文章编号:1671-4598(2022)12-0264-06 DOI:10.16526/j.cnki.11—4762/tp.2022.12.040 中图分类号:TP332 文献标识码:A

基于 ZYNQ 的深度学习卷积神经网络加速平台设计

刘之禹[1]，李 述[2]，王英鹤[2]
(1. 哈尔滨理工大学 计算机科学与技术学院，哈尔滨 150080；
2. 哈尔滨理工大学 电气与电子工程学院，哈尔滨 150080)

摘要：针对将各种卷积神经网络（CNN）模型部署在不同硬件端来实现算法加速时所遇到的耗费时间，工作量大等问题，采用 Tengine 工具链这一新兴的深度学习编译器技术来设计通用深度学习加速器，来将卷积神经网络模型与硬件后端高效快速对接；深度学习加速器的平台采用 ZYNQ 系列的 ZCU104 开发板，采用软硬件协同设计的思想，将开源的英伟达深度学习加速器（NVDLA）映射到可编程逻辑门阵列（FPGA）上，与 ARM 处理器构成 SoC 系统；NVDLA 整体架构规范，包含软硬件设计，采用 Tengine 工具链代替原来官方的编译工具链；之后在搭建好的 NVDLA 平台上实现 lenet-5 和 resnet-18 的网络加速，完成了 mnist 和 cifar-10 的数据集图像分类任务；实验结果表明，采用 Tengine 工具链要比 NVDLA 官方的编译工具链推理速度快 2.5 倍，并且量化工具使用方便，网络模型部署高效。

关键词：深度学习加速器；NVDLA；卷积神经网络；FPGA；硬件加速

Design of Deep Learning Convolutional Neural Network Acceleration Platform Based on ZYNQ

LIU Zhiyu[1], LI Shu[2], WANG Yinghe[2]
(1. School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China;
2. School of Electrical and Electrical Engineering, Harbin University of Science and Technology, Harbin 150080, China)

Abstract: Aiming at the problems of time consuming and heavy workload when various convolutional neural network (CNN) models are deployed on different hardware to achieve algorithm acceleration, by using the Tengine tool chain, an emerging deep learning compiler technology is used to design a general deep learning accelerator, which can efficiently and fast connect the network model and hardware backend. The platform of deep learning accelerator uses ZYNQ and ZCU104 development board, the idea of software and hardware co-design is used, the open source Nvidia Deep Learning Accelerator (NVDLA) is mapped on Field Programmable Gate Array (FPGA), and the SoC system is formed with ARM processor. The architecture of NVDLA completes the standard design, including software and hardware design, the Tengine tool chain is used to replace the original official compilation tool chain. After that, the network of lenet-5 and resnet-18 is realized on the built NVDLA platform, and the image classification task of the mnist and cifar-10 datasets is completed. Experimental results show that the inference speed of Tengine toolchain is 2.5 times faster than that of NVDLA official compilation toolchain, and the quantitative tools are easy to use, and the deployment of network model is efficient.

Keywords: deep learning accelerator; NVDLA; convolution neural network; FPGA; hardware acceleration

0 引言

当今时代人工智能技术发展迅速，深度学习也是其中的重点领域之一，尤其是卷积神经网络用途最广泛，图像分类，目标检测等图像处理技术都需要它[1]。随着算法的不断优化，其网络性能也在不断提升，但其参数量也在大幅度增长，如 VGG 系列，残差网络系列，有的甚至可以达到几亿参数，面对如此庞大的数据量就需要提升神经网络的推理速度[2]。深度神经网络的硬件部署情况各有优劣，其在 CPU 上运行速度过慢，在 GPU 上运行虽然速度很快，但其功耗过高，并且价格昂贵，不适合部署在移动端设备上。而 FPGA 由于其高度并行性，可重构性，功耗低，并且相对于专用集成电路成本低，因此可以较好地实现卷积神经网络的加速[2]。但目前大部分研究人员设计的传统的硬件加速器，一般只能针对某种特定的神经网络模型进行加速。目前各种神经网络算法迭代速度快，硬件设计卷积神经网络架构较难，周期长，容易跟不上算法的快速发展，并且当将不同的神经网络部署在不同的硬件设备上时，要进行适配，性能优化，写算子代码等大量重复性工作，及其耗费时间人力，因此通用深

收稿日期:2022-05-16； **修回日期**:2022-06-11。
基金项目:国家自然科学基金项目（51971086）；黑龙江省博士后科研启动基金（LBH-Q16118）；黑龙江省高校基础研究基金（LGYC2018JC004）。
作者简介:刘之禹(1998-)，男，黑龙江哈尔滨人，硕士生，主要从事 FPGA 深度学习加速器方向的研究。
通讯作者:李 述(1980-)，男，黑龙江齐齐哈尔人，博士，教授，博士生导师，主要从事机器学习，FPGA 开发方向的研究。
引用格式:刘之禹，李 述，王英鹤. 基于 ZYNQ 的深度学习卷积神经网络加速平台设计[J]. 计算机测量与控制，2022，30(12)：264-269.

度学习加速器应运而生。例如中科院和寒武纪研究的 Diannao 系列芯片[4]，谷歌的 TPU，英伟达的 NVDLA，都是性能不错的通用型硬件加速芯片，可对多种神经网络进行部署加速。但在这些芯片在设计出来之后，还要解决如何让不同的深度学习框架训练出来的网络都能与该芯片兼容的难题，这就需要深度学习编译器来解决，它可以作为神经网络模型与硬件之间的桥梁，当今应用较多的深度学习编译器框架有 TVM，GLOW 等[5]。

本设计针对 NVDLA 的官方深度学习编译器进行了优化，采用了 Tengine 这一开源的 AI 推理框架来代替官方的编译器，并且在软硬协同的 ZYNQ 平台上设计了一套完整的深度学习加速器平台。

1　NVDLA 介绍

NVDLA 是英伟达官方于 2017 年发布的一款开源深度学习加速器[6]，它可用于卷积神经网络的推理操作。它是一款高度可配置的 CNN 加速器框架，包含 full，large，small 版本，small 版本规模最小，仅支持 int8 类型的运算，适合部署在嵌入式移动端设备上，并且可以在它上面部署不同的神经网络模型。总体框架包含了软件与硬件设计两个部分，small 的系统结构如图所示。CPU 与 CSB 接口相连，负责配置 NVDLA 的内部寄存器数据来完成所要运行的操作，IRQ 接口为外部中断接口，NVDLA 会向 CPU 发送一些中断信号。DBBIF 接口与外部 DRAM 相连，负责从 NVDLA 中存入和取出输入输出数据，CPU 也与 DRAM 相连获取操作指令数据，NVDLA 系统框架如图 1 所示。

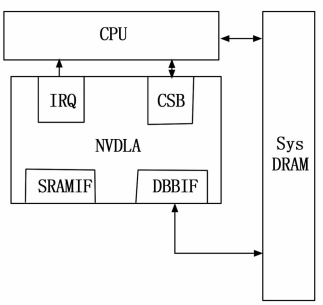

图 1　small 版本 NVDLA 系统框图

NVDLA 内部包含寄存器配置模块，与 CSB 接口相连，卷积运算模块，卷积缓冲运算模块，数据点操作激活函数模块（SDP），本地响应规范化模块（CDP），池化模块（PDP），以及 small 版本不支持的重塑 reshape 模块和桥接 DMA 模块。其中每个模块均与内存接口模块相连，每个模块都有一个寄存器配置接口和一个数据传送接口，图 2 为 NVDLA 内部数据流向模块图。数据从卷积模块处开始输入，一路送至 CDP 模块，最后通过内存接口输出数据。

NVDLA 框架整体包含软件与硬件两个层次，硬件即 NVDLA 的电路逻辑单元结构，是进行 CNN 推理运算的主要部分，主要通过硬件部分的高度并行性来加速神经网络的大量数据运算，其中包含了卷积运算，激活函数，池化，

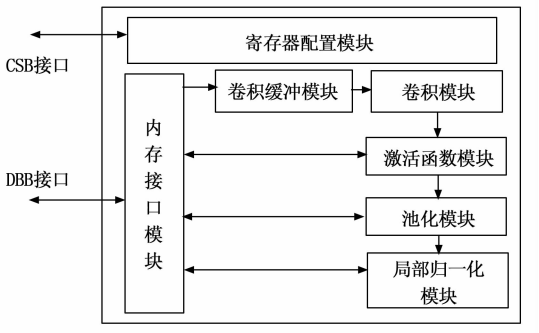

图 2　small 版本 NVDLA 内部模块

归一化 4 种算子。软件部分包括在 CPU 上运行的 compiler 与 runtime 两部分，compiler 即深度学习编译器部分，负责接收由 caffe 训练的 caffemodel 模型，将其转化为一种中间表示（IR）形式，一种计算图表示结构，然后 compiler 再对该图进行一些网络结构的图优化，再转化为硬件层面上的抽象语法树结构，最后生成硬件可以识别的文件格式，便于不同神经网络在硬件上部署。runtime 部分又分为 UMD 和 KMD 两个部分，UMD 为用户模式驱动程序，通过接受 compiler 生成的 loadable 文件并解析。然后发送给 KMD，即内核模式驱动程序，它接收到要进行推理的指定神经网络后，将硬件配置信息发送给硬件部分的配置寄存器，从存储器中取数据，执行所配置的寄存器操作，开始运行卷积网络并输出结果，NVDLA 软件栈工作流程如图 3 所示。

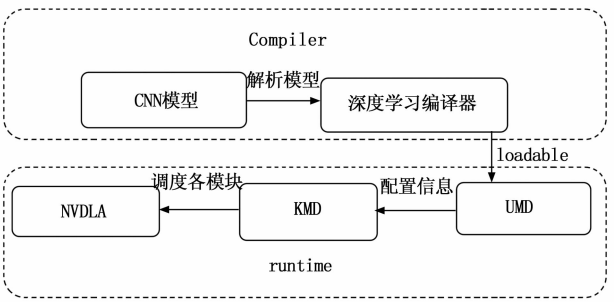

图 3　软件栈流程

NVDLA 硬件与软件部分代码完全开源，内容规范完整，尤其是它的硬件架构和软件编译器部分十分重要，为深度学习加速器的研究提供了极大的参考价值。

2　深度学习加速器

2.1　卷积运算特点

随着深度学习算法的不断优化，以及各种为了便携式嵌入式设备而研究的轻量型深度神经网络的出现，因此相应的深度学习加速器也成为了研究人员展开研究的热点。卷积神经网络更是深度神经网络中的主流方向，在深度学习加速器中，卷积相关的运算一般都占了一半以上的运行时间，绝大部分数据也都要进行卷积运算的过程，但卷积运算是十分规则的矩阵乘累加运算[7]，并且其中涵盖着大量的数据重用[1]，可以针对 CNN 的卷积运算的一致性而专门针对卷积运算设计相应的硬件电路结构[8]，因此可以用

FPGA 或 ASIC 这种并行度高的硬件电路来设计卷积运算框架，来加速卷积运算[7]。卷积运算分为直接卷积运算与快速卷积运算，本设计只针对直接卷积运算，即将一个卷积核在输入图像上按指定步长滑动，卷积核的参数与相应位置的图像进行乘累加运算得到输出特征图的一个像素点的值。快速卷积计算包括 FFT 和 Winograd 变换等方法[10]，通过优化卷积运算来减少耗费资源和功耗的乘法次数来提升卷积操作性能。卷积运算之后还会紧跟池化层也叫下采样层可用于降维并将特征信息进行压缩减小运算量，分为最大池化，最小池化，平均池化三重计算。然后有非线性层，它对卷积层或全连接层的输出施加激活函数来为网络引入非线性特征，来增强网络的学习能力，如 Relu，sigmoid 函数等。一般最后有全连接层，每一层的神经元都与下一层的每个神经元相连，全连接层的参数量一般占了网络中的大部分参数，全连接层的输出结果连接 softmax 函数，其结果为分类结果，卷积神经网络结构[10]如图 4 所示。

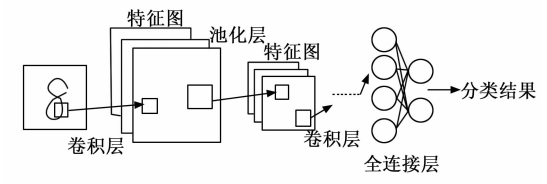

图 4 卷积神经网络结构

式（1）为卷积神经网络计算的公式，O 为输出特征图，I 代表输入特征图，W 为卷积核，b 为偏置，C 为卷积通道数，F 为卷积核的宽和高，S 为指定的步长。

$$O[n][x][y] = Relu(b[n] + \sum_{k=0}^{C-1}\sum_{i=0}^{F-1}\sum_{j=0}^{F-1} I[k][Sx+i]\times W[n][k][i][j] \quad (1)$$

2.2 深度学习编译器

由于 CNN 算法框架的增多，如 yolo，bert 系列等，它们的网络结构不同，算子种类也不尽相同，算子组合方式多种多样，并且应用场景也随之被不断发掘。随着各种嵌入式移动设备广泛应用，因此就需要将各种深度学习算法部署在各种硬件上[12]，如 CPU，GPU，FPGA，NPU（Neural Nework Unit）等。以前的方法是由人工针对各种网络结构去做优化，写出算子代码，并且还要针对不同的硬件后端，还要去做优化测试和适配，而这会带来极大的工作量[13]。因此就要想办法高效的将不同的神经网络可以用一种较为容易的方法去部署到不同的硬件上，让它们之间形成一道通用的桥梁，因此深度学习编译器就被提了出来[10]。

广义的编译器技术在计算机中应用广泛，它是一种程序，将某种高级语言如 Java，Go，Python 等，转换为一个中间表示，再把他翻译成等价含义的另一种目标语言程序，一般为机器语言或低级语言。深度学习编译器的过程与其类似，因为深度神经网络结构本身抽象化之后就是一个有向图结构，图的每一个节点代表一个卷积计算过程，两个节点之间包含着所对应的数据，因此深度学习编译器主要是对计算图相关的优化，包括前端，后端[5]。编译器前端先实现一个解析的功能，先接受由深度学习训练框架训练好的网络，再把它们转换成一个中间的表示形式，构建了一个有向计算图的表示形式。再对该有向图进行硬件无关的优化，比如算子融合，层融合等优化方法，可以将计算图的结构变的简化一些，减少神经网络的推理运行时间。然后将优化后的图 IR 送给后端，后端与部署目标硬件关系十分紧密，负责基于特定硬件的优化，如对硬件在运行推理操作时的数据存取，内存管理，计算单元调度等操作做出一系列指定，生成针对目标硬件体系结构的优化代码[11]。有时不同的硬件对应着不同的体系结构，如嵌入式设备一般采用 ARM 体系结构[14]，主机采用 X86 架构，以及现如今发展最为火热的 RISC-V 等[11]，因此针对不同体系结构的硬件后端，设计编译器时也要有所区分。

NVDLA 的官方也发布了相应的编译器，它只能接受 caffe 训练的模型，tensorflow 与 caffe 都是目前流行的深度学习框架，而 caffe 训练框架已逐渐被其余训练框架取代[15]。NVDLA 官方的编译器设计流程为，首先接受 caffemodel 后，通过解析器解析网络模型结构，生成 IR 中间表示，然后构建一个抽象语法树，也就是一个跟硬件架构无关的图优化结构，然后又根据硬件特点转化成一个跟硬件相关的的抽象语法树，该语法树包含了硬件的寄存器配置和内存分配信息，经过转化生成了一个针对硬件的图优化结构。与硬件无关的抽象语法树就是整个卷积神经网络各算子按顺正常序连接的形式。

2.3 Tengine 深度学习推理框架的应用

Tengine 是一个灵活，轻型的边缘 AI 计算框架，并且开源免费，它能够快速将深度学习前端模型部署到各种 CPU，GPU，NPU 平台，也具有深度学习编译器功能，可以进行模型转换，模型量化，多种端侧部署功能，可以高效地发挥端侧不同芯片的异构计算能力。它可以接收大部分训练框架如 caffe，tensorflow 等，通过序列化操作[5]，转换成抽象计算图形式，然后再对其进行图优化操作，统一保存为一种 tmfile 格式的文件。如果有量化需求的话可以使用 Tengine 自带量化工具进行 int8 的对称和非对称量化两种量化形式，直接导入网络模型和训练集即可完成量化。在后端进行部署时，Tengine 有着独特的自动切图机制，可将支持 NPU 的算子在 NPU 上运行，不支持 NPU 的算子切到 CPU 或其他设备上去运行，这充分发挥了设备的异构计算能力。对接后端时要注意，针对不同的后端设备，可以通过修改 Tengine 的接口函数文件，给硬件分配各节点和输入数据的内存以及切图模式，将已经转换为 Tengine 的与硬件无关的计算图变成与硬件相关的计算图结构。在不同的硬件后端上部署神经网络时，要添加该网络对应的算子，Tengine 官方针对不同训练框架所对应的算子与硬件相关的就是根据硬件架构将每个算子映射到 NVDLA 的不同模块上的各模块连接方式。例如卷积模块包含一个输入与权重相乘的乘法模块和其相乘结果与偏置相加的加法模块，这

样一个卷积算子就映射到了两个模块上，接着生成针对 NVDLA 硬件的优化代码，最后生成 loadable 数据流文件。然后是进行 runtime 阶段，负责硬件方面的配置和调度，它接收 loadable 文件，然后开始配置 NVDLA 的寄存器配置，内存分配，处理终端等配置。官方的编译器的缺点明显，比如现如今 tensorflow，pytorch 等已成为主流训练框架，而它只能接受 caffe 训练的框架，其次量化方面必须要下载 TensorRT，并且还要写 python 代码来进行操作，过程复杂。此外官方的支持的算子数量少，官方源码难以修改去添加其他算子，而以上缺点采用 Tengine 推理框架可以得到较好的解决，官方编译器的工作流程如图 5 所示[5]。

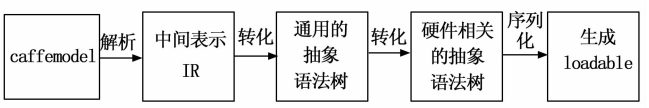

图 5　NVDLA 官方的 compiler 流程

大部分都已经添加，因此可以直接从源码文档中寻找对应算子自行添加。Tengine 在执行时，通过将待分类的图像输入给 Tengine 编译好的可执行分类程序，就可以输出分类结果。Tengine 系统框架如图 6 所示。并且 Tengine 还提供了灵活的调试功能，它可以输出每个节点的输入输出数据，以及每个算子中的参数和设备运行算子的执行时间，为调试后端硬件功能提供了极大便利。

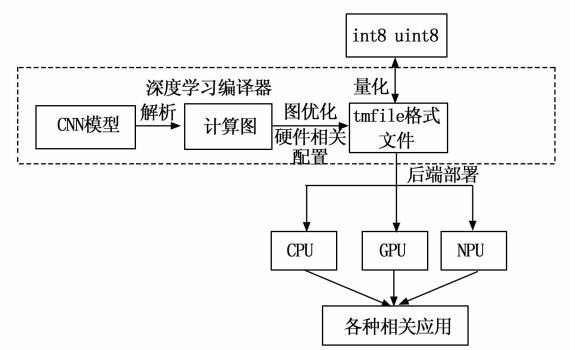

图 6　Tengine 系统框架图

本设计采用 Tengine 编译器代替官方的编译工具，即在原来官方编译流程中的通用抽象语法树处插入 Tengine 的编译器。在这之前仍然采用官方的编译器去生成中间表示 IR，然后在通用语法树处接入 Tengine 编译工具，这样可以采用 Tengine 编译器来对计算图结构进行切图，图优化等操作，使 NVDLA 得到由 Tengine 优化的结构，将原来生成 loadable 文件的过程去掉，直接将 tmfile 文件交给 runtime 部分。与后端设备对接时要实现以下几个功能，首先要初始化数据，为优化的计算图结构开辟内存，然后要把数据送入 NVDLA，以及取出 NVDLA 的输出数据，最后，释放空间内存，要写出以上函数来实现对接。Tengine 代替原来编译器流程的结构如图 7 所示[14]。

2.4　量化技术

神经网络的参数量化技术，目前也是压缩神经网络的

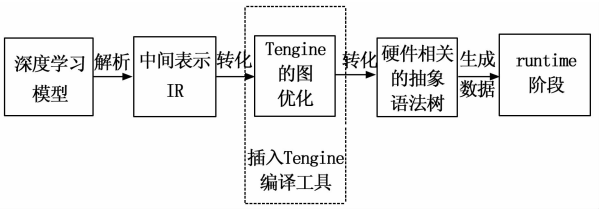

图 7　Tengine 框架编译流程

主流方法之一。神经网络训练时都是 32 位浮点数，量化是指将参数量化成 16 位 8 位 4 位等低精度格式，目前主流量化方案是 int8 量化，这样可以降低设备对于网络模型的存储压力[16]，大量减少了数据访存次数和计算量同时降低了功耗[17]。并且也有研究表明，由于 CNN 对于噪声不敏感，因此 int8 量化后的网络与之前浮点 32 位的网络精度相差无几[18]，并且硬件比较适合进行定点运算[19]。量化方案包含对称量化与非对称量化，对称量化是将待量化数据绝对值的最大值映射到新数据范围内的最大值，非对称量化是指将待量化数据的最大值和最小值映射到新范围内的最大值最小值，本设计针对对称量化展开研究。

量化的过程为先计算数据的最大最小值，计算出缩放因子 scale，然后得到它的数据分布，最后对其进行数据截断。Tengine 的量化原理与 TensorRT 类似，采用一种反量化方法，即将输入数据和权重进行 int8 量化，即把它们与 scale 相乘，偏置值不量化，然后输入数据与权重相乘，之后再转换成浮点 32 位，最后再与偏置值相加得到结果[20]。Tengine 的量化工具经过官方研究表明其效果好于大部分 NPU 自带的量化工具。Tengine 已经根据量化原理设计好量化工具，在工具库里可以找到，直接输入要量化的 tmfile 格式的文件模型执行量化程序来量化，量化时还需要加上训练集中的一部分数据来进行量化时的校准，使得量化结果更为准确。

3　实验结果与分析

3.1　实验环境介绍

本实验硬件平台采用赛灵思的 ZYNQ UltarSCALE+系列的 ZCU104 开发板搭载双核 ARM A53 处理器，PC 端使用 vivado 和 petalinux 开发软件和 pycharm 软件进行神经网络训练，采用 tensorflow 框架训练网络，PC 端的中央处理器型号为 i5-1035G1。

使用 mnist 手写数字训练集训练 lenet-5 网络和 cifar-10 训练集训练 resnet-18 网络，将训练好的模型分别采用 NVDLA 官方的深度学习编译器和 Tengine 的编译器去将该模型转化为 NVDLA 硬件可以接受的形式，在开发板上对测试集进行推理，最后分析实验结果。

3.2　NVDLA 在 FPGA 上的映射

官方发布的 NVDLA 是针对 ASIC 设计的，现要将其映射到 FPGA 上。首先要将 NVDLA 中的 RAM 结构替换为 FPGA 上的 BRAM 资源，因为官方的代码描述的 RAM 是 RTL 行为级的，直接映射到 FPGA 上的话，会使的 FPGA

上的查找表资源来实现 RAM，会浪费大量资源。然后还要关闭门控时钟，因为 FPGA 上的时钟资源与 ASIC 不同，如果采用 ASIC 的门控时钟行为，会导致保持时间违例[13]。

图 8 和图 9 为 NVDLA 的寄存器配置部分仿真图和数据传输接口部分仿真图。NVDLA 作为从机，与其进行数据交互的 CPU 作为主机。如图 8 所示，当输入信号 valid 与 NVDLA 的给出的输出信号 ready 同时为高时，代表握手信号成功，write 写信号被拉高，寄存器地址和要写入的数据发送给 NVDLA。如图 9 所示，当 valid 信号和 ready 信号同时为高时，数据写入 NVDLA 的内部存储单元内。

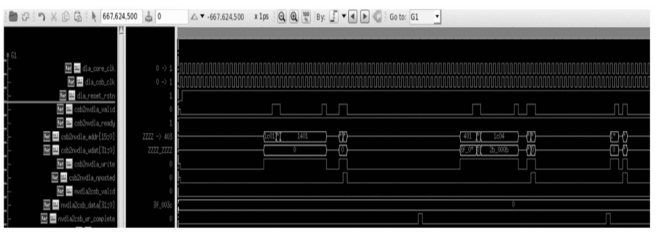

图 8 NVDLA 寄存器配置接口仿真图

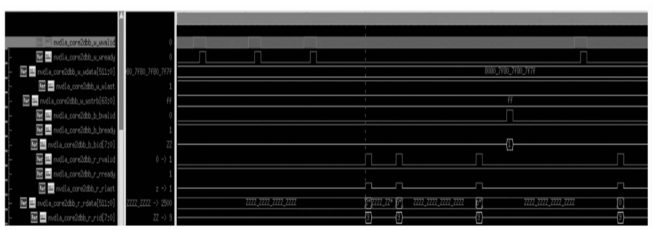

图 9 NVDLA 数据传输接口仿真图

例化 NVDLA 模块后还需要将它与 CPU 相连，ZCU104 上的 ARM 处理器可当作 CPU 使用。NVDLA 的 CSB 接口上加一个 CSB 转 APB 协议的转接器模块，并且想要与 ARM 通信只能采用 AXI 协议，因此要设计一个 APB 转 AXI 的桥接模块将 NVDLA 与 ARM 相连。还要将 csb _ clk 与 core _ clk 两个时钟短接，它们负责寄存器的读写最后得出，时钟频率定为 100 Mhz，时序符合要求，FPGA 的资源占用情况见表 1。还要生成 xsa 文件，用于在 petalinux 中定制操作系统。

表 1 资源使用情况

资源	消耗量	总量	消耗占比/%
LUTs	78 023	230 400	34
Registers	89 547	460 800	19
BRAM	64	312	21
DSPs	32	1 728	2

与 NVDLA 的加速任务，异步电路的存在使得 vivado 难以进行静态时序分析，因此要把它转换成同步时序电路。将各个模块集成连线，设计成完整的 SOC 系统，进行逻辑综合布局布线，得到时序符合要求的设计，图 10 为 NVDLA 总体电路结构图。

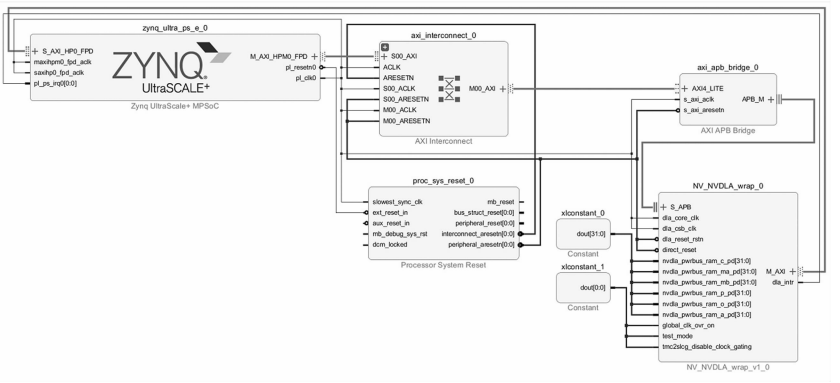

图 10 NVDLA 总体电路结构图

3.3 网络模型训练

在 pycharm 软件上采用 tensorflow 框架训练 lenet-5 和 resnet-18 两个网络。mnist 手写数字数据集训练集 60 000 张图片，测试集 10 000 张图片，cifar-10 数据集训练集有 50 000张图片，测试集有 10 000 张图片，resnet-18 网络比 lenet-5 复杂。训练模型的精确度如表 2 所示，均为 32 位浮点数形式，tensorflow 训练模型输出为 pb 格式文件。

表 2 PC 端训练模型精确度

网络模型	数据集	测试集准确率/%
lenet-5	mnist	99.7
resnet-18	cifar-10	91.1

3.4 开发板推理网络模型对比

为了在开发板上完整设计 NVDLA 平台，还需要 ZYNQ 上移植一个 ubuntu16.04 的操作系统，存放在 SD 卡上，插到开发板卡槽上，打开开发板启动开关即可运行 Linux 系统。同时还要将由 vivado 生成的 xsa 文件导入 petalinux 软件中，定制操作系统镜像文件，将 NVDLA 设计增加到了系统的设备树中，这样就可以在 Linux 系统中使用 NVDLA 外设。整个平台的架构如图 11 所示，通过键盘输入 Linux 系统下的各种指令，在显示屏上查看结果。

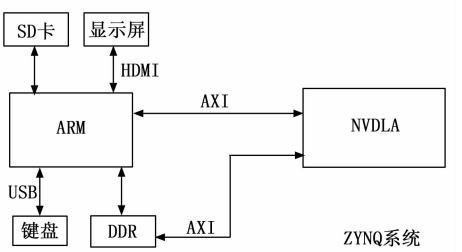

图 11 系统架构图

采用 NVDLA 官方编译器与 Tengine 编译器分别优化转换以上两个网络，进行 int8 量化，将测试集的图片输入，分析它们的运行时间后再将两个网络在 CPU 上运行，得到数据分析对时间和准确率，然后并将它们对比，结果如表 3 所示。

表 3　不同方案数据对比

推理平台	NVDLA 官方编译器	NVDLA Tengine 编译器	i5-1035G1
工作频率	500 MHz+100 MHz	500 MHz+100 MHz	1.19 GHz
lenet5 单帧时间/ms	10.88	11.20	14.33
resnet18 单帧时间/ms	36.30	14.30	130.37
功耗/W	3.87	3.87	25.20

如表 3 对比可知，由于 lenet-5 网络简单，因此使用不同编译器的网络优化效果不明显，对比使用 CPU 推理网络有大约 30%的加速效果。对于 resnet-18 网络，由于该网络较为复杂，因此可以明显看出采用 Tengine 的编译器优化结构效果要好于采用官方编译器，速度增加了约 2.5 倍，对比 CPU 增加了约 3.7 倍，并且功耗也比 CPU 低约 6.5 倍。

量化后的测试集准确率如表 4 所示，由于采用官方编译器与 Tengine 编译器的量化方案不同，因此准确率会有一些差别，与表 2 中原来的 32 位浮点数形式的准确率对比，误差在合理范围之内。

表 4　准确率对比　　%

编译器方案	官方编译器	Tengine 编译器
cifar-10 准确率	80.7	82.3
lenet-5 准确率	97.5	97.7

4　结束语

针对多种卷积神经网络实现硬件加速时，在不同硬件后端部署时遇到的效率低的问题，提出了采用 Tengine 这一深度学习推理框架的方案来解决该问题，在 ZYNQ 平台上部署了完整流程的深度学习加速器。又对深度学习编译器和 NVDLA 特性结构进行研究，将 Tengine 这一框架应用于 NVDLA 上，使用 Tengine 的编译器来将 CNN 模型与 NVDLA 硬件部分对接。通过实验在 ZCU104 开发板上搭建了 NVDLA 这一加速器平台，实现了 lenet-5 和 resnet-18 两个网络的图像分类任务。实验结果表明使用 Tengine 工具链的深度学习加速器可以支持接受多种深度学习训练框架模型，并且其图优化结果要好于官方的深度学习编译器，使得卷积神经网络推理速度更快，量化工具等其余功能使用起来也很方便，对于研究卷积神经网络的落地应用有一定的研究意义。

参考文献：

[1] ZHANG C，SUN G，FANG Z，et al. Caffeine：toward uniformed representation and acceleration for deep convolutional neural networks [J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2018，38 (11)：2072－2085.

[2] QIU J，SONG S，WANG Y，et al. Going deeper with embedded FPGA platform for convolutional neural network [C] //the 2016 ACM/SIGDA International Symposium，ACM，2016：26－35.

[3] GILAN A A，EMAD M，ALIZADEH B. FPGA-based implementation of a real-time object recognition system using convolutional neural network [J]. IEEE Transactions on Circuits and Systems II：Express Briefs，2019：755－759.

[4] CHEN Y，CHEN T，XU Z，et al. DianNao family：energy-efficient hardware accelerators for machine learning [J]. Communications of the ACM，2016，59 (11)：105－112.

[5] LI M，LIU Y，LIU X，et al. The deep learning compiler：a comprehensive survey [J]. IEEE Transactions on Parallel and Distributed System，2021，32 (3)：708－727.

[6] NVIDIA corporation. hardware manual [EB/OL]. [2021-07-02]. http://nvdla.org/hw/contents.html.

[7] SZE V，CHEN Y H，YANG T J，et al. Efficient processing of deep neural networks：atutorial and survey [J]. Proceedings of the IEEE，2017，105 (12)：2295－2329.

[8] KWON H，SAMAJDAR A，KRISHNA T. MAERI：Enabling flexible dataflow mapping over DNN accelerators via reconfigurable interconnects [J]. ACM SIGPLAN Notices，2018：461－475.

[9] 赵　烁，范　军，何　虎. 基于 FPGA 的 CNN 加速 Soc 系统设计 [J]. 计算机工程与设计，2020，41 (4)：939－944.

[10] ZHU C，HUANG K，YANG S，et al. An efficient hardware accelerator for structured sparse convolutional neural networks on FPGAs [J]. IEEE Transactions on Very LargeScale Integration (VLSI) Systems，2020，28 (9)：1953－1965.

[11] 林振钰，张志杰，刘佳琪. 基于 ZYNQ 的高清图像显示及检测系统设计 [J]. 计算机测量与控制，2021，29 (2)：30－39.

[12] 万紫微. 面向物端专用人工智能芯片的实时人脸识别技术 [D]. 北京：中国科学院大学，2020.

[13] 张芳芳. 基于 CNN 加速器的深度学习编译器设计与实现 [D]. 西安：西安电子科技大学，2021.

[14] LeiWang：ZYNQ-NVDLA [EB/OL]. [2021-08-29]. https://github.com/LeiWang1999/ZYNQ-NVDLA.

[15] 李　林，张盛兵，吴　娟. 基于深度学习的实时图像目标检测系统设计 [J]. 计算机测量与控制，2019，27 (7)：15－19.

[16] HAM T J，JUNG S J，KIM S，et al. accelerating attention mechanisms in neural networks with approximation [C] //2020 IEEE International Symposium on High Performance Computer Architecture (HPCA). IEEE，2020：328－341.

[17] WU J，CONG L，WANG Y，et al. Quantized convolutional neural networks for mobile devices [C] //Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016：4820－4828.

[18] 周立君，刘　宇，白　璐，等. 使用 TensorRT 进行深度学习推理 [J]. 应用光学，2020，41 (2)：337－341.

[19] 周志远. 基于 RISC-V 和 NVDLA 的人工智能芯片开发平台研究 [D]. 杭州：杭州电子科技大学，2020.

[20] FARSHCHI F，HUANG Q，YUN H. Integrating NVIDIA deep learning accelerator (NVDLA) with RISC-V soc on fireSim [J]. 2019 2nd Workshop on Energy Efficient Machine Learning and Cognitive Computing for Embedded Applications (EMC2)，2019：21－25.