SCALE-Sim TPU：验证并扩展适用于 TPU 的 SCALE-Sim

Jingtian Dang 佐治亚理工学院 美国佐治亚州亚特兰大
dangjingtian@gatech.edu

Ritik Raj 佐治亚理工学院 美国佐治亚州亚特兰大

Changhai Man 佐治亚理工学院 美国佐治亚州亚特兰大

Jianming Tong 佐治亚理工学院 美国佐治亚州亚特兰大

Tushar Krishna 佐治亚理工学院 美国佐治亚州亚特兰大

arXiv:2603.22535v1 [cs.AR] 23 Mar 2026

摘要

周期精确模拟器广泛用于研究脉动阵列加速器，但由于缺乏针对真实硬件的充分验证，以及与现代机器学习编译器技术栈的集成不佳，其准确性和易用性往往受到限制。本文提出了 SCALE-Sim TPU，这是面向 TPU 风格加速器、经过验证并扩展的 SCALE-Sim v3 版本。具体而言，我们作出三项贡献：(1) 我们基于 Google TPU v4 的测量结果，验证了 SCALE-Sim 的脉动阵列通用矩阵乘法模型，并表明模拟周期数与硬件延迟呈现很强的线性相关性，从而能够建立简单的周期到延迟的映射。(2) 我们为非脉动阵列逐元素操作引入了轻量级学习型延迟模型，仅使用张量大小和形状即可将相对误差中位数控制在 3% 以下，大幅提升端到端延迟估算的准确性。(3) 我们集成了基于 StableHLO 的前端，使来自 JAX 和 PyTorch 等现代机器学习框架的工作负载能够通过统一的编译器中间表示直接进行模拟。总体而言，这些贡献提升了 TPU 上针对完整模型性能分析的周期精确模拟的保真度、覆盖范围和实用性。

CCS 概念

• **计算机系统组织** → **脉动架构**；神经网络；• **软件及其工程** → 编译器。

关键词

SCALE-Sim、张量处理单元、脉动阵列、StableHLO、性能建模

1 引言

Google 的脉动阵列加速器（例如张量处理单元（TPU））是现代机器学习工作负载的主流计算基础。因此，周期精确模拟器被广泛用于分析性能瓶颈、指导架构探索并支持基准测试。然而在实践中，现有脉动阵列模拟器存在两个关键局限，降低了其有效性。首先，由于缺乏针对真实生产规模硬件的充分验证，其预测有效性往往不明确。其次，大多数模拟器主要关注脉动内核（例如通用矩阵乘法和

卷积），同时忽略或过度简化非脉动阵列运算，并且需要专门定制的算子描述，这些描述与现代机器学习编译流水线并不契合。

本文通过验证并扩展 SCALE-Sim v3[9]，，这一面向脉动加速器的模块化周期精确模拟器，来弥补这些不足。我们的目标是实现准确的全模型延迟

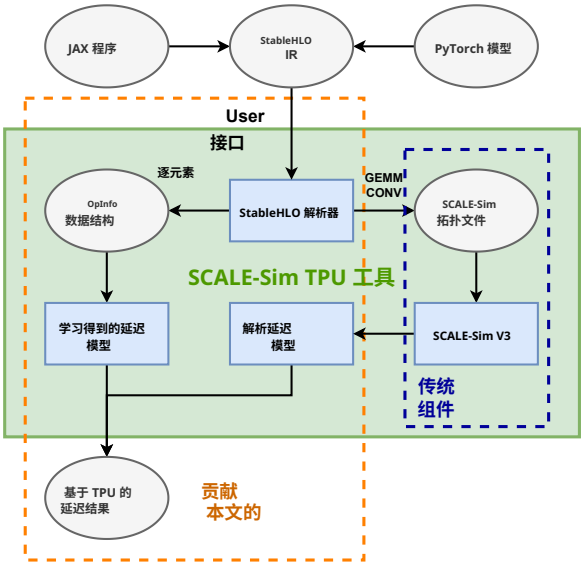


图 1：SCALE-Sim TPU 工作流与贡献概览。使用 JAX 或 PyTorch 编写的机器学习程序会被编译为 StableHLO，作为统一的输入接口。StableHLO 解析器会提取算子元数据，并将脉动阵列操作（例如通用矩阵乘法和卷积）交由 SCALE-Sim v3 处理，而非脉动阵列操作则由轻量级分析模型或学习得到的延迟模型处理。圆形表示数据工件或文件，矩形表示处理组件。绿色区域代表完整的 SCALE-Sim TPU 工具链。蓝色虚线区域标示从先前工作（SCALE-Sim v3）继承的传统组件，橙色虚线区域则标示本文引入或扩展的组件。

直接从现代编译器中间表示对 TPU 风格加速器进行估算，同时保持轻量级的建模复杂度。我们的贡献如下：

• **基于 TPU 的 SCALE-Sim 脉动阵列模型验证**。我们在多种通用矩阵乘法形状下，将模拟周期数与 Google TPUv4 上测得的内核延迟进行比较，从而对 SCALE-Sim 的脉动阵列性能模型进行基于 TPU 的验证。我们表明，一个简单的解析周期

模型结合线性周期到时间映射后，可以有效地将 SCALE-Sim 的周期数转换为 TPU 执行时间，同时也揭示了这种近似准确性较低的情况。

• **逐元素运算的可学习延迟建模**。为了更准确地计算非脉动阵列算子对端到端模型延迟的贡献，我们为 SCALE-Sim 扩展了轻量级、可学习的逐元素运算延迟模型。我们采用基于直方图的梯度提升回归，将逐元素算子延迟建模为张量大小和形状的函数。生成的模型在 TPU 硬件上具有较高的预测准确度；对于加法和 ReLU 等典型算子，相对误差中位数低于 3%，与仅考虑脉动内核相比，显著提升了全模型延迟估算的准确度。

•**基于 StableHLO、框架无关的加速器模拟用户界面**。我们引入了一个基于 StableHLO 的前端，使其能够对包括 JAX 和 PyTorch 在内的现代机器学习技术栈进行框架无关的模拟。该前端解析 StableHLO 以提取算子元数据，并识别 SCALE-Sim TPU 支持的运算，从而免去临时配置格式，并使模拟器与框架特定的中间表示解耦。受支持的脉动运算会交由经过验证的解析模型处理，而受支持的非脉动运算则由学习型延迟模型处理，从而能够对编译器生成程序中涵盖的子集进行端到端延迟估算。

这些贡献共同提升了周期精确的脉动阵列模拟的**保真度、覆盖范围**和**易用性**。SCALE-Sim TPU 提供了一套实用的工具链，可用于评估 TPU 上的真实机器学习工作负载。该工具链以硬件测量为依据，并兼容现代编译器基础设施。
图 1 总结了端到端工作流程，并突出展示了本文引入的组件。

2 背景与动机

2.1 SCALE-Sim V3

SCALE-Sim v3[9] 是一款模块化的周期精确模拟器，旨在对基于脉动阵列的加速器进行性能分析。它在早期版本的基础上进行了扩展，支持采用时空划分的多核架构、稀疏矩阵乘法、通过 Ramulator 实现的详细 DRAM 建模、片上数据布局的影响，以及通过 Accelergy 进行能耗估算。这些增强功能使 SCALE-Sim v3 能够更准确地捕捉计算、存储器与数据移动之间的交互，从而支持对现代人工智能工作负载下延迟、带宽和能耗之间的架构权衡进行详细探索。

2.2 TPU v4

TPU v4[3] 是 Google 的第四代张量处理单元，旨在利用基于脉动阵列的矩阵乘法加速大规模机器学习工作负载。与早期 TPU 代际相比，它显著提高了计算密度，提升了片上和片外存储器的带宽，并改进了互连网络，从而能够高效执行以通用矩阵乘法和卷积为主的大型工作负载。TPU v4 是面向生产规模的加速器平台，可作为代表性目标，用于根据真实硬件行为验证周期级性能模型。

2.3 动机

为何要针对真实硬件进行验证？尽管基于模拟器的研究十分常见，但许多脉动阵列模拟器很少针对量产加速器进行验证。为使 SCALE-Sim 的建模假设贴近实际，我们采用最先进的脉动阵列加速器 Google TPU 作为代表性平台，并分析 SCALE-Sim 何时能够正确反映性能行为。

为何要对逐元素运算建模？端到端机器学习模型的延迟并非仅由通用矩阵乘法和卷积等脉动阵列内核决定。现代机器学习工作负载包含大量非通用矩阵乘法算子，包括逐元素加法、激活函数、归一化和其他内存受限操作；它们累积产生的延迟可能相当可观。近期的工作负载特征分析表明，在不同模型和平台上，非通用矩阵乘法操作占总推理时间的**11.3%–73.6%**；即使经过算子融合，它们仍占端到端延迟的**15%–48%** [4]。随着通用矩阵乘法和卷积日益由专用硬件加速，这些非脉动阵列算子成为主要的性能瓶颈。因此，先前关于推理延迟预测的研究强调，为了在多样化的硬件和软件栈上准确估算端到端延迟，必须进行操作级建模，其中包括对逐元素算子的建模 [6]。因此，要实现忠实的全模型延迟估算，就必须在对通用矩阵乘法和卷积建模的同时，也对非脉动阵列逐元素操作进行建模。

为什么选择 *StableHLO*？现代机器学习工作负载通常使用 JAX 和 PyTorch 等框架编写，并且越来越多地经由编译器中间表示流转。StableHLO 提供了一种便捷、统一的表示形式，可用于提取算子形状和属性，避免为每个算子手动编写配置格式，并简化集成 [7]。

3 相关工作

以往关于加速器性能建模的研究主要集中于脉动架构的周期级模拟或解析建模。TimeLoop [8] 提供了一种灵活的解析方法，用于将深度神经网络层映射到加速器架构；但它依赖自定义配置文件，并通过解析方式（或与较低层级的模型对比）验证结果，而非直接在生产硬件上进行测量。

COCOSSim [1] 对脉动阵列和向量单元进行周期精确模拟，并报告了针对选定内核和神经网络层与 TPU v3 的比较结果。其向量单元抽象将执行过程表示为一系列归约和广播阶段，从而能够准确建模 Softmax 和注意力等算子。然而，这种基于阶段的抽象具有专门性，并未将通用逐元素运算作为一等内核提供。

SCALE-Sim TPU 基于 SCALE-Sim v3 构建，旨在弥补以往工作的不足。通过对照 TPU 硬件进行验证、将支持范围扩展至脉动内核之外，并采用 StableHLO 作为用户界面，它能够直接从编译器中间表示进行内核级

表 1：加速器性能模拟器与建模框架的比较。我们从三个方面比较已有工作：真实硬件验证、对逐元素（非脉动阵列）运算的支持，以及用户界面。SCALE-Sim TPU（本研究）已通过 TPU v4 硬件验证，在 SCALE-Sim v3 的基础上扩展了对逐元素算子的显式建模，并采用编译器级的 StableHLO 作为统一的元数据接口，从而避免使用特定框架的输入格式。

Work	真实硬件验证	逐元素运算	用户界面
SCALE-Sim v3 [9]	No	No	CSV
TimeLoop [8]	No	No	YAML
COCOSSim [1]	是（TPU v3）	No	PyTorch
SCALE-Sim TPU（本工作）	**是（TPU v4）**	**Yes**	**StableHLO**

延迟估算。由于 StableHLO 在基于 MLIR 的编译工具链中应用广泛，这种接口为评估真实工作负载提供了切实可行的途径，无需手动指定拓扑结构。

4 方法论

我们围绕 SCALE-Sim V3 的三项扩展来组织方法论：(i) 基于 TPU 的 GEMM 验证，(ii) 针对非脉动阵列算子的可学习延迟建模，以及 (iii) 基于 StableHLO 的前端。

4.1 Google TPU 上的通用矩阵乘法建模

4.1.1 针对 *TPU* 延迟验证 *SCALE-Sim*。我们在有意贴近 Google Cloud TPU v4 脉动阵列执行模型的配置下验证 SCALE-Sim。[3]

设置与模拟器配置。我们配置 SCALE-Sim，使其尽可能贴近 TPU 风格的脉动阵列执行：采用包含 128×128 个乘加网格的二维脉动阵列（与 TPU v4 相匹配）。我们使用结构化参数扫描评估形如 $\mathbf{C} = \mathbf{AB}$ 的通用矩阵乘法工作负载，其形状为 $(M, K) \times (K, N)$。

对于每种情形，我们分别以固定步长在相同范围内扫描三个维度中的每一个(M, K, N)：

- **小型：** 32–128，步长为 16
- **中型：** 128–1024，步长为 128
- **大型：** 1024–4096，步长为 512

这三个规模区间旨在体现 TPU 风格脉动阵列的不同执行行为，并不意味着片上存储容量溢出。小型 区间（32–128）涵盖矩阵维度与阵列大小相当或更小的情况，此时会出现利用率不足，流水线填充与排空效应也会占据主导地位。中型 区间（128–1024）涵盖矩阵形状适度超出阵列维度的情况，此时利用率有所提高，稳态脉动阵列执行成为决定性能的主要因素。大型 区间（ 1024–4096 ）代表在片上存储限制范围内仍可容纳、但需要 TPU 编译器采用日益复杂的分块、累加和调度决策的工作负载。在此区间内，执行行为受原始阵列占用率的影响较小，更多取决于计算如何在各个分块间划分与重叠。

在小规模区间中使用较小的步长，以捕捉细粒度的利用率效应；而在中等和大规模区间中，性能趋势随规模变化更加平滑，因此较大的步长便已足够。

收集 *SCALE-Sim* 周期数和 *TPU* 通用矩阵乘法内核运行时间。对于每种通用矩阵乘法形状，我们记录：(1) SCALE-Sim 预测的周期数；(2) 在 TPUv4 上测得的内核运行时间。测得的运行时间仅反映芯片上的执行情况，不包括 HBM 到核心的数据传输，从而隔离出 TPU 的脉动阵列的核心计算行为。

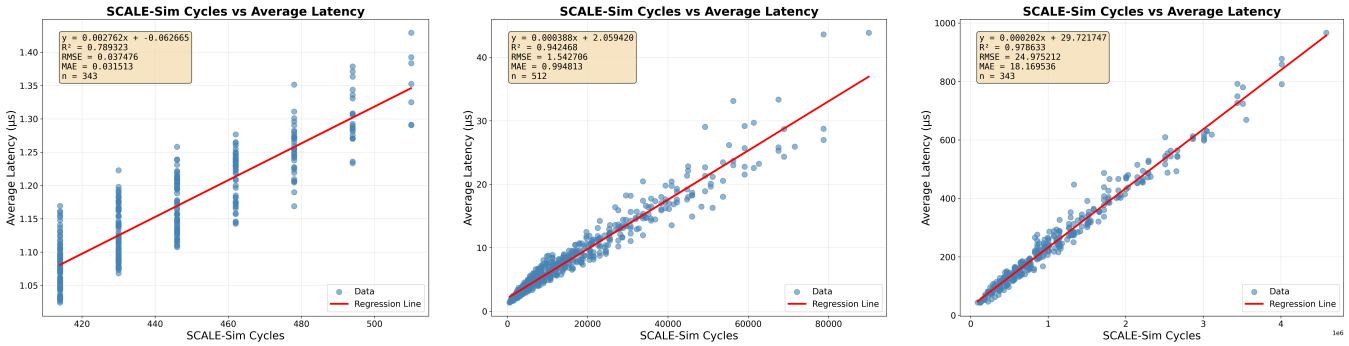
用于验证的线性回归（周期到时间的校准）。我们拟合了一个简单的线性模型，将 SCALE-Sim 周期数映射到 TPU 融合时间：$\hat{t} = \alpha \cdot$ 周期数 $+ \beta$，其中 α 表示每个模拟周期的有效时间，$\alpha \cdot$周期数表示与周期数相关的执行时间，β 则表示 SCALE-Sim 未建模的固定开销。我们分别报告三个规模区间的回归拟合结果，以突出不同规模区间的行为差异。

线性回归结果。如图 2 所示，在所有三个规模区间内，SCALE-Sim 的周期预测与实测 TPU v4 融合内核延迟均呈现稳定的线性关系。拟合回归的拟合优度中等至较高，其中 R^2 值从小规模区间的约 0.79 到中、大规模区间的 0.97 以上不等。这里，R^2 衡量线性模型所解释的实测 TPU 延迟方差比例；数值越高，表示模拟周期数解释了观测到的运行时间变化中的更大部分。这些结果表明，对于各种问题规模，模拟周期数都能捕捉 TPU 脉动阵列通用矩阵乘法执行开销中的很大一部分。

误差分析与见解。 在中等规模区间，阵列利用率较高，执行行为与理想化的脉动阵列模型最为接近，因此预测误差最小。在小规模区间，利用率不足和流水线效应会带来更大的相对波动，因此尽管绝对误差较小，R^2 仍较低。在大规模区间，方差增大，原因在于 SCALE-Sim 的计算核心模型未涵盖的一些因素，包括张量处理单元编译器的分块决策、布局变换，以及内存带宽和片上缓冲的限制。

总体而言，观察到的误差表现稳定，并随问题规模平滑变化，表明在所评估的通用矩阵乘法形状中不存在明显的系统性偏差。

4.1.2 启用 *TPU* 延迟的直接估算。基于在所有规模区间中观察到的强且一致的线性相关性（第 4.1.1 节），我们扩展了 SCALE-Sim，使其能够直接报告脉动阵列通用矩阵乘法和卷积内核的 TPU 执行时间估算值。具体而言，我们复用验证期间获得的、针对不同规模区间的线性回归函数



(a) 小规模区间的矩阵大小 **(b) 中等规模区间的矩阵大小** **(c) 大规模区间的矩阵大小**

图 2：针对三个规模区间内脉动阵列通用矩阵乘法的 SCALE-Sim 到 TPU v4 回归。每张图展示了 SCALE-Sim 预测的周期数（x 轴）与实测 TPU v4 内核延迟（y 轴）的关系，数据来自在 128×128 脉动阵列上执行的通用矩阵乘法工作负载。每个点对应扫描中的一种通用矩阵乘法形状，实线表示最小二乘线性回归；插图报告了 R^2、均方根误差、平均绝对误差和样本数。在所有规模区间中，SCALE-Sim 周期数与实测 TPU 执行时间均呈现明显的线性关系，表明模拟周期数可有效预测脉动阵列通用矩阵乘法的延迟。

，将模拟周期数映射为实际运行时间延迟。具体而言，我们复用验证期间获得的线性回归函数，将模拟周期数映射为实际运行时间延迟。

此扩展不会引入验证中未使用的额外建模假设或参数。相反，它利用观测到的关系，为用户提供以物理时间单位表示的延迟估算，从而无需手动后处理或外部校准。回归参数针对目标 TPU 平台设定；若要面向不同 TPU 代际进行预测，可重新推导这些参数。

通过这一扩展，SCALE-Sim TPU 可同时提供周期级和基于时间的性能估算，从而能够直接基于 StableHLO 派生的工作负载，进行实用的内核级延迟分析和比较。

4.2 逐元素运算的可学习延迟模型

目标算子。我们的目标是端到端机器学习工作负载中普遍存在、但不在张量处理单元的脉动阵列上执行的常见非脉动阵列算子，初步重点关注逐元素运算（例如加法、乘法、ReLU）。

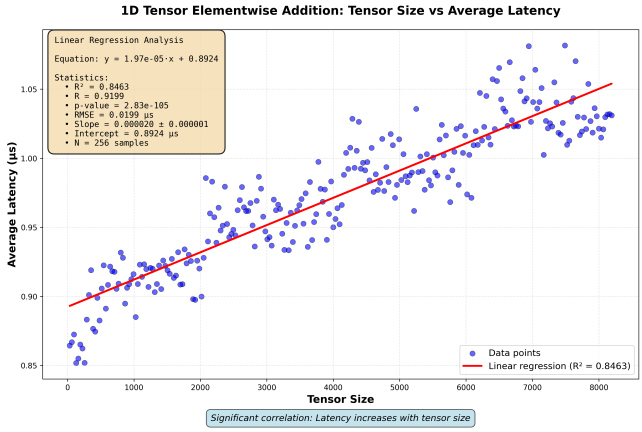
探索性分析与动机。作为初步探索步骤，我们手动收集受控的一维和二维张量形状扫描中 bf16 逐元素加法的测量数据，以了解延迟如何随张量大小变化。对于一维张量，我们将长度从 32 扫描到 8192（步长为 32）。对于二维张量，我们将每个维度从 64 扫描到 1024（步长为 64）。图 3a 和 3b 分别绘制了一维和二维扫描中平均延迟随张量大小的变化。

在两组扫描中，延迟与张量大小大致呈线性关系，这表明运行时间主要由处理的元素数量决定。不过，我们也观察到一些小幅波动：张量大小相同但形状不同的张量，其延迟可能略有差异。这促使我们在可学习的非脉动阵列延迟模型中纳入张量形状信息（除元素总数外）。

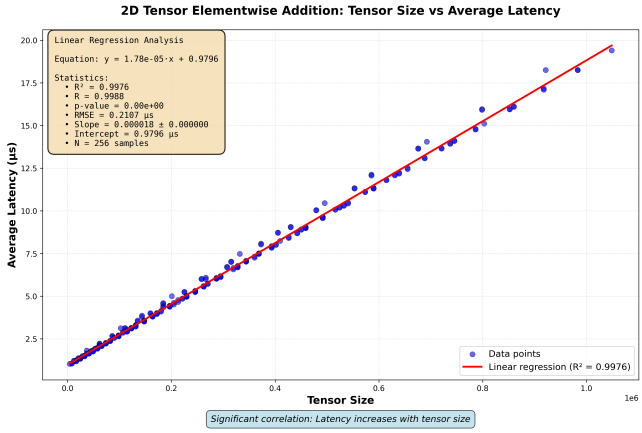
特征选择。我们使用张量大小和张量形状作为输入特征。对于逐元素操作而言，张量大小能够反映其主要的缩放行为，因为每个元素都会产生固定的计算量和内存流量。然而，测量结果显示，即使形状相同的张量大小一致，不同形状之间的延迟仍存在稳定差异，这表明性能也会受到与形状相关的因素影响，例如向量化粒度、对齐和调度阈值。将张量形状纳入模型，可以使其学习这些有规律的偏差。这两项特征都是静态已知的（编译时元数据），且独立于运行时值，因此适合轻量级的模拟器集成。

模型选择。我们使用基于直方图的梯度提升回归器（HGBR）来预测逐元素算子延迟。梯度提升决策树具备较高的数据效率，且对噪声具有鲁棒性：它们会迭代地拟合简单的决策树以捕捉残差误差，并将其汇总为一个强预测器 [2]。基于树的模型能够天然地捕捉非线性、不平滑和分段行为，因此非常适合对由分块、对齐和资源阈值等硬件边界导致的性能不连续性进行建模。基于直方图的梯度提升还通过将连续特征离散化为分箱，进一步提升效率，同时降低计算和内存开销并保持预测准确性 [5]。与单一线性模型相比，HGBR 能够表示离散跃迁和分段趋势，同时避免神经网络模型通常所需的更多数据以及调优复杂度。

训练数据。我们通过测量多种 bf16 张量形状下的逐元素算子延迟来构建训练数据集。张量大小总量按对数均匀分布采样，最大不超过固定上限（最多 ~16M 个元素）。为捕捉形状相关的性能影响，数据集包含总元素数相同的多种因式分解，使模型能够区分



(a) 一维扫描。



(b) 二维扫描。

图 3：一维（32–8192，步长 32）和二维（64–1024，每维步长 64）扫描中，bf16 逐元素加法延迟随张量大小的变化；整体近似线性增长，但会随形状略有波动。

大小相同但维度布局不同的张量。

此外，我们特意纳入了接近常见硬件相关边界条件的形状，例如维度对齐到 2 的幂次（2^n）。这些边界情况可能会触发不同的编译器调度决策、向量化策略和 TPU 硬件上的内存对齐行为，即使张量大小相近，也会导致可测量的延迟差异。纳入此类情况后，数据集既能呈现平滑的扩展趋势，也能体现对齐和调度相关因素所引发的不连续性，从而让延迟模型接触到这两类现象。

对于每种形状，我们都会多次测量延迟，并使用中位数来降低多次运行之间的噪声。

训练与验证协议。 我们使用部分张量大小训练模型，并在此前未见过的大小上对其进行评估，以衡量其泛化能力。我们对模型进行优化，使其以实时单位最小化绝对延迟误差。我们报告验证集上的绝对误差和相对误差，以衡量模型在大小张量上的准确性。

4.3 StableHLO 接口

动机与概述。 StableHLO 是一种标准化的中间表示（IR），现代机器学习框架（如 JAX 和 PyTorch）会用它来描述可在加速器上执行的程序。通过将 StableHLO 作为输入接口，SCALE-Sim TPU 无需手动将程序转换为自定义的 SCALE-Sim 拓扑文件，即可评估真实的机器学习工作负载，从而降低使用门槛，并提升不同工作负载和工具链之间的可复现性。

图 1 概述了 SCALE-Sim TPU 工作流。使用 JAX 或 PyTorch 编写的机器学习程序会被编译为 StableHLO，随后由 StableHLO 解析器提取操作级元数据，并将其转换为与模拟器兼容的表示。脉动阵列操作通过解析性能模型映射到 SCALE-Sim v3，而非脉动阵列操作则由轻量级学习型延迟模型处理。

StableHLO 解析。 StableHLO 解析器会处理编译器生成的 StableHLO 模块，并为每个操作提取一种轻量级表示。对于每个 StableHLO 操作，解析器都会记录：

- 操作类型（例如，点积_通用、卷积、加法）；
- 输入和输出张量形状；• 数据类型（例如，bf16）；
- 以及相关属性，例如卷积参数。

这些信息存储在统一的内部数据结构（OpInfo）中，使前端 IR 与后端性能模型相互解耦，并支持可扩展的算子处理。

操作转换。 解析后的 StableHLO 操作会根据其执行特征进行分类，并转换为模拟器级算子。

点积转通用矩阵乘法。 符合矩阵乘法模式的 StableHLO dot_general 操作会转换为通用矩阵乘法算子。输入张量形状和收缩维度用于推导相应的 M、N 和 K 参数，适用于C = AB，随后将这些参数传递给 SCALE-Sim 的通用矩阵乘法建模接口。此转换可直接复用 SCALE-Sim 经验证的脉动阵列性能模型。

卷积。 在适用情况下，StableHLO 卷积操作会映射为 SCALE-Sim 卷积输入。张量形状和卷积属性（例如，步幅、滤波器维度和通道数）会转换为 SCALE-Sim 卷积参数，包括 IFMAP 维度、滤波器大小、通道数和滤波器数量。这种映射保留了 SCALE-Sim 卷积模型所需的结构信息，同时排除未明确建模的预处理和数据布局转换开销。

其他运算。 目前，将不在脉动阵列上执行的运算归类为逐元素运算，包括加法、减法、乘法、最大值和最小值。对于这些运算，解析器会提取张量形状信息，并将其传递给学习型延迟模型。这种方法使得

能够高效地对现代机器学习工作负载中常见、但纯解析模型难以准确捕捉的逐元素运算进行建模。

5 评估

我们从三个与研究贡献相对应的维度评估 SCALE-Sim TPU。第 4.1.1 节介绍通用矩阵乘法验证结果（包括线性回归诊断）。

5.1 周期到延迟的映射评估（通用矩阵乘法）

我们通过比较不同矩阵大小下的预测延迟与实测 TPU 执行时间，评估通用矩阵乘法内核的周期到延迟映射精度。如图 4 所示，预测值与实测延迟呈明显正相关（$R^2 = 0.893$），表明 SCALE-Sim 能够捕捉脉动阵列执行的主要缩放行为。对于小型和大型工作负载，大多数数据点都接近理想的 $y = x$ 线，说明在这些区间内预测与实测结果较为吻合。然而，对于中等大小的工作负载，预测延迟与运行时间之间的偏差更大，且预测结果往往低估或高估运行时间。这些差异可能源于模型未涵盖的系统级影响，例如调度开销、数据移动和内核融合行为，而简单的线性映射无法捕捉这些因素。因此，尽管该映射能够对 TPU 延迟提供合理的一阶估计，但其精度会因工作负载区间而异，平均绝对百分比误差（MAPE，32.2%）等总体指标也会受到这些中间区间偏差的影响。

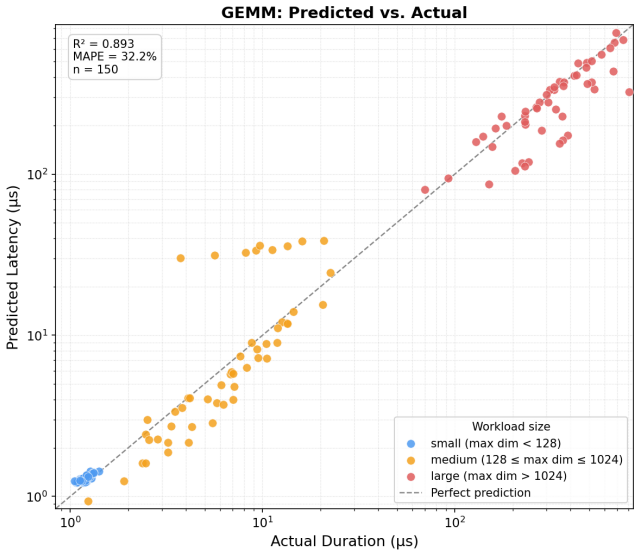


图 4：TPU v4 上 GEMM 延迟的预测值与实际值。每个点代表一种 GEMM 配置，并按工作负载大小（小、中、大）分组。虚线表示完美预测（$y = x$）。虽然 SCALE-Sim TPU 保留了不同大小下的整体缩放趋势（$R^2 = 0.893$），但中等大小工作负载的偏差更为明显，导致整体误差较高（平均绝对百分比误差 = 32.2%）。

5.2 逐元素延迟预测

逐元素运算的学习型延迟模型评估。我们使用具有代表性的逐元素内核（具体而言，即加法和 ReLU（最大值））评估用于非脉动阵列操作的学习型延迟模型。这些运算涵盖了逐元素行为的两大主要类别：纯算术运算，以及带比较的算术运算。对于每种运算，我们都将学习型模型估算出的延迟与不同张量形状下实测的张量处理单元延迟进行比较。

图5展示了逐元素加法和 ReLU 的估算延迟与实测延迟。每个点对应一种张量形状，虚线对角线表示预测完全准确。对于这两种运算，在张量大小的整个范围内，预测延迟都与实测延迟高度吻合。

对于逐元素加法，学习型模型的 R^2 值为 0.9973，绝对误差中位数为 1.04 μs，相对误差中位数为 1.78%。对于会引入额外比较和控制逻辑的 ReLU，模型的 R^2 为 0.9980，绝对误差中位数为 1.65 μs，相对误差中位数为 2.55%。这些结果表明，该模型既能捕捉由大小主导的缩放规律，也能捕捉次要的形状相关影响。

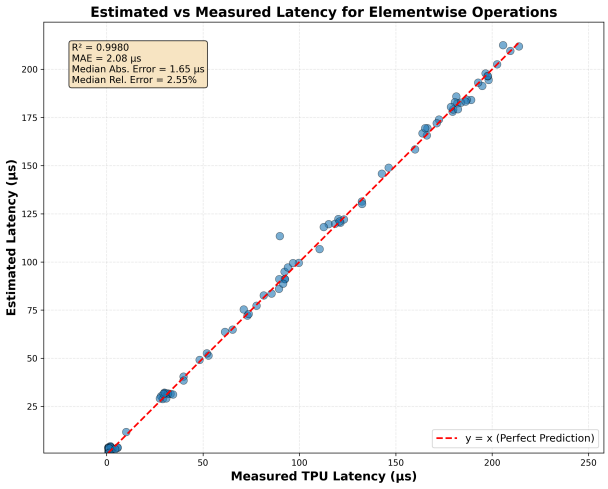
在这两种操作中，较大的绝对误差主要出现在张量大小较小时，此时固定开销在执行时间中占主导地位。尽管如此，在所评估的范围内，相对误差仍然较低，这表明该学习模型能够很好地泛化到训练时未使用的特定形状之外。

6 结论

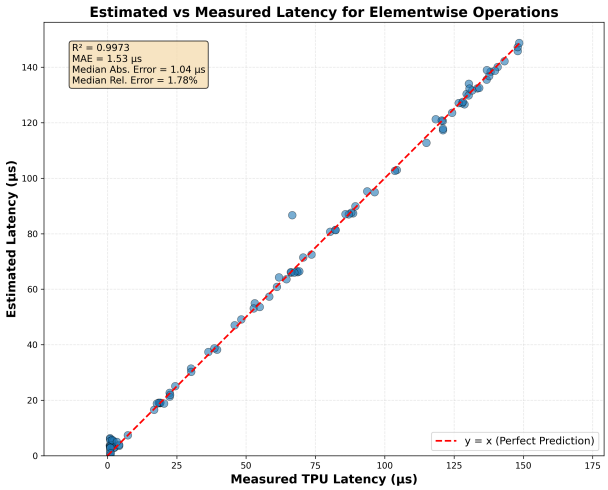
本文验证并扩展了 SCALE-Sim v3，使其能够用于面向 TPU 的性能分析。通过与真实 TPU 测量结果进行比较，我们证明，SCALE-Sim 的周期级 GEMM 预测与硬件执行时间呈现很强的线性关系，因此可以建立简单而有效的周期到延迟的映射。基于这一观察，我们扩展了 SCALE-Sim，使其能够直接报告 TPU 延迟估算，并使用轻量级学习模型对常见的非脉动阵列逐元素操作进行建模。此外，我们还将 StableHLO 集成为前端，使现代机器学习编译器技术栈中的工作负载无需手动指定拓扑即可进行模拟。总之，这些扩展提高了 SCALE-Sim 在 TPU 上进行整模型延迟估算的准确性和易用性。

参考文献

[1] Mansi Choudhary、Chris Kjellqvist、Jiaao Ma 和 Lisa Wu Wills。2025 年。COCOSSim：用于异构脉动阵列架构的周期精确模拟器。载于*2025年IEEE系统与软件性能分析国际研讨会 (ISPASS)*。 174 185 10 1109 64960 2025 00025 – 。[2] 2001 doi: 。/ISPASS 。 。Jerome H. Friedman。 。贪心函数逼近：梯度提升机器。《统计年鉴》29，5（2001），1189–1232。doi:10.1214/aos/ 1013203451[3] Norm Jouppi、George Kurian、Sheng Li、Peter Ma、Rahul Nagarajan、Lifeng Nai、Nishant Patil、Suvinay Subramanian、Andy Swing、Brian Towles、Clifford Young、Xiang Zhou、Zongwei Zhou 和 David A Patterson。2023 年。TPU v4：用于机器学习、支持嵌入的光学可重构超级计算机。载于第 *50* 届年度国际计算机体系结构研讨会论文集（美国佛罗里达州奥兰多）*(ISCA '23)*。计算机协会，美国纽约州纽约市，文章 82, 14 10 1145 3579371 3589350 页码。[4] 2025 doi: 。/ 。Rachid Karami、Sheng-Chun Kao 和 Hyoukjun Kwon。 。通过非 GEMM 工作负载了解最新机器学习工作负载的性能上限。载于



(a) 逐元素加法。



(b) ReLU（最大值）。

图5：非脉动（逐元素）操作的学习得到的延迟模型评估。对于一组多样的张量形状，比较（上）逐元素加法和（下）ReLU（最大值）的估算与实测 TPU 延迟。每个点代表一种张量形状；虚线对角线表示预测完全准确。

2025 IEEE 国际性能分析系统与软件研讨会（*ISPASS*）. 1–14. doi:10.1109/ISPASS64960.2025.00011 访问日期：2026-02-07。[5] 柯国林、孟奇、Thomas Finley、王泰峰、陈伟、马卫东、叶其伟和刘铁岩。2017。LightGBM：一种高效的梯度提升决策树。载于神经信息处理系统进展，Isabelle Guyon、Ulrike von Luxburg、Samy Bengio、Hanna Wallach、Rob Fergus、S. V. N. Vishwanathan 和 Roman Garnett（编），第30卷。Curran Associates, Inc.，3146–3154。https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boost[6] 李卓瑾、Marco Paolieri 和 Leana Golubchik。2024。异构移动设备和机器学习框架上的卷积神经网络推理延迟预测。性能评估 165 2024 102429 10 1016 2024 102429 ( )，。 2024 doi:。/j.peva.。[7] OpenXLA 项目。。StableHLO：稳定的高级操作方言。https://openxla.org/stablehlo。访问日期：2026-02-05。

[8] Angshuman Parashar、Priyanka Raina、Yakun Sophia Shao、Yu-Hsin Chen、Victor A. Ying、Anurag Mukkara、Rangharajan Venkatesan、Brucek Khailany、Stephen W. Keckler 和 Joel Emer。2019 年。TimeLoop：一种系统化的 DNN 加速器评估方法。载于*2019*年*IEEE*系统与软件性能分析国际研讨会 *(ISPASS)*。304–315。doi:10.1109/ISPASS.2019.00042[9] Ritik Raj、Sarbartha Banerjee、Nikhil Chandra、Zishen Wan、Jianming Tong、Ananda Samajdhar 和 Tushar Krishna。2025 年。SCALE-Sim v3：用于端到端系统分析的模块化周期精确脉动阵列加速器模拟器。载于*2025*年*IEEE*系统与软件性能分析国际研讨会 *(ISPASS)*。186–200。doi:10.1109/ISPASS64960.2025.00026