RESPECT：基于强化学习的流水线 Coral Edge TPU 上的边缘调度

JiaqiYin犹他大学
盐湖城，美国
jiaqi.yin
@utah.edu

Yingjie Li犹他大
学盐湖城，美国
yingjie.li
@utah.edu

Daniel Robinson犹他大
学盐湖城，美国
u0714849
@umail.utah.edu

Cunxi Yu犹他大
学盐湖城，美国
c
unxi.yu@utah.edu

arXiv:2304.04716v1 [cs.AR] 10 Apr 2023

摘要—**深度神经网络具有较高的计算和内存需求，其计算图的编译对资源受限（例如受计算、I/O和内存限制）的边缘计算系统的性能有重大影响。虽然高效执行其计算图需要有效的调度算法，但生成最优调度方案是一个具有挑战性的*NP*难问题。此外，考虑到内存通信成本以及不断增大的深度神经网络规模，在流水线多核系统上调度DNN计算图的复杂性将进一步增加。本文使用合成图作为训练数据集，提出了一种基于强化学习（RL）的调度框架RESPECT。该框架学习最优优化算法的行为，并以较短的求解运行时间开销生成近似最优的调度结果。在物理Coral EdgeTPU系统上部署十个热门的ImageNet模型后，我们的框架相较于商业编译器，实测片上推理运行时间最高可获得 $\sim 2.5\times$ 倍加速。此外，与精确优化方法相比，所提出的RL调度将调度优化运行时间最高提升 683×倍（相较于商业编译器），并以最高 930× 倍加速达到精确最优解。最后，我们进行了全面的泛化能力测试，结果表明，RESPECT成功地将从小型合成图到大型真实世界DNN计算图中的最优求解行为进行了模仿。**

I. 引言

为了高效利用硬件平台，深度学习编译器中实现的调度算法对于部署这些维度极高、计算密集型的工作负载至关重要，而这属于经典的*NP-hard* 组合优化（CO）问题。供应商专用库大多依赖手工设计的领域专用启发式方法来优化执行，这种做法以调度求解时间换取执行性能。具体而言，现有调度方法的局限性可概括如下：**(1)** 现有算法要么是优化质量不高的启发式方法，要么是缺乏可扩展性的精确算法/穷举法[5], [12], [21], [24]。大规模深度学习执行面临的挑战仍在增加，尤其是在边缘设备上；**(2)** 手工设计的启发式方法虽然高效，但其开发过程需要在编译和硬件系统方面投入大量工程工作并具备领域知识；**(3)** 近年来出现了基于机器学习（ML）的设计自动化 [13], [22], [23] 和编译框架 [4], [9], [14]。然而，这些框架要么局限于特定的图结构/规模，要么需要在线训练或重新训练 [4], [9]，[14]。更重要的是，现有的机器学习（ML）方法都依赖真实世界数据收集，这既耗时，又限制了其普适性 [11], [13], [15]。

本文提出了 RESPECT，这是一种新颖的组合强化学习框架，通过模仿图组合算法的行为，在边缘设备上调度 DNN计算图。RESPECT旨在边缘计算平台上实现近最优调度；与精确算法或蛮力调度算法相比，其运行时复杂度显著降低。此外，所提出的强化学习模型无需重新训练，即可泛化至同一边缘设备领域内的任意DNN计算图。本文的贡献总结如下：**(1)** 为实现具有更好可扩展性的近最优调度性能，我们提出了一种新颖的基于强化学习的调度框架，该框架模仿现有最优调度算法的算法行为，并采用与数据无关的训练设置，即完全使用合成数据集进行训练；**(2)** 实验在一套物理多级流水线Edge TPU系统上进行， [3], [20] 通过 USB 3.0 通信接口连接。实验结果表明，对于十种常见的 ImageNet 深度神经网络模型，相较于商用 Edge TPU 编译器，该系统显著缩短了 Edge TPU推理运行时间。同时，RESPECT 的调度求解时间最高可加速 $683\times$，性能最高可提升 $2.5\times$；以及**(3)** 最后，我们将所提出的强化学习方法与精确最优调度方法进行了比较。结果显示，所提出的强化学习方法在保持近最优 Edge TPU推理运行时间性能的同时，调度速度最高可提升 $930\times$。我们的研究结果和 Edge TPU部署基础设施已公开发布 [1]。

II. 背景

在现代深度学习框架中，机器学习算法表示为计算图，其中每个图都是一个有向图 $G(V, E)$，节点 V 描述操作，边 E 表示连接算子与输入/输出张量的数据流。实际上，在硬件加速器上部署深度神经网络时，计算图大多表示为有向无环图，其中无环路径会被展开，以最大限度地提升硬件性能。计算图大多通过静态编译生成。

具体而言，调度计算图的优化目标可定义如下：**给定：** *(1)* 一个DAG $G(V, E)$，其中 V 表示DNN计算图中的操作集合，E 表示边集合；*(2)* 一组调度约束，可能包括依赖约束、资源约束、执行时间、内存分配等。**目标：** 构建一个精确的最优调度方案 $S = s_0、s_1、...s_n、 n \leq |V|$，其中 n表示调度阶段的数量。计算图中的操作集合 V 将被分配到满足所有调度约束的 S 。例如，在图1a所示的多级流水线Edge TPU系统中，生成的调度方案将计算节点 N_i 分配到 s_0 (EdgeTPU:0)，将 N_k 分配到 s_1 (Edge TPU:1)，将 N_l 和 N_o 分配到s_2 (Edge TPU:2)，将 N_j、N_p 和 N_q 分配到 s_3 (Edge TPU:3)等。

由于计算单元、内存和通信带宽有限，边缘设备上的推理性能对计算图的调度方案非常敏感。边缘设备上的计算图调度优化问题类似于经典的资源约束调度（RCS）问题，该问题已得到广泛研究。这些研究形成了一系列

[1]https://github.com/Yu-Utah/RESPECT

启发式调度方法（Hu 算法）、列表调度和力导向调度 [12], [19]。人们认为，Google Edge TPU 编译器采用启发式方法来调度计算图，并能在可接受的调度求解时间内完成。尽管启发式方法能够轻松获得解决方案，但调度方案可能与最优方案相差甚远。另一种可行的调度方法是迭代元启发式方法，例如模拟退火、蚁群算法和动态规划优化 [10]。例如， [1]提出了一种基于动态规划的自适应预算调度技术，该技术能够高效地优化调度，同时无需性能保证，也不要求显著增加优化运行时间。

另一方面，RCS 问题可以借助现代形式化方法求解器（如 SAT 和 SMT 求解器），或通过整数线性规划（ILP）等约束求解方法来求得精确最优解：首先将其映射为由线性约束的逻辑连接构成的约束满足问题 [8], [21], [24]。这类方法将专用求解器与命题可满足性搜索技术相结合，以实现冲突驱动学习 [6]。尽管这类精确方法能够得出最优解，但其缺点是调度求解时间较长，因此难以应用于超大规模计算图调度。由此可见，现有 RCS 调度方法在调度求解时间复杂度与调度方案质量之间存在明显的权衡。本研究旨在推动传统调度方法在边缘设备机器学习推理场景中的帕累托前沿，以快速启发式方法的优化成本提供近似最优的调度方案。

III. 方法

图 1a 展示了 RESPECT 框架的概览——步骤 ❶：RESPECT 接收真实世界的深度神经网络模型作为输入，并根据计算节点及节点间的依赖关系将其转换为超图。步骤 ❷：RESPECT 生成计算图的嵌入，其中包含节点属性（拓扑层级、节点 ID）、节点依赖关系（父节点拓扑层级、父节点 ID）以及内存消耗（节点内存）。步骤 ❸：采用通过强化学习（RL）训练的 LSTM 指针网络，根据步骤 ❷ 的嵌入输入队列 q 生成调度序列。请注意，LSTM-PtrNet 模型使用合成数据集进行训练，以减少真实世界数据收集所需的工作量。步骤 ❹：该模型已准备就绪，可连同步骤 ❸ 生成的调度序列一起部署到设备上。此外，还会应用带有领域特定约束的推理后处理，使部署与物理设备兼容。例如，RESPECT 会对 TensorFlow 模型进行量化，并使用 TensorFlow Toco 转换器将子模型部署到真实世界的硬件设备上。

A. 深度神经网络计算图嵌入

出于学习目的，计算图通常会嵌入向量空间中。考虑到在边缘设备上进行调度的性能，计算图的嵌入由图 1a 所示的四个组件组成：**(1)** 基于拓扑排序生成的绝对坐标 。例如，对于节点 N_i，其绝对坐标将在其嵌入的第一列中为 T_i 。请注意，给定有向无环图的拓扑排序有许多变体。在本文中，我们使用尽可能早（ASAP）排序，其中每个节点都尽可能靠近源节点进行排序；**(2)**相对坐标 由父节点的绝对坐标和父节点 ID 组成。例如，对于节点 N_j，其相对坐标将是其父节点的拓扑层级 T_i 和父节点 ID N_i。对于源节点，绝对坐标设置为 0，其 ID 设置为 -1；**(3)** 节点 ID 是通过对所有算子名称进行哈希生成的唯一整数。对于节点 N_i，其节点 ID 将为 N_i；

(4) 缓存和内存局部性是深度神经网络加速中最关键的指标之一。因此，我们在最后一个嵌入列中加入每个算子的内存消耗。具体而言，绝对坐标嵌入表示每个节点的拓扑顺序，模拟给定节点的可行调度空间。相对位置同时使用绝对坐标和节点 ID进行编码。直观而言，相对坐标保留了依赖约束和绝对坐标。例如，在图1a 中， N_p 的数据依赖约束是其父节点的完整调度，即 N_l 和 N_o。与此同时， N_j 和 N_p 可以在可用计算资源上同时调度，因为它们之间不存在数据依赖。此外，节点 N_j 及其父节点 N_i 的绝对坐标分别为 2和1。因此，可以计算出 N_j 的有效调度阶段，即 N_i 和 N_j的绝对坐标之间的任意可用阶段。当 N_i 部署到 s_0时，用于部署 N_j 的可用阶段包括 $\{s_0, s_1, s_2, s_3\}$。

B. 公式化与神经网络架构

如前所述，穷举法和精确方法调度算法可以通过搜索生成最佳调度方案 [5], [8],[24] ，但代价是较高的运行时复杂度，因此无法扩展到大规模问题。尽管启发式算法能够在大规模场景下更高效地优化调度，但其结果质量往往不佳。因此，我们旨在开发一种强化学习框架，模仿任意最优调度算法（例如精确方法或穷举法）的算法行为，使其能够在推理运行时间内以多项式时间完成调度，并获得近似最优的结果。

深度神经网络计算图调度的强化学习公式化– 给定一个计算图有向无环图 $G(V, E)$，强化学习智能体经过训练来制定策略 π，按照目标精确方法或穷举法所采用的相同顺序选择计算节点。在本文中，我们将通过本方法生成的序列顺序定义为 $\pi(i)$ $i \leq |V|$，并将给定确定性精确调度方法确定的序列顺序定义为 $\gamma(i)$ $i \leq |V|$。该精确调度方法可以使用整数线性规划进行公式化，从而针对给定的优化目标提供精确的最优调度。奖励函数被设计为两个序列之间的相似度比较：

$$R = \frac{\sum(\pi(i) \cdot \gamma(i))}{\max(\sqrt{\sum \pi(i)^2} \cdot \sqrt{\sum \gamma(i)^2}, \epsilon)} \tag{1}$$

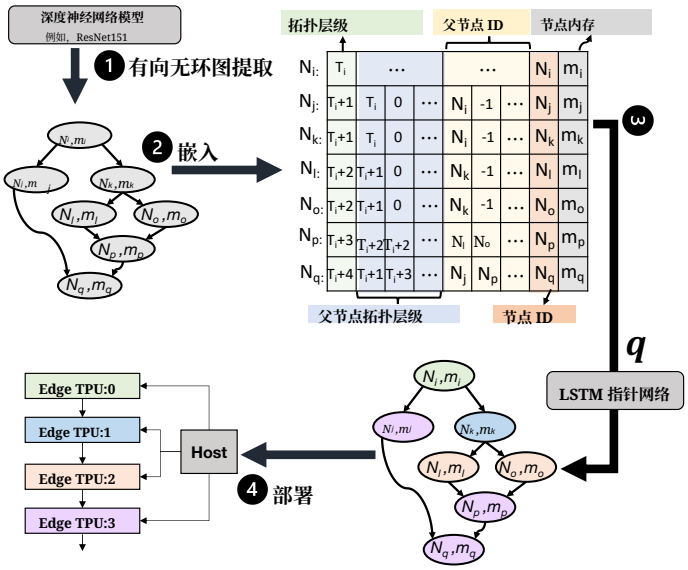
其中 ϵ 是一个较小的常数，用于保证方程的有效性。令 $N_i \rightarrow s_k$, $i \in [1, |V|]$, $k \in [0, n]$ 为在 s_k对节点 N_i 进行调度的解。令 $S' = \{s_0, s_1, ...s_n\}$ 为我们的策略 $\pi(i)$ 生成的序列，其中 n 是 EdgeTPU 阶段的数量，即对于给定计算图，强化学习智能体的输出。由精确调度方法生成的目标调度方案是真实标签序列，记为 S。 S' 和 S 可按如下方式计算：

$$S' = \rho(\pi(i), s_k); \quad S = \rho(\gamma(i), s_k), i \in [1, |V|], k \in [0, n] \tag{2}$$

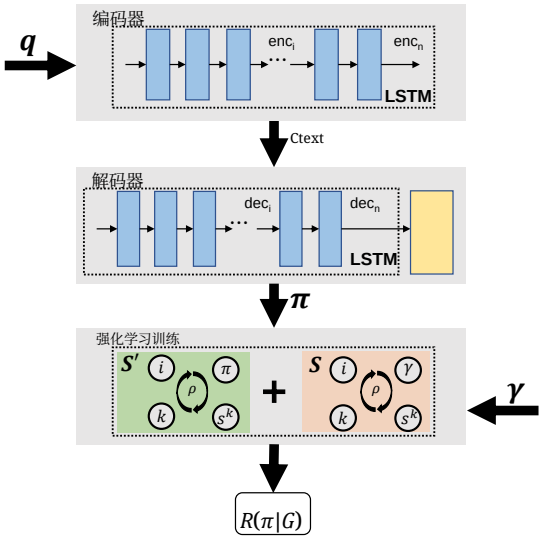
其中， ρ 表示针对特定 Edge TPU 的调度算法。因此，为了模仿目标精确调度方法的行为，奖励通过生成的调度序列 S' 与真实序列 S 之间的余弦相似度来设计（公式 3）。

$$R = \frac{\sum(S_i^k \cdot S'^k_i)}{\max(\sqrt{\sum S_i^{k^2}} \cdot \sqrt{\sum S'^{k^2}_i}, \epsilon)}, i \in [1, |V|], k \in [0, n] \tag{3}$$

在最大化奖励函数 R 的同时，将优化随机策略 $p(\pi|G)$的参数，使其为更接近



(a) RESPECT框架概览 –步骤 ❶：真实世界深度神经网络模型有向无环图提取；步骤 ❷：基于拓扑层级、节点ID 和内存消耗的图嵌入；步骤 ❸：通过基于编码器-解码器的 LSTM 指针网络图调度器进行推理；步骤 ❹：硬件部署。



(b) LSTM-PtrNet模型的详细架构。编码器网络将处理嵌入的输入队列 q，并将其转换为上下文序列 $Ctext$，同时生成潜在记忆状态序列 enc_i。解码器网络将生成预测序列 π，并生成潜在记忆状态 dec_i。强化学习模型训练基于序列 π 和真实标签 γ。

图 1：基于强化学习的调度框架概览。

算法1： 指针机制解码流程

初始化：$d \leftarrow$ 解码器输入；$C = \{Ctext_i\}_{i=1}^{|V|}$；$dec_0$；
$\{emb_i\}_{i=1}^{|V|}$；θ, ω, β(可训练参数);
对于 $i = 1$ 到 n **执行**
  h, $dec_i \leftarrow$Dec(d, dec_{i-1});
  h $\leftarrow$glimpse(C$*\boldsymbol{\theta_g}$, $\boldsymbol{\omega_g}$·h+$\boldsymbol{\beta_g}$);
  $P^i \leftarrow$ 指针(tanh(C$*\boldsymbol{\theta_p}$, $\boldsymbol{\omega_p}$· h + $\boldsymbol{\beta_p}$));
  $idx_i \leftarrow$ argmax $\frac{\exp P^i_i}{\sum_{j=1}^{n} \exp P^i_j}$;
  d $\leftarrow emb_{idx_i}$
end
$S' \leftarrow \rho(\pi(i), s_k)$; S $\leftarrow \rho(\gamma(i), s_k)$, $i \in [1, |V|]$, $k \in [0, n]$;

目标序列的序列顺序分配更高的概率。用于分解序列概率分布的链式法则可表示为：

$$p(\pi|G) = \prod_{i=1}^{|V|} p(\pi(i)|\pi(< i), G) \tag{4}$$

其中，$\pi(< i)$ 表示由前几个周期生成的输出序列。在每个周期，$p(\pi|G)$都会根据输入计算图和之前的序列 $\pi(< i)$ 进行更新。

强化学习智能体架构– 我们扩展指针网络（PtrNet）架构 [2]，将其作为强化学习智能体。PtrNet 擅长寻找能够优化目标的路径，在解决图上的一些组合问题（如旅行商问题 [2]）方面取得了巨大成功。借助注意力机制[16]，，PtrNet 强化了节点间的依赖约束，并克服了使用固定大小图输入学习组合图算法的局限性。请注意，为了全面评估强化学习在物理边缘设备上的调度性能，结果需要满足特定领域的硬件执行要求，例如数据依赖。

强化学习智能体的架构由编码和解码组件构成，如图1b所示。具体而言，每个组件都是一个长短期记忆网络（LSTM）单元。**编码器网络**– 编码器网络处理由节点组成的编码输入队列 q，并将其转换为一系列上下文 $\{Ctext\}_{i=1}^{|V|}$，以表示每个节点的信息，其中 $Ctext_i \in R^d$, d 是 LSTM 的隐藏维度。在解码步骤中，所有上下文的拼接结果将作为已选节点的参考矩阵 C，$C(i) = Ctext_i$, $i \in [1, |V|]$。使用 LSTM 作为编码器时，它会生成一系列潜在记忆状态 $\{enc_i\}_{i=1}^{|V|}$，记录从第一个节点到当前节点传播的编码消息，同时生成相应的上下文，其中最终状态 $enc_{|V|}$ 是解码的初始潜在记忆状态。**解码器网络**– 使用 LSTM 作为解码器网络时，它会在每个解码步骤 i 生成自身的潜在记忆状态 $dec_i \in R^d$，以更新前一个状态。详细的解码过程见算法1。解码器利用来自编码器的上下文参考矩阵 C，通过指针机制生成候选计算节点上的选择概率分布。LSTM 被用作解码器（Dec），它以节点嵌入 d 和最后一个潜在记忆状态 dec_{i-1} 为输入，生成注释 h，并更新潜在记忆状态 dec_i。随后使用包括*glimpse* 和*Ptr-Net* [17] 在内的注意力网络生成序列顺序 π。具体而言，*glimpse* 利用上下文矩阵 C 更新 h，而*Ptr-Net* 则利用编码器（enc_i）和解码器（dec_i）的隐藏状态信息，生成候选项上的概率。通过 softmax 选择下一个节点。一旦选出概率最高的计算节点 N_{idx_i}，其嵌入 emb_{idx_i} 便会在下一个解码步骤中与最后一个潜在记忆状态 dec_{i-1} 的信息一起，作为输入 d 传递给 LSTM。网络会将解答中已出现节点 π 的 logits 设为 $-\infty$，以保证解答的有效性。第一个解码步骤 dec_0 的输入是一个可训练参数，其维度与节点嵌入相同

具有节点嵌入的维度。

强化学习训练– 我们使用基于策略的无模型强化学习训练方法来优化指针网络的参数，该网络在 θ 中表示。学习目标是节点分布的预期相似度。给定输入图 G，优化目标是最大化以下余弦相似度奖励：

$$J(\theta|G) = \mathbb{E}_{\pi\sim p_\theta(\cdot|G)}(1 - R(\pi|G)) \quad (5)$$

在这项工作中，我们采用策略梯度算法和随机梯度下降算法来优化参数 [18]。我们将奖励作为评估所得分布概率的分数，并将其用作梯度计算中的系数，从而将公式 5 的梯度构造如下：

$$\nabla_\theta J(\theta|G) = \mathbb{E}_{\pi\sim p_\theta(\cdot|G)}[(1 - R(\pi|G) - b(G))\nabla_\theta \log p_\theta(\pi|G)] \quad (6)$$

其中， $b(G)$ 表示 LSTM-PtrNet模型中的一个基线，该基线在所有过去的训练迭代中表现最佳，即滚动基线[7], ，并将其应用于损失函数，以降低梯度的方差。请注意，与现有的基于强化学习的调度方法不同，我们的框架会模仿任意调度算法的行为，因此生成的强化学习智能体能够像所模仿的确定性算法一样通用。

合成训练数据集– 用于训练 RL 模型的计算图数量有限，这限制了 RL 在解决基于图的组合优化问题时的泛化能力。此外，为了同时最大限度地减少训练开销，训练数据集需要平衡数据覆盖度和图规模（即 $|V|$ 和 $|E|$ 在 $G(V, E)$ 中）。在强化学习训练期间使用合成数据采样具有一项关键优势：合成图采样器能够全面控制图复杂度以及算子的内存属性，从而比真实世界的深度神经网络计算图提供更好的数据覆盖度。因此，我们仅在训练阶段使用合成图，并将有向无环图采样器集成到强化学习训练框架中，以随机生成具有 $|V| = 30$ 但图复杂度各异的网络图。合成数据集旨在模拟深度神经网络计算图的结构和属性。训练图的复杂度可以随不同的图的度而变化（$deg(V) \in \{2, 3, 4, 5, 6\}$），该度是图 $G(V, E)$ 中连接到顶点 V的入边最大数量。RESPECT 使用 100 万个随机图进行训练，其中每个 $deg(V)$在 {2, 3, 4, 5, 6}中包含 200,000 个图。

推理后处理– 与旅行商问题算法不同，深度神经网络计算图调度需要满足领域特定约束，才能成功将调度后的图部署到硬件平台上。因此，我们在强化学习框架中加入了推理后处理流程，该流程在部署阶段执行，并将推理输出 S' 作为输入。具体而言，针对我们在 Edge TPU平台上的评估，该流程只需将涉及的节点向后移动，即可纠正依赖关系违规；这是一个确定性步骤，对强化学习解决方案的改动最小。此外，Edge TPU硬件要求任一节点的子节点位于同一流水线中，因此推理后处理流程会将这些节点分配到预测阶段中最早的阶段。

IV. 实验结果

在本节中，我们旨在使用真实的 Edge TPU 设备，通过实验验证所提出的基于强化学习的调度方法的有效性。该方法能够以推理运行时间为代价，提供近似最优调度方案。两种对比基线包括商用 Edge TPU 编译器（例如 Google 的 Edge TPU 编译器[2]），它采用启发式方法对计算图进行调度；另一种是基于整数线性规划（ILP）求解器 [8], [24] 的约束求解调度最优方法。我们从三个方面评估 RESPECT：1）片上推理运行时间，2）求解运行时间，以及 3）与精确最优方案之间的优化差距。

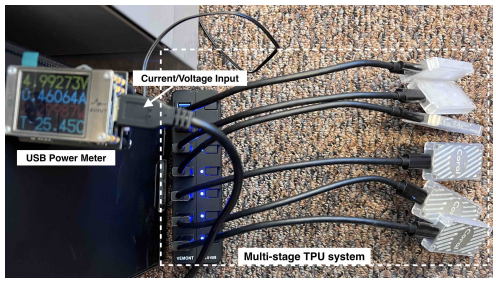


图 2：多阶段流水线 Edge TPU 及通过 USB 3.0 通信接口进行能效评估的系统示意图。

我们的框架将 RESPECT 调度与 Tensorflow-Lite（TFLite）和 Edge TPU 编译器集成，以完成部署流程。它接收单个或多个深度神经网络模型以及流水线阶段数作为输入，并输出 n 个分区后的子图，以部署到 Edge TPU 设备上。该框架包括图构建、通过 RESPECT 求解、解决方案提取和部署。**实验与评估设置** – 我们使用十个 热门 ImageNet 分类模型（表 I）作为基准。所提出的强化学习方法的训练、推理和探索均在一台配备 Intel Xeon Gold 6230 x20 CPU 的 Nvidia 2080 Ti 上进行。对于精确方法，使用 IBM ILOG CPLEX 求解优化整数线性规划公式。启发式方法的调度解决方案取自 Google 的 Edge TPU 编译器[1]。RESPECT 基于 PyTorch 实现。神经网络架构的编码器和解码器由包含 256 个单元的长短期记忆网络构建。训练使用 Adam 优化器进行 300 个 epoch，学习率为 10^{-4}，批量大小为 128。此外，我们搭建了一个集中式 EdgeTPU 系统，用于评估实验结果，如图 2 所示。

表 I：用于评估流水线 Edge TPU 系统推理运行时间的深度神经网络模型及其计算图统计信息。

	Xception	ResNet50	ResNet101	ResNet152
$\|\mathbf{V}\|$	134	177	347	517
度(V)	2	2	2	2
深度	125	168	338	508
	DenseNet121	ResNet101v2	ResNet152v2	DenseNet169
$\|\mathbf{V}\|$	429	379	566	597
度（V）	2	2	2	2
深度	428	371	558	596
	DenseNet201	InceptionResNetv2		
$\|\mathbf{V}\|$	709	782		
度（伏）	2	4		
深度	708	571		

*A. Edge TPU*推理运行时评估

我们将 RESPECT 的调度结果与基于 Edge TPU 编译器的启发式方法以及基于整数线性规划求解器的精确方法进行比较。图 4 中的结果是对 1,000 次 ImageNet 推理进行 10 轮测试所得的平均运行时间。纵轴表示推理运行时间结果，其值已根据使用 Edge TPU 编译器所得的运行时间进行了归一化。

如图 4所示，所提出的强化学习调度器始终优于商用Edge TPU 编译器。具体而言，相较于 Edge，强化学习调度器可实现 1.06×、1.08×、1.65× 倍加速。

[2]https://coral.ai/docs/edgetpu/compiler/

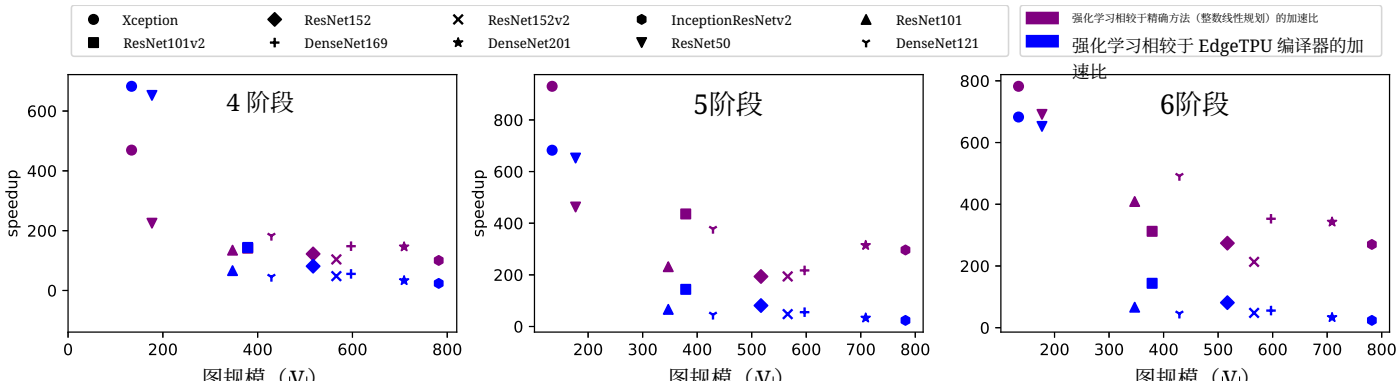


图 3：十个真实 ImageNet 模型在不同图规模下的调度求解时间。横轴和纵轴分别表示图规模 $|V|$ 和相对于基线的加速比。强化学习方法相较于 Edge TPU 编译器的加速比为 $24-683\times$，相较于精确方法的加速比为 $100-930\times$。

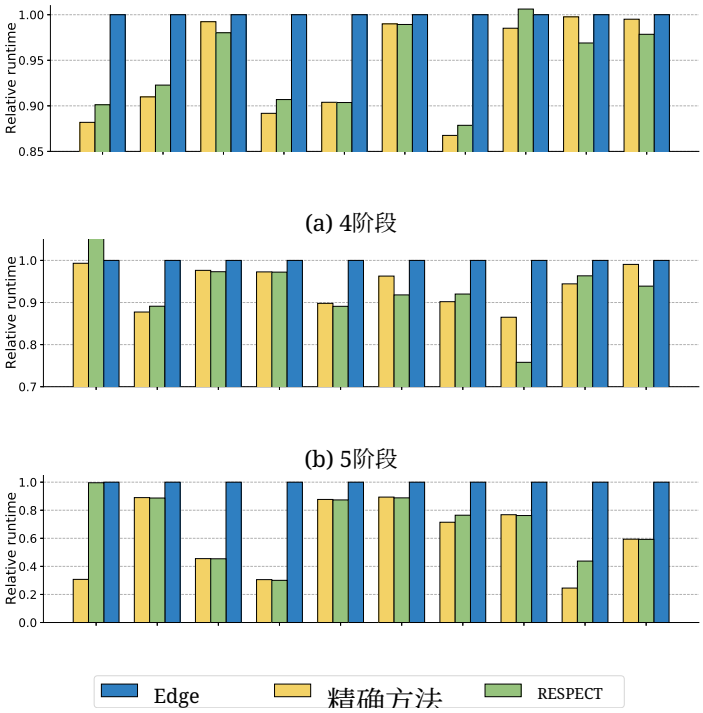


图 4：所提出的强化学习方法、精确方法与商业 Edge TPU 编译器之间的多阶段流水线式 Edge TPU 运行时间比较（基准规模=1）。强化学习方法提供了近似最优的调度分配。在 4、5 和 6 阶段流水线式 Edge TPU 系统中，运行时间性能均持续优于商业编译器（例如，ResNet101v2 和 ResNet152 的执行 $2.5\times$ 速度快于 Edge TPU 编译器）。

TPU 编译器在 4、5 和 6 级流水线选项下。例如，在 6 级流水线下，ResNet101v2 和 ResNet152 的执行速度比 Edge TPU 编译器快 2.5× 倍。其次，与通过整数线性规划求解的精确最优调度方法生成的调度方案相比，所提出的强化学习方法展现出近乎最优的 Edge TPU 推理性能，同时显著缩短了调度运行时间。有关求解时间复杂度的详细讨论见第 IV-B 节。第三，平均 Edge TPU 推理运行时间随着流水线阶段数量的增加，所提出的强化学习方法相较于 EdgeTPU编译器的改进也随之提升。例如，我们提出的强化学习方法在 4 阶段系统中为 ResNet152 提供了 $1.03\times$ 的加速效果，并且在 6 阶段系统中，推理运行时间的加速效果提升至 $2.28\times$。其主要原因是，随着阶段数量的增加，调度优化的复杂度也随之增加，而启发式方法的性能将进一步下降。

性能建模失相关 – 从图 4 所示的运行时间性能分析中，我们观察到，精确方法有时表现不如 RESPECT。这是由于计算图的高层抽象建模与实际脉动硬件规格之间存在失相关，而硬件架构和后端编译均为闭源。高层性能建模中的此类失相关可用于指导调度优化，但会给实际片上性能引入优化噪声。具体而言，在本工作中，精确方法和 RESPECT 都从内存分配和通信成本两个方面对深度神经网络模型调度进行优化。需要注意的是，鉴于 EdgeTPU 计算组件和缓存／内存系统的复杂性，无法直接根据计算图及其调度对运行时间进行建模或估算 [3]。

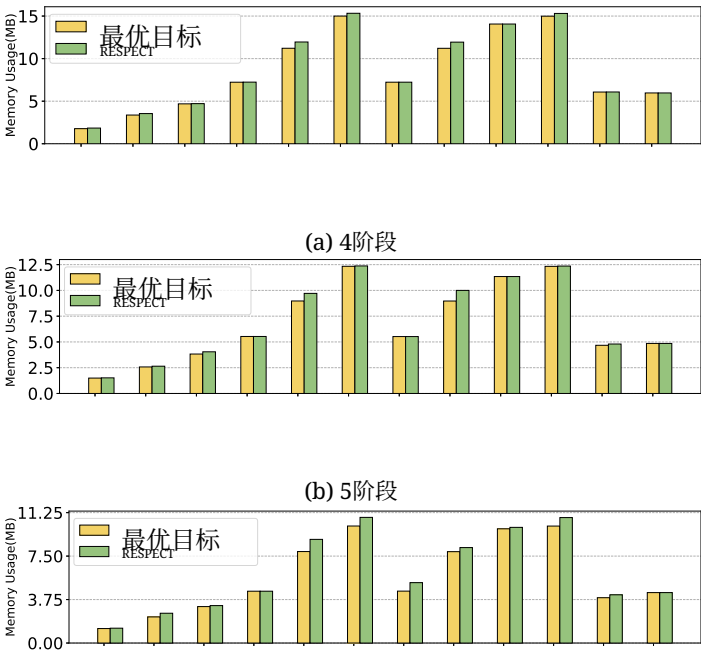
B. 求解可扩展性评估

在本节中，我们进一步将所提出的强化学习框架的调度求解运行时间，与相同一组基线方法在相同一组深度神经网络模型上的运行时间进行比较。结果如图 3 所示，横轴表示图规模，即计算图 $G(V,E)$ 中的 $|V|$。纵轴表示强化学习框架相较于两种基线方法在求解运行时间上的加速比。

从图 3 可以总结出三个重要结论：**(1) 与启发式方法（Edge TPU 编译器）相比，所提出的强化学习方法的主要优势在于提升了 Edge TPU推理运行时中的调度质量**。具体而言，在生成调度方案时，我们的强化学习方法相比 Edge TPU 编译器可实现 $24\times$ 至 $683\times$ 倍的加速，并带来最高 $2.5\times$ 的推理运行时间提升，详见第 IV-A 节。例如，对于 ResNet152，强化学习生成的调度方案可使 Edge TPU推理运行时加速 $2.4\times$ 倍，同时将调度求解时间缩短 $85\times$。**(2) 与基于约束求解调度的精确方法相比，我们的强化学习方法在生成调度方案时可实现 $100\times$至 $930\times$ 倍的加速，而 Edge TPU推理运行时几乎相同**。例如，对于

对于 Xception 模型，强化学习方法和精确方法的 Edge TPU推理运行时几乎相同，而强化学习方法的调度求解时间比精确方法快 930× 。

C. 与最优解的差距分析



(a) 4阶段

(b) 5阶段

(c) 6阶段

图 5：RESPECT 与精确调度之间的差距分析。(a)–(c)：ImageNet 深度神经网络模型在 4、5、6 阶段流水线配置下的参数缓存结果。

最后，我们进行泛化能力分析，以展示 RESPECT 框架生成近似最优调度方案的能力。我们通过评估以下指标，验证强化学习框架的泛化能力：RESPECT 推理结果相对于精确最优解的结果质量，即 与最优解的差距 。具体而言，我们收集各流水线阶段的参数缓存值，包括缓存内和缓存外的内存缓存值，并计算精确最优调度方案与 RESPECT 调度方案在各阶段峰值内存使用量上的绝对差异。实验基于 ImageNet 深度神经网络模型进行，结果如图 5a 至 5c 所示。可以看出，与精确最优解相比，RESPECT 在参数缓存方面取得了近似最优的结果，在所有 4、5、6 阶段流水线配置下，平均性能差距分别仅为 2.26%、2.74% 和 6.31%。

V. 结论

这项工作提出了一种基于强化学习的调度框架，该框架模仿最优优化算法的行为，同时显著降低运行时复杂度。我们的框架在三种不同的实体流水线 GoogleEdge TPU系统上，使用十个流行的 ImageNet 模型进行测试，相较于商用 Edge TPU编译器，运行时间最高加速达 2.5× 倍。更重要的是，与通过启发式方法和穷举法求解的精确优化方法相比，所提出的强化学习调度将调度求解时间缩短了几个数量级。

致谢 –本工作得到美国国家科学基金会（NSF）在 NSF-2047176、NSF-2019336、NSF-2008144 和 NSF-2229562 项目资助下的支持。

参考文献

[1] B. H. Ahn、J. Lee、J. M. Lin、H.-P. Cheng、J.Hou和H.Esmaeilzadeh，“排序混乱：面向边缘设备中非规则连接神经网络的内存感知调度”，机器学习与系统会议论文集，第 2 卷，第 44–57页，2020 年。[2] I. Bello、H. Pham、Q. V. Le、M. Norouzi和 S.Bengio，“利用强化学习进行神经组合优化”，arXiv预印本 arXiv:1611.09940，2016 年。 [3] A. Boroumand、S.Ghose、B. Akin、R. Narayanaswami、G. F. Oliveira、X.Ma、E. Shiu 和 O. Mutlu，“缓解边缘机器学习推理瓶颈：关于加速 Google Edge 模型的实证研究”，arXiv预印本 arXiv:2103.00768，2021 年。[4] H. Chen和M.Shen，“基于深度强化学习的 FPGA高层次综合调度器”，载于 2019 IEEE/ACM 国际计算机辅助设计会议（ICCAD）。IEEE，2019 年，第1–8 页。 [5] J.Cong 和 Z. Zhang，“基于 SDC表述的高效且通用的调度算法”，设计自动化会议（DAC），2006 年。 [6] L. De Moura和 N. Bjørner，“模理论可满足性：介绍与应用”，ACM 通讯，2011 年。 [7] W. Kool、H. Van Hoof 和 M. Welling，“注意力机制：学习解决路径规划问题！”arXiv预印本 arXiv:1803.08475，2018 年。 [8] C.E.Leiserson 和 J. B.Saxe，“同步电路的重定时”，Algorithmica，第6 卷，第1期，第 5–35 页，1991 年。 [9] H. Mao、M. Schwarzkopf、S. B.Venkatakrishnan、Z.Meng 和M. Alizadeh，“学习数据处理集群的调度算法”，载于 ACM 数据通信特别兴趣小组会议论文集，2019 年，第 270–288 页。 [10] G. D. Micheli，《数字电路的综合与优化》。McGraw-Hill 高等教育，1994年。 [11] D. Pal、C. Deng、E. Ustun、C. Yu和 Z.Zhang，“用于敏捷 FPGA 设计的机器学习”，《机器学习在电子设计自动化中的应用》，第 471–504页，2022 年。 [12] P.G. Paulin和 J. P. Knight，“用于专用集成电路行为级综合的力导向调度”，IEEE 集成电路与系统计算机辅助设计汇刊，第8 卷，第 6 期，第 661–679 页，1989 年。 [13] H.Ren和 J. Hu，《机器学习在电子设计自动化中的应用》。施普林格·自然，2023 年。[14] S.Sheng、P. Chen、Z.Chen、L. Wu和 Y. Yao，“物联网边缘计算中的基于深度强化学习的任务调度”，Sensors，第 21卷，第 5 期，第 1666 页，2021 年。[15] E.Ustun、S. Xiang、J. Gui、C.Yu 和 Z. Zhang，“LAMDA：用于 FPGA 设计收敛的学习辅助多阶段自动调优”，载于2019 IEEE 第 27 届年度现场可编程定制计算机国际研讨会（FCCM）。IEEE，2019 年，第 74–77页。 [16] A. Vaswani、N.Shazeer、N. Parmar、J.Uszkoreit、L.Jones、A. N. Gomez、Ł.Kaiser 和I.Polosukhin，“注意力机制就是你所需要的一切”，载于《神经信息处理系统进展》，2017 年，第 5998–6008 页。 [17] O. Vinyals、M.Fortunato和 N. Jaitly，“指针网络”，载于《神经信息处理系统进展》，2015 年，第 2692–2700 页。[18] R.J. Williams，“用于连接主义强化学习的简单统计梯度跟随算法”，《机器学习》，第8 卷，第 3 期，第 229–256 页，1992 年。 [19] T. Yang 和A. Gerasoulis，“带通信延迟和不带通信延迟的列表调度”，Parallel Computing，1993 年。[20] A. Yazdanbakhsh、K. Seshadri、B. Akin、J. Laudon 和 R. Narayanaswami，“卷积神经网络EdgeTPU 加速器评估”，arXiv 预印本 arXiv:2102.10423，2021 年。[21] J. Yin、Z.Zhang和C. Yu，“流水线式 EdgeTPU 上深度神经网络的精确内存与通信感知调度”，载于 2022IEEE/ACM第 7届边缘计算研讨会（SEC）。IEEE，2022 年，第 203–215 页。 [22] C.Yu、H. Xiao 和 G.DeMicheli，“在没有人类知识的情况下开发综合流程”，载于第55届年度设计自动化会议论文集，2018 年，第 1–6页。[23] C. Yu 和 Z. Zhang，“ 布局上的绘画：使用条件生成对抗网络预测布线拥塞”，载于第56 届年度设计自动化会议论文集 2019，2019 年，第1–6 页。[24] Z.Zhang 和 B. Liu，“基于 SDC 的模调度用于流水线综合”，计算机辅助设计国际会议（ICCAD），2013 年。