HeteroEdge：解决异构协作自主系统中的不对称性

穆罕默德·赛义德·安瓦尔[1]，埃蒙·德伊[1,2]，马洛伊·库马尔·德夫纳特[1]，因德拉吉特·戈什[1,2]，奈玛·汗[1,2]，杰德·弗里曼[4]，蒂莫西·格雷戈里[4]，尼兰詹·苏里[4]，卡斯图里·贾亚拉贾[2]，斯里尼瓦桑·拉马萨米·拉马穆尔蒂[3]，尼尔马利亚·罗伊[1,2]
[1]马里兰大学巴尔的摩县分校（UMBC）信息系统系，[2]CARDS，UMBC[3]鲍伊州立大学计算机科学系，[4]DEVCOM 陆军研究实验室[1](saeid.anwar, maloyd1)@umbc.edu, [1,2](edey1, indrajeetghosh, nkhan4, nroy)@umbc.edu, [2]kasthuri@umbc.edu,[3]sramamurthy@bowiestate.edu, [4](jade.I.freeman2, timothy.c.gregory6, niranjan.suri).civ@army.mil

arXiv:2305.03252v1 [cs.DC] 5 May 2023

摘要——**对于面向智慧城市和非争夺环境开发的系统而言，获取周围环境信息并为自主系统生成态势感知至关重要。例如，大范围监控系统通常配备摄像头和激光雷达等多模态传感器，并需要执行深度学习算法，以识别动作、人脸、行为和物体。然而，由于这些系统无处不在，它们受到功率和内存限制。因此，优化传感数据的处理方式、向深度学习算法提供数据的方式，以及模型推理结果的通信方式，至关重要。本文考虑一个由两辆无人地面车辆（UGV）和两台 NVIDIA Jetson 设备组成的测试平台，并提出一种自适应优化框架，该框架能够在多个异构节点上协同调度多个任务（存储、处理、计算、传输、推理）的工作负载，同时处理多项任务。该自适应优化框架包括压缩和遮蔽输入图像帧、识别相似帧，以及对设备执行各类任务的性能进行分析，以获取优化框架的边界条件。最后，我们提出并优化一种新颖的参数**分割比**，该参数表示在考虑测试平台中设备的网络带宽、繁忙系数、内存（CPU、GPU、RAM）和功耗约束时，需要卸载到另一台设备的数据比例。我们在同时执行多项任务（例如 PoseNet、SegNet、ImageNet、DetectNet、DepthNet）时进行的评估表明，与基线配置（在**主节点**上执行）相比，在**辅助节点**上执行 70% 的数据（**分割比**=70%）可将卸载延迟降低 ≈ 33%（从 18.7 毫秒/图像降至 12.5 毫秒/图像），并将总运行时间缩短 ≈ 47%（从 69.32 秒降至 36.43 秒）。**

索引词—**协作系统、深度边缘智能、自主系统**

I. 引言

近年来，无人机、无人地面车辆等自主系统在各种应用领域中广受欢迎，例如监控、摄影与摄像、制图与测绘、农业、环境监测、搜索与救援以及配送服务。配备摄像头、激光雷达、雷达和 GPS 等传感器的无人车辆，使我们能够在各种环境中收集数据并执行任务（例如目标检测、场景检测等）。在执行这些任务时，如果依赖车载计算单元（尤其是先进的深度学习模型）涉及的功耗有限，最终会影响自主系统的运行与安全 [1]。近期有多项研究调查了运行时间和电源对自主系统的影响 [2]–[4]，并指出，由于需要运行机载子系统（例如导航单元、摄像头和通信系统），系统的运行能力会受到严重影响 [4]。除了这些子系统之外，为执行态势感知任务而部署深度神经网络（DNN）算法（通常十分耗费功率、内存和计算资源），也会进一步加剧可用功率本已有限的问题 [5]。此外，鉴于导航和通信等系统操作对昂贵自主系统的安全至关重要，优化 DNN 对于节省有限的可用功率十分必要。

使用深度神经网络节省功率的一种方法，是将推理任务卸载到具有多余功率和计算能力的远程设备（云服务器或连接到同一网络的设备）上 [6]。然而，此类解决方案会受到网络可用性、可靠性、低带宽和延迟的影响 [7]；这些问题是由通信链路质量以及（**主节点**）（即将卸载数据的设备）与**辅助节点**（可分担主节点工作负载的远程服务器或同一网络中的边缘设备）之间的距离造成的。以往有关卸载的研究存在一些局限，包括将任务卸载到同质设备（例如 MASA [8]）和智能手机 [9]，而使用远程云服务则成本高昂 [10]。对于自主系统的态势感知等场景，我们假设，利用系统内的另一台设备，相较于成本高昂的云服务，可以降低延迟并帮助节省功率。基于以上讨论，本文的总体目标是通过将繁忙**主节点**的数据卸载到相对空闲的**辅助节点**，对任务进行优化和调度。为实现这一目标，我们提出以下贡献。

（i）数据驱动的资源感知型卸载框架。该框架通过优化任务处理复杂度、内存利用率、带宽和可用功率等系统参数，促使**主节点**

将部分数据卸载到辅助节点。此外，我们引入了一个名为分割比的新参数，以帮助我们高效地将数据卸载到辅助节点。我们的分析表明，采用 0.7-0.8 的分割比 进行卸载，可以提升任务执行性能，但会增加功率和内存使用量（主节点 + 辅助节点）。

（ii）用于系统评估的测试平台开发。我们设计了一个由两台 Nvidia Jetson 设备和两辆无人地面车辆组成的系统，用于评估优化框架。设备配备了基于 MQTT 的发布-订阅协议，用于将辅助节点的系统参数共享给主节点，并将数据卸载至辅助节点。为展示所提出的优化框架的有效性，我们评估了其在具有多种数据模态的深度神经网络应用中的性能，包括姿态估计（识别站立、坐着或躺下等人体姿态）、语义分割（将图像中的每个像素分类到其所属对象，以便详细理解场景）和目标检测。

（iii）通过数据压缩提升优化性能。随着数据量增大，卸载数据的成本也随之增加。因此，我们采用帧压缩和掩膜技术，剔除相似帧并从待卸载数据中提取感兴趣目标，从而缩短推理时间并降低通信开销，最终提升数据密集型环境中的整体性能。

(iv) 模拟真实场景以评估卸载策略。 在真实的自主系统运行过程中，各个节点都处于运动状态，这意味着节点之间的距离会影响网络参数。因此，我们模拟了一个节点持续运动的场景。随着距离增加，卸载延迟也随之上升，这引发了一个问题：何时应停止卸载图像？这项探索能带来宝贵的见解，例如了解距离、卸载与延迟之间的关系，确定提高效率的卸载阈值，并优化动态环境中的图像卸载策略，从而最终提升真实场景下的性能。

II. 相关工作

多个并发深度神经网络任务的边缘推理和卸载。对于任何自主系统的实时运行而言，同时进行多个深度神经网络任务的推理是一项至关重要的显著特性。受此启发，heimdall 的作者 [5]开发了一个面向新兴增强现实应用的移动 GPU 协调平台，在这些应用中，由于多深度神经网络 GPU 竞争，帧率会下降，推理延迟会显著增加。该平台采用了一种伪抢占机制，具体包括：(i) 将多深度神经网络任务拆分为更小的单元；(ii) 对同时运行的 GPU 任务进行优先级排序并灵活调度。此外，在 BAND [11], 中，作者开发了一个基于移动设备的推理系统。BAND 根据既定的调度目标，动态支持创建深度神经网络执行计划，并在处理器上调度深度神经网络。相比之下，在本文中，我们考虑了更广泛的边缘设备和应用场景，最终尽量降低带宽消耗和延迟。

基于任务的调度：基于任务的调度在优化推理并减少已分配任务的推理开销方面发挥着至关重要的作用。 [12] 的作者针对约束优化问题提出了一种新颖的方法，并建议采用贪心启发式方法，在保真度和资源预算受限的条件下选择最佳的并发应用子集。此外，LaLaRAND [13],是一种实时层级深度神经网络调度框架，支持对单个深度神经网络层进行 CPU/GPU 调度，并采用细粒度的 CPU/GPU 分配方案。这项工作解决了实时深度神经网络任务的可调度性、深度神经网络任务分别在 CPU 和 GPU 上执行时的非对称性，以及缺乏基于任务的 CPU/GPU 感知分配方案调度等问题。相比之下，我们的工作通过将数据拆分到不同节点上并发运行多个深度神经网络应用，同时考虑每台设备的能力和任务需求。

基于帧的压缩技术。 先前的研究表明，可以利用光流估计多个摄像头视角下的物体运动，使系统能够跟踪在摄像头之间移动的物体 [14]。这些信息有助于安排视频帧处理，在满足实时视频分析需求的同时，将延迟降至最低并优先处理相关帧。另一方面，AdaMask [15]提出了一种自适应帧掩码方法，可在边缘计算环境中高效地传输和处理视频，重点在于降低通信开销并加速深度神经网络推理。然而， [14] 和 [15] 都旨在压缩由静态摄像头采集的帧。相比之下，我们同时考虑静态和移动自主设备，以优化图像卸载策略，从而提升各种真实场景中的效率和效能。

III. 系统概述

如图1所示，我们的系统架构由设备性能分析器和在线调度器组成。输入为图像数据，并根据资源可用性进行拆分。

在设计该框架时，我们考虑三个重要部分：1. 帧掩码、性能分析引擎和优化。这包括设计一种高效的帧掩码方案，以确保边缘设备之间卸载时的最佳性能；还包括创建一个性能分析引擎，准确评估主节点和辅助节点的性能，同时考虑内存、功率和推理时间，并适应无人地面车辆移动等动态条件。此外，还需要一个优化框架，在规定范围内确定卸载决策的最佳拆分比例，从而为多深度神经网络系统提供全面且自适应的解决方案。

A. HeteroEdge 组件

设备分析器。 为了实现内存和功耗感知调度，我们分析并监控两台设备的性能，重点关注设备内存、功耗、推理时间和网络延迟等关键指标。通过

通过收集这些信息，我们能够实时评估每台设备的资源可用性，并就处理任务的分配做出明智决策。

任务调度器。在我们的优化框架中，我们集成了一个任务调度器，用于在多节点环境中智能地管理任务卸载。通过收集主设备和辅助设备的性能分析数据，该调度器能够评估资源可用性和多DNN工作负载。随后，任务调度器会判断主节点是否需要进行卸载，并计算最优的数据拆分比例，以实现高效的资源利用。作为一个智能决策系统，它根据资源可用性和性能能力分配工作负载，从而缩短推理时间、降低内存利用率，并提升整个系统的效率。

在资源受限的边缘环境中，为多DNN执行系统设计高效的卸载框架面临着一些困难。该框架必须解决设备异构性、资源约束、性能变化和能效问题，同时适应速度和距离变化等动态条件。在移动条件下，由于设备之间的距离发生变化，卸载延迟可能随之波动，从而导致潜在的效率低下。在边缘环境中处理大型图像数据也面临挑战，因为数据传输量的增加会消耗更多带宽，并导致更高的延迟。此外，处理更大的图像需要更强的计算能力，这会给资源受限设备带来更大的负担。为了解决这一问题，我们引入了一种帧压缩技术，具体而言，即帧掩码，这对于确保卸载过程中的最佳性能至关重要。帧掩码能够减小图像数据的大小，从而最大限度地降低带宽消耗、减少延迟并提高能效。

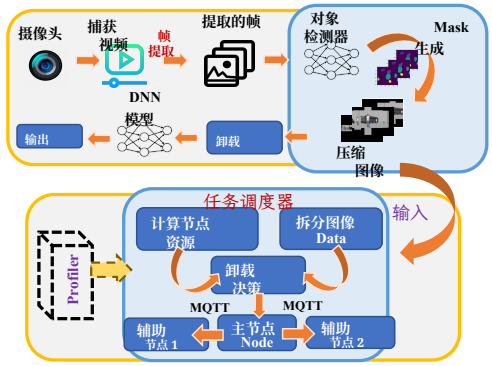


图1：优化框架概览：帧压缩技术与任务调度流程。

B. 系统假设

在此背景下，我们假设，与处理能力更强的服务器或计算机相比，资源受限设备的处理能力和内存资源有限。当同时运行多个DNN模型时，预计会消耗更多能量，并可能出现性能下降。

在这些设备上并发执行多个 DNN 模型时，模型推理也可能成为一个问题。此外，由于无人地面车辆处理不断增加的 DNN 模型数量或日益增长的复杂度的能力有限，因此会产生可扩展性方面的顾虑。该系统假设存在多种处理能力、内存容量和能耗特征各不相同的无人地面车辆。这些无人地面车辆通过通信网络连接，从而能够在设备之间进行数据卸载和通信。性能分析引擎可以准确测量内存利用率、功耗和推理时间等性能指标。该系统假设无人地面车辆可能处于运动状态，导致它们之间的距离发生变化，并影响通信延迟。

IV. *HeteroEdge* 性能分析引擎

在我们提出的系统中，各节点持续监控多DNN工作负载下的系统变量，以确定最优协作配置。我们首先介绍用于分析异构系统中多DNN工作负载的测试平台设置，然后展示对各类设备和网络属性进行分析所得的定量见解。

A. 测试平台设置

我们考虑一个由两个异构边缘平台组成的网络，它们类似于一对资源异构的自主系统：(i) 一台资源较少的 Jetson Nano，配备四核 ARM Cortex-A57 MPCore 处理器、4GB LPDDR4 内存和 128 核 NVIDIA Maxwell GPU；(ii) 一台 Jetson Xavier，配备八核 NVIDIA Carmel ARM v8.2 CPU、8GB LPDDR5 和 512 核 Volta GPU。在所有实验中，我们假设低端设备（即 Nano）会持续监控系统参数，以便将其工作负载卸载到性能更强的设备上，从而执行多个深度神经网络，为下游应用提供支持。图 2 展示了测试平台的实验设置。Jetson Xavier 设备固定在一个位置，而搭载 Jetson Nano 的无人地面车辆则以不同角度和速度移动，以模拟各种移动条件。我们采用发布者-订阅者架构 [16] （具体来说，是消息队列遥测传输协议 (MQTT) [17] ）在两台设备之间传递消息。

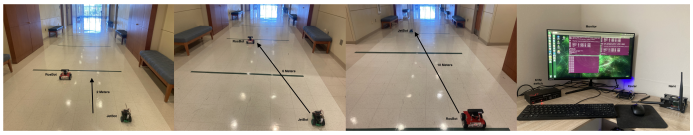


图 2：测试平台的实验设置：(a) 2 米距离下的无人地面车辆设置 (b) 6 米距离 (c) 10 米距离 (d) 静态设备（Nano-Xavier）设置。

B. 设备分析

HeteroEdge 性能分析引擎在主节点和辅助节点上运行，持续记录两台设备的内存利用率、功耗和推理时间。而在我们的

表 I：语义分割和姿态估计模型在测试平台上的性能分析结果

r（拆分比例）	T_1 (Xavier) (s)	P_1 (Xavier) (w)	M_1 (Xavier) (%)	1-r	T_2 (Nano) (s)	T_3 (卸载延迟) (s)	P_2 (Nano) (w)	M_2 (Nano) (%)
0	0	0.95	10.2	1	68.34	0	5.89	69.82
0.3	8.45	4.59	36.67	0.7	39.03	0.43	5.35	63.77
0.5	13.88	5.42	45.61	0.5	28.35	0.89	5.63	52.54
0.7	16.64	5.73	51.23	0.3	19.54	1.25	4.75	45.58
0.8	17.24	6.17	56.96	0.2	13.34	1.44	4.48	40.34
1	19.001	6.38	59.37	0	0	1.56	0.77	16

实验中，为简化起见，我们将低资源设备（即 Jetson Nano）视为主节点，将 Jetson Xavier 视为辅助节点；但实际上，网络中的所有节点都可以担任主节点或辅助节点。*HeteroEdge* 使用 Jetson Stats [18] 测量内存利用率和平均功耗。

深度神经网络工作负载。 由于*HeteroEdge* 面向需要运行多个并发计算密集型任务的自主系统，我们使用多进程池运行两个示例任务，即语义分割和姿态估计。*HeteroEdge* 利用 Nvidia Jetson InferenceLibrary [19] 对各种深度神经网络模型进行分析，以优化卸载决策。

拆分比例（r）。在我们的工作中，我们提出了**拆分比例**，表示卸载到辅助节点的图像所占比例。其范围为 0（所有图像均在本地处理）到 1（所有图像均卸载到辅助节点）。最优拆分比例可在最大化协作系统吞吐量的同时，最大限度地降低资源消耗。符号 $T1$、 $P1$ 和 $M1$ 分别表示辅助节点的运行时间、功率和内存使用量，而 $T2$、 $P2$ 和 $M2$ 则表示主节点对应的指标。$Offlatency$ 表示图像卸载产生的网络延迟。

在表 I 中，我们报告了两台设备在不同配置下处理一批 100 张图像的实测性能—— r 范围从 0 到 1。正如预期，我们观察到，尽管节点的总体功耗相近，但在相同工作负载下，辅助设备的处理延迟显著更低。例如，在 $r = 0.5$ 下，主设备的处理时间（≈ 28.35 秒）是辅助设备（≈ 13.88 秒）的两倍，而两者的功耗相近，分别为 5.63 W 和 5.42 W。同时，我们还注意到，卸载延迟的变化极小（介于 0 到 1.56 秒之间），且 r，这支持了我们关于智能卸载的前提。

C. 网络特征分析

此外，我们研究了两种不同网络配置下的网络延迟，具体而言，是使用两个不同频段（2.4 GHz 和 5 GHz）的 WiFi。在图 3 中，我们绘制了延迟（y–轴）与以下因素之间的关系：（a）不同大小的图像；（b）不同的拆分比例；以及（c）主设备和辅助设备之间的距离（ x–轴）。我们注意到，较高的频段具有更低的延迟，并且随着拆分比例和设备间距离的增加，延迟也会相应增加。

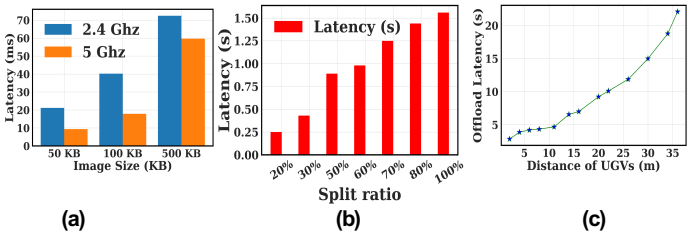


图 3：不同网络频段和图像尺寸下的 MQTT 延迟：(a)；不同拆分比例下的延迟：(b)；不同距离及无人地面车辆速度下的延迟：(c)。

在下一节中，我们将介绍 *HeteroEdge* 求解器，该求解器考虑经过性能分析的变量，以输出适用于视觉任务的最优协作系统。

V. *HeteroEdge*求解器

我们设计了 *HeteroEdge* 求解器，使其能够根据可用资源动态调整数据拆分比例，从而优化整体吞吐量和资源利用率。这种经济高效且可扩展的解决方案适用于各种边缘计算场景，能够应对资源限制和能量约束，同时以分布式方式高效处理大量数据。在以下小节中，我们将介绍构建优化框架的步骤。

A. 延迟和能量建模

在表 II 中，我们列出了优化公式中使用的变量。

表 II：重要符号及其含义。

符号说明	含义
I	计算任务的输入大小
N	执行所需的 CPU 周期数一位计算数据
C_{cpu}	完成特定任务所需的总周期数
T_{exec}	总执行时间
S	客户端设备的计算速度
E_{exec}	总执行能耗
B	传输带宽
D_R	数据速率
T_0	卸载延迟
r	拆分比例
T_s	运行比例卸载代码所需的时间
E_0	卸载能耗
E_s	运行比例卸载代码所需的能量

1) 执行周期： I 表示卸载设备的计算任务输入大小。N 表示执行单位输入计算数据所需的 CPU 周期数。因此，$C_{cpu} = NI$ 表示完成所选计算任务所需的周期数。所以，执行时延可表示为 $T_{exec} = \frac{C_{cpu}}{S}$，其中 S 是设备的计算速度，单位为周期/秒。

我们将 CPU 的功耗建模为 $P = \mu(S)^3$，如 [20] 所示。因此，每个周期的能量消耗为 $\mu(S)^2$。深度模型处理的能耗则为 $E_{exec} = C_{cpu}\mu(S)^2$ ，其中 μ 是一个取决于

芯片架构的系数。我们假设服务器 CPU 的计算速度上限为 S^{max}，因此可能有 $0 \leq S \leq S^{max}$。考虑到 r 是拆分比例，

$$E_{exec} = E_1 r + E_2(1-r)$$
$$T_{exec} = T_1 r + T_2(1-r)$$

这里，E_1 和 E_2 是参与完成一项任务的两个节点的执行能耗。类似地，T_1 和 T_2 是两个设备在特定拆分比例下的执行时间。

2) 卸载时段：如果 B 是传输带宽，d 是两个设备之间的距离，e 是路径损耗指数，N_0 是高斯噪声功率，则可使用香农-哈特利算法 [21] 计算卸载任务的传输数据速率。

$$D_R = B\ \log_2(1 + \frac{d^{-u} P_t}{N_0})$$

这里，P_t 是设备在卸载过程中的传输功率。如果介质无损，则我们可以令 $u = 0$。

然后，卸载延迟为 $T_o = \frac{C}{D_R}$。这里，C 将取决于所选的拆分比例。

总时延可表示为 $T = T_{exec} + T_o + T_s$。运行拆分比例选择代码所需的能量为$E_s = P_k T_s$，其中 P_k 是运行求解器代码的设备的额定功率。

卸载能耗需求可分为两部分，

$$E_0 = T_0 \sum_{i=0}^{N} P_i$$

其中，P_t 表示发送设备所需的功率，P_r 表示接收所发送数据时消耗的功率。因此，总能量可表示为 $E = E_{exec} + E_s + E_o$。

3) 求解器：由于我们的目标是在最小化延迟（即提高整体吞吐量）的同时，使系统兼顾内存和能量，因此我们推导出了执行过程中能量与内存之间的关系。具体而言，我们考虑了执行过程中能量与所需内存之间的二次关系。在求解优化问题时，我们可以采用变量替换方法。对于次优解，我们只需考虑以下方程的实部。

$$T = r(T_1 + T_3) + (1-r)T_2$$

这里，T1 是 Jetson Xavier 的操作时间，T_2 是 Jetson Nano 的操作时间。T_3 是图像传输的往返时间。

$$T_1 = a_1 r^2 + a_2 r + c_1 \quad (1)$$
$$T_2 = b_1(1-r)^2 + b_2(1-r) + c_2$$
$$E_1 = a_1 r^3 + a_2 r^2 + a_3 r + c_1 \quad (2)$$
$$E_2 = b_1(1-r)^3 + b_2(1-r)^2 + b_3 r + c_2$$
$$M_1 = a_1 r^2 + a_2 r + c_1 \quad (3)$$
$$M_2 = b_1(1-r)^2 + b_2(1-r) + c_2$$

这里，$E1$和 $E2$ 分别表示 Jetson Xavier 和 Nano 的能耗。$M1$ 和 $M2$也分别表示所需的内存 Jetson Xavier 和 Nano 的数值。a_1、a_2、a_3、b_1、b_2、b_3系数可通过使用一些实验值进行曲线拟合得到。问题建模，

min(T) 个约束：

$$C1: T \leq \frac{\tau}{k}, \qquad C2: 0 \leq P_k \leq P^{\max},$$
$$C3: 0 < r < 1, \qquad C4: 0 \leq S \leq S^{\max},$$
$$C5: E^k_{\text{exe}} \leq W^k, \qquad C6: M^k_{\text{exe}} \leq M^k. \quad (4)$$

这里，τ 表示整个操作由单个设备完成时的执行延迟，k 表示设备总数。W^k和 M^k分别表示各个设备的最高功率额定值和内存额定值。

4) 电池与充电约束：我们根据剩余电池续航时间（即需要充电前的剩余时间）设定了额外约束，以专门适用于我们的移动自主系统。为此，我们考虑了两款无人地面车辆：RosBot[1] 和 JetBot[2]，它们的电池容量为 4000 mAh，放电率为 70%。它们运行在与我们此前分析中所考虑的平台类似的计算平台上。无人地面车辆的行驶时间约为 20–25 分钟，在此期间会消耗约 15–20 瓦的功率。在机器人上额外运行一个深度神经网络模型会进一步降低其功率容量。如果我们的深度神经网络模型运行 50–60 秒，将持续消耗 5 到 6 瓦的功率。公式 5 和 6 展示了在这些额外约束下可用功率的计算方法。

因此，优化还需满足额外的功率阈值约束；当可用功率 (P) 低于阈值时，无人地面车辆便会更积极地向辅助系统卸载（请注意，运行并发深度神经网络的功耗高于卸载图像的功耗，见表 I）。这确保无人地面车辆能够完成任务而不会耗尽电量。

$$E_{available} = C_0 k - E_{dnn} - E_{drive} \quad (5)$$

$$P_{available} = E_{available} / \left((1-k)\left(t_{dnn} + t_{drive}\right) / 3600 \right) \quad (6)$$

其中，$E_{available}$ 表示可用能量，k 表示电池的放电率，而 E_{dnn} 和 E_{drive} 分别表示运行深度神经网络模型和驾驶无人地面车辆的能耗。t_{dnn} 是深度神经网络模型的运行时长（秒），t_{drive} 是无人地面车辆的驾驶时长（秒）。通过曲线拟合，我们将内存和功率表示为推理时间的函数。通过经验分析，我们发现二次函数的拟合效果最佳，其调整后 R^2 值分别为0.976和0.989。

5) 移动性约束：为了应对其中 的动态场景下的卸载延迟，我们引入一个卸载延迟最小阈值，记为 β。该阈值通过考虑无人地面车辆不断变化的运动，帮助更高效地管理卸载过程。

[1]https://husarion.com/manuals/rosbot/ [2] https://jetbot.org/master/

为了计算无人地面车辆之间的距离，我们采用以下公式：

$$d = (V_{\text{primary}} + V_{\text{auxiliary}}) \times t$$

该公式根据两辆无人地面车辆的速度 V_{primary} 和 $V_{\text{auxiliary}}$ 以及给定的时间间隔 t，计算它们之间的距离 d。公式考虑了无人地面车辆的相对运动；在此时间间隔内，随着两车相互远离，距离会增大。两辆无人地面车辆之间的延迟（L）与距离（d）的关系通过以下公式建模，该公式由曲线拟合得出：

$$L = a_1 \times d^2 - a_2 \times d + a_3$$

该公式表示从一辆无人地面车辆向另一辆发送图像时的时间延迟（延迟）。随着设备之间的距离 d 增大，延迟 L 也会增加，从而影响卸载过程的效率。

当延迟达到或超过阈值 β 时，系统将停止发送数据：

$$\text{If } L \geq \beta, \text{ stop sending data}$$

这种方法可确保卸载过程能够适应无人地面车辆运动的动态变化，从而提高资源利用率和整体性能。

B. 实现

我们使用 Python 的 GEKKO 库 [22] 来求解优化问题。目标函数、变量和约束均按照前文所述进行设定，并使用非线性优化求解器（即 IPOPT 求解器 [23]）。

算法 1 拆分比例选择算法**要求：** 已连接设备的 IP 地址 n，节点的内存配置文件 M，设备的推理时间 1 T_1，设备的推理时间 2 T_2，往返时间 T_3**确保：** 确定实现最优运行时间的拆分比例 r 1：在主节点上：2：根据两个设备的内存计算设备可用性因子 λ 。**计算** 通过曲线拟合从方程 3 中得到的系数 a_1,a_2, b_1, b_2, c_1, c_2 3：**如果** M_1, $M_2 \geq \lambda$ 并检查延迟， $L \leq \beta$ **则**4：根据方程 3 为以下目标函数分配约束：5：检查电池容量和可用的无人地面车辆功率：5 和 6$P_{available} = E_{available}/$ ((1 − k) ($t_{dnn} + t_{drive}$) /3600) 如果 $P_{available} >= E_{max}$6：使用内点优化器方法，在给定约束条件下求解所构建的问题。7：向订阅节点发送推导出的数据量。

VI. 帧级压缩

为进一步优化性能，*HeteroEdge* 会在运行下游深度神经网络之前，从图像中提取感兴趣区域推理（例如姿态检测、图像分割等）。此步骤可确保在可行时，将这些推理任务卸载到辅助节点所产生的网络延迟也能降低。*HeteroEdge* 首先使用一种先进的目标检测模型生成二值掩码，其中检测到的对象所在像素标记为 1，其余像素标记为 0。将二值掩码与原始图像逐元素相乘，可得到一张压缩图像（见图 4），从而分离出感兴趣的对象并去除多余的背景。

为了展示带宽利用率和推理时间方面的节省效果，我们使用两个下游 DNN模型开展了微基准测试实验。我们在 Gazebo 仿真器中生成了一个虚拟环境 [24] ，其中生成了 3100 张图像，共包含 9 类常见物体，例如行人和车辆。首先，我们使用 faster-RCNN 目标检测器 [25]生成压缩图像，然后将这些图像输入典型的下游 DNN 模型，分别用于语义分割（SegNet [26]）和姿态检测（PoseNet [27]）。图 4 展示了针对这两项任务，在压缩帧上生成的输出结果示例。总体而言，我们观察到总计算时间减少了 13%（在 Jetson Nano 设备上），同时带宽消耗最多可降低 28%（即通过压缩将数据量从 8 MB 降至 5.8 MB），前提是这些图像需要卸载传输。尽管压缩仅导致推理准确率下降了 2%，但细心的读者会注意到，不完善的目标检测器模型可能会为下游计算机视觉任务引入伪影；我们将深入研究这一问题留待未来工作。

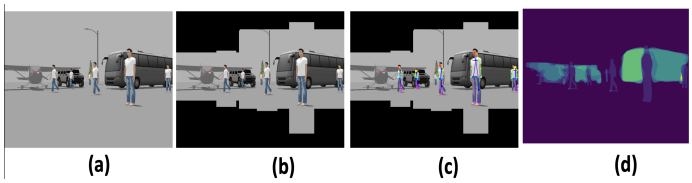


图 4：(a) 原始帧 (b) 压缩帧与输入 (c) 压缩帧的姿态估计结果 (d) 压缩帧的语义分割结果。

VII. 评估

本节对我们在前文第 VI 节中描述的基于 Gazebo 的数据集上提出的优化框架进行了深入评估。

A. 推导优化约束

我们推导了公式4中提出的优化约束，该公式给出了在优化过程中必须满足的不同资源的限制和要求。我们考虑推导重要资源的约束，即内存、功率和推理时间，以优化任务卸载过程。

通过在主节点上进行的实验，我们获得了在单个设备上运行多个模型的基础处理时间。处理 100 张图像的总时间为 68.34 秒，如表 I 所示。我们获得了 200

表 III：静态条件下实时系统的结果。

r（拆分比例）	T3（卸载延迟）（s）	P1（Xavier）(w)	M1（Xavier）(%)	1-r	T1+T2 (s)	P2（Nano）(w)	M2（Nano）(%)
0.2	.67	4.87	32.09	0.8	55.38	6.96	75.12
0.35	1.23	5.12	41.56	0.65	51.89	6.11	70.17
0.45	1.98	5.78	49.55	0.55	42.87	6.24	65.66
0.5	2.34	5.57	50.09	0.5	43.09	5.69	54.65
0.6	2.90	6.35	53	0.4	39.45	5.88	57.77
0.7	3.23	6.03	59.56	0.3	36.43	5.17	47.13
0.8	3.55	6.34	63.45	0.2	34.90	5.35	43.34
0.9	3.56	7.12	69.09	0.1	28.23	4.89	40.11

张输出图像，这些图像由两个模型根据 100 张输入图像生成。我们将其作为推理时间的基线，其中优化后的总体推理时间必须少于 68.34 秒。正如第五节所述，我们使用曲线拟合分析推理时间与数据拆分比例、内存与拆分比例以及功率与拆分比例之间的关系，这些关系分别对应方程 1、2、3。这样，我们便能够预测不同拆分比例下的推理时间，并确定在考虑任务卸载过程中内存和功耗约束的情况下，可实现最低推理时间的最优数据拆分比例。图 5(a) 和图 5(b) 展示了我们提出的*HeteroEdge* 求解器针对不同拆分比例所对应的时间、内存和功率。

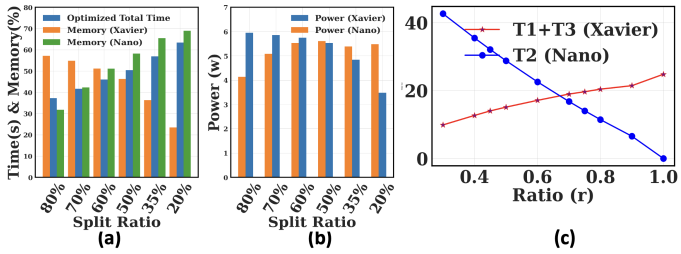


图 5：不同拆分比例下的优化结果。(a) 不同拆分比例下的总时间和内存使用量，以及 (b) 两台设备的功耗；(c) 两台设备的计算时间随拆分比例 r 的变化。

通过求解器，我们得到的最佳拆分比例为 70%，同时满足预期的内存和功耗约束。对于这一拆分比例，Xavier（70 张图像）的总推理时间为 17.72 秒，Nano（30 张图像）的总推理时间为 16.79 秒。对于两个模型和 200 次运行，平均总耗时为 34.51 秒。

B. 实证评估

我们对静态和动态条件下的场景进行了实证评估。以下将通过两个案例研究讨论静态和动态条件。

案例 1： 在此案例中，两辆无人地面车辆彼此相距固定，且速度相同，因此它们之间没有相对运动。由于在我们的实验中它们仅相距 4 米，整个评估过程中通信开销和延迟都保持不变。在 MQTT 通信稳定的情况下，卸载图像从主无人地面车辆传输到辅助无人地面车辆时，会因两者之间的固定距离而产生稳定的延迟。这种恒定的延迟使卸载过程能够有效进行，且不会因距离变化而产生额外的通信开销。随后，可以在此静态条件下评估优化框架，以了解其在延迟变化极小的受控环境中的性能。

如表 III 所示，在静态条件下，实时测试平台上针对不同拆分比例测得的性能指标与求解器得出的优化结果一致。这里，$T_1 + T_2$ 表示 Jetson Nano 和 Xavier 的总运行时间，T_3 表示无人地面车辆在静态条件下相距 4 米时的卸载延迟。从结果中可以看出，拆分比例变化时，卸载延迟略有变化。在这种情况下，只有拆分比例较高时卸载延迟才会增加，而无人地面车辆之间的距离保持不变。

使用 IPOPT 优化求解器后，我们估计，在特定的内存与功耗约束下，将 70% 的图像卸载到性能更强的 Xavier 上是最佳选择。随后，我们在实时系统中评估了结果（表 III），并在测试平台上测试了不同的拆分比例，最终得到了预期结果，这些结果支持我们的优化框架。总的来说，这证明了我们的内存与功耗感知任务卸载框架的有效性，也展现了进一步优化的潜力。

情况 2： 在此情况下，两辆无人地面车辆以不同的速度和/或方向运动，导致它们之间的距离不断变化。随着两辆无人地面车辆之间的距离随时间变化，将图像从主节点卸载到辅助节点所需的通信开销和延迟也会随之变化。由于此场景具有动态特性，MQTT 通信的延迟可能会随着两辆无人地面车辆之间距离的变化而波动。这可能会给优化框架带来挑战，因为卸载过程可能需要考虑通信开销和延迟的变化。可以在此动态条件下评估优化框架，以了解其在两辆无人地面车辆之间距离不断变化的真实环境中的性能。这有助于发现潜在问题，并确定可能需要进行的调整，从而提高框架对不同通信条件的适应能力。

在图6中，我们记录了两者的总运行时间（$T_1 + T_2$）

表 IV：测试平台中不同已部署模型的概览。

应用	深度神经网络模型	拆分比例（r=0） 100% 在以下节点运行 主节点 T2（Nano）（秒） （原始图像）	拆分比例(r=0) T2（Nano） (s) （掩码图像）	拆分比例 (r=0.5) T1（Xavier）+ T2（Nano） (s) （原始图像）	拆分比例（r=0.5） T1（Xavier）+ T2（Nano） (s) （掩码图像）	拆分比例 (r=0.7) T1(Xavier)+ T2（Nano） (s) （原始图像）	拆分比例（r=0.7） T1（Xavier）+ T2（Nano） (s) （掩码图像）
图像识别+ 目标检测	ImageNet, DetectNet	74.68	69.90	56.74	49.78	44.13	38.98
目标检测+ 深度感知	DetectNet, DepthNet	76.90	71.34	64.2	57.89	43.17	40.32
语义分割 + 深度感知	SegNet, DepthNet	71.25	65.56	58.43	53.66	48.37	43.2
图像识别+ 深度感知	ImageNet, DepthNet	69.66	61.47	50.64	46.45	43.54	38.43
目标检测+ 姿态估计	DetectNet, PoseNet	67.28	64.89	51.59	46.89	39.69	35.9

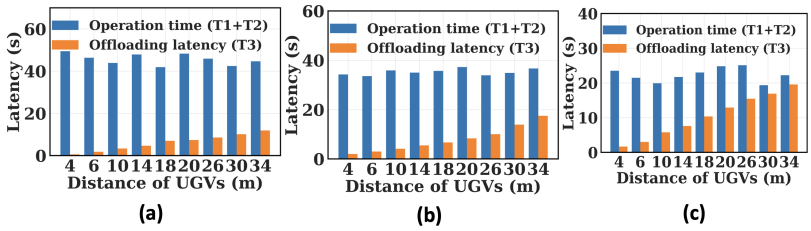


图6：测试平台的实验设置用于评估系统在不同距离下的性能。(a) 30% 拆分比例 (b) 70% 拆分比例 (c) 100% 拆分比例。

无人地面车辆及卸载延迟 T_3 随无人地面车辆间距离变化的情况。在此设置中，我们采用三种不同的拆分比例：30%、70% 和最坏情况下的 100%；主无人地面车辆和辅助无人地面车辆的速度分别为 $V_{primary}$=1 m/s 和 $V_{auxiliary}$=3 m/s。评估结果表明，无人地面车辆之间的距离与卸载延迟呈正相关。随着距离增加，卸载延迟也会随之上升，从而影响优化系统的有效性。例如，在距离为 26 米时，从主节点到辅助节点的平均卸载延迟为 13.9 秒，这增加了通信开销，并损害了系统性能。

为应对这一挑战，我们提出了卸载延迟阈值。系统可以在运行过程中有效地跟踪卸载延迟。如果延迟超过阈值，主节点便会停止向辅助节点卸载图像，并寻找一个低于先前值的、更合适的拆分比例。如果无法在限定范围内找到最优拆分比例，主节点就会在本地执行所有处理任务。这种自适应方法通过最大限度地减少通信开销并避免过长的延迟，维持最佳性能。

C. 模型异构性评估

为了全面评估*HeteroEdge*的性能，必须使用多种代表不同用例和应用的模型对其进行验证。为此，我们精心挑选了一系列模型，涵盖从目标检测、图像分类到深度估计等不同计算需求。作为示例，我们选择了用于目标检测的计算机视觉模型 ImageNet[28] 、用于目标检测的 DetectNet 和用于单目深度估计的 DepthNet [30] 。我们将每个模型（包括前面提到的 PoseNet 和 SegNet）部署在*HeteroEdge* 测试平台上，并同时运行这些模型，同时测量总运行时间、功耗和资源利用率等关键性能指标。

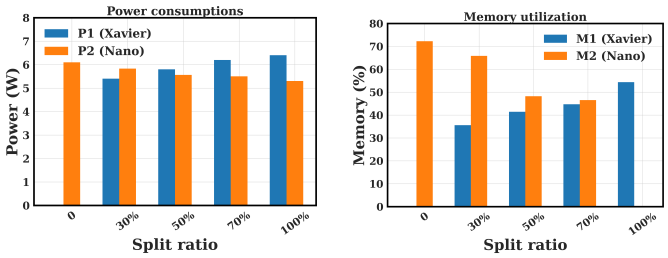


图 7：(a) 平均功耗 (b) 平均内存利用率。

表 IV 展示了多个深度神经网络模型同时在主节点和辅助节点上运行的结果。我们对 100 张图像进行了测试；当拆分比例为零时，如表 IV 所示，所有处理都仅由主节点完成。我们观察到，在拆分比例分别为 0.5 和 0.7 时，主节点分别将 50% 和 70% 的图像发送至辅助节点。值得注意的是，主节点还需要额外运行一段时间来进行目标检测和掩码生成。使用轻量级 faster-RCNN 模型时，我们记录到每张图像的平均延迟为 3–4 毫秒 [25]。我们注意到，与原始帧相比，掩码帧的总运行时间更短（平均缩短 9%）

尽管功耗和内存利用率发生了显著变化。图 7(a) 显示，功耗略有增加：在拆分比例为 = 0 时，与所有处理均在本地完成的基线相比，平均增加 4–5%。相反，图 7(b) 显示，内存使用量较基线内存使用量显著降低，即在拆分比例为 = 0时降低了 ≈ 7223%。例如，在拆分比例为 70% 时，两个设备的平均内存使用量均为 47%，与基线配置相比几乎降低了 34%。

VIII. 结论与未来工作

总之，我们的优化工作聚焦于通过将图像处理卸载到更强大的计算设备上，降低深度神经网络模型推理的延迟。我们提出了逐帧优化的拆分比例指标，用于确定需要卸载的图像比例，并使用求解器根据无人地面车辆的内存和功耗约束确定最优拆分比例。总体而言，我们的工作为降低资源受限设备上的深度神经网络模型推理延迟提供了实用方案。结果表明，使用 MQTT 进行卸载，并根据可用功率动态调整拆分比例，可以进一步降低延迟并提升性能。在本工作中，我们采用了本质上属于分层卸载的主辅节点架构。未来，我们希望扩展当前研究，考虑采用星形拓扑来执行卸载任务。在星形拓扑中，中心节点（“枢纽”）管理多个边缘设备（“辐射节点”）之间的通信与协调，从而实现更高效的资源分配和数据共享。

致谢

本研究部分获得美国国家科学基金会 CAREER 奖 # 1750936 和美国陆军资助 #W911NF2120076。

参考文献

[1] Syed Agha Hassnain Mohsan、Nawaf Qasem Hamood Othman、Yanlong Li、Mohammed Alsharif 和 Muhammad Khan。无人机（UAV）：实际应用、应用领域、开放性挑战、安全问题及未来趋势。*Intelligent Service Robotics*，2023年1月。[2] Pedram Beigi、Mohammad Sadra Rajabi 和 Sina Aghakhani。无人机能耗因素与模型概述。*arXiv* 电子预印本，第 arXiv:2206.10775 页，2022年6月。[3] 无人机新型智能电池管理系统，收录于 *ASME* 国际机械工程大会暨展览会第6卷：能量，2019年11月。V006T06A028。[4] Aakash Khochare、Yogesh Simmhan、Francesco Betti Sorbelli 和 Sajal K. Das。无人机机群任务中边缘分析与路径协同调度的启发式算法。载于 *IEEE INFOCOM 2021 - IEEE* 计算机通信会议，第1–10页，2021年。[5] Juheon Yi 和 Youngki Lee。Heimdall：面向增强现实应用的移动 GPU 协调平台。载于第*26*届移动计算与网络年度国际会议论文集，2020年。[6] Min Xue、Huaming Wu、Guang Peng 和 Katinka Wolter。DDPQN：一种适用于本地—边缘—云协同环境的高效深度神经网络卸载策略。*IEEE* 服务计算汇刊，15(2)：640–655，2022年。[7] H. Chen、W. Qin 和 L. Wang。物联网云边协同计算框架中的任务划分与卸载：综述。*Journal of Cloud Computing*，11(1)，2022年12月3日。[8] Bart Cox、Jeroen Galjaard、Amirmasoud Ghiassi、Robert Birke 和 Lydia Y. Chen。MASA：边缘侧的响应式多深度神经网络推理。载于*2021*年 *IEEE* 普适计算与通信国际会议（*PerCom*），第1–10页，2021年。

[9] Quang-Huy Nguyen 和 Falko Dressler。智能手机视角下的计算卸载——综述。计算机通信，159:133–154，2020。[10] Erol Gelenbe、Ricardo Lent 和 Markos Douratsos。选择本地云还是远程云。载于*2012* 年第二届网络云计算与应用研讨会，第 25–30 页，2012。[11] Joo Seong Jeong、Jingyu Lee、Donghyun Kim、Changmin Jeon、Changjin Jeong、Youngki Lee 和 Byung-Gon Chun。BAND：异构移动处理器上的协同多深度神经网络推理。载于第 *20* 届年度移动系统、应用与服务国际会议论文集，第 235–247 页，2022。[12] Nirmalya Roy、Vasanth Rajamani 和 Christine Julien。支持动态传感器网络中具备多保真感知能力的并发应用。载于*2010* 年第 *8* 届 *IEEE* 普适计算与通信国际会议研讨会，第 43–49 页。IEEE，2010。[13] Woosung Kang、Kilho Lee、Jinkyu Lee、Insik Shin 和 Hoon Sung Chwa。LaLaRAND：面向实时深度神经网络任务的灵活逐层 CPU/GPU 调度。载于*2021* 年 *IEEE* 实时系统研讨会（*RTSS*），第 329–341 页。IEEE，2021。[14] Shengzhong Liu、Tianshi Wang、Hongpeng Guo、Xinzhe Fu、Philip David、Maggie B. Wigness、Archan Misra 和 Tarek F. Abdelzaher。车载实时视频分析的多视图调度，以最大限度降低帧处理延迟。载于第 *42* 届 *IEEE* 分布式计算系统国际会议（*ICDCS 2022*），意大利博洛尼亚，*2022* 年 *7* 月 *10–13* 日，第 503–514 页。IEEE，2022。[15] Shengzhong Liu、Tianshi Wang、Jinyang Li、Dachun Sun、Mani Srivastava 和 Tarek Abdelzaher。Adamask：通过用于深度神经网络推理卸载的自适应帧掩码，实现以机器为中心的视频流传输。载于*MM '22*：第 *30* 届 *ACM* 多媒体国际会议，葡萄牙里斯本，*2022* 年 *10* 月 *10–14* 日，第 3035–3044 页。ACM，2022。[16] Roberto Beraldi 、Sara Tucci-Piergiovanni 和 Antonino Virgillito。关于发布/订阅通信系统的建模。实践与实验，17:1471–1495，2005 年 10 月。[17] Urs Hunkeler、Hong Linh Truong 和 Andy Stanford-Clark。MQTT-S——一种面向无线传感器网络的发布/订阅协议。载于*2008* 年第 *3* 届通信系统软件与中间件国际会议及研讨会（*COMSWARE '08*），第 791–798 页，2008。[18] Jetson Stats。https://https://github.com/rbonghi/jetson stats。访问日期：2023-02-03。[19] A. Kurniawan。《深度学习计算》。https://link.springer.com/chapter/10.1007/978-1-4842-6452-2 7，2020 年 12 月 12 日。[20] Weiwen Zhang、Yonggang Wen、Kyle Guan、Dan Kilper、Haiyun Luo 和 Dapeng Oliver Wu。随机无线信道下能量最优的移动云计算。*IEEE* 无线通信汇刊，12(9):4569–4581，2013。[21] 香农–哈特利定理。https://https://en.wikipedia.org/wiki/ Shannon-Hartley theorem。访问日期：2022-08-06。[22] Logan D. R. Beal、Daniel C. Hill、R. Abraham Martin 和 John D. Hedengren。Gekko 优化套件。过程，6(8)，2018。[23] L.T. Biegler 和 V.M. Zavala。使用 IPOPT 进行大规模非线性规划：企业级动态优化的集成框架。计算机与化学工程，33 582，2009 3 575 17 ( ): – 。选自 2007 年 5 月在罗马尼亚布加勒斯特举行的第 t h 届欧洲计算机辅助过程工程研讨会论文集。[24] Gazebo。https://gazebosim.org/。访问日期：2022-08-06。[25] Ross Girshick。Fast R-CNN。载于*IEEE* 计算机视觉国际会议论文集，第 1440–1448 页，2015。[26] Vijay Badrinarayanan、Alex Kendall 和 Roberto Cipolla。SegNet：用于图像分割的深度卷积编码器—解码器架构。*IEEE* 模式分析与机器智能汇刊，39(12):2481–2495，2017。[27] Alex Kendall、Matthew Grimes 和 Roberto Cipolla。PoseNet：用于实时六自由度相机重定位的卷积网络。载于*IEEE* 计算机视觉国际会议论文集（*ICCV*），2015 年 12 月。[28] Agus Kurniawan。《深度学习计算》，第 107–119 页。Apress，美国加利福尼亚州伯克利，2021。[29] Andrew Tao、Jon Barker 和 Sriya Sarathy。DetectNet：用于数字图像中目标检测的深度神经网络。*Parallel Forall*，4，2016。[30] Arun CS Kumar、Suchendra M. Bhandarkar 和 Mukta Prasad。Depth-net：用于单目深度预测的循环神经网络架构。载于 *IEEE* 计算机视觉与模式识别会议（*CVPR*）研讨会论文集，2018 年 6 月。