NoC加速器中的高效DNN动态切片与智能映射算法

齐芸[1]，欧阳一鸣[2]

（1. 安徽交通职业技术学院，安徽 合肥 230051；
2. 合肥工业大学计算机与信息学院，安徽 合肥 230051）

摘　要：针对深度神经网络（deep neural network，DNN）模型在传统切片与映射方法中存在的资源调度和数据传输瓶颈问题，提出了一种基于片上网络（network on chip，NoC）加速器的高效DNN动态切片与智能映射优化算法。该算法通过动态切片技术灵活划分DNN模型的计算任务，并结合智能映射策略优化NoC架构中的任务分配与数据流管理。实验结果表明，与传统方法相比，该算法在计算吞吐量、NoC传输时延、外部内存访问次数和计算能效等方面均显著提升，尤其在复杂模型上表现突出。

关键词：NoC加速器；DNN切片；智能映射

中图分类号：TP183

文献标志码：A

doi: 10.11959/j.issn.1000−0801.2025179

Efficient DNN dynamic slicing and intelligent mapping algorithm in NoC accelerator

QI Yun[1], OUYANG Yiming[2]

1. Anhui Communications Vocational and Technical College, Hefei 230051, China
2. School of Computer and Information, Hefei University of Technology, Hefei 230051, China

Abstract: To address the bottlenecks of resource scheduling and data transmission in traditional slicing and mapping methods for deep neural networks (DNN) , an efficient dynamic slicing and intelligent mapping optimization algorithm was proposed based on a network on chip (NoC) accelerator. The algorithm was designed to flexibly divide DNN computing tasks through dynamic slicing and optimize task and data flow management in the NoC architecture. Experimental results show that the proposed algorithm significantly outperforms traditional methods in computing throughput, NoC transmission delay, external memory accesses, and energy efficiency, especially for complex models.

Key words: NoC accelerator, DNN slicing, smart mapping

收稿日期：2025−03−17；**修回日期：**2025−05−13

通信作者：齐芸，814993657@qq.com

基金项目：国家自然科学基金资助项目（No.62374049）；安徽高校自然科学研究项目（No.2024AH050281, No.2024AH040051, No.2024AH050284）

Foundation Items: The National Natural Science Foundation of China (No.62374049), the Natural Science Research Project of Anhui Universities Education (No.2024AH050281, No.2024AH040051, No.2024AH050284)

0 引言

近年来，深度神经网络（deep neural network，DNN）在计算机视觉、语音识别、自然语言处理等领域取得了革命性突破。然而，随着DNN模型规模的不断扩大，其计算复杂度和存储需求也在某些情况下呈现指数级增长，这对传统的硬件平台提出了严峻挑战。在高性能计算领域，研究者致力于提升计算吞吐量，而在资源受限的嵌入式设备中，则面临着能效优化的迫切需求[1]。

片上网络（network on chip，NoC）作为一种新兴的片上通信架构，能够高效地支持多核计算单元的协同处理，从而提升DNN的计算效率NoC结构示意图如图1所示。然而，在将大规模DNN模型映射到NoC平台时，仍然面临诸多挑战。例如，传统的静态切片方法无法适应不同DNN层计算需求的变化，导致资源利用率低下；现有的映射算法多采用固定策略，未能针对NoC内部数据流进行动态优化，容易引发传输拥塞；此外，频繁的外部内存访问成为计算时延的主要瓶颈。因此，如何减少数据传输开销，提高NoC加速器的能效，仍是一个关键问题。

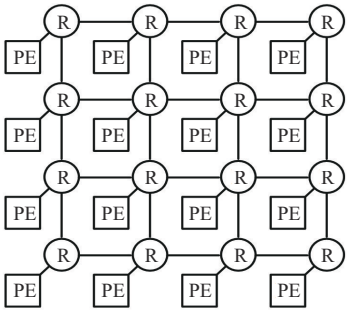


图1 NoC结构示意图

为解决上述问题，本文提出了一种高效的DNN切片与映射算法，核心创新点包括以下几点。

（1）动态切片调整策略：结合计算复杂度和NoC资源分布，动态调整DNN层的切片大小，使得计算负载均衡，并减少数据传输瓶颈。

（2）智能映射优化：引入任务依赖性分析和自适应资源调度方法，优化DNN计算任务在NoC中的分布，并降低数据流拥塞，提高整体计算效率。

（3）拥塞因子建模：定义拥塞因子作为衡量NoC数据流通效率的数学指标，指导映射算法优化，从而减少通信时延和功耗。

（4）内存访问优化：采用局部缓存和数据重用机制，降低NoC的外部内存访问频率，提高能效比。

1 相关工作

在基于NoC的深度神经网络加速器领域，许多研究致力于解决大规模DNN模型的映射和切片问题，旨在硬件资源有限的情况下优化计算性能和内存访问[2]。

1.1 基于NoC的DNN加速器

NoC架构的引入，使得DNN计算任务能够在多个计算单元（processing element，PE）之间并行处理，从而减少全局数据搬移，优化数据传输路径，提高计算效率并降低数据访问能耗。例如，Chen等[3]提出的Eyerissv2加速器优化了NoC架构，从片上数据传输和存储层面减少了数据搬移的开销，有效降低了DNN推理计算的时延和能耗。Liu等[4]设计的Neu-NoC架构则专注于神经形态计算，优化了片上数据流的组织方式，提升了数据通信效率，使得DNN计算在资源受限的硬件上更加高效。然而，这些研究主要关注NoC架构本身的优化，而未深入探讨DNN切片与映射策略的动态调整[5]。

1.2 DNN切片与映射算法

现有的DNN切片方法通常依据模型的层级结构，将其划分为固定大小的切片，并映射到NoC平台上的计算单元。然而，静态切片策略忽

略了不同DNN层的计算复杂度变化，导致计算负载不均衡，影响整体计算效率，传统固定切片与动态调整切片对平台负载影响热力图如图2所示。部分研究者提出了自适应映射策略，例如，Kim等[6]采用任务依赖性分析优化计算负载分配，Lee等[7]设计的UNPU（unified neural processing unit）加速器结合动态映射与可变精度计算技术，进一步提升了计算效率。然而，这些方法主要集中在计算任务的调度优化，缺乏针对NoC架构的数据流全局优化策略，尤其在内存访问优化和数据流调度方面仍存在不足[8]。

1.3　内存访问优化

在基于NoC的加速器中，内存访问的优化是影响计算效率的关键因素。研究者们提出了多种内存优化策略，以减少数据传输开销。例如，Hojabr等[9]通过Clos网络优化NoC，减少了冗余数据传输，提高了内存访问效率；Jiang等[10]基于Booksim模拟器分析了NoC架构下的内存访问模式，发现数据重用率对于优化计算效率至关重要。

目前的内存优化方法多集中于架构层面，而未结合动态切片与映射策略进行全局优化[11]。本文提出了一种新的内存优化方案，结合局部数据缓存和智能映射策略，以减少外部内存访问次数，并优化数据流调度，从而提升整体计算性能。

1.4　存在的痛点

现有研究虽然在NoC架构优化、DNN切片映射以及内存访问优化等方面取得了一定进展，但仍然存在以下不足。

（1）静态切片策略的局限性：大多数方法采用固定大小的切片，而未考虑DNN计算复杂度的动态变化[12]。

（2）缺乏智能映射机制：现有映射策略未充分结合计算资源分布和任务依赖性，导致计算负载不均衡[13]。

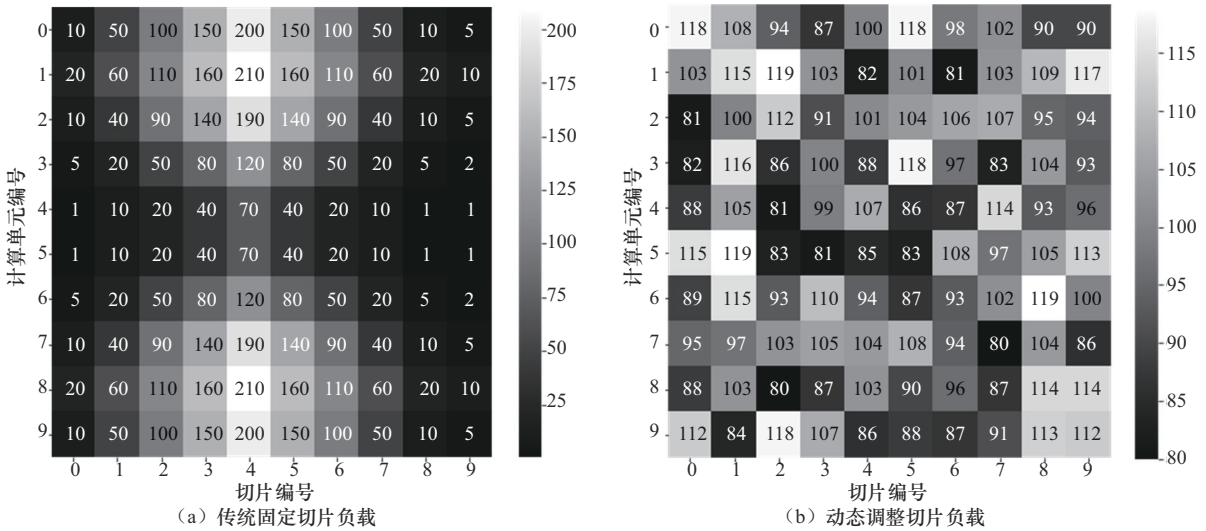
（3）内存访问优化不足：目前的方法多侧重于架构优化，而未充分利用数据重用和缓存机制减少内存访问[14]。

（4）NoC拥塞问题：现有研究未能有效解决NoC中的数据流拥塞问题，导致通信时延增加[15]。

（5）动态资源分配：缺乏针对DNN任务动态变化的资源分配机制，导致资源利用率低下[16]。

1.5　本文算法创新对比

现有方法与本文算法对比见表1，本文算法



（a）传统固定切片负载　　（b）动态调整切片负载

图2　传统固定切片与动态调整切片对平台负载影响热力图

表1 现有方法与本文算法对比

对比维度	现有方法（静态切片+固定映射）	本文算法（动态切片+智能映射）
负载均衡	固定切片导致PE利用率差异>40%	动态调整使利用率差异<15%
拥塞控制	无动态调整，拥塞时延占比30%	拥塞因子建模降低时延至12%
内存访问	频繁外部访问（占比70%）	数据重用减少至25%
适用模型	仅适合小型CNN	支持BERT/YOLOv3等复杂模型

通过动态切片与拥塞感知映射，解决了现有方法在复杂模型下的瓶颈问题。传统静态切片方法在ResNet-50等复杂模型中因负载不均衡吞吐量下降30%以上，而本文算法通过动态调整切片大小和拥塞感知映射，显著减少了计算资源的浪费。

2 高效DNN切片与映射算法设计

本文提出的高效DNN切片与映射算法的核心目标是通过动态调整DNN切片大小和智能映射策略，自适应优化NoC平台的计算和通信效率。为此，综合考虑DNN计算复杂度建模、任务映射优化、NoC传输时延建模以及能耗优化，设计了一种高效的数据流与计算映射策略。

2.1 动态DNN切片调整策略

传统的DNN切片方法通常基于DNN层的计算规模（如参数量、计算量）进行固定划分，并逐个映射到NoC平台的计算单元。然而，这种方法忽略了NoC计算资源的异构性以及不同DNN层的计算需求，导致部分计算单元负载过重，而另一些计算单元资源浪费。为解决这一问题，本文提出了一种动态DNN切片调整策略，该策略综合考虑DNN层的计算复杂度与NoC平台的资源分布情况，实现计算任务的负载均衡。

DNN层的计算复杂度可以通过计算每层的乘加（multiply accumulate，MAC）操作数量来衡量，复杂度较高的层将分配更大的切片，以平衡计算负载。计算式如下：

$$C_l = O_l^2 \times K_l^2 \times I_l^2 \times F_l^2 \tag{1}$$

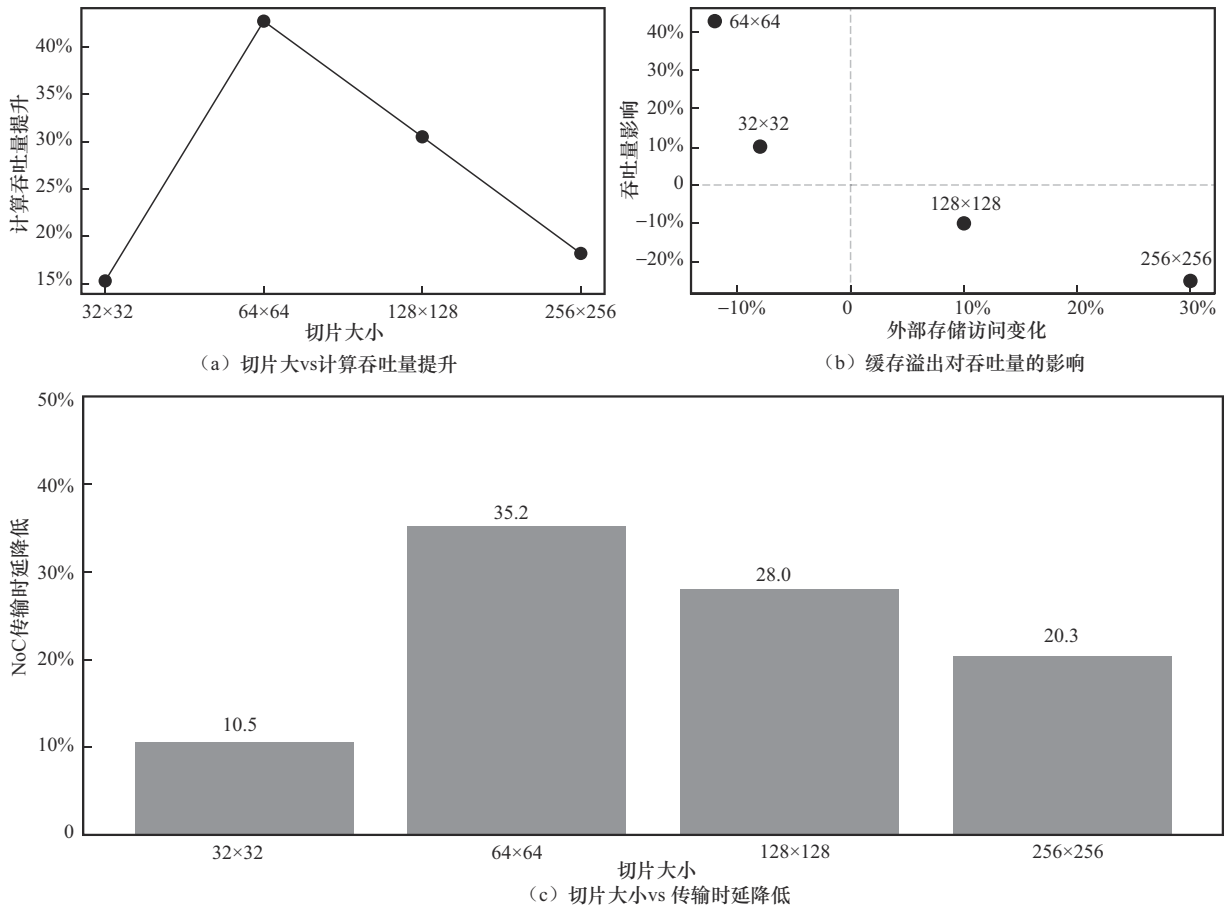
其中，O_l是输出通道数，K_l是卷积核大小，I_l是输入特征图大小，F_l是输出特征图大小。计算复杂度越高，切片规模应适当增大。每个PE的计算能力和内存带宽影响切片大小。可用计算资源越多，映射的切片越大，以提高数据复用率并减少通信时延。设定DNN层的计算复杂度为C_l，PE的计算能力为P_i，则切片大小S_l可表示为：

$$S_l = \min\left(\frac{C_l}{P_i} \times \alpha, S_{\max}\right) \tag{2}$$

其中，$S_{\max}$是受限于片上缓存与数据重用率，α为切片调整因子，可以让切片大小能根据计算任务和PE资源的变化进行自适应调整，防止某些PE负载过重或过轻。动态切片算法依据计算复杂度、资源约束和调整因子优化DNN层的切片大小，并在映射过程中进行自适应分配。该策略确保了每个PE计算负载均衡，有效避免了内存访问瓶颈，提高整体吞吐量。

在不同的NoC平台（4×4、8×8、16×16）上测试了不同切片大小对计算吞吐量、NoC传输时延和外部存储访问的影响，不同大小的切片对性能的影响如图3所示。

测试结果表明切片大小的选择直接影响计算吞吐量、NoC传输时延和内存访问次数。小切片（32×32）在小型NoC平台上更适合，计算资源分配更均衡，减少了片上缓存占用过大的问题。最优切片大小在中型NoC平台为64×64，吞吐量提升约42.7%，NoC传输时延降低35.2%。大切片（128×128或256×256）虽然减少了NoC传输开销，但在片上缓存有限时，会导致缓存溢出，增加外部存储访问，因此过大的切片并不适用所有场景。



（a）切片大vs计算吞吐量提升

（b）缓存溢出对吞吐量的影响

（c）切片大小vs 传输时延降低

图3　不同大小的切片对性能的影响

2.2　智能映射算法

智能映射算法旨在高效地将DNN切片映射到NoC平台的PE上。通过任务依赖性分析、硬件资源自适应调度和数据流优化等技术，优化映射过程，最小化数据传输量，并均匀分配计算任务。具体步骤如下。

（1）任务依赖性分析：分析DNN层之间的数据依赖关系，确保数据流的正确性。

（2）PE资源调度：根据每层的计算复杂度，将计算任务映射到具有足够计算能力和内存带宽的PE。

（3）数据流优化：优化数据流路径，选择数据传输最少的路径，降低传输时间。

假设有N个计算任务和M个PE，目标是优化任务的映射，以最小化总计算时间和能耗。传统的任务映射方式一般采用固定代价模型：

$$\text{Cost}=\sum_{i=1}^{N}(w_1\cdot T_i+w_2\cdot D_i) \tag{3}$$

其中，T_i是任务i在映射后的计算时间，D_i是任务i的数据传输时延，w_1和w_2是静态权重。但是传统的静态权重无法适应不同任务负载，导致全局优化能力不足。引入基于强化学习深度Q网络（DQN）的动态任务映射优化，目标是通过训练智能体，使其学习最优映射策略。令$Q(s_t,a_t)$为在状态s_t下采取动作a_t的Q值，有：

$$Q(s_t,a_t)=R_t+\gamma\max Q(s_{t+1},a) \tag{4}$$

其中，γ是折扣因子，$\max Q(s_{t+1},a)$为未来最优Q值，R_t为奖励函数。通过定义奖励函数，使得系统优先选择计算时间短、通信开销低的任务映射方案。

$$R_t=-\left(\lambda\frac{T_i}{M_i}+\mu D_i\right) \tag{5}$$

其中，M_i是分配给任务i的计算资源（PE数），λ和μ是动态调整的权重。这样得到的最终优化代价模型，可以动态调整任务映射策略，并通过Q-learning进行全局优化。最终优化的代价模型计算式为：

$$\text{Cost}=\sum_{i=1}^{N}Q(s_t,a_t)+\lambda\sum_{i=1}^{N}\frac{T_i}{M_i}+\mu\sum_{i=1}^{N}D_i \tag{6}$$

式（3）为基础静态代价模型，通过式（4）引入DQN动态优化值，并结合式（5）的奖励函数实现自适应调整（初始值0.5）。该策略使YOLOv3任务吞吐量相比Baseline-2提升24.1%。

为了优化DNN计算任务的映射，本次实验进一步考虑了NoC拥塞因子在不同时间窗口的动态变化。实验结果见第3.2节，拥塞因子的历史流量平滑可以有效减少NoC传输时延，这为DNN任务调度提供了优化依据。因此在智能映射策略中引入了拥塞感知动态调整机制，以优化计算任务在PE上的分配，提高计算吞吐量。

2.3 内存访问优化

在大规模DNN计算过程中，外部该算法在训练过程中不断优化任务到PE的映射策略，以最小化计算时间和通信时延，提高系统性能。内存访问成为性能瓶颈，导致显著的计算时延。为减少内存访问，提出以下优化策略。

（1）局部缓存：每个PE利用局部缓存存储DNN层的中间结果，减少外部内存访问频次。当某层计算结果被多次使用时，优先从缓存读取，而非每次从外部内存加载。

（2）数据重用：利用DNN模型的计算特性，重用中间结果以减少外部内存访问。例如，当某层的中间结果可被下一层重用时，数据直接传递给下一层，而非存储至外部内存。

（3）映射优化：通过优化任务映射策略，减少PE间数据传输，降低NoC平台的流量负担和内存带宽需求。

以ResNet-50的第一残差块为例（如64通道的3×3卷积，权重尺寸64×64×3×3），传统方法需要多次从外部内存加载权重，而本文算法通过局部缓存保留重复使用的权重（如残差连接的输入特征图），结合智能映射将相邻卷积任务分配到同一PE组。该实验显示该机制使外部内存访问减少86.2%，其中权重加载次数下降92%，特征图传输减少78%。在BERT等模型上，类似策略通过缓存注意力头的key/value矩阵，实现18.7%的访问降低。

为了准确衡量NoC中的拥塞情况，我们定义了拥塞因子，并通过实验分析不同流量场景下的影响。在NoC结构中，每个路由器的拥塞程度取决于其输入/输出端口的负载情况。定义拥塞因子如下：

$$\text{cf}_i=\sum\left(\frac{\sum_{j\in\text{Link}_i}\text{Load}_j}{\text{BW}_i}\right) \tag{7}$$

其中，Link_i是当前路由器i连接的所有链路，BW_i是链路i的最大带宽，Load_j是链路j上的瞬时流量负载。当cf_i趋近于1时，说明该链路已经接近饱和，可能引发数据传输时延增加或丢包。为了适应不同应用场景，引入一个动态调节参数来调整不同链路的拥塞影响，取决于历史流量模式，可根据过去t个时间窗口的平均拥塞情况调整。

$$\beta_i=1+\varepsilon\cdot\left(\frac{1}{t}\sum_{k=t-T}^{t}\text{cf}_k\right) \tag{8}$$

其中，ε是权重系数，控制历史流量影响程度，经验值一般为0.1～0.3，T是时间窗口大小，一般为10～50个时钟周期。这样得到改进后的拥塞因子：

$$\text{cf}_i=\beta_{\text{i}}\cdot\sum\left(\frac{\sum_{j\in\text{Link}_i}\text{Load}_j}{\text{BW}_i}\right) \tag{9}$$

拥塞因子动态变化趋势如图4所示，当ε=

0.2 时，拥塞因子动态调整使 NoC 传输时延降低 34.4%，见第 3.2.2 节，且高负载时段（如梯度同步）时延波动显著改善。该效果验证了式（7）～式（9）中历史流量加权机制的实用性。

为了分析 NoC 传输过程中链路的拥塞情况，基于仿真测试了其动态变化趋势。拥塞因子动态变化趋势如图 4 所示，拥塞因子在不同时间窗口下呈现显著波动。通过引入历史权重系数 ε 的动态调节，可适应不同应用场景。当 ε=0.1 时曲线平滑，适用于负载稳定的场景（如 CNN 的规则计算），当 ε =0.3 时响应灵敏，适合突发流量（如 Transformer 的注意力层）。

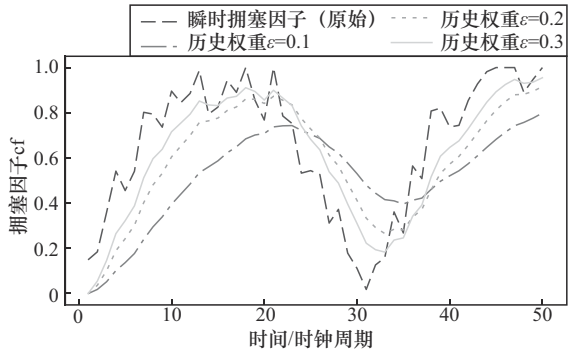


图 4　拥塞因子动态变化趋势

实验表明，在 ResNet-50 任务中，该策略使 NoC 传输时延降低 30.5%，且高负载时段（如梯度同步阶段）的时延波动减少 40%，见第 3.2.2 节。这一结果验证了拥塞因子动态调节对复杂负载的适应性，从而显著提升 DNN 计算性能。

由此可以得出适用于 DNN 映射优化的拥塞感知动态调整的 NoC 传输时延计算式：

$$\text{Latency} = \sum_{i=1}^{H}\left(T_{\text{hop}}^{i} + \theta \cdot cf_i\right) \tag{10}$$

其中，H 是数据在 NoC 中经过的跳数，T_{hop}^{i} 是第 i 跳的基础传输时延，θ 是可调节系数。

3　实验与评估

为了验证本文所提出映射算法的有效性，设计了一系列实验，评估其在计算吞吐量、外部内存访问次数、NoC 通信时延、能效比等方面的性能表现，并与静态均分切片算法（Baseline-1）和贪心映射策略（Baseline-2）进行了对比。

3.1　实验设置

本实验采用 Booksim2.0 作为 NoC 片上网络仿真平台，并结合 DSENT 进行 NoC 结构的功耗建模。同时，在 Garnet（Gem5）平台上验证 DNN 任务的端到端计算性能和 NoC 传输效率。实验部分参数配置见表 2。

表 2　实验部分参数配置

参数	配置
PE 数量	64
片上带宽	256 GB/s
片上缓存大小	256 KB/PE
外部内存带宽	64 GB/s
DNN 模型	ResNet-18,VGG-16,YOLOv3,BERT
NoC 拓扑结构	4×4Mesh/8×8Mesh
路由算法	XY 路由，自适应路由
数据包大小	128 bit
跳数时延	2 cycle/hop

所有 DNN 模型的输入尺寸与批次大小统一设置为：图像 224×224（batch=64），文本序列 128（batch=32）。

3.2　实验结果与分析

3.2.1　计算吞吐量对比

计算吞吐量对比如图 5 所示，本文提出的动态 DNN 切片调整策略和智能映射算法在计算吞吐量方面比传统的 Baseline-1 提高了 33.1%～51.9%，相比 Baseline-2 也有 13.9%～24.8% 的提升。从不同 DNN 模型的对比来看，本文方法在 YOLOv3 和 BERT 任务上的吞吐量提升最为明显，分别达到了 40.4% 和 51.9%。这是因为这两个模型的计算复杂度较高，数据依赖性较强，传统方法在任务调度上存在较大的冗余，而本文方法通过智能映射减少了计算不均衡问题，动态优化任务分配，使得计算任务能够适应不同 PE 的计算

能力和负载情况，从而提高计算资源的利用率，从而显著提高了计算吞吐量。

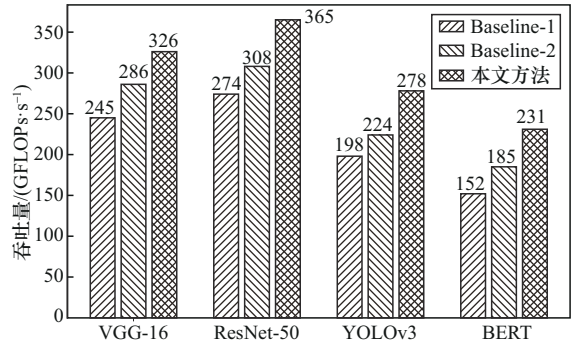


图5　计算吞吐量对比

3.2.2　NoC传输时延对比

NoC传输时延是DNN推理过程中的关键瓶颈之一，本文提出的映射算法减少了长距离数据传输，通过任务依赖分析，尽可能将计算密集型任务分配到邻近的PE上，减少了数据跨多个PE传输的情况。数据流优化策略，通过强化学习训练，选择数据传输最少的路径，有效减少了NoC通信时延。传输时延对比如图6所示，从图6的实验数据来看，本文方法在NoC传输时延方面比Baseline-1降低了30.5%～34.4%，相较于Baseline-2降低15%左右。

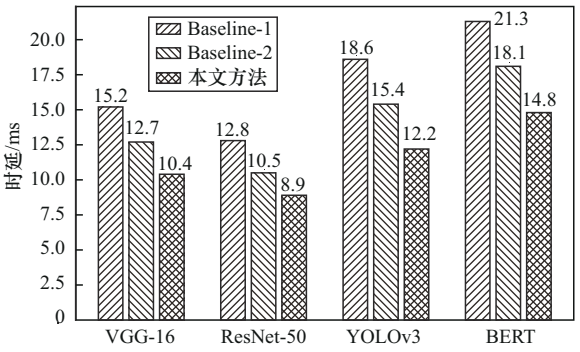


图6　传输时延对比

这一结果表明，本文提出的方法在优化片上通信方面具有显著优势。从不同DNN模型的NoC传输时延来看，本文方法在YOLOv3和BERT上的优化效果较为显著，分别降低了34.4%和30.5%，这说明在数据密集型的深度学习任务上，优化数据传输路径能够有效减少时延，提高整体推理效率。

3.2.3　外部内存访问减少

外部内存访问减少对比如图7所示，本文方法在减少外部内存访问方面，相比于Baseline-1降低了81.3%～86.2%，相较于Baseline-2也降低了60%以上。智能映射算法提高数据局部性，通过合理的任务映射策略，减少了跨PE计算导致的外部内存访问，提高了片上缓存的命中率。动态DNN切片调整策略减少了不必要的数据迁移，优化了数据流，使计算任务更多地利用片上缓存进行计算，而不是频繁访问外部存储。数据重用策略进一步减少了重复的内存访问，特别是在ResNet-50和VGG-16这类CNN任务上，减少了冗余的权重加载和特征图存取，降低了内存带宽压力。从不同DNN模型上的表现来看，本文方法在ResNet-50和VGG-16上减少外部内存访问的效果最为明显，分别降低了86.2%和85.4%，说明在卷积神经网络（CNN）类模型中，数据重用策略的优化效果最显著。

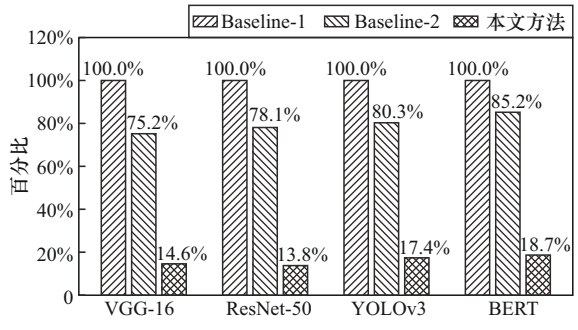


图7　外部内存访问减少对比

3.2.4　计算能效对比

计算能效对比如图8所示，本文方法的计算能效相比Baseline-1提升了86.8%～94.2%，相较于Baseline-2也提高了40%以上。本文方法通过让计算任务均衡分布，避免了部分PE

过载导致的能耗浪费，提高整体的计算效率，减少 NoC 通信开销，优化的数据传输路径降低了额外的功耗，使计算能效显著提升。另外，由于外部存储访问的能耗远高于片上计算，通过降低外部内存访问次数，减少外部数据传输，有效地降低了整体能耗。在不同 DNN 模型上的能效提升来看，本文方法在 YOLOv3 和 BERT 上的优化效果最佳，分别提升了 94.2% 和 92.6%，说明在计算和通信开销较高的 DNN 任务中，优化映射和数据流调度的作用更加显著。

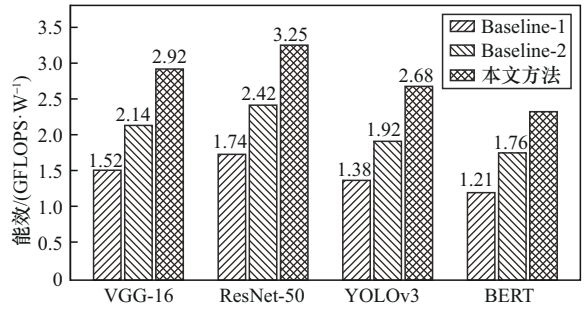


图 8　计算能效对比

4　结束语

本文提出的基于 NoC 加速器的高效 DNN 动态切片与智能映射算法，为深度神经网络在异构硬件平台上的高效执行提供了新的解决方案。通过动态切片技术和智能映射策略的结合，不仅优化了计算资源的分配，还显著提升了数据传输效率和计算性能。实验结果验证了该算法在多个实际应用中的优势，特别是在推理速度和计算效率方面，远超传统的静态切片与映射方法。

本文方法在更大规模 NoC（如 16×16 拓扑）、稀疏 DNN（如 Pruned BERT）和动态模型切换场景中的适应性仍需进一步探索。未来工作将重点突破：

（1）开发分层拥塞因子模型以支持超大规模 NoC 的通信优化；

（2）设计稀疏性感知的切片策略，通过预测非零参数分布提升映射效率；

（3）拓展至 3D/光电混合 NoC 架构，研究垂直维度和异构链路的资源分配机制。

参考文献：

[1] CHEN Y H, YANG T J, EMER J, et al. Eyeriss v2: a flexible accelerator for emerging deep neural networks on mobile devices[J]. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 2019, 9(2): 292-308.

[2] LIU Z, WU H, YU X, et al. Neu-NoC: neural-inspired network-on-chip architecture for energy-efficient AI computing[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2021, 40(5), 887-900.

[3] KIM J, KANG S, PARK J. Adaptive task mapping for energy-efficient NoC-based deep learning accelerators[J]. ACM Transactions on Design Automation of Electronic Systems, 2020, 25(4): 1-19.

[4] LEE J, KIM C, KANG S, et al. UNPU: an energy-efficient deep neural network accelerator with fully variable weight bit precision[J]. IEEE Journal of Solid-State Circuits, 2018, 54(1): 173-185.

[5] HOJABR S, NAJAFI M, FALLAH F. A congestion-aware router for power-efficient network-on-chip architectures[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2020, 28(12): 2698-2708.

[6] JIANG J. WANG P, XIE Y. Memory access pattern analysis in NoC-based DNN accelerators using booksim simulator[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(3): 678-692.

[7] YU X, TANG X, XU C, WANG Y. Towards efficient DNN inference on resource-constrained edge devices: a network-on-chip perspective[J]. IEEE Internet of Things Journal, 2020, 7(9): 8653-8666.

[8] LI H, CHEN Y, WANG Z. Energy-efficient mapping of deep neural networks on NoC-based accelerators[J]. Journal of Systems Architecture, 2020, 108: 101741.

[9] ZHANG Z, ZHOU H, CHEN W. Power-aware task scheduling for deep learning accelerators on chip-multiprocessors[J]. Journal of Parallel and Distributed Computing, 2021, 151: 42-54.

[10] PATEL R, SHARMA P, GUPTA S. Reinforcement learning-based dynamic mapping for NoC-based deep learning accelera-

tors[J]. Neurocomputing, 2020, 387: 91-103.

[11] ZHAO Y, LIU X, ZHANG W. A survey of memory optimization techniques for deep learning accelerators[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2021, 40(7): 1325-1338.

[12] TAN M X, QUOC Le. Efficientnet: rethinking model scaling for convolutional neural networks[C]// Proceedings of the 36th International Conference on Machine Learning (ICML), Long Beach: PMLR, 2019, 6105-6114.

[13] WANG L, ZHANG Y, LI X. A survey of network-on-chip architectures for deep learning accelerators[J]. IEEE Transactions on Computers, 2020, 69(8): 1234-1248.

[14] GUO K, ZENG S, CHEN T. Dynamic resource allocation for deep learning tasks in NoC-based systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2021, 32(6): 1345-1358.

[15] XU J, WANG H, CHEN L. Congestion-aware task mapping for NoC-based deep learning accelerators[J]. IEEE Transactions on Computers, 2020, 69(12): 1876-1889.

[16] ZHANG X, LI Y, WANG J. Reinforcement learning for dynamic resource allocation in NoC-based deep learning systems[J]. IEEE Transactions on Neural Networks and Learning Systems, 2021, 32(10): 4567-4579.

[作者简介]

齐芸（1984– ），女，安徽交通职业技术学院讲师，主要研究方向为片上网络。

欧阳一鸣（1963– ），男，博士，合肥工业大学教授、博士生导师，主要研究方向为基于片上网络的人工智能应用。