计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2020,29(2):49−57 [doi: 10.15888/j.cnki.csa.007257]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

主流卷积神经网络的硬件设计与性能分析①

徐青青, 安　虹, 武　铮, 金　旭

(中国科学技术大学 计算机科学与技术学院, 合肥 230027)
通讯作者: 徐青青, E-mail: tsqua@mail.ustc.edu.cn

摘　要: 作为深度学习领域中最具有影响力的网络结构之一, 卷积神经网络朝着更深更复杂的方向发展, 对硬件计算能力提出了更高的要求, 随之出现了神经网络专用处理器. 为了对这类处理器进行客观比较, 并指导软硬件优化设计, 本文针对卷积神经网络提出了宏基准测试程序和微基准测试程序. 其中, 宏基准测试程序包含主流的卷积神经网络模型, 用于处理器性能的多方位评估和对比; 微基准测试程序包含卷积神经网络中的核心网络层, 用于细粒度定位性能瓶颈并指导优化. 为了准确描述这套基准测试程序在真实硬件平台上的性能表现, 本文选取了 I/O 等待延迟、跨节点通信延迟和 CPU 利用率 3 大系统性能评测指标以及 IPC、分支预测、资源竞争和访存表现等微架构性能评测指标. 基于评测结果, 本文为处理器的硬件设计与架构改进提出了可靠建议.

关键词: 卷积神经网络; 网络层; 基准测试程序; 性能分析; 微体系结构

引用格式: 徐青青,安虹,武铮,金旭.主流卷积神经网络的硬件设计与性能分析.计算机系统应用,2020,29(2):49–57. http://www.c-s-a.org.cn/1003-3254/7257.html

Hardware Design and Performance Analysis of Mainstream Convolutional Neural Networks

XU Qing-Qing, AN Hong, WU Zheng, JIN Xu

(School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)

Abstract: As one of the most influential networks in the field of deep learning, convolutional neural network is deeper and deeper, and proposes higher demand for computing capabilities. Various dedicated processors have emerged. In order to compare such processors fairly and help to optimize software and hardware, this study proposes macrobenchmarks and microbenchmarks for convolutional neural networks. The macrobenchmarks include mainstream convolutional neural networks for evaluating processors, the microbenchmarks include core layers in them for analyzing bottlenecks and guiding optimization. This study characterizes the behaviors of benchmarks from both system and microarchitecture aspects. The system metrics include I/O wait, cross-node communication and CPU utilization, the microarchitecture metrics include IPC, branch prediction, back-end resource competition and memory access. Based on the performance results, this study provides reliable advice for helping optimizing processors.

Key words: convolutional neural network; network layer; benchmark; performance analysis; microarchitecture

1　概述

近些年来, 深度学习技术蓬勃发展, 广泛应用于各大领域并接连取得了突破性成就. 卷积神经网络作为该领域中最具影响力的网络结构之一, 在计算机视觉领域中长期占据着主导地位, 相关技术趋向成熟. 目前, 卷积神经网络主要基于通用 CPU 和 GPU 进行训练,

① 基金项目: 国家重点研发计划 (2016YFB1000403); 中央高校基本科研业务费专项资金 (YD2150002001)
Foundation item: National Key Research and Development Program of China (2016YFB1000403); the Fundamental Research Funds for the Central Universities of China (YD2150002001)
收稿时间: 2019-06-22; 修改时间: 2019-07-16; 采用时间: 2019-07-26; csa 在线出版时间: 2020-01-16

而通用处理器在面对网络训练的庞大计算需求时, 表现出较低的能效比. 而且, 随着网络结构朝着更深更复杂的方向发展, 对硬件计算能力提出了越来越高的要求, 随之出现了各种类型的专用处理器. 为了对这类处理器进行评估并指导其优化设计, 需要一套基准测试程序作为指导标准.

为此, 本文面向卷积神经网络提出了一套基准测试程序. 基准测试程序的设计分为两部分, 在为宏基准测试程序选定好卷积神经网络后, 为了把握网络的整体性能表现, 本文从系统层面对网络程序进行评测. 但是, 网络结构的复杂性使得难以对其进行深入的微架构性能分析和瓶颈定位, 这就需要对网络中各个组成部分做进一步的分析. 考虑到卷积神经网络是由网络层构成的, 除了输入层, 网络训练过程中每个网络层都是作用于相邻层输出的张量结果. 将这些网络层看做独立的计算单元, 将其从网络中抽取出来并为其提供输入集, 使其成为完整的测试模块, 构建微基准测试程序. 由于不同的网络层具有不同的程序特性, 包括计算特性和访存特性等, 通过对这些程序独立进行分析, 明确各个网络层的行为特征, 定位它们的性能瓶颈, 从而有针对性地进行处理器的结构设计.

本文在给出基准测试程序后, 在通用 CPU 平台上运行基准测试程序, 从系统和微架构层面对测试程序进行性能评测. 通过分析性能数据, 明确测试程序的行为特征和性能瓶颈, 进而给出处理器的优化建议.

2 相关工作

早期的神经网络基准测试程序不具有讨论价值, 以 BenchNN[1]为例, 它包含的是多层感知器等早期神经网络. 若用这样过时的基准测试程序对处理器进行评测, 不能准确反映出先进神经网络和应用的行为特征, 不能对处理器的设计提供正确指导.

DeepBench[2]是由百度开发的一款神经网络基准测试程序, 旨在评测神经网络中最核心的网络层和基础操作的性能表现, 因此它仅选取了卷积层、循环层和矩阵乘法作为测试程序. 由于 DeepBench 包含的网络层有限, 不能全面反映出神经网络的行为特性.

Data Motifs[3]是面向大数据应用和神经网络任务的基准测试程序, 与神经网络相关的测试程序包括卷积层、池化层、激活层和矩阵乘法. 与 DeepBench 相比程序集更丰富, 但是仍缺乏全面性.

BenchIP[4]的测试程序较多, 包括 11 个神经网络和 10 个网络层, 但是测试程序的选取有待改进. BenchIP 没有选取主流的 Inception 系列网络等; 在选取人脸识别网络时, 未包含识别效果最佳的 FaceNet[5]; 在选取网络层时, 没有剔除不再流行的 LRN 等层, 而未包含流行的 Concat 和 Eltwise 等层.

针对神经网络基准测试程序的研究工作还有很多, 但是它们在设计时存在诸多不足, 主要表现在: 大多数基准测试程序仅从网络或网络层的单一角度进行设计; 在选取目标网络或网络层时缺乏全面性; 没有充分考虑所选网络或网络层的流行性; 仅针对网络的前向计算过程进行设计; 缺少从系统、微架构层面对测试程序进行全面的性能分析过程. 本文基于卷积神经网络, 克服现有基准测试程序存在的种种缺陷, 提出了一套基准测试程序.

3 设计方法

3.1 卷积神经网络选取

为了使构建出的基准测试程序能够有效指导硬件设计, 本文充分调研了卷积神经网络的主流应用领域和各领域的网络情况, 为宏基准测试程序的构造奠定基础. 基于流行性和代表性选取得到的卷积神经网络如下:

(1) 手写数字识别网络: LeNet[6]的简单网络结构对手写数字图像的识别效果较好, 因此被选作手写数字识别领域的代表网络, 采用的数据集为 MNIST.

(2) 图像分类网络: 从 2012 年开始, 图像分类领域涌现出了众多卷积神经网络, 最初大多都基于 ImageNet 数据集进行训练, 并在 ILSVRC 竞赛中表现出了优异的分类效果, 包括 2012 年的 AlexNet[7], 2013 年的 ZFNet[8], 2014 年的 Vgg[9]和 GoogLeNet[10], 2015 年的 ResNet[11], 以及表现优于 ResNet 的 DenseNet[12]. 在选取 Inception 和 ResNet 系列网络时, 本文将其多个版本均包含进基准测试程序, 这是因为不同版本的网络复杂度差异较大且均被广泛使用. 在选取以上网络作为图像分类领域的代表网络后, 统一为它们提供计算机视觉标准数据集 ImageNet.

(3) 轻量型分类网络: 卷积神经网络大多关注训练精度, 在精度要求不高的情况下, 小规模网络模型通常具有训练速度快、带宽要求低等优点, 且能够很好地部署到 FPGA 等硬件上, 因此, 轻量型分类网络得以提

出, 著名的有 SqueezeNet[13], ShuffleNet[14]和 MobileNet[15]. 本文选取这 3 个网络作为轻量型网络的代表, 并选择 ImageNet 作为其数据集.

(4) 目标检测网络: 该领域主要包括两类卷积神经网络, 分别是 R-CNN 系列网络和 YOLO 系列网络, 这两类网络在目前的目标检测领域均得到了广泛应用并占据着主导地位. 本文分别选取 R-CNN[16]和 YOLO v3[17]作为该领域中 R-CNN 系列和 YOLO 系列的代表网络, 采用目标检测数据集 PASCAL VOC.

(5) 语义分割网络: 图像分割技术在 2014 年之后取得了突破性进展, 这得益于 FCN[18]的提出, 该网络确定了语义分割的基础框架, 实现了对输入图像的逐像素分类, 随后出现了很多基于 FCN 的研究工作. 本文选取 FCN 作为语义分割的代表网络, 为其提供数据集 PASCAL VOC 2012.

(6) 医学影像分割网络: U-Net[19]常作为 Kaggle 竞赛中解决医学影像分割问题的优选方案, 该网络采用特殊的 U 型结构, 具有从少量数据中学习特征的能力. 本文选取 U-Net 作为医学影像分割领域的代表网络, 并为其提供肺部结节检测数据集 LUNA16.

(7) 人脸识别网络: 已有很多前沿项目将卷积神经网络应用到人脸识别任务中, 最著名的包括 DeepID[20], DeepFace[21]和 FaceNet, 表现最优的 FaceNet 在人脸数据集 LFW 上的识别准确率高达 99.63%, 已超出人眼识别的 99.25% 准确率. 本文选取 FaceNet 作为人脸识别领域的代表网络, 并为其提供数据集 LFW.

本文为图像分类领域选取的网络多达十几个, 但是这些网络的作用不仅仅局限于图像分类, 由于它们具有很好的特征提取能力, 目前被广泛应用于各大领域. 如神经风格迁移应用 fast-style-transfer 的核心网络为 Vgg; 主流实例分割网络 Mask R-CNN[22]的核心网络为 ResNet; Faster R-CNN[23]的核心网络为 ZFNet 或 Vgg 等. 综上所述, 本文共选取了 20 个流行的卷积神经网络并为各个网络配置了数据集, 这些网络涉及到的应用领域众多, 具有很好的代表性, 它们共同构成了宏基准测试程序.

3.2 网络计算量和参数量分析

在提出和改进卷积神经网络模型的过程中, 很多研究工作都是着眼于降低模型的计算量和参数量. 网络计算量在很大程度上决定了网络模型的训练时间和预测时间; 网络携带的参数量又与网络在分布式训练过程中产生的跨节点通信量有关. 在网络进行分布式数据并行训练时, 参数服务器对各个计算节点上的参数梯度进行收集后求平均值, 再将处理后的梯度回传给计算节点用于更新本地参数, 当网络的参数较多时, 参数服务器和计算节点之间的参数梯度传输量较大, 可能会产生较高的通信延迟, 影响网络的训练速度.

把握网络的计算量和参数量有助于估计网络的计算耗时和通信耗时情况, 在衡量网络模型的计算量时, 针对单个样本输入, 选取网络计算过程中产生的乘加操作次数 MACCs 作为评测指标. 在衡量网络模型的参数量时, 选取卷积层和全连接层携带的参数个数作为评测指标, 这是因为它们具有的参数量通常占据了网络参数总量的绝大部分, 而其他网络层不具有参数或只具有很少的参数. 图 1 给出了所选网络的计算量和参数量, 从图中可以看出, 各个网络的计算量和参数量存在较大的差异性.

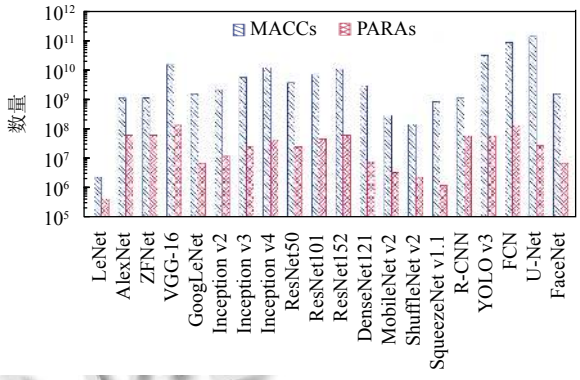

图 1 卷积神经网络的计算量和参数量

3.3 热点网络层分析

在针对目标程序进行软硬件优化时, 首先需要对程序进行热点分析. 本文通过对所选网络中网络层的出现频率和执行时间占比进行统计分析, 定位出各个网络中频繁出现、较耗时的网络层, 对这些网络层进行优化通常能够明显提升网络的计算效率. 图 2 给出了目标网络中网络层的出现频率, 图 3 基于 Intel Xeon E5-2695 给出了网络层在网络中的执行时间占比.

不同的卷积神经网络包含的网络层存在差异性, 且各个网络层在网络中的出现频率不尽相同, 本文所选网络主要涉及 11 种网络层. 由图 2 可以看出, 所有网络均包含卷积层和 ReLU 层, 两者的出现频率较高且相当, 频率总和在大多数网络中高达 50% 以上. 这

是因为卷积层和 ReLU 层是卷积神经网络中最核心的操作, 卷积层在网络中反复出现起到逐步提取特征的作用, ReLU 层一般作用于卷积层之后, 为网络引入非线性. 图中多数网络包含归一化层 (LRN 层、BatchNorm 层), LRN 层出现在早期提出的网络中, 随后被 BatchNorm 层取代, BatchNorm 层在网络中的出现频率通常高达 30%.

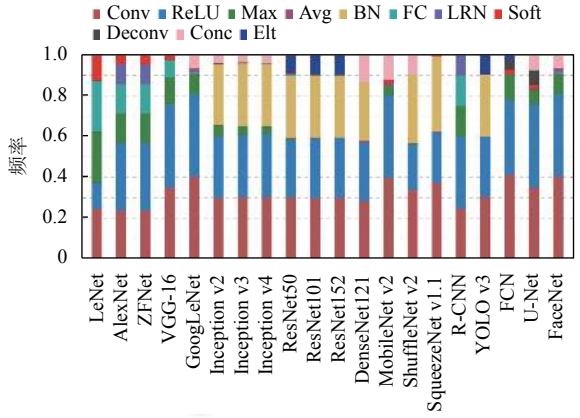

图 2 网络层的出现频率

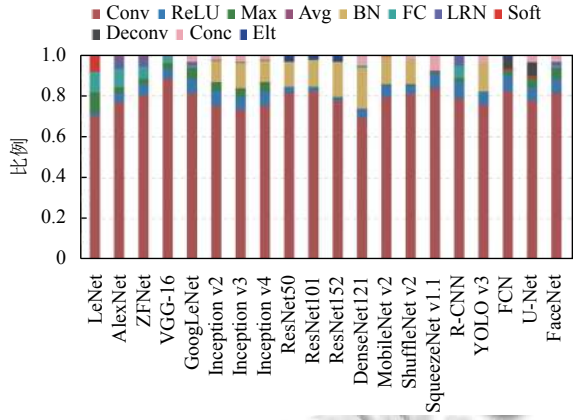

图 3 网络层的执行时间占比

此外, 最大池化层出现在大多数网络中且出现频率较高, 平均池化层一般以全局平均池化的方式出现在网络中且出现频率一律较低. 全连接层通常出现在网络的后几层且出现次数不超过 3 次, Softmax 层在多数网络中单次出现, 反卷积层对于语义分割领域的网络 (FCN、U-Net 等) 具有不可或缺的影响, 两类融合层 (Concat 层、Eltwise 层) 以较高频率出现在多数网络中.

图 3 显示卷积层的执行时间占比在所有网络中均高达 70% 以上, 这是因为卷积层在各个网络中的出现频率较高, 且单个卷积层产生的计算量较大. 相比于卷积层, ReLU 层尽管有着相当的出现频率, 但是执行时间占比均在 5% 以下, 这是因为 ReLU 层的计算是基于元素级别的, 产生的计算量较少. BatchNorm 层一旦被卷积神经网络所采用, 出现频率一般较高, 且执行时间占比能够达到 10% 至 20%. 全连接层和 Softmax 层在较浅的网络中有着较高的执行时间占比, 但是均不超过 10%. 而在较深的网络中, 它们对应的执行时间占比很低, 这是因为全连接层和 Softmax 层通常出现在网络的最后几层, 且全连接层的出现次数不超过 3 次, Softmax 层的出现次数一般为单次, 这两类网络层的执行时间相对于众多的其他层而言非常少. 最大池化层的执行时间占比在多数网络中不超过 5%, 平均池化层的执行时间占比极低, 两类融合层在网络中的执行时间比例不足 10%.

综合网络层的出现频率和执行时间占比情况, 本文最终选取卷积层、ReLU 层、最大池化层、平均池化层、BatchNorm 层、全连接层、Softmax 层、反卷积层、Concat 层和 Eltwise 层构建微基准测试程序. 值得注意的是, 本文没有选取 LRN 层是因为该层当前已不再流行, 将其纳入基准测试程序不具有实际意义. 微基准测试程序涵盖了目前卷积神经网络中流行的网络层, 且大多数网络层在卷积神经网络中有着较高的出现频率或执行时间占比, 所选的网络层对于卷积神经网络的构建发挥着不可或缺的作用.

3.4 基准测试程序集

通过详尽的调研与分析, 本文最终选取 20 个主流的卷积神经网络构成宏基准测试程序, 而微基准测试程序集如表 1 所示.

表 1 微基准测试程序集

网络层	输入数据集	特征
Conv331	Small, Medium, Large	计算密集
Conv332	Small, Medium, Large	计算密集
Conv111	Small, Medium, Large	计算密集
Conv551	Small, Medium, Large	计算密集
Conv772	Small, Medium, Large	计算密集
ReLU	Small, Medium, Large	计算密集
MaxPool222	Small, Medium, Large	计算密集
MaxPool332	Small, Medium, Large	计算密集
AvgPool777	Small, Medium, Large	计算密集
BatchNorm	Small, Medium, Large	计算访存密集
FullyConnect	Small, Medium, Large	计算密集
Softmax	Small, Medium, Large	计算访存密集
Deconv	Small, Medium, Large	计算密集
Concat	Small, Medium, Large	N/A
Eltwise	Small, Medium, Large	访存密集

根据实际应用情况, 本文为卷积层构造了 5 种常用配置, 分别是 (3×3, 1), (3×3, 2), (1×1, 1), (5×5, 1), (7×7, 2), 配置给出了卷积核尺寸和卷积步长; 为平均池化层构造了 2 种配置, 分别是 (2×2, 2), (3×3, 2), 为最大池化层给出的配置为 (7×7, 7), 配置给出了池化窗口大小和池化步长. 因此, 微基准测试程序共包含 15 个子测试模块. 此外, 在输入集方面, 本文提供了小中大 3 种规模, 分别为 (64, 256, 56×56), (64, 128, 112×112), (64, 64, 224×224), 规模参数依次给出了批量样本数、通道数和输入张量的尺寸.

图 4 给出了基准测试程序的实现及评测框架, 本文基于通用 CPU、GPU 和国产神威平台上的高效深度学习库, 给出了基准测试程序在这些平台上的实现.

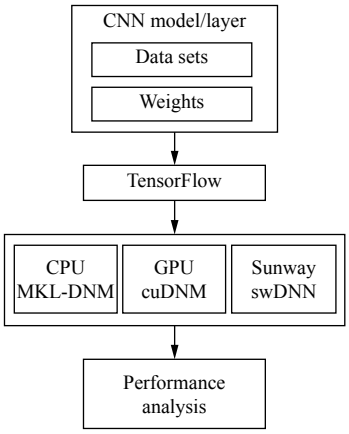

图 4 基准测试程序实现及评测框架

4 实验结果及分析

4.1 实验平台和工具

实验基于 3 台商用 Intel 服务器, 每台服务器的硬件配置具体如表 2 所示.

表 2 实验服务器的硬件配置

参数		配置
CPU		Intel(R) Xeon E5-2695 v4 36 cores @ 2.10 GHz
Cache	L1 DCache	36×32 KB
	L1 ICache	36×32 KB
	L2 Cache	36×256 KB
	L3 Cache	45 MB
Memory Size		110 GB, DDR4
Disk		SSD
Ethernet		56 GB/sec (4X FDR)
Hyper Threading		Disabled

实验采用性能分析工具 Intel VTune Amplifier XE 和 Perf, 在程序执行过程中, 捕获系统中发生的硬件事件. 通过对众多事件有选择地进行选取和计算, 得到所需的性能数据.

4.2 系统性能评测

卷积神经网络的训练过程是基于大数据的, 大量的训练数据最开始存放于磁盘, 在每次迭代前, 输入层从磁盘读取批量样本数据, 整个训练过程中针对大量数据的读取操作可能会造成较高的 I/O 等待延迟. 另外, 为了加速网络的训练过程, 基于多节点的分布式数据并行训练[24]就显得尤为必要, 然而, 这一过程中网络参数梯度需要在参数服务器和计算节点间相互传输, 可能会产生大量的节点间通信从而造成较高的通信延迟. 为了把握宏基准测试程序中各个网络的整体性能表现, 在 3 台服务器上执行网络的分布式数据并行训练, 图 5 给出了这些网络在 batchsize 为 2 时的 I/O 等待延迟、跨节点通信延迟和 CPU 利用率等 3 个系统性能指标的评测结果.

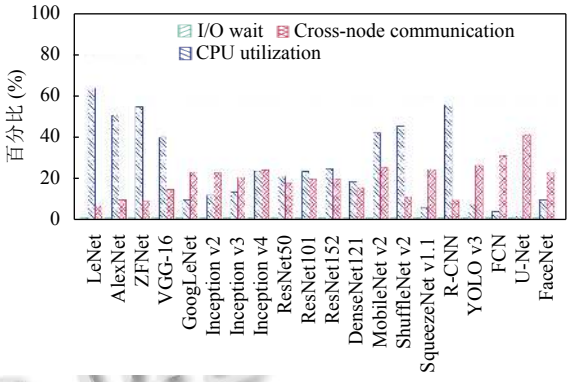

图 5 I/O 等待延迟、跨节点通信延迟和 CPU 利用率

由图 5 可知, 所有网络的 I/O 等待时间均不足 1%(图中一律显示为 1%). 虽然训练数据量巨大, 但是, 网络在每次迭代训练的过程中, 仅由输入层从磁盘读取一批次训练数据, 产生 I/O 行为. 在数据到达网络后, 其他网络层的计算都是基于上一层产生的输出结果, 这些操作数存放在内存中, 中间网络层的计算过程不涉及磁盘的读写操作, 通信过程同样不存在 I/O 行为. 输入层读磁盘产生的 I/O 等待延迟相对于网络计算时间和通信时间而言极低, 读磁盘造成的 I/O 等待不是影响网络模型训练性能的因素.

然而, 很多网络的通信延迟显示较高, 部分网络的通信时间高达 40%. 结合图 6 给出的网络计算量与参数量的比值来看, 可以发现比值较低的网络通常具有

较高的通信时间占比, 而对于计算量与参数量比值最大的网络 U-Net, 通信时间不足 2%. 这是因为当网络参数量相对于计算量越小时, 通信时间相对于计算时间也就越少, 通信时间占比越低, 因此, 可以从网络计算量与参数量的比值对网络通信延迟作出估计.

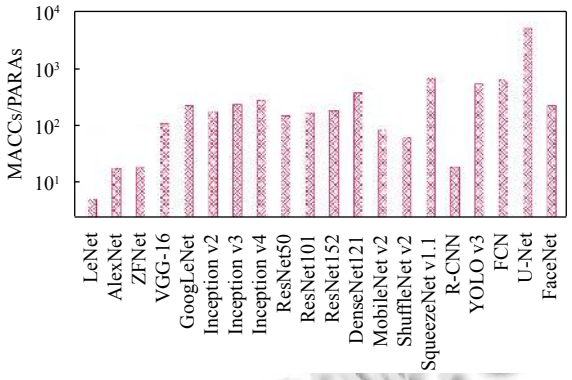

图 6 网络计算量与参数量的比值

最后, 从 CPU 利用率来看, 各个网络的这一指标值存在差异性, 通信时间占比较低的网络通常具有较高的 CPU 利用率, 对计算资源的利用更加充分.

经过进一步的探究发现, 随着 batchsize 的增大, 这些网络的通信时间占比减小且 CPU 利用率增大, 这是因为网络计算量大幅增加, 而网络参数量近似不变. 因此, 为网络训练过程选取较大的 batchsize 有利于减少通信开销且更充分地利用 CPU 计算资源, 网络能够较快地训练完所有数据. 为了支持网络在较大 batchsize 下的训练, 在构建网络训练平台时, 需要配置足够的内存量.

4.3 微架构性能评测

接下来针对微基准测试程序中的网络层, 从微架构层面进行自顶向下的性能评测, 明确不同网络层的行为特征并定位它们的性能瓶颈, 从而有针对性地对处理器的微架构改进提出建议. 后续实验都是基于各个网络层程序在大规模输入集下的执行过程所得.

首先关注测试程序的整体性能表现, 图 7 给出了它们的 IPC 和 Retiring. IPC 表示周期指令数, Retiring 表示执行有效微操作的流水线槽数占流水线总槽数的比例. 实验平台采用 Intel 的四发射处理器, 假设每条程序指令被解码为单个微操作, 当 Retiring 为 100% 时, IPC 可达到理论最大值 4. 图中显示 IPC 与 Retiring 具有高度的一致性, 多数程序的 IPC 高达 2 以上, 程序执行性能表现较好. BatchNorm、Softmax、Concat 和 Eltwise 层的 IPC 明显较低, 程序的指令并行性仍有待发掘. 然而, Retiring 较高并不代表目标程序没有性能优化空间, 通过进一步分析发现, 池化层和 ReLU 层的向量化程度显示极低. 可以通过对 Intel MKL-DNN 中池化层和 ReLU 层实行向量优化, 使得单条指令就能完成多个浮点计算, 由此来提升程序执行性能.

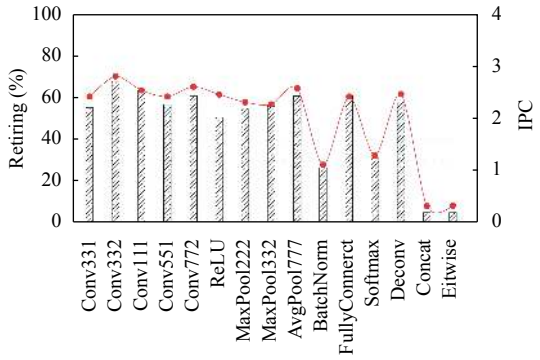

图 7 IPC 和 Retiring

对于非空的流水线槽, 其上执行的微操作如果最终成功退役, 则该流水线槽被归类为 Retiring, 不产生性能瓶颈. 但是, 当分支预测发生错误时, 流水线槽上执行的微操作在退役前被取消, 导致流水线槽的无效执行. 图 8 给出了各个程序的分支指令比重和分支预测错误率, 可以看出 ReLU 层、最大池化层和 Softmax 层具有较高比例的分支指令, 且分支预测错误率较高, 程序性能在很大程度上受到错误分支预测的影响. 这是因为这几个程序中存在大量的比较操作, 且操作数之间的大小存在不可预测性. 其他程序中的分支预测行为主要产生于循环控制部分, 这类的分支跳转能够很好地被当前基于历史的分支预测机制所处理, 因此表现出较低的分支预测错误率.

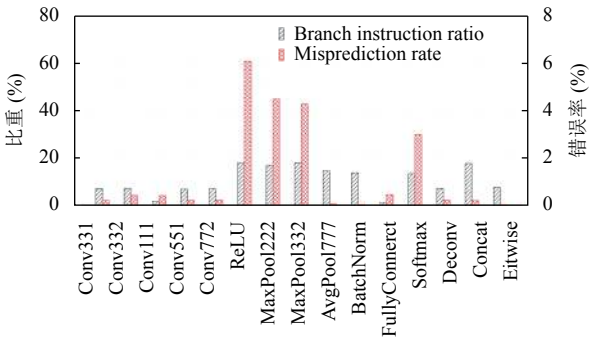

图 8 分支指令比重和分支预测错误率

然而, 并不是所有的流水线槽都会被占用, 空的流水线槽表现为 CPU 停顿, 从 CPU 时钟周期上微操作的执行情况考虑, 可以将总的时钟周期划分为工作周期和停顿周期. 在 CPU 停顿周期上, 执行单元空闲, 没有微操作在执行, 频繁的 CPU 停顿必然会造成程序性能的极大损失.

引起 CPU 停顿的原因有很多, 从前端来看, 取指译码阶段造成的 CPU 停顿表现为指令饥饿. 通过对目标程序的指令饥饿表现进行探究, 发现具有较高分支预测错误率的 ReLU 层、最大池化层和 Softmax 层对应较高的指令饥饿, 测试程序的指令饥饿在较大程度上由错误的分支预测造成. 这是因为当分支预测发生错误后, 流水线需要被重新刷新, 在程序恢复正确执行路径之前, 执行单元没有来自于前端的可执行指令, 处于等待指令的空闲状态. 改善分支预测机制对于这 3 个程序的性能提升会带来明显效果, 不仅使执行无效微操作的流水线槽减少, 还能降低指令饥饿.

前端造成的停顿一般较少, 很大一部分的 CPU 停顿由后端执行阶段造成, 由于后端资源有限, 当产生资源竞争时, 微操作便不能被发射. 乱序执行过程中需要竞争的资源主要包括保留站、读缓冲、写缓冲和重排序缓冲. 通过详尽探究资源的使用情况, 最终定位出测试程序的资源竞争集中在保留站和写缓冲, 图 9 给出了这些程序的保留站满载率和写缓冲满载率.

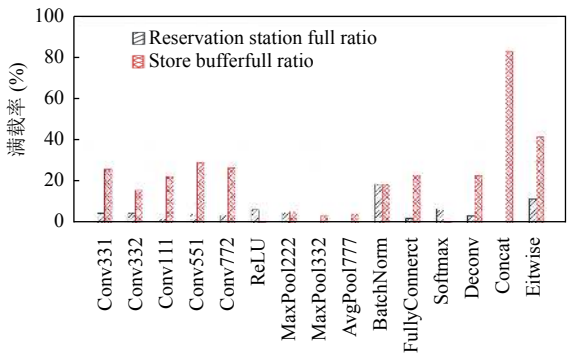

图 9 保留站满载率和写缓冲满载率

由图 9 可知, BatchNorm 层的保留站满载率最高, 其 20% 的满载率在很大程度上由程序内部频繁的除法操作造成, 由于除法操作通常更加耗时而除法单元配置较少, 除法单元长时间被占用导致后续连续的除法微操作不能被分配到执行单元, 微操作滞留于保留站中造成目标资源的频繁满载. 由此可见, 优化除法操作、增加除法执行单元对 BatchNorm 层的性能提升有较大的帮助. 与保留站竞争相比, 写缓冲竞争对测试程序造成的性能损失更普遍且更明显. 大部分程序的写缓冲满载率高达 20% 以上, 其中, Concat 层和 Eltwise 层的写缓冲满载率分别高达 80% 和 40%, 这与程序内部大量连续的存储操作密切相关, 进行写缓冲资源的扩容对程序性能提升有着重要意义.

在后端执行过程中, 复杂的依赖关系、计算资源受限和访存受限均会造成程序执行性能的损失, 接下来从访存表现进行探究. 在高速缓存的 3 个级别中, L1 DCache 离 CPU 最近, 速度最快, 较高的 L1 DCache 命中率能够很好地解决访存与计算速度的不匹配问题. 然而, 一旦程序执行过程中频繁发生 L1 DCache 的访问缺失, 程序执行性能会受到很大影响, 图 10 给出了各个程序的 L1 DCache 缺失率.

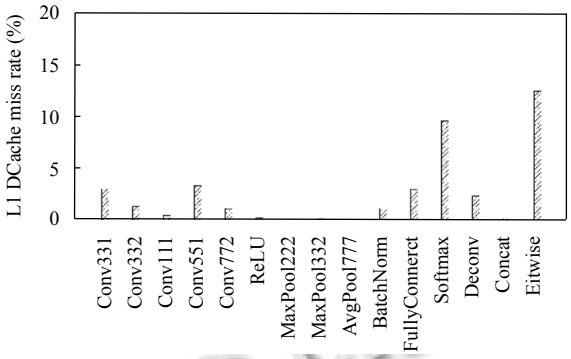

图 10 L1 DCache 缺失率

可以发现, 大部分测试程序的 L1 DCache 缺失率较小, 这是因为它们基于数据块做循环计算, 程序具有良好的数据局部性, 当一个 Cache Line 的数据从内存被取进 L1 DCache 后, 在接下来的一系列操作中, 前面读进来的数据都能被命中. 其中, ReLU 层、池化层、Concat 层的 L1 DCache 缺失率极低, 不足 0.3%, 卷积层、全连接层和反卷积层的 L1 DCache 缺失率均在 3% 以下. Softmax 层和 Eltwise 层的 L1 DCache 缺失率较高, 后者的 L1 DCache 缺失率最高, 达到 12% 以上, 这是因为 Eltwise 层的主要计算是矩阵的按元素相加操作, 内部计算较为简单, 几乎不存在数据依赖, 在程序执行过程中, 产生大量的同时取数据操作, 由此造成大量的数据缓存缺失.

当 L1 DCache 命中失败时, 需要访问 L2 Cache, 图 11 给出了测试程序的 L2 Cache 局部缺失率和全局

缺失率.

L2 Cache 的局部缺失率即为 L2 Cache 的缺失次数与其访问总次数的比值, L2 Cache 的全局缺失率是其局部缺失率与 L1 DCache 缺失率的乘积结果. 图中显示部分程序的 L2 Cache 局部缺失率高达 80%, L2 Cache 的局部缺失率不具有说服力, 这是因为 L1 DCache 中存储的数据是最容易被命中的, L2 Cache 只会在 L1 DCache 发生缺失时才会被访问. 因此, 在评测 L2 Cache 缺失率时, 需要选取全局缺失率, 大多数程序的 L2 Cache 全局缺失率不足 1%, Softmax 层和 Eltwise 层的 L2 Cache 全局缺失率在 10% 左右, 这是由其极差的数据局部性造成.

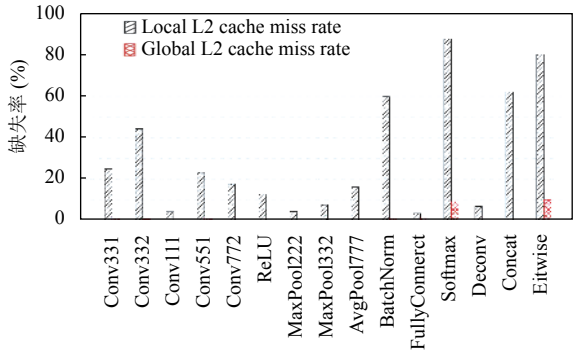

图 11 L2 Cache 的局部缺失率和全局缺失率

综合 L1 DCache 缺失率和 L2 Cache 的全局缺失率来看, 绝大多数测试程序的取数据需求能够被前两级缓存很好地满足, 程序对 L3 Cache 的访问需求极小, 本文不再对 L3 Cache 的缺失率进行分析. 在此基础上给出了进一步的分析, 以 Conv331 为例, 在 GEM5 体系结构模拟器上探究了 L1 DCache 和 L3 Cache 的 6 种配置组合对目标程序的影响. 其中, L3 Cache 的容量被减小为 25 MB, 相联度保持 20 路不变, L1 DCache 的配置分别为 2 路 32 KB, 4 路 32 KB, 8 路 32 KB, 2 路 64 KB, 4 路 64 KB, 8 路 64 KB. 通过对比分析这些配置下的 L1 DCache 缺失率和目标程序的执行时间发现, 相联度产生的影响极小, 因此重点关注容量配置带来的影响. 相对于 L1 DCache 容量为 32 KB 的情况, 在容量增至 64 KB 时 L1 DCache 的缺失率降低到 80% 以下, 目标程序的执行时钟周期数也有所减少, 由此可见, 增大 L1 DCache 容量在较大程度上降低了缺失率, 对于目标程序的执行时间优化具有较好的效果. 另外, 对比第三种配置与真实硬件配置情况发现, 目标程序的执行时间没有明显变化, 减小 L3 Cache 的容量对于目标程序的执行时间没有明显影响. 考虑到 L3 Cache 具有较大的容量, 占用了较大的芯片面积, 却没有带来程序性能的明显提升, 可以考虑减少 L3 Cache 的容量, 增大 L1 DCache 的容量.

5 结论与展望

本文给出了一套卷积神经网络基准测试程序, 包括由网络构成的宏基准测试程序和由网络层构成的微基准测试程序, 同时为所选网络提供了典型数据集, 为网络层提供了常见的配置, 并为它们构造了不同规模的输入集. 最后从系统层面和微架构层面给出了这套基准测试程序的性能评测实例, 结合程序的性能表现和程序本身进行分析, 可以证明测试程序能够准确反映卷积神经网络的程序特性, 能够用于处理器的评测和优化设计指导. 并且, 通过分析性能评测结果, 明确了目标程序的行为特征和性能瓶颈, 为处理器的设计提出了一些改进建议.

下一步将继续完善基准测试程序, 使其包含更多领域的卷积神经网络, 提高基准测试程序的代表性. 待国产神威硬件平台上的软件环境包括深度学习框架、卷积神经网络库和性能分析工具完善后, 利用这套基准测试程序为国产处理器面向神经网络训练任务的优化提供指导.

参考文献

1 Chen TS, Chen YJ, Duranton M, *et al*. BenchNN: On the broad potential application scope of hardware neural network accelerators. Proceedings of 2012 IEEE International Symposium on Workload Characterization (IISWC). La Jolla, CA, USA. 2012. 36–45.

2 Narang S. DeepBench. https://svail.github.io/DeepBench. [2016-09-26].

3 Gao WL, Zhan JF, Wang L, *et al*. Data dwarfs: A lens towards fully understanding big data and AI workloads. arXiv: 1802.00699, 2018.

4 Tao JH, Du ZD, Guo Q, *et al*. BENCHIP: Benchmarking intelligence processors. Journal of Computer Science and Technology, 2018, 33(1): 1–23. [doi: 10.1007/s11390-018-1805-8]

5 Schroff F, Kalenichenko D, Philbin J. Facenet: A unified embedding for face recognition and clustering. Proceedings of 2015 IEEE Conference on Computer Vision and Pattern

Recognition. Boston, MA, USA. 2015. 815–823.
6 LeCun Y, Bottou L, Bengio Y, *et al*. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 1998, 86(11): 2278–2324. [doi: 10.1109/5.726791]
7 Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. Proceedings of the 25th International Conference on Neural Information Processing Systems. Lake Tahoe, NV, USA. 2012. 1097–1105.
8 Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. Proceedings of the 13th European Conference on Computer Vision. Zurich, Switzerland. 2014. 818–833.
9 Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv: 1409.1556, 2014.
10 Szegedy C, Liu W, Jia YQ, *et al*. Going deeper with convolutions. Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA, USA. 2015. 1–9.
11 He KM, Zhang XY, Ren SQ, *et al*. Deep residual learning for image recognition. Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, NV, USA. 2016. 770–778.
12 Huang G, Liu Z, Van Der Maaten L, *et al*. Densely connected convolutional networks. Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, HI, USA. 2017. 4700–4708.
13 Iandola FN, Han S, Moskewicz MW, *et al*. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. arXiv: 1602.07360, 2016.
14 Zhang XY, Zhou XY, Lin MX, *et al*. ShuffleNet: An extremely efficient convolutional neural network for mobile devices. Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA. 2018. 6848–6856.
15 Howard AG, Zhu ML, Chen B, *et al*. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv: 1704.04861, 2017.
16 Girshick R, Donahue J, Darrell T, *et al*. Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA. 2014. 580–587.
17 Redmon J, Farhadi A. YOLOv3: An incremental improvement. arXiv: 1804.02767, 2018.
18 Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA, USA. 2015. 3431–3440.
19 Ronneberger O, Fischer P, Brox T. U-Net: Convolutional networks for biomedical image segmentation. Proceedings of the 18th International Conference on Medical Image Computing and Computer-Assisted Intervention. Munich, Germany. 2015. 234–241.
20 Sun Y, Wang XG, Tang XO. Deep learning face representation from predicting 10, 000 classes. Proceedings of 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA. 2014. 1891–1898.
21 Taigman Y, Yang M, Ranzato MA, *et al*. Deepface: Closing the gap to human-level performance in face verification. Proceedings of 2014 IEEE Conference on Computer Vision and Pattern Recognition. Columbus, OH, USA. 2014. 1701–1708.
22 He KM, Gkioxari G, Dollár P, *et al*. Mask R-CNN. Proceedings of 2017 IEEE International Conference on Computer Vision. Venice, Italy. 2017. 2961–2969.
23 Ren SQ, He KM, Girshick R, *et al*. Faster R-CNN: Towards real-time object detection with region proposal networks. Proceedings of the 28th International Conference on Neural Information Processing Systems. Montreal, Canada. 2015. 91–99.
24 李垠桥, 阿敏巴雅尔, 肖桐, 等. 基于数据并行的神经语言模型多卡训练分析. 中文信息学报, 2018, 32(7): 37–43. [doi: 10.3969/j.issn.1003-0077.2018.07.005]