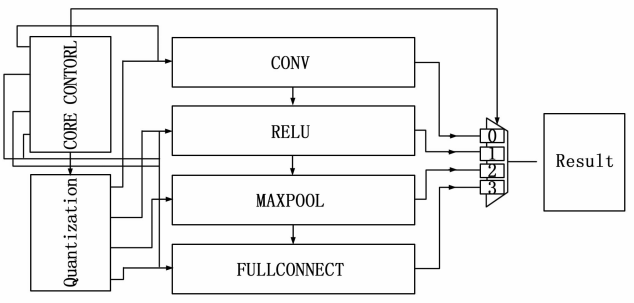

图 9　神经网络计算单元核心模块

浮点型的运算电路，减少加速系统对 DSP 使用的同时减少运算数据的位数达到电路系统在运行时降低功耗降低计算量。

控制模块（CORE CONTORL）通过代码控制图像数据输入到指定算子开始执行运算，并通过指定算子运行结束后输出结果，比如只需要数据从卷积计算到全连接，则控制使能卷积并输入数据，控制选择器将全连接的数据和计算完成标志输出结果（Result）。深度神经网络算子模块中的算子由卷积到全连接串联而成的计算模块，顺序是由卷积到全连接的计算顺序，因为每个模块之间都是通过先入先出队列模块 FIFO（First Input First Output）做中间层衔接，每个模块之间的输入数据和输出数据都是规定为一个有利于每个模块间交叉衔接。

深度神经网络算子模块中的算子由卷积到全连接串联而成的计算模块，顺序是由卷积到全连接的计算顺序，因为每个模块之间都是通过先入先出队列模块 FIFO（First Input First Output）做中间层衔接，每个模块之间的输入数据和输出数据都是规定为一个有利于每个模块间交叉衔接。卷积计算模块如图 10 表示。

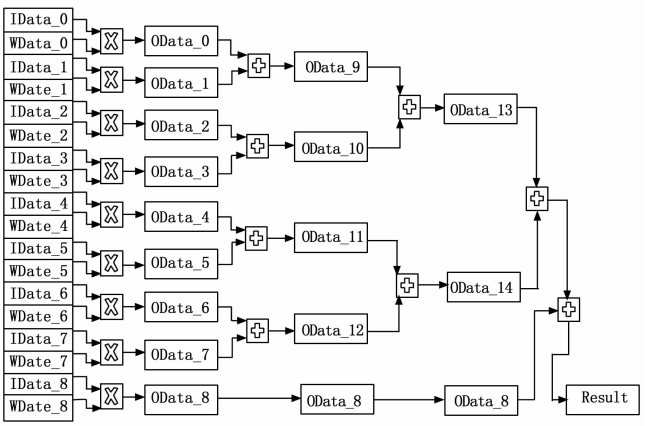

图 10　卷积模块设计

卷积计算模块的设计是由加法树和 9 个乘法器组成，由数据输入完成后将输入数据用乘法器相乘得到第一层数据，再通过加法器将数据相加得到第二层数据，进而再进行三次相加和两次数据平移，最后结果由最后两个数据相加得到。

激活层是当卷积数据输出时通过比较器与 8 位数据 127 值作对比，若比 127 大则可以认为在算法中大于 0 输出原值，若小于 127 则在算法中认为小于 0 输出数据 0。

全连接层则是通过一个乘法器和一个加法器，实现单个数据通过 FIFO 输入后进行全连接的运算后通过 FIFO 输出。

3.3　单元核心控制仲裁多核加速

控制模块里面包含了多个单元核心仲裁器，每个仲裁核心会将处理数据优先级通过优先分配算法分配给空闲的神经网络运算单元核心进行卷积神经网络的运算，结构如图 11 所示。

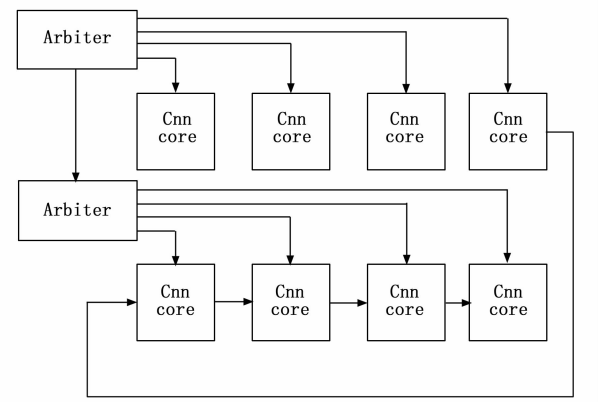

图 11　仲裁模块结构

当数据传入时，控制模块会查看当前空闲的神经网络单元核心算子，若当前仲裁卷积神经网络算子都处于繁忙阶段，则会将消息传输到下一个仲裁中，仲裁都处于繁忙阶段的情况下，则当前计算直接跳过，当一个神经网络算子完成当此计算，会将数据发到下一个神经网络算子，且让仲裁控制器控制当前数据是否要继续计算或再传送到下一个节点。通过将数据同时发送到多个子仲裁模块，实现数据并行运算。卷积运算中需要运算多个输出通道，可以按 FPGA 平台性能合理执行单次最大并行运算核心或通过自定义计时器和有限状态机 FSM（Finite State Machine）控制数据的再传输和再计算实现单个计算核心组的多次使用。

4　实验结果与分析

利用 GPU 平台以及 Pytorch 软件框架可以有效地对深度卷积神经网络模型进行大规模的数据集训练，其中数据集使用了 Fashion MNIST（FM）和 CIFAR-10（CR）为数据集 。

因为要计算在量化深度网络结构在网络计算量和网络参数的压缩率与识别效果，需要在平台上记录不同网络、不同数据集、不同前推运算的条件下记录相关参数。通过计算量化前 LeNet-5、VGG-16 和 ResNet-50 的网络大小与量化后网络的面积计算压缩率，压缩效果和量化网络对数据集的识别效果见表 2 和表 3。

表 2 量化面积

网络	网络大小/(M)	量化网络大小/(M)	压缩率/%
LeNet-5	1.66	0.41	24.6%
VGG-16	527	70.5	13.3%
ResNet-50	98.1	24.0	24.4%

可以看到 LeNet-5 的网络大小在量化前是 1.66 M，量化网络后大小变成了 0.41，压缩率为 24.6%，因为 LetNet-5 网络有两层卷积层数和两层全连接层，通过量化计算参数数据，压缩卷积层中占大量计算的数据位数，由原来的 FP32 变成 INT8 在数据上减少了 75%的位数据，在卷积层方面的压缩效果明显。通过融合全连接层，LetNet-5 本来由原来的两层全连接网络，融合成一层全连接网络，在计算全连接上不仅在量化中减少了大量的数据，在全连接上减少了实际运算次数，达到明显的压缩效果。

从表中得知 VGG-16 网络的压缩效果是最明显的。由于 VGG-16 的全连接层的数据量大部分占据在全连接层，全连接层的层数是 3 个网络中层数最多的，在全精度的情况下 VGG-16 的网络大小是 527 M 而其中全连接的占比可以达到 89%即 469.3 M，全连接融合有效的将中间的全连接层数运算过程融合在了一起，将全连接的大小压缩到了 15.55 M。因为全连接层的数量对准确率有一定的影响，但是全连接层的融合是经过训练后合并而成的，不会影响训练后用于前推的神经网络算法。

表中 ResNet-50 的压缩效果与 LeNet-5 的压缩效果相似。在 ResNet-50 中不只是在计算上，在层数上都是以卷积计算占据主要部分，并且全连接层只有一层，融合全连接的方法在 ResNet-50 中不起作用。ResNet-50 中占据主要压缩优化效果的是由量化参数起作用，可以将原来 98.1 M 的网络大小压缩到 24 M，优化方案适应于 ResNet-50。

表 3 量化精度

网络	FP32 准确率/%	INT8 准确率/%	损失/%
LeNet-5，MN	89.9	89.0	0.9
LeNet-5，CR	59.8	59.0	0.8
VGG-16，MN	91.2	90.5	0.7
VGG-16，CR	88.5	88.0	0.5
ResNet-50，MN	92.3	91.7	0.6
ResNet-50，CR	90.68	90.09	0.5

从表 3 可以看到量化后的网络在在识别准确率上都维持在 1%以下的损失率。表明了在通过有效去除异常数据量化拟合和融合全连接层的方法可以有效地降低深度神经网络的网络大小而不会有过大的损失率，量化推理方案是有助于降低深度神经网络的复杂性并有助于降低电路设计相关深度神经网络算法电路的复杂性。

通过训练后的网络模型，在 GPU 平台下运行 FP32 全精度的识别 Fashion MNIST（FM）和 CIFAR-10（CR）的验证集测试并记录准确率。获取 GPU 平台下，网络训练模型中的参数，使用 C＋＋语言搭建前推深度计算网络模型，输入验证集。将编译好的 bitstream 文件导入到 FPGA 开发板中，可以得到加速器利用资源情况见表 4。

表 4 加速器资源利用

FPGA 资源	总量	使用量	利用率/%
LUT	203 800	65 640	32.20
FF	407 600	39 850	9.77
BRAM	445	250	56.17
DSP	220	840	26.1

本设计因为使用了多个深度神经网络计算单元核心模块，在一定程度上使用更多通道的并行运算上在，并且在经过推理量化后的计算不需要使用大量的 DSP 功能进行计算，本设计的 CNN 加速系统在计算性能上相比于其他同为 8 位计算的 FPGA 实现由较大的提升，其中峰值 154.95 GOPS，性能提高了 2 倍，详细对比见表 5。

表 5 加速器对比效果

对比	文献[19]	文献[20]	
FPGA	XC7Z020	XC7Z045	XC7K325
工作频率/MHz	214	330	450
量化方式	8-bits fixed	8-bits fixed	8-bits fixed
峰值性能 GOPS	84.45	62.85	154.95

5 结束语

为了提高硬件移动设备上实现卷积深度神经网络（CNN）的运行性能和降低算法运算给设备带来的功耗问题，通过对 CNN 的网络结构以及计算流程特性，设计出使用 DBSCAN 聚类算法实现量化值域截取有效得截取 INT8 推理算法的阈值，改变阈值选取方法以及全连接融合减少具有大量全连接层的深度网络数据量，并针对 FPGA 硬件和深度神经网络运算特性，设计出多个深度神经网络计算单元核心模块加速器。在 Fashion MNIST（FM）和 CIFAR-10（CR）的验证集上进行了性能测试。实验结果表示，在量化后的神经网络中，损失率在 1%以内，LeNet-5、VGG-16 和 ResNet-50 压缩分别为原来的 24.6%，13.3%和 24.4%，设计的加速器最高性能可以达到 154.95GOPS，提高了 2 倍。

参考文献：

[1] RUSSAKOVSKY O，DENG J，SU H，et al. Imagenet large scale visual recognition challenge [J]. International journal of computer vision，2015，115 (3)：211－252.

[2] HE K，ZHANG X，REN S，et al. Deep residual learning for image recognition [C] //Proceedings of the IEEE conference on computer vision and pattern recognition. 2016：770－778.

[3] WILLIAMS S，WATERMAN A，PATTERSON D. Roofline：an insightful visual performance model for multicore architectures [J]. Communications of the ACM，2009，52 (4)：65－76.

[4] LU L，LIANG Y，XIAO Q，et al. Evaluating fast algorithms

for convolutional neural networks on FPGAs [C] //2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM). IEEE, 2017: 101 - 108.
[5] 许思琦. 基于 Linux 的 FPGA＋ARM 高速数据采集系统设计 [J]. 计算机测量与控制，2017，25 (4)：34 - 34.
[6] 王　海，阙沛文. 超声信号采集模块的设计 [J]. 计算机测量与控制，2007，15 (6)：816 - 819.
[7] LIU D, CHEN T, LIU S, et al. Pudiannao: A polyvalent machine learning accelerator [J]. ACM SIGARCH Computer Architecture News, 2015, 43 (1): 369 - 381.
[8] ZHANG X, WANG J, ZHU C, et al. DNNBuilder: An automated tool for building high-performance DNN hardware accelerators for FPGAs [C] //2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). IEEE, 2018: 1 - 8.
[9] 巩　杰，赵　烁，何　虎，等. 基于 FPGA 的量化 CNN 加速系统设计 [J]. 计算机工程，2022，48 (3)：170 - 174，196.
[10] 满　涛，郭子豪，曲志坚. 卷积神经网络的 FPGA 并行加速设计与实现 [J]. 电讯技术，2021，61 (11)：1438 - 1445.
[11] 马晓光，蒋占军. 卷积神经网络图像识别算法的 FPGA 加速优化研究 [J]. 兰州交通大学学报，2021，40 (5)：51 - 57.
[12] SCHUBERT E, SANDER J, ESTER M, et al. DBSCAN revisited, revisited: why and how you should (still) use DBSCAN [J]. ACM Transactions on Database Systems (TODS), 2017, 42 (3): 1 - 21.
[13] 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. 2016: 26 - 35.
[14] ZHOU Q, GUO S, QU Z, et al. Octo: {INT8} Training with Loss-aware Compensation and Backward Quantization for Tiny On-device Learning [C] //2021 USENIX Annual Technical Conference (USENIX ATC 21). 2021: 177 - 191.
[15] 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. 2015: 161 - 170.
[16] ZHANG J, LI J. Improving the performance of OpenCL-based FPGA accelerator for convolutional neural network [C] //Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. 2017: 25 - 34.
[17] GUPTA S, AGRAWAL A, GOPALAKRISHNAN K, et al. Deep learning with limited numerical precision [C] //International conference on machine learning. PMLR, 2015: 1737 - 1746.
[18] GOEL A, LEE W R. Formal verification of an IBM CoreConnect processor local bus arbiter core [C] //Proceedings of the 37th Annual Design Automation Conference. 2000: 196 - 200.
[19] GUO K, SUI L, QIU J, et al. Angel-eye: A complete design flow for mapping cnn onto customized hardware [C] //2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). IEEE, 2016: 24 - 29.
[20] MAO W, WANG J, LIN J, et al. Methodology for efficient reconfigurable architecture of generative neural network [C] //2019 IEEE International Symposium on Circuits and Systems (ISCAS). IEEE, 2019: 1 - 5.

(上接第 161 页)

[6] 叶　娜，严昱欣，张　翔，等. 基于 BIM＋Cesium 三维可视化校园系统的设计与实现 [J]. 计算机测量与控制，2021，29 (1)：140 - 145.
[7] 喻凡坤，胡超芳，罗晓亮，等. 无人系统故障知识图谱的构建方法及应用 [J]. 计算机测量与控制，2020，28 (10)：66 - 71.
[8] 段妍羽，巩青歌，彭圳生. 基于数据挖掘的本体构建与重构技术研究 [J]. 计算机测量与控制，2017，25 (8)：244 - 247.
[9] 庄钰莹，熊　峰，吕　洋，等. 基于 GIS 的城市建筑群动力响应分析模型建模方法研究 [J]. 世界地震工程，2021，37 (4)：137 - 147.
[10] 陈　彬，王志英，甘　莹，等. 基于模糊运算的非结构化数据特征挖掘模型 [J]. 电子设计工程，2021，29 (21)：137 - 140.
[11] LI X, WEN Q H, LIN H, et al. Overview of CCKS 2020 Task 3: Named Entity Recognition and Event Extraction in Chinese Electronic Medical Records [J]. Data Intelligence, 2021, 3 (3): 376 - 388.
[12] HUANG Y Y, WANG W Y. Deep Residual Learning for Weakly- Supervised Relation Extraction [J]. 2017, 4 (2): 57 - 66.
[13] 谢　腾，杨俊安，刘　辉. 基于 BERT-BiLSTM-CRF 模型的中文实体识别 [J]. 计算机系统应用，2020，29 (7)：48 - 55.
[14] GAO W C, ZHENG X H, ZHAO S S. Named entity recognition method of Chinese EMR based on BERT-BiLSTM-CRF [J]. Journal of Physics: Conference Series, 2021, 1848 (1): 477 - 490.
[15] CUI Y M, CHE W X, LIU T, et al. Pre-Training with Whole Word Masking for Chinese BERT [J]. CoRR, 2019, abs/1906.08101.
[16] 徐红霞，李春旺. 科技文献内容知识点抽取研究综述 [J]. 数据分析与知识发现，2019，3 (3)：14 - 24.
[17] 江　旭，钱雪忠，宋　威. 残差 BiLSTM 句袋内与句袋间注意力机制的关系抽取 [J/OL]. 计算机工程：1 - 9 [2022 - 04 - 19]. https://kns.cnki.net/kcms/detail/detail.aspx?dbcode = CAPJ&dbname = CAPJLAST&filename = JSJC20211223003&uniplatform = NZKPT&v = g9BjGJf5ZLXAujjP - Ssj2om78rUrLoRG - sNhWyXlrpCFX28Bisnc3AwznGL_j1D.
[18] PRAVEENA RACHEL KAMALA S, JUSTUS S. Concept Relation Knowledge Visualization with CR Logic using Neo4j [J]. International Journal of Recent Technology and Engineering (IJRTE), 2019, 8 (4): 210 - 218.
[19] 穆　磊. 基于 BIM 的建筑消防自动审图研究 [D]. 北京：北京建筑大学，2020.
[20] 冯　姣，刘志勤，黄　俊，等. 基于 Three.js 的飞行仿真系统设计 [J]. 计算机测量与控制，2020，28 (2)：216 - 219.
[21] 建筑内部装修设计防火规范：GB 50222 - 2017 [S]. 2018.