

分布式模型缓存赋能端云协同自演进视频分析

廖天俊¹, 赵茆哲², 刘生钟², 吴帆²

1. 军事科学院, 北京 100091;

2. 上海交通大学, 上海 200240

摘要

在移动视频分析领域, 数据漂移是影响系统推理效果的主要问题之一。对此, 提出了一种移动端主导的端云协同持续学习框架。具体而言, 该框架在资源受限的移动设备上部署了模型高速缓存存储器, 并结合云服务器设计了一套“重用、微调、重训练”的分层演进策略。通过构建基于领域属性的前缀树存储结构, 其实现了快速模型查找, 降低了演进中云端的排队时延。评估结果显示, 该框架在目标检测和分类任务中优于现有最佳方案, 并在多端设备系统中取得了最优的整体演进精度。这体现了该框架在提升响应速度和系统扩展性方面的显著潜力。

关键词

端云协同; 视频分析; 边缘计算; 持续学习

中图分类号: TP399

文献标志码: A

doi: 10.11959/j.issn.2096-0271.2025036

Distributed model cache-driven device-cloud collaboration in adaptive video analytics

LIAO Tianjun¹, ZHAO Maozhe², LIU Shengzhong², WU Fan²

1. Academy of Military Science, Beijing 100091, China

2. Shanghai Jiao Tong University, Shanghai 200240, China

Abstract

In video analysis, data drift is one of the main issues affecting system inference performance. To address this, this paper proposes a mobile-oriented edge-cloud collaborative continuous learning framework. Specifically, this framework deploys a model caching on resource-constrained mobile devices and designs an edge-cloud collaborative hierarchical adaptation strategy that integrates “reuse, fine-tuning, and retraining”. Constructing a prefix tree storage structure based on domain attributes enables rapid model retrieval, thereby reducing cloud queuing delays during adaptations. Evaluation results demonstrate that this framework outperforms existing state-of-the-art solutions in object detection and image classification tasks, achieving a 34.5-times reduction in reuse response time and a 1.99-times reduction in retraining time while achieving optimal overall accuracy in multi-device system adaptations. This highlights the framework’s significant potential for enhancing response speed and system scalability.

Key words

device-cloud collaboration, video analytics, edge computing, continuous learning

0 引言

如今,智能视频分析技术已被广泛应用于城市交通管理^[1]、企业安全^[2]、交通监控^[3]、自动驾驶^[4-5]等众多领域,展现出极高的应用前景和实际价值。然而,要完全实现理论构想中的视频分析系统部署,仍需从实验到实际应用的进一步研究。在此过程中,一项不可避免的挑战被称为数据漂移。数据漂移指的是实际数据流中的数据分布与预先训练的数据分布存在偏差。当出现数据漂移时,预训练模型在实际数据中的表现往往达不到预期效果。

针对此问题,当前移动视频分析架构^[6-7]的主流方法是持续学习^[8-9]。这种方法采用了两种不同计算量的模型:专门应用于特定场景但缺乏泛用性的小型“特化模型”和适用于大部分场景但计算开销较大的大型“泛化模型”。在服务器上,泛化模型用于为不同场景数据生成标签,而特化模型通过这些数据和标签进行重训练,使其在特定场景中接近泛化模型的推理水平。完成重训练后,特化模型会被部署到需要它们的端设备上推理。当这种思路合理地部署在系统中时,系统会从一个泛化模型开始,逐渐生成一系列小型特化模型,以应对不同场景,从而在性能上逼近泛化模型,同时在重训练开销上逼近小型特化模型。

这种小型模型的应用将输入的数据流离散化为若干数据片段,每一片段代表一个特定的场景。这些具有特定数据分布的场景被定义为领域,而从一个领域到另一个领域的变化被称为领域迁移。在移动持续学习系统中,有效地检测领域迁移并进行针对性的演进任务以避免系统性能的长

时间下降,是设计的核心问题。

然而,本文研究发现,先前的工作将持续学习方法中所有的模型演进任务集中在云服务器上,这带来了两个难以克服的挑战。首先,云侧模型训练会产生大量由网络传输引发的时延。例如,从端侧数据采集完成到完全传输给云侧服务器之间,以及在云侧完成模型选择到实际加载使用之间,都存在显著的时延。尽管云侧服务器拥有更强大的计算资源,但是这种传输时延削弱了其优势。其次,繁重的重训练任务占用了大量服务器计算资源,集中式的重训练在设备数量增加时使服务器成为系统效率和精度的瓶颈。当设备数量增加、重训练任务过多时,任务将在云端堆积,其导致推理结果与未进行演进任务的系统相当。

为应对这些挑战,本文设计了一套基于端云协同体系的持续学习框架。其核心思想是将模型演进任务分布到各个端侧设备上,以避免端云之间的拥塞,并降低网络传输时延。具体来说,本文工作的研究贡献有以下3点。

- 设计了一个非集中式的模型选择过程:在服务器上维护一个模型库进行共享,并在每个端设备上部署一个模型高速缓存用于模型重用。这种设计显著降低了服务器的处理开销。
- 构建了基于领域语义信息的层次化分类结构:根据领域语义信息特征,将模型库中的所有特化模型组织成一个层次化的分类结构,并实现了一种距离算法,用于快速查找模型的索引。
- 扩展了模型演进策略,引入了端设备单层模型微调:通过在端设备上重新部署模型演进策略,释放了服务器的计算资源,降低了服务器的负载,并避免了云端的排队时延。

1 研究背景与相关工作

持续学习旨在开发能够持续获取、积累和应用知识的算法和模型。传统的机器学习模型在一个固定的数据集上训练完成后即停止学习，而持续学习模型能够在不断接收新数据和新任务时持续进行学习和调整。

在移动视频分析领域，基于持续学习的工作十分普遍。ODIN^[10]从数据层面入手，在云端构筑了一套双重对抗生成网络体系，将对高维数据的分析工作降低到低维空间，并通过对已学习数据进行无监督聚类，达到数据分类和漂移数据检测的效果；在此基础上，通过对新类别漂移数据进行持续学习，生成新模型，完成了整个系统针对动态环境的持续更新。

Ekya^[11]发现了已有的技术在内存分配方面存在提高空间。针对端设备生成的多个任务，系统在云端使用一个资源调度器搭载小偷调度算法，完成了多个模型重训练任务与推理任务的计算资源优化分配，大大提高了整体精度，并显著减少了对计算资源的需求。

EdgeFM^[12]是一种新颖的边缘云协作系统，旨在利用基础模型（foundation model, FM）实现开集学习。其首先在边缘设备上对传感器数据进行初步处理，并通过模型选择模块确定适合的轻量级小型模型架构。然后，EdgeFM会根据当前数据的不确定性和实时网络状况动态决定是否向云端查询FM。当边缘设备无法处理某些复杂数据时，它会将这些数据上传至云端，由云端的FM进行推理，并获取相应的知识反馈。在此过程中，FM会根据用户的兴趣类别和传感器数据的语义特征，

不断优化和定制小型模型，生成伪标签，从而实现无标签的定制微调。定期更新时，EdgeFM会根据收集到的新数据和环境变化，更新云端的文本嵌入池和小型模型，并将这些更新传输到边缘设备。最终，边缘设备可以使用这些定制的小型模型进行高效推理，避免了传统方式中的数据标注和模型重训练工作。

RECL^[13]在使用模型库的基础上，使用了一种将重训练与重用结合的新持续学习策略。其在云端部署了一个定期更新的门网络，用于检测实时数据是否发生数据漂移。在发生漂移时，除了开启传统的模型重训练任务，系统还会使用上传的数据根据大模型的标注结果在模型库中选出最合适的重用模型，并下发到端设备，从而在完成重训练任务前进行一次初步的任务精度提升。这种策略大大提高了从情景产生变化到重训练任务完成之间这一段空窗期的推理精度，从而提高了系统的整体精度和资源利用。

2 端云协同持续学习理论分析

2.1 端云协同持续学习

大量研究表明，持续学习在移动视频分析领域是一种有效的方法。然而，随着研究的深入，持续学习逐渐触及了瓶颈，难以从正面获得突破。在近期的视频分析架构中，研究者更多地探索新的手段以突破持续学习的固有限制。同理，本文从持续学习出发，以移动视频分析领域中端设备的视角来探索持续学习，发掘可以进一步优化的组件。

在资源受限的移动视频分析体系中，主流的视频分析下游任务与模型，例如用于物体识别^[14]的YOLO模型和用于图像分

类的 ResNet 模型，都需要大量的计算资源。同时，有研究表明，一个模型的训练成本平均是模型推理成本的 3 倍。因此，在持续学习系统的端设备上，耗费大量资源进行模型重训练是不现实的。

例如，在本文测试的 NVIDIA Jetson Nano 设备上，计算资源和存储资源都十分有限。它无法支持常用物体检测任务的 YOLOv5-x 模型进行正常推理；而对于 YOLOv5-x 的小型化版本 YOLOv5-s（大模型的 1/12 总浮点运算次数）模型，也仅能以每秒 1.72 张 640×640 尺寸图片的速度进行重训练。但是泛化模型又是不可或缺的。如图 1 所示，泛化模型在多个域中能够保持较高的精度且具有较小的精度波动，但特化模型仅在其训练域中能保持理想的精度。

因此，本文的解决方案依旧将持续学习作为框架的最底层设计，目标是进一步降低持续学习的开销，突破其现有限制，以便在移动视频分析架构中实现更高效的应用。

2.2 重用策略

对于持续学习繁重开销的弊端，最合理的优化方向就是通过类似于“剪枝”的方式剔除或替代部分非必要的重训练部分，采用更轻量化手段来换取可以接受的精度效果。而在这个方向之上，重用展现出了极佳的性能。

从理论上来说，重用脱胎于数据流的时空规律。在实际场景中，数据会具体地存在时空上的规律。对于一台传感设备，类似的领域（即数据分布）很有可能会在一天相同的时间或相同天气下出现。并且，对于多台传感设备，在其中一台传感设备出现的域很有可能会在别的传感设备移动到相似场景时出现。这些普遍的现象说明出现过的域

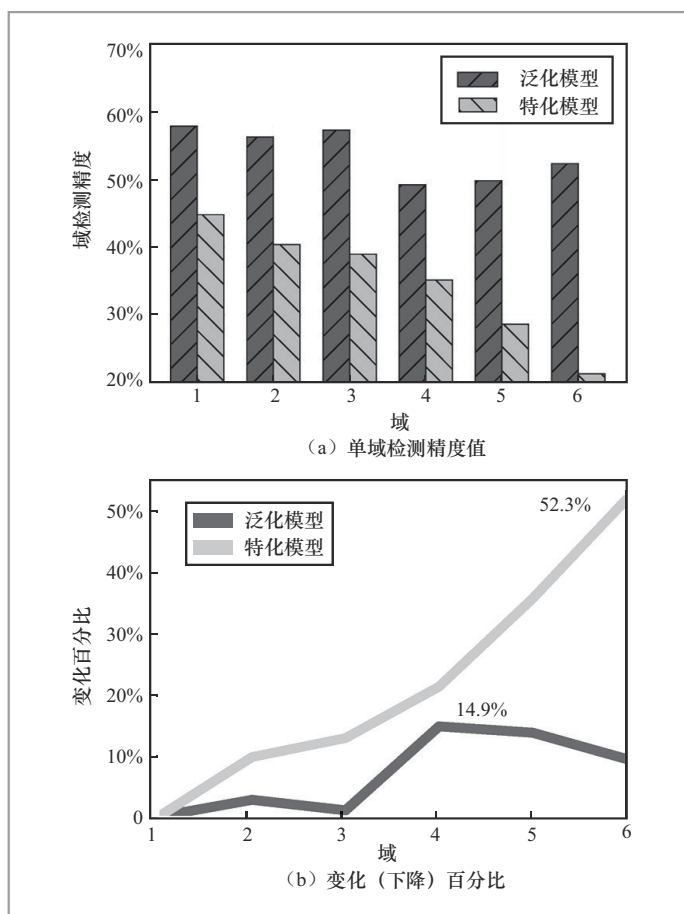


图1 泛化模型和特化模型在不同领域上的效果对比

在未来存在被再次使用的价值。如果系统存储了历史特化模型，当检测到该模型适用的域再次出现时，重用就可以发挥效果，直接取代重训练或者成为重训练的起始权重，使重训练更快地达到精度要求。

在这方面，本文进一步指出，重用并不仅在相同域的重复出现中有效。在相似域中，对模型的重用和重训练起点选择仍然是一种有效且高效的选择，这已在图 1 中得到了体现：除了自身的训练领域，特化模型在相邻几个域的数据中也保持了一定的精度。从本质上来看，带有数据漂移的同一域和两个相似的域实际上是类似的情况。当数据漂移足够大时，漂移前后的同一域就变成了两个相似的域。“领域迁

移”这一概念将连续数据流离散化，将二者原本模糊的边界过分地清晰化了。

同时，本文考虑了重用中数据传输问题带来的响应效率降低。如图2所示，在不高的带宽中，传输占到整个过程的50%以上；在最坏的情况下，传输可能使重用的响应时间拉长4倍。为此，笔者重新评估并认为重用最佳的使用部件为端设备，并创见性地提出如下设计理念。

2.3 层次化设计

在前文的基础上，本文提出设计一个

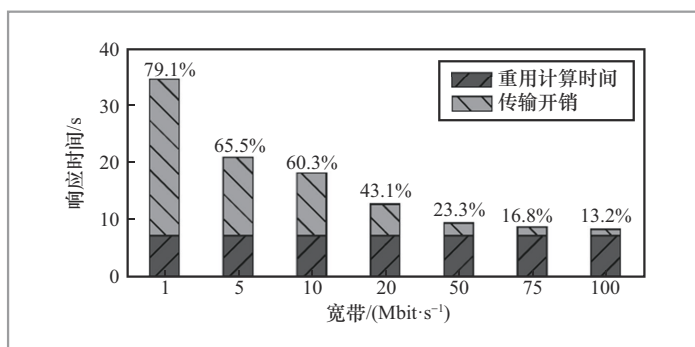


图2 重用中传输开销

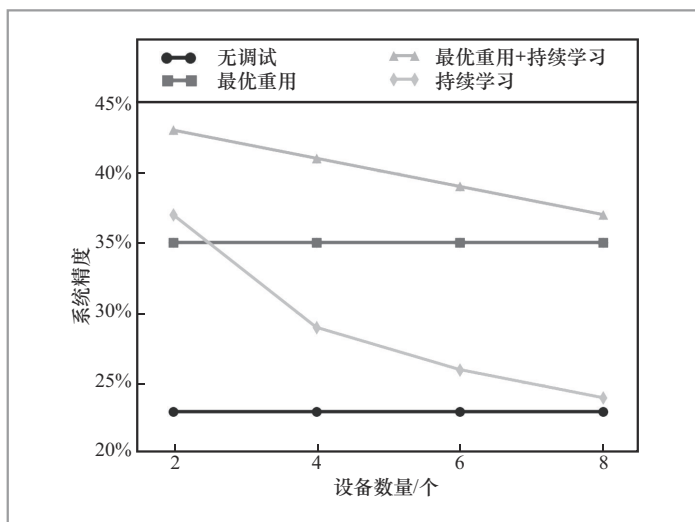


图3 排队时延带来的多设备系统精度下降

层次性的模型演进系统。这个“层次性”可以从两个维度来解释。首先，模型的重训练任务能够被层次化地定义：从模型重用、对少数层的快速微调，再到完整重训练，逐步提升泛用性，同时缩短响应时间。其次，模型演进能够在移动端设备和云服务器上协同进行，以实现响应效率和演进效果的平衡。

使用模型重用来降低模型重训练频率已经被 RECL 重点强调过。本文进一步结合了轻量级微调，形成了3种演进手段：重用、微调和重训练。轻量级微调仅针对部分层进行微调，相比完整重训练消耗更少的资源。在无法直接重用时，可以快速进行轻量级微调，以获得一个折中的、较快且较好的模型，从而为整体的模型演进工作提供更大的操作空间。

此外，移动设备的存储和计算能力在模型适配中尚未得到充分探索。直观地说，移动设备与云资源之间的关系可以类比为现代计算机系统的分层存储。移动设备上有限但快速访问的资源（类似于缓存），而云服务器上有丰富但距离较远的资源（类似于主存储器）。从存储方面来看，在移动设备上保存一些常用的特化模型可以避免网络传输时延和云端排队时延（如图3所示）。而从计算方面来看，在移动设备上分配轻量级的微调任务，同时将重训练任务集中在云端，可以在特化模型更新时平衡响应时延和处理吞吐量。如果要重用的特化模型已经缓存在设备上，系统可以在很短的时间内直接加载它，而不必等待云服务器的分发，从而提高整体响应速度。同时，对于更多的端设备和更多的数据传输，相同的工作端设备并不会显著增加时间开销，而先前的工作中云服务器会因为排队时延增加开销。

2.4 语义索引

实现上述层次模型演进的关键步骤之一是高效地从所有历史特化模型（即模型库）中检索出适合新域的最佳候选模型。现有方法往往通过评估基准数据集遍历所有模型，或者维护一个特化模型选择的门网络。然而，这些方法都存在一些缺陷，比如需要在模型库变化时频繁重新训练，或者忽略了重要的易识别语义信息^[15-16]。

因此，为了提高模型检索和更新的效率，可以利用域的语义信息^[17]将历史特化模型有机地组织成一个结构化的数据库，而不是简单地维护一个无序列表。这样的结构化数据库能够更好地反映部署环境的特征，例如时间、天气、位置等。将模型与特定的语义信息相关联，可以更准确地定位和检索与当前环境最匹配的模型，从而提高模型选择的效率和精度。

利用强大的多模态基础模型^[18]可以轻松地以零样本的方式识别语义信息。这些基础模型是在广泛数据集上预训练的大规模机器学习模型，可以通过微调演进各种特定任务。现有的基础模型，如 LLaVA，可通过连接图像输入和文本描述（提示文本）来理解多模态输入。在不进行微调的情况下，预训练的基础模型在检测输入数据的领域时已经提供了有保障的准确性。基础模型在每张输入图像（大小为 640 像素 × 640 像素）上花费 500 ms。用户只需要预先定义领域语义维度及其值选项，这是一个难度相对较低且一次性的元级别工作。

3 视频分析系统具体设计

3.1 整体结构

基于先前的理论分析，本文设计了端

云协同持续学习的具体解决方案。它拥有一个较强计算资源和存储资源的云服务器和一系列有限资源和网络带宽的端设备（例如 NVIDIA Jetson Nano、NVIDIA Jetson TX2）。本系统能够在移动视频分析任务（图像分类、物体识别等）中运行环境领域迁移大量产生的情况下，保持良好的推理精确性。为了优化响应效率和端设备拓展性，其将低计算量的实时演进工作（直接重用和快速微调）下放到了端设备中，而云服务器仅进行低频次、高计算量的计算更新工作（完成重训练和标注）。

在有限移动计算资源和网络传输时延要求下，本系统更新过程的核心任务包括：①确定域迁移发生时间；②确定端侧缓存中特化模型更新策略；③提高系统响应力（降低短时间更新的响应时间）；④提高系统拓展性（使服务器支持更多端设备接入）。

为此，本系统在移动设备上进行了领域迁移检测，并基于各种实时状态进行端侧缓存中特化模型的轻量级模型重用和快速微调（仅在模型的最后一层进行更新）。而在服务器上，本系统将精确的目标模型检索或者完整的模型重训练作为后端支撑。本系统的表现高度取决于高效高速的历史模型检索。不同于 RECL 中使用一段小型基准数据对无序模型库进行检索来获得重用模型的策略，本文通过在元信息层面进行领域语义标记（如时间、天气、位置）来构建一个有序模型库分类表，使本系统能够比 RECL 更快地完成实时模型检索。

图 4 展示了端设备和云服务器上的主要部件。在端侧，除了对于持续数据流的常规模型推理部分，移动设备周期性地检测域迁移的出现并完成快速模型重用和微调。为达到目的，每个端设备由端侧模型演进模块、端侧特化模型缓存、推理模块、

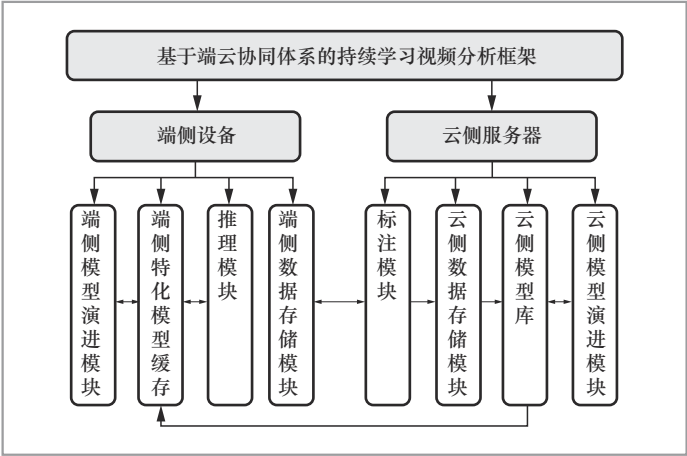


图4 解决方案整体架构

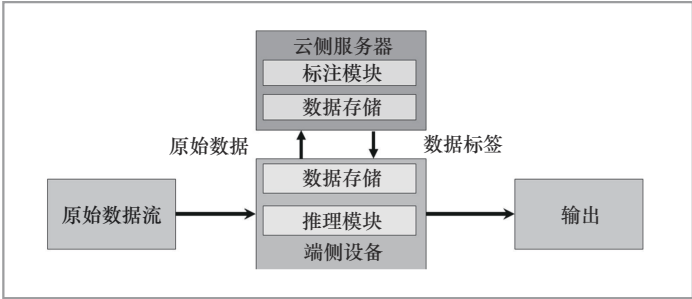


图5 数据交换流程

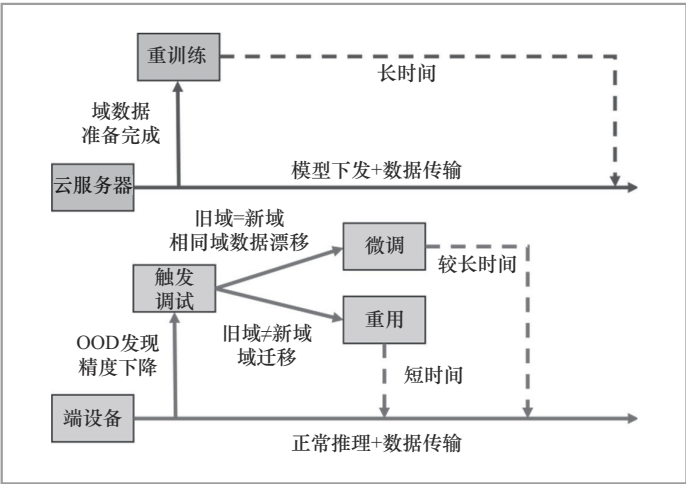


图6 演进工作流程

端侧数据存储模块4个部分构成。在云侧，云服务器在后端进行重量级（相较于移动端设备）存储和计算任务来协助端设备部件。它维护了一个模型库来管理来自不同端设备的模型，并存储了历史域的历史数据用于重训练。此外，它维护了大模型来标注端设备上传的不同域的数据。为达到目的，云服务器部署了标注模块、云侧数据存储模块、云侧模型库、云侧模型演进模块4个部分。

3.2 协同模式

因为视频流是持续采样的，可以认为域迁移发生的频率远小于采样频率（例如25 FPS），并且相邻帧之间不会存在复数的域迁移现象。因此，时间被划分成了固定长度的时间窗口（默认为30 s），并且将每个时间窗口设定为进行域迁移检测和演进的最小单元。这个设定和先前的研究工作相似。依据任务时间维度和性质来划分，本系统分为两个部分进行工作。

第一部分为数据交换，它比较简单但始终贯穿整个系统，是本系统工作的基础，如图5所示。

第二部分为演进工作，其包含了3种任务，分别为重用、微调 and 重训练，如图6所示。

从设备层次来讲，本系统分为端设备和云服务器两部分协同工作。作为一个移动端主导的系统框架，其大多数工作在移动端设备上被启动或确认，而云服务器仅简单地服务来自端设备的请求。在每一个时间窗口中，移动端设备的主要任务包括：①使用部署的特化模型在每一帧上进行常规推理任务；②在推理中针对提取的信息进行域迁移检测；③上传部分帧（以一个固定的速率下采样）到云服务器进行标注，

这些帧将用于之后的模型演进；④在检测到域迁移后，通过当前推理模型域、域标注信息、模型分类表3个信息进行模型选择。域标注信息可以获得当前域信息；当前域信息和模型分类表可以获得缓存中的本地最适模型和模型库中的全局最适模型。分为3种情况：①在当前推理模型域与当前领域信息相同时，表明当前域中发生了数据偏移，进行快速微调；②在当前推理模型域与当前域信息不同时，表明发生了域迁移，需要转变当前域，端设备快速更换当前模型为本地最适模型，并判断本地最适模型是不是全局最适模型，当本地最适模型不是全局最适模型时，申请全局最适模型并完成部署；③在完成快速演进后，缓存在后端进行模型更换，为下一次快速演进做准备。

为了有效地支持端设备，云服务器在每个窗口中主要进行如下任务：①当接收到来自端设备的数据帧，使用泛化模型为数据进行标注，随后使用基础模型与特定的提示文本来为数据帧鉴别出域语义标注；②当接收到来自端设备的模型请求时，下发对应的模型；③周期性地检查云数据池中每个域积累的数据，当数据达到阈值时，触发完整重训练任务（对旧域的重训练或生成新域模型），并在完成重训练后有需求时更新云端语义模型分类表，并下发给所有端设备。

值得注意的是，端设备上在多数模型缓存命中的情况下，本系统相当于直接没有了“上传数据、等待服务器检测、下发模型”这一流程。而在那些缓存未命中的情况下，本系统也仅仅是像之前的方法一样需要等待模型的下发，而去掉了“上传数据、等待服务器检测”的流程。需要指出，虽然服务器检测和端设备自己检测在时间上相差无几，但先前方法中服务器检

测并不能做到每个设备提交检测时立即进行检测，需要等待其他端设备依次完成检测才可以进行，存在排队时延。因此，无论缓存命中还是未命中，本系统都做到了一个更短的时延响应数据漂移。

3.3 设计细节

（1）端设备推理模块

端设备需要决定何时开始重用，更确切地说，确定数据漂移和域迁移发生的时间。之前的方法在云服务器上专门部署了深度神经网络（DNN）用于探测，但在本框架中，检测数据漂移的手段需要部署于端设备以规避端云间的传输成本，而端设备本身的资源不足以再部署一个DNN用于检测。因此，本框架从早期传统方法中探究，形成了一个轻量化检测分布偏移的计算方法。该方法使用马氏距离（Mahalanobis distance）维护了一个OOD（out of distribution，超出分布范围）分数，使设备可以判定当前数据流是否发生了数据漂移。

在每个模型中，本系统对每个训练数据使用一个固定的简单降维方法（固定选择模型中一层输出向量，并对该向量进行一次额外的平均池化），获得对应的低维向量。随后，对每个域 c 的向量计算平均值 $\hat{\mu}_c$ ，并使用平均值和单个向量的差值计算协方差矩阵 Σ 。这里，每个域的 $\hat{\mu}_c$ 、模型的 Σ 和平均分数就是系统用来检测的参数。在实时推理任务中，使用同样的方法获得输入数据的对应向量 $\mathbf{x}$ ，代入马氏距离公式获得单个数据的 $S(\mathbf{x}_i)$ ，并以此获得整个窗口的平均分数 $S_{\text{win}}(\mathbf{x})$ ，具体计算式如下。

$$S(\mathbf{x}_i) = \min_c \left[\left(f(\mathbf{x}_i) - \hat{\mu}_c \right)^T \hat{\Sigma}^{-1} \left(f(\mathbf{x}_i) - \hat{\mu}_c \right) \right] \quad (1)$$

$$S_{\text{win}}(\mathbf{x}) = \frac{1}{N_{\text{win}}} \sum_{\mathbf{x}_i \in \mathbf{x}} S(\mathbf{x}_i) \quad (2)$$

其中, N_{win} 是单个窗口的数据总数。在特化模型生成过程中, 本系统添加了一小部分验证数据模拟当前域的输入数据, 并计算了平均 OOD 分数, 作为该特化模型的一个特征。在之后的实时任务中, 本系统将窗口的平均分数 $S_{\text{win}}(\mathbf{x})$ 与平均 OOD 分数进行比较, 判断窗口数据是否符合当前域。判断依据是, 将平均 OOD 分数设置的函数作为分数线, 当窗口平均分数超过这个值, 即认为触发了域迁移。作为整个系统的一个超参数, 该分数线可以在实时任务前进行设置。本文为其设置了一个线性函数, 并进行了测试微调来获得一个比较理想的效果。这个分数线粗略地预测了领域相对于平均值的上限位置。

(2) 端侧模型缓存

就好比一个 CPU Cache (高速缓存) 需要在完成一次读取后进行预取一样, 本系统也需要在完成重用后对缓存中的模型进行预取以更高地命中下一次域迁移的可能性。尽管领域迁移难以预测, 时空相关性仍然给了系统一些方法来在资源受限的设备上进行预测。在对比实验后, 本文确定了如下替换策略。其一, 保留上一个被使用的域模型。这种选择的实际情况是在

自动驾驶中存在大量反复的例子, 进出隧道、过桥、上下高速会让两个域的模型在这些情况下被交替使用。其二, 选择那些具有更高相似性的域进行加载。从时间性上来说, 具有更高相似性的域的衔接更可能在现实中发生, 突变的域更多出现在非连续的拼接视频和特殊现象中。

(3) 端侧模型演进

如前文所述, 当发生相同域的数据漂移时, 本系统不能用重用来解决精度问题, 而需要使用微调。在发生这种情况时, 系统会在窗口中记录一个标志来表明准备微调。当来到一个窗口的末尾时, 系统会积攒足够的标签数据用于重训练。需要指出的是, 这种端设备上的微调工作并不能做到像云端上的重训练一样面面俱到。因此在有限的资源中, 系统选择进行有模型起点的单层模型微调工作。经过测试, 在 NVIDIA Jetson TX2 设备上, 单层模型微调工作的训练速度为 103 ms/img, 并且一次完整的微调工作时长为 110 s。虽然这只略快于云上的完整重训练工作, 但其具有不会产生排队时延的优点。当服务器接入更多设备时, 每个端设备的重训练任务如果都增加到服务器上, 会严重增加服务器的负载, 同时对各个端设备产生极长的排队时延。在充分利用端设备性能的情况下, 系统拓展性大幅提高。

(4) 云侧模型库

根据用户提供的语义属性, 本系统生成了一棵前缀树来组织各个领域, 如图 7 所示。在这棵树中, 每一层代表一个属性。每一层的子节点表示对应的父节点在该层所对应的属性上拥有对应值的限制。这导致整棵树越往下, 节点上的数据受限制越大。同时, 叶节点代表该节点上的数据对所有的属性都有对应的值, 这个叶节点上所有的数据便构成了一个具有相同语义属

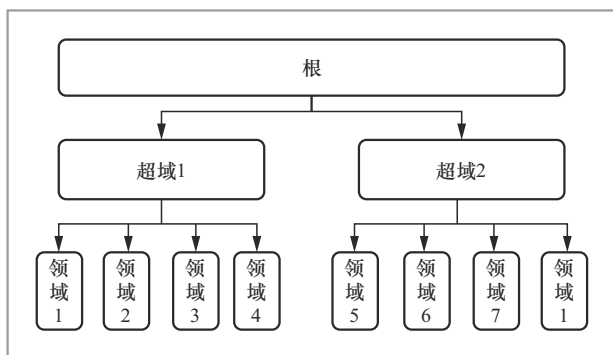


图7 模型库前缀树示意图

更新模型语义分类表时有区别，即新模型会记录进分类表，而先前的训练任务参数都是一样的。系统选择的重训练模型起点来源于该域在分类表上的祖先节点（系统确保了最起码根节点模型是存在的），从理论上来讲，祖先节点的数据更丰富、具有更好的泛化性，实验也验证了这个观点。

4 实验结果

4.1 实验环境

云服务器使用的 CPU 型号是 Intel Xeon Gold 5420+，GPU 型号是 RTX 4090。移动端设备型号为 NVIDIA Jetson Nano 和 NVIDIA Jetson TX2，具体配置见表 1。两者均为 Ubuntu 18.04 操作系统，并使用了 CUDA 10.2 和 cuDNN 7.6.5 来支持 GPU 计算。此外，深度学习框架选择了 PyTorch 1.10，以确保模型训练和推理的高效性。

本系统为服务器和端设备配置了不同的 Python 环境。在服务器上，因为需要进行基础模型 LLaVA 的本地部署，相应配置

了 Python-3.10.14（考虑仅使用基础模型 API，不部署本地模型时可以酌情修改该部分），两种端设备均为 NVIDIA 特供的 Python-3.6.9。因为端设备系统环境为 Python3.6，在更高版本的 YOLO 中所必需的底层库 Ultralytics 最低支持 Python 3.7，故在物体识别任务中，使用不依赖 Ultralytics 库的最高 YOLO 版本 YOLOv5-5.0、特化模型 YOLOv5-s、泛化模型 YOLOv5-x。在图像分类任务中，本系统使用的特化模型为 ResNet18，泛化模型为 ResNet50。本文使用的目标检测数据集有 CIFAR-10、Office-31、BDD100K (Berkeley DeepDrive Dataset^[19])。其中，Office-31 拥有 3 个标签类别一致、内容相似但具有整体风格差异的数据领域；而 BDD100K 拥有超过 100 个基于“时间、地点、季节”分类的领域，包含 10 万张驾车途中的风景照。

4.2 单系统微实验

本节主要针对本系统进行实验验证。图 9 在相同的初始条件下，使用相同的源自 Office-31 的测试数据，测试微调和重

表1 实验环境

项目	训练服务器	NVIDIA Jetson Nano	NVIDIA Jetson TX2
CPU	AMD R9-7950X	四核 ARM Cortex-A57	双核 NVIDIA Denver 2 CPU +四核 ARM Cortex-A57
CPU 主频	4.5 GHz	1.43 GHz	Denver 2: 2 GHz Cortex-A57: 2 GHz
线程数	36	4	8
内存	64 GB DDR5	4 GB LPDDR4	8 GB LPDDR4
算力单元(GPU)	GeForce RTX 4090	128-core NVIDIA Maxwell™ architecture GPU	256-core NVIDIA Pascal™ architecture GPU
显存	24 GB	与内存共享	与内存共享
GPU 频率	2.52 GHz	921 MHz	1.3 GHz
系统	Ubuntu 20.04	Ubuntu 18.04	Ubuntu 18.04
单精度算力	82.58 TFLOPS	0.5 TFLOPS	1.33 TFLOPS

用带来的效果。作为对比，选择了3种方式，分别是无演进、重用和重用+微调。重用+微调获得了最好的结果，这是符合本文预期的，它的平均精度均值比无演进高15%，并在发生微调的窗口比重用的平均精度均值高5%。重用整体的平均精度均值比无演进高12%，说明重用在端设备上的有效性。由图9可知，重用和微调都是有效提高整体精度（mAP）的策略，具有实际使用价值。

图10所示为在相同的初始条件下，使用相同的源自CIFAR-10的测试数据，测试端侧重用于云侧重用在不同设备数量的效果。可以看到，端侧重用随着设备数量增多并不会有明显起伏，这是因为每台设备并不会因为设备数量的增多而消耗更多资源，而云侧重用会因为更多的设备产生的重用任务在云侧堆积而逐渐升高每个重用任务的响应时间。

4.3 多系统对比实验

本节针对多个系统进行对比，主要使用了5个模型。

- 无演进：没有进行任何重训练或重用的系统。
- 一次演进：仅在过程中进行一次重训练的系统。
- ODIN^[10]：使用聚类算法来分类数据并进行域漂移检测的方法，使用重训练来提高精度，为了所有方法公平比较，笔者为ODIN添加了重用机制。
- RECL^[13]：首个提出并结合了重训练和重用的系统。其对端设备没有进行优化；在云端，其设计了模型选择器、模型演进控制器和模型演进调度器完成了重训练和重用的复杂调度；同时，它还设计了在多设备下对GPU资源进行分配的策略。

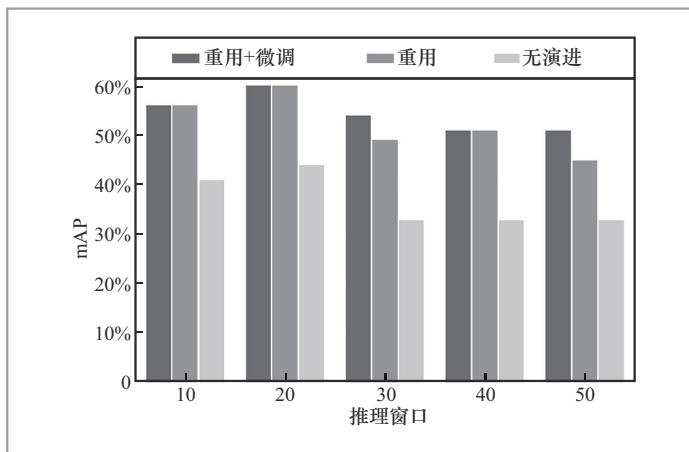


图9 重用与微调效果

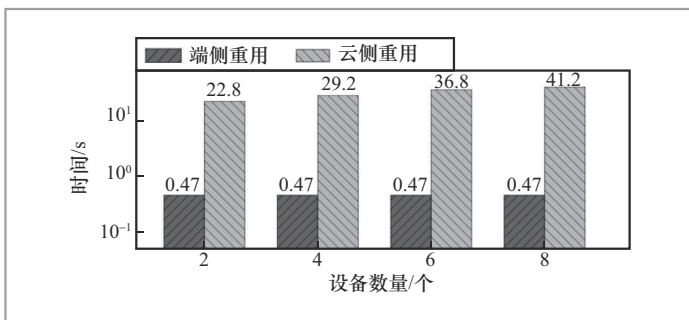


图10 端侧重用与云侧重用的响应时间

- 本文提出的方案。

其中，RECL为当前最优的方法，有最好的精度；添加了重用机制的ODIN有当前最快的响应速度。Ekya^[11]的工作重心在于对计算资源与内存的优化，因为没有重用机制且没有良好的模型库管理，在精度与响应时间的研究中弱于RECL；EdgeFM^[12]的工作重心在于获得开放集识别的能力，这与本工作获得开放集识别能力的方式类似，但同时，本工作更关注系统整体对数据漂移的响应速度和数据漂移时系统整体的精度，并针对系统设备数量的提升有较强的鲁棒性，这都是EdgeFM所不具备的，在以下关于精度和响应时间的验证中，EdgeFM性能弱于RECL。为

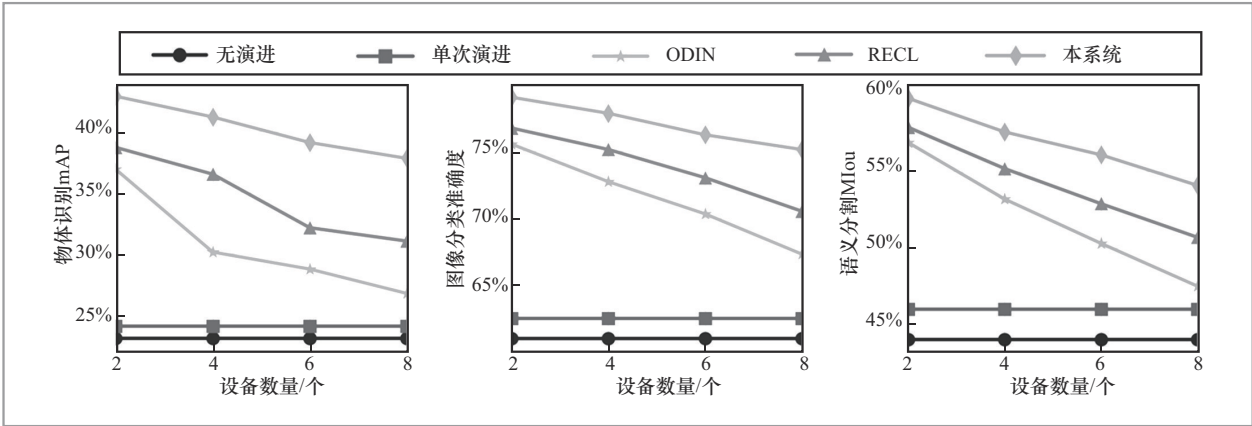


图 11 3 类任务多系统对比实验结果

此，当本系统性能优于 RECL 和带重用的 ODIN 两项工作时，即可证明本系统相比 RECL、ODIN、EKYA、EdgeFM 4 项之前的工作拥有更高的响应速度和精度。

图 11 采用的数据集是 BDD100K。实验选择从测试集中采样 70 000 张图片应用到每个端设备上，每一个视频被采成连续的图片，并且当一个视频结束时衔接下一个视频

的数据。这个序列中包含数个域迁移。为了明显展示不同系统的性能差异，笔者着重测量了有数据漂移发生的时间段。可以看到，在图 11 中，持续学习模型在 3 项下游任务（图像分类、物体识别与语义分割）中，都展现出了比非持续学习模型更好的效果。本系统因为更短的响应时间和层次化的演进策略，拥有最优的精度。随着设备数量上升，本系统的精度逐渐和 RECL 拉开差距，这是因为本系统良好的扩展性得到了体现。本系统对服务器拥有更低的资源要求，这在设备数量增多时会越来越明显。当 RECL 因为过多的端设备数量导致排队延时，本系统的服务器还远没有达到阈值。

在响应时延方面，实验测试了本系统、RECL、ODIN 的具体延时，图 12 和图 13 给出了 8 台端设备进行测试的具体样例。本系统平均重用时延为 8.37 s，而 RECL 的平均重用时延为 17.76 s，ODIN 的平均重用时延为 14.95 s，与预期相符合。同时，在完成了重用后，系统还有一段微调工作能二次提高推理精度，这使本系统整体和 RECL 的重用精度一致，而 ODIN 的门网络未及时响应数据漂移带来的变化，精度低于另外两种方法。而在重训练任务上，本系统和 ODIN 均在 180 s 左右完成了

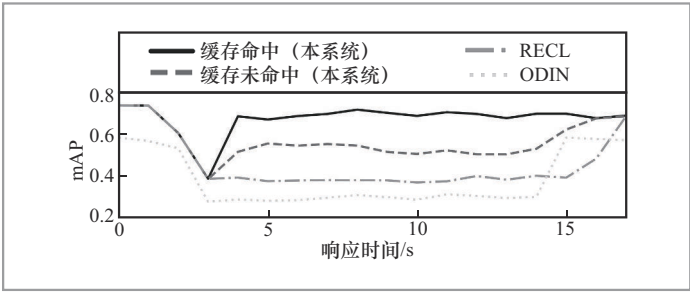


图 12 物体识别中重用响应样例

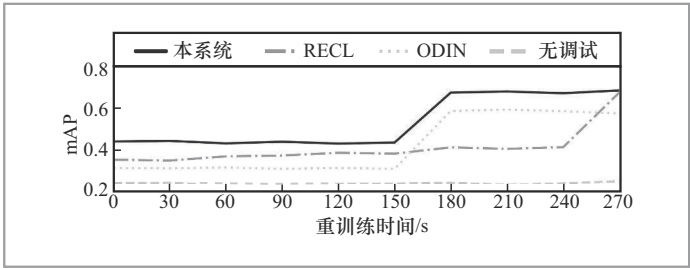


图 13 物体识别中重训练任务样例

重训练任务，而 RECL 因为过度的重训练任务触发导致其在云侧的重训练任务会有排队不能及时开始重训练的现象，需要 270 s 才能完成任务。但是本系统和 RECL 均能达到较好的精度需求，而 ODIN 因为受限于自身门网络分类数据无法及时响应数据漂移的特性，无法达到较好的精度。

5 结束语

本文设计了一个端云协同的持续学习框架，涵盖了模型的重用、微调以及端设备重训练三大演进工作。该框架在端设备与云服务器之间实现了高效的协同工作，既能充分利用云服务器的计算资源，又能利用端设备的实时响应能力。

实验结果表明，在图像分类和物体识别两项任务中，持续学习模型相较于传统的非持续学习模型展现出了更好的性能。特别是在动态环境下，持续学习模型能够更好地演进变化，保持较高的精度。在域迁移检测方面，之前的工作 RECL 虽然表现出较高的检测精度，但在设备数量增加时，其性能明显下降，而本系统由于其良好的扩展性，表现出更稳定的性能。在响应时延方面，因为端侧模型缓存模块的确立，本系统拥有最低的响应时延与最高的响应速度。此外，本系统在完成模型重用后，通过进一步的微调工作能显著提高推理精度，而 RECL 在这方面的效果比较有限。这表明，在需要高精度和快速响应的应用场景中，本系统的综合性能更优。

未来，可进一步研究以下方向：在模型库前缀树结构中，如何应对更广泛的模型增加带来的更新维护压力；在端侧设备微调任务中，如何应对更复杂、更极端场

景带来的精度提升压力等。

参考文献：

- [1] GUIDONI D L, MAIA G, SOUZA F S H, et al. Vehicular traffic management based on traffic engineering for vehicular ad hoc networks[J]. IEEE Access, 2020, 8: 45167-45183.
- [2] PANCHENKO V, ZAMULA A, KAVUN S, et al. Intelligent management of the enterprise personnel security system [C]//Proceedings of the 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies. Piscataway: IEEE Press, 2018: 469-474.
- [3] BUTLER L, YIGITCANLAR T, PAZ A. Smart urban mobility innovations: a comprehensive review and evaluation[J]. IEEE Access, 2020, 8: 196034-196049.
- [4] LIU L K, LU S D, ZHONG R, et al. Computing systems for autonomous driving: state of the art and challenges[J]. IEEE Internet of Things Journal, 2021, 8(8): 6469-6486.
- [5] YURTSEVER E, LAMBERT J, CARBALLO A, et al. A survey of autonomous driving: Common practices and emerging technologies[J]. IEEE Access, 2020, 8: 58443-58469.
- [6] JIANG J C, ANANTHANARAYANAN G, BODIK P, et al. Chameleon: scalable adaptation of video analytics[C]//Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication. New York: ACM, 2018: 253-266.
- [7] LI Z Z, HUANG Y L, FU X M. EVA: a distributed optimization architecture for efficient video analytics for smart cities

- [C]//Proceedings of the 18th Workshop on Mobility in the Evolving Internet Architecture. New York: ACM, 2023: 7–12.
- [8] 乐俊青, 谭州勇, 张迪, 等. 面向车联网数据持续共享的安全高效联邦学习[J]. 计算机研究与发展, 2024, 61(9): 2199–2212.
- LE J Q, TAN Z Y, ZHANG D, et al. Secure and efficient federated learning for continuous IoV data sharing[J]. Journal of Computer Research and Development, 2024, 61(9): 2199–2212.
- [9] PARISI G I, KEMKER R, PART J L, et al. Continual lifelong learning with neural networks: a review[J]. Neural Networks, 2019, 113: 54–71.
- [10] SUPREM A, ARULRAJ J, PU C, et al. ODIN: automated drift detection and recovery in video analytics[EB]. arXiv preprint, 2020, arXiv:2009.05440.
- [11] BHARDWAJ R, XIA Z X, ANANTHANARAYANAN G, et al. Eky: continuous learning of video analytics models on edge compute servers[EB]. arXiv preprint, 2020, arXiv:2012.10557.
- [12] YANG B F, HE L X, LING N W, et al. EdgeFM: leveraging foundation model for open-set learning on the edge[C]//Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems. New York: ACM, 2023: 111–124.
- [13] KHANI M, ANANTHANARAYANAN G, HSIEH K, et al. RECL: responsive resource-efficient continuous learning for video analytics[C]//Proceedings of 20th USENIX Symposium on Networked Systems Design and Implementation. [S.l.:s.n.], 2023: 917–932.
- [14] 杨静, 吉晓阳, 李少波, 等. 具有正则化约束的脉冲神经网络机器人触觉物体识别方法[J]. 电子与信息学报, 2023, 45(7): 2595–2604.
- YANG J, JI X Y, LI S B, et al. Spiking neural network robot tactile object recognition method with regularization constraints[J]. Journal of Electronics & Information Technology, 2023, 45(7): 2595–2604.
- [15] PAN T L, SUMALEE A, ZHONG R X, et al. Short-term traffic state prediction based on temporal-spatial correlation[J]. IEEE Transactions on Intelligent Transportation Systems, 2013, 14(3): 1242–1254.
- [16] 杨静, 何瑶, 李斌, 等. 基于门控机制与重放策略的持续语义分割方法[J]. 电子与信息学报, 2024, 46(7): 2908–2917.
- YANG J, HE Y, LI B, et al. A continual semantic segmentation method based on gating mechanism and replay strategy[J]. Journal of Electronics & Information Technology, 2024, 46(7): 2908–2917.
- [17] 董次浩, 陈雷鸣, 黄子凌, 等. 基于图持续学习的时序数据分析[J]. 计算机系统应用, 2024, 33(2): 188–197.
- DONG C H, CHEN L M, HUANG Z L, et al. Time series data analysis based on graph continuous learning[J]. Computer Systems and Applications, 2024, 33(2): 188–197.
- [18] BOMMASANI R, HUDSON D A, ADELI E, et al. On the opportunities and risks of foundation models[EB]. 2021, arXiv preprint, arXiv:2108.07258.
- [19] YU F, CHEN H F, WANG X, et al. BDD100K: a diverse driving dataset for heterogeneous multitask learning[C]//Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway: IEEE, 2020: 2636–2645.

作者简介



廖天俊（1985–），男，博士，军事科学院战略评估咨询中心评估员，主要研究方向为群体智能优化与评估算法。



赵茆哲（2001–），男，上海交通大学计算机学院博士生，主要研究方向为边缘智能、持续学习。



刘生钟（1995–），男，博士，上海交通大学计算机学院副教授，主要研究方向为边缘智能与物联网、智能实时系统与信息物理系统、机器学习。



吴帆（1981–），男，博士，上海交通大学计算机学院教授，主要研究方向为无线网络与移动计算、博弈论算法及其应用、数据管理与隐私保护。

收稿日期: 2025-03-31
通信作者: 刘生钟, shengzhong@sjtu.edu.cn
基金项目: 国家重点研发计划项目 (No. 2022ZD0119100); 国家自然科学基金项目 (No. 62472278, No. 62025204, No. 62432007, No. 62441236, No. 62332014, No. 62332013)
Foundation Items: The National Key Research and Development Program of China (No. 2022ZD0119100), The National Natural Science Foundation of China (No. 62472278, No. 62025204, No. 62432007, No. 62441236, No. 62332014, No. 62332013)