MIGPERF：面向多实例 GPU 深度学习训练与推理工作负载的综合基准测试

Huaizheng Zhang [1] **Yuanming Li** [1] **Wencong Xiao** [1] **Yizheng Huang** [2] **Xing Di** [3] **Jianxiong Yin** [4] **Simon See** [4] **Yong Luo** [3] **Chiew Tong Lau** [5] **Yang You** [6]

arXiv:2301.00407v1 [cs.LG] 1 Jan 2023

摘要

如今，A100 等新型架构的图形处理器已搭载多实例 GPU（MIG）技术，可将 GPU 划分为多个彼此隔离的小型实例。这项技术为用户提供了更大的灵活性，以支持深度学习训练和推理工作负载，但要高效利用它仍然颇具挑战。本文旨在为 MIG 提供更全面、实用的基准测试研究，从而免去繁琐的手动基准测试与调优工作。为实现这一目标，本文推出了 MIGPerf，这是一款可简化 MIG 基准测试研究流程的开源工具。借助 MIGPerf，作者开展了一系列实验，包括 MIG 上的深度学习训练与推理特征分析、GPU 共享特征分析，以及 MIG 的框架兼容性研究。实验结果为用户有效使用 MIG 提供了新的见解与指导，并为进一步研究 MIG 上混合训练与推理工作负载的编排奠定了基础。代码和结果已发布于 https://github.com/MLSysOps/MIGProfiler。这项工作仍在进行中，更多结果将于近期发布。

1 引言

深度学习（DL）模型已广泛应用于众多行业，从电子商务（Liu et al., 2016; Wu et al., 2019; Zhang et al., 2020b）和视频分析（Liu et al., 2020; Oprea et al., 2020; Zhang et al., 2020a; Jiao et al., 2021），到自动驾驶（Grigorescu et al., 2020）和金融（Heaton et al., 2017; Ozbayoglu et al., 2020）。为了高效支持这些计算密集型模型，用户必须使用图形处理器等新型硬件设备。在这些 GPU 中，配备多实例 GPU（MIG）技术（NVIDIA, 2022a; Choquette et al., 2021）的 A100 和 A30 等新近发布的 GPU 引起了广泛关注。借助 MIG，像 A100 这样的整块 GPU 可划分为多个相互隔离的小型 GPU 实例（GI），从而更灵活地支持深度学习训练和推理工作负载。例如，用户可以将一块 A100 划分为七个 GPU 实例（1/7 GI），以运行 ResNet50 （ He et al., 2016 ）等小型模型。这样一来，一块 A100 就能并行支持七个 ResNet50 模型，同时仍满足服务级别目标（SLO）。此外，在训练 RoBERTa（Liu et al., 2019）等大语言模型时，用户可以使用未划分的整块 GPU（7/7 GI）。此外，用户还可以配置三个分别为 4/7、2/7 和 1/7 的 GPU 实例，并且每同时运行训练（在 4/7 GPU 实例上）和推理（在 2/7 和 1/7 GPU 实例上）工作负载。这种灵活性带来了提高 GPU 利用率、节省成本的新机会。

尽管这种灵活性带来了提高 GPU 利用率、节省成本的新机会，但也带来了挑战。首先，NVIDIA 通过设置硬编码规则来限制分区（Tan 等人，2021）。例如，对于 A100，用户无法同时使用 4/7 和 3/7 的 GPU 实例。因此，用户无法像划分 CPU 或磁盘那样自由地划分 GPU。其次，由于每个划分出的 GPU 实例彼此隔离，且拥有各自的容量（例如 L2 缓存、内存等），要了解不同深度学习工作负载在 GPU 实例上的性能，需要进行大量手动基准测试。最后，上述两个问题导致目前用于分配 CPU 和内存的调度方法难以直接应用于配备新型 MIG GPU 的集群。

这些问题促使越来越多的研究开始探讨不同工作负载下的 MIG 性能。两项具有代表性的研究在更好地利用 MIG GPU 方面迈出了重要一步。第一项研究（Kaas 等人，2022）聚焦于深度学习训练。该研究采用了三种图像识别模型：ResNet-26（小型）、ResNet-50（中型）和 ResNet-152（大型），以探索 MIG GPU 在不同模型大小下执行训练工作负载的表现。第二项研究（Tan 等人，2021）为 MIG GPU 设计了推理服务。该研究首先对 PyTorch 和 TensorFlow Hub 中的数十种模型（PyTorch，2022；TensorFlow，2022）在不同批次大小和 MIG 分区下进行基准测试。随后，研究设计了一个包含优化器和控制器的服务系统，以

[1]阿里巴巴集团 [2]新加坡科技研究局资讯通信研究院 [3]武汉大学 [4]NVIDIA AI 技术中心 [5]南洋理工大学 [6]新加坡国立大学。通讯请联系：Yuanming Li <yuanming.li@alibaba-inc.com>。

初步成果。正在评审中。请勿传播。

更具成本效益地提供深度学习推理工作负载（节省 40% 的 GPU 成本）。

尽管这些研究为如何利用 MIG 提供了启示，但我们仍缺乏**全面且实用的**MIG 基准测试研究。具体而言，这一局限体现在几个方面。首先，先前的研究仅关注单一工作负载，即训练或推理，因此研究范围有限。因此，我们需要针对训练和推理工作负载开展全面的基准测试研究。其次，许多不具备 MIG 功能的高性能 GPU 仍在集群中占主导地位。这些 GPU 只能采用基于软件的共享方法，而现实场景中 MIG 与基于软件的 GPU 共享之间的比较仍然缺失。第三，当前训练和服务框架与 MIG 的兼容性尚不明确。了解这种兼容性有助于为现有框架添加新功能，也有利于 MIG 优化，因此应将其视为优先事项。最后，现有研究缺少许多重要指标，例如能耗，而这对绿色人工智能至关重要（Schwartz 等人，2020）。

为了解决这些问题并推动该领域的研究，我们提出从多个角度探索 MIG，以提供全面且实用的基准测试报告。首先，我们使用多种真实场景中的训练和推理工作负载对 MIG 进行实践评估，并修复基准测试过程中发现的若干潜在问题。其次，我们在多种设置下将 MIG 与基于软件的 GPU 共享方法进行比较，以了解各自适用的场景。第三，我们还探讨了深度学习框架与 MIG 的兼容性。

此外，我们实现了一款名为 MIGPerf 的开源工具，旨在简化 MIG 基准测试流程。该工具首先从在 MIG GPU 上对训练和推理工作负载进行基准测试的过程中抽象出通用工作流，帮助用户通过几个简单步骤启动基准测试。其次，它提供了采用高度模块化后端的应用程序接口，便于扩展。具体而言，MIGPerf 包含一个控制器，用于将 MIG GPU 划分为不同的 GPU 实例；还包含一个 MIG 分析器，用于生成基准测试工作负载并收集结果。结果会经过格式化，以便 Jupyter 和 Prometheus 等第三方工具快速使用。此外，该工具基于 Python 实现，并集成了 DCGM 等多种工具，以便在云端更轻松地监控性能。

我们使用 MIGPerf 开展了多项基准测试研究
在两种 MIG GPU（A100 和 A30）上开展初步研究。流畅的基准测试过程证明了其有效性。同时，我们展示了许多新的基准测试结果，涵盖序列长度的影响、能耗、GPU 共享对比以及框架影响等方面，以揭示更多见解。此外，这些初步研究还展现了若干极具前景的研究方向。例如，我们可以在 MIG GPU 上设计一种混合深度学习训练与推理工作负载编排方案。此外，如何

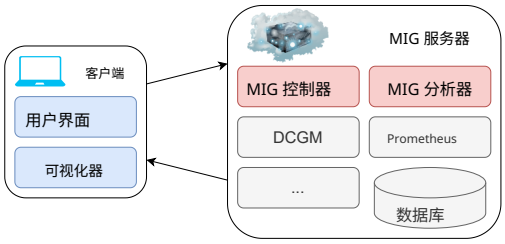


图 *1.* 所提出的基准测试系统概览。系统首先接收用户的基准测试任务，然后将任务分配给专用服务器自动完成。最后，系统会向用户发送详细报告和指南。

将 MIG 集成到当前的服务框架中以节省推理成本，仍是一个尚待解决的问题。

本文其余部分安排如下：首先在第 2 节介绍相关工作。接着在第 3 节介绍系统实现及所采用的方法。在第 4 节使用我们的系统执行基准测试任务并评估其性能。最后，在第5 节总结本文并指出未来的研究方向。

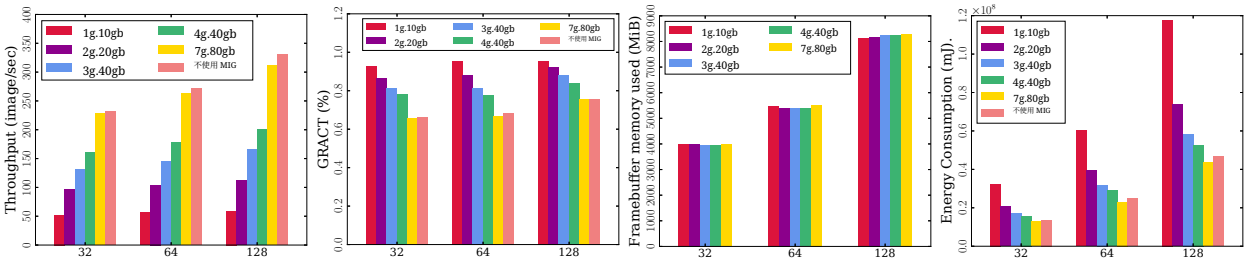
2 相关工作

我们将相关工作分为两类：深度学习（DL）基准测试和 GPU 共享。本节将简要介绍这两类工作。

2.1 深度学习基准测试

基准测试工具在推动深度学习的发展方面发挥着至关重要的作用。这些工具可分为两类：宏基准测试和微基准测试。DawnBench（Coleman 等人，2017）、Fathom（Adolf 等人，2016）、AI Benchmark（Ignatov 等人，2019）和 MLPerf Inference（Reddi 等人，2020）等宏基准测试针对的是高层模型或硬件评估。这些工具收集多种模型，并在不同硬件上运行，以通过延迟、吞吐量等多项指标评估其性能。微基准测试则探索底层组件，这些组件决定了模型或硬件的运行速度。AI Matrix（阿里巴巴，2018）和 DeepBench（百度，2017）等代表性研究探讨了层和计算内核对最终模型性能的影响。

近期有两项关于 MIG 的研究可归类为微基准测试。其中一项使用三个 ResNet 模型探索 MIG 上的训练性能，另一项则研究 MIG 上的推理性能，以设计更好的推理调度器。相比之下，我们的工作提供了一款开源工具，让用户能够轻松探索 MIG 上的训练和推理性能。同时，我们还考虑了更多指标、框架（例如 Triton (NVIDIA, 2022b)）以及新模



(a) 吞吐量（批次/秒） (b) GRACT（%） (c) 帧缓冲内存（MiB） (d) 能耗（毫焦耳）

图 *2*。输入批次大小对 A100 单个 GPU 实例（GI）上训练的影响。我们展示了批次大小对吞吐量、计算利用率、内存利用率和能耗的影响。

型，以便对 MIG 进行全面的性能评估。此外，我们提供了一个公开排行榜，持续更新近期关于 MIG 的基准测试研究。总的来说，我们的系统是对现有基准测试工具的补充。

2.2 GPU 共享

为提高深度学习工作负载的图形处理器利用率，近期研究投入了大量精力。NVIDIA 推出的多进程服务（MPS）（NVIDIA，2020）技术旨在避免同一图形处理器上多个工作负载造成的高昂上下文切换开销。因此，这些工作负载可以在共享图形处理器上实现相对稳定的运行速度，同时满足服务等级目标（SLO）。然而，由于 MPS 无法提供真正的隔离，尤其是面对大型工作负载时，干扰无法避免。为缓解这一问题，Gandiva（Xiao 等人，2018）提出了一种在共享与非共享模式之间切换的回退方法。此外，Salus（Yu 和 Chowdhury，2019）尝试基于经性能分析的深度学习工作负载的内存使用量，构建一种细粒度图形处理器共享的基础机制。此外，许多公司开发了虚拟图形处理器（vGPU）（NVIDIA，2022c）技术，使多个云端深度学习工作负载能够按需共享这些虚拟图形处理器。与这些技术相比，MIG 是 NVIDIA 提供的首项能够实现可配置物理隔离的技术。然而，我们仍然缺乏一项全面且实用的研究来比较这些技术。本文提供了一些初步结果，我们将继续对此进行探索。

3 系统设计与实现

本节首先从用户视角介绍系统工作流程，然后介绍几个核心模块实现。

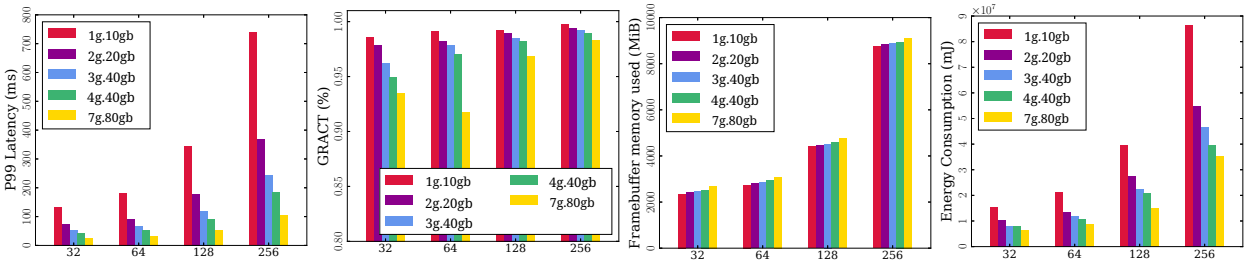
3.1 系统概述

首先，应在配备 MIG GPU 的服务器上安装并启动 MIGPerf。然后，用户可以调用 MIG 控制器的 API，根据基准测试需求将 GPU 划分为不同的 GPU 实例（GPU 实例）。接下来，用户可以通过指定工作负载类型（例如训练）、模型名称以及其他设置（例如批次大小）来调用 MIG 分析器，执行基准测试流程。在基准测试期间，结果将以时间序列的方式保存到本地文件中。同时，保存的结果还将导出为不同格式，以便 Prometheus 等其他部署完善的工具直接使用。最后，用户可以使用所提供的可视化工具快速分析结果并获取洞见。此外，用户还可以在自己的笔记本电脑上安装 MIGPerf 客户端，以远程控制整个流程并在本地进行分析。

3.2 核心组件

MIG 控制器 帮助用户以更易于理解的方式管理分区 GI。它提供 Python API，以便：1）在图形处理器上启用 MIG；2）执行分区操作；3）跟踪 GPU 实例。对于每个 GPU 实例，控制器还可以进一步创建或销毁计算实例（CI）。这样一来，用户便拥有更大的灵活性。例如，在同一 GPU 实例中运行的作业可以隔离计算资源，同时共享内存资源。该模块基于 NVIDIA 的 MIG 命令构建。

MIG 分析器抽象出通用的深度学习（DL）训练与推理工作负载，并通过多种指标监控其运行性能（详情见 A）。它包含两部分：工作负载执行器和性能聚合器。工作负载执行器模拟训练和推理进程，并根据用户指定的设置执行基准测试。它基于 HuggingFace 等多种广泛使用的模型开发框架实现，因此用户可以轻松扩展该模块。性能聚合器监控工作负载性能和系统资源使用情况，并将其保存到数据库中。



(a) P99 延迟（毫秒） (b) GRACT (%) (c) 帧缓冲内存 (MiB) (d) 能耗 (mJ)

图 *3.* 其影响 推理输入序列长度对单个 GPU 实例（GPU 实例）来自 A100。

它基于 DCGM 等工具开发。

其他组件包括结果导出器、可视化工具和客户端。它们的设计目的都是提升用户体验。例如，导出器可以对已保存的性能结果进行格式化，以便使用不同的性能分析工具展示这些结果。客户端可以安装在用户的笔记本电脑上，用于远程控制整个流程。

4 基准测试研究

本节将若干具有代表性的基准测试案例作为初步研究进行介绍。我们首先简要介绍评估设置，然后详细说明评估指标。最后，我们讨论训练和推理工作负载的实验结果。详细设置和更多结果请参见附录。

4.1 评估设置

硬件。我们分别在两种支持 MIG 的 GPU 上进行实验：A100 和 A30。这两种 GPU 分别安装在两台服务器上，服务器的详细信息请参见附录A。

模型。我们从 Hugging Face Transformers、TorchHub 等知名公共模型库中选取候选基准模型。从这些模型库中，我们选择了涵盖多个领域的模型，包括图像分类（例如视觉变换器）、语言建模（例如多语言 BERT）、图像生成（例如扩散）等。更多模型详情请参见附录 A。

4.2 评估指标

延迟 衡量处理速度。对于推理工作负载，我们使用两个指标：平均延迟和尾延迟。平均延迟衡量一个请求批次的平均处理时间，而尾延迟表示 X（例如 99）百分位数的请求，其延迟低于给定值。

吞吐量 衡量设备在单位时间内（例如 1 分钟）能够处理的训练样本或推理请求数量。

图形引擎活动 (GRACT) 用于衡量图形处理器的利用率，即已分配计算资源与可用资源总量的比值。此外，也可以使用 NVIDIA 提供的计算公式来衡量每个分区 GPU 实例的利用率；该公式已在我们的工具中实现。

帧缓冲区 (FB) 用于衡量运行中工作负载占用的 GPU 显存。虽然可以通过将模型和输入数据大小相加来计算，但我们提供了 FB 的结果，以免用户手动计算。

能耗 是对在特定时间段（例如 5 分钟）内运行工作负载所消耗电量的估算。我们提供这些结果，帮助用户了解 MIG 是否能够节省电费。

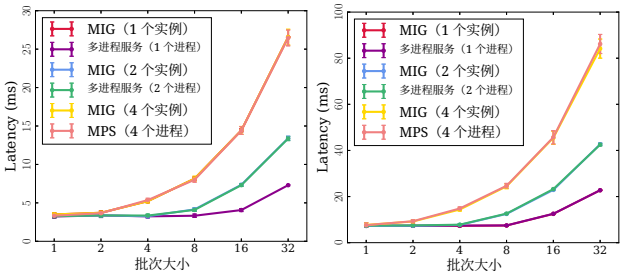
4.3 MIG 训练特征分析

第一项实验探究了不同 MIG 分区大小和批量大小下的 MIG 训练性能。我们展示了 BERT 模型的结果，如图 2(更多结果请参见附录B。

图 2a 显示，对于 1g.10gb 这样的小型实例，增加批次大小并不能提高吞吐量。我们认为这是因为 GPU 实例的计算资源已被充分利用。由于批次大小 = 32 已经占用了全部计算资源，增加批次大小会降低处理速度，因此吞吐量不会提高。图 2b也证实了这一点。小型实例的利用率很高且稳定，而大型实例的利用率较低。

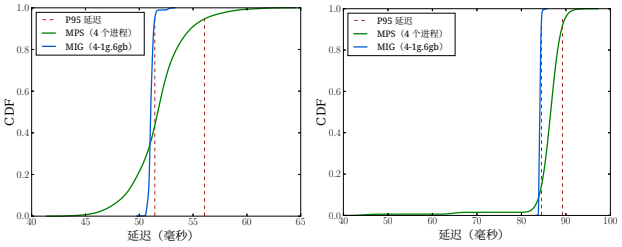
图 2c展示了内存利用率。当批次大小固定时，不同大小的 GPU 实例之间内存使用量没有差异。我们认为这是因为内存容量较大。即使是最小的 GPU 实例，也能轻松运行 BERT。

图 2d 展示了能耗情况。这并不令人意外，



（a）ResNet18 平均延迟　(b) ResNet50 平均延迟

图 *4.* MPS 和 MIG 上不同模型的平均延迟对比。我们展示了不同批量大小下 ReNet18 和 ResNet-50 的结果。结果表明，当批量大小较小时，MPS 的性能与 MIG 非常接近。



(a) ResNet18 尾延迟　(b) ResNet50 尾延迟

图*5*。MPS 和 MIG 上不同模型的尾延迟对比。我们设置批次大小 = 8，并评估 ResNet18 和 ResNet50 两种模型。结果表明，MIG 优于
在较大批次大小下，MPS 在尾延迟和稳定性方面都差得多。

小批次大小消耗的能量更少。令人惊讶的是，在批次大小相同的情况下，实例越大，能耗越低。我们推测，这是因为较大的 GPU 实例处理速度更快。由于我们发送的请求数量固定，较大的 GPU 实例能够更快地完成处理，因此能耗更低。

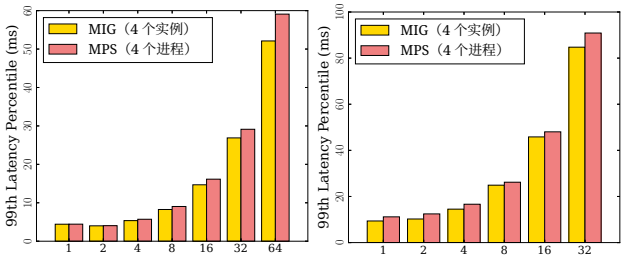
4.4 MIG 推理特征分析

本节介绍 MIG 上的推理基准测试。我们以 BERT 为例，更多结果见附录B。图 3a 展示了延迟结果。当 GPU 实例分区较小时，批次大小对延迟的影响很大；而对于较大的 GPU 实例，这种影响则微乎其微。这可归因于大型 GPU 实例拥有过多的计算资源。图3b展示了图形处理器利用率。对于所有批次大小，利用率都保持在较高水平。随着 GPU 实例大小增加，利用率会降低，表明工作负载无法充分利用 GPU 实例。图3c 展示了内存使用量。批次较小时，内存使用量仅略有增加；但随着批次大小增加，内存使用量则大幅增加。我们认为，这是因为一开始内存主要由模型大小决定，随后则被较大的批量输入所占用。图 3d 展示了能耗结果，与训练特征分析的结果相同。

4.5 GPU 共享特征分析

在本节中，我们在启用 MIG 或多进程服务的机器上运行多个深度学习工作负载，以探究这两种 GPU 共享技术的性能。除非另有说明，所有实验均在搭载 PyTorch 框架的 A30 GPU 上进行。

第一个实验展示了**平均延迟的比较**，如图4所示。对于评估的两个模型，当推理批量大小较小时，MIG 和多进程服务的平均延迟几乎相同。随着批次大小增加，标准差变大，表明发生了更多干扰，



（a）ResNet18 尾延迟　（b）ResNet50 尾延迟

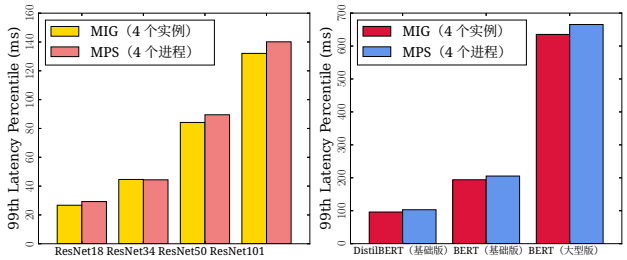
图*6.* MPS 和 MIG 在不同批次大小下的尾延迟对比。实验表明，在批次大小较小时，MPS 的表现与 MIG 相当；批次较大时，MPS 的表现则逊于 MIG。

这证明 MIG 并非总是优于多进程服务。当工作负载较小时，即使是多进程服务这样的软件共享方式，也能实现良好的性能。

为了进一步探究出现的干扰，我们展示了**尾延迟对比**，如图5所示。在该实验中，我们将批次大小设为 = 8，因为此时干扰变得非常显著。图5 从尾延迟的角度来看，MIG 的表现远胜于多进程服务。MIG 的延迟更低，并且能够稳定地处理用户请求。然而，如果用户设置较大的服务级别目标（SLO）（例如，在离线场景中），这一优势便会消失。

我们进一步在不同批次大小下进行实验，以验证上述结论。图6展示了实验结果。结果表明，对于 MIG 和多进程服务上的这两种模型，批次大小较小时，尾延迟差距非常小；随着批次大小增加，差距也随之扩大。这一发现与先前的结果一致，并进一步揭示了更多细节。

本节最后一项实验考察了



（a）ResNet 系列模型　　（b）BERT 系列模型

图 7。 MPS 和 MIG 上不同大小模型的尾延迟对比。当批次大小固定时，MIG 和 MPS 都能很好地支持小型模型，但与 MPS 相比，MIG 对大型模型具有更低的延迟。

不同模型大小在 MIG 和多进程服务上的性能。我们将批次大小设为 8。图 7 显示，对于小型模型，MIG 和多进程服务都能很好地应对干扰；但对于大型模型，MIG 的表现优于多进程服务。这可归因于物理隔离。

4.6 MIG 框架兼容性

本节测试深度学习训练和服务框架与 MIG 的兼容性。除非另有说明，所有实验均在配备 NVIDIA 驱动程序 470.82.01、CUDA 11.4 和 CuDNN 8.2 的 A30 GPU 上进行。

表 1展示了四种训练框架的结果。我们在 A30 GPU 上创建了两个 GPU 实例。然而，由于 NVIDIA 提供的支持有限，所有测试的框架都无法检测到第二个 GPU 实例。因此，只有第一个 GPU 实例可用于训练。同样，对于表 2 中评估的所有服务框架，只有 MIG 0 可用于部署和提供深度学习模型服务。

这一限制可以首先通过利用 Docker 技术来应对。用户可以将一个 Docker 容器绑定到一个 GPU 实例上，以便在指定的 GPU 实例上执行任务，而不受限于 MIG 0。然而，这也会带来其他问题——我们无法在一个 Docker 容器内直接划分或调整 GPU 实例资源。也就是说，每次需要调整 GPU 实例时，都必须停止正在运行的 Docker 容器，并解除其与 GPU 实例的绑定。然后重新配置 GPU 实例的大小，将 Docker 容器绑定到新的 GPU 实例，最后重新运行 Docker 容器。这些复杂且不友好的操作促使我们提出更好的方法，以改进 MIG 上的训练框架和服务框架。我们将其留作未来的工作。

5 总结与未来工作

MIG 是一项技术，可让用户轻松编排深度学习应用的训练和推理工作负载。MIGPerf 是一款开源工具，通过开展一系列

表 *1*。训练框架与 MIG 的兼容性。所有受测框架均只能使用 MIG 0 进行训练。

训练框架	版本	可见设备数量	在 MIG 0 上训练	在 MIG 1 上训练
PyTorch	1.13.0	0	Yes	无设备
TensorFlow	2.11.0	1	Yes	无设备
MxNet	1.9.1	1	Yes	无设备
PaddlePaddle	2.4.1	1	Yes	无设备

表 *2.* 服务框架与 MIG 的兼容性。只有 MIG 0 能够在所有测试的框架上提供机器学习模型服务。

服务框架	版本	服务于 MIG 0	在 MIG 1
TensorFlow Serving	2.8.4	Yes	设备未已找到
Triton 推理服务器	21.09	Yes	未找到设备已找到
Ray Serve	2.2.0	Yes	未找到设备已找到

基准测试研究。这些研究包括训练与推理特征分析、GPU 共享比较和框架兼容性。研究结果揭示了如何有效且高效地使用 MIG。此外，研究结果还指出了若干值得探索的未来研究方向，例如针对 MIG 上训练与推理的混合调度，以及 MIG/MPS 编排。

参考文献

Adolf, R., Rama, S., Reagen, B., Wei, G.-Y. 和 Brooks, D. Fathom：现代深度学习方法的参考工作负载。载于 *2016*年 *IEEE* 国际工作负载特征研讨会 *(IISWC)*，第 1–10 页。IEEE，2016 年。

阿里巴巴。AI Matrix。https://github.com/alibaba/ai-matrix，2018。访问日期：2020-09-02。

百度。不同硬件上的深度学习操作基准测试。https://github.com/baidu-research/DeepBench，2017。访问日期：2020-09-02。

Choquette, J., Gandhi, W., Giroux, O., Stam, N., and Krashinsky, R. NVIDIA A100 张量核心 GPU：性能与创新。*IEEE Micro*，41(2):29–35，2021。

Coleman, C., Narayanan, D., Kang, D., Zhao, T., Zhang, J., Nardi, L., Bailis, P., Olukotun, K., R´e, C., and Zaharia, M. DawnBench：端到端深度学习基准测试与竞赛。训练，100(101):102，2017。

Grigorescu, S., Trasnea, B., Cocias, T., 和 Macesanu, G. 自动驾驶深度学习技术综述。《现场机器人学杂志》，37(3):362–386，2020。

He, K., Zhang, X., Ren, S., 和 Sun, J. 用于图像识别的深度残差学习。见*IEEE* 计算机视觉与模式识别会议论文集，第 770–778 页，2016。

Heaton, J. B., Polson, N. G., 和 Witte, J. H. 面向金融的深度学习：深度投资组合。《商业与工业中的应用随机模型》，33(1):3–12，2017。

Ignatov, A., Timofte, R., Kulik, A., Yang, S., Wang, K., Baum, F., Wu, M., Xu, L., 和 Van Gool, L. AI 基准：2019 年智能手机深度学习全览。见*2019*年 *IEEE/CVF* 国际计算机视觉会议研讨会 *(ICCVW)*，第 3617–3635 页。IEEE，2019。

Jiao, L., Zhang, R., Liu, F., Yang, S., Hou, B., Li, L., 和 Tang, X. 新一代深度学习视频目标检测：综述。*IEEE* 神经网络与学习系统汇刊，2021。

Kaas, A. F., Paleykov, S. P., Robroek, T., 和T¨oz¨un, P. 多实例 GPU 上的深度学习训练。*arXiv* 预印本 *arXiv:2209.06018*，2022。

Liu, D., Li, Y., Lin, J., Li, H., 和 Wu, F. 基于深度学习的视频编码：综述与案例研究。*ACM*计算调查 *(CSUR)*, 53(1):1–35, 2020.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., 和 Stoyanov, V. RoBERTa：一种稳健优化的 BERT 预训练方法。*arXiv* 预印本 *arXiv:1907.11692*, 2019.

Liu, Z., Luo, P., Qiu, S., Wang, X., 和 Tang, X. Deepfashion：借助丰富的标注，推动稳健的服装识别与检索。载于*IEEE*计算机视觉与模式识别会议论文集, pp. 1096–1104, 2016.

NVIDIA. 多进程服务。https://docs.nvidia.com/deploy/pdf/CUDA_Multi_Process_Service_Overview.pdf，2020。访问日期：2020-06-28。

NVIDIA. NVIDIA 多实例 GPU 用户指南。https://docs.nvidia.com/datacenter/tesla/mig-user-guide/index.html，2022a。访问日期：2022-12-15。

NVIDIA。Triton 推理服务器提供了一种针对 NVIDIA GPU 优化的云端推理解决方案。https://github.com/triton-inference-server/server，2022b。访问日期：2022-12-15。

NVIDIA。通过虚拟 GPU 释放下一代性能。https://www.nvidia.com/en-sg/data-center/virtual-solutions/，2022c。访问日期：2022-12-15。

Oprea, S., Martinez-Gonzalez, P., Garcia-Garcia, A., Castro- Vargas, J. A., Orts-Escolano, S., Garcia-Rodriguez, J., 和Argyros, A. 深度学习视频预测技术综述。*IEEE*模式分析与机器智能汇刊，2020。

Ozbayoglu, A. M., Gudelek, M. U., 和 Sezer, O. B. 面向金融应用的深度学习：综述。应用软计算，93:106384，2020。

PyTorch。Pytorchhub。https://pytorch.org/hub/，2022。访问日期：2022-12-15。

Reddi, V. J., Cheng, C., Kanter, D., Mattson, P., Schmuelling, G., Wu, C.-J., Anderson, B., Breughe, M., Charlebois, M., Chou, W. 等。MLPerf 推理基准测试。发表于*2020* 年 *ACM/IEEE* 第 *47* 届年度国际计算机体系结构研讨会（*ISCA*），第 446–459 页。IEEE，2020。

Schwartz, R., Dodge, J., Smith, N.A., 和 Etzioni, O. 绿色人工智能。*ACM* 通讯，63(12)：54–63，2020。

Tan, C., Li, Z., Zhang, J., Cao, Y., Qi, S., Liu, Z., Zhu, Y., 和 Guo, C. 使用多实例 GPU 为深度神经网络模型提供服务：可重构机器调度问题案例。*arXiv* 预印本 *arXiv:2109.11067*，2021。

TensorFlow。TensorFlow Hub。 https://www.tensorflow.org/hub，2022。访问日期：2022-12- 15。

Wu, Z., Lin, G., Tao, Q., 和 Cai, J. M2e-try on net：Fashion，从模型走向每个人。载于第 *27* 届 *ACM* 多媒体国际会议论文集，第 293–301 页，2019 年。

Xiao, W., Bhardwaj, R., Ramjee, R., Sivathanu, M., Kwatra, N., Han, Z., Patel, P., Peng, X., Zhao, H., Zhang, Q., 等。Gandiva：面向深度学习的内省式集群调度。载于第 *13* 届 {*USENIX*} 操作系统设计与实现研讨会（{*OSDI*} *18*），第 595–610 页，2018 年。

Yu, P. 和 Chowdhury, M. Salus：面向深度学习应用的细粒度 GPU 共享原语。*arXiv* 预印本 *arXiv:1902.04610*，2019 年。Zhang, H., Dong, L., Gao, G., Hu, H., Wen, Y., 和 Guan, K. Deepqoe：用于预测视频体验质量（QoE）的多模态学习框架。*IEEE* 多媒体汇刊，22(12):3210–3223，2020a。

Zhang, H., Luo, Y., Ai, Q., Wen, Y., 和 Hu, H. 看、读、感受：通过多模态多任务学习对广告理解进行基准测试。载于第 *28* 届 *ACM* 多媒体国际会议论文集，第 430–438 页，2020b。

基准测试研究的详细评估设置

硬件。 如4.1所述，我们分别在两种不同的 GPU 服务器上开展基准测试研究：A30 服务器和 A100 服务器。表格3 列出了这些服务器的详细信息。

表 *3*。用于基准测试研究的两台 GPU 服务器的硬件详情。

	A100 服务器	**A30 服务器**
物理 CPU 型号	Intel Xeon Platinum 8369B	AMD EPYC 7302P
CPU 插槽数量	2	1
CPU 核心数量	64	16
虚拟 CPU 数量	128	32
内存大小	1024 吉比字节	128 吉比字节
内存通道	32	8
内存类型	DDR4 3200 MT/s	DDR4 3200 MT/s
GPU 型号	8 ×NVIDIA A100（80GB）	2 × NVIDIA A30
NVIDIA 驱动程序版本	470.82.01	515.65.01
CUDA 版本	11.4	11.6
CuDNN 版本	8.2	N.A.
OS	CentOS 7.9.2009	Ubuntu 20.04
Linux 内核	3.10.0-1160.80.1.el7 x86 64	5.15.0-56-generic

模型。我们已对许多开源深度学习模型进行了 MIG 训练和推理特性基准测试。所有模型的详细信息见表4。

机器学习任务	**模型名称**	**开源模型仓库**
图像分类	ResNet-18	Torch Hub
	ResNet-34	Torch Hub
	ResNet-50	Torch Hub
	ResNet-101	Torch Hub
文本序列分类	DistilBERT	Hugging Face
	BERT	Hugging Face
	BERT Large	Hugging Face

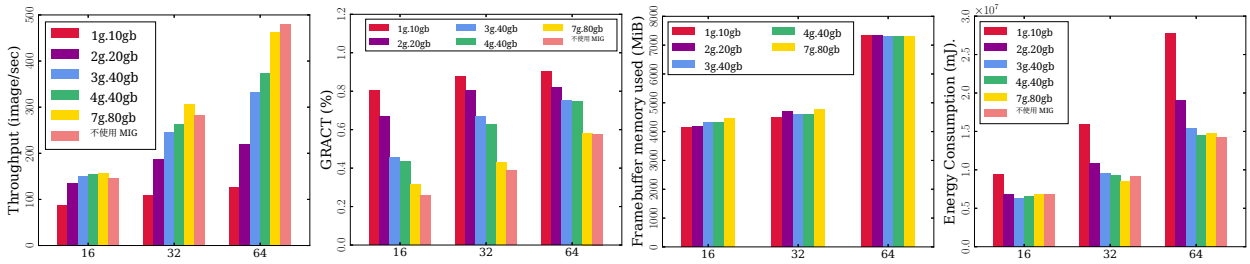
表 *4*。本文所使用的深度学习模型详情

B MIG 训练与推理特征分析

如上一节所述 4.3和 4.4，我们针对更多开源模型进行了 MIG 训练与推理基准测试。本节在图 8 中展示 ResNet-50 的训练结果，并在图 9中展示 ResNet-50 的推理结果。

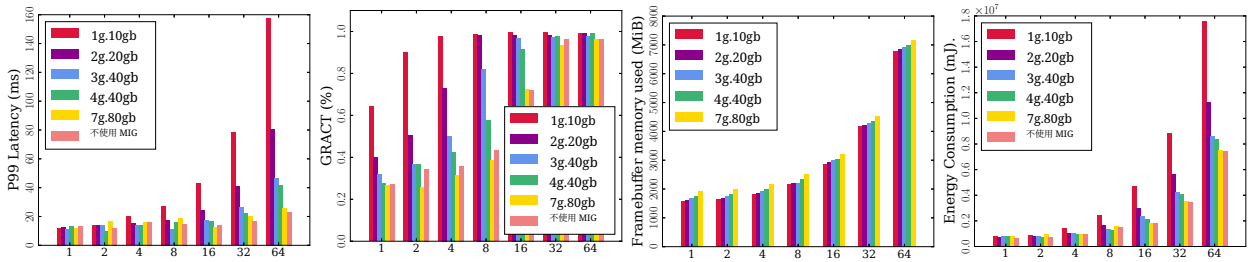
C 图形处理器共享特性分析

除第 4.5 节中的实验外，我们还研究了 MIG 和多进程服务在不同在线推理请求工作负载下的性能。我们运行 4 个简单的 PyTorch 推理服务器，并同时向每台服务器发送异步请求，采用不同的请求工作负载（即请求到达率）。我们将批次大小设为 = 1。图10和图 11 展示了不同在线请求工作负载下，多进程服务和 MIG 的尾延迟。



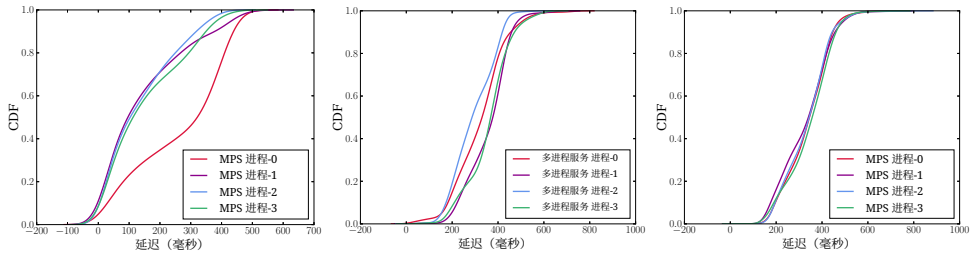
(a) 吞吐量（批次/秒） (b) GRACT（%） (c) 帧缓冲内存（MiB） (d) 能耗（毫焦耳）

图 *8*。输入批次大小对 A100 单个 GPU 实例（GPU 实例）上训练 ResNet-50 的影响。我们展示了批次大小对吞吐量、计算利用率、内存利用率和能耗的影响。



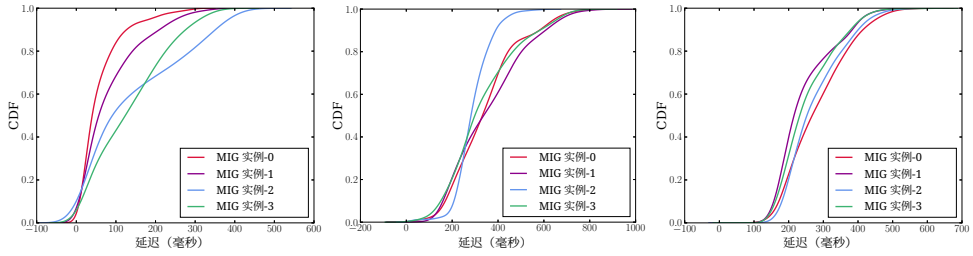
(a) P99 延迟（毫秒） (b) GRACT (%) (c) 帧缓冲内存 (MiB) (d) 能耗 (mJ)

图 *9.* 在 A100 的单个 GPU 实例 (GI) 上进行 ResNet-50 推理时，输入批次大小的影响 .



(a) 到达率 = 25 (b) 到达率 = 50 (c) 到达率 = 200

图 *10*。A30 上 4 个采用多进程服务的 ResNet-50 推理进程在不同请求到达率下的尾延迟对比。



(a) 到达率 = 25 (b) 到达率 = 50 (c) 到达率 = 200

图 *11*。A30 上 4 个 MIG 1g.6gb GPU 实例中的 4 个 ResNet-50 推理进程在不同请求到达率下的尾延迟对比。