**密级： 保密期限：**

xm 拷贝

**硕士学位论文**



**题目： 面向TTA架构ASIP设计的**

**深度神经网络优化**

**学 号：**

**姓 名：**

**专 业： 信息与通信工程**

**导 师：**

**学 院： 信息与通信工程**

**2019年1月11日**



**A Thesis for Master Degree**

**TITLE: OPTIMAL DESIGNS OF DEEP NEURAL NETWORKS ON TTA BASED ASIP**

|  |  |
| --- | --- |
| **Student No.:** |  |
| **Author:** |  |
| **Major:** | **Information and Communi-**  **cation Engineering** |
| **Supervisor:** |  |
| **School:** | **Information and Communi-**  **cation Engineering** |

**Jan. 11th, 2019**

独创性（或创新性）声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名： 日期：

关于论文使用授权的说明

本人完全了解并同意北京邮电大学有关保留、使用学位论文的规定，即：北京邮电大学拥有以下关于学位论文的无偿使用权，具体包括：学校有权保留并向国家有关部门或机构送交学位论文，有权允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，有权允许采用影印、缩印或其它复制手段保存、汇编学位论文，将学位论文的全部或部分内容编入有关数据库进行检索。（保密的学位论文在解密后遵守此规定）

本人签名： 日期：

导师签名： 日期：

面向TTA架构ASIP设计的深度神经网络优化

摘 要

近年来，深度神经网络在计算机视觉、自然语言处理等诸多现实任务中取得了优异的成绩，并得到广泛地应用。依托电子芯片技术的发展，神经网络的结构被设计得更深更大，以应对更加复杂和抽象的场景。随之而来的问题是深度神经网络在训练和推理时需要进行大规模的浮点运算，对存储资源的占用和设备功耗的需求都很高。开发专用指令集处理器（Application Specific Instruction Set Processor，ASIP）是解决该问题的途径之一，它既能够提供专用集成电路级别的高性能和低功耗，又能够提供处理器级别的指令集灵活性。传输触发架构（Transport Triggered Architecture，TTA）是一种ASIP的系统架构，它利用数据传输来触发相应功能单元的具体操作的思想对于神经网络数据传输频繁而数据操作相对简单的特点较为适用。

卷积神经网络是最有代表性和最通用的深度神经网络，本文的工作主要是面向TTA架构ASIP设计的卷积神经网络进行优化。具体的工作主要包含以下两方面的内容。其一，本文针对神经网络的硬件实现的需求，提出了一套端到端完整的8比特量化方案。该方案根据权重、激活值和梯度的各自特点，为它们制定了不同的量化策略，并提出了一种近似的批量归一化算法。其二，本文提出了一种基于乘法结合律量化卷积的方案，设计了新的卷积核与卷积运算的表达方式，提供了符合量化卷积特性的循环展开、数据交换方案，并对量化卷积加速器的整体结构进行了描述。

本文提出的量化方案在多数据集、多模型结构的实验中都取得了与全精度网络相当的精度，领先于部分当下常用的量化方案。本文提出的量化卷积方案将查找表资源融合到卷积运算中，缓解了有限的乘法器资源对于神经网络中并行度的限制。

**关键词：**深度神经网络 量化神经网络 卷积运算 FPGA

OPTIMAL DESIGNS OF DEEP NEURAL NETWORKS

ON TTA BASED ASIP

ABSTRACT

In recent years, deep neural networks have achieved state-of-the-art results in many practical tasks, such as computer vision and natural language processing, and have been widely applicated. With the development of electronic chip technology, the structures of neural networks are designed to be deeper and larger to cope with more complex and abstract scenarios. The following problem is that deep neural networks need large-scale floating-point operations during the training and inference stages, leading to the high demand for storage resources and power consumption. Developing Application Specific Instruction Set Processors (ASIPs) is one of the approaches to solve this issue, which can not only bring high performance and low power consumption, but also own the flexibility of instruction sets. Transport Triggered Architecture (TTA) is an ASIP system architecture, whose core idea is to trigger the specific operations of the corresponding function units through data transmission. This characteristic fits in with the need of frequent data transmission and relatively simple data operations that happens in neural networks.

Convolutional neural networks are the most representative and wildly-used deep neural networks. The principal work of this paper is to optimize convolutional neural networks on TTA based ASIP, which mainly includes two aspects of work. Firstly, aiming at the requirements of hardware implementation of neural network, this paper proposed an end-to-end 8-bit quantization scheme. Different quantization strategies were designed according to the characteristics of weights, activations and gradients, and an approximate batch normalization algorithm was also put forward in the quantization work. Secondly, this paper proposed a quantized convolution scheme based on multiplication combining law and designed a novel representation of convolution kernels and convolution operations. The loop unrolling and data exchange schemes that accord with quantized convolution characteristics were provided and the overall structure of quantized convolution accelerator was descripted.

In the experiments on multiple datasets and multiple network structures, the proposed quantization scheme achieves the comparable accuracy as the full-precision neural networks, ahead of some frequently-used schemes. And the proposed quantized convolution scheme integrates look-up table resources into convolution operations, alleviating the parallel restriction in neural networks caused by the limited number of multipliers.

**KEY WORDS:** deep neural networks quantized neural networks convolution operation fpga

目录

[第一章 绪论 1](#_Toc535273306)

[1.1 课题研究背景 1](#_Toc535273307)

[1.2 课题研究内容 3](#_Toc535273308)

[1.3 论文组织结构 3](#_Toc535273309)

[第二章 相关理论介绍 5](#_Toc535273310)

[2.1 卷积神经网络的模型结构发展 5](#_Toc535273311)

[2.1.1 LeNet-5与卷积神经网络的起源 5](#_Toc535273312)

[2.1.2 AlexNet与深度学习的崛起 5](#_Toc535273313)

[2.1.3 不断刷新纪录的新模型结构 6](#_Toc535273314)

[2.2 卷积神经网络的常用结构 8](#_Toc535273315)

[2.2.1 卷积层 8](#_Toc535273316)

[2.2.2 归一化层 9](#_Toc535273317)

[2.2.3池化层 10](#_Toc535273318)

[2.2.4 激活层 10](#_Toc535273319)

[2.3 神经网络的优化技术 11](#_Toc535273320)

[2.3.1 神经网络的压缩 11](#_Toc535273321)

[2.3.2 神经网络的加速 12](#_Toc535273322)

[2.4 TTA架构的主要特点[29] 13](#_Toc535273323)

[第三章 神经网络的量化方案设计 15](#_Toc535273324)

[3.1 量化技术调研 15](#_Toc535273325)

[3.1.1 二值化连接[16]与二值化权值网络[17] 16](#_Toc535273326)

[3.1.2 二值化神经网络[18]与异或网络[17] 16](#_Toc535273327)

[3.1.3 三值化权值网络[19] 17](#_Toc535273328)

[3.1.4 DoReFa-Net[20]与梯度的量化 17](#_Toc535273329)

[3.1.5 其它低比特量化网络 18](#_Toc535273330)

[3.2 本文量化方案的整体思路 18](#_Toc535273331)

[3.3 本文量化方案的具体实现 20](#_Toc535273332)

[3.3.1 权重的对称仿射量化 20](#_Toc535273333)

[3.3.2 近似的批量归一化 22](#_Toc535273334)

[3.3.3 激活层与激活值的量化 24](#_Toc535273335)

[3.3.4 变精度的梯度量化 25](#_Toc535273336)

[第四章 神经网络的加速方案设计 27](#_Toc535273337)

[4.1 基于乘法结合律的量化卷积 27](#_Toc535273338)

[4.1.1 量化神经网络的前向传播 27](#_Toc535273339)

[4.1.2 量化卷积的原理 28](#_Toc535273340)

[4.1.3 量化卷积实现方式的探究 31](#_Toc535273341)

[4.1.4 量化卷积核的表示 33](#_Toc535273342)

[4.2 量化卷积的实现 35](#_Toc535273343)

[4.2.1 卷积核的分块 35](#_Toc535273344)

[4.2.2 卷积运算的循环展开 36](#_Toc535273345)

[4.2.2 前向传播的数据交换 38](#_Toc535273346)

[4.3 量化卷积加速器的整体结构 40](#_Toc535273347)

[第五章 实验结果与分析 43](#_Toc535273348)

[5.1 近似批量归一化的实验 43](#_Toc535273349)

[5.2 量化方案的实验 44](#_Toc535273350)

[5.2.1 SVHN数据集上的实验 45](#_Toc535273351)

[5.2.2 CIFAR-10数据集上的实验 45](#_Toc535273352)

[5.2.3 ImageNet数据集上的实验 46](#_Toc535273353)

[5.3 量化卷积运算的实验 47](#_Toc535273354)

[5.3.1 实验配置 47](#_Toc535273355)

[5.3.2 资源占用的对比 48](#_Toc535273356)

[5.3.3 片上功率的对比 49](#_Toc535273357)

[第六章 总结与展望 51](#_Toc535273358)

[6.1 本文工作总结 51](#_Toc535273359)

[6.2 不足与展望 52](#_Toc535273360)

[参考文献 53](#_Toc535273361)

[致谢 57](#_Toc535273362)

[攻读学位期间发表的学术论文目录 58](#_Toc535273363)

第一章 绪论

1.1 课题研究背景

在2012年，AlexNet[1]被提出，一举获得ILSVRC[2]（ImageNet Large Scale Visual Recognition Competition，ImageNet大规模视觉识别竞赛）的第一名，显示出深度学习在视觉任务方面的巨大潜力，也吸引了越来越多的研究者投入到这相关工作中。近年来，深度学习在图像分类、语音识别、自然语言处理等诸多领域里得到了蓬勃发展，在解决高级抽象认知问题上获得了令人耳目一新的成果，甚至在许多任务中超越了人类自身的性能。

深度学习在过去短短的几年间获得巨大进步的原因可以从许多角度来分析。从研发投入的角度出发，深度学习是当前学术界和工业界不遑多让的热点，吸引众多科研院所和国内外顶级高科技公司大量的投入，并由此产生了丰硕的成果。从算法改进的角度出发，更先进的深度学习框架被提出，更精细的模型网络结构被广泛地使用，诸如批处理算法和损失函数的设计等神经网络的实现细节被迭代优化，使得模型能够更准确地学习和反映复杂原始数据的特性。从数据增强的角度出发，越来越大的数据集被公开，基于VAE或GAN的生成方法也日渐成熟，为数据增强的实现提供了更多的选择，使得能够用于神经网络训练的数据更加丰富多样，这对于基于大数据的神经网络模型的训练大有裨益。从硬件发展的角度出发，高性能的通用处理器如NVIDIA公司的GPU（Graphics Processing Unit，图像处理单元）的内部存储空间不断扩大，并行计算能力日益增强，使得大型网络结构的部署成为了可能。

如果要对上文中提到的种种原因进行总结，深度学习的近年来取得的斐然成果应该最大程度地归功于深度神经网络模型规模的扩大。它是由更鲁棒的网络结构，更大的内存空间，更强的计算能力，更大的数据集共同驱动的。得益于更大的网络在复杂任务中能够实现更高的精度，深度神经网络的模型规模在将来势必会继续保持稳定而高速的增长，而如何完成它们的训练或推理则是深度学习领域的重要课题。

在讨论深度学习算法的通用处理器实现时，基于CPU（Central Processing Unit，中央处理器）的方案虽然依旧占据一席之地，但是已经渐渐边缘化。导致这一结果的根本原因在于，CPU的内部结构中控制器和寄存器资源丰富，而逻辑单元资源相对有限，因此其数据处理的能力相比GPU有明显的差距，在神经网络模型越来越大，需要的计算越来越多的大背景下，CPU无法提供令人满意的训练与推理速度。GPU的内部结构与CPU有很大不同，它的逻辑单元的规模庞大，十分适用于大量数据并行计算的场景。同时GPU针对深度学习进行了专门的优化，使其具备了高带宽、高主频、高并行性的特点。因此，复杂的神经网络更多地借助GPU来完成训练。然而，GPU的功耗较高，单个GPU的功耗往往要高于同期主流的CPU功耗，对比FPGA等硬件平台，更是达到了几十倍甚至上百倍的差距。因此，虽然复杂的神经网络的训练多借助于GPU实现，但是GPU的应用局限于学术研究等少数场景中，对于日常生活更为常见的移动端设备而言，基于GPU的深度学习方案的落地存在实际的困难。在这些场景中，深度学习的应用更多地发生在推理阶段，该阶段对于大规模并行计算的需求要远小于训练阶段。因此，寻找通用处理器之外的具有低功耗，高性能，低延时特点的加速深度神经网络的硬件实现是当下深度学习领域的热门课题之一。

利用ASIC进行神经网络的前向推理是该课题下的一个研究方向。这类方案属于纯硬件加速方案，通过为某种具体的深度学习算法定制电路，实现最高的专用性，最低的设备功耗和最小的芯片面积。虽然ASIC方案有诸多优势，但是它最突出的问题在于它是一种一次性的方案，只适用于固定的算法。一旦算法发生了些许的改动，ASIC就需要被重新设计，因此它的灵活性是最差的。随着深度学习的快速发展，神经网络的算法日新月异，而ASIC芯片从设计到制造，开发周期往往在一年左右，不但开发陈本居高不下，硬件侧也难以跟上算法侧发展的脚步。种种原因导致ASIC目前并没有被广泛地用于深度学习的实现中去。利用FPGA（Field－Programmable Gate Arra，现场可编程门阵列）进行深度学习开发的设计思路恰好解决了ASIC专用性太强的问题。虽然同等条件下FPGA的运行速度要慢于ASIC，但是FPGA的设计更加灵活多变，通过内部逻辑结构的定义就能够实现硬件加速器的功能，是一种即插即用的解决方案。当算法发生变化时，FPGA只需通过代码的相应更改就能够完成硬件的更新。此外，FPGA不依赖于冯诺依曼结构，一个计算结果无需在主存储器临时保存便可以进行下一步操作，因此FPGA不仅对于存储器带宽需求比CPU或GPU要低得多，还拥有流水处理的优势。

本文提出的ASIP是一种介于GPU与ASIC之间的解决方案，它比GPU更加专用，主要面向深度学习模型里面的通用计算类型，比如卷积、池化等，通过对特殊的功能单元的设计，更好地支持单指令多数据流的实现；它又比ASIC更加灵活，当算法发生变化的时候，ASIP可以通过编程重构，重新修正函数，实现新的算法，而无需进行整个芯片的重新开发。对比FPGA，由于相应的功能单元已经实现，因此ASIP不需要进行底层硬件语言的重新编辑，编程相对简单。

总结上述的解决方案，CPU、GPU、FPGA、ASIP、ASIC，它们的通用性和灵活性依次减弱，但是专用型和高效性却依次增强。某种程度上说，基于ASIP的解决方案是它们中高效性和灵活性之间平衡点。本文研究的便是针对ASIP实现的深度神经网络的优化。

1.2 课题研究内容

本文的工作围绕“面向TTA架构ASIP设计的深度神经网络优化”这一目标展开。在本文之外的工作中，与作者同一实验室的XXX同学完成了“基于TTA的大型卷积神经网络处理器架构设计”，因此，本文的工作更多地聚焦在深度神经网络的优化中。具体来说，本文的主要工作可以分为两点。

其一，本文设计了一套端到端的可以从零训练的深度神经网络的8比特量化方案。量化是神经网络在FPGA等硬件平台上部署时的通用方法，也是实现深度神经网络的ASIP必不可少的操作。与早前的量化方案不同，本文根据神经网络中各成分的不同特点，为它们制定了针对性的量化策略。该方案的创新点包括权重的对称仿射量化，近似的批量归一化与以及变精度的梯度量化。在作者有限的知识范围内，这些量化策略都是第一次被提出的。本文对量化方案在通用处理器上进行了验证，在对不同数据集、不同模型结构进行的实验中，本文提出的量化方案均能够达到与全精度模型相当的准确率。

其二，本文提出了一种基于乘法结合律的量化卷积的方案。卷积运算是卷积神经网络中最常用也是占用运算资源最多的运算，与传统的卷积方式不同，本文提出的卷积方式，发掘了量化操作为卷积运算带来的潜在的优化空间，设计了新的卷积核以及卷积运算的表达方式，同时也对该方案下的循环的分块和展开都进行了相应的优化。本文对量化卷积的方案进行了详尽的原理阐述和理论推导，并通过硬件仿真实验证明，通过使用一部分查找表资源，量化卷积能够缓解有限的DSP乘法器对于神经网络中并行计算的限制，在同等的硬件条件下，较大限度地提升了系统的并行度，从而达到时间复杂度与空间复杂度之间更好的平衡。

1.3 论文组织结构

根据本文的工作内容，本文共有六个章节。文章的组织结构安排如下：

第一章：绪论。本章简要地介绍了与本文相关的课题背景，总结性地交代了本文的两点主要研究工作，并简单罗列了本文的组织结构。

第二章：相关技术研究。本章对近年来相关领域的一些工作与成果进行了回顾与总结，主要包括神经网络的模型结构发展，卷积神经网络的常用结构与神经网络的加速技术。

第三章：神经网络的量化方案。本章详细描述了本文提出的量化方案，包括对权重、激活值、梯度的具体量化策略以及一种近似的批量归一化算法。

第四章：神经网络的加速方案。本章详细介绍了本文提出的加速方案，重点说明了量化卷积的原理与实现，并给出了相应的循环分块与循环展开的策略。

第五章：实验结果与分析。本章分别展示了量化方案与加速方案的实验结果，并对其进行了必要的分析。

第六章：总结与展望。本章对本文的工作成果进行了全面的总结，对本文的不足之处进行了反思，并提出了后续的改进方案。

第二章 相关理论介绍

2.1 卷积神经网络的模型结构发展

本文的课题背景中提到，深度神经网络在过去的几年中取得了巨大进步很大程度上来自于深度神经网络的模型结构的改良与优化。因此，本节将以具有代表性的卷积神经网络的模型结构为线索，概括性地介绍卷积神经网络的发展，回顾它的历史，展望它的未来。

2.1.1 LeNet-5与卷积神经网络的起源

虽然神经网络的历史可以追溯到19世纪的40年代，但是彼时缺少完整的理论基础，而受限于运算能力的不足，神经网络实验的开展十分困难，因此没有诞生有说服力的实验结果。目前业内的一个共识是，1994年被提出的LeNet-5[3]是当前的神经网络的起源。作为最早的卷积神经网络之一，LeNet-5最初被用于手写数字识别。虽然从今天的视角看，它的网络架构非常基础，面向的任务也非常简单，但是LeNet-5能够利用极其有限的资源进行完成神经网络的训练与推理已属不易。更重要的是，LeNet-5提出了许多先驱性的概念与算法，其中最大的贡献在于它提出了卷积神经网络中卷积层、池化层、非线性层这三层序列，并解释了各层操作的作用：卷积操作用于提取特征空间，池化操作用于降采样，非线性层增强网络的表达能力。这一框架对深度学习的整个工作都有巨大的意义，整体结构一直沿用至今。

LeNet-5在手写数字识别任务中的成功并没有引起深度神经网络研究的热潮，从上世纪90年代到本世纪初的十多年里，神经网络处于一个蛰伏期，发展比较缓慢。然而，在此期间其他行业的快速发展，为神经网络后来的爆发埋下了伏笔。其一是随着拍照手机与数码相机的流行，越来越多数字图像被生产，对数字图像的加工与理解的需求与日俱增；其二是通用处理器的快速发展，使得大规模的神经网络在计算机上的部署成为了可能。

2.1.2 AlexNet与深度学习的崛起

2012年AlexNet以明显的优势战胜传统的视觉算法，夺得该年度ILSVRC的冠军，是标志着深度学习领域崛起的事件。因此，2012年也被成为“深度学习的元年”。

ILSVRC是ImageNet大规模视觉识别竞赛的简称，它主要评价算法在大尺度上对物体检测和图像分类的效果。竞赛的一个主要目的是通过利用大量的人工标记训练数据，激励研究者们来比较他们的算法在多种多样的物体检测上的效果；另一个主要目的是检验计算机视觉技术在大尺度图像的检索和标注方面的进步。由于ILSVRC作为开展较早与参与人数众多的视觉竞赛，因此它的名次一直是衡量一个研究机构或企业技术水平的重要标尺。

图2-1展示了AlexNet的8层模型结构。AlexNet在LeNet-5的基础之上，设计了一个更大更深的网络结构，并通过它学习更加复杂的对象特征，实现更加困难的分类任务。除了网络结构的变化，AlexNet也引入了一些神经网络算法实现上的变化。其中比较重要的几点包括将线性整流单元设置为网络的激活函数，模拟了人类神经系统的激活反应；在全连接层使用dropout进行随机去连接，缓解了过拟合的情况；引入最大池化代替平均池化，增强了模型的特征表达。这些细节的设计，使得AlexNet在ImageNet数据集上执行1000类的分类任务时，远远超出了传统视觉算法所能取得的准确率。

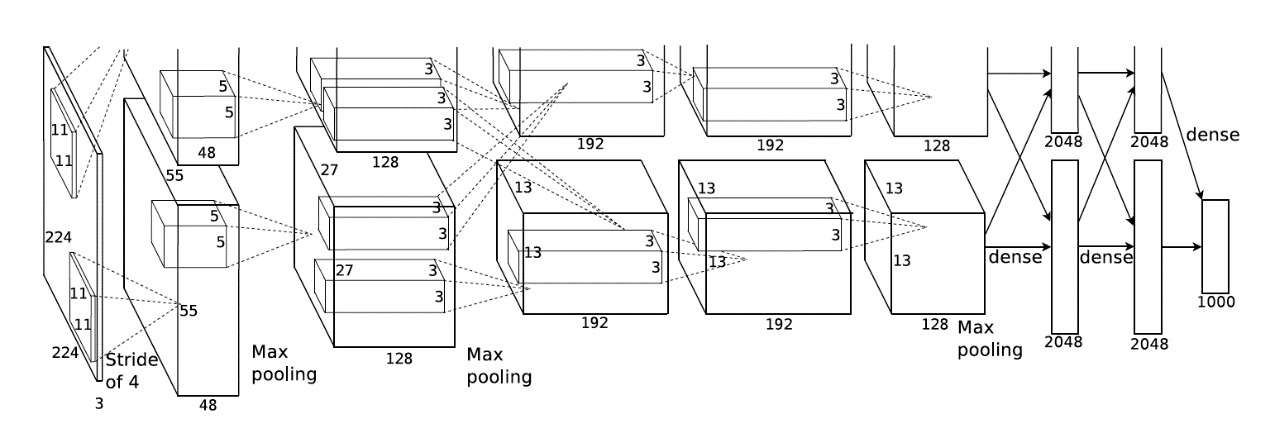


图2-1 AlexNet的模型结构

2.1.3 不断刷新纪录的新模型结构

AlexNet的成功为计算机视觉的发展带来了巨大的正面效应，以卷积神经网络为代表的深度神经网络渐渐成为了分析数据特征，处理复杂任务的理想选择之一。到了2014年，深度神经网络的又迎来了具有里程碑意义的事件。在该年的ILSVRC中，有两个崭新的网络结构被提出，并且纷纷刷新了赛事记录。

其一是VGG-Net[4]，它的主要贡献是在每个卷积层中都是用较小的3×3的卷积核，代替了AlexNet中11×11或9×9等较大的卷积核的使用。VGG-Net提出了连续的3×3的卷积可以模拟出较大的卷积核的卷积效果的观点，比如，可以将两个连续的3×3的卷积核等效成一个5×5的卷积，并且由于层数的加深，模型能得到更强的非线性。这一做法被后来的许多工作所借鉴。然而由于众多3×3卷积核的组合，VGG-Net的模型规模十分庞大，时至今日，它的两个变种VGG-16与VGG-19仍是处理相同问题时所需要的最大两个模型。参数量过大会带来许多问题，其中最显著的就是模型的训练困难，这也是VGG-Net存在的主要问题。

其二是GoogLeNet[5]，它的主要贡献在于提出了Inception模块以及Bottleneck Layer的思想，平衡了网络的深度与宽度之间的关系。Inception模块使用了不同大小的卷积核对输入特征图进行卷积操作，再将这些从不同感受野获得的特征进行了融合，使得每一层的特征拥有更丰富的尺度，增大了特征图的规模。Bottleneck Layer采用了Network in Network[6]的结构，通过1×1的卷积核的合理使用，不仅大大减小了特征的数量和网络的运算量，还去除了输入特征之间的相关性,达到了去冗余的目的。通过这两点创新，GoogLeNet的模型规模要远远小于VGG-Net，进行一次推理所需要的运算也仅为VGG-Net的十分之一，却在当年的比赛中取得了比VGG-Net更好的成绩并夺得冠军。

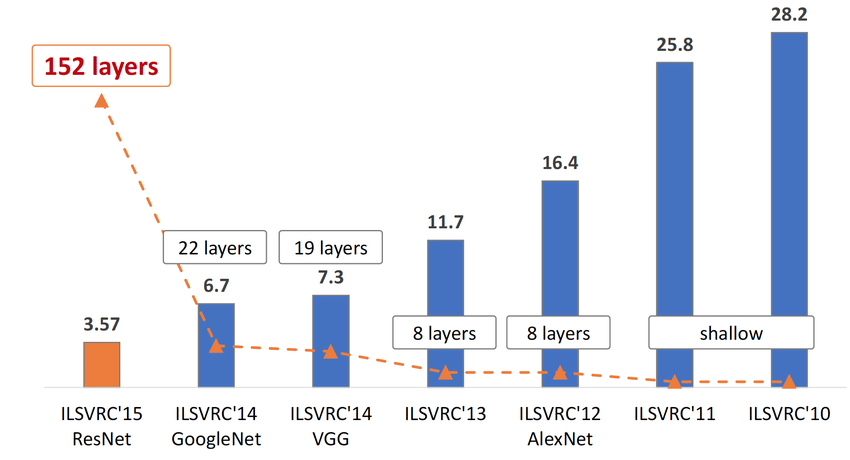


图2-2 ILSVRC历年的重要模型结构与其网络深度

2015年初，GoogLeNet完成了一次升级，新的模型结构被命名为Inception-V2[7]。Inception-V2引入了BN（Batch Normalization，批量归一化）算法，它计算了输出特征图的均值和标准差，并利用它们对输出的网络的响应进行了归一化处理，使得模型的训练实现了稳定的加速。BN算法至今仍被广泛运用于各类模型中，本文的第三章会对该算法进行详细的介绍。

2015年底，另一个具有重大意义的模型ResNet[8]被提出。ResNet解决了随着网络的加深，训练集的准确率反而出现了下降的问题，使得深度神经网络真正可以往深度上尽可能扩展，从而训练出成百上千层深度的网络。ResNet之所以有这一优势，是因为它引入了残差模块的概念，利用“Shortcut”的连接，将网络的任务从学习输入到目标之间的映射转变成学习输入与目标之间的残差，即使加深网络的深度，依然能保证网络的性能。ResNet在ImageNet数据集上能够将Top-5错误率控制在5%以下，并在该任务中首次取得了超越了人类的表现。残差模块是当前最流行的神经网络模块之一，在各类深度学习任务中都被广泛地应用。

从AlexNet提出到如今深度学习领域百花齐放的景象的出现，不过短短6年多的时间。图2-3展示了几种神经网络在Top-1准确率、模型运算量与模型大小的对比，根据图中信息并总结前人的工作经验可以发现：虽然深度神经网络的模型结构不断地在迭代变化，但是由卷积、归一、激活、池化组成的网络结构基本保持了稳定；虽然网络规模的增大加大了模型训练的难度，但是不断开拓网络结构的深度进行更复杂的任务的推理与完成是模型设计的大势所趋；虽然通用处理器的运算能力越来越强，但是减少网络的训练与推理所需要消耗的运算与资源的目标始终如一。

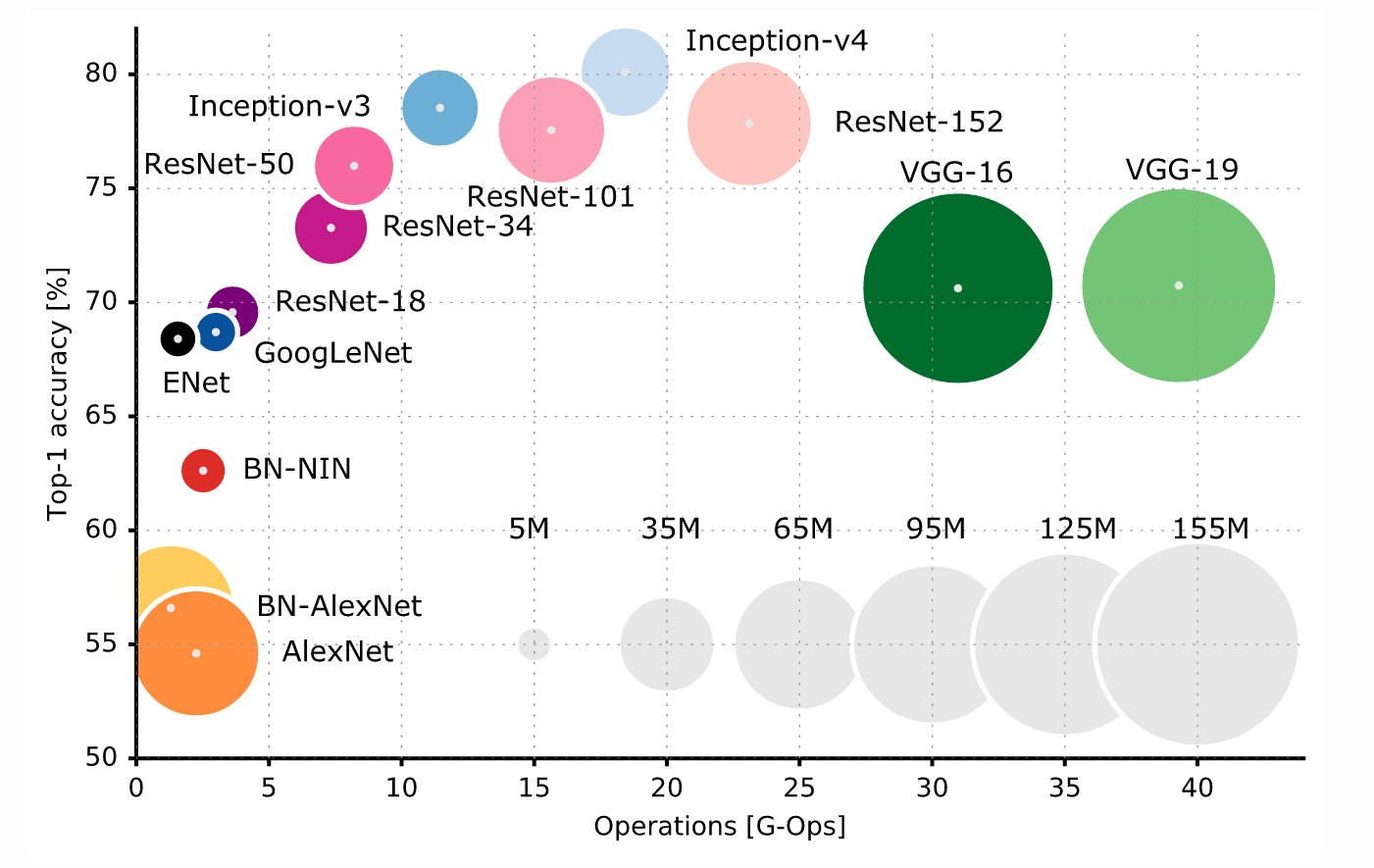


图2-3常用的神经网络结构的对比

2.2 卷积神经网络的常用结构

2.1节对卷积神经网络的模型结构发展进行了回顾，本节将对2.1节中的内容进行提炼，着重介绍其中对当前的卷积神经网络设计依然发挥重要作用以及与本文的工作有密切关联的卷积神经网络的常用结构。这些常用结构包括数据输入层、卷积层、归一化层、池化层和激活层以及全连接层。其中，数据输入层是网络的入口，主要用于对原始数据的处理，包括去均值、去相关等。全连接层通常位于网络的末端，用于分类或回归等任务的最终实现。而其他四层则常以组合的形式出现在网络之中，以完成神经网络对数据特征的提取。接下来将对各层的特点与作用进行说明。

2.2.1 卷积层

卷积层是卷积神经网络中最重要的一层，也是卷积神经网络的名字由来，它的作用是进行特征的提取和映射。卷积核是实现卷积层的关键，卷积核可以看做是一个大小为的三维矩阵，其中分别是它的行宽，列高与纵深。在平面上，是卷积层的卷积窗口大小，也被叫做感受野大小，可以根据实际的需求来定义，的大小等于输入特征图的通道数。图2-4展示了不考虑深度时，平面卷积运算的示例。卷积层包含以下几个重要思想。

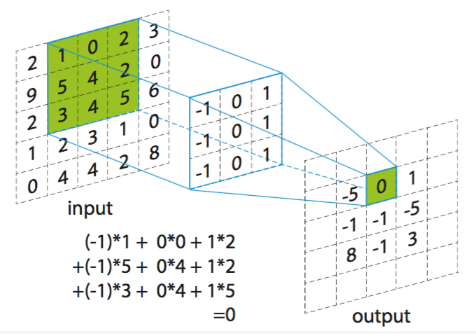


图2-4 卷积运算的示意图

其一是局部感知：在多层感知器中，每个神经元都要与输入特征图的每个像素相连接，巨大的权重数量造成网络难以训练。卷积层的出现改变了这一现象，卷积层中的每个神经元连接的权重个数等于卷积核的大小，意味着每个神经元只与特征图的部分像素相连接，从而大大减少了模型的大小。

其二是权值共享：卷积核的参数更新发生在网络的反向传播阶段，而在网络的前向传播阶段，卷积核的参数是固定不变的。由于卷积核的感受野有限，权值共享通过对输入特征图不同位置的特征的遍历，完成了对图片整体特征的提取，一定程度上是对局部感知的补充。

其三是多核卷积：单个卷积核中的权重个数有限，只能对应有限的特征的提取。卷积层通常使用大量的卷积核，来增强特征提取的能力。

卷积层占据了神经网络运算量的绝大部分，是限制卷积神经网络训练与推理速度的一大瓶颈。本文工作的一大重心即是优化卷积核的表示和卷积运算的实现，并据此提出了一种量化卷积的操作。

2.2.2 归一化层

归一化算法的发展经历了许多次迭代，AlexNet中引入了LRN（Local Response Normalization，局部响应归一化层），InceptionV2中提出了BN算法并沿用至今。期间还有Layer Normalization[42]、Instance Normalization[43]以及 Group Normalization[44]等一系列归一化算法被提出。它们主要的不同在于对数据进行归一化的尺度不一致，而它们的核心思想是相通的，即通过对数据的归一化操作，加快模型的收敛，提升模型训练的稳定性，增强模型的泛化性能。

目前来看，BN算法仍是在常用的卷积神经网络中使用最为广泛的归一化算法。在本文中，一种近似的批量归一化算法被提出，改变了传统的BN算法在硬件实现时存在的一系列问题。

2.2.3池化层

池化层的作用是对输入的特征图进行降维，一方面减小了模型的参数数量，降低了网络的复杂度；另一方面完成了对主要特征的提取。池化操作是网络中最简单的操作之一，常用的池化方式有最大值池化和平均池化，它们根据任务的差异，分别选择了池化窗口的最大值和平均值作为输出，前者常用于卷积层的特征提取过程中，后者更多地用于损失函数前，代替全连接层的功能。图2-5展示了一个池化窗口大小为2，池化步长为2的最大值池化的实现过程。

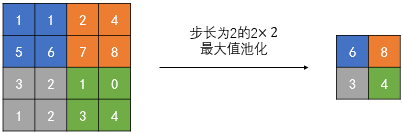


图2-5 最大值池化的示意图

2.2.4 激活层

激活层也叫非线性层，通过激活函数对数据进行逐元素的激活操作，以模拟了人的神经系统只会对部分刺激做出反应的功能。在早期的多层神经网络的结构中没有引入激活函数，因此网络只具备线性映射能力，隐层的作用无法体现；而激活层通过引入激活函数，增强了网络的非线性和表达能力。常用的激活函数有Sigmoid、Tanh和ReLU等，其中，ReLU函数是最简单也是被使用最多的激活函数。后期的一些工作在ReLU的基础上进行改进，并提出了 Leaky-ReLU、ReLU6等变种。图2-6展示了ReLU激活函数与ReLU6激活函数的不同形式。本文中的激活函数借鉴了ReLU6的思想，使用的是带动态上限的ReLU函数，具体实现方式将在第三章中进行陈述。

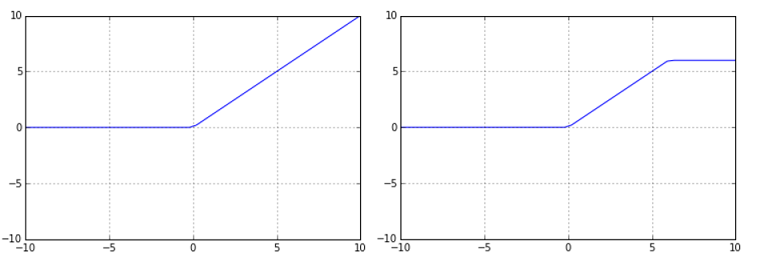


图2-6 ReLU激活函数（左）与ReLU6激活函数（右）

2.3 神经网络的优化技术

深度神经网络的加速技术伴随着深度神经网络的崛起而备受关注。虽然芯片技术不断发展使得计算设备的运算能力的边界不断拓宽，但是，深度神经网络的在中小型设备上的部署仍然困难重重，对于深度神经网络的压缩与加速算法的研究正在被广泛地开展。深度神经网络的加速是一门非常复杂的系统的技术，它既涉及到算法的优化，又要考虑到模型的性能，还要兼顾硬件本身的特性，才能综合地设计出一套完整的深度神经网络的加速方案。通常这样的一套方案包含两部分的技术，其一是神经网络的压缩，主要解决的是神经网络的模型规模过大的问题，它涉及到的主要技术有低秩分解，剪枝，量化，知识蒸馏、模型精简等；其二是神经网络的加速，主要解决的是神经网络在硬件上的加速问题，它主要研究的是运算加速、并行性的设计和内存访问的优化。这两部分的技术是相对独立又相互联系的，独立体现在关于它们的研究可以只专注于各自领域，同样能够取得一定的成果；联系体现在压缩的实现过程中通常也会产生网络加速的效应，而加速的实现需要通常要以网络的压缩作为前提。

2.3.1 神经网络的压缩

深度学习的许多成果无法被移动电话等设备中所共享的最主要原因之一就是神经网络规模较大，导致模型无法存储在本地设备中，因此模型压缩对于深度学习成果的落地而言是非常重要的。当前，神经网络的压缩主要有5种办法，图2-7展示了2017年至2018年中NIPS等行业顶级会议中收录的神经网络压缩方向的论文的数量统计，从中可以看出，量化与剪枝是最为流行的压缩方式，而其他三者的研究热度相对较小。下文中将就各种压缩方式做简要介绍。

图2-7神经网络压缩方向的论文数量

其一是低秩分解[9] 10]：它从卷积运算的本质出发，通过奇异值分解等矩阵分析方法，将较大的卷积核分解成一组小卷积核的组合，既减小了模型的大小，又加速了模型的推理。低秩分解主要作用于较大的卷积核，随着卷积核VGG-Net、Bottleneck等结构的流行，较小的卷积核得到了更多的使用，低秩分解的研究变得越发困难。

其二是网络剪枝[11][12][13]：它的核心思路在于各个神经元对于整个神经网络的功能并非是同等重要的，通过剪去不重要的连接，剪枝的办法能够把网络的复杂度降低到十分理想的水平。剪枝操作可以在不同的尺度展开，既可以随机剪去离散的神经元，又可以剪去连续的一组神经元，甚至可以直接剪去某些卷积核。剪枝是一种比较有效的神经网络的压缩算法，早期的随机剪枝存在的对硬件不友好的问题也被后期的结构化剪枝所改进。

其三是知识蒸馏[14][15]：它的原理是通过引入与复杂的教师网络相关的软目标作为总体损失的的一部分，以诱导精简的学生网络的训练，实现知识迁移的过程。学生网络的模型规模可以根据实际情况来制定，它的训练是一个新的端到端的深度学习任务。通过知识蒸馏的方法训练出的学生网络在精度上较有保障，但是训练学生网络所需要的资源和迭代周期都比较长。

其四是量化[16][17][18][19][20]：它将模型的参数从32比特的全精度浮点数值量化成低比特的定点数值，能够直接将模型的大小压缩若干倍。此外，量化操作还可以发生在模型的训练阶段，对激活值与梯度都进行低比特量化，从而降低网络的带宽需求。量化是当前进行神经网络的优化，特别是硬件上的优化最重要的方式之一，也是本文采用的方式。在第三章中会详细介绍。

最后是模型结构的优化[21][22]：它不同于上述的四种利用原有的神经网络模型结构进行优化，进而实现模型的压缩的方法，直接从网络结构设计的角度出发，利用更精简的网络进行训练和推理，生成较小的网络。MobileNet[21]、ShuffleNet[22]是它们的代表，由于它们与本文的工作在思路上不相关，因此不做赘述。

2.3.2 神经网络的加速

本小节中，神经网络的加速指的是神经网络在FPGA硬件平台上的加速。事实上，在CPU、GPU等通用硬件平台上，同样有许多加速神经网络的工作正在进行中或是已经被广泛应用。比较有代表性的有CPU上的OpenBlas以及GPU上的CUDA等，前者使CPU能够更好地支持矩阵运算，后者提高了GPU上的并行处理效率。然而，由于FPGA所具备的更高的并行度，更低的功耗等优势，神经网络在FPGA上的实现与加速仍是当前重要的科研方向，它主要要考虑的问题减轻模型的运算量、优化数据的交换与提高计算的并行度等问题。

对于神经网络的运算，与CPU和GPU等通用平台上的一些软件库一样，也有各式各样的变换用于FPGA上的神经网络的运算，包括GEMM变换[23]、Winograd变换[24]、快速傅里叶变换[25]等，它们用于实现特征图和卷积核的向量化，减少推理过程中出现的算术运算次数。

数据吞吐最优化是神经网络能否在FPGA快速实现的关键之一。由于FPGA中的寄存器的资源十分有限，进行神经网络的训练或推理必然要涉及到寄存器与片上缓存之间的数据交换。而随着神经网络的模型的增大，FPGA的片上缓存空间的大小已经难以满足所有模型参数的存放，因此片上缓存与片外存储之间的数据交换也在所难免。数据交换的过程牵扯到了数据的分块、循环展开的顺序、部分和的存取等一系列问题，在实际情况中要进行协调。许多工作[26][27][28]在理论上对相关内容进行了讨论。

神经网络在FPGA的加速效果主要体现由计算的并行度体现。首先神经网络的训练与推理都涉及到了大量的卷积操作，卷积操作的本质是乘法累加运算，这些操作是可以并行进行的；其次，深度神经网络包含了内在的并行性，包括卷积核的并行、特征图的并行以及同一批次样本的并行。能够支持大规模的并行计算是FPGA的一大优势，但是在实际部署中，计算的并行度还是要结合系统的带宽和内存空间，数据的吞吐量，乘法累加单元的数目等具体情况进行具体分析。

上述的三个部分是对神经网络加速的几个主要问题的简述，由于神经网络的加速通常涉及到具体硬件的综合实现，是非常繁复的内容，本文在此处不做详细展开。

2.4 TTA架构的主要特点[29]

TTA架构是传输触发架构（Transport Triggered Architecture）的简称，它的核心思想利用数据传输来触发相应功能单元的具体操作。当前比较流行的系统架构是操作触发架构（Operation Triggered Architecture），简称OTA架构，包括RISC（Reduced Instruction Set Computer，精简指令集计算机）、CISC（Complex Instruction Set Computer，复杂指令集计算机）、VLIW（Very Long Instruction Word,超长指令字）等传统的计算机系统结构中都是OTA架构体系的成员。在OTA中，编程是通过指定运算来完成的，当运算操作完成时，运算结果会被放入指定的寄存器中，完成数据的转移，换言之，数据的传输过程是隐性的。而TTA架构下的编程模式正好相反，在TTA架构中，数据传输在系统结构中唯一可见，编程是通过指定功能单元间的数据传输来进行。数据在各个功能单元间传输的同时，触发功能单元进行运算，传输触发的概念也就来源于此。

图2-8展示了TTA处理器的一般结构。TTA架构中不同的功能单元通过寄存器连接到交换网络上，不同的功能单元再通过交换网络连接成一个系统整体。MOVE指令是TTA架构中唯一的操作指令，它将数据从一个功能单元传输到另一个功能单元，并触发相应的操作。相比较传统 RISC结构和 VLIW 结构的指令格式，TTA架构的指令格式单一，指令译码非常简单，并且TTA架构的每条子指令都含有指令码，非常适合 SIMD（Single Instruction Multiple Data，单指令多数据流）和MIMD（MultipleInstructionStreamMultipleDataStream，多指令多数据流）结构的设计，用于深度学习等大规模数据的处理。

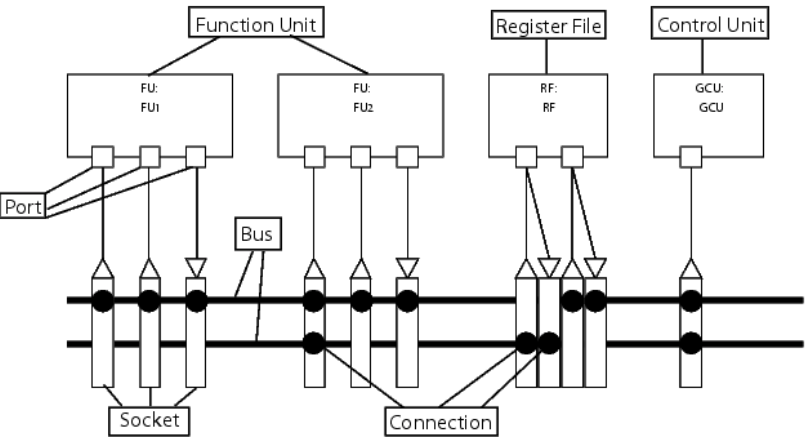


图2-8 TTA处理器的一般结构

在流水线的设计上，TTA架构的功能单元通常采用的是叫复合流水线的结构。在该结构中，只有在本级流水线的结果不会覆盖下一级流水线的结果时，流水线才会执行。在锁定机制的选择上，TTA架构有功能单元锁定和数据传输锁定两种选择，前者制使得功能单元与交换网络尽可能地分离，功能单元也提供了数据的暂存功能，直接减少了通用寄存器的数量；后者不需要编译器在程序中插入空指令，为程序存储器节省了空间。

TTA架构的这些特性使得它在物理实现时具有运行频率高，功能单元灵活，可拓展性强等优点。但是，TTA架构的简单性带来的影响是编译器的设计变得比较复杂，这也是TTA架构下处理器设计的难点，一定程度上制约着TTA结构的发展。本文的工作内容即在模型量化与运算加速两方面，设计部署在TTA架构ASIP上的卷积神经网络的优化方案。

第三章 神经网络的量化方案设计

3.1 量化技术调研

深度神经网络是当下解决计算机视觉、自然语言处理和语音识别等许多现实任务最先进的方法。然而，其在训练和推理过程中消耗大量内存和设备电池寿命，使得模型在资源受限的移动或嵌入式平台上难以部署这些模型。深度神经网络的量化是能够满足深层神经网络模型极端内存要求的最有效方法之一。

量化神经网络不再使用32位浮点格式来储存和表示权重、激活甚至梯度，转而使用少量比特数，比如8比特甚至1比特来进行模型的训练和推理的运算。量化神经网络可以有效地缩小模型大小，加快训练和推理过程。虽然量化操作牺牲了模型的预测性能，但它提供了一种有效的解决方案，在模型推理的精度损失有限的情况下，极大程度减小了模型大小和能量消耗。

人工神经网络的量化可以追溯到19世纪90年代。在早期，量化这些模型的主要原因是为了简化数字硬件的实现。近年来，由于深层神经网络的成功以及其背后庞大的模型规模，量化神经网络的研究重新兴起，许多对神经网络的量化技术被提出。

神经网络的量化可以从权重、激活值和梯度这三个部分展开。量化它们的动机和方法都不尽相同。通过权重的量化，我们能够得到较小的模型大小，辅以激活值的量化，我们能够加速模型前向推理的过程。在分布式训练环境中，通过梯度的量化，可以节省训练机之间大量的通信开销，加快网络的反向传播过程。一般来说，量化梯度比量化权重和激活要更困难，因为要使优化算法收敛需要高精度的梯度。

神经网络的量化可以根据是否采用码表展开成两个分支。一个分支是标量或矢量量化，该技术最初用于数据压缩并且有着悠久的历史。在进行标量或矢量量化时，原始数据使用码本和一组量化码来表示，码本包含一组量化中心，而量化码用于指示量化中心的分配。一般来说，量化中心的数量要远远小于原始数据的数量，因此，量化码可以通过哈夫曼编码[11]等无损编码方法编码，或者通通过低比特的定点数表示，从而实现很高的压缩比，PQ（Product Quantization，乘积量化）算法[30]是这类量化方式的代表。另一个分支是定点化，该方法将32位的全精度的浮点数值量化成低比特定点数值，例如，或者2的幂次数，同样是减小模型规模、降低网络资源消耗的有效方法。大量关于定点化的工作被开展，它们中的一部分成果甚至使量化神经网络具有与全精度神经网络相近的精度水平。

在上述的两个分支中，神经网络的定点化是当下进行神经网络量化所采用的更主流的技术。本文对比较重要的神经网络的定点化工作中进行了充分的调研。

3.1.1 二值化连接[16]与二值化权值网络[17]

二值化连接（Binary Connect）和二值化权值网络（Binary Weight Networks）共同之处是都将卷积核的参数，经过较简单的操作，转化为的二值化表示。由于只需要适用1比特来表示二值化的模型参数，因此它们都将或者几乎将模型大小减小至32比特全精度模型的1/32，并且使得乘法累加操作被简单的加减法操作所取代，从而达到加速模型推理的目的。

二值化连接分析了确定性二值化和随机二值化两种量化方式，分别表示为：

其中，表示为：

确定性二值化和随机二值化均是后期许多神经网络的量化工作借鉴的量化方式。论文对两种量化方式进行了实验，在MNIST，CIFAR-10和SVHN数据集上，随机二值化的结果均优于确定性二值化，并且两者都取得了与全精度的模型相当的准确率。

二值化权值网络与二值化连接的最大的不同在于，前者为了最小化量化权重和浮点权重之间的L2距离，引入了一个比例因子，并将卷积操作表示为：

上式中，表示的是一个不需要任何乘法操作的卷积运算。的表达形式如下：

上式表达的是比例因子的取值为卷积核所有参数的平均值。在后期开展的量化工作中，虽然比例因子的取值方式各有不同，但是作为一种最小化量化值与实际值之间的距离较为直接的方法，比例因子始终被广泛地采用。

在ImageNet数据集上量化AlexNet模型时，二值化权值网络能够达到与全精度模型相近的准确率。在CPU训练加速的实验中，二值化权值网络达到了约2倍的加速效果。

3.1.2 二值化神经网络[18]与异或网络[17]

二值化神经网络（Binarized Neural Networks）不仅对卷积核参数进行了二值化，还对网络的激活值进行了二值化。此外它讨论了传统的批量归一化的方式存在的问题并提出了提出了基于移位的批量归一化。在卷积神经网络中，基于移位的批量归一化，采用左移与右移操作代替了求平方的操作，并且不会使模型的精度产生明显的损失。

二值化神经网络通过一些技巧的组合，相比全精度模型，能够减少60%的训练时间。文中对于激活值的量化与对于批量归一化的优化是后期大量的神经网络的量化与加速的工作中被广泛地采用的思路，有着重要的意义。

异或网络（XNOR-Net）是二值化权值网络作者的后续工作。异或网络通过引入比例因子的，因此在ImageNet数据集上的表现相比二值化神经网络，相比二值化神经网络有了明显地提升。

由于权重与激活值都进行了二值化，二值化神经网络与异或网络的论文中都提出了可以用比特计数操作代替卷积操作中所包含的大量的乘法运算，这一变化能够显著地加速神经网络的运算。考虑到比特计数对硬件资源的依赖要远小于乘法运算，因此这一变化对深度学习专用硬件的设计也有很大的影响。

3.1.3 三值化权值网络[19]

三值化权值网络提出一种将网络权重简化为的方法,在引入比例因子的同时，还引入了一个阈值Δ来确定每一个权重的取值。三值化权重的取值方式如下：

论文[19]给出了关于如何确定Δ的取值的详细推导，由于其中包含了大量复杂的数学运算，增加网络的训练与推理的时间，因此论文中提出了近似获得Δ的取值的方法：

由于每个三值化权重都需要用2个比特来表示，因此三值化权值网络的模型相比二值化权值网络要扩大一倍。三值化权值网络增加了额外的0值， 对于一个3×3的卷积核，三值化权值组合拥有种配置，相比二值化权值网络的种，前者的的表达能力相比后者增加40倍左右，这对于比较复杂的任务而言是巨大的收益。由于0值不增加任何的乘法运算，因此三值化权值网络中的乘法累加操作次数与二元权值网络相比保持不变。三值化权值网络的在MNIST，CIFAR-10和ImageNet数据集上的准确率都要优于二值化权值网络，尤其是在ResNet等较深的网络结构中。

3.1.4 DoReFa-Net[20]与梯度的量化

虽然上文中提到了许多量化权重话激活值的方法，但是它们在反向传播过程中的都保留了全精度的梯度的使用，因此该过程依然涉及了32比特浮点数与1比特量化值之间的乘法。这就意味着，在训练阶段，大部分训练时间都会消耗在反向传播过程中。

关于量化梯度的工作比较有限， DoReFa-Net是第一个在保持相当的准确率的要求下对梯度进行低于8比特的量化的工作。实际上，DoReFa-Net并不指定神经网络应该被量化到多少比特，而是设计了一套完整的量化神经网络的解决方案，能够将神经网络量化到任意的比特，与之前的工作有很大的创新。

在权重的量化上，考虑到比例因子的加权会是反向传播中无法开展比特运算，因此DoReFa-Net没有引入比例因子，而是通过Tanh函数将权重的值限制在的区间内，k比特权重的量化方式如下：

在激活值的量化上，DoReFa-Net同样没有采用比例因子，而是简单地使用Clamp函数将激活还限制在的区间内。

在梯度的量化上，DoReFa-Net使用了随机性量化的策略，因此引入了一个随机噪声，并且通过实验验证了它有非常明显的效果。k比特梯度的量化方式如下：

DoReFa-Net在SVHN数据集与ImageNet数据集上都有良好的表现。作为一套完整的神经网络的量化方案，DoReFa-Net的工作对本文的开展提供了丰富的实践经验与巨大的指导意义。

3.1.5 其它低比特量化网络

除了上述论文所介绍的量化策略外，还有许多论文也提出了各自的量化策略，同样具有启迪性。比如，论文[31]提出了一种先量化权重后量化激活值，并在训练过程中逐步减少量化位宽的策略。论文[[32]](https://arxiv.org/pdf/1807.00301.pdf)的主要贡献在于对不同粒度的量化方法进行了讨论，分析了像素级，向量级，矩阵级，卷积核级等不同级别下量化的性能。论文[33]研究了量化对损失的影响，提出了量化问题是一个可以被近似牛顿算法解决的最优化问题。论文[34]在理论上推演了8比特量化神经网络在整型运算平台上进行推理的可行性，并进行了大量的验证，证实了通过合适的方法，8比特量化能够适用于当前近乎所有的深度学习网络结构，并能够在各种数据集上完成深度学习任务……

3.2 本文量化方案的整体思路

上文中介绍的各种量化策略以及它们留下的一些基本方法与宝贵经验，是本文中的量化策略的制定的理论基础与实践指南。结合本文的工作是面向TTA架构ASIP设计的深度神经网络的优化，本文要考虑的因素与目标又与上文中的种种量化策略有所不同。本文策略的立足点主要是以下几个：

一是模型支持。模型支持是本文工作的基础，它指的是量化的方案能够适用于当下常用的神经网络模型。上文中介绍的二值化量化策略或是三值化量化策略，虽然都极大地压缩了模型的大小，并且也能达到加速模型的训练或者推理的效果，然而，它们中的大部分方案都只能在SVHN或者CIFAR-10这种较小的数据集上达到比较理想的效果，在ImageNet数据集上的测试的结果与全精度模型仍存在较大的精度损失因此，立足于模型支持这一基础，本文不追求1比特、2比特这种极低比特的量化在模型压缩上带来的红利，而是权衡模型大小与模型支持之间的关系之后，选择了8比特作为本文的量化位宽。8比特的量化策略能够达到4倍的压缩效率，被许多工作[34][20]证明能够支持当前常见的数据集与常用的模型。此外，不仅能够在本文中依托的FPGA平台上较好的实现，8比特的运算也被许多其他的硬件平台支持。

二是算法优化。算法优化是本文工作的核心，它指的是量化的方案能够达到对神经网络传统的实现方式能够有所优化。神经网络中最主要的操作是卷积操作，传统的卷积操作涉及了大量的乘加运算。虽然当前的FPGA都集成了较多的DSP模块，使得乘法累加操作能够在一个时钟之内完成，但是，考虑两点：一是DSP模块相比于FPGA上的简单操作，它的运算在能耗上来说是比较贵的，特别是在神经网络的部署中，一次前向操作就可能调动数亿次的MAC操作，DSP模块的消耗需要被考虑，卷积操作的优化需要被考虑；二是DSP是否真的不再是神经网络在FPGA上部署的瓶颈。虽然卷积操作是神经网络的主要运算，但是量化神经网络中的量化操作同样涉及到大量可能需要DSP支持的运算，DSP资源如何分配，如何解决FPGA上时间依赖于空间依赖的平衡，是本文在算法优化上的重点。此外，本文优化了神经网络训练中被广泛使用的批量归一化算法，在3.3.3小节中会详细介绍。

三是硬件适应。硬件适应是本文工作的目标，它指的是量化的方案能够适用于FPGA等硬件的实际部署。神经网络处理器应该能够实现神经网络的训练与推理两个阶段的任务。上文中提到的许多量化工作，大部分只量化了神经网络的权重与激活值，而没有优化梯度，这也就意味着，它们只针对神经网络的前向传播进行了加速。由于反向传播过程中全精度的梯度的使用，模型其实仍然较难在FPGA等硬件平台上进行训练，即便可以，仍然要有大量的时间与资源需要消耗在梯度更新的阶段。因此，本文的方案不仅对模型的权重与激活进行量化，对模型训练过程的梯度也进行了量化，并且对梯度传播的不同阶段实施了不同量化比特的设计，在保证训练时间的同时，提升了训练精度。本文的量化方案是端到端完整的支持训练与推断的全量化的方案。

本文提出的量化方案主要有以下创新：

（1）本文提出了一种关于0对称的权重仿射量化策略。它的特点是通过含有比例因子与偏置因子的仿射变换，充分利用了整个权重的量化空间。并且关于0对称的策略，对后文中设计的量化卷积运算的实现，能够节约一半的LUT（Look-Up-Table，显示查找表）资源。

（2）本文提出了一种简化的近似批量归一化的处理方式。通过充分分析批量归一化的之所以能够为神经网络的训练带来优化的原因，将传统的批量归一化中包含的求标准差的操作转变为简单的求取范围的操作，在基本保证模型精度的情况下，节省了大量的操作。

（3）本文提出了一种变精度的梯度量化的方案。它与其它工作的主要不同在于，在梯度传播的路径上使用8比特精度的量化，在梯度作用在当前层的权重更新时，使用16比特精度的量化。优化了梯度的量化，既提升了权重更新的精度，又不影响模型的效率。

3.3 本文量化方案的具体实现

3.3.1 权重的对称仿射量化

权重也就是卷积核的参数。对于权重的n（n>2）比特量化而言，当前主要有两种量化策略，一是对称的量化，它量化的范围是，并通过比例因子将它转换为。然而，在实际情况中，权重的大小往往不是关于0对称的，因此该量化方式对于量化精度存在着一定程度上的浪费。比如，当某个卷积核的参数全部为正数时，只有一半的精度在该卷积核的量化中起了作用。另一种量化策略是仿射的量化，它量化的范围是，通过比例因子和一个偏置因子，将它转换到空间中，从而充分地利用了整个量化空间。基于此分析，本文采用的是仿射的量化策略。

权重通常会被映射到两种形式的量化空间中，它们可以大致表示为与。在第一种量化空间中，权重被量化成以2的负指数次幂，它的优势在于，在进行卷积操作时，能够利用权重的特点，将乘法运算转换为移位运算。而它的问题在，量化的间隔是不均匀的，大量的精度都消耗在了0值附近，而较大的权重获得的精度非常有限。论文[35]指出，数值上较大的权重会对模型的精度产生较大的影响，显然，第一种量化空间只关注了2的负指数次幂对于运算的优势，却在量化权重对于模型精度的影响上本末倒置。因此，本文选择第二种量化空间。

权重在第二种量化把权重量化成了分母为，分子为整数的分数，它们在空间中的间隔是均匀的，使得每一个量化值对于量化精度的增益最大化。考虑到FPGA上可以利用DSP模块在一个时钟周期内完成乘法累加操作，因此，对比两种量化空间，它们在运算的时间复杂度上其实是没有差别的。特别地，本文针对采用第二种量化空间时的卷积运算进行了优化，降低了乘法运算对于DSP模块的依赖，使其在能源利用率上也有了一定提升。

本文的关于权重的量化是以卷积核为尺度的，这也是在众多的权重量化工作中最常被使用的量化尺度。对于一个高度为，宽度为，深度为的卷积核，我们令表示其位于位置上的权重，其中。本文的量化规则如下：

上面一组式子中，和分别表示卷积核的的最大值与最小值，表示卷积核参数的最大值与最小值之间的范围，表示一个偏置因子，它的数值等于卷积核中参数范围的中心与0值之间的距离。考虑到和的个数仅等于卷积核的个数，对它们使用较高的精度并不会很大程度地影响模型的大小以及模型的运算规模，因此，二者在本文中均选用32比特来表示。式3-11代表的是量化操作，它又分为两个步骤实现，一步是取整（Round）操作，这里选取的是确定性取整的策略，即对操作数采用四舍五入的规则进行取整；另一步是钳制（Clamp）操作，是为了避免取整后的值成为或的。

上述的操作进行完之后，量化后的权重就分布在之间的整数中。如上文分析的，仿射的方法保证了量化空间能够被最大限度地利用，而对称的量化适用于本文中后续提出的的量化卷积运算，能够使卷积运算所需占用的内存资源与LUT资源减少一半，这部分内容将在第四章中详细描述。

在模型的前向传播阶段，被量化的权重应该从的整数空间被仿射回它们的真实值中。这一过程中包含的变换如下：

在上式中，是与维度相同的全“1”的卷积核。通过上式能够得到模型前向传播极端所需要的量化的权重。完成量化过程之后，一个全精度的卷积核可以通过个带符号的8比特整型数以及两个32比特的定点化值表示，量化后的模型在模型的存储空间需求与运行时的数据交换需求相比较之前的模型都减小了近。

关于全连接层权重的量化，论文[46]提出，维持全连接层的参数精度没有维持卷积层的参数精度重要。论文[47]指出，由于全连接层的权重在模型中占据了较大的部分，因此对全连接层的压缩在模型的整体压缩中更为重要。此前的一些量化工作，在全连接层与卷积层采用不同精度的量化的方案被实施过，比如，对卷积层的权重采用8比特的量化，而对全连接层的权重采用4比特的量化，并且取得了与对二者进行相同比特的量化时几乎相同的结果。本文认同这些工作的成果，但是在本文的方案设计，并没有采用相同的策略。本文对全连接层权重的量化与卷积层是大致相同的，不同之处在于全连接层是以层作为量化尺度的，也就是说，求范围和求偏置的操作是在整个全连接层这一尺度上进行的。

3.3.2 近似的批量归一化

BN算法是论文[7]提出的一种加快收敛速度，提升训练稳定性的算法，在图像分类等需要提取数据特征的任务中能起到巨大的优化效果。

在BN算法被提出之前，深度学习的模型训练存在较大的不稳定性，卷积核参数的初始化，学习率等超参数的初始化，Dropout结构的设计等因素对模型的效果都会产生不确定的影响，导致大量的人力物力都被消耗在了这些缓解的调试。BN算法的出现深刻地改变了这种局面，神经网络模型开始可以从较大的学习率开始训练，并且不再需要Dropout和L2正则化等算法，却能取得比之前更快的收敛速度。文章将这些红利归因于BN算法减小了隐层协变量偏移。

在训练过程中，随着网络的加深，神经网络在做非线性变换前的输入值的分布会逐渐发生偏移或者变动。比如，对于Sigmoid函数，它们的整体分布会逐渐往非线性函数的取值区间的上下限两端靠近，这导致反向传播到神经网络的低层时产生梯度消失的现象，这是训练深度神经网络时收敛越来越慢的本质原因。BN算法的基本思想其实是通过一定的规范化手段，把隐层神经网络中任意神经元的输入值的分布拉回到均值为0方差为1的标准正态分布，使得激活函数的输入值落在非线性函数对输入比较敏感的区域，从而避免大部分输入值落在激活函数的饱和区造成的引起的梯度消失的问题。梯度的稳定能够保证模型的学习能够以较快的速度收敛，进而加快了训练速度。

传统的BN算法的是这样实现的。假设某个卷积层的一个批次的大小为，每一个特征图输入包含个维度，将该批次的输入记做，对于是由对应的个特征构成的。那么对于，它的批量归一化表示为：

上式中，是的期望，也就是该批次特征图在维度上的平均值，是的方差。为了保证数据大小上的一致性，BN算法也引入了和作为它的缩放参数和平移参数，并且在网络的训练过程中，二者是可学习的。通过上述较为繁琐的运算，最终能够得到批量归一化后的值。对于神经网络的FPGA实现而言，它可能存在两个问题。一是上式包含了大量的平方求和的运算，它们的动态范围是比较大的，这要求在求解上式的时候需要较大的运行资源；二是上式还包含了开方和除法的操作，为了减小偏差，这些操作精度的要求比较高。

在实践中，BN算法在许多深度学习任务中被广泛采纳并极大地提升了模型性能，然而在理论上，BN算法并没有完备的数理推导，在一定程度上可以认为是一个经验性的成果。通过批量归一化能够把隐层激活函数的输入拉回到一个均值为0，方差为1的分布中，并且根据中心极限定理，可以把这个分布认为是近似的标准正太分布。但是，实际情况中，一个批次的输入往往难以满足中心极限定理的前提条件。一方面是，考虑到一个批次中的样本数量的可能比较小，比如16或者32等，不能满足中心极限定理需要大样本数量的条件；另一方面是，同一个卷积窗口的输入可能存在较大的相关性，它们并不是完全独立，因此它并不一定能满足中心极限定理的样本是独立随机变量之和的条件。基于这两点考量，本文的观点认为，BN算法的核心在于它将隐层中激活函数的输入转换到一个均值为0，方差为1的分布中，减小了卷积层输出特征之间的偏移。然而，由于存在可学习的缩放参数因子，随着模型的训练，方差的实际值是一直变化的。

本文提出了一种利用输入样本的范围与个数的组合替代方差的近似的批量归一化算法。对于一组共个来自于同一个分布的高斯变量，其中，它的最大值的期望与变量的个数以及标准差之间存在如下数学关系：

显然，对于它的最小值的期望，对应的有：

令表示变量的范围，则有：

利用高斯分布的这一性质，本文将近似批量归一化的形式表示为：

其中，是一个常数，表示的是的范围与与它的方差和批次大小之间乘积的近似比例关系。与原始的BN算法相同的是，近似的批量归一化算法同样保留了缩放因子和偏移因子。而二者的差别在于，对于一个确定的神经网络的训练过程，是一个常数，因此可以把看做一个常数项，此时，式3-13中求取方差的过程简化成上式中求取范围的简单运算，解决了BN算法中需要高动态范围和高定点精度的问题。

近似的批量归一化将较为复杂的求取平方和开根号的操作替换成数值比较和减法操作，虽然批量归一化的运算相比卷积的运算在深度神经网络的训练中占比有限，但是由于深度学习所需要的整体运算量巨大，这样的近似带来的运算资源的节约是不容忽视的。需要说明的是，因为近似的批量归一化算法同样是一个经验性的成果，所以的数值并没有明确地数学上的规定。在本文的实验中，当的被设置为常数2时，网络的训练能够取得相对较好的结果，但是相近的取值之间的差别对于结果的影响十分有限。本文中的第五章展示了近似的批量归一化算法的实验性能，验证了它的可靠性。

3.3.3 激活层与激活值的量化

使用ReLU（Rectified Linear Unit，线性整流单元）作为激活函数有三方面明显的优势：其一，相比Sigmoid函数或者Tanh函数，ReLU函数不存在饱和区，在反向传播过程中能够产生稳定的梯度，不易引起梯度小时，保证网络较快地收敛；其二，ReLU函数在众多激活函数中实现起来最简单的激活函数，前向传播时只需要判断输入的符号就并将小于0的输入置为0输出即可，反向传播时也不涉及复杂的求导，能使网络快速地计算；其三，ReLU函数能够使一部分的输入变为0，增加了网络的稀疏性，减小了参数之间的相互依赖关系，一定程度上解决了模型过拟合的问题。

综合以上的优势，本文中的实验均采用了带有动态上限的ReLU作为激活函数，并将输入经过激活层与激活值的量化这两个过程融合在一起。对于小于0的输入，直接将其置为0；对于大于0的输入，也就是激活值，它的量化的思路基本延续了权重量化的思路，引入了激活值的范围作为比例因子，但是在范围的选取上有所不同。下面先给出本文中激活值量化的方法：

激活值的量化是以整个输出层特征为尺度的。式3-18中，是表示的是输出特征的最大值，代表量化方案中激活值的范围，但它不是绝对范围，它在数值上等于与输入的最大值最接近且小于输入的最大值的2的乘方。这一选择是从两方面考虑的，其一是使用2的乘方作为激活值的范围，在进行量化可以使用移位运算代替除法，简化了量化操作；其二是ReLU作为激活函数时，对于较大的激活值，在反向传播时存在梯度过大导致神经元永久失活的问题，通过限制激活值的上限能够在一定程度上缓解这个问题。式3-19中的操作与式3-11类似，均包含了取整与钳制。式3-20是量化后激活值的表示。

3.3.4 变精度的梯度量化

与权重的量化和激活值的量化不同的是，对梯度进行量化的工作相对较少。这主要是因为相比权重和激活值，梯度的边界较大，因此梯度对精度的需求相应地提高。本文中梯度量化的方法与之前的工作相比，创新指出在于，本文考虑到了梯度在神经网络的反向传播过程中的差异性。具体来说，反向传播时，神经网络的梯度包含了两种类型，第一种类型的梯度是用于梯度的反向传播的，它们仍需要继续回传，该过程涉及到网络中大量的乘法操作；第二种类型的梯度是用于模型权重的更新的，它们只涉及与学习率以及权重之间的简单运算。结合这两种类型的梯度的特点，本文设计了一种变精度的梯度量化策略，具体来说，对于第一种类型的梯度，采用了8比特的量化；对于第二种类型的梯度，采用了32比特的量化。

与权重和激活值的量化相同，本文对梯度的量化同样采用了带有比例因子的量化方法。考虑到梯度在反向传播过程中不断扩散的特点，本文中梯度量化的尺度选择的是网络中层级的尺度。由于当前模型的训练通常会引入固定批量的样本进行训练，并且每一个批量的输入数据可能被拆分为若干个小批量，在这种情况下，进行梯度量化时比例因子的选取是以每一个小批量为单位的。

本文中梯度量化的方法如下：

式3-21中，是神经网络第层的梯度，表示的是比例因子，在数值上等于该层梯度绝对值的最大值。表示的是量化后的权重，它的取值范围是。式3-22给出了量化梯度的表示形式。由于第一种类型的梯度是反向传播的，那么对于，层的梯度表示为：

其中，表示的是第层的量化权重。得到后，可以将其继续进行8比特量化并反向传播。

对于第二种类型的梯度，也就是用于第层权重更新的梯度，它可以表示为：

其中，表示的是第层的激活值输入，也是第层的特征输入。如上文中介绍的，本文对该梯度将进行32比特的定点化后，直接用于权重的更新。本文结合应用实际情况引入了这种变精度的梯度量化策略，既能够为权重的更新提供更高的精度，又不影响网络整体的复杂度。

在模型参数更新阶段，量化的神经网络的反向传播方式与传统神经网络有所不同。由于量化过程所引入的运算是离散的，因此本文沿袭了许多先前的工作通用的做法，引入了STE[36]（Straight Through Estimator，直通估计）方法，它的思想是在反向传播中忽略量化函数和阈值函数等使数据离散化的分段函数，令通过量化函数输出的梯度等于它输入的梯度，从而解决了分段函数的梯度处处为0的问题。

第四章 神经网络的加速方案设计

本文的第二章总结了进行神经网络的加速方案设计可以考虑的几个方向，包括模型的运算量、数据的吞吐与计算的并行度，它们也是本章的优化方案的设计立足点。本文的第三章提出的神经网络量化方案，它既可以看做是独立的一部分工作，因为该量化方案可以在主流平台上进行训练和推理；又可以看做是本文中神经网络的加速方案设计的基础，因为其与本章中的加速方案互为补充的。

在AlexNet，InceptionNet和ResNet等当前常用的深度神经网络中，卷积操作包含的运算在神经网络的所有运算中的占比都达到90%以上。结合这一情况，本章的核心工作在于面向FPGA上的量化神经网络，加速卷积运算的操作。本章提出了一种量化卷积运算与量化卷积核的全新表达形式，并提供了配套的卷积运算的循环展开、卷积核与特征图的分块以及前向传播中的数据交换的设计方案。

4.1 基于乘法结合律的量化卷积

4.1.1 量化神经网络的前向传播

卷积运算是量化神经网络的前向传播中最重要的运算。对于全精度的神经网络中第层的一个大小为的卷积核，它与第层对应的特征输入（即层的激活值）进行卷积，产生第层的一个输出特征，所需要进行的运算如下：

上式中，即为第层卷积的一个输出特征。该卷积运算所需要进行的乘法累加操作的次数为次，假设第层输出的大小为，那么该层所需要进行的乘法累加操作的总次数将达到次。毫无疑问，全精度的神经网络中的卷积运算所需要的计算资源是非常巨大的。

量化神经网络的前向传播与全精度的神经网络的前向传播在原理上是相同的。当权重和激活值按照第三章中的方法量化后，对应式4-1，卷积运算可以描述为：

上式可以进一步展开，得到：

直观上，式4-3要比式4-1复杂得多。先分析式4-3的左半部分，它包含的可以理解为传统的卷积运算，而其余的操作则是由于量化方案在表示量化权重与量化激活值时都使用到了比例因子或偏置因子而产生的。这也是目前众多引入了8比特的量化的模型加速工作所存在的问题，压缩模型的大小、减少网络的交换是大部分量化工作的目的，而对于卷积运算所包含的大量的乘法累加操作，并未进行特殊的处理。

对于式4-3的右半部分，同样属于卷积运算，它的计算与在理论上是完全相同的。然而由于是全“1”的卷积核，因此大部分量化工作都利用乘法结合律对它的运算进行了优化，使其化简成：

式4-4也是本文基于乘法结合律的量化卷积方案的启蒙点之一。对于完成本文的量化方案后的量化卷积核，它参数所包含的量化值仅有255个，考虑到对称性后，实际包含的量化值仅为0~127之间的整数。也就是说，的运算同样可以采取乘法结合律的方式进行简化。

4.1.2 量化卷积的原理

4.1.1节讨论了对于一个大小为的卷积核，与对应的输入特征进行卷积得到一个输出特征所需要的乘法累加操作的次数为次。对于不同大小的卷积核，的大小在几百次到数千次之间不等。由于输入特征图的深度通常与网络的深度正向相关，因此的大小也会随着网络的加深而增大。图4-1展示了AlexNet与ResNet-18两种网络的各层或各模块为了得到一个对应输出所需要进行的乘法累加操作的次数统计。从图中可以看出，除了比较特殊的第一层外，其它层的卷积运算都是非常消耗计算资源的。

在FPGA中，乘法操作主要依赖DSP模块来完成。DSP乘法器能够在一个时钟周期里对两个一定位长的输入由硬件作快速并行乘法，比如常见的DSP48E单元，支持18×25的二进制补码乘法。在传统的卷积操作的硬件实现中，使用DSP乘法器进行乘法操作是不可缺少的操作。随着集成电路技术的发展，FPGA上能够集成了越来越多的DSP模块。尽管如此，有限的DSP单元仍然是制约神经网络在FPGA上加速的主要原因之一。

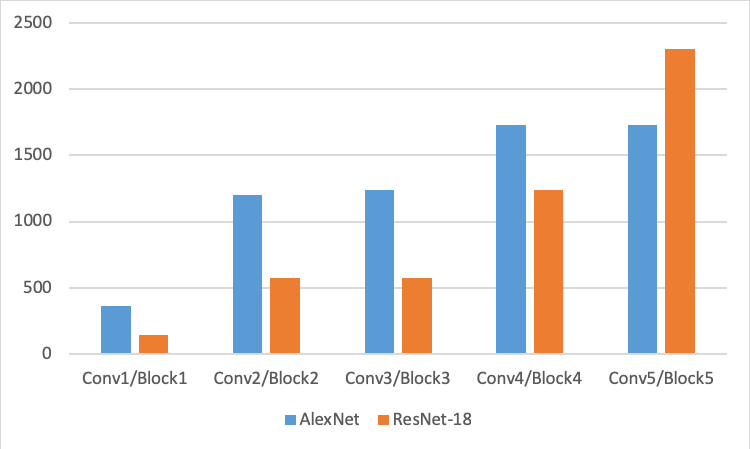


图4-1 AlexNet与ResNet-18各层或模块的一个卷积运算包含的操作数

传统的卷积运算的实现基于乘法累加操作，对于浮点数的卷积而言，它们之间的乘法累加操作并没有优化的空间。然而，与通用处理器进行卷积运算时的操作数来自连续的浮点数空间不同的是，在FPGA上进行卷积运算时，它的操作数是已经完成了量化的定点数值，它们的取值在一个有限的离散的量化空间中，有限集的整数之间的乘法为卷积运算带来了采用不同计算方法的可能性。

前人的一些工作进行了这方面相关的探索。比如论文[37][38]中都提到了利用查找表来优化卷积运算的方法。前者将卷积运算中所有的低比特乘法乘完全用查找表来实现；后者利用查找表来部分取代DSP的功能，二者配合DSP完成卷积的过程。它们对于本文而言共同存在的问题主要有两点：第一，对于更低的比特而言，查找表不失为一种好的替代方法，然而8比特的乘法操作所需要的查找表的占用空间是比较大的，在查找表资源同样有限的情况下，该方式能提供的优化效果有限；第二，使用查找表本质上并没有改变两数相乘的运算方式，仍然需要提供两个输入，通过一个时钟，得到一个输出，在时间上与利用DSP进行乘法相比，没有任何的提升。

论文[39]中提出对于a、b、c三个8比特数，如何利用一个DSP48E芯片，在一个时钟周期内高效地完成a×c和b×c两次乘法累加运算的设计，并表示该设计对神经网络的在FPGA上的实现很有帮助。该方法在理论上能将DSP的资源利用率提高一倍，并且a×c和b×c的形式对于拥有天然的并行性质的神经网络而言也很容易获得，是一个比较好的优化方案。

与上述的工作不同，本文从另一个角度出发来思考优化量化值之间的卷积的问题。对于神经网络中的一个卷积核，通常拥有比它的量化空间大小（本文中为28）更多的参数个数。如果能够通过将相同的参数对应的输入特征先完成累加，再送入乘法器，就能够将卷积运算对乘法的需求减小到比较低的水平。因此，本文提出一种基于乘法结合律的量化卷积运算的方法，先将相同权重所对应的输入元素相加，再进行对应权重的相乘与累加操作。相比受限于DSP单元个数的乘法操作，加法操作只需要使用查找表资源与连线资源，综合实现相对简单且可利用资源更加丰富，可以减轻DSP乘法器有限的数量对神经网络的并行计算产生的影响。

一般的，对应一个输出特征所需要的卷积运算的形式展开如下：

其中和分别是第层量化的权重和量化的激活值。根据第三章中提出的量化策略，在模型的前向运算时，卷积核的参数会被量化到一个的整数空间中，特征图则会量化到一个的整数空间中，在此优先关注卷积核的量化。由于卷积核参数的取值是之间的整数，根据乘法结合律，可以把式4-5改写成：

由于卷积核参数的量化空间是关于0对称的，因此，式4-6可以继续简化为：

式4-7即本节提到的基于乘法结合律的量化卷积方法的数学表达。量化卷积的方法，将获取一个卷积核参数所需要的次的乘法运算与加法运算简化成次的加法运算和128次的乘法运算。由于在数值上要远远大于128，因此，基于乘法结合律的量化卷积在理论上不仅削减了卷积运算的运算量，也缓解了卷积运算中的乘法操作对DSP的需求。

在此对第三章中引入的对称的量化方式引来的利弊进行简要分析。对称的量化存在的弊端在于它引入了符号，将无符号数的加法转为有符号的加法，增加了符号的拼接。然而，FPGA中有符号的加法与无符号的加法在性能上是基本相当的，在实现上并不会增加运算复杂度与时延。对称的量化带来的好处则非常明显。首先，相比非对称的策略，它能够减少一半的乘法运算，这一点与本文的出发点密切相关；其次，根据中心极限定理，卷积核参数服从高斯分布，也就是说，量化值为0的权重对应的输入是比较多的，由于0值对应的累加运算可以忽略，因此还能产生理论上的优化效果；最后一点是，在极端糟糕的情况下，它实现单个量化值对应的乘法所需要的加法次数与非对称的策略是相同的，对称的量化策略能够节省一半的整体运算量和总耗时。关于极端糟糕的情况，将在下一节中进行分析。

4.1.3 量化卷积实现方式的探究

利用4.1.2节中的乘法结合律的方式，量化卷积方法相比传统卷积方法在理论上能够取得的加速倍数为：

图4-1展示了ResNet-18各模块的卷积运算为了求取一个对应的特征输出所需要的乘法操作次数的对比，从图中可见，对于ResNet-18的第五个Block结构，加速倍速值高达18。然而，具体到神经网络的FPGA实现时，这一理论值是比较难以获得的。虽然FPGA可以根据实际需求来定制电路，实现资源调度的最优化，但是一旦电路被定制，就无法修改其内部实现。在本文面向的TTA架构ASIP中，卷积属于一个通用计算单元，对于通用计算单元，它应该适用于绝大多数的卷积运算场景，而且满足极端情况的需求。如果想要达到理论的加速效果，需要对各个卷积层进行个性化的定制，这种做法会极大地增加代码的复杂性，影响系统的通用性，不符合本文的ASIP设计思路。本小节将针对大小为3×3×64的卷积核，对它满足量化卷积的通用形式进行分析。

图4-2 ResNet-18各模块的一个卷积运算需要的乘法操作次数

上图中的理论加速的值之所以难以获得，是因为对于量化卷积而言，卷积核中每一个参数所对应的输入元素个数是不同且变化的，这意味着对应到乘法结合律时，每个乘数需要结合的加数的数量不同，造成代码实现上的困难。为了解决该问题，本文工作的前期进行了量化卷积的通用形式的设计方式进行了深入地探究。

方式1非常直观，它直接用一块地址进行部分和的存储，卷积核参数与对应的输入特征同时读入，卷积核参数可以看作是部分和地址的索引，输入特征则与该地址中存放的值进行一次加法操作。采取对称的8比特量化，共需要128块地址空间。考虑到极端情况，也就是该卷积核中所有的参数的量化值都相同时，所有的特征输入需要被累加到同一个部分和中，则该部分和的存放需要的地址空间的大小为：

那么，128个部分和需要的空间大小为：

仅是进行一个卷积核异步的运算所需要的部分和空间。当实现FPGA非卷积核内部的并行计算，也就是多个卷积核同时对某块特征图进行卷积运算，或是某个卷积核对多块特征图进行卷积运算时，该空间大小将成倍增加。这意味着该空间只能存在于片上缓存中，此时数据的读写很容易就成为卷积运算新的瓶颈。另外，该方式难以实现卷积核内部的并行化，相比传统的利用DSP计算乘法的卷积，它对于提高系统并行度几乎没有帮助。

方式2在方式1的基础上，加入了卷积核内部的并行计算。对于一个大小为3×3×64的卷积核，在取数时每次读取若干个卷积核参数及其对应的特征值，此处举8个为例。考虑到8个卷积核参数的大小相同的情况，因此，它们对应的特征值与对应部分和的累加不能同步进行。此时可以考虑两种备选方案，一是对于每一个部分和，都外接一个加法树的结构，加法树的输入数据个数为8。一个8输入的加法树需要的加法器个数为7个。不考虑非卷积核内部的并行，完成一个卷积核的操作需要的128个加法树的结构。在FPGA上，加法器是利用LUT资源实现的，这一数量级对于当前的FPGA上的LUT资源而言比较友好。当采用流水的设计时，方案2对于方案1有8倍的并行度的提升，然而，它并没有解决部分和所需要的巨大的内存资源的问题，并且，当进行卷积核或者特征图之间的并行计算时，需要的加法器的资源也会成倍增长，此时，LUT资源会成为卷积运算新的瓶颈。

上述两种方式共同的问题在于，它们无法快速地取出量化值相同的卷积核参数所对应的特征值进行累加操作，这是因为卷积核参数本身是无序的，而这种无序性对于模块化的运算而言难以优化。基于以上的分析，本文量化卷积核和量化卷积运算的设计方案的创新点在于，通过改变卷积核的结构来实现量化卷积。它的主要思想是将量化绝对值相同的卷积核参数的位置信息记录下来，并聚合在一起，使其结构化。由于卷积核参数与特征值是之间一一对应的，因此可以根据该位置信息对应的地址索引到相同的特征值，保证同一批次的输入可以直接进行累加，相当于将数据的处理在卷积核结构化时完成，而在进行量化卷积运算时节约了大量的逻辑判断与内存资源。下面是量化卷积的具体实现方法。

4.1.4 量化卷积核的表示

对于一个大小为3×3×64的量化后的卷积核，我们可以将该卷积核可以表示成一个类似“字典”（下文中简称字典）的形式，它的关键字为卷积核参数的量化值的绝对值，它的键值为一个固定长度的数组，数组中存放的是该量化值的符号以及它出现的相对地址。由于该固定长度可能无法满足某个量化值所对应的全部长度的存储，因此该字典可能会出现若干个相同的键值，这与传统意义上的“字典”结构键值唯一的特点不同。

关于数组长度的设置，最好的情况是，所有的量化权重出现的次数是相同的，那么将字典的大小设置为128，当数组的长度

时，即可取得最优解。然而，在实际应用时，系统的性能服从木桶原理，受限于最坏的情况。而量化卷积可能面临的最坏情况是，所有的量化权重相同，如果要在一个数组内存放该权重对应的所有特征值的地址，那么数组的长度为

这一长度直接导致字典的存储需要消耗巨大的空间，并且完成这一运算也需要相当多的加法器资源。

实际上，数组的长度与系统的并行效率是直接相关的。数组的长度太短，会造成并行度太低，基于乘法结合律的卷积运算的优化效果无法显现；数组的长度太长，理论的并行度提高了，但是它的提高是建立在大量的加法器资源的浪费的基础上的。本文设置的数组的长度是8，它是综合考量了时间复杂度与空间复杂度之后的折中选择，原因主要有两个。一是利用加法树的实现时累加操作时，5个加数的累加与8个加数的累加所需要的时钟数是相同的，选择数组长度为2的幂次数，有利于充分利用加法树的解构与实现。二是该长度在极端情况依然能有较高的空间利用率，当最好的情况发生时，字典的大小是128，此时数组的利用率为5/8；而最坏的情况发生时，字典的大小

式4-13的含义是，相同的量化值对应的特征需要72个数组存储，而其他127个量化值对应的数组则置空。此时数组的利用率为：

因此，在大小为3×3×64这一常见的卷积核结构中，本文将字典的大小设置为199，数组的长度设置为8，数组中的每个相对地址的存储需要的比特数

其中1是符号代表符号位。因此整个字典结构的存取需要的比特数

而原始的卷积核存取需要的比特数

量化卷积核的字典结构需要的存储资源有所增加，但是不论是寄存器资源或是片上缓存资源，都能够满足该大小的字典结构的存储。当选择合适的并行方式时，卷积核参数不需要反复地进行读写，因此也不存在加重数据带宽负荷的问题。

图4-3与图4.4共同展示了传统卷积核到量化卷积核的字典结构的转换。在图4-3中，卷积核沿着深度方向展开。在图4-4中，数组的长度固定为8，出于可视化的需求，将数组中每个元素前半部分表示卷积核参数的符号，0代表符号位为正，1则相反；后半部分表示卷积核的相对位置。例如，键值为1的数组对应的第一个元素（1，4）的含义即该参数出现在卷积核中的相对位置为4，符号为负，实际值为-1。对于大小为3×3×64的卷积核，传统的卷积方式所需要进行的乘法累加运算的次数为576次，而量化卷积的方式需要进行的加法次数虽然增长为1592次，但是只需要进行199次乘法操作。如上文中提到的，加法器的资源要远远多于乘法器的资源，并且加法操作在实现时要比乘法操作便宜得多，因此量化卷积无论是在资源利用率上还是在计算的并行性上都要领先于传统的卷积。

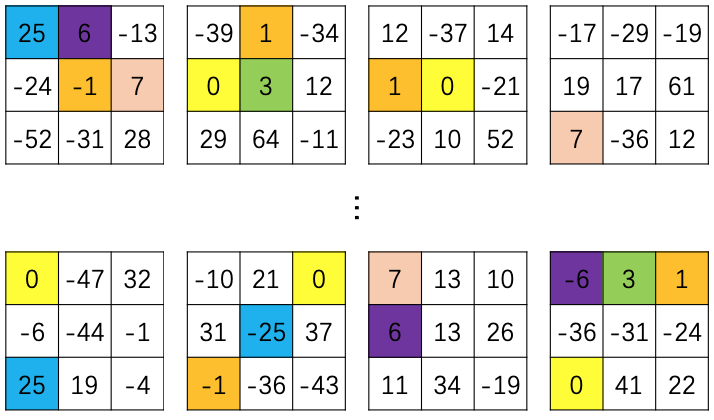


图4-3卷积核沿着深度方向展开

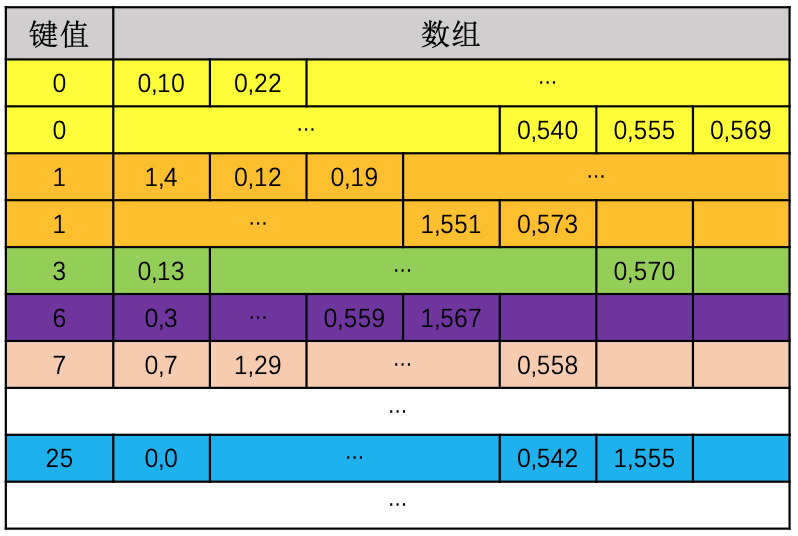


图4-4 量化卷积核的字典结构

4.2 量化卷积的实现

4.2.1 卷积核的分块

FPGA中的片上缓存往往无法满足神经网络所有权重与特征值加载的需求，因此需要设计循环分块方案，对它们进行切分，使其单个分块的大小能够被片上缓存所加载，并最大化利用片上缓存，以减少数据在DRAM（Dynamic Random Access Memory，动态随机存取存储器）通道上既耗时又耗能的传输。本小节讨论的是对卷积核的分块，这一工作除了能解决存储的问题以外，还有一点非常重要的是，通过对卷积核合理的分块，不同大小的卷积核封装成相同的形式，能够增强代码的复用性以及系统的并行度，有较大的收益。

本文在上一节中详细描述了大小为3×3×64的卷积核的设计，并从数学上分析了它的理论性能，实际上，这个卷积核大小并不是随意选取的。在卷积神经网络中，3×3是应用最广泛的卷积核窗口大小，AlexNet、GoogLeNet和ResNet等常用的模型都使用到了该大小的卷积窗口；而64也是比较常见的特征图通道数，还是128、256、512等其他常见特征图通道数的公约数，非常有利于卷积核的分块。因此，本文选择的卷积核分块的大小即为3×3×64。而在真实的网络结构中，卷积核的窗口大小未必是3×3，卷积核的深度也未必是64的倍数。接下来将解决不同的卷积核大小所带来的相关问题。

首先，该问题会存在与网络的第一层。对于网络的首层，它的输入通常是三通道或单通道的图片，虽然此时卷积核的窗口大小通常较大，但是卷积运算所需要的总体操作相对较少。例如，AlexNet的首层的一个卷积运算只需要363次乘法累加操作，ResNet的首层则更少，仅需要147次。在这种情况下，采用量化卷积的方式的收益不大，因此，本文采用的是传统的卷积方式进行网络中第一个卷积层的计算。

其次，一些特殊的网络结构也不适用于量化卷积。例如，NIN网络中的卷积窗口大小仅为1×1，而深度通常不超过512，意味着完成一个卷积运算所需要的乘法累加操作的次数也不超过512。由于这种网络结构中的卷积运算的计算量有限，因此本文没有对齐进行针对性的优化。

最后，对于类似5×5×256等大小的卷积核，本文的量化卷积方式比传统卷积方式更为适用。在传统卷积算法中，可能采取的方法之一是把大小为5×5的卷积窗口用4个3×3的卷积窗口来代替表示[53]；而在本文中，由于卷积核结构的重新定义，真正影响卷积核分块的因素是卷积核参数的个数。对于5×5×256的卷积核，它的卷积核参数的个数为6400个，因此，表示它所需要的3×3×64大小的卷积核分块的个数

而根据传统的卷积核分块方案，需要的分块数

相较而言，量化卷积所需要的分块结构更少，显示出量化卷积方案的一定优势。量化卷积运算要求至少与卷积核参数等量的输入特征，因此，本文中输入特征值的分块大小也为3×3×128。

4.2.2 卷积运算的循环展开

卷积神经网络的前向传播过程包含了三维输入特征图与三维卷积核之间的卷积操作，考虑到卷积核的个数，前向传播包含了六重循环。在理论推导时，习惯将同一通道的二维输入特征图和对应的卷积核参数在结构上都看做是一维数据，因此卷积操作可以看做是由四重循环组成的，图4-5图形化展示了卷积运算的四重循环。

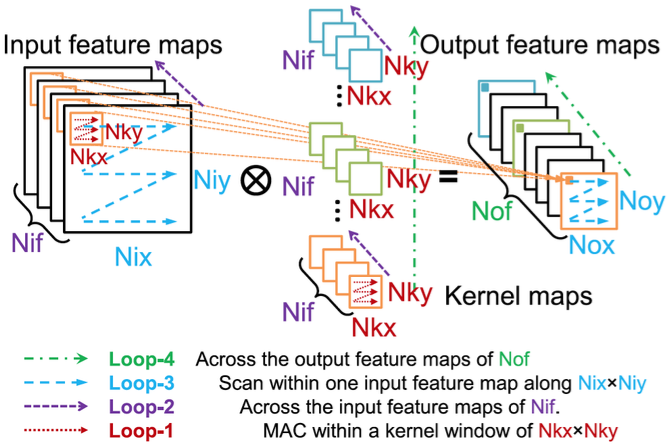


图4-5 卷积运算的四重循环的图形化[40]

图4-5中，在第一重循环中遍历的是卷积核的宽Nkx和高Nky；在第二重循环中遍历的卷积核的深度，同时也是输入特征图的深度Nif；在第三重循环和第四重循环中遍历的分别是输出特征图的宽Nox和高Noy以及输出特征图的深度Nof。这种遍历最直接的方式就是使用For循环实现。图4-6展示了For循环实现一个卷积层运算的伪代码，伪代码中使用了6个For循环来实现一个卷积层的运算。

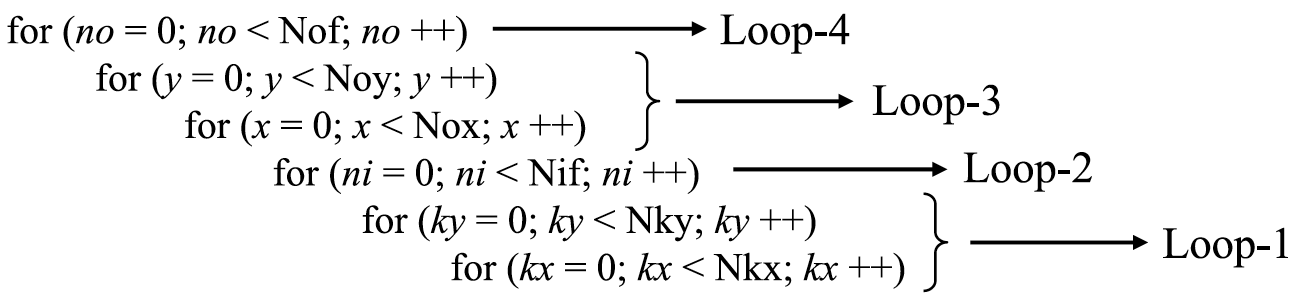


图4-6 卷积运算的四重循环的伪代码

最内层循环将实现卷积核参数与特征输入之间的乘法累加运算：

上式中，S为卷积窗口的步长。For循环的内部实现虽然十分简洁，但是它内部的遍历无法并行处理，因此它是一种以计算性能的降低作为代价来实现编程的简单化的循环实现的思路。然而，前向传播中大规模的计算对并行度有很高的需求，因此在进行卷积操作时需要用到循环展开的思想。

循环展开是一种常用的程序优化技术，它的思路与For循环恰好相反，是一种牺牲程序的尺寸来加快程序的执行速度的优化方法。循环展开的合理应用能够是减少循环开销，提高指令级并行，增强寄存器局部性。由于卷积运算的循环体内部实现的功能是单一的乘法累加操作，不涉及到复杂的依赖关系与调用关系等，十分符合循环展开的条件，利用循环展开进行卷积操作内部的并行如今的神经网络加速技术比较通行的做法。根据展开的循环体的不同，当前对于循环展开的方式主要有四种。

第一种展开方式展开了第1/2/4重循环，使得卷积操作的并行可以应用在卷积核的长宽，输入特征图的深度展开和输出特征图中深度展开；第二种展开方式展开了第2/4重循环，使得卷积操作的并行可以应用在输入特征图和输出特征图的深度展开中；第三种展开方式展开了第1/3重循环，使得卷积操作的并行可以应用在卷积核的平面展开和输出特征图的平面展开；第四种展开方式展开了第3/4重卷积，使得卷积操作的并行可以应用在输出特征图的平面展开和深度展开。

循环展开的选择需要进行对几个问题进行综合的分析，比如说计算时延、部分和存储、数据复用和数据交换等等。由于对这些问题处理的优先级不尽相同，因此产生了上述四种不同的展开方式。对于第一种展开方式，由于它展开第1/2重循环，因此每次并行操作能够获得输出特征图上的一个特征，这样做最大的优点在于减小了部分和的存储；同时它也展开了第4重循环，使得卷积核与输入特征图都能够得到复用。它的问题在于不同卷积层的卷积核大小是不同的，功能单元需要根据卷积核的不同重新配置参数，增加了控制的复杂度。对于第二种展开方式，它没有展开第1重循环，避免了第一种方式中卷积核大小不同带来的问题，但是随之而来的是巨大的部分和存储的需求；另一方面，网络的前几层中特征图的深度通常较浅，特别是第一层，输入的深度通常只有1维或3维，导致计算的并行性得不到充分地发挥。对于第三种展开方式，循环在卷积核和输出特征图的平面展开，较大限度地挖掘了硬件在并行性能，但是循环没有在输出特征图的深度维度上展开，由此带来的直接影响是输入特征图的数据复用非常有限，进而导致系统的数据交换会比较频繁。对于第四种展开方式，第3/4重循环都被展开，能够使硬件的并行能力得到充分的发挥，并且输入特征图和卷积核参数都能够得到复用，但是比较显而易见的问题是，它在卷积核上维度上的运算需要通过遍历来完成，因此它计算出下一层的一个特征输出所需要的时延以及部分和存储的需求都是最大的。

通过对当前较常用的几种循环展开方式的特点进行学习和分析，并结合上一节中提出的量化卷积操作的特点，本文并没有采用以上四种展开方式的任意一种，而是设计了一种对第1/2/3/4重循环进行展开的方案。原因之一是对卷积核的字典化结构实现以后，原本由卷积核大小的不同产生的问题就不复存在，第一种循环存在的问题就自然地消除了，因此可以优先考虑展开第1/2重循环。原因之二在于展开当数据的读取能够满足要求时，完成第3/4重循环的展开能够最大化卷积核参数与特征值的复用。4.2.2节将给出数据交换的方案。

4.2.2 前向传播的数据交换

图4-7展示了FPGA的存储资源与神经网络前向运算之间的关系。神经网络前向运算的数据流，是由片外存储进入片上缓存，再由片上缓存进入寄存器与处理引擎中的，经过一系列的运算之后，处理器将输出特征或结果返回到片上缓存或片外存储中。从片外存储到片上缓存再到寄存器，这三种存储资源的大小是递减的，而数据交换的速率则是递增的，如何实现数据交换的优化也是神经网络的前向传播中需要考虑的问题。本小节将结合前文中的循环分块与循环展开的方案，分析数据交换的策略。

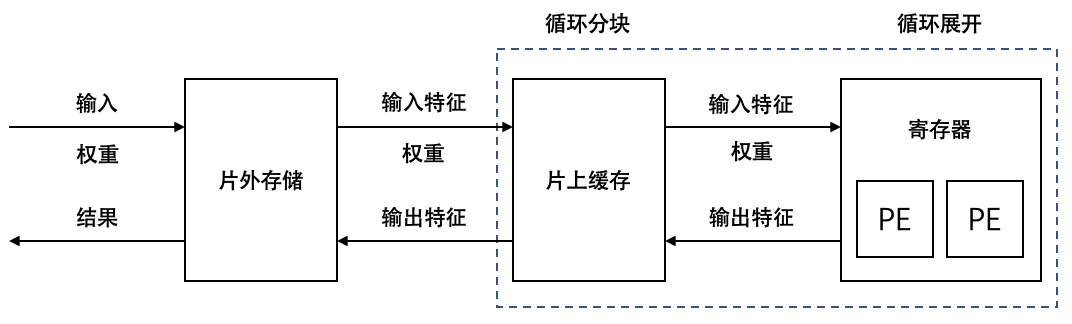


图4-7 FPGA的存储资源与神经网络前向运算的关系

首先讨论部分和的存取的问题。部分和是卷积运算的中间结果，当优先第1/2重循环完全展开时，部分和存储的需求最小。本文选择的展开方案恰好满足最小的部分和的要求。对于深度大于64的卷积核，一次量化卷积无法求取相应的特征输出，为了加速运算，部分和结果将会存储在寄存器中，方便DSP模块的调用。

其次讨论片上缓存的读写。片上缓存资源主要指的是FPGA开发板的BRAM（Block Random Access Memory，随机存取存储器）资源。当前的FPGA开发板通常拥有数百块BRAM资源，每个BRAM单个时钟可以完成单口16比特、双口32比特的数据传输。这里按照量化卷积要求的数据读写的基本单位讨论两方面的问题一是如何读取量化卷积核，二是如何读写特征值。

对于量化卷积核的读写，如果要完全展开第1/2重循环，片上缓存每个时钟要向寄存器的功能单元中送入199×8个11比特的数，这超出了一般的FPGA开发板中BRAM读写能力，因此，无法完全展开第1/2重循环。本文给出的权重读写方案是，将字典结构的权重的199个数组分别存放在199个BRAM中，每个时钟199个BRAM同时向寄存器送入199对的数组元素，每4个时钟完成一个量化卷积核的传输。

对于特征值的读写，第1/2重循环的一个卷积运算需要的特征图大小为3×3×64。本文的方案是将特征窗口按列展开，重构为3×(3×64)的结构，存放在192个BRAM中，每个时钟192个BRAM同时向寄存器送入3个8比特的特征值。当卷积步长为1时，前3个时钟完成一次卷积运算所需的特征窗口的传输，而后每一个时钟都可以产生一个新的特征窗口；对于特征值的写入，由于本文的量化卷积优先展开第1/2重循环，且部分和存储在寄存器中，因此特征值写入的需求较小。图4-8展示了本文中BRAM上的数据读取形式。

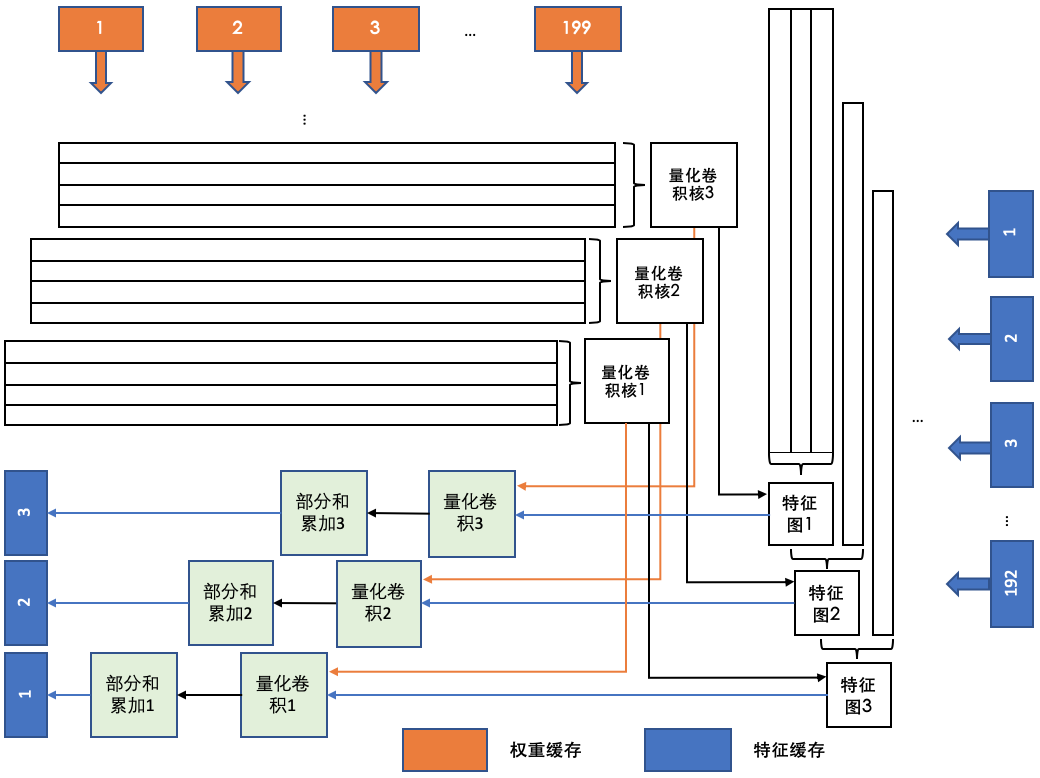


图4-8 BRAM上的数据读取

最后讨论片外存储的读写。片外存储的读写对比片上缓存的读写，是非常耗时且非常耗能的。然而，片上缓存无法满足整个神经网络的权重与特征值的存储，因此，片外存储资源又是十分重要的资源。最小化片外存储的读写的方法非常简单，只需要保证片上缓存能够覆盖一个卷积层运算所需要的全部权重与特征输入即可。在实际的FPGA实现时，该条件也同样无法得到保证。在这种情况下，实现片外存储的读写优化需要尽可能地使权重或输入特征二者中任意一个能被完全利用。对此，本文的处理方式是加入对某个卷积层一个卷积核总大小与特征图总大小的判断。当片上缓存大小对二者都能够满足时，优先在片上缓存中写入更大的一方；当片上缓存大小能够满足二者之一时，优先在片上缓存中写入能够满足的一方；当二者均不能够被满足，则用更多的片外存储的资源来买组该层卷积的运算。由于部分和存储在寄存器中，因此片外存储的写入同片上缓存的写入一样，达到了最优化。

4.3 量化卷积加速器的整体结构

图4-8展示了量化卷积加速器的整体结构。其中，黑色细线代表控制信号，橙色粗线代表权重数据流，蓝色粗线代表特征数据流，绿色方框代表运算单元。

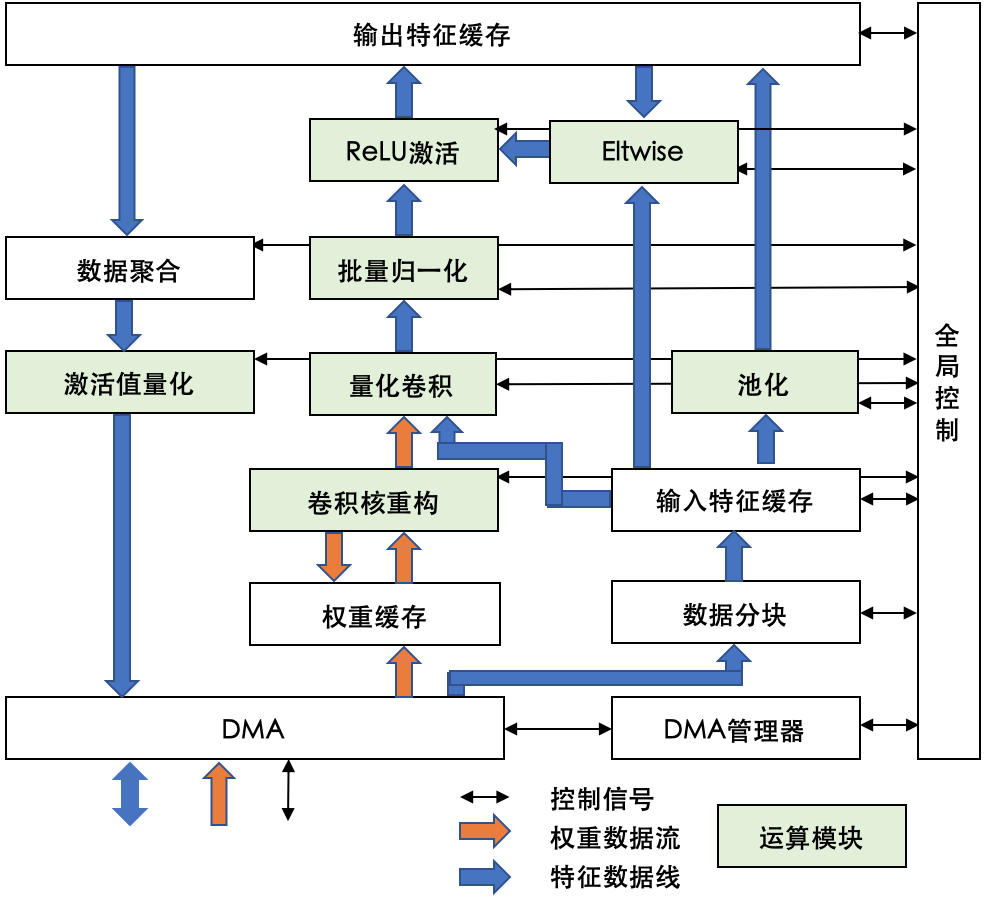


图4-8 量化卷积加速器的整体结构

对于卷积神经网络的前向传播，控制信号控制权重与特征从片外存储送入到片上缓存中，片上缓存的资源会进一步被送入各功能单元中。卷积核会经历字典化重构，进而分块后的特征值进行量化卷积。由于卷积核没有被完全的复用，因此重构后的卷积核将重新写入片上缓存中等待下次调用。卷积运算结束之后，依次进行批量归一化与ReLU激活。Eltwise单元进行的是元素级的操作，主要面向ResNet中层间相加的场景。激活值被存放在片上缓存中，由于激活值量化的尺度是以层为单位的，因此当一个卷积层的运算完成后，各分块的数据被聚合后，激活值的量化才会开始。量化后的激活值会再次被送回片外存储中，等待下一层的调用。当图4-8的过程发生在前向推理时，卷积核重构与批量归一化的过程可以被省略。

图4-9进一步描述了量化卷积功能单元的结构。假设输入特征已经从片上缓存写入寄存器中。量化卷积核根据保留的地址信息从寄存器中取出某个权重对应的特征值，并将它们送入加法树中进行求和，每个加法树组将该量化卷积核的字典结构中199个权重所对应的和。随后，每个加法树组所得到的和送入DSP模块，与该权重进行相乘，相乘的结果与上一次的部分和进行累加，直至量化卷积核的分块遍历完成，将DSP中保留的和作为输出特征值送如到下一个模块中。由于卷积运算展开了第三层与第四层循环，因此卷积核参数与输入特征值都能得到复用。

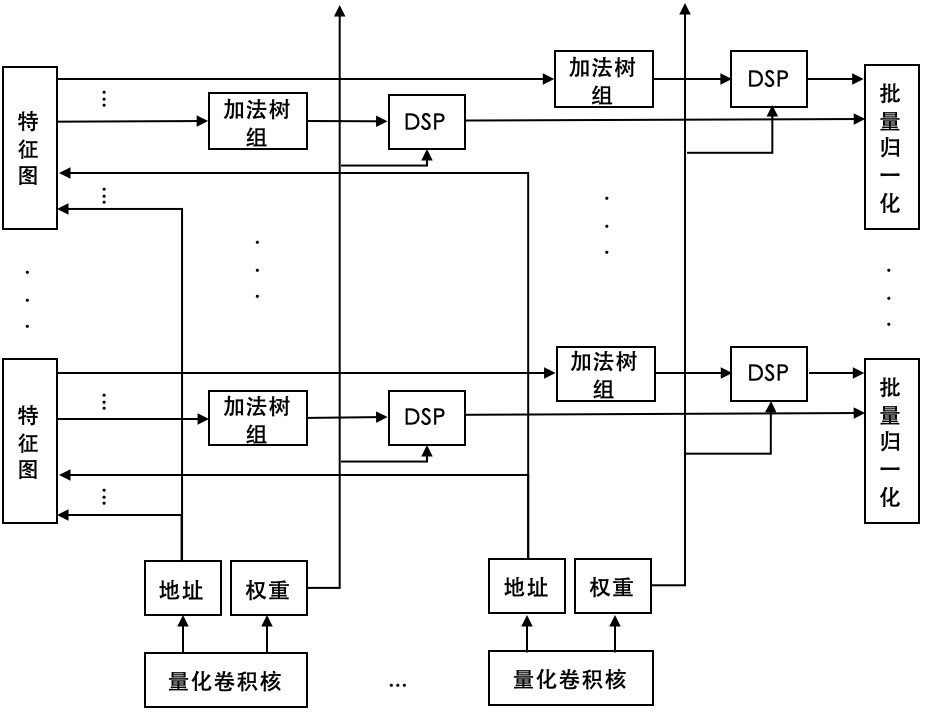


图4-9 量化卷积单元的结构

第五章 实验结果与分析

5.1 近似批量归一化的实验

近似批量归一化算法在3.3.2节中，为了验证它的有效性，本小节在相同的实验配置下，比较了它与传统的批量归一化算法之间在ImageNet上的性能差距。ImageNet是一个包含1000类图像，126万张训练集图像，5万张验证集图像和10万张测试集图像的大型数据集，也是验证计算机视觉任务成果最重要且最通用的数据集之一。本小节中的实验基于TensorFlow框架[52]实现，部分重要的实验参数如下：训练机是4块NVIDIA 1080Ti显卡，对于每块显卡上的一个批次，它的批量大小是64，总的批量大小是256。近似批量归一化算法中的常数c设置为2，优化器的选择都是Adam优化器，学习率的选择上，AlexNet模型结构设置为0.001，ResNet模型结构设置为0.002。在送入网络前图像大小被调整为224×224，并且仅使用单次中心裁剪的策略。

图5-1展示了带有批量归一化算法的AlexNet模型的实验结果。从图中主要可以提取出两点主要信息。首先，定性地来分析，在训练的各个阶段，近似的批量归一化算法都与传统的批量归一化算法都有相近的收敛效果。3.3.2节中提到，BN算法的作用是加快收敛速度，提升训练稳定性。从验证集上错误率曲线下降的趋势可以看出，近似的批量归一化算法同样具备了这两点作用。其次，定量地来看，经过了近90次迭代，模型的损失在训练集上都不再下降时，两者在验证集上取得了几乎相同的Top-1和Top-5错误率。具体的数据是，加入了传统的批量归一化算法的网络，在训练结束时能够取得20.50%的Top-5错误率以及42.33%的Top-1错误率，而加入了近似的批量归一化算法的网络，在相同的迭代周期内，能够取得20.90%的Top-5错误率与42.79%的Top-1错误率，虽然从数据上看，近似的批量归一化算法的性能相比传统的批量归一化算法有大约0.5个百分点的下降，但是二者的性能都要优于不加入批量归一化算法的AlexNet网络的原型在ImageNet数据集上所取得Top-1错误率大于45%的成绩。

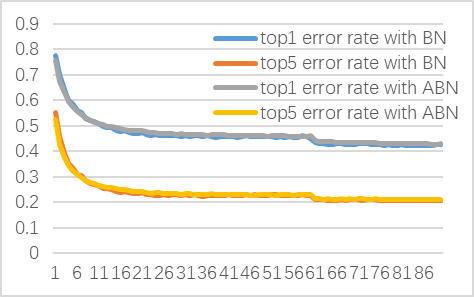


图5-1 近似批量归一化在AlexNet上的错误率曲线

图5-2展示的是近似的批量归一化算法在ResNet模型上的表现。该图所呈现的结果与图一基本保持了一致。在ImageNet验证集上的结果显示，两者在Top-5错误率上的分值分别是10.33%和11.71%，在Top-1错误率上的分值则是29.38%和29.53%。虽然传统的批量归一化算法在两项重要数据的得分均领先于近似的批量归一化算法，但是分差在一个可以接受的范围内。尤其是在Top-1错误率上，两者的差距非常有限。

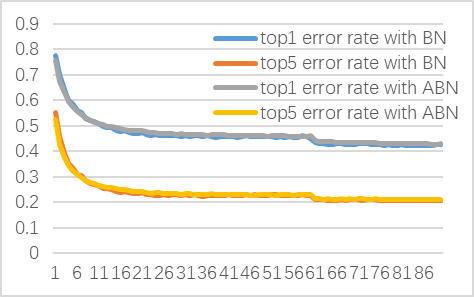


图5-2 近似批量归一化在ResNet-18上的错误率曲线

分析两张图中错误率曲线的细节还能发现，在应用传统的批量归一化算法时，验证集上错误率的下降还存在抖动的现象；而应用近似的批量归一化算法时，错误率的下降呈现出一种更加平稳的趋势。这一现象在ResNet模型结构中尤其明显。本文无法从数学上推导造成这一现象的原因，然而，对于深度学习任务而言，训练阶段模型更加平稳地收敛对于测试阶段模型的性能有正向的作用。

5.2 量化方案的实验

本文的第三章介绍了本文提出的量化方案，其中包括了权重、激活值与梯度的量化。本节对整体量化方案在SVHN、CIFAR-10与ImageNet三个数据集上进行了实验。

5.2.1 SVHN数据集上的实验

SVHN数据集是一个真实的数字识别数据集，由谷歌街景图像中的房屋编号照片组成，其中包含训练图像共73257张，测试图像共26032张。SVHN的数据及相对较小，任务相对简单，本文选取它用于评估量化方案对与小型任务的适用性能。本文在预处理阶段将数据集裁剪为以单个字符为中心的32×32的像素彩色图像，在送入网络之前将图像像素调整到40×40。实验选用的模型结构由7个卷积层和1个全连接层组成，最大的特征通道数为128。实验环境为单块NVIDIA 1080 Ti显卡，优化规则为Adam优化器，初始化学习率为0.001。实验的结果如图5-3所示。

图5-3中三条曲线分别反映带有使用传统批量归一化算法的非量化网络、使用近似批量归一化算法的非量化网络以及使用近似批量归一化算法的量化网络的在SVHN验证集上Top-1错误率的情况。由于SVHN数据集对于神经网络是一个比较简单的学习任务，网络能在短时间内达到较高的准确率，因此图中的刻度较小，视觉上三条曲线都有起伏，实际上三者的Top-1错误率都在2.5%-3%之间，没有明显的差距。此外，从整个训练过程来看，三者的下降趋势也是相仿的。该图证实了本文提出的量化方案在小型的图像分类任务上能够取得与全精度的量化网络相当的准确率。

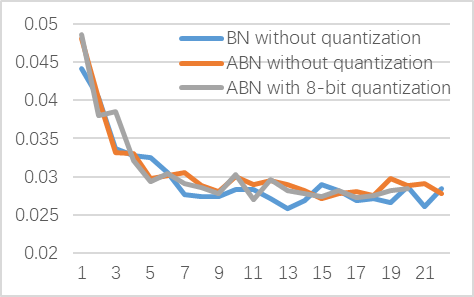


图5-3 三种训练方案在SVHN数据集上的错误率曲线

5.2.2 CIFAR-10数据集上的实验

CIFAR-10数据集同样是评估深度学习算法性能较常使用的数据集之一，它由包含了飞机、汽车、狗等10个类的60000个32×32像素的彩色图像组成，每个类有6000个图像，按照5：1分配训练集与测试集。本文使用ResNet-18网络结构在CIFAR-10数据集上完成了训练，实验环境为单块NVIDIA 1080 Ti显卡，优化规则为Adam优化器，初始化学习率为0.002。实验结果如图5-4所示。

图5-4反映的实验结果与SVHN数据集上的实验结果相似。在整个训练过程中，三条曲线几乎完全重叠，意味着三者在验证集上的是几乎相同的。在经历了100个迭代周期后，验证集上的准确率基本保持稳定。训练在200个迭代周期后终止，此时，三种方案的错误率均为7%左右，使用本文所提到的量化方案所产生的结果甚至稍好于全精度的训练方案。

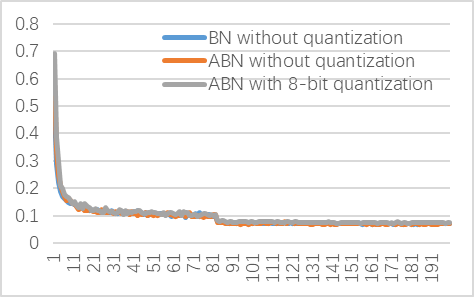


图5-4 三种训练方案在CIFAR-10数据集上的错误率曲线

5.2.3 ImageNet数据集上的实验

本节的前两个小节的结果说明了使用本文的量化方案训练的模型在较小的数据集上能够取得与全精度的模型相当的成绩。实际上，SVHN与CIFAR-10虽然数据较少且任务较简单，但是它们与嵌入式设备所要面临的场景与需求更加贴近。尽管如此，为了验证本文的量化方案在更难的数据集上的表现，本文同样在ImageNet数据集进行了实验。实验选择的模型结构是AlexNet-BN，实验的相关设定与5.1节相同。验证集上的Top-1下降曲线分别展示在图5-5中。图5-5所呈现的规律与上文的各实验类似，并且量化方案带来的精度损失上要稍大于在SVHN与CIFAR-10上的实验。由于ImageNet数据集的分类任务难度大于前两者，因此产生该结果符合预期。

表5-1展示了集中量化方案对AlexNet模型进行量化是在ImageNet数据集上的表现。在使用本文提出的8比特量化方案进行训练的模型中，Top-1与Top-5的错误率的数据分别是43.58%和21.46%，比全精度模型分别增加了1.25%与0.96%，虽然存在一定差距，但是对比其他量化方案动辄10%以上的精度损失，本文的量化方案取得了相对较好的结果。

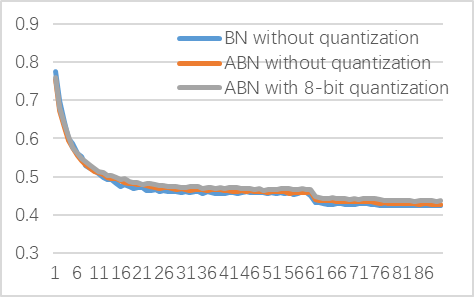


图5-5 三种训练方案在ImageNet数据集上的Top-1错误率曲线

表5-1 几种量化方案使用AlexNet模型在ImageNet数据集上准确率

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 量化方案 | BNN | XNOR-Net | DoReFa-Net | 本文 | 全精度 |
| Top-1准确率 | 27.9% | 44.2% | 53.0% | 56.42% | 57.67% |
| Top-5准确率 | 50.2% | 69.2% | - | 78.54% | 79.50% |

5.3 量化卷积运算的实验

5.3.1 实验配置

本文的第四章详细介绍了量化卷积的原理与实现，本节就ResNet-18的第三个Block的第一个卷积层进行了传统卷积与量化卷积的实验对比。该卷积层的卷积核大小为3×3×64，个数为128个，输入特征图的大小28×28×64，卷积步长为1。实验仿真在Vivado 2016.2硬件仿真平台上进行，实验面向的硬件开发平台为Xilinx kintex-7 XC7K325T（以下简称“K325T”）。K325T平台拥有840个DSP模块，理论上可以同步完成840个乘法操作。对于传统的卷积运算，本文将它的第1/2重循环展开的参数设置为3×3×64，与量化卷积运算的第1/2重循环展开相同。由于展开第1/2重循环需要576个DSP模块，840个DSP模块无法支持两个第1/2重循环展开的并行，因此本文对传统的卷积运算没有进行第3重与第4重循环的展开。

对于量化卷积运算，第1/2重循环完全展开需要199个DSP模块，因此K325T平台最多可以同时支持4个第1/2重循环的并行，此时一个时钟需要调用796个DSP模块。此外，量化卷积还需要796个加法树结构的支持。8个数的累加需要的加法树深度为3。加法树的第一层结构的输入是8个9比特的特征，其中包括1比特的符号位与8比特的量化比特，第一层加法完成后得到4个10比特的输出，同时作为第二层的输入，第二层得到2个11比特的输出作为第三层的输入。加法器最终的输出是一个12比特的数。这样一个加法树结构需要的LUT资源数为74，而K325T平台上拥有的LUT资源的个数是203800，因此LUT资源也完全能够满足796个加法树结构并行。

5.3.2 资源占用的对比

本小节按照上述配置进行仿真实验，以对比传统卷积与量化卷积在实现卷积运算时对K325T开发板的片上资源占用的情况。实验结果中的关键资源占用情况表5-2所示，由于量化卷积引入了更高的并行度造成的，因此其在实现时使用的关键资源都要高于传统卷积。对于LUT资源，由于量化卷积引入了大量的加法运算，而加法器的需要依赖于LUT资源实现，因此量化卷积对LUT资源的需求要高于传统卷积。对于FF资源，主要的差距体现在量化卷积核的字典结构与传统卷积核的原始结构之间对寄存器的需求的差异。对于DSP资源，它的不同体现了系统乘法运算的并行度的差异。对于BRAM资源，它则是由特征值与卷积核参数的存储的实际需求决定的。

表5-2 传统卷积与量化卷积资源使用情况对比

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 卷积方式 | 关键资源 | | | |
| LUT | FF | DSP | BRAM |
| 传统卷积 | 15002 | 28096 | 576 | 288 |
| 量化卷积 | 118912 | 115768 | 796 | 391 |

图5-6则展示了传统卷积与量化卷积关键资源的使用在占K325T平台的整体比例。从图中可以看出，量化卷积时，除了DSP资源的占用率超过90%以外，其他资源的占用率都在一个比较合理的水平。其中，LUT资源的占用率为58.3%，FF资源的占用率为28.4%，BRAM资源的占用率为87.8%。实际上，DSP模块的使用情况是可以根据不同的开发板的实际资源情况动态分配，并能保证实现的，因此较高的DSP资源占用率并不会成为本文量化卷积方案实现时的障碍。

图5-6的结果可以表明对于相同配置的硬件，本文的量化卷积方案能够更大限度地利用开发板上拥有的资源进行卷积运算，并且在达到4倍并行度优化的情况下，能够通过硬件仿真平台的综合实验。

图5-6 传统卷积与量化卷积资源占用情况

本小节没有展示量化卷积与传统卷积的运行时间的差距，是由于实验中没有为本文的传统卷积方案设计最大的并行度，使得运行时间的比较并不公平。然而，即便在K325T上对传统卷积方案进行最优化定制，使其能够充分调用DSP模块，将乘法累加操作的并行度提高到840，传统卷积方案能够取得的并行度依然低于量化卷积方案的并行度。后者的并行度为：

也就是说，在K325T上，量化卷积运算的通用形式相对于传统卷积能够有近3倍的并行度的提升。

5.3.3 片上功率的对比

本小节利用Xilinx Power Estimator工具，对二者的片上功率进行对比。本实验中，传统卷积的片上总功率为3.316W，量化卷积的片上总功率为5.572W，K325T开发板上各关键资源的片上功率情况如表5-3所示。

表5-3 传统卷积与量化卷积关键资源的片上功率对比

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 卷积方式 | 片上功率 | 关键资源 | | | | | |
| CLOCK | LOGIC | BRAM | DSP | I/O | Static |
| 传统卷积 | 功率（W） | 0.388 | 0.332 | 0.823 | 0.423 | 0.775 | 0.216 |
| 占比（%） | 12 | 10 | 25 | 6 | 23 | 7 |
| 量化卷积 | 功率（W） | 1.052 | 1.429 | 1.138 | 0.287 | 0.775 | 0.275 |
| 占比（%） | 19 | 26 | 21 | 5 | 14 | 5 |

图5-7对传统卷积与量化卷积关键资源的片上功率对比的实验结果进行了可视化。结合表5-3和图5-7呈现的各项数据可以发现，量化卷积的 在DSP功率外的各项功率均要高于传统的卷积运算，尤其是LOGIC资源的功率以及功率占比要远远高于传统的卷积运算。这个结果符合本文中的量化卷积思路用LUT资源换取DSP资源的思想。对比总的片上功率发现，量化卷积的总功率约为传统卷积的1.8倍，考虑到量化卷积的并行度是传统卷积的4倍，因此，在本实验中，量化卷积在完成相同层级的卷积操作时，对比传统卷积能够有约2倍的计算能效比的提升。这个结果对于部署在嵌入式设备中，对低功耗存在实际需求的深度学习任务的硬件实现有很大的好处。

图5-7 传统卷积与量化卷积关键资源的片上功率对比

第六章 总结与展望

6.1 本文工作总结

本文面向TTA架构ASIP设计的深度神经网络优化的工作，经历了确定论文题目、调研相关工作、设计工作方案、进行理论推导、完成实验验证、完成中期报告、完善论文工作到当前完成论文等几个关键阶段，为期总计超过一年。前期工作的成果主要在本文的第一章与第二章体现，它们是本文工作的基础。而本文的主要工作可以分为两个部分，其一是卷积神经网络的量化，其二是卷积运算的加速。二者是相辅相成，互为补充的。

对于第一部分的工作而言，卷积神经网络的量化是当前深度学习研究领域非常热门的一个问题。本文对此进行了大量的调研工作，发现尽管大量的工作聚集在卷积神经网络的量化中，但量化工作仍有可创新与可改良的部分。本文以模型支持、算法优化和硬件适应作为本文量化工作的出发点，完成了本文的量化方案的设计。量化工作在本文的第三章中详细描述，其创新之处在于：

（1）采用了权重的对称仿射量化策略，既保证了量化精度的最大化利用，又为本文的加速策略提供了基础。

（2）引入了近似的批量归一化算法，在保证近似精度的条件下，降低了批量归一化的运算开销与内存需求，更适用于硬件的实现。

（3）提出了变精度的梯度量化策略，取消了权重更新时无谓的梯度量化步骤，既不影响硬件的整体负载，还能为权重的更新提供更高的精度。

通过以上几点创新，本文的8比特量化方案既能够支持当下常用的AlexNet、ResNet等主流模型，并将绝大多数的运算转换为了线性运算，同时也为硬件加速方案的设计提供了支持。符合本文量化工作的出发点。

对于第二部分的工作而言，本文通过查阅相关文献发现，卷积神经网络的FPGA实现的并行度很大程度上有开发板上的DSP资源决定。DSP资源虽然能够像加法运算等简单操作一样在一个时钟内得到运算结果，然而DSP资源相对有限且运算开销要大于加法运算。本文提出了基于乘法结合律的量化卷积方案的设计，将DSP资源的需求减少到传统卷积运算的1/3。优化工作的细节在本文的第四章中展开。其不仅详细介绍了传统卷积与量化卷积的原理，并未它们进行了全面的对比，也对卷积运算的循环分块、循环展开以及数据交换进行了分析。第四章的最后一节给出了量化卷积加速器的实现方案。

第五章中开展了相关的实验以验证本文提出的量化方案与优化方案的可行性。实验结果表明，本文的8比特量化方案能够取得与全精度方案相近的模型准确率；本文的量化卷积方案能够在仿真平台上通过硬件系统仿真。

6.2 不足与展望

实现深度神经网络专用处理器的设计与实现是一项非常工程量巨大的工作，当前业界也仅有非常少数的AI公司在从事相关的工作并有所产出，但是对于未来万物互联与万物智能的生活场景，深度神经网络专用处理器是深度学习成果落地的重要途径之一。由于本文作者的能力与毕设的时间有限，本文的工作还存在一些不足，需要进一步改进。

（1）本文提出的量化方案与加速方案仅完成了软件级别的仿真。虽然本文在理论推导阶段始终结合着硬件实现的考虑，通用处理器平台上的实验结果或者软件仿真结果也达到了预期的效果，但是，对于硬件实现过程中可能存在的未知的问题，本文无法给出解释与处理方案。实现基于TTA架构的专用处理器的实现，将是本文的未来的长期工作和本文的价值所在。

（2）本文的量化方案中包含了梯度的量化策略，目的是能够满足在深度神经网络专用处理器上完成模型训练的需求。然而，本文仅针对神经网络的前向传播进行加速方案的设计。虽然前向推理是深度学习更加常用的任务，但是反向传播过程中包含的大规模运算同样存在可以优化空间。如果能够对反向传播过程进行系统的分析，并提出响应的加速方案，本文的优化方案将得到更加完整的补充。

参考文献

1. Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.
2. Deng J, Dong W, Socher R, et al. Imagenet: A large-scale hierarchical image database[C]//Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. Ieee, 2009: 248-255.
3. LeCun Y. LeNet-5, convolutional neural networks[J]. URL: http://yann.lecun. com/exdb/lenet, 2015: 20.
4. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.
5. Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 1-9.
6. Lin M, Chen Q, Yan S. Network in network[J]. arXiv preprint arXiv:1312.4400, 2013.
7. Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[J]. arXiv preprint arXiv:1502.03167, 2015.
8. 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.
9. Jaderberg M, Vedaldi A, Zisserman A. Speeding up convolutional neural networks with low rank expansions[J]. arXiv preprint arXiv:1405.3866, 2014.
10. Chen W, Wilson J, Tyree S, et al. Compressing neural networks with the hashing trick[C]//International Conference on Machine Learning. 2015: 2285-2294.
11. Han S, Mao H, Dally W J. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding[J]. arXiv preprint arXiv:1510.00149, 2015.
12. Guo Y, Yao A, Chen Y. Dynamic network surgery for efficient dnns[C]//Advances In Neural Information Processing Systems. 2016: 1379-1387.
13. Lebedev V, Lempitsky V. Fast convnets using group-wise brain damage[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016: 2554-2564.
14. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network[J]. arXiv preprint arXiv:1503.02531, 2015.
15. Romero A, Ballas N, Kahou S E, et al. Fitnets: Hints for thin deep nets[J]. arXiv preprint arXiv:1412.6550, 2014.
16. Courbariaux M, Bengio Y, David J P. Binaryconnect: Training deep neural networks with binary weights during propagations[C]//Advances in neural information processing systems. 2015: 3123-3131.
17. Rastegari M, Ordonez V, Redmon J, et al. Xnor-net: Imagenet classification using binary convolutional neural networks[C]//European Conference on Computer Vision. Springer, Cham, 2016: 525-542.
18. Courbariaux M, Hubara I, Soudry D, et al. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1[J]. arXiv preprint arXiv:1602.02830, 2016.
19. Zhu C, Han S, Mao H, et al. Trained ternary quantization[J]. arXiv preprint arXiv:1612.01064, 2016.
20. Zhou S, Wu Y, Ni Z, et al. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients[J]. arXiv preprint arXiv:1606.06160, 2016.
21. Howard A G, Zhu M, Chen B, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications[J]. arXiv preprint arXiv:1704.04861, 2017.
22. Zhang X, Zhou X, Lin M, et al. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices[J]. 2017.
23. Bottleson J, Kim S Y, Andrews J, et al. clcaffe: Opencl accelerated caffe for convolutional neural networks[C]//2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). IEEE, 2016: 50-57.
24. DiCecco R, Lacey G, Vasiljevic J, et al. Caffeinated FPGAs: FPGA framework for convolutional neural networks[C]//Field-Programmable Technology (FPT), 2016 International Conference on. IEEE, 2016: 265-268.
25. Ko J H, Mudassar B, Na T, et al. Design of an energy-efficient accelerator for training of convolutional neural networks using frequency-domain computation[C]//Design Automation Conference (DAC), 2017 54th ACM/EDAC/IEEE. IEEE, 2017: 1-6.
26. 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. ACM, 2015: 161-170.
27. Rahman A, Lee J, Choi K. Efficient FPGA acceleration of convolutional neural networks using logical-3D compute array[C]//Design, Automation & Test in Europe Conference & Exhibition (DATE), 2016. IEEE, 2016: 1393-1398.
28. Zhang C, Wu D, Sun J, et al. Energy-efficient CNN implementation on a deeply pipelined FPGA cluster[C]//Proceedings of the 2016 International Symposium on Low Power Electronics and Design. ACM, 2016: 326-331.
29. 姜雷. 传输触发体系结构处理器的软件流水[D].上海交通大学,2009.
30. Jegou H, Douze M, Schmid C. Product quantization for nearest neighbor search[J]. IEEE transactions on pattern analysis and machine intelligence, 2011, 33(1): 117-128.
31. Faraone J, Fraser N, Blott M, et al. SYQ: Learning Symmetric Quantization For Efficient Deep Neural Networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 4300-4309.
32. Faraone J, Fraser N, Blott M, et al. SYQ: Learning Symmetric Quantization For Efficient Deep Neural Networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 4300-4309.
33. Hou L, Kwok J T. Loss-aware Weight Quantization of Deep Networks[J]. arXiv preprint arXiv:1802.08635, 2018.
34. Jacob B, Kligys S, Chen B, et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference[J]. arXiv preprint arXiv:1712.05877, 2017.
35. Zhou A, Yao A, Guo Y, et al. Incremental network quantization: Towards lossless cnns with low-precision weights[J]. arXiv preprint arXiv:1702.03044, 2017.
36. Bengio Y, Léonard N, Courville A. Estimating or propagating gradients through stochastic neurons for conditional computation[J]. arXiv preprint arXiv:1308.3432, 2013.
37. Memon T D, Pathan A. An approach to LUT based multiplier for short word length DSP systems[C]//Signals and Systems (ICSigSys), 2018 International Conference on. IEEE, 2018: 276-280.
38. Véstias M, Duarte R P, de Sousa J T, et al. Parallel dot-products for deep learning on FPGA[C]//Field Programmable Logic and Applications (FPL), 2017 27th International Conference on. IEEE, 2017: 1-4.
39. Dettmers T. 8-bit approximations for parallelism in deep learning[J]. arXiv preprint arXiv:1511.04561, 2015.
40. Ma Y, Cao Y, Vrudhula S, et al. Optimizing the Convolution Operation to Accelerate Deep Neural Networks on FPGA[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2018 (99): 1-14.
41. Ma Y, Cao Y, Vrudhula S, et al. An automatic RTL compiler for high-throughput FPGA implementation of diverse deep convolutional neural networks[C]// 2017 27th International Conference on Field Programmable Logic and Applications (FPL). IEEE, 2017.
42. Ba J L, Kiros J R, Hinton G E. Layer normalization[J]. arXiv preprint arXiv:1607.06450, 2016.
43. Vedaldi V L D U A. Instance Normalization: The Missing Ingredient for Fast Stylization[J]. arXiv preprint arXiv:1607.08022, 2016.
44. Wu Y, He K. Group normalization[J]. arXiv preprint arXiv:1803.08494, 2018.
45. Andersen J. Theory, Operation, and Application of Neural Networks[J]. 2018.
46. Krishnamoorthi R. Quantizing deep convolutional networks for efficient inference: A whitepaper[J]. arXiv preprint arXiv:1806.08342, 2018.
47. Cheng J, Wang P, Li G, et al. Recent advances in efficient computation of deep convolutional neural networks[J]. Frontiers of Information Technology & Electronic Engineering, 2018, 19(1): 64-77.
48. Abdelouahab K, Pelcat M, Serot J, et al. Accelerating CNN inference on FPGAs: A Survey[J]. arXiv preprint arXiv:1806.01683, 2018.
49. Guo Y. A survey on methods and theories of quantized neural networks[J]. arXiv preprint arXiv:1808.04752, 2018.
50. Blott M, Preußer T B, Fraser N J, et al. FINN-R: An End-to-End Deep-Learning Framework for Fast Exploration of Quantized Neural Networks[J]. ACM Transactions on Reconfigurable Technology and Systems (TRETS), 2018, 11(3): 16.
51. Venieris S I, Kouris A, Bouganis C S. Toolflows for Mapping Convolutional Neural Networks on FPGAs: A Survey and Future Directions[J]. ACM Computing Surveys (CSUR), 2018, 51(3): 56.
52. Wu, Y., et al.: Tensorpack. https://github.com/tensorpack/, 2016.
53. 朱学亮，柴志雷, 钟传杰, et al. 基于FPGA的图像卷积IP核的设计与实现[J]. 微电子学与计算机, 2011, 28(6):188-192.

致谢

攻读学位期间发表的学术论文目录

[1]