研发与探讨

基于远程GPU虚拟化的边缘智能体应用

徐玉清[1]，吕轩民[2]，肖潇[1]
（1. 上海理想信息产业（集团）有限公司，上海 201315；
2. 常州技师学院，江苏 常州 213032）

摘　要：在人工智能的飞速发展中，大型语言模型（LLM）以其在自然语言处理（NLP）领域的革命性突破，引领着技术进步的新浪潮。然而，这些模型传统上主要部署在云端服务器上，这种做法虽然保证了强大的计算力支持，却也带来了一系列挑战：网络延迟、数据安全、持续的联网要求等。正因如此，将LLM部署在端侧设备上的探索应运而生，它不仅能够提供更快的响应速度，还能在保护用户隐私的同时，实现个性化的用户体验。然而，市场上可用的边缘智能体平台都是低功耗设备，计算能力有限。提出了一种新颖的方法，通过使用远程GPU虚拟化技术，为边缘智能体提供算力资源，而不会影响其功耗。与使用本地算力相比，在边缘设备上使用远程GPU虚拟化性能提升了3.2倍。

关键词：远程GPU虚拟化；机器学习；聚类算法；边缘计算
中图分类号：TP393
文献标志码：A
doi: 10.11959/j.issn.1000−0801.2025084

0　引言

自2017年Transformer架构诞生以来，我们见证了从OpenAI的GPT系列到Meta的LLaMA系列等一系列模型的崛起，它们不仅在技术层面上不断刷新我们对机器理解与生成人类语言能力的认知，更在实际应用中展现出巨大的潜力和价值。

随着技术的不断进步，边缘AI市场的全球规模正以惊人的速度增长。预计从2022年的152亿美元增长到2032年的1 436亿美元，这近十倍的增长不仅反映了市场对边缘AI解决方案的迫切需求，也预示着在制造、汽车、消费品等多个行业中，边缘AI技术将发挥越来越重要的作用。2022—2032年全球终端用户端AI市场规模如图1所示。

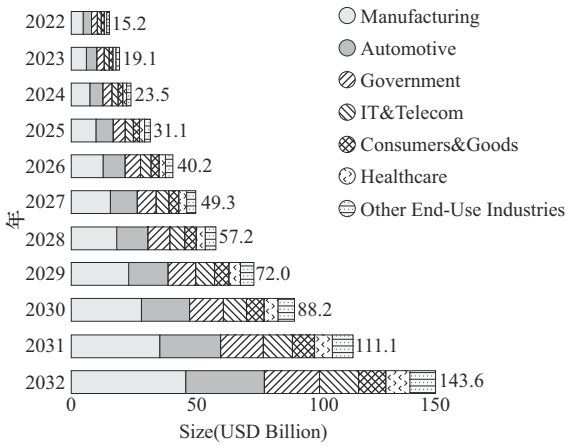

图1　2022—2032年全球终端用户端AI市场规模
（Market.us Online Report, July 2024）

边缘AI市场将以25.9%的复合年增长率增长。预计2032年的市场规模为1 436亿美元（美国2024年）。

在这样的背景下，本文深入探讨了以一种透

收稿日期：2024−12−09；修回日期：2025−02−07

明的方式，向AI应用提供位于远程节点上的算力资源（主要是GPU），该技术充分提高了GPU利用率，减少了GPU在空闲状态下的资源浪费。

目前业界常用的几种GPU虚拟化解决方案有VCUDA、DS-CUDA、GViM、rCUDA、GVirtus、Grid-CUDA等，这些框架提供了与NVIDIA CUDA库兼容的类CUDA API实现。它们通常是客户端-服务器的部署方式，客户端是一个透明请求GPU资源的本地节点。服务器端是运行在物理GPU资源所在远端节点上的守护进程。当AI应用程序执行CUDA调用时，它都会被客户端捕获并转发到服务器以在物理GPU上执行。一旦CUDA函数被执行，结果将返回给客户端并转发给CUDA应用程序。在这个过程中，AI应用程序并不知道CUDA被劫持的过程。此外，这些解决方案还允许在多个AI应用程序之间并发地共享远程GPU资源。GPU虚拟化带来的好处是有效增强GPU利用率，减少集群物理GPU卡的数量，降低GPU的开销和能耗，同时缩短作业的执行时间。

本文提出一种在不增加设备功耗的前提下，能够显著增加边缘智能体的计算能力的方法。为此，我们对基于rCUDA的远程GPU虚拟化方案进行了全面评估，客户端节点使用NVIDIA Jetson AGX Xavier代表边缘智能体。与其它可用的远程GPU虚拟化解决方案相比，这种方式提供了较好的性能和CUDA兼容性。本文采用模拟极值算法FM（fuzzy minimals）进行性能评估，并提供了FM算法在多GPU的实现，并且对NVIDIA Jetson AGX Xavierf在边缘设备的负载执行情况进行了深入评估。

1 相关工作

1.1 智能边缘体

在人工智能的浪潮中，端侧大型语言模型（on-device LLM）正以其迅猛的发展速度和广泛的应用前景，成为技术革新的新宠。自2023年起，随着参数量低于10B的模型系列如Meta的LLaMA、Microsoft的Phi系列等的涌现，我们见证了LLM在边缘设备上运行的可行性和重要性。这些模型不仅在性能上取得了长足的进步，更通过混合、量化和压缩等技术，保持了参数量的优化，为边缘设备的多样化场景应用提供了强大支持。

进入2024年，新模型的推出愈发密集，Nexa AI的Octopus系列、Google的Gemma系列等，它们不仅在文本处理上有所增强，更在多模态能力上展现了新的可能性，如结合文本与图像等多模态输入，以适应更复杂的用户交互需求。

然而，要在资源受限的设备上部署这些强大的模型，我们必须面对内存和计算能力的双重挑战。相较于完全依赖云端的LLM服务，边缘端侧推理的优势显而易见。它不仅减少了数据传输的延迟，更保护了用户数据的隐私安全。端侧推理的低延迟特性，尤其适用于需要实时响应的应用场景，如Google的Gemini Nano支持的Talk-Back功能，即便在完全离线的情况下也能正常工作。

图2展示了2023年以来的重要模型和发展里程碑，如LLaVa series、QwenVL、Gemini Nano，以及Yi VL等。这些模型在边缘设备进行了部署多模态LLM的尝试，以适应边缘智能体上更复杂和不断变化的用户场景。

在边缘设备上部署大模型正成为人工智能领域的新挑战。面对有限的内存和计算能力，架构创新变得尤为关键，其中包括参数共享、模块化设计以及紧凑的表示形式。例如，MobileLLM通过深度和瘦身的模型结构优化了参数量在十亿以下的模型，而EdgeShard框架则通过边缘云协作计算实现了模型的分布式处理，显著降低了延迟并提高了吞吐量。

在边缘设备上部署大模型时，保持性能的同

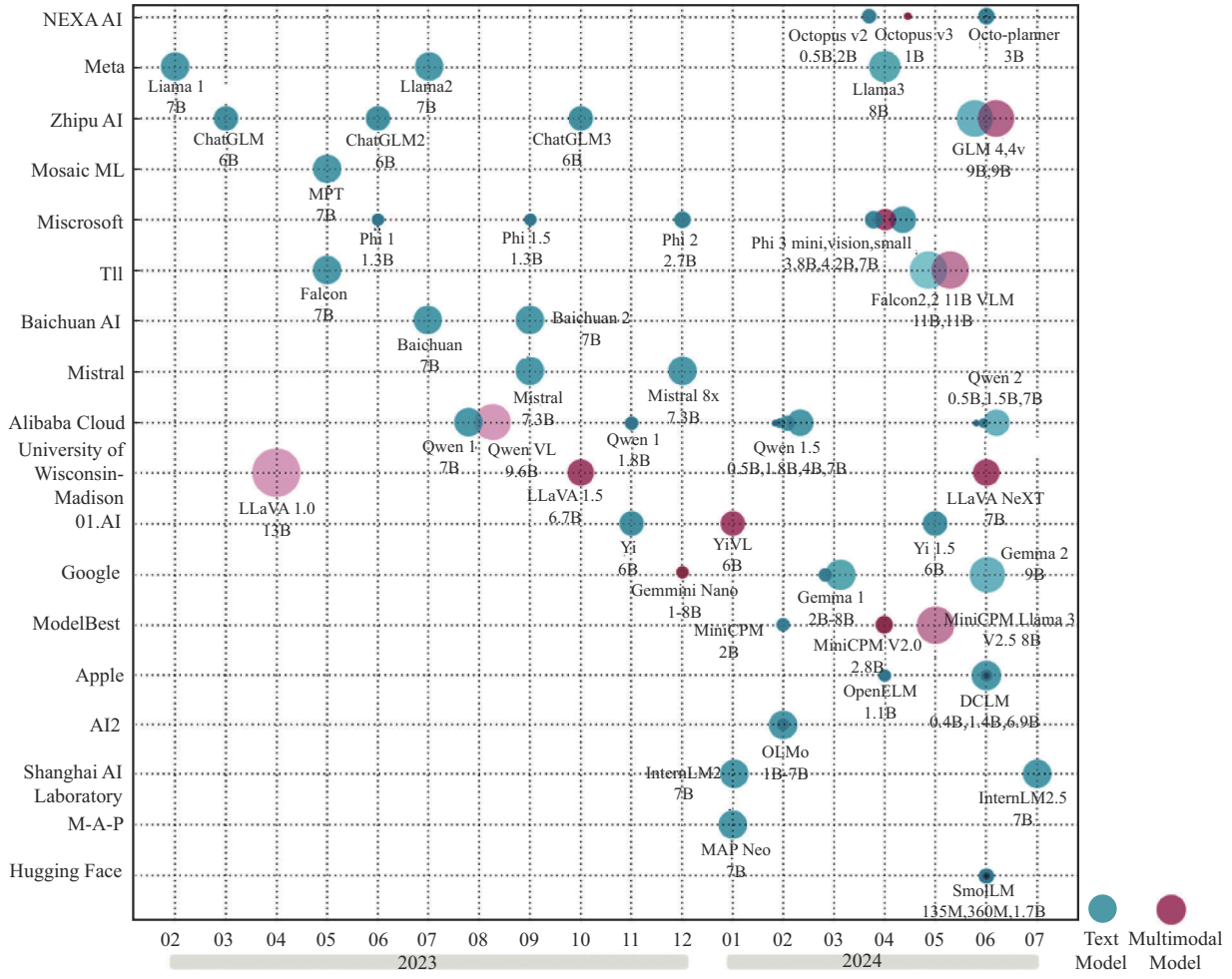

图2　边缘设备LLM发展概况

时提升计算效率尤为关键。衡量端侧LLM性能的指标包括延迟、推理速度、内存消耗等。这些指标直接关系到模型在边缘设备上的实际运行效果，以及用户的使用体验。本文采用远程GPU虚拟化的方式，端侧设备可以通过网络远程使用服务端的算力资源（主要是GPU），使得这些性能指标能得到进一步的优化，使得端侧大语言模型能在更多场景下发挥其潜力。

本文主要探讨通过远程GPU虚拟化技术，来提升边缘智能体使用数据中心GPU资源性能的可行方案以及评估方法。

1.2　远程GPU虚拟化

GPU在现代计算需求中扮演着举足轻重的角色，云计算提高资源利用率和降低总拥有成本的关键组成部分是虚拟化。采用GPU虚拟化技术实现GPU共享，提高资源利用率。由于GPU资源非常昂贵，为了更好地利用GPU设备，引入了硬件辅助虚拟化技术，如NVIDIA GRID、NVIDIA MIG和AMD MXGPU。然而，目前主要是支持本地GPU访问，只有配备了GPU的服务器可以托管GPU虚拟化，这个约束限制了部署位置，而且使得在该位置以外的GPU资源无法访问。

为了提供更大的灵活性，远程GPU虚拟化是解决方案之一，通过将GPU访问请求与GPU资源分配解耦，将虚拟机及其虚拟GPU VGPU分配到不同的机器，允许应用程序通过网络远程访问

数据中心任何可用的GPU资源，从而提供更大的灵活性，以达到更好的性能。GPU虚拟化还允许用户根据计算需求分别选择不同的VM和VGPU实例，而且不需要额外的硬件支持。

尽管带来的好处很多，但是远程GPU虚拟化至今仍是一个具有挑战性的问题，多个vGPU共用一个GPU，虚拟机VMs与远程分配的vGPU的通信开销可能会导致性能下降。另一个棘手的困难是如何放置VMs和vGPU以降低开销。目前一个较广泛的实现是rCUDA技术，图1展示了基于rCUDA的远程GPU虚拟化架构。VMware vSphere BitFusion提供了商业解决方案，可虚拟化硬件加速器（如GPU），以提供可通过网络访问的共享资源池。此外，Amazon Elastic Graphics（已停服）可利用API远程连接，使vGPU能够连接到虚拟机。

由于技术上的困难，以前工作的重点主要集中在采用GPU的API远程处理上，但是不正确的资源配置可能会导致资源碎片化和失败率的增加。此外，过多的GPU共享和远程vGPU访问可能会导致性能下降，VMs的放置问题的复杂性也随着远程vGPU的引入而增加。

如图3所示，远程GPU虚拟化的架构遵循客户端-服务器的方式，客户端通过拦截AI应用向CUDA发出的调用请求，将其转发给远程搭载GPU的服务端。其中rCUDA对AI应用完全透明的方式工作，不必更改其应用程序的源代码。实际上，如果应用程序被编译为动态使用CUDA库，那么AI应用程序甚至不需要重新编译就可以使用rCUDA执行。在这方面，rCUDA与CUDA是二进制兼容的，并实现了整个CUDA运行时和驱动程序API（除了图形功能）。它还支持CUDA中包含的库（CuBLAS，CuFFT等）。此外，还支持TCP/IP、RoCE和InfiniBand等网络通信协议的底层互联。另外，rCUDA在客户端和服务器之间的数据交换是流水线的，rCUDA中的内部管道缓冲区使用预分配的固定内存，以具有更高的吞吐量。rCUDA支持应用程序使用多租户，可以为同一个客户端应用程序提供相同远程GPU的多个虚拟实例。也就是说，使用rCUDA可以将物理GPU划分为几个虚拟实例，并将它们提供给同一个客户端应用程序。通过这种方式，客户端应用程序在逻辑上可以访问多个远程GPU，所有这些远程GPU最终都映射到同一个物理GPU资源上面。

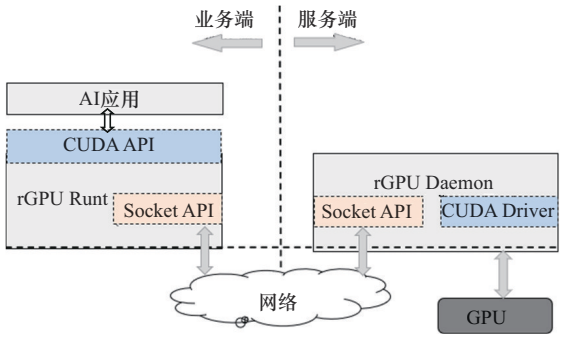

图3　远程GPU虚拟化架构

需要注意的是，rCUDA保证在映射到相同物理GPU的虚拟GPU实例之间具有与CUDA所保证的特性相同的隔离和安全性，这是通过在同一虚拟GPU上运行的每个虚拟GPU实例使用不同的GPU上下文来实现的。因此，使用rCUDA时，只要CUDA保证GPU上下文之间的隔离，就能在并发使用相同物理远程GPU的应用程序时避免数据泄漏。

1.3　模糊极值算法（FM）

聚类分析是把相似的对象通过静态分类的方法分成不同的组别或者更多的子集，这样让在同一个子集中的成员对象都有相似的一些属性，而不同子集的成员对象尽可能与前者不同。这种相似性过程基于对目标函数的评估（即最小化），目标函数包括距离、连通性和/或强度等度量。根据数据集特征或应用需求，可以选择不同的目标函数。

模糊聚类（fuzzy clustering）是一种聚类分析方法，与传统的硬聚类（hard clustering）不同，

它允许样本属于多个聚类的成员关系程度不同。在模糊聚类中，每个数据点都被赋予属于每个聚类的隶属度（membership degree），而不是严格地归属于某一个聚类。这使得模糊聚类对于那些难以明确划分到某个特定聚类的数据更具有鲁棒性。

模糊聚类是一组分类算法，其中每个数据点可以属于多个聚类。通常用一个向量来表示一个数据点的归属，向量中哪个维度的数值更大，意味着该数据点距离该维度对应簇更近，也可以理解为是归属于该簇的概率越大。模糊聚类最常用的方法包括模糊 C 均值（fuzzy c-means，FCM）算法和模糊极小（fuzzy minimals，FM）算法。FCM 将数据点与聚类中心之间的距离作为样本与聚类的隶属度的衡量标准，通过迭代优化聚类中心和样本的隶属度来最小化目标函数（通常是样本与其所属聚类中心之间的加权平方误差）。在每次迭代中，样本的隶属度会根据与各个聚类中心的距离进行更新，直到达到收敛条件。FM 算法最初是由 Flores-Sintas 等提出的，作者证明它符合分类算法的预期特征，即稳定性、适应性、自驱动、稳定性、数据独立。此外，FM 算法不需要在数据集中设置要识别的原型的数量，因为它使用的是 k-means 或 FCM 算法。

FM 算法分为两个主要函数：Calculate_rFactor 和 Calculate_Prototypes。它们有不同的计算模式，会影响它们在目标虚拟化环境中的性能，下面给出了 FM 算法的概要描述。

函数 Calculate_rFactor 的定义如下。

设 X 为 n 个数据点的集合，使得 $X=\{x_1,x_2,\cdots,x_n\}\subset R^F$，其中，$F$ 是向量空间的维数，X 为待分类的输入数据集，V 为包含聚类过程发现的原型的算法输出。

- 选择 ε_1 和 ε_2 作为标准参数。
- 初始化 $V=\{\}\subset R^F$。
- 加载数据集 X。
- $r=$Calculate_r_Factor(X)。

计算原型 Calculate_ Pr ototypes($X,r,\varepsilon_1,\varepsilon_2,V$)。

FM 算法有两个主要过程：通过因子 r 的计算和原型的计算来完成集合 V。因子 r 是测量数据集中各向同性的参数。每当同质性和各向同性被打破时，就会在特征空间中创建聚类。

$$\frac{\sqrt{\left|C^{-1}\right|}}{nr^F}\sum_{x\in X}\frac{1}{1+r^2d_{xm}^2}=1 \tag{1}$$

因子 r 的计算基于式（1），它是一个非线性表达式，其中 $\left|C^{-1}\right|$ 为协方差矩阵逆的行列式，m 为样本 X 的均值，d_{xm} 是 x 和 m 的欧氏距离，n 是样本中元素的个数。

当计算出因子 r，通过原型的计算，得到集合 V 中的聚类结果。FM 的目标函数由式（2）给出。

$$J_{(v)}=\sum_{x\in X}\mu_{xv}\cdot d_{xv}^2 \tag{2}$$

函数 Calculate_Prototypes()的定义如下。

for $k=1;k<n;k=k+1$ do
　$v_{(0)}=x_k,\ t=0,\ E_{(0)}=1$
　while $E_{(t)}\geqslant\varepsilon_1$ do
　　$t=t+1$
　　$\mu_{xv}=\dfrac{1}{1+r^2\cdot d_{xv}^2}$, using $v_{(t-1)}$
　　$v_{(t)}=\dfrac{\sum_{x\in X}(\mu_{xv}^{(t)})^2\cdot x}{(\mu_{xv}^{(t)})^2}$
　end while
　if $\sum_{\alpha}^{\mathrm{F}}(v^{\partial}-w^{\partial})>\varepsilon_2,\forall w\in V$ then
　　$V\equiv V+\{v\}$
　end if
end for

其中，

$$\mu_{xv}=\frac{1}{1+r^2\cdot d_{xv}^2} \tag{3}$$

式（3）给定元素 x 与集群的关联程度，其中 v 为原型。FM 算法是一个迭代过程，其目的是通过式（4）最小化目标函数，给出每个聚类所代表的原型。ε_1 和 ε_2 是输入参数，它们表示最小估

计中的误差程度，显示与潜在最小值的差异。

$$v=\frac{\sum_{x\in X}\mu_{xv}^{2}\cdot x}{\sum_{x\in X}\mu_{xv}^{2}} \tag{4}$$

2 GPU集群的并行化策略

本节介绍了GPU集群环境下FM算法的并行化计算。FM算法本来是在单个GPU上进行计算，本文对FM算法进行重新设计，以支持GPU集群环境。rCUDA允许应用程序以共享内存的方式，在单个客户端节点上使用多个GPU。FM算法的分布式并行化，称为并行模糊极小值（PFM），是FM算法的并行版本，可以提高数据并行性。与其它算法不同，FM算法不需要聚类比较来最小化目标函数。PFM利用这一特性将原始数据集分割成不同的子集，并独立应用FM。聚类过程不受这种划分的影响，因为原始数据集的全局属性不会丢失。实际上，因子r是一个全局参数，具有有关整体数据结构的信息，每个子集都可以使用包含因子r的目标函数进行分类。如图4所示。

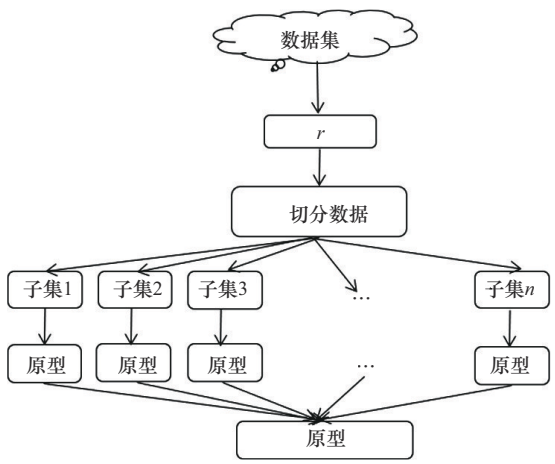

图4 模糊极值算法的实现

为了充分利用PFM的内在并行性，计算因子r时需要遍历整个数据集，因子r计算随着聚类变量的数量增长呈指数增长，这可能会成为一个瓶颈。因此，首先利用GPU集群计算因子r。因子r处理过程对数据集执行两个主要任务，一是计算模糊协方差矩阵，二是计算行列式。这是个迭代过程，迭代次数同数据集行数。这些迭代任务可以独立地进行，因此它们可以均匀地分布在集群系统中每个可用的GPU中。并行计算可使用OpenMP来实现因子r的简化，OpenMP是由Architecture Review Board牵头提出的，并已被广泛接受，是为共享内存并行系统的多处理器程序设计的一套指导性编译处理方案。此外，性能还受到矩阵维数大小的影响，因为计算的执行时间将随着矩阵维数的增加呈指数增长。通过LU分解，计算模糊协方差矩阵获得因子r值。

对原型计算进行多GPU并行化处理，将输入数据集均匀地分成子集，并把这些子集分发到rCUDA客户端节点上可用的GPU资源。使用每个GPU线程来计算数据集的每行与原型之间的距离，从而获得行数据属于子集的概率。在每个GPU中，将每个数据块简化后获得数据集中各点的总相关度。在这个过程中，CPU和GPU之间存在大量的同步和数据交互。

本文中使用的NVIDIA Jetson Xavier和集群节点的组合代表了一种通用的硬件解决方案。在后续的实验中使用NVIDIA Jetson Xavier设备作为边缘智能体部署的代表。其中FM应用程序将与rCUDA客户机一起执行。rCUDA提供的远程硬件资源执行FM计算，而不使用位于NVIDIA Jetson Xavier节点内的GPU资源。

NVIDIA AGX Jetson Xavier与GPU服务器的连接是主要的基础设施瓶颈之一。为了计算边缘智能体的设备能耗，以秒为间隔测量了所使用的每个设备消耗的功率。NVIDIA Jetson AGX Xavier所消耗的功率已使用Watts Up Pro功率计进行测量。对于GPU卡的功耗是使用NVIDIA管理库（NVML）测量的。

需要注意的是，使用远程虚拟化技术的GPU

性能并不完全取决于数据大小。实际上，影响性能的重要因素还包括CPU与GPU之间通信的次数、应用程序执行的同步、cudaMemcpy和CUDA内核启动的数量等。

实验结果表明，当使用多个虚拟实例时，边缘节点所需的能量明显减少，随着多租户程度的增加，边缘体的总能耗降低。另一方面，总体能耗总是大于局部执行FM算法时的能耗。随着虚拟GPU实例数量的增加，Xavier的CPU工作负载也会增加，这将转化为略高的功耗，但与使用Xavier的本地GPU相比，边缘体节省了更多的功耗。特别是使用的虚拟GPU实例越多，节省的功耗就越多，实验结果符合预期。

3　结束语

本文评估了边缘智能体设备通过远程API访问GPU虚拟化，在不增加设备功耗的情况下为它们提供计算能力。研究结果表明，这种方式为边缘计算作为智能应用提供了一个较好的使用算力的方案。GPU的远程虚拟化可以在不增加边缘体功率预算的情况下提高计算能力。通过使用单个远程GPU运行多个虚拟GPU实例，性能获得了高达3.2倍的加速因子。此外，边缘设备的功耗降低30%，通过将计算任务分发到GPU服务器，获得了更多的算力资源。整体能耗的增加被稀释，使得边缘智能体的节能更具吸引力。

虚拟化与边缘计算的结合，目前还处于相对早期的阶段。到目前为止，本文只测试了这种场景下的一个相对简单的类型，还有许多其他类型的数据科学算法仍有待探索，而且rCUDA框架也存在不少问题，可能并不是实现远程GPU虚拟化的最佳实践，还存在可以进一步优化的地方。

最后，未来的研究方向还包括云协作，通过智能缓存、请求分析和资源分配算法，优化数据在边缘设备与云服务器间的传输。另一个方向是模拟性能与功耗，本文中使用的不同硬件参数，提供了一个有用的工具来评估在边缘计算中使用远程GPU虚拟化的效果。

参考文献：

[1] 王浩, 王浩枫. 面向CPUs-GPUs系统的OpenCL任务调度框架[J]. 计算机工程与设计, 2022, 43(7): 1955-1963.

[2] 崔雪冰, 张延红, 李国徽. 基于通用计算的GPU-CPU协作计算模式研究[J]. 微电子学与计算机, 2009, 26(8): 30-33.

[3] 崔嘉. 试析虚拟计算环境中资源池的资源聚合机制的研究[J]. 自动化技术与应用, 2017, 36(6): 35-37, 41.

[4] 查乾. 基于GPU虚拟化的资源优化调度[D]. 武汉: 武汉理工大学, 2022.

[5] 吴再龙, 王利明, 徐震, 等. GPU虚拟化技术及其安全问题综述[J]. 信息安全学报, 2022, 7(2): 30-58.

[6] 梁桂才, 李玉荣. 混合现实中基于GPU虚拟化的AI计算优化[J]. 通信与信息技术, 2024(02): 114-120.

[作者简介]

徐玉清（1978− ），男，上海理想信息产业（集团）有限公司、高级工程师、事业部经理，主要研究方向为云计算及算力、云端协同、GPU虚拟化技术。

吕轩民（1983− ），男，常州技师学院高级工程师、信息服务学院副院长，主要研究方向为基于5G移动通信技术的高速工业互联网架构设计与搭建，公有云信息化系统架构设计、集成、开发、运维，微服务信息化系统架构设计、开发、运维。

肖潇（1998− ），女，上海理想信息产业（集团）有限公司工程师，主要研究方向为人工智能、大模型应用。