于雪昂, 罗璟嵩, 刘博生, 等. 基于块浮点的混合精度卷积神经网络加速器[J]. 广东工业大学学报. doi: 10.12052/gdutxb.250056.
Yu Xueang, Luo Jingsong, Liu Bosheng, *et al*. The hybrid precision convolutional neural network accelerator based on block floating point[J]. Journal of Guangdong University of Technology. doi: 10.12052/gdutxb.250056.

基于块浮点的混合精度卷积神经网络加速器

于雪昂，罗璟嵩，刘博生，武继刚

（广东工业大学　计算机学院, 广东　广州 510006）

摘要: 随着深度学习中卷积神经网络(Convolutional Neural Network, CNN)模型的不断发展，计算复杂度和硬件资源消耗成为了限制计算效率的重要瓶颈。本文提出了一种基于块浮点(Block Float Point, BFP)的混合精度卷积处理单元(Hybrid Processing Element, HPE)，该单元通过使用数字信号处理器(Digital Signal Processor, DSP)代替传统的查找表(Look-Up Table, LUT)，结合数据打包技术，在硬件架构中优化了卷积计算单元的设计。该设计通过灵活切换INT4和BFP8两种计算模式，显著提高了计算性能并降低了硬件资源消耗。实验结果表明，HPE在使用混合精度(INT4和BFP8)计算模式时，相比于基准设计，LUT和寄存器(Flip-Flop, FF)的开销显著减少，且硬件资源使用效率分别提高了123.40%和58.16%。此外，通过数据打包技术，HPE的计算加速比达到传统方法的2倍，极大地提升了计算性能。本文的研究为深度学习加速提供了高效的硬件解决方案，具有广泛的应用潜力，特别是在需要高效计算和资源优化的深度学习任务中。

关键词: 块浮点；卷积神经网络；混合精度；数字信号处理器；硬件加速器
中图分类号: TP391.4　　**文献标志码:** A　　**文章编号:** 1007–7162(2025)00–0001–07

The Hybrid Precision Convolutional Neural Network Accelerator Based on Block Floating Point

Yu Xueang, Luo Jingsong, Liu Bosheng, Wu Jigang

(School of Computer Science and Technology, Guangdong University of Technology, Guangzhou 510006, China)

Abstract: With the continuous development of convolutional neural networks (CNNs) in deep learning, computational complexity and hardware resource consumption have become the significant bottlenecks limiting computational efficiency. This paper proposes a hybrid processing unit (HPE) based on Block Floating Point (BFP) , which optimizes the design of the convolution computation unit in the hardware architecture by replacing the traditional Look-up Table (LUT) with DSP and employing data packing techniques. This design enables flexible switching between INT4 and BFP8 computation modes, significantly improving computational performance and reducing hardware resource consumption. Experimental results show that, when using a hybrid precision (INT4 and BFP8) computation mode, HPE significantly reduces LUT and FF overhead, with hardware resource utilization efficiency increasing by 123.40% and 58.16%, respectively, compared to the baseline. Furthermore, the data packing techniques enable the HPE to achieve 2× higher throughput than the conventional implementations. This study provides an efficient hardware solution for deep learning acceleration, with broad potential applications, especially in deep learning tasks requiring high computational efficiency and resource optimization.

Key words: block floating point; convolutional neural networks; hybrid precision; digital signal processor; hardware accelerator

深度学习作为人工智能领域的核心技术，凭借其强大的特征提取和非线性建模能力，在图像识别、自然语言处理、自动驾驶等多个领域取得了显著成果[1-3]。其中，卷积神经网络(Convolutional Neural

收稿日期： 2025-03-05　　**录用日期：** 2025-04-22　　**网络首发日期：** 2025-05-22
基金项目： 国家自然科学基金–青年科学基金资助项目 (62302102)
作者简介： 于雪昂(1999–)，男，硕士研究生，主要研究方向为软硬件协同计算，E-mail：912213233@qq.com
通信作者： 武继刚(1963–)，男，教授，博士，主要研究方向为高性能计算，E-mail：asjgwucn@outlook.com

Networks, CNN)作为深度学习的典型代表[4-6]，通过局部感知和权值共享机制，在处理图像、视频等高维数据时展现出了卓越的性能，成为计算机视觉任务中的核心算法。然而，随着网络层数的增加和模型规模的扩大，卷积神经网络的计算复杂度和资源消耗也呈指数级增长，这对硬件算力和内存带宽提出了更高的要求，亟需寻找更为高效的计算方法来应对这一挑战[7-9]。

为应对这一问题，块浮点(Block Floating Point, BFP)作为一种高效的数值表示方法[10-11]，逐渐在深度学习优化中崭露头角。块浮点通过将一组数据共享同一个指数，在保证较高计算精度的同时，显著降低了内存占用和带宽需求。这一特性使得块浮点在CNN的计算中表现尤为突出，尤其是在处理大规模特征图和权重矩阵时，能够有效降低存储和传输开销，从而提升整体计算效率[12]。

另一方面，混合比特卷积在CNN加速中被广泛使用，足以证明混合精度卷积计算的重要性。混合精度通过在不同计算阶段灵活使用不同精度的数据类型，在保证模型精度的同时，显著降低了计算复杂度和内存占用[13-15]。

在此背景下，混合精度BFP为卷积操作提供了更加高效的解决方案。混合精度BFP在传统块浮点的基础上，引入了动态位宽调整机制，使其能够根据计算需求灵活选择4位或8位精度，从而在存储效率和计算精度之间实现更优的平衡。由于其在效率与精度之间的有效平衡，混合精度BFP展现出了在推进先进数字处理技术和加速深度学习任务中的巨大潜力，为计算密集型应用开辟了全新的优化路径。

最近的研究表明，专用硬件解决方案，如现场可编程门阵列(Field Programmable Gate Array, FPGAs)，在混合精度计算方面取得了显著进展。具体而言，FPGA通过其可定制的硬件架构，提供了高度灵活性，能够针对特定的混合精度卷积操作进行优化并实现高效并行处理[16-17]。为了最大化混合精度的优势，一些研究采用了BFP8浮点数据表示方法，能够在减少精度损失的同时提高计算效率。混合精度卷积通过为卷积层的不同部分分配不同的精度级别，进一步优化了计算资源的使用[18]。例如，FAST采用可变精度的BFP表示(如8位或更低比特位)，并将其应用于基于FPGA的卷积计算。结合随机舍入计算技术，FAST实现了高效的性能提升。然而，现有研究普遍忽略了充分利用数字信号处理器(Digital Signal Processor, DSP)的能力，导致FPGA资源的利用效率未能得到充分发挥，这为进一步优化硬件性能提供了研究空间。

本文设计了一种基于BFP的混合精度卷积处理单元，其中使用了DSP代替查找表(Look-Up Table, LUT)实现尾数的乘加计算，将混合的INT4和BFP8数据打包送入DSP中，使DSP能够单次处理两组尾数，提高了整体的硬件资源利用效率；除此之外，该卷积处理单元能够完成INT4和BFP8计算模式的灵活切换，实现了高性能的卷积计算；在该卷积处理单元的基础上，本文提供了一种基于本文所提出卷积处理单元的数据映射方式，结合了INT4和BFP8的两种数据输入模式，优化了卷积计算期间的DSP利用率。

1 研究背景与动机

1.1 块浮点

图1为浮点数与块浮点数两种数据表达形式的对比。块浮点(Block Floating-Point, BFP)是一种用于提高存储效率和计算性能的数值表示方法，主要用于深度学习、信号处理和科学计算等领域[19]。BFP通过在一个数据块内共享相同的指数，而每个数值仍然保留独立的尾数，从而减少存储需求并优化计算效率。相比标准浮点数，BFP在指数存储方面更加紧凑，适用于数值范围较集中的数据处理任务。

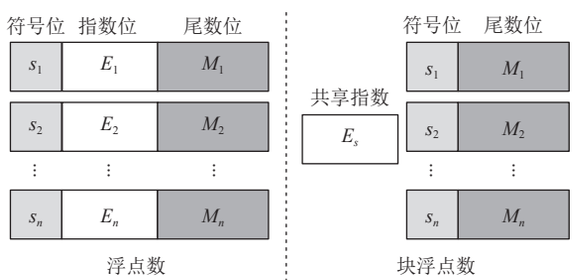

图 1 浮点数与块浮点数的对比

Fig.1 The comparison of the FP and BFP numbers

BFP的计算过程主要包括3个步骤：首先，确定数据块内的共享指数E_s，通常取决于块内所有浮点数的指数$E_i(i=1,2,\cdots,n)$的最大值。

$$E_s=\max(E_1,E_2,\cdots,E_n) \tag{1}$$

其次，对所有数值进行指数对齐，将每个数的尾数M_i调整为

$$M_i^{'}=M_i\times 2^{(E_i-E_s)},i=1,2,\cdots,n \tag{2}$$

最后，所有数值使用共享指数E_s进行存储，计算时基于该指数执行数学运算。这种方法减少了指数存储的冗余，提高了并行计算效率。然而，由于所有数值使用相同的指数，块内数值的动态范围受限，可能会导致尾数部分精度损失。尽管如此，在存储和计算效率要求较高的应用中，BFP仍然是一种高效的数值表示方式。

1.2　卷积神经网络加速器

CNN因其强大的特征提取能力，在计算机视觉、自然语言处理和信号处理等领域得到广泛应用。然而，CNN的计算复杂度较高，尤其是卷积运算涉及大量的乘加操作(Multiply Accumulate, MAC)，导致计算资源消耗大、存储访问频繁。为了提高计算效率，CNN加速器的发展重点在于降低计算开销、优化数据存储，并利用低精度计算减少功耗[20-21]。

1.2.1　混合精度卷积神经网络

混合精度计算(Hybrid Precision Computing, HPC)是 CNN加速器中的重要优化策略，它允许不同的计算阶段采用不同的数值表示，以减少数据传输、存储需求和计算开销。相比于传统的单精度浮点(FP32)计算，混合精度CNN主要采用更低精度的数据格式，如半精度浮点(FP16)、8-bit浮点(FP8)、甚至定点(INT4)计算。通常，CNN计算过程中的前向传播、反向传播、权重存储等不同部分可以使用不同精度。

这种混合精度计算方式能够在保证模型精度的同时，提高 CNN 计算的能效比，适用于深度学习训练和推理加速。

1.2.2　基于块浮点的卷积神经网络

BFP是一种用于优化CNN计算的数据表示方法[22-23]，它通过在一个数据块(如特征图、卷积核块)内共享指数，减少指数存储需求，并优化数据计算。CNN计算中的关键操作，如卷积、矩阵乘法和批量归一化(Batch Normalization, BN)，均涉及大量浮点计算。采用BFP进行CNN加速时，具有卷积计算优化、数据存储压缩、低功耗计算等优势。

BFP结合混合精度计算，在CNN加速器中展现出较高的计算效率和存储优化能力，特别适用于存储受限的硬件平台，如AI加速芯片、边缘设备和高性能计算系统[24-25]。

1.3　动机

卷积神经网络(CNN)作为推动深度学习发展的核心引擎，已在图像识别、自然语言处理等领域展现出革命性影响。然而，随着模型层数深化与参数量激增，其计算复杂度和硬件资源消耗呈现指数级攀升趋势，这一矛盾在卷积层的乘加运算(MAC)中尤为突出：大规模特征图与权重矩阵的频繁存取导致内存带宽压力剧增，而高精度计算需求进一步加剧了算力与能效的失衡。传统基于浮点运算和查找表(LUT)的计算架构虽能保障计算精度，却面临存储资源占用过高、硬件扩展性受限等固有缺陷，尤其在边缘设备等资源约束场景中，LUT的静态位宽分配机制已成为制约计算效率提升的关键瓶颈。

为解决上述问题，混合精度计算技术通过动态调整数据位宽(如INT4定点数与BFP8块浮点数)，在计算精度与硬件效率间构建可调节平衡。其中，BFP表示法采用块内指数共享策略，可将存储密度提升至传统浮点格式的1.6倍，显著优化了大规模卷积核的数据压缩效率。然而，现有BFP硬件实施方案仍存在两大技术局限：其一，LUT密集型乘法单元导致逻辑资源利用率低下；其二，固定位宽计算模式难以适配CNN不同层级对精度敏感度的差异性需求。

基于此，本文提出一种融合异构计算与动态精度调控的创新架构：首先，采用DSP阵列替代传统LUT乘法单元，利用DSP原生高位宽特性实现双数据流并行计算，突破LUT资源利用率的天花板效应；其次，设计INT4/BFP8混合精度自适应切换协议，结合跨层数据打包技术降低存储带宽需求。该方案通过算法层精度需求与硬件层资源约束的协同映射，旨在攻克传统架构中计算密度与能效比的帕累托前沿难题，为高实时性、低功耗的AI应用场景提供新一代硬件加速基础。

2　硬件架构

2.1　架构设计

图2为基于BFP的混合精度卷积处理单元。相比于传统的浮点数表示，该加速器通过共享指数的方式减少数据存储开销，并能够支持定点计算以进一步降低功耗和提高计算性能。该设计主要包括了输入缓存模块、数字信号处理器(DSP)、累加模块(Accumulator, ACC)、指数处理模块(Exponential Processor, EP)及输出寄存器等，各模块协同工作，以支持高效的CNN计算。

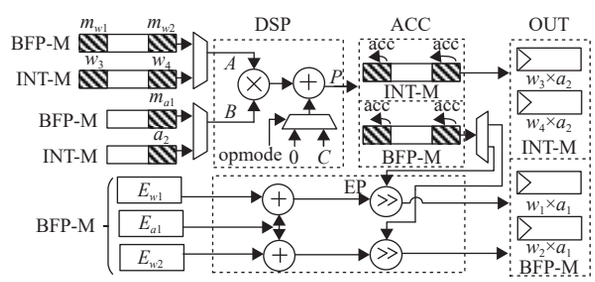

图 2　HPE架构图

Fig.2　Architecture of HPE

输入缓存模块包含了两种寄存器，分别为BFP-M和INT-M，每种寄存器各有两个，分别用于处理权重数据和激活值数据。与传统的独立浮点存储方式不同，BFP-M采用共享指数的方式，即每个数据块内的数值共享相同的指数E_s，仅存储独立的尾数M_i，以减少指数存储冗余。为了兼容不同精度的计算需求，INT-M用于存储INT数据，从而在低精度计算模式下

进一步优化计算性能。此外，输入缓冲模块将对权重数据进行打包处理，将两个权重的尾数打包为成对数据，如图中的m_{w1}和m_{w2}、w_3和w_4，分别占据数据的低位和高位，共同作为DSP模块的输入。

数字信号处理器(DSP)用于实现数据的乘法计算。在本设计中，DSP既支持BFP数据的计算，也支持INT数据计算。在BFP或INT计算模式下，DSP将分别对尾数部分或是INT数据进行乘法操作；除此之外，由于成对尾数内的低位数据(即m_{w2}和w_4)被视为无符号数据，因此需要进行调整以纠正输出差异，通过添加寄存器C来实现。C被定义为

$$C = s_{w2} \times m_{a1} \times 2^n \tag{3}$$

其中，s_{w2}为BFP-M中权重的低位数据的符号位，m_{a1}为激活值的尾数，n则表示数据位宽。

累加模块(ACC)用于完成BFP的尾数或INT类型数据的累加操作，同样支持打包后的两组数据的同时处理。当完成累加计算后，ACC模块将根据计算模式的不同，来决定数据的走向：若此时为INT数据计算模式，累加结果将被直接作为输出送入输出寄存器；若为BFP数据计算模式，则将累加结果送入EP模块进行移位计算。

指数处理模块(EP)仅在BFP数据计算模式下生效，用于实现BFP数据中指数部分的加法计算和基于指数的尾数移位计算，以完成基于BFP的卷积计算操作。当尾数完成移位后，计算结果将被送入输出寄存器。输出寄存器将作为内部数据计算结果和外部存储之间的缓存使用。

图3为两次卷积计算单元的完整计算流程。其中，图3(a) 表示BFP模式下的输入数据和输出数据之间的联系；图3(b) 表示INT模式下的输入数据和输出数据之间的联系；图3(c) 则为该卷积处理单元的完整卷积计算流程示例。在第一个周期中，BFP-M和INT-M中的前两份权重数据被打包为成对数据，与对应的激活值数据一同送入DSP模块进行计算。由于流水线设计，在第二个周期时，DSP完成计算后，后两份权重数据也将被打包与对应的激活值一同送入DSP模块进行计算。DSP模块的计算结果被送入ACC模块进行累加计算。累加结果为INT数据的计算结果时，数据被直接送入输出寄存器；累加结果为BFP的计算结果时，数据被送入EP模块进行移位计算。指数的计算与尾数的计算同时进行，在EP模块完成指数的加法操作后，将等待尾数计算结果送入EP模块，随后进行移位计算，并将计算结果送入输出寄存器。

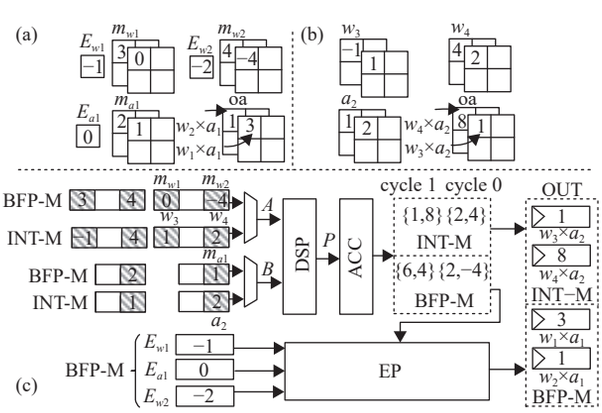

图 3 HPE计算实例

Fig.3 Example of HPE

2.2 数据流映射

如图4所示的数据流映射中，本设计的BFP卷积计算单元持续对输入特征与权重进行逐元素乘法，并在多个计算周期内复用数据以提升整体效率。具体而言，尾数与指数会按时序顺序依次输入，随后在各个阶段被重复使用，从而避免重复读写开销，形成流线化的计算流程。如图中标注的m_{21}, m_{22} 等尾数以及E_{21}, E_{22}等指数都可以在后续运算中被再次调用，而无需重新加载或对齐。

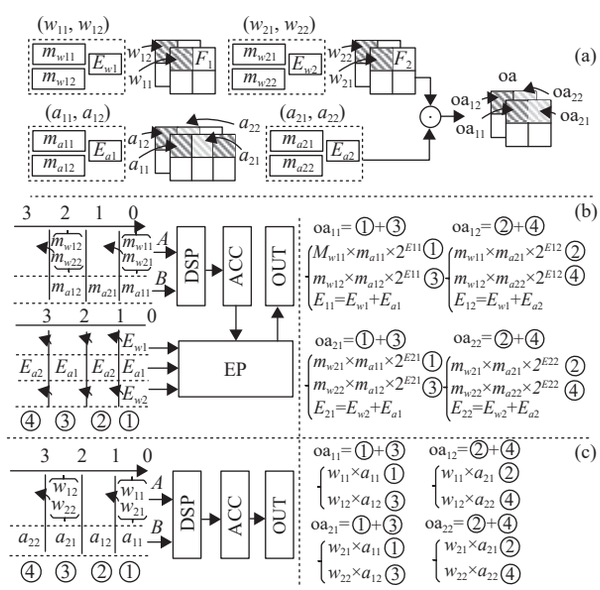

图 4 HPE数据流映射

Fig.4 Mapping dataflow based on HPE

图4(a) 展示了在HPE架构中，基于块浮点(Block Floating Point, BFP)格式的数据流映射关系。该图以两个权重块(w_{11}, w_{12})、(w_{21}, w_{22})与两个激活块(a_{11}, a_{12})、(a_{21}, a_{22})为例，说明了输入张量在经过块浮点解码后，如何按照乘法−累加的数据路径映射到输出结果oa的各个位置。每个张量块均由一组整数部分(mantissa)与一个共享指数(exponent)构成，分别记作m_w、m_a和E_w、E_a。图中箭头清晰标识了张量元素之间

的对应关系与流向：每个输出元素oa_{ij}由多个权重与激活元素经指数对齐后的乘积累加得到，例如，oa_{11}对应于$w_{11}\cdot a_{11}$与$w_{21}\cdot a_{21}$，而oa_{22}则由$w_{12}\cdot a_{12}$和$w_{22}\cdot a_{22}$累加生成。图4(a) 描述了输入与输出张量之间的元素级映射关系，作为数据流路径建立的重要组成部分。

图4(b) 为BFP模式下的数据流映射，其中展示了数据是如何输入和处理的。每个周期输入一组激活值的尾数，而对于权重的尾数，由于进行了打包处理，则两个周期输入DSP一次。与此同时，指数部分每个周期输入一组激活值的指数，并且由于权重数据使用了BFP的共享指数，每次只需要传入E_{w1}和E_{w2}两个指数分别对应打包的成对尾数高位和低位两个部分。除此之外，图4(b) 右侧的公式展示了在完整流程下，卷积计算单元内部计算的数据流是如何执行的。

图4(c) 为INT模式下的数据流映射，同图4(b) 。INT类型的权重数据被打包为成对数据，将两个权重值分别作为高位和低位，与激活值一同送入DSP模块进行计算。

3　实验结果及讨论

3.1　实验设置

本文使用了文献[26]中的BiE和文献[27]中的Tender作为测试基准。其中，BiE通过双共享指数设计，优化了对异常值的处理，提升了量化精度，并且在ASIC上设计了一种基于BFP的DNN加速器，在传统BFP计算的基础上，实现了对于激活值中异常值的计算；Tender针对INT4的量化方式进行设计，通过“2的幂”比例因子和隐式重量化技术，实现了高效的INT4量化方案，并在ASIC上实现了加速器设计。本文所设计的混合精度处理单元(HPE)则是在传统BFP的硬件计算上，使用DSP代替LUT，采用了一种数据打包方式，实现了两组尾数同时计算，并且加入了INT4和BFP8的模式切换，能够同时支持两种精度的混合计算。

实验中使用的CNN网络模型为文献[28]中的VGG16和文献[29]中的ResNet18，数据集为ImageNet。本文基于Vivado工具实现了设计，验证的平台是ZNYQ XC7VX485，其中部署的DSP为文献[30]中的DSP48E1。DSP48E1由一个25位预加法器、一个25×18位乘法器、一个48位运算逻辑单元和一个48位累加器组成。

本文将分两步对HPE进行验证实验：首先，分别对只使用LUT、使用DSP和使用DSP并打包数据3种卷积计算实现方式下的硬件资源开销进行了评估，以证明DSP和数据打包处理相较于传统方法的优势；其次，在硬件资源开销、硬件资源使用效率和加速比3个方面，使用本文设计与基准BiE和Tender进行了比较，以证明本文设计相较于前沿研究的优势。

3.2　实验结果

表1展示了实现卷积计算单元的3种方法，分别是：① 只使用LUT进行计算；② 使用DSP代替LUT进行计算，且不进行数据打包；③ HPE所提出的使用DSP并采用数据打包的方法进行计算。由表中数据可知，使用方法②时，LUT使用量在输入精度为INT4、BFP8和混合精度的情况下分别降低至方法①的23.04%、26.28%和68.96%；FF使用量分别降低至方法①的63.31%、57.78%和73.50%。由此可见，尽管方法②需要额外消耗16个DSP，但是可以在较大程度上降低LUT和FF开销。

表 1　本文设计的硬件资源开销

Table 1　Hardware resources overhead of the proposed design

资源类型	①只使用LUT			②使用DSP+不打包数据			③使用DSP+打包数据		
	INT4	BFP8	混合	INT4	BFP8	混合	INT4	BFP8	混合
LUT	408	586	770	94	154	531	168	243	564
FF	278	379	468	176	219	344	220	276	416
DSP	0	0	0	16	16	16	8	8	8

HPE(即方法③)相较于方法①，在3种不同输入精度的情况下，LUT使用量降低至方法①的41.18%、41.47%和73.25%；FF使用量降低至方法①的79.14%、72.82%和88.89%。由此可见，方法③相较于方法①同样能降低LUT和FF的开销。同时，尽管方法③相较于方法②并未降低LUT和FF的开销，但是DSP的数量从16个减少到了8个，降低了50%。这是由于HPE利用了DSP高位宽的特点，使用了数据打包的方法，将两组输入数据打包为成对数据，分别存储在低位和高位，这使得两组数据可以在一个DSP中进行计算，从而使DSP开销降低了一半。

表2展示了HPE与基准设计的硬件开销对比。由表中数据可知，相较于只使用INT4的Tender和只使用BFP8的BiE，使用了混合精度(INT4+BFP8)的LUT开销分别增加了72.65%和22.22%，FF开销分别增加了68.35%和32.20%。在HPE利用DSP代替LUT进行计算之后，LUT和FF开销相较于Tender分别只增加了26.46%和49.64%；相较于BiE，LUT开销降低了10.48%，FF开销增加了17.51%。由此可见，经过HPE的优化，混合精度所带来的额外开销问题得到了有效的缓解，特别是在和BFP8的基准进行比较时，消除了混合精度导致的LUT额外开销。

表 2 与基准设计的硬件开销对比

Table 2 Comparison of hardware overhead with baseline

资源类型	Tender	BiE	混合精度(LUT)	混合精度(HPE)
LUT	446	630	770	564
FF	278	354	468	416
DSP	0	0	0	8

图5为HPE与基准设计在不同网络模型下的硬件资源使用效率对比，使用了吞吐量与LUT开销的比值来进行评估。由图可知，在使用VGG16和ResNet18时，HPE的硬件资源使用效率相较于BiE提升了123.40%，相较于Tender提升了58.16%。由此可见，由于使用混合精度计算，硬件架构的吞吐量得到了有效的提高。同时，使用DSP代替LUT完成乘法计算能够抵消掉混合精度带来的额外LUT开销，并且基于DSP的数据打包方法能够在一定程度上提高HPE的吞吐量。因此HPE架构的硬件资源使用效率相较于BF8和INT4基准都得到了显著的提升。

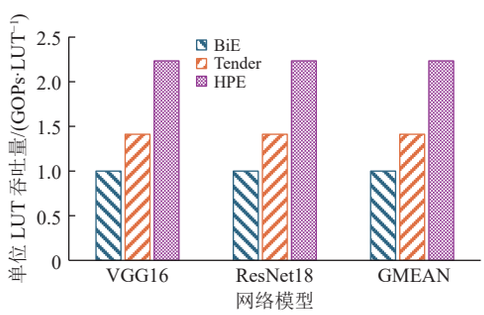

图 5 与基准测试的硬件资源使用效率对比

Fig.5 The comparison of hardware resources usage

图6展示了HPE和基准测试的归一化加速比的比较结果。由图可知，HPE的加速比达到了BiE和Tender的2倍，这是由于HPE采用了数据打包的方法，利用DSP内部结构位宽较大的特点，同时处理两组输入数据，因此硬件的处理效率达到了传统方法的2倍。

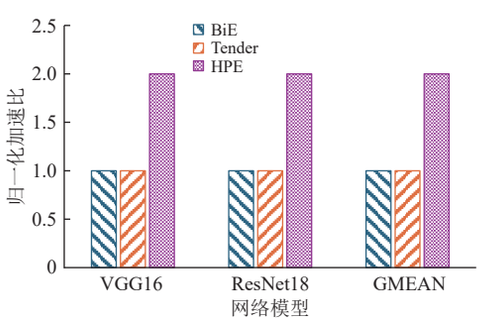

图 6 与基准设计的加速比对比

Fig.6 The comparison of speedup

值得注意的是，HPE设计高度依赖DSP的乘加能力与位宽结构。除本文所基于的DSP48E1外，更高端FPGA芯片中的DSP48E2、DSP58等结构在计算位宽、后处理灵活性方面更具优势。例如，DSP48E2支持可编程的后移位与更丰富的数据路径选择，有助于进一步优化数据打包策略与精度控制逻辑。未来HPE可在此基础上探索对更复杂卷积算子的支持，提升在大模型与高频运行场景下的适用性与能效表现。

4 结论

本文提出了一种基于BFP的混合精度卷积处理单元(HPE)，通过采用DSP替代LUT计算单元，并结合数据打包技术，实现了卷积计算硬件架构的多维度优化。实验结果表明：在硬件资源效率方面，HPE的混合精度(INT4/BFP8)模式显著降低了LUT和触发器(FF)的资源开销，相较于基准设计(BiE和Tender)，硬件资源使用效率在VGG16和ResNet18模型中分别提升123.40%和58.16%；在计算性能方面，数据打包技术通过特征图与权重的协同压缩存储，使计算加速比达到传统方法的两倍，充分验证了架构的高效性。上述成果表明，HPE通过“硬件替代(DSP-LUT)+ 精度自适应(INT4/BFP8)+ 数据流优化(打包压缩)”的协同设计策略，有效解决了计算效率与资源消耗的平衡难题。未来研究可进一步探索HPE在边缘计算芯片与异构计算平台中的移植适配性，并扩展其对Transformer等新兴模型的加速支持，以推动深度学习硬件加速技术的普适化发展。

参考文献：

[1] DARVISH R B, LO D, ZHAO R, *et al*. Pushing the limits of narrow precision inferencing at cloud scale with microsoft floating point[J]. Advances in Neural Information Processing Systems, 2020, 33: 10271-10281.

[2] DRUMOND M, LIN T, JAGGI M, *et al*. Training DNNs with hybrid block floating point[J]. Advances in Neural Information Processing Systems, 2018, 31.

[3] NI C, LU J, LIN J, *et al*. LBFP: logarithmic block floating point arithmetic for deep neural networks[C]//2020 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS) . Ha Long: IEEE, 2020: 201-204.

[4] LI Z, LIU F, YANG W, *et al*. A survey of convolutional neural networks: analysis, applications, and prospects[J]. IEEE Transactions on Neural Networks and Learning Systems, 2021, 33(12): 6999-7019.

[5] 谢伟立, 张军. 一种基于多尺度的多层卷积稀疏编码网络[J]. 广东工业大学学报, 2024, 41(6): 125-132. doi: 10.12052/gdutxb. 230205.
XIE W L, ZHANG J. A multi-layer convolutional sparse coding network based on multi-scale[J]. Journal of Guangdong University of Technology, 2024, 41(6): 125-132. doi: 10.12052/gdutxb.230205.

[6] 章云, 王晓东. 基于受限样本的深度学习综述与思考[J]. 广东工业大学学报, 2022, 39(5): 1-8
ZHANG Y, WANG X D. A review and thinking of deep learning with a restricted number of samples[J]. Journal of Guangdong University of Technology, 2022, 39(5): 1-8.

[7] LO Y C, LIU R S. Bucket getter: a bucket-based processing engine for low-bit block floating point (BFP) DNNs[C]//Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture. New York: IEEE, 2023: 1002-1015.

[8] LIANG Y, LU L, XIAO Q, *et al*. Evaluating fast algorithms for convolutional neural networks on FPGAs[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2019, 39(4): 857-870.

[9] AHMAD A, PASHA M A. FFConv: an FPGA-based accelerator for fast convolution layers in convolutional neural networks[J]. ACM Transactions on Embedded Computing Systems (TECS) , 2020, 19(2): 1-24.

[10] BHATTI U A, TANG H, WU G, *et al*. Deep learning with graph convolutional networks: an overview and latest applications in computational intelligence[J]. International Journal of Intelligent Systems, 2023, 2023(1): 8342104.

[11] ZHANG F, GAO Z, HUANG J, *et al*. HFOD: a hardware-friendly quantization method for object detection on embedded FPGAs[J]. IEICE Electronics Express, 2022, 19(8): 20220067.

[12] FAN H, WANG G, FERIANC M, *et al*. Static block floating-point quantization for convolutional neural networks on FPGA[C]//2019 International Conference on Field-Programmable Technology (ICFPT) . Tianjin: IEEE, 2019: 28-35.

[13] WEN J B, FENG Y, LI Z Q. A high-throughput hardware architecture for bilateral filter with configurable convolution and cost-effective MAC unit[J]. IEICE Electronics Express, 2024, 21(13): 20240276.

[14] WANG Z, IKEDA M. High-throughput and fully-pipelined ciphertext multiplier for homomorphic encryption[J]. IEICE Electronics Express, 2024, 21(6): 20230628.

[15] HUANG W, WU H, CHEN Q, *et al*. FPGA-based high-throughput CNN hardware accelerator with high computing resource utilization ratio[J]. IEEE Transactions on Neural Networks and Learning Systems, 2021, 33(8): 4069-4083.

[16] DARVISH R B, ZHAO R, ELANGO V, *et al*. With shared microexponents, a little shifting goes a long way[C]//Proceedings of the 50th Annual International Symposium on Computer Architecture. Orlando: IEEE, 2023: 1-13.

[17] JUNAID M, ARSLAN S, LEE T G, *et al*. Optimal architecture of floating-point arithmetic for neural network training processors[J]. Sensors, 2022, 22(3): 1230.

[18] YANG D, LI J, HAO G, *et al*. Hardware accelerator for high accuracy sign language recognition with residual network based on FPGAs[J]. IEICE Electronics Express, 2024, 21(4): 20230579.

[19] XU Y, WANG S, LI N, *et al*. Design and implementation of an efficient CNN accelerator for low-cost FPGAs[J]. IEICE Electronics Express, 2022, 19: 20220370.

[20] LIAN X, LIU Z, SONG Z, *et al*. High-performance FPGA-based CNN accelerator with block-floating-point arithmetic[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019, 27(8): 1874-1885.

[21] OOTOMO H, YOKOTA R. Recovering single precision accuracy from tensor cores while surpassing the FP32 theoretical peak performance[J]. The International Journal of High Performance Computing Applications, 2022, 36(4): 475-491.

[22] OH Y H, KIM S, JIN Y, *et al*. Layerweaver: maximizing resource utilization of neural processing units via layer-wise scheduling[C]//2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA) . Seoul: IEEE, 2021: 584-597.

[23] KIM Y, JANG J, LEE J, *et al*. Winning both the accuracy of floating point activation and the simplicity of integer arithmetic[C]//The Eleventh International Conference on Learning Representations. Kigali: Open Review, 2023.

[24] ZENG Z, YANG L, LI G, *et al*. A high speed and high accurate floating-point EEG signal bandpass filter based on FPGA[C]//2023 17th International Conference on Complex Medical Engineering (CME) . Suzhou: IEEE, 2023: 16-19.

[25] ZHANG S Q, MCDANEL B, KUNG H T. Fast: DNN training under variable precision block floating point with stochastic rounding[C]//2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA) . Seoul: IEEE, 2022: 846-860.

[26] ZOU L, ZHAO W, YIN S, *et al*. BiE: bi-exponent block floating-point for large language models quantization[C]// Forty-first International Conference on Machine Learning. Vienna: PMLR, 2024.

[27] LEE J, LEE W, SIM J. Tender: accelerating large language models via tensor decomposition and runtime requantization[C]//2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA) . Buenos Aires: IEEE, 2024: 1048-1062.

[28] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[C]//3rd International Conference on Learning Representations (ICLR 2015) . San Diego: Computational and Biological Learning Society, 2015.

[29] 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. Las Vegas: IEEE, 2016: 770-778.

[30] PRASAD B M P, PARANE K, TALAWAR B. High-performance NoC simulation acceleration framework employing the xilinx DSP48E1 blocks[C]//2019 International Symposium on VLSI Design, Automation and Test (VLSI-DAT) . Taiwan: IEEE, 2019: 1-4.

（责任编辑：王威娜　英文审核：费伦科）