# 异构多芯粒神经网络加速器

朱郭益, 马胜*, 张春元, 王波

(国防科技大学计算机学院 长沙 410073)
(masheng@nudt.edu.cn)

**摘 要:** 随着神经网络技术的快速发展, 出于安全性等方面考虑, 大量边缘计算设备被应用于智能计算领域. 首先, 设计了可应用于边缘计算的异构多芯粒神经网络加速器其基本结构及部件组成. 其次, 通过预计算异构芯粒上的计算负载, 将计算任务在神经网络通道上进行划分, 不断加入新的任务, 逐芯粒测试并进行迭代, 选取异构芯粒组合以构建神经网络加速器. 最后, 分别在抽样构造的测试神经网络、MobileNet 以及 ShuffleNet 上使用这种粗粒度优化的方法构建了异构多芯粒神经网络加速器, 并测试了其能耗与性能表现. 实验结果表明, 这种异构的设计方法可以在控制能耗同时, 分别取得 7.43, 2.30 和 5.60 的加速比.

**关键词:** 芯粒; 异构; 神经网络; 加速器
**中图法分类号:** TP391.72 **DOI:** 10.3724/SP.J.1089.2023.19451

# Heterogeneous Multi-Chiplets Neural Network Accelerator

Zhu Guoyi, Ma Sheng*, Zhang Chunyuan, and Wang Bo

(*College of Computer, National University of Defense Technology, Changsha* 410073)

**Abstract:** With the rapid development of neural network technology, a large number of edge computing devices are used in the field of intelligent computing for security reasons. Firstly, this paper designs the basic structure and components of a heterogeneous multi-chiplets neural network accelerator that can be applied to edge computing. Secondly, we precompute the computational load on the heterogeneous cores, divide the computational tasks on the neural network channels, continuously add new tasks, test and iterate chipet by chiplet, and select the combination of heterogeneous chiplets to build the neural network accelerator. Finally, the heterogeneous multi-chiplets neural network accelerator is constructed on the test neural network, MobileNet, and ShuffleNet with this coarse-grained optimization method, and its energy consumption and performance are tested respectively. The experimental results show that this heterogeneous design approach can achieve acceleration ratios of 7.43, 2.30, and 5.60, respectively, while controlling the energy consumption.

**Key words:** chiplet; heterogeneous; neural network; accelerator

鉴于人工智能在多个领域的突出表现, 许多神经网络模型已经被应用于人们生活中的各类任务, 如图像识别分类[1-2]、自然语言处理、自动驾驶[3]、语音助手和智能机器人[4]等. 为了实现更高的预测与识别精度, 神经网络模型正朝着网络层数更深、计算参数更多以及网络结构更复杂的方向

收稿日期: 2021-11-01; 修回日期: 2022-05-16. **基金项目**: 国家自然科学基金(62172430); 湖南省自然科学基金(2021JJ10052); 国家重点研发计划(2021YFB0300300). **朱郭益**(1994—), 男, 硕士研究生, CCF 学生会员, 主要研究方向为人工智能加速器设计; **马胜**(1986—), 男, 博士, 教授, 硕士生导师, CCF 高级会员, 论文通信作者, 主要研究方向为微处理器设计; **张春元**(1964—), 男, 博士, 教授, 博士生导师, CCF 高级会员, 主要研究方向为计算机体系结构; **王波**(1998—), 男, 硕士研究生, 主要研究方向为人工智能加速器设计.

发展. 受数据安全性、传输延时等因素的影响, 联邦学习和边缘计算[5]等概念被提出, 这对边缘计算设备的能耗与性能提出了更高的要求.

现有的神经网络模型需要运行在多种不同类型的计算设备上. 考虑计算设备在体积以及能耗等方面的差异, 其设计过程中首位考虑的因素也不尽相同. 例如, 针对数据中心服务器设计的神经网络加速器, 如 TPU[6], NVIDIA Tesla V100 等, 这些设备需要进行模型的训练与推理工作, 故在设计中主要需要考虑其性能表现, 能耗则在设计中居于次位; 边缘设备中的神经网络加速器, 如 Edge TPU, NVIDIA Jetson Nano[7], 其主要工作是在低功耗设备上执行神经网络的推理工作, 其可控能耗下的性能表现是设计中最重视的因素.

## 1 相关工作

为了实现神经网络的高效计算, Yin 等[8]提出了 Thinker 系列加速器, 其基于粗粒度可重构体系结构 (coarse-grained reconfigurable architecture, CGRA)设计, 其中的计算单元针对指定的计算负载如卷积层和全连接层/循环神经网络(recurrent neural network, RNN)层进行了专用定制优化, 并提供了可以面向负载重构的计算单元, 以在计算中实现负载均衡. 为了控制神经网络加速器的设计成本, Shao 等[9]提出的基于芯粒的深度学习推理可拓展体系结构(scaling deep-learning inference with chiplet-based architecture, SIMBA)是一种使用同构多芯粒技术进行层次化互联的体系结构, 其设计选择了单一类型的计算芯粒, 针对大型数据中心与边缘设备分别构建不同面积的衬底与不同芯粒数量的计算芯片, 有效地在保证芯片计算效率的同时实现了设计的复用, 降低了设计的成本开销.

然而, 现有面向负载优化的神经网络加速器设计中, 由于需要根据负载的不同考虑设计部件的定制, 各部件之间耦合程度较高, 设计过程复杂, 设计与制造的复用性较差, 因此导致现有神经网络加速器的设计与制造成本始终居高不下.

此外, 现有基于芯粒技术的神经网络加速器仅仅使用单一的芯粒单元, 未考虑在边缘计算设备中针对单一神经网络计算负载任务使用多种芯粒优化的可能性.

基于此, 本文提出了异构多芯粒神经网络加速器, 这是一种可以应用于边缘计算设备、实现一次设计多次复用的异构神经网络加速器. 用户仅仅需要设计一次芯粒, 依据边缘计算设备上运行的神经网络负载和性能能耗需求选用不同数量、不同种类的芯粒, 即可比同构芯粒的神经网络加速器取得较高加速比.

本文首先提出了基于异构芯粒技术构建的神经网络加速器架构, 有效地提高了设计的复用性, 降低了边缘计算芯片的设计与制造成本; 将芯粒技术在边缘计算设备结合其工作负载的特点进行设计优化, 使得神经网络加速器的体系结构设计能够紧紧与其工作负载贴合, 在能耗可控的情况下, 极大地提升了其性能.

## 2 异构多芯粒神经网络加速器的相关概念

### 2.1 边缘计算

随着物联网设备的快速发展, 数十亿的微处理器和微控制器已经部署至智能汽车、智能农业、个人健康以及智慧家庭等领域. 传统物联网设备主要用于收集环境信息, 将这些信息上传云端并进行相关计算后, 向微控制器返回决策. 而许多新型的智能物联网设备出于安全性或决策延迟时间的考虑, 需要进行本地计算. 通过减少传统物联网设备向云服务器传送与接收数据的延迟, 边缘设备的计算表现出精度要求低且延迟低的特点, 且由于这些设备的计算能耗较低、数量较多, 其设计成本与受限能耗下的性能是这类设备设计与制造中主要需要考虑的因素. 常见的边缘计算设备有 Edge TPU, NVIDIA Jetson Nano 等.

### 2.2 芯粒技术

受新工艺的研发成本与良率问题影响, 芯片的制造成本正在飞速上升. 为了有效地降低芯片的设计与制造成本, 研究者提出芯粒技术以应对芯片制造中的面积与成本矛盾.

芯粒技术可以将芯粒这样的小芯片作为设计与制造的基本单元. 芯粒技术的优势在于芯粒之间的耦合程度较低, 设计时仅需关注整体结构与关键部分微结构, 呈现较好的模块化性质. 单个小芯粒的面积较小, 可提高制造时晶圆的有效使用面积与制造良率, 极大地降低芯片的制造成本. 通过在不同大小的硅衬底上封装不同数量、不同种类的芯粒, 达到一次设计制造后就可以面向不同计算任务, 制造不同功能与性能的芯片. 在相似功能的芯片设计中, 芯粒技术体现了较好的设计复用性. 例如, 制造不同计算性能需求的 CPU, 仅需设

计一次计算芯粒; 因此在保持整体架构不变的条件下, 考虑通过在单个芯片上集成数量不同的计算芯粒, 实现多种性能的芯片制造.

主流的芯片公司也分别提出了各自基于芯粒技术的解决方案, 例如, Intel 公司的嵌入式多芯片互连桥接(embedded multi-die interconnect bridge, EMIB)技术[10]以及 AMD 公司的 Infinity Fabric 技术等. 依据 AMD 公司的统计数据[11], 在使用先进工艺的芯片上应用芯粒技术, 在得到性能提升的同时, 可有效地降低约 50%的成本.

### 2.3　轻量神经网络模型

神经网络模型的部署与其相关的计算平台密不可分. 虽然现在神经网络模型的设计向着层数更深、参数更多的方向发展, 但是移动设备受限于其体积、能耗与性能约束, 或部分移动设备(如车载智能传感器等)对于延迟的要求较高, 需要高效、实时的本地计算. 因此, 应当设计一个参数量与计算量均较小的网络模型, 使其可以快速地在边缘设备上执行, 并在其规模、计算效率以及准确率上取得较好的平衡.

MobileNet 模型[12]的提出就是为了解决此矛盾, 其首次提出了深度可分离卷积的概念. 深度可分离卷积将传统神经网络中的卷积运算分解成了 2 个阶段, 即逐通道卷积阶段与逐点卷积阶段, 通过减少每一阶段的参数量与计算量获得了较好的性能表现, 并维持了较高的模型准确率. 逐通道卷积运算仅需一组卷积核, 生成一个多维的输出张量. 逐点卷积运算与传统卷积运算基本上完全相同, 但逐点卷积运算规定了卷积核参数在每个通道上的大小仅为 1×1. 类似的神经网络结构还有在 MobileNet 上改进的 ShuffleNet[13]等, 其基本运算操作也是以卷积操作、逐通道卷积操作以及全连接操作为主.

### 2.4　硬件数据流

神经网络在计算上可以表示为一个嵌套的多重循环, 其计算可以通过数据在时间与空间上的展开来实现.

现有的神经网络加速器的主要层次结构为“DRAM-全局缓存-计算单元”, 如图 1 所示.

不同的神经网络加速器主要区别在于其计算与存储上的映射模式, 被称为神经网络加速器计算的数据流[14].

常用的硬件数据流有以下几种:

(1) 权重固定数据流. 在神经网络的计算中, 同一组权重会与多组输入的特征图进行计算, 权重在神经网络的计算过程中存在复用的机会. 基于此计算特点, 权重固定数据流的神经网络加速器在计算中先把权重放入计算单元的片上存储中进行存储, 再通过不断地更换输入特征图和输出的部分和完成神经网络的计算. 其典型代表为英伟达公司的 NVDLA.

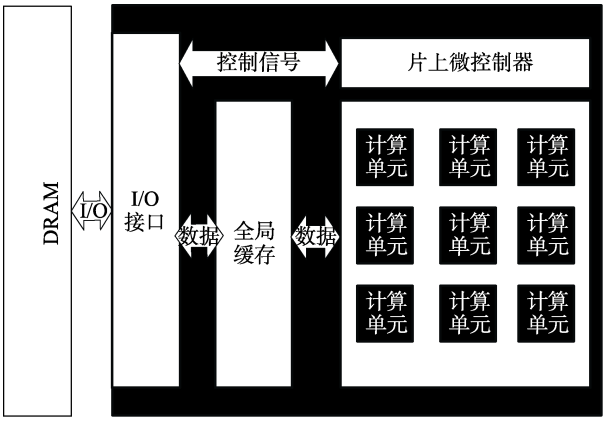



图 1　一般神经网络加速器的基本结构

(2) 输出固定数据流. 输出固定数据流的神经网络加速器在片上寄存器中存放每个周期计算完成的部分和. 通过在计算过程中不断地更换计算时的输入数据与权重数据, 将结果累加到之前的部分和中, 最后完成输出数据的计算与数据的换入/换出操作. 其典型代表为 Google 公司的 TPU.

(3) 行固定数据流. 行固定数据流是最早由 MIT 的 Chen 等[15]设计并提出, 它充分利用了卷积计算中的复用性. 由于卷积运算中可以将高维的卷积操作拆分为一维的行卷积操作, 通过在依网格排布的计算单元中横向广播权重、斜向广播输入特征图, 在计算单元中实现输入特征图中一行与权重中一行的乘累加操作, 再在纵向进行一维卷积部分和的累加操作, 得到单层卷积计算的输出结果. 其典型代表为 MIT 的 Eyeriss.

## 3　异构多芯粒神经网络加速器的基本结构

针对不同的计算负载, 不同神经网络加速器的计算性能与能耗各具特色.

首先, 本文以经典的 NVDLA 结构和 Eyeriss 结构为例, 具体分析不同神经网络加速器的数据流特点.

NVDLA 是英伟达公司推出的一套免费的、开源的体系结构设计, 其旨在提供一个神经网络推理加速器的设计标准. NVDLA 包括了体系结构的

RTL 级模型代码、编译器、Linux 下驱动、测试样例以及用户级开发工具等, 为神经网络推理加速器的构建提供了一个完整的参考解决方案. NVDLA 采用权重固定数据流, 在进行神经网络计算时, 将一组多通道的输入特征图传入广播至多个计算单元后, 再将多组权重在这些计算单元间展开. 显然 NVDLA 主要是从输入和输出通道的角度考虑数据的复用与计算的并行, 以提高神经网络的计算效率.

Eyeriss 的硬件基本组成单元与 NVDLA 相似, 但是其使用了独特的数据流“行固定数据流”来进行数据的传递与计算. 由于二维的卷积运算可以转化为输入特征图和权重张量在单通道的多行运算, 因此, 把神经网络计算任务中的行作为计算的基本单元, 将全部计算任务按通道和行进行切分, 通过数据广播实现行与行之间的数据并行复用, 在减小带宽需求的同时, 有效地提高了运算单元的利用率.

不同数据流神经网络加速器的能耗与性能对比如图 2 所示. 本文在上述 2 种结构上分别运行普通卷积运算、逐通道卷积运算以及全连接运算, 可以看到其针对不同的任务负载, 具有不同的性能与能耗表现.

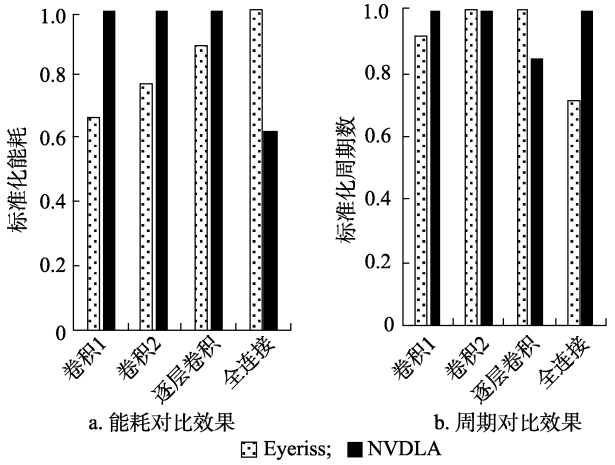



图 2 不同数据流神经网络加速器的能耗与性能对比

图 2 表明, 由于 NVDLA 数据流限制, 计算任务在卷积通道上平铺展开, 其加速效果在输入通道数较少时不具备优势. 例如, 在逐通道卷积的计算过程中, 由于其输入的通道数相当于 1, 因此, 在 NVDLA、脉动阵列等面向大规模常规卷积计算任务设计的计算设备上, 难以充分地利用其结构取得较高的能效与加速比[16-17]; 在全连接层的计算过程中, NVDLA 虽然取得了较好的性能表现, 但是其能耗也比 Eyeriss 加速器更高.

同样值得注意的是, 即使在不同神经网络加速器上运行计算层的类型相同、计算周期数接近, 但是工作负载的参数规模不一样, 其能耗差距也较大. 例如, 图 2 中的 2 个卷积层受到其输入/输出通道数影响, 2 种数据流的神经网络加速器在运行能耗与性能方面体现出了完全不同的效果.

为了有效地降低设计与制造成本, 芯粒技术被提出并应用于神经网络加速器的构建. 通过在同一块衬底上集成封装不同数量的芯粒, 可以制造面向不同计算场景和不同计算任务规模的神经网络加速器. 2021 年, 英伟达公司的 Shao 等[9]提出的 SIMBA 是第 1 个具有此结构的多芯粒神经网络加速器. SIMBA 通过在同一块芯片上封装不同数量的 NVDLA 芯粒, 实现同时为数据中心和边缘设备的计算任务提供定制神经网络加速芯片, 有效地降低了专用神经网络加速器的设计与制造成本.

在同构芯粒的计算阵列中, 所有的芯粒均采用相同的数据流结构, 针对不同的工作负载通过调整芯粒数量和硅衬底的面积, 构建同时可面向数据中心与边缘计算设备的神经网络加速器. 但是采用同构芯粒的神经网络加速器未考虑边缘计算任务的特殊性, 边缘计算设备的计算任务相对固定, 且其中多了一类特殊的逐通道卷积操作, 这对于同构的芯粒阵列不是非常友好.

基于此, 本文设计了一种异构多芯粒神经网络加速器, 如图 3 所示, 其拓扑结构类似于传统的单个神经网络加速器. 异构多芯粒神经网络加速器的组成部分主要为 I/O 芯粒模块、控制单元以及计算芯粒阵列.

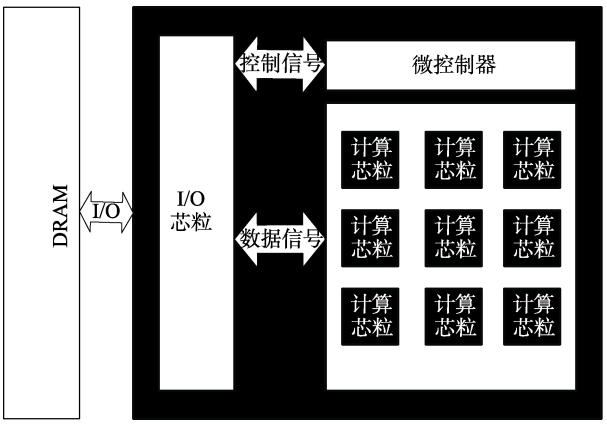



图 3 异构芯粒神经网络加速器的基本硬件结构

在芯粒阵列的接口设计规范方面, 本文参考了英特尔公司开源的高级接口总线(advanced in-

terface bus, AIB)设计[18]. AIB 是一种开源的物理层规范, 其使用全新的高密度封装技术, 提供了一个超宽的并行接口, 并提供了一定的 I/O 接口冗余. 一方面, 通过在芯粒阵列中为芯粒提供统一的总线接口引脚, 可以方便异构芯粒的集成与互联设计; 另一方面, AIB 规范具有带宽高、占用空间小的优势, 能够为提高芯片集成度、芯片间数据传输与控制传递速度, 并降低芯片间通信延时提供有效保障.

本文设计的 I/O 芯粒主要负责控制单元的信号传输以及计算芯粒阵列与 DRAM 间的数据交换. 主要功能是传输数据信号至邻近的计算芯粒, 传输外部的控制信号至控制模块, 接收控制单元的控制信号, 并向外部设备传出计算完成的数据与设备中断信号, 其控制信号与控制逻辑较为简单.

本文设计中的控制单元主要功能为通过 I/O 芯粒接收外部的写入请求、向外部设备发出忙碌信号和中断信号、向内部芯粒发送数据以及查询内部芯粒的忙碌状态.

异构计算芯粒阵列中的计算芯粒使用片上 mesh 网络进行互联, 该阵列中的每一个芯粒单元均类似于传统的神经网络加速器, 每个芯粒拥有自己的片上缓存与片上计算单元, 可以异步执行分配的计算任务, 计算任务的数据包通过片上网络进行转发, 控制信号则由控制单元通过一对多的方式轮询与发送.

为了使计算芯粒阵列的内部互联标准化, 单个计算芯粒的外部连接接口均需要划分为接收块、发送块、时钟块与异步块, 并采用相同大小的接口设计. 考虑邻近的计算芯粒工作状态未知, 单个计算芯粒需要使用异步块查询相邻的计算芯粒是否忙碌. 当邻近的计算芯粒接口处于非忙碌状态时, 该邻近芯粒可以接收数据信号, 此时接收数据的计算芯粒需要将异步接口状态信号置为忙碌, 同时在其接收块接口处读入其他芯粒送出的数据信号与时钟信号, 以解码并行传输的数据.

## 4　异构多芯粒神经网络加速器的设计过程

本文提出的异构多芯粒神经网络加速器主要是针对边缘计算设备等负载任务较为确定的设备进行优化. 以边缘计算设备为例, 边缘计算设备既是数据的采集端, 也是数据的处理端, 在同一时刻均存在不止一组数据进行计算, 并且这些数据之间相互独立, 因此, 这些计算设备存在跨层流水线并行与乱序执行的可能性.

为了达到较高的能效, 异构多芯粒神经网络加速器必须做到负载均衡. 对于卷积层和全连接层的计算, 通过其计算任务在输出通道上的划分来实现任务的独立并行. 而针对逐通道卷积操作, 其任务负载相当于在同一时刻传入了多组单通道输入, 可以在输入/输出通道上进行任务的分组, 实现任务的独立并行. 为了探索任务分割在各类计算层中的影响, 分别将不同类型的计算层输入/输出的工作负载在其通道上按比例缩减, 并进行了测试. 负载缩减测试效果如图 4 所示, 可以看出, 在输入/输出通道上进行分割的计算任务以及其性能与能耗基本保持了线性关系, 因此, 可以用线性模型估算分割后的各类模型的性能与能耗.

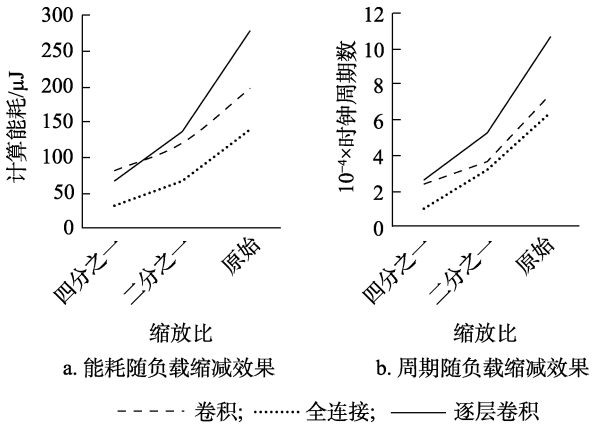



图 4　负载缩减测试效果

考虑任务分割粒度过细可能会带来的额外软硬件开销, 本加速器的设计采用粗粒度的任务负载分割方法, 通过选取合适的标准大致划分原先的输入任务负载, 通过每次在设计过程中更换新的计算芯粒, 利用合适的算法迭代以获得更高的能效.

在迭代寻求异构芯粒组合前, 需要设定超参数 $\beta$, 用于控制能耗与性能的权衡关系. 同时, 需要输入芯粒集合 $S$ 、工作负载 $W$ 和芯粒数量 $N$. 为了防止计算任务的偏态分布对任务划分的影响, 在计算过程中, 选取负载周期的均值和中位数中较小的作为标准值 $D$, 对于高于标准值 $D$ 的任务沿通道进行分割, 将任务分解为更小的独立任务模块方便调度, 从而达到更高的计算能效.

渐进式优化设计的算法基本步骤如下.

输入. 芯粒数量 $N$ 、芯粒集合 $S$ 、工作负载 $W$ 以及超参数 $\beta$ (默认为 2).

输出. 芯粒组合列表 $C$.

Step1. 选取 $S$ 中的芯粒元素，与 $W$ 中的不同工作负载生成笛卡儿积，通过计算工具获得单个芯粒元素在不同工作负载下计算性能与能耗参考表 $R$.

Step2. 选取 $S$ 中不同元素，通过对于较大任务的划分，创建新的工作任务表 $T$：

Step2.1. 选取 $R$ 中全部周期数的中位数以及计算周期数均值两者中较小的元素作为标准 $D$；

Step2.2. 依据标准 $D$ 划分 Step1 的芯粒任务. 若芯粒计算周期数大于 $D$，则将任务按照通道划分成多个计算任务；计算周期数小于 $D$ 的计算任务不修改. 将修改后的计算任务数量、周期数以及能量消耗记录加入工作任务表 $T$，并依据 $T$ 中芯粒计算周期最大值与最小值的比值降序进行工作负载集合 $W$ 的排序.

Step3. 选取 Step2 中 $T$ 总时钟周期数最小的芯粒元素，构造并初始化 $N$ 个这样的芯粒组成的同构芯粒集合 $C$，获得初始化无任务负载的芯粒结构.

Step4. 依据 Step2.2 中生成的负载表顺序，加入新的工作负载，并将负载依次尝试加在所有类型的芯粒上，每次考虑调整单个芯粒，尝试将该芯粒上的全部任务分摊至同类的其他芯粒，并加入新的芯粒，对加入新的芯粒重新进行该类型芯粒间的负载均衡操作. 若芯粒间计算周期数比值大于等于 $\beta$，则主要依据划分后的时钟周期与划分后的能耗乘积大小来判断是否为占优的芯粒；否则，从芯粒的计算能耗指标来选取占优的芯粒. 如果存在占优芯粒，那么确认芯粒的更换并尝试继续更换该种芯粒；否则，尝试下一种芯粒直到全部尝试完毕.

Step5. 继续 Step4，直至负载全部添加完毕.

Step6. 输出最后的 $C$.

面对芯粒数量较多、拓扑结构复杂的情况，可以将芯粒按照数量均匀划分成多个组，通过递归迭代的方式进行芯粒的选择与优化. 例如，针对一个包含 1 000 个计算芯粒的芯片，可以先行将芯粒划分成 10 组，每组 100 个同类芯粒，使用本文算法进行粗粒度的芯粒组迭代选择优化；待初次迭代完毕后，将所有的芯粒继续拆分至 100 个组，每组 10 个芯粒，再次使用本文算法在现有芯粒集合基础上进行进一步的调整优化；依此循环迭代，直至芯粒组拆分完毕. 这样可以有效地减少迭代次数，提高算法在芯粒数量较大情况下的迭代效率.

## 5 实验与结果分析

### 5.1 实验方法与参数

为了验证异构芯粒神经网络加速器的加速效果，本文实验选择了 Eyeriss 和 NVDLA 这 2 种数据流完全不同的神经网络加速芯片作为芯粒的设计参考，组成了异构的芯粒集合.

为了公平性起见，实验部分统一了 2 种芯粒的设计工艺，修改 NVDLA 设计中的位宽至 16 位，并修改了 Eyeriss 的片上缓存大小与计算单元数量，以确保两者在位宽设计、理论性能以及面积上的接近，其设计参数如表 1 所示.

**表 1 Eyeriss 和 NVDLA 设计基准对比**

| 设计参考 | 工艺/nm | 面积/mm² | 位宽/位 | 计算单元数量 | 全局缓存容量/KB |
|---|---|---|---|---|---|
| NVDLA | 45 | 13.02 | 16 | 256 | 129 |
| Eyeriss | 45 | 13.14 | 16 | 256 | 1 875 |

Timeloop[19]是一个能够通过设置体系结构参数、数据流参数以及工作负载网络参数，以提供精准时钟周期、面积以及能耗的开源神经网络加速器模拟器. 本文实验使用 Timeloop 来获取相关体系结构的实验数据，并测试本文提出的异构多芯粒设计的实际效果.

### 5.2 测试标准

实验首先从常见的轻量神经网络结构中采样选取了具有代表性的 8 个层，计算类型有卷积层、逐通道卷积层以及全连接层，每种类型的计算层均选取了多个不同的参数维度，构造了一个简单的测试网络，其参数如表 2 所示. 所选取的各层在计算模式和计算规模上差距较大，且计算层的类型比例接近于实际的负载情况，在边缘计算任务中具有足够的代表性.

**表 2 测试网络结构**

| 测试网络 | 输入维度 | 卷积核维度 | 计算类型 |
|---|---|---|---|
| 层 1 | 224×224×3 | 32×3×3×3 | 卷积 |
| 层 2 | 112×112×32 | 3×3×32 | 逐通道卷积 |
| 层 3 | 112×112×32 | 1×1×32×64 | 卷积 |
| 层 4 | 112×112×64 | 3×3×64 | 逐通道卷积 |
| 层 5 | 56×56×64 | 1×1×64×128 | 卷积 |
| 层 6 | 14×14×256 | 1×1×256×512 | 卷积 |
| 层 7 | 7×7×1 024 | 1×1×1 024×1 024 | 卷积 |
| 层 8 | 1×1×1 024 | 1 024×1 000 | 全连接 |

为了简单起见，本文实验选取一个 3×3 的芯粒阵列作为实验的芯粒规模. 为了测试的公平性，实验参照现有的 SIMBA 结构，通过等比例缩小的方式，构建了一个同样芯粒数量的同构神经网络加速器作为实验的参考设计基准. 依据第 4 章的粗粒度迭代算法，得到一个包含 5 个

Eyeriss 芯粒和 4 个 NVDLA 芯粒的异构芯粒神经网络加速器作为优化后的设计, 与同构芯粒的神经网络加速器在能耗与性能上进行比较.

最后, 通过在 MobileNet 和 ShuffleNet 上使用上述 2 种芯粒来生成芯粒组合, 以验证本文算法的正确性, 并测试在这些负载上异构芯粒神经网络加速器的加速效果.

### 5.3　测试结果

首先, 实验分别在具有 9 个芯粒的同构神经网络加速器 SIMBA 结构与按照算法搭建的异构多芯粒加速器上运行了测试网络, 并给出了实际的能耗与时钟周期数.

如图 5 所示, 展示了 2 种加速器在指定负载下的能量消耗对比. 可以看到, 使用异构芯粒的神经网络加速器在大多数任务中与使用同构芯粒的神经网络加速器的能耗非常接近. 经计算, 在测试网络结构上, 异构芯粒的神经网络加速器比同构芯粒的设计 SIMBA 约有 1%的能耗提升.

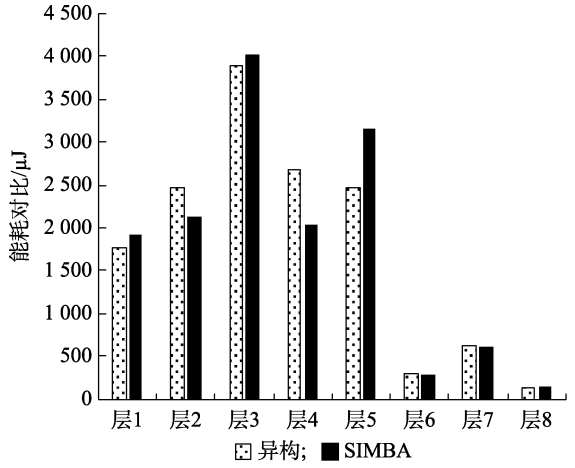



图 5　异构神经网络加速器的能耗对比

如图 6 所示, 展示了异构多芯粒神经网络加速器和采用同构芯粒的神经网络加速器 SIMBA 间的性能对比. 可以看到, 异构多芯粒神经网络加速器在计算效率上的优势.

在逐通道卷积的计算中, 受单一采用 NVDLA 芯粒组成的同构神经网络加速器数据流的限制, 每个计算任务在计算单元上均只能从输入/输出的通道的维度展开, 因此, 对于单通道的逐通道卷积计算而言, 其计算单元利用率较低, 性能也要远低于异构的神经网络加速器. 在多通道的卷积层计算过程中, 异构芯粒的加速器与同构芯粒的加速器性能差距不大. 在全连接层的计算过程中, 出于能耗考虑, 异构芯粒的神经网络加速器性能略低

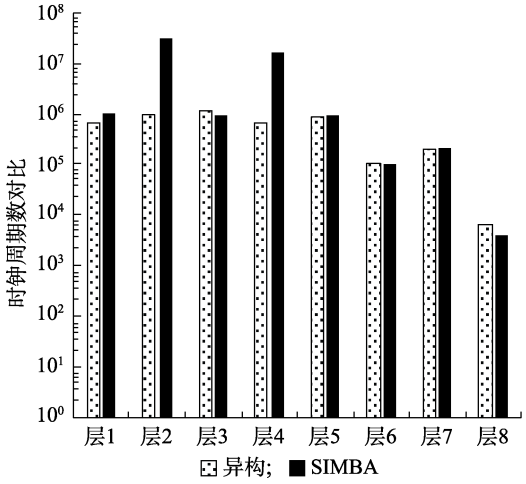



图 6　异构神经网络加速器的性能对比

于同构芯粒的神经网络加速器, 但是计算能耗要明显低于相应的同构神经网络加速器. 经 5 次测试,与同构芯粒设计的神经网络加速器 SIMBA 相比, 采用异构芯粒的神经网络加速器在测试网络结构上能获得 6.15~7.43 的加速比.

为了测试异构芯粒神经网络加速器的加速效果并验证优化算法的正确性, 实验选取了简化的 MobileNet 与 ShuffleNet 作为验证网络, 分别选取 $\beta$=2, 4, 在生成的异构芯粒神经网络加速器和 SIMBA 上同时进行了测试, 结果如表 3 所示.

**表 3　不同测试网络加速效果**

| 测试网络 | 芯粒数量 | | 能耗/μJ | $10^{-4}$×计算周期 |
|---|---|---|---|---|
| | NVDLA | Eyeriss | | |
| MobileNet | 6 | 3 | 4587 | 46.49 |
| | 9 | 0 | 4278 | 20.07 |
| ShuffleNet | 2 | 7 | 3112 | 19.32 |
| | 9 | 0 | 2075 | 108.43 |

可以看到, 通过调节 $\beta$ 的大小可以控制周期和能耗在异构设计中的重要性. 此外, 通过这些网络结构也可以验证异构设计的优势, 这样的异构芯粒神经网络加速器分别在 MobileNet 和 ShuffleNet 上取得了 2.30 与 5.60 的加速比.

## 6　结　语

本文探索了一种针对边缘计算设备进行优化的异构多芯粒神经网络加速器设计. 现有的面向负载优化的神经网络加速器(如 Thinker 等)面临设计复杂、设计成本高且制造成本高的问题, 因此, 本文采用芯粒技术构建面向负载优化的神经网络

加速器, 提高了内部设计的复用性.

一方面, 使用芯粒技术可以支持将多种小芯片作为芯片制造的基本单元, 从而维持较低的制造成本; 另一方面, 芯粒技术也为芯片设计提供了区别于 IP 的新的基本设计单元, 有助于降低芯片设计的难度. 而现有的基于芯粒技术的神经网络加速器采用同构的芯粒结构(如 SIMBA), 并未考虑从设备工作环境与负载的角度进行设计优化.

本文主要针对边缘计算设备上的固定负载, 预计算芯粒上的计算负载, 将计算任务在通道上进行划分, 通过逐任务、逐芯粒迭代方式构建异构多芯粒神经网络加速器, 与现有的同构芯粒组成的神经网络加速器相比, 在控制能耗的同时, 可以取得较高的加速比. 未来的工作将从芯粒选择优化算法与通信模式方面进行改进, 进一步提高异构芯粒神经网络加速器的计算效率.

## 参考文献(References):


[1] Redmon J, Divvala S, Girshick R, *et al.* You only look once: unified, real-time object detection[C] //Proceedings of the Computer Vision and Pattern Recognition. Los Alamitos: IEEE Computer Society Press, 2016: 779-788

[2] Hu X, Yang W J, Wen H, *et al.* A lightweight 1-D convolution augmented transformer with metric learning for hyperspectral image classification[J]. Sensors, 2021, 21(5): Article No.1751

[3] Yurtsever E, Lambert J, Carballo A, *et al*. A survey of autonomous driving: common practices and emerging technologies[J]. IEEE Access, 2020, 8: 58443-58469

[4] Zhai Y Z, Ding B, Liu X, *et al*. Decentralized multi-robot collision avoidance in complex scenarios with selective communication[J]. IEEE Robotics and Automation Letters, 2021, 6(4): 8379-8386

[5] Way T, Bertrand E. Machine learning on the edge: azure ML and azure IoT edge integration[J]. MSDN Magazine, 2019, 34(2): 26-31

[6] Jouppi N, Young C, Patil N, *et al*. Motivation for and evaluation of the first tensor processing unit[J]. IEEE Miceo, 2018, 38(3): 10-19

[7] Kurniawan A. IoT projects with NVIDIA jetson nano: AI-enabled internet of things projects for beginners[M]. Berkeley: Apress, 2021

[8] Yin S Y, Ouyang P, Tang S B, *et al*. A 1.06-to-5.09 TOPS/W reconfigurable hybrid-neural-network processor for deep learning applications[C] //Proceedings of the Symposium on VLSI Circuits. Los Alamitos: IEEE Computer Society Press, 2017: C26-C27

[9] Shao Y S, Cemons J, Venkatesan R, *et al*. Simba: scaling deep-learning inference with chiplet-based architecture[J]. Communications of the ACM, 2021, 64(6): 107-116

[10] Mahajan R, Sankman R, Patel N, *et al*. Embedded multi-die interconnect bridge (EMIB) - a high density, high bandwidth packaging interconnect[C] //Proceedings of the 66th IEEE Electronic Components and Technology Conference. Los Alamitos: IEEE Computer Society Press, 2016: 557-565

[11] Teja S, Sundar R, Deepesh J, *et al*. 3.2 Zen: a next-generation high-performance ×86 core[C] //Proceedings of the IEEE International Solid-State Circuits Conference. Los Alamitos: IEEE Computer Society Press, 2017: 52-53

[12] Howard A G, Zhu M L, Chen B, *et al*. MobileNets: efficient convolutional neural networks for mobile vision applications[OL]. [2021-11-01]. https://arxiv.org/abs/1704.04861

[13] Zhang X Y, Zhou X Y, Lin M X, *et al*. ShuffleNet: an extremely efficient convolutional neural network for mobile devices[C] //Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Los Alamitos: IEEE Computer Society Press, 2018: 6848-6856

[14] Chen Guilin, Ma Sheng, Guo Yang. Survey on accelerating neural network with hardware[J]. Journal of Computer Research and Development, 2019, 56(2): 240-253(in Chinese)
(陈桂林, 马胜, 郭阳. 硬件加速神经网络综述[J]. 计算机研究与发展, 2019, 56(2): 240-253)

[15] Chen Y H, Krishna T, Emer J S, *et al*. Eyeriss: an energy-efficient reconfigurable accelerator for deep convolutional neural networks[J]. IEEE Journal of Solid-State Circuits, 2017, 52(1): 127-138

[16] Rui X, Ma S, Wang Y H, *et al*. Configurable multi-directional systolic array architecture for convolutional neural networks[J]. ACM Transactions on Architecture and Code Optimization, 2021, 18(4): Article No.42

[17] Rui X, Ma S, Wang Y H, *et al*. HeSA: heterogeneous systolic array architecture for compact CNNs hardware accelerators[C] //Proceedings of the Design, Automation & Test in Europe Conference & Exhibition. Los Alamitos: IEEE Computer Society Press, 2021: 657-662

[18] Wade M, Anderson E, Ardalan S, *et al*. TeraPHY: a chiplet technology for low-power, high-bandwidth in-package optical I/O[J]. IEEE Micro, 2020, 40(2): 63-71

[19] Parashar A, Raina P, Shao Y S, *et al*. Timeloop: a systematic approach to DNN accelerator evaluation[C] //Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software. Los Alamitos: IEEE Computer Society Press, 2019: 304-315