一种多终端视频流智能识别模型共进演化方法研究

王乐豪[1)] 刘思聪[1)] 於志文[1),2)] 于昊艺[1)] 郭 斌[1)]

[1)](西北工业大学计算机学院 西安 710072)

[2)](哈尔滨工程大学 哈尔滨 150006)

摘 要 在泛在的智能物联网终端部署深度模型并提供智能应用/服务受到越来越多关注.但是,受限于终端硬件资源,研究人员从模型轻量化技术入手,为深度模型的轻量化、高精度部署提供技术支撑.然而基于轻量化深度模型的视频应用会面临实际场景中的数据漂移问题,导致推理精度急剧下降,并且该问题在移动场景中尤为显著.边缘辅助的模型在线演化是解决数据漂移问题的一种有效方式,可实现自演化的可成长的智能计算系统.然而,模型演化速度会影响终端模型高精度服务时间占比,从而影响模型全生命周期推理性能.为了提升多终端协同的模型演化精度和速度,本文提出基于软硬一体理念的多终端视频流智能识别模型共进演化方法和系统.一方面,本文提出了新颖的多终端互学习共进演化方法,借助终端新场景数据,克服模型数据异构挑战,实现多终端模型和全局模型的高增益协同演化和共进学习;另一方面,结合互学习算法特点,提出基于存内计算的训练加速方法,利用自适应数据压缩和模型训练优化提升系统性能,在保证演化精度增益的同时加速多个终端模型的演化速度.最后,通过不同真实移动场景下的轻量化模型持续演化任务实验验证,并对比六种基准方法证明 NestEvo 可以有效减少 51.98%演化延迟,并提升 42.6%终端轻量化模型平均推理精度.

关键词 数据漂移;模型共进演化;互学习;训练加速方案;存内计算;智能物联网

中图法分类号 TP393 **DOI 号** 10.11897/SP.J.1016.2024.00947

Research on Co-Evolution Method of Multi-Terminal Video Stream Intelligent Recognition Models

WANG Le-Hao[1)] LIU Si-Cong[1)] YU Zhi-Wen[1),2)] YU Hao-Yi[1)] GUO Bin[1)]

[1)](*School of Computer Science, Northwestern Polytechnical University, Xi'an* 710072)

[2)](*Harbin Engineering University, Harbin* 150006)

Abstract Developing Artificial Intelligence of Things (AIoT) technology and boosting the construction of a ubiquitous computing digital infrastructure system are important directions. In order to overcome the privacy issues brought by cloud computing and meet the needs of low-latency applications, deploying deep models on ubiquitous intelligent IoT terminals to provide intelligent applications/services has attracted more and more attention. But the terminal deployment of the deep model has many challenges. Limited by the available resources of the terminal hardware platform, researchers start with model compression technology and hardware accelerators to provide technical support for the lightweight and high-quality deployment of deep models. However, the video application based on the deep model will inevitably face the problem of data drift in the actual mobile scenes. Moreover, this problem is especially noticeable in mobile scenes and devices

收稿日期:2023-06-17;在线发布日期:2024-01-19.本课题得到国家自然科学基金重点项目(61960206008)、国家杰出青年科学基金(62025205)和国家自然科学基金面上项目(62102317)资助.**王乐豪**,博士研究生,主要研究方向为移动嵌入式计算.E-mail:lehaowang@mail.nwpu.edu.cn.**刘思聪**(通信作者),博士,副教授,主要研究方向为智能物联网、移动嵌入式计算.E-mail:scliu@nwpu.edu.cn.**於志文**(通信作者),博士,教授,主要研究领域为普适计算、智能物联网.E-mail:zhiwenyu@nwpu.edu.cn.**于昊艺**,硕士,主要研究方向为智能物联网、移动计算.**郭 斌**,博士,教授,主要研究领域为普适计算、群体智能、移动群智感知.

because of more severe distribution fluctuations and sparser network structures. Under the influence of data drift, the accuracy of the deep model will decrease significantly, making it difficult to meet the performance requirements. Edge-assisted model online evolution is an effective way to solve the problem of data drift, which can realize an intelligent computing system that can evolve and grow. Previous model evolution systems only focus on improving the accuracy of the terminal model. But in multi-terminal system, the global model is also affected by data drift due to the ore complex and varied scenario data from different terminals, resulting in a decrease in accuracy gain in the system. In order to provide stable and reliable knowledge transfer to the terminal models, it is necessary to use federated learning to evolve the global model. However, traditional federated learning will face multiple challenges of terminal model heterogeneity, and data distribution heterogeneity in multi-model evolution systems. What's more, the speed of online evolution will affect the time proportion of high-accuracy services in the terminal models, decreasing their life-cycle performance. In order to collaboratively improve the accuracy and speed of model evolution for multiple terminal models, the paper proposes a method and system for the co-evolution of multi-terminal video stream intelligent recognition models based on the concept of software and hardware integration. On the one hand, we develop a novel multi-terminal mutual learning and co-evolutionary evolution method, which overcomes the challenge of model data heterogeneity with the help of new terminal scene data, and realizes high accuracy gain co-evolution and co-evolutionary learning of multi-terminal models and global models. On the other hand, combined with the characteristics of mutual learning algorithms, a training acceleration method based on in-memory computing is proposed, which uses adaptive data compression and model training optimization to improve hardware performance, and accelerates the evolution speed of multiple terminal models while ensuring the evolution accuracy gain. Finally, through the experimental verification of the continuous evolution task of the lightweight model in different real mobile scenarios, and comparing six benchmark methods, it is proved that NestEvo can effectively reduce the evolution delay by 51.98% and improve the average inference accuracy of the lightweight model of the terminal by 42.6%.

Keywords data drift; model evolution; mutual learning; training acceleration scheme; in-memory computing; Artificial Intelligence of Things

1 引 言

近几年,物联网(IoT)和人工智能(AI)技术的交叉融合催生出一个极具前景和挑战的新兴前沿领域——智能物联网(Artificial Intelligence of Things, AIoT),其关注基于小型化物联网终端设施,主动建立与环境融为一体的感知计算模式.并且深度学习驱动应用已逐步融入城市管理、公共安全、工业制造等多个国家重大需求领域.然而,将深度模型部署到移动终端面临着资源不适配的挑战.随着深度学习技术的快速发展,视觉深度模型的准确性显著提高.但是,深度学习驱动的视觉数据分析与环境理解依赖于大量计算资源(如存储、算力等),如何在资源有限的小型化物联网终端设施上部署执行成为挑战.由于深度模型存在大量冗余的节点,仅仅只有少部分(5%~10%)重要权值参与大部分输入的计算[1].因此,研究人员提出了深度学习模型压缩方法[2],以较少精度损失为代价减少模型冗余的计算量和参数量,以适配资源受限的终端硬件平台.

然而,基于轻量化深度模型的终端视频分析应用会面临严重的数据漂移问题[3-4].这是因为移动终端真实感知的数据与模型训练数据集分布不同,导致模型在真实场景中推理精度下降.此外,轻量化模型的参数稀疏性导致的泛化能力下降以及移动场景数据分布存在未知性和动态性等原因,加剧了数据漂移对于移动端压缩模型的精度影响.边缘设备辅助的在线模型演化[5-6],即终端模型在边缘服务器端利用实时累积数据进行模型重训练,是解决真实移动场景数据漂移问题的一种理想方案.在模型演化

系统中，终端进行实时视频推理的同时会定期向服务器发送演化请求，两次连续演化请求之间的时间间隔为模型的生命周期. 为了提升终端模型全生命周期内的平均推理精度，可以从提高演化精度增益以及缩短演化延迟两方面入手.

终端实时感知的无标签数据会为模型演化的持续精度增益引入了错误累计. 而知识蒸馏可以在无需依赖于标签信息的前提下，使低精度的“学生”模型能够学习到“教师”模型在数据上的决策边界和类别之间的关系. 因此，基于知识蒸馏[7-8]的学习方法是实现模型演化的理想途径. 具体地，位于边缘服务器端精度更高的复杂全局模型作为“教师”以终端实时捕捉、筛选并上传至服务器的视频帧作为媒介，向作为“学生”的终端模型传递知识，“学生”模型在“教师”模型的指导下，模拟“教师”模型的输出概率分布，获得更好的推理性能.

然而，在不同移动场景下多终端模型同时发起演化请求的情况下，作为“教师”的全局模型会受到数据漂移的影响导致输出的监督信号质量下降，进而影响终端模型的演化效果. 为了节约演化成本并充分利用服务器的硬件资源，在共享的边缘服务器上进行多终端深度学习服务变得愈发流行. 而多终端演化相比于单终端演化，全局模型面对的数据特征分布变化更加复杂，全局模型更易受到数据漂移影响而发生精度下降. 如实验 7.2.1 节所示，这会影响终端模型的演化精度增益. 因此，多终端模型与全局模型的高效共进演化，提高全局模型的监督信号质量对于保证终端模型演化精度增益尤为重要.

此外，资源竞争和存算隔离带来的演化延迟很难满足实时视频分析应用对模型演化延迟的要求. 具体地，在每个终端模型生命周期内，在终端发起演化更新请求后，终端依旧会使用旧的低精度模型进行推理，直到基于新感知数据更新的新模型返回部署后，终端才会利用新的高精度终端模型进行推理. 因此，我们需要加快模型演化速度，尽快将演化后的高精度新模型部署回终端，增加新模型推理时间在模型演化窗口内的比例，提高终端模型全生命周期推理性能(Life-Cycle Performance，LCP). 对于缩短演化延迟，一方面，利用基于 GPU 的边缘服务器进行多终端深度学习服务在充分利用硬件资源，提升系统整体吞吐量的同时会引发共享资源(如显存、计算资源、带宽资源等)竞争，致使演化任务停滞. 另一方面，由于 GPU 存算隔离的特性，多终端模型演化会触发更加频繁的高级别存储(L3 缓存和 DRAM)访问，进而导致多终端模型演化延迟的急剧增加. 随着近些年 AI 硬件加速器的发展，相比于算法层面的优化改善，从硬件层面加速模型推理和训练过程取得了卓越效果并逐渐成为主流手段.

基于上述问题，本文提出了一种多终端视频流智能识别模型共进演化方法(NestEvo)，旨在提升受到数据漂移影响的终端模型推理精度. 我们利用模型在线演化技术，自适应地触发终端模型和全局模型的协同在线演化，并为边缘服务器端模型演化过程选择合适的加速方案. 与此同时，根据模型所属的硬件特点进行模型训练优化，使软硬件更好地适配，提升系统的整体性能. 本文在算法层面实现终端模型和全局模型的协同在线演化，提升终端模型的演化效果和推理精度；在系统层面上加速模型演化速度，降低演化延迟，实现更快的新模型部署，以进一步提高系统内终端模型的平均推理精度.

本文的主要贡献如下：

(1) 本文提出了新颖的多终端互学习共进演化方法，实现了多个终端模型和全局模型的协同在线演化. 通过互学习增强演化中多终端的知识交互和共进学习，改善全局模型提供的监督信号置信度和终端模型的演化精度增益，提升受到数据漂移影响的长生命周期终端模型推理精度. 通过此学习模式可以完善现有模型演化系统，突破性能瓶颈.

(2) 本文提出了模型全生命周期推理性能优化指标，并提出了基于存内计算的训练加速方案，降低多终端互学习共进演化延迟. 同时，针对 GPU 存储特点和存内计算噪声明显的缺点，我们通过自适应数据压缩机制和模型抗噪声训练优化，在加速模型演化的同时保证精度. 通过此硬件加速方案的理论分析和模拟验证可以指导未来模型在线演化系统中边缘服务器的异构加速器设计.

(3) 通过在不同压缩程度的终端个性化轻量模型、五个实时视频流数据上进行实验验证，并对比 6 种基准方法，验证 NestEvo 在模型精度增益、演化速度和减少硬件资源占用等方面都有较大提升. 平均演化延迟减少 51.98%，终端模型平均推理精度可提升 42.6%.

本文第 2 节分析相关工作；第 3 节介绍本文研究动机；第 4 节介绍系统的整体设计；第 5 节介绍多终端互学习共进演化方法；第 6 节介绍多终端并发演化任务调度与存内计算加速；第 7 节进行实验验证；最后总结全文.

2 相关工作

本节将对终端深度模型轻量化方法、轻量化深度模型在线演化方法、多终端深度模型互学习以及基于存内计算的深度模型训练加速的相关工作进行介绍和分析.

2.1 终端深度模型轻量化方法

模型压缩技术可以在可接受的模型精度损失范围内大幅度降低模型体积和运算量,使其适配于资源紧张的终端设备,实现终端深度模型的部署.现在主流的压缩技术有剪枝[9-12]和权重量化[13-16],或是直接设计使用轻量化网络.Han 等人[9]建议将训练好的神经网络中不重要的权重修剪掉,从而通过稀疏格式存储模型来减少所需存储空间和计算量,但这种方法只能通过专用的稀疏矩阵运算库或硬件来实现加速.运行时内存节省也非常有限,因为大部分内存空间被激活映射(仍然密集)占据而不是被权重消耗.文献[17]修剪归一化(Batch Normalization,BN)层中较小的权重来达到精简网络的目的;文献[18]计算每一个过滤器上权重绝对值之和,去掉 m 个权重最小的过滤器,并同时去掉与其相关的特征图及下一层的所有相关输入过滤器来进行剪枝;文献[19]提出了一个统一的卷积神经网络(Convolutional Neural Network,CNN)模型训练和剪枝框架.特别地,通过在 CNN 的某些结构上引入缩放因子和相应的稀疏正则化,将其转化为一个联合稀疏正则化优化问题,并且作者利用改进的随机加速近端梯度方法,利用稀疏正则化联合优化 CNN 和标度因子的权重.与以往的启发式算法相比,该方法无需进行微调和多阶段优化,具有更稳定的收敛性和更好的结果.

量化是将浮点存储(运算)转换为整型存储(运算)的一种模型压缩技术.文献[20]采用剪枝与量化相结合的方式,首先根据权重对网络进行剪枝,之后再进行参数量化,最后对结果进行霍夫曼(Huffman)编码.文献[21]表示深度量化可能会给模型造成重大精度损失,由于层与层之间的依赖性很强,因此为网络中的每一层选择最佳的量化位宽并不容易,该文献提出了一种新的正弦正则化方式,通过学习正弦函数的周期来学习每层量化位宽以及一个比例函数,同时利用正弦函数的周期性、可微性和局部凸性轮廓,自动推进网络权值在共同确定的水平上进行量化,降低精度损失.

另外,使用轻量级的神经网络是这几年非常流行的终端深度学习服务部署方式,研究者们也设计出了各种轻量化的网络,例如 MobileNet v1[22]、MobileNet v2[23]、MobileNet v3[24]、ResNet[25]等,这些模型复杂度低、体积小、运算量少,非常适合在终端部署.

本文提出的多终端模型演化系统能够支持不同资源约束下的不同压缩程度的轻量化模型演化.并且通过定期对终端模型进行高效低延迟的在线演化可以实现终端模型的稳定高水平推理精度,从而满足实时视频分析的需求.

2.2 轻量化深度模型在线演化方法

部署在终端的简单的浅层深度学习模型会因为场景变化等因素造成的数据漂移问题而推理精度不佳,因此,需要利用对应终端收集到的实时数据对模型进行在线演化,以保证其精度可以达到使用需求.知识的可迁移性是实现模型在线演化的基础,通过迁移学习可以从多维度提升终端模型的性能,实现终端模型的终生持续演化,而知识蒸馏[8]是应用于演化系统的一种主流迁移学习方式.在模型压缩的思想下,知识蒸馏可以实现从功能强大的复杂大型网络或集成网络到结构简单、运行迅速的小型网络的知识传递.而例如文献[26]的模型演化就是利用知识蒸馏技术,使终端设备间断性地向服务器发送视频帧,并在云端进行全局模型和终端模型的知识蒸馏,之后将新模型重新加载至终端,从而完成模型演化.但是,将数据和模型上传至云端会造成用户隐私泄露和模型演化延迟高等问题.文献[27]则在资源有限的边缘节点上同时实现模型推理和模型的定期演化以解决边缘视频分析系统的数据漂移问题,并在边缘端设计了资源调度算法以平衡推理和模型演化的资源占用.文献[6]研究了如何协调边缘端和云端资源,同时优化模型性能(包括准确性和鲁棒性)和资源成本以执行具有成本效益的在线学习,他们利用李雅普诺夫(Lyapunov)优化理论,设计并分析了一套高效优化框架,用于针对各种 AIoT 应用动态采集的新数据样本,在准入控制、传输调度和资源供应方面做出在线决策.文献[28]提出了基于随即相干性和注意力轨迹的在线学习方法,他们利用随即相干性对视频流进行压缩选择,提取关键帧并保存信息,同时通过建立注意力轨迹模型对关键帧进行分类预测,在新的视频流输入后,模型会根据注意力轨迹和历史信息对关键帧进行筛选和更新,并实现在线学习.而 Khani 等人[29]通过模型演化与模型重用的结合降低了多终端模型演化的延迟和所需

硬件资源.

上述工作只关注终端模型的训练过程，而忽略了作为“教师”的全局模型在多终端复杂情境中也会受到数据漂移的影响，发生精度下降，影响监督信号质量，从而损害终端模型的演化效果. 本文提出的NestEvo会自适应地进行终端模型和全局模型的协同演化，以保证监督信号质量，从而实现终端模型演化效果的稳定高效.

2.3　多终端深度模型互学习方法

互学习[30]是从知识蒸馏中衍生出的一项技术，其打破了传统知识蒸馏中预先定义好的主从关系，提出了一种知识交互、共进学习的策略. 在此策略下，多个学生网络在整个学习过程中利用彼此的输出信号相互指导，而不是教师和学生之间的单项转换通路. 互学习通过鼓励模型之间的协作和知识交流，使模型获得更高水平的准确性、灵活性和鲁棒性，先前工作[31-34]已经展示了互学习在提升深度模型性能和可解释性方面的能力和潜力. Nie 等人[31]提出了一种用于视频应用，联合人类动作识别和动作定位的互学习方法，该方法通过共享两类模型学习到的特征并进行相互学习从而同时提高了两类模型对于复杂视频场景中多个人类的识别精度. Zhao 等人[32]在其设计的一个端到端跟踪框架中利用互学习的方式加强骨干网络，取得了很好的效果. 在小样本学习中，Zhou 等人[33]发现利用全局视图下所有类的常规训练与采用局部视图中少数类的元任务训练方法之间具有互补性，因此他们提出了一个基于互学习的统一的小样本学习框架，通过视图内和跨视图建模实现了全局视图和局部视图的兼容性，大大提高了模型分类的准确性. 而在无监督学习中，Xu 等人[34]提出了一种用于多源域无监督域自适应的互学习框架，为了减轻域之间的差异并整合来自不同域的信息，他们提出了一个基于互学习的对齐网络来结合联合域对齐和独立域对齐两种方式，利用互学习使两个分支协同学习，从而挖掘丰富的知识，获得了更好的性能. 而我们面对的多终端情境其实是“多源域和多目标域”情况，我们需要使多个终端模型适应多个不同测试数据. 并且，多终端强调在一个边缘服务器服务的多个移动终端，包括不同视角/不同设备，在这种情境下，多终端之间以及多终端实时捕捉到的视频数据之间存在关联性. 因此，我们可以设计新的系统机制，利用此种关联性提升多终端模型性能.

本文提出了多终端互学习共进演化方法，实现了多终端模型和全局模型的知识交互和共进学习，提升演化精度增益，满足多终端用户需求.

2.4　基于异构模型和数据的模型聚合

模型聚合指的是各个基于本地数据训练的终端模型进行参数聚合，形成一个性能良好的全局模型的过程，在广泛研究的联邦学习范例中，每个参与的终端会需要使用或构建具有相同体系结构的本地模型，从而利用 FedAvg 算法[35]实现模型聚合. 而在实际的智能物联网应用中，由于个体需求和硬件资源的差异，每个移动终端期望以专门化的方式设计本地模型架构[36]. 并且，联邦学习强依赖于一个重要假设：每个数据中心的数据都是独立同分布的. 然而，在实际移动多终端情境中，数据通常是以非独立同分布(数据异构)的形式呈现，进而导致终端模型聚合后生成的全局模型精度不佳. 为解决上述异构性问题，先前工作[37-41]已经从模型个性化、终端模型筛选和终端模型集群的角度对模型聚合进行了优化. Guo 等人[37]利用 MoE 技术，通过机器学习模型挖掘终端模型与已有全局模型之间的相关性，从而聚合生成一个优秀性能的全局模型，从而克服模型和数据异构的问题. Tang 等人[42]提出了一种建立在基于相关性的模型筛选策略之上的联邦学习框架以提升受到数据异构性影响的全局模型收敛速度. 他们利用高斯过程对终端模型之间的损失相关性进行建模，基于高斯模型进行模型筛选，利用筛选的模型进行聚合得到全局模型，从而减少全局损失. 并且，Qin 等人[43]认为，在终端模型筛选时，除了考虑终端独立性之外，还需要关注终端之间的协同性. Xie 等人[44]则从模型集群的角度出发，提出了一种多中心聚合机制，即从数据中学习学习多个全局模型作为聚类中心，同时获得每个终端模型与中心之间的最佳匹配，通过随即期望最大化算法对该优化问题进行有效求解.

上述方法大多只解决了数据异构问题，而基于黑盒模型的异构模型聚合方法并不利于我们进行系统的性能优化. 而本文提出的多终端互学习共进演化方法可以同时克服模型和数据异构问题，并且通过基于存内计算的训练加速方案可以对系统延迟进行进一步的优化.

2.5　基于存内计算的深度模型训练加速方法

随着越来越多的深度学习应用融入各行各业，人们对于性能更好精度更高的深度模型有着迫切的

需求,然而这些模型的体积和计算量呈现指数型的增长,传统架构的处理器难以提供足够的内存和计算资源,尤其是边缘运行的多终端深度学习服务.而基于模拟计算的存内计算技术[45-46]为边缘大型计算提供了新的解决思路.其利用内存单元(如忆阻器[47])进行神经网络权重的存储和计算,消除了数据移动带来的额外成本,并能在向量运算期间并行发生数十万个乘加运算,其高性能和低延迟的特点使其在深度模型加速领域有着巨大的潜力.存内计算实现的高效矩阵乘法对于仅推理的深度学习应用非常有吸引力,目前已有大量工作将其应用于神经网络推理等领域[48-52].Ankit 等人[53]提出了一种基于可编程超高效忆阻器的加速器,旨在保持忆阻器交叉开关的存储密度,从而仅使用片上存储映射机器学习应用,作为第一个使用混合互补金属氧化物半导体(CMOS)忆阻器技术构建的可编程通用机器学习推理加速器,其在不影响可编程性的情况下,保持了存内计算和模拟电路的效率,实现了低能耗低延迟的推理加速.结合了开源指令架构(RISC-V)控制单元和模拟电路的 Mythic 的智能处理单元[54-55]也被设计应用于嵌入式视觉和数据中心应用等领域.而在工业界,英飞凌和台积电在 2022 年底宣布会将台积电的电阻非易失性存储器技术引入英飞凌的下一代 AURIX 微控制器,Weebit Nano 和昕原半导体等企业也正在极力打造基于存内计算的新型 AI 芯片以满足智能物联网设备等对于高性能低延迟加速效果的要求.而具有延迟敏感性的多终端模型演化系统对于存内计算的需求尤为强烈.然而,相比于目前的 GPU 架构,由于新型存储器电导噪声的影响,基于模拟计算的存内计算会使深度模型的推理精度发生不同程度的精度下降[56],这对于精度敏感的深度学习应用来说是无法容忍的.现有工作[57-61]讨论了硬件噪声对于具有不同数据类型权重的深度模型的训练影响,为了减少硬件噪声带来的推理精度损失,Gokmen 等人[57]和 He 等人[58]在训练期间加入与推理时相似的噪声以减少精度下降,但由于噪声的随机性和不确定性,为每次推理采用对应的不同噪声进行训练是不现实的,文献[56,59-60]也表示在训练过程中添加更多的噪声可以减少更多的精度损失.

本文将存内计算应用于边缘辅助的模型演化系统,结合模型演化系统的特点,设计了一套异构的边缘演化训练加速方案,旨在更加快速有效地实现模型演化.此外,针对两种架构的特点,本文通过自适应的数据压缩和模型训练优化技术,减少了 GPU 的内存资源占用和计算量以及存内计算中电导噪声导致的精度下降.

3 研究动机

边缘辅助的模型在线演化是解决数据漂移的一种理想方法,然而更为复杂多变的多终端模型演化情境在算法和系统层面带来更多的挑战,具体表现在多终端轻量化模型和感知数据异构性导致的演化增益下降以及多终端并发演化训练任务抢占服务器资源和存算隔离导致的演化延迟过高.

3.1 多终端轻量化模型和感知数据异构性导致的低演化增益

在复杂多变的移动多终端演化情景中,终端模型的演化精度增益会大幅度下降.终端部署的用于实时视频推理的轻量化模型不可避免地会面临数据漂移问题,导致模型推理精度的下降.边缘辅助的模型在线演化是解决数据漂移问题的一种有效方法,其主要思想是利用位于边缘服务器的高精度模型(全局模型)根据终端实时采集的上传视频数据为终端模型提供监督信号,使终端模型从中获取知识,实现精度提升.而面对复杂的多终端演化场景时,由于数据分布变化更为剧烈,全局模型也会受到数据漂移的影响,如 7.2.1 节所示,我们通过实验展示了全局模型在数据漂移影响下同样会发生不可忽视的精度下降.为了保证终端模型的精度增益,即演化效果,需要提升全局模型提供的监督信号质量,因此,全局模型在场景变化中需要自适应地完成演化更新.全局模型和终端模型目标不同,终端模型主要面向单一场景,而全局模型需要适应更加广泛的输入和场景.联邦学习[62-63]可以利用数据多样性和分散的模型训练方式高效地获得泛化性强且性能良好的模型,因此联邦学习成为了更新全局模型的有效手段.然而,利用联邦学习实现全局模型的更新演化在多终端模型演化系统中需要面对多终端异构导致聚合精度下降的挑战.

具体来说,终端异构主要体现在两个方面:首先是任务模型异构性,由于终端硬件资源的差异性,为了保证终端模型的推理性能,终端模型应针对终端硬件做出定制化适配处理.因此,部署在不同终端的模型结构或压缩程度各不相同.而传统的联邦学习算法,如 FedAvg[35]算法,是通过平均终端模型权重来进行模型聚合,因此难以处理异构终端模型的情况.而先前工作[37]通过全局模型个性化和学习个性化模型实现了异构模型融合,然而两种方法受制于

模型种类的限制或技术(如 MoE)的黑盒属性,并不利于后续的系统优化和加速;而数据分布异构性是联邦学习存在的普遍问题. 由于多终端所处场景的特异性,不同终端所捕捉到的数据一般是非独立同分布(non-iid)的,由于权重分歧,如果直接将由 non-iid 数据训练得到的终端模型进行聚合,则会导致较大的精度损失,致使生成的全局模型精度不佳[64-66]. 因此,设计一套可以克服多终端异构,实现快速有效的全局模型演化框架是必要且具有挑战性的.

3.2　多终端并发演化任务的服务器资源抢占和存算隔离导致的过高演化延迟

边缘服务器同时为多个移动终端提供模型演化服务,而 GPU 是目前应用于边缘服务器的最主流神经网络硬件加速器. 针对神经网络加速,英伟达公司在 GPU 上不断适配,增加了张量核心(tensor core)等模块. 因此,GPU 能作为大部分神经网络模型的硬件加速器,加速训练过程. 但是,由于目前 GPU 架构的局限性,在服务多终端应用时,GPU 存在着共享资源抢占和存算隔离两方面问题.

在面对伴随着激烈共享资源竞争的多终端模型演化情景时,基于民用和商用的主流 GPU 的硬件系统很难保证演化的低延迟. 具体来说,如今深度模型训练通常采用随机梯度下降法,每个训练周期会被分为一些小批量,而每个批量一般都应经历一个完整的数据流:向前传播计算损失,反向传播获得梯度以及基于梯度的参数更新. 因此,模型训练通常会占用大量的显存和计算资源等共享资源. 尤其当 GPU 服务来自多个终端的演化任务时,多终端模型演化任务彼此竞争有限的共享资源,降低了大部分任务的训练速度,甚至使某些任务陷入长时间停滞,这些都会导致不必要的模型演化延迟累积,使终端以低精度的旧模型长时间运行,影响平均推理精度.

此外,大批量训练可以帮助模型实现高精度和稳定收敛[67-68],然而,除了 H100 和 A100 等高成本 GPU 具有几十 GB 的大容量显存外,大部分商用和民用的主流常见 GPU(例如 RTX 4080、RTX 3080)的有限显存不足以支持深度学习模型大批量训练的需求,从而影响模型的演化效果.

另一方面,GPU 存算隔离的特点产生了大量不必要的访问延迟. 由于 GPU 芯片内部并没有大面积的存储结构,其片外内存(即显存)与计算单元相互隔离,而在数据量庞大的深度神经网络计算中,GPU 计算单元需要频繁读写内存,该部分占据了进程近 50%的执行时间[69],造成了大量计算之外的延迟. 并且,roofline 模型表示当计算平台没有达到最大计算强度时,其理论性能由硬件的带宽上限与模型自身的计算强度决定,而一般来说,部署于终端的轻量化模型计算强度相对较低,很容易陷入内存边界[70],在有限的显存带宽下,存算单元交互速度很难使 GPU 达到最大计算强度,如 7.3.1 节所示,小批量训练的速度受到显存带宽的影响,而短缺的显存容量很难支持大批量训练. 随着多终端模型演化系统服务的终端数量增加,数据量会急剧上升,造成演化延迟的急剧积累,从而导致终端模型长期陷于低精度推理,严重影响用户体验. 因此,基于单一硬件架构(例如 GPU)的边缘 AI 系统并不适合多终端演化服务的要求. 如何克服或避免共享边缘服务器在面向多终端演化服务时的资源瓶颈是加速模型演化的一大挑战.

4　NestEvo 系统设计

多终端视频流智能识别模型共进演化系统(NestEvo)的整体结构如图 1 所示. 本节基于问题分

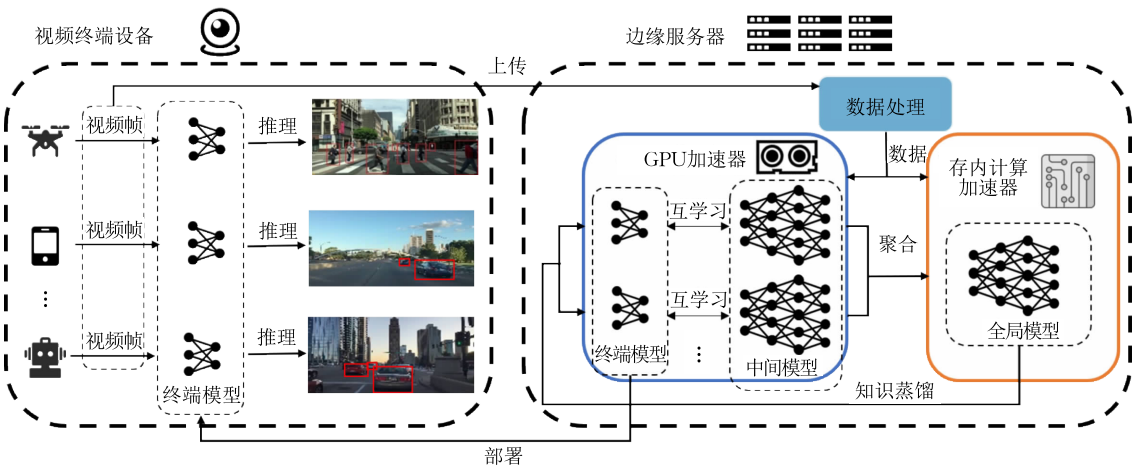

图 1　多终端视频流智能识别模型共进演化系统结构概览

析与建模，分别从系统的多终端互学习共进演化模块和多终端并发演化任务调度与存内计算加速模块对本文提出的多终端视频流智能识别模型共进演化系统进行介绍说明.

4.1 问题分析与建模

终端设备对于实时视频分析应用的需求愈发强烈，例如快递物流小车、自动跟踪无人机[71]和移动VR/AR[72]等，为了减少推理延迟和避免用户隐私问题，使用部署在终端的深度模型直接对其采集到的视频数据进行推理成为了实时视频分析应用的有效解决方案. 但是由于终端的硬件资源有限，我们只能在其上部署轻量化的深度模型. 而基于深度学习的终端视频分析不可避免地存在数据漂移问题并且数据漂移对结构简单的压缩模型的影响尤为明. 边缘辅助的模型在线演化是解决数据漂移问题的有效方法. 但是，由于真实场景中实时捕获到的数据是无标签的，因此模型演化系统通常利用知识蒸馏这种迁移学习方式进行知识的传递. 作为“学生”的终端模型通过作为“教师”的全局模型学习本地知识，可以有效提升终端模型的推理精度. 以无人机搜寻为例，如图2所示，无人机利终端模型进行实时视频推理，当面对数据漂移问题时，终端模型的推理精度会下降，而无人机将它们从多个视频流采集到的视频帧上传至边缘服务器端，利用知识蒸馏等技术在服务器端对多个无人机终端模型进行演化，提高终端模型在所处场景中的推理精度，之后将更新后的模型参数传回无人机进行部署，完成模型的在线演化. 而在以往的模型在线演化系统中，面对移动场景中多终端的复杂多变视频数据，作为“教师”的全局模型同样会受到数据漂移的影响，导致精度下降而影响系统整体性能.

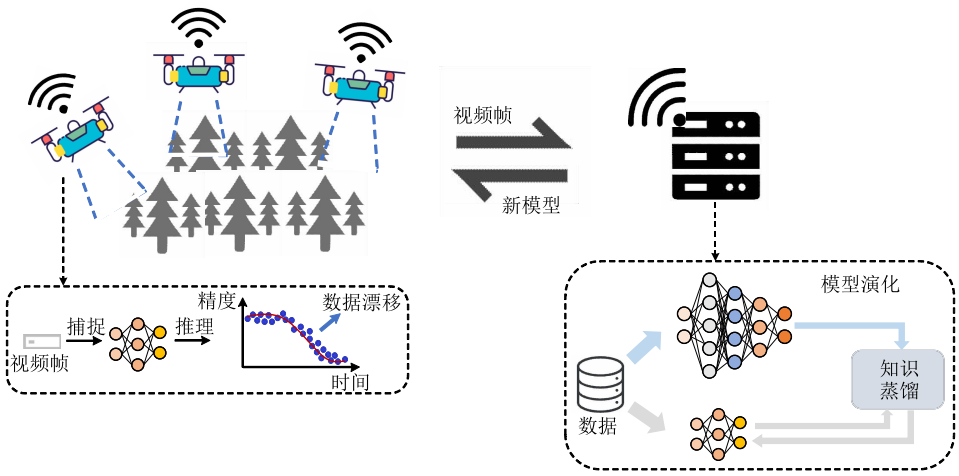

图2 边缘辅助的模型在线演化系统示意图

基于以上问题，本文提出了一种多终端视频流智能识别模型共进演化方法(NestEvo)，高效地实现了终端模型和全局模型的协同在线演化.

在算法层面，为了提高受到数据漂移影响的终端模型的推理精度，本文提出了多终端互学习共进演化方法，利用联邦学习技术校对全局模型的推理精度，并在此过程中利用数据生成以克服数据异构问题，并且最后利用基于本地数据的知识蒸馏技术，通过全局模型将知识传授给终端模型，从而最终提高终端模型的推理精度，提升用户的体验. 在此过程中，假设在实时视频分析场景中存在有 m 个移动终端设备(均配备有摄像头并且硬件资源受限)，移动终端模型集合表示为 $R=\{r_1,r_2,\cdots,r_m\}$，在相同且稳定的通信环境与延迟的条件下，上传的数据总量的最大值为 $D_{\max}$，距离相近的移动终端设备组成一个小组，m 个终端模型可以分为 n 个小组，小组集合表示为 $T=\{t_1,t_2,\cdots,t_n\}$，终端设备上传到边缘服务器的数据集合表示为 $D=\{d_1,d_2,\cdots,d_m\}$，每个小组的数据集表示为 $C=\{c_1,c_2,\cdots,c_n\}$，位于边缘服务器的全局模型集合表示为 $G=\{g_1,g_2,\cdots,g_m\}$. 而多终端互学习共进演化方法的最主要优化目标是：在移动终端捕捉到的新的数据集下，最大程度地提高终端模型的推理精度(式(1))，其中 $ACCURRCY_{\text{before}}(r_i-d_i)$ 和 $ACCURRCY_{\text{after}}(r_i-d_i)$ 分别为演化前后终端模型 r_i 在 d_i 数据集上的精度，并最大程度提升部署在边缘服务器上的全局模型的推理精度(式(2))，其中 $ACCURRCY_{\text{before}}(g_i-c_i)$ 和 $ACCURRCY_{\text{after}}(g_i-c_i)$ 分别为演化前后全局模型 g_i 在 c_i 数据集上的精度. 同时，需要满足以下约束条件：每个终端模型只属于一个小组(式(3))，每个小组与一个全局模型一一对应(式(4))，系统的通信代价满足要求(式(5)).

优化目标：

$$\text{Maximize}\ [ACCURRCY_{\text{after}}(r_i-d_i)-ACCURRCY_{\text{before}}(r_i-d_i)],$$

$$r_i \in R,\ d_i \in D \tag{1}$$

$$\begin{aligned}\text{Maximize}\ [&ACCURRCY_{\text{after}}(g_i - c_i) - \\ &ACCURRCY_{\text{before}}(g_i - c_i)],\\ &g_i \in G,\ c_i \in C\end{aligned} \tag{2}$$

约束条件：

$$r_x \in t_y,\ r_x \in R,\ t_y \in T \tag{3}$$

$$r_i \leftrightarrow g_i,\ r_i \in R,\ g_i \in G \tag{4}$$

$$\sum_{0}^{m} d_i \leq D_{\max},\ d_i \in D \tag{5}$$

在模型演化系统中，终端设备在利用终端模型进行实时视频推理任务的同时会定期向服务器发送演化请求，上一个演化请求到下一个演化请求之间的时间段被称为模型演化窗口，即终端模型的生命周期. 终端模型全生命周期推理性能与模型（LCP）演化速度密切相关，可以表示为

$$LCP_{\text{avg}} = \frac{ACCURRCY_{\text{before}}(r_i - d_i) \times t_{\text{evolving}} + ACCURRCY_{\text{after}}(r_i - d_i) \times (t_{\text{total}} - t_{\text{evolving}})}{t_{\text{total}}} \tag{6}$$

其中，t_{evolving} 为模型演化时间，t_{total} 表示模型演化窗口时间. 模型演化速度越快，新的高精度模型就可以更快地部署到终端，替代旧模型进行推理，进而终端的平均推理精度就会更高. 我们将长生命周期内的终端模型平均推理精度进一步进行细化：

$$LCP_{\text{avg}} = \frac{[ACCURRCY_{\text{before}}(r_i - d_i) + \Delta A] \times t_{\text{total}} - \Delta A \times (t_c + t_w + t_r)}{t_{\text{total}}} \tag{7}$$

$$LCP_{\text{avg}} = \frac{ACCURRCY_{\text{before}}(r_i - d_i) \times t_{\text{total}} + \Delta A \times (t_{\text{total}} - t_{\text{evovling}})}{t_{\text{total}}} \tag{8}$$

其中 ΔA 为终端模型的演化精度增益，t_c 为终端与服务器通信时间（上传视频帧和下载部署新模型），t_w 和 t_r 分别表示演化任务的等待时间和模型训练时间. 式(7)表示，演化任务的等待时间和训练时间越短，终端模型全生命周期推理性能越高. 并且，式(8)表示在优化式(1)的同时维持高且稳定的演化精度增益对于 LCP 也至关重要.

因此，在系统层面，本文提出了基于存内计算的模型演化训练加速方案，即为演化系统中的各个模型选择合适的训练加速方案，并根据模型所属的硬件特点优化模型，使软硬件更加适配，提升系统性能. 我们定义模型在线演化中的等待时间和训练时间长度分别为 t_w 和 t_r，模型经历一次演化后的精度提升为 ΔA，系统精度提升可接受的最大损失为 $\Delta A_{\max}$. 基于存内计算的模型演化训练加速方案的优化目标是最小化所有终端的平均等待时间和训练时间（式(9)）. 同时，需满足的约束条件是在线演化系统中模型演化带来的精度提升的损失在可接受的范围内（式(10)）.

优化目标：

$$\text{Maximize}\ \frac{1}{n}\sum_{i}^{n}(t_w^i + t_r^i) \tag{9}$$

约束条件：

$$\Delta A_{\text{before}} - \Delta A_{\text{after}} \leq \Delta A_{\max} \tag{10}$$

4.2　NestEvo 系统概述

NestEvo 包括两大核心模块：多终端互学习共进演化模块和多终端并发演化任务调度与存内计算加速模块. 如图 3 所示，多终端互学习共进演化模块

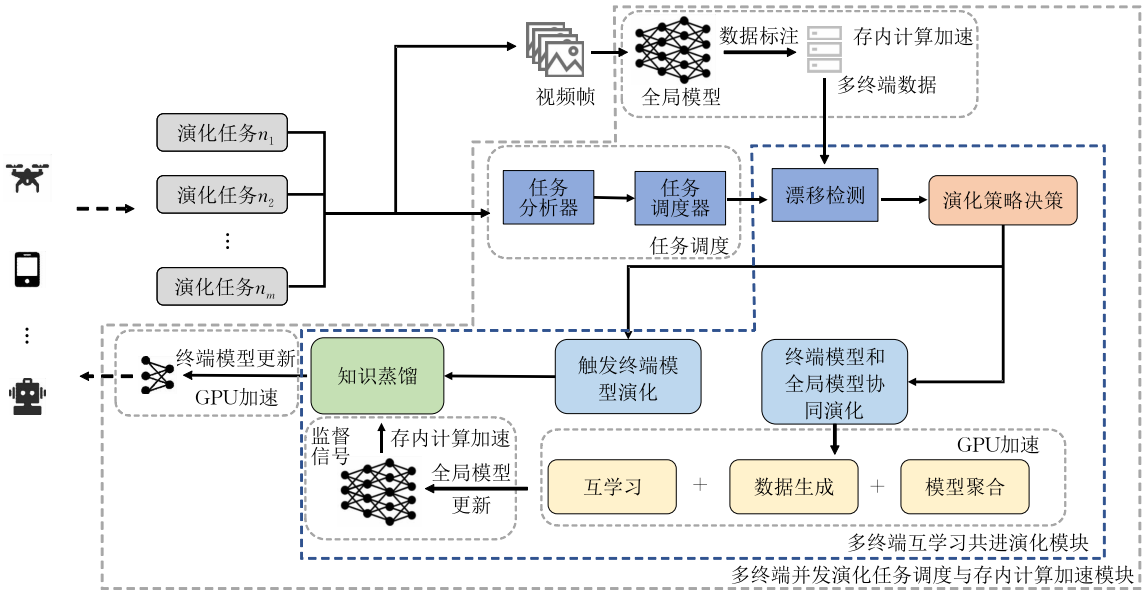

图 3　多终端互学习共进演化方法流程图

包括演化方法的决策和终端模型和全局模型协同演化,而多终端并发演化任务调度与存内计算加速模块负责通过多终端演化任务调度和基于存内计算的模型训练加速以降低模型演化延迟.本小节将对两大模块的工作内容进行详细的介绍说明.

4.2.1 多终端互学习共进演化模块

为了高效提升受到数据漂移影响后的终端模型的推理精度,本文采用的多终端互学习共进演化方法能够针对多终端数据漂移情况,自适应的触发仅终端模型演化或终端模型与全局模型协同在线演化.具体地,多个不同移动终端将筛选后的视频帧(筛选方法类似于自适应视频流[73]中的帧采样方法,即根据上一段视频内容的数据漂移程度调整该次演化时的帧采样率)上传至边缘服务器,由于较大的参数体积和较强的泛化能力,全局模型相比于终端模型对数据漂移有着一定的鲁棒性.为了避免过多的冗余更新,降低资源占用和演化延迟,我们基于漂移检测的结果使系统自适应地触发不同的演化策略:仅终端模型演化或终端模型和全局模型的协同演化.当触发仅终端模型演化时,只需进行知识蒸馏即可完成演化任务.而当触发终端模型和全局模型协同演化时,如图1所示,我们为每个终端模型配备一个与全局模型配置相同的中间模型,利用终端模型和中间模型的互学习首先完成中间模型的训练更新.在此过程中利用数据生成,减少各个终端数据特征分布之间的搬土距离,在保留原本数据特征的基础上,使每个终端模型-中间模型组合学习到的视频数据特征分布趋于近似,从而缓解数据分布异构的影响.之后将中间模型聚合生成全局模型完成全局模型的更新.中间模型的引入可以将终端模型和全局模型相互隔离,全局模型不再由终端模型聚合生成,因此,避免了任务模型异构的影响.最后,利用新的全局模型为终端模型进行知识蒸馏即可完成全部演化任务.

4.2.2 多终端并发演化任务调度与存内计算加速模块

为了尽快完成新的高精度终端模型的演化部署,本文为基于互学习的多终端压缩模型在线演化系统设计了一套基于存内计算的模型演化异构硬件加速方案以降低演化延迟,提高终端模型的平均推理精度.首先,NestEvo针对并发的多终端演化训练任务进行任务调度,在充分利用边缘服务器有限硬件资源的同时减少所有任务的平均等待时间.具体地,任务调度模块中包括任务性能分析器和任务调度器.NestEvo以任务训练特征(批量大小、训练周期等)为输入,利用任务分析器可以预测得到每个任务的性能指标(显存占用和训练时间).并且,以此作为任务调度器的输入,任务调度器可以筛选出优先级最高的任务组合率先提供演化服务.每当有任务完成时,任务调度器会从等待任务中选择新的任务组合,直到所有任务完成为止.

基于多终端演化任务调度的输出结果,NestEvo提出了基于存内计算的异构模型演化训练加速方案以缩减多终端并发演化任务的训练时间,从而进一步减少演化延迟.具体地,对于传统的GPU加速器而言,当触发仅终端模型演化时,只需进行知识蒸馏即可完成演化任务,此时NestEvo利用GPU加速终端模型的训练速度,实现终端模型的稳定有效演化.而当触发终端模型和全局模型协同演化时,GPU负责终端模型和中间模型的互学习以及终端模型的知识蒸馏训练加速.与此同时,我们对GPU加速训练中的数据进行自适应压缩,减少数据冗余,从而降低了GPU显存开销,减少了GPU加速训练的演化延迟.

存内计算加速器负责全局模型的输出加速,并利用模型训练优化提升全局模型的权重噪声鲁棒性.一方面,多终端在复杂移动场景中实时捕捉到的视频数据是无标签的,当它们被上传到服务器后,需要利用全局模型的推理结果作为提供监督信号,即数据伪标签或软标签(soft target)供给GPU加速的模型进行训练学习.因此,存内计算可以利用高效的矩阵乘法大幅度加快全局模型推理速度,更快为终端模型和中间模型训练提供监督信号,加快系统整体演化速度.另一方面,存内计算在实现高效推理加速的同时,模型推理会受到电导噪声的影响,导致精度下降.我们利用互学习特性以及模型训练优化手段提升了演化后新的全局模型的权重噪声鲁棒性,降低了全局模型的推理精度损失并提高了输出稳定性,使软硬件更加适配,提高系统的整体性能.通过对此训练加速方案的理论分析和模拟验证,指导了在未来模型在线演化系统中边缘服务器的异构加速器的设计和优化.

5 多终端互学习共进演化方法

本节将对多终端互学习共进演化方法进行详细的解释说明,该方法包括演化策略决策以及终端模型和全局模型协同演化.该方法通过此方法可以实

现终端模型和全局模型在线演化，提升受到数据漂移影响后的终端模型的推理精度.

方法设计. 为了提升终端模型的演化效果，多终端互学习共进演化方法根据终端实时数据漂移程度自适应地触发仅终端模型演化或终端模型和全局模型协同演化，以高效完成模型演化任务，提升终端模型推理精度. 面对多个不同终端上传至边缘服务器的无标签视频数据，首先利用全局模型对其进行标注，并分别统计各个终端数据以目标识别类别划分的数据特征. 为了避免过多的全局模型更新，我们以全局视频数据漂移程度作为指标使系统自适应地选择是否进行全局模型更新. 具体地，我们计算终端模型 i 对应的实时视频数据与全局模型训练数据之间的数据偏差 δ_i：

$$\delta_i=\sum_{c=1}^{C_i}\left\|\frac{g_\varphi(X^c_{\text{traini}})-g_\varphi(X^c_{\text{testi}})}{\sigma(g_\varphi(X^c_i))}\right\|_2 \tag{11}$$

其中，C_i 为终端模型 i 目标识别的类别数，$g_\varphi(X^c_{\text{traini}})$ 和 $g_\varphi(X^c_{\text{testi}})$ 分别表示终端模型 i 训练数据和实时捕捉到的真实数据的平均特征，$X^c_i=X^c_{\text{traini}}\cup X^c_{\text{testi}}$，$\sigma(\,)$ 表示用于特征归一化的标准差，$\|\ \|_2$ 表示 L2 范数. 以此作为输入，计算全局数据漂移程度 δ_{global}：

$$\delta^i_{\text{global}}=\begin{cases}1, & \delta_i>\gamma\\ 0, & \text{其他}\end{cases} \tag{12}$$

$$\delta_{\text{global}}=\sum_i^n\delta^i_{\text{global}},\ \delta^i_{\text{global}}\geqslant 0 \tag{13}$$

其中，γ 为数据偏差阈值，n 为终端数量，当终端模型实时推理的视频数据与全局模型训练数据之间的偏差 δ_i 大于 γ 时，我们认为此时发生了较为严重的数据漂移. 而当 δ_{global} 低于预先设定阈值 ϵ 时，说明此时的数据偏移在全局模型可接受范围内，全局模型依旧可以提供高质量的监督信号，因此，我们只进行终端模型的更新，即利用当前的全局模型对终端模型进行知识蒸馏，完成终端模型的演化任务.

而当 δ_{global} 高于 ϵ 时，此时数据漂移对全局模型具有较大的影响，需要对全局模型进行演化更新以保证其提供给终端模型的监督信号质量，因此，我们需要进行终端模型和全局模型的协同演化. 由于位于边缘服务器的全局模型为所有终端模型提供监督信号，对于全局模型而言，所有终端模型都是平等的，所以全局模型需从所有终端捕获的数据中收集知识. 此外，多终端数据之间相互关联，并且由于移动情景变换具有随机性，终端未来数据之间的相关性存在不确定性. 因此，需要所有终端模型均参与共进演化过程以获得稳定且优秀的演化性能增益. 为此，我们首先需要完成全局模型的更新演化. 面对多终端异构导致的精度损失的挑战，我们引入数据生成[74]和联邦互学习[75]. 具体地，在全局模型完成终端数据标注后，我们引入如图 4 所示的数据生成器 G_ω，其中 θ_p 代表模型的预测模块，Y 代表数据的标签. 文献[74]表示利用数据的标签和模型的预测模块可以学习得到一个生成器，该生成器可以获得全局的特征分布. 我们学习得到这样一个生成器并将其广播给各个终端模型和中间模型，终端模型和中间模型可以在特征分布上采样得到增广样本，辅助其进行互学习. 通过数据生成可以使不同终端模型的训练数据分布趋向于整体数据分布，在保留原本数据特征的基础上，减少不同终端模型训练数据分布之间的搬土距离，在减少因数据分布异构带来的全局模型精度下降的同时，扩展数据分布，增强模型的泛化能力. 算法 1 阐述了基于数据生成的数据异构消除算法具体步骤. 首先根据终端数据的样本标签和中间模型预测模块更新生成器，得到每个终端相应的数据分布和全局数据分布，之后将该生成器广播给各个终端模型-中间模型组合，所有终端模型-中间模型并行地在全局数据分布进行采样得到增广样本并更新其原本数据集.

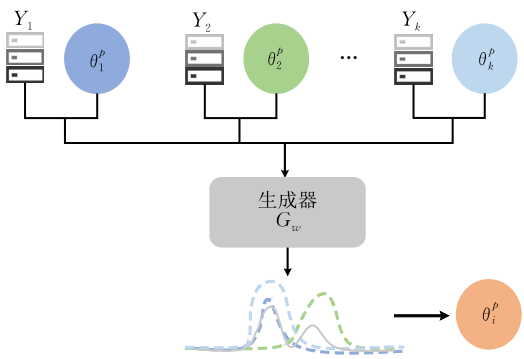

图 4　服务于互学习以克服数据异构的数据生成器工作原理图

算法 1.　基于数据生成的数据异构消除算法.

输入：终端模型的样本标签 $Y_i(i=1,2,\cdots,k)$，中间模型预测模块 $\theta_{ip}(i=1,2,\cdots,k)$

输出：新数据集

1. 根据样本标签和输入模块更新 G_ω，得到特征分布
2. FOR 每个终端模型 k in parallel do
2.　　根据特征分布进行采样
3.　　更新数据集
4.　　根据新数据集更新模型
5. END FOR

数据处理完成后，我们对全局模型进行演化更新. 由于任务模型异构的影响，先前的联邦学习工

作[37,42-44]难以对不同结构和压缩程度的定制化终端深度模型进行高性能聚合的同时足够透明，从而便于研究者们对系统进行性能调优.因此，我们引入中间模型并采用互学习的方式辅助完成全局模型的演化更新.如图 2 所示，中间模型结构与全局模型相同，中间模型在系统中主要起到两点作用：首先，中间模型将终端模型和全局模型相互隔离，全局模型不再由终端模型聚合而是由与其结构相同的中间模型聚合，从而避免了任务模型异构带来的问题；其次，多个中间模型并行训练，使全局模型的演化更加高效.我们令终端模型与其对应的中间模型基于更新后的数据集进行互学习，互学习中终端模型和中间模型的损失函数改写如下：

$$L_{\text{local}}=\alpha L_{C\text{local}}+(1-\alpha)D_{\text{KL}}(P_{\text{mid}}\|P_{\text{local}}) \quad (14)$$

$$L_{\text{mid}}=\beta L_{C\text{mid}}+(1-\beta)D_{\text{KL}}(P_{\text{mid}}\|P_{\text{local}}) \quad (15)$$

其中 α 和 β 是超参数，用于控制来自实时视频数据和其他模型知识的比例，$L_{C\text{local}}$ 和 $L_{C\text{mid}}$ 分别为终端模型和中间模型基于数据标签的损失函数，P_{local} 和 P_{mid} 分别是终端模型和中间模型的推理结果.在互学习之后，中间模型进行聚合得到全局模型，完成对全局模型的演化更新.互学习在模型演化系统中具有四大优势：首先，神经网络输出的类别概率估计会在一定程度上恢复出数据中不同类别之间的联系信息，因此网络之间类别估计的交互可以传递并学习数据分布特性，从而提高模型的泛化性；其次，互学习会起到正则化作用，真值标签的独热编码会使模型在训练过程中对预测结果太过确信，容易导致模型过拟合，而利用互学习学习彼此的类别概率，可以有效防止此现象；然后，深度模型在训练过程中参考其他模型的学习经验来调整自己的学习过程，可以使结果收敛到一个更平缓的极小值点，使模型具备更好的泛化能力，并降低模型对于噪声的敏感度，这使得系统的训练加速方案有了更多的选择；最后，互学习作为由知识蒸馏衍生出的一种涉及到多个模型协作演化更新和的技术，它可以在终端模型和全局模型隔离的情况下，通过终端模型和中间模型并行的交流学习以及中间模型的聚合，使全局模型完成演化更新.

而在完成全局模型演化更新之后，为了使终端模型拟合其所处场景的视频数据，实现实时高精度推理，我们基于每个终端上传至服务器的原始视频数据，而不采用数据生成后的新数据，利用新的全局模型和与其对应的多个终端模型进行知识蒸馏.其中新全局模型作为“教师”模型，向作为“学生”的终端模型传授知识，从而得到新的终端模型.最后，将终端模型参数传回至终端，从而完成终端模型的在线演化.基于互学习的终端压缩模型在线演化系统的整体算法如算法 2 所示.

算法 2. 基于互学习的终端压缩模型在线演化算法.

输入：本地视频数据(X_i)，旧终端模型 $Local_t^k$，旧中间模型 Mid_t^k，旧全局模型 $global_t$

输出：新终端模型 $Local_t^{final}$，新中间模型 Mid_t^{k+1}，新全局模型 $global_{t+1}$

1. 利用 $global_t$ 为本地视频数据进行标注，得到(X_i,Y_i)
2. 基于(X_i,Y_i)得到每个终端的数据偏差以及全局数据漂移程度
3. IF $\delta_{\text{global}}<\epsilon$ DO
4. 基于(X_i,Y_i)利用 $global_t$ 对 $Local_t^k$ 进行知识蒸馏
5. 更新得到 $Local^{final}$
6. ELSE DO
7. 根据样本标签和输入模块更新 G_ω，得到特征分布
8. FOR 每个终端模型 k 和每个中间模型 k in parallel DO
9. 根据特征分布进行采样
10. 更新数据集
11. 基于新数据集对终端模型 $Local_t^k$ 和中间模型 Mid_t^k 进行深度互学习
12. 更新得到 $Local_{t+1}$ 和 Mid_{t+1}
13. END FOR
14. Merge：$global_{t+1}\leftarrow\frac{1}{K}\sum_{k=1}^{K}Mid_{t+1}^k$
15. 基于(X_i,Y_i)利用 $global_{t+1}$ 对 $Local_{t+1}$ 进行知识蒸馏
16. 更新得到 $Local^{final}$

6 多终端并发演化任务调度与存内计算加速

本节将详细介绍说明系统中多终端并发演化任务调度与存内计算加速.本文通过多终端并发演化任务调度与存内计算技术缩短了多终端模型演化的平均等待时间和训练时间，加快了演化后高精度模型的部署，减少低精度模型的推理时间，从而提升多终端模型在模型演化窗口时间中的平均推理精度.具体地，本文首先利用基于多终端并发演化任务优先级的任务调度，筛选出部分演化任务优先进行以减少所有任务的平均等待时间，缩短演化延迟.之后利用存内计算的训练加速，减少模型的训练时间.在此过程中，我们利用软硬件结合的思想，基于系统中

不同模型的训练推理特点为其选择适合的训练加速方案.并且,利用基于视频数据的自适应数据压缩机制和模型抗噪声训练优化对模型进行优化调整,使软硬件更加适配.通过该方案的模拟验证,可以为未来模型演化系统中边缘服务器的异构训练加速器设计提供指导.

6.1　多终端演化任务调度

多终端演化任务调度包括用于预测任务性能指标的任务分析器(第 6.1.1 节)以及用于动态选择任务组合的任务调度器(第 6.1.2 节).

6.1.1　任务分析器

边缘服务器有限的硬件资源很难同时服务多个任务演化,即某些任务会因为可用资源不足而处于等待停滞的状态.为了减少所有任务的平均等待时间,需要实现准确有效的动态任务调度,根据每个演化任务的性能指标确定任务优先级并制定合理的任务调度方案.而让用户自行标注任务的各项性能是不准确和不公平的.因此,本文在边缘服务器端提供了一个任务分析器用于为每个演化任务的性能指标提供精确预测分析,包括任务的显存占用和单位计算单元下的训练时间.

(1) 显存占用.对于传统的存算隔离的 GPU 加速器而言,每个演化任务的显存占用主要分为五个部分:模型的参数 m_p,中间结果 m_f,向后传播梯度 m_g,优化器参数 m_{opt} 以及库函数所需的 m_{ws}[76].因此,第 i 个演化任务的显存占用 $M_i=m_p+m_f+m_g+m_{opt}+m_{ws}$.具体地,模型参数占用 m_p 是所有层(包括卷积层、全连接层和 BN 层)的参数量乘位宽,举例来讲,对于输入输出通道数分别为 C_{in},C_{out},卷积核为(K_1,K_1),步长为 s,填充为 p 的卷积层而言,其显存占用 $m_{pconv}=(C_{\text{in}}\times C_{\text{out}}\times K_1\times K_1)\times B_p$,位宽 B_p 由参数量化决定.中间结果的显存占用也可以直接通过计算得到,对于上述卷积层而言,若其输入为 $w_1\times h_1\times d_1$ 的张量,则其输入的中间结果显存占用 $m_{fconv}=(w_1-K_1+2\times p)/(s+1)\times(w_1-K_1+2\times p)/(s+1)\times B_p\times Batch\_size$.由于梯度是由中间结果产生的,因此梯度的显存占用与中间结果的保持一致,即 $m_g=m_f$.而优化器用于更新模型参数,并且不同优化器的显存占用不同,对于具有一阶动量的优化器,其显存占用为 $m_{opt}=2\times m_p$,而无动量优化器的显存占用 $m_{opt}=m_p$.最后,我们依据经验将 CUDA11.1 和 CuDNN8.0.5 下的 m_{ws} 设置为 850 MB.

(2) 训练时间.我们利用具有三个全连接层的神经网络回归模块来建立重训练时间和任务特征之间的映射关系.影响任务重训练的时间的因素主要包括模型的参数量、重训练数据量、重训练周期以及批量大小.我们收集了 200 个不同设置下的样本以及对应的重训练时间以离线训练回归网络,从而获得紧凑而高效的训练时间预测模块.

6.1.2　任务调度器

任务调度器以任务分析器输出为输入,将演化任务的资源需求与边缘服务器的资源供给进行匹配,筛选出最优的任务组合以减少所有并发任务的平均等待时间,缩短演化延迟.任务调度器的设计基于我们的两点见解:首先,尽可能充分利用可用的硬件资源可以最大化边缘服务器的吞吐量,从而加快所有任务的平均完成速度;其次,利用短作业优先原则可以有效减少所有任务的平均等待时间.因此,边缘服务器需要从所有并发演化任务中选择最优的任务组合以获得最短的任务平均等待时间.我们将演化任务的选择问题定义为了一个背包问题[77],即在不超出背包最大容量的情况下,打包最有价值的物品组合.我们定义了时间价值 α/t_r^i 作为物品价值,这表示更短的任务会获得更高的优先级.为了高效解决该问题,我们利用动态规划算法,将从所有任务中的最优组合选择问题分解为具有任务子集的子问题,如算法 3 所示.我们定义了一个二维数组 $dp[k][rm]$,每个 $dp[k][rm]$ 表示在剩余显存 rm 下从 k 个任务中选择的最佳任务组合的价值.对于每个 $dp[k][rm]$,如果第 k 个演化任务的显存需求高于现有的可用显存,则第 k 个任务无法被选择.否则,需要进一步比较选择任务 k 和不选择任务 k 两种任务组合下的物品价值,并将最大价值对 $dp[k][rm]$ 进行更新.我们对 $dp[k][rm]$ 进行迭代更新直到获得 $dp[N][M_c]$,即在边缘服务器剩余总显存 Ms 下从所有 N 个任务中选择出的最优演化任务组合具备的最大时间价值,之后便可 以获得表示任务组合的 n 元向量(φ_1,φ_2,…,φ_N).在每次有任务完成后,我们继续利用动态规划算法选择新的任务组合,直到所有任务完成为止.

算法 3.　多终端并发演化任务调度算法.

输入:演化任务(r_1,r_2,…,r_N),每个任务的预测训练时间(t_r^1,t_r^2,…,t_r^N),每个任务的时间价值(v_1,v_2,…,v_N),每个任务的显存占用(M_1,M_2,…,M_N)和 GPU 可用显存资源 M_c

输出:任务组合(φ_1,φ_2,…,φ_N)

1. 创建 $dp[0:N][0:M_c+1]$
2. 利用 M_1,v_1 初始化 dp
3. FOR 演化任务 k 从 1 到 $N-1$ DO

```
FOR 剩余显存 rm 从 1 到 M_c DO
  IF M_k>rm THEN
    dp[k][rm]←dp[k−1][rm]
  ELSE
    dp[k][rm]←MAX(dp[k−1][rm],
                  dp[k−1][rm−M_k]+v_k)
  END FOR
END FOR
初始化(φ_1,φ_2,…,φ_N)←0
WHILE N>1 DO
  IF dp[N−1][M_c]!=dp[N−2][M_c] THEN
    φ_{N−1}←1, M_c←M_c−M_{N−1}
  N←N−1
IF dp[0][M_c]>0 THEN
  φ_0←1
```

6.2 基于存内计算的多终端演化任务训练加速

基于多终端演化任务调度的输出结果，NestEvo 提出了基于存内计算的异构模型演化训练加速方案以缩减多终端并发演化任务的训练时间，从而进一步减少演化延迟，提高多终端模型全生命周期推理性能，如图 5 所示. 通过此方案可以为未来模型在线演化系统中边缘服务器的异构加速器的设计提供指导. 具体来说，我们针对模型训练推理特点为系统的不同模型选择了合适的训练加速方案：由于终端模型和中间模型之间进行互学习训练，因此，终端模型和中间模型部署在模型演化主流且稳定的 GPU 上；而全局模型在系统中的主要任务是为上传到边缘服务器端的视频帧进行标注并在知识蒸馏过程中提供 soft target，所以全局模型在系统中只进行推理. 并且基于模拟计算的存内计算加速器的高效乘加运算对于仅推理的深度学习应用有着极大的加速效果. 因此，我们将全局模型部署在可以大幅度加快推理速度的存内计算加速器上以更快地为互学习和知识蒸馏过程提供监督信号，从而加快整体演化速度. 此外，我们针对各个硬件的特点，对部署在其上的神经网络模型进行了优化，使系统获得更好性能.

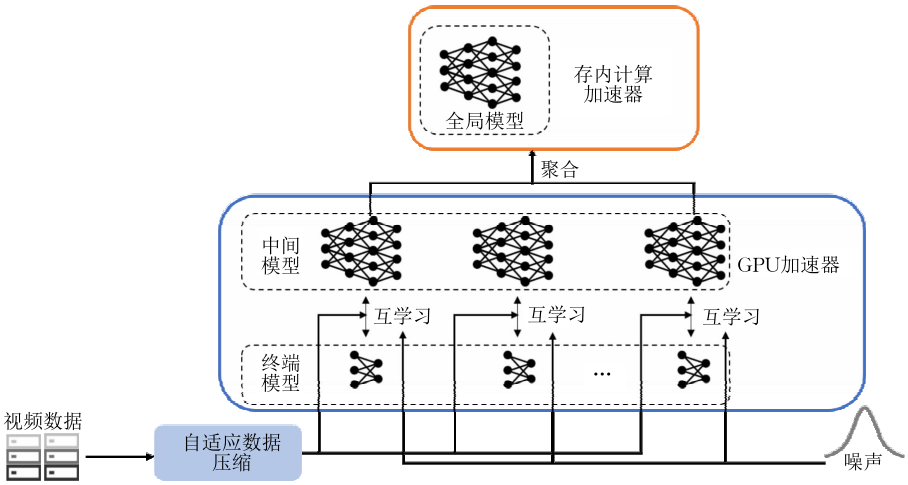

图 5 基于存内计算的多终端模型演化训练加速方案示意图

对传统的存算隔离的 GPU 加速器而言，深度模型运行过程中的显存占用主要分为四个部分：模型权重、权重梯度、中间结果和库函数所需的 workspace，其中模型演化过程中产生的中间结果占据了主要部分[76]，并且互学习的引入加大了 GPU 的显存负载，因此，针对存算隔离硬件架构特点，对 GPU 上进行的模型训练过程进行显存削减是必要的. 并且，演化任务在 GPU 上的训练时间也与其计算量和计算强度有着密切联系. 因此，基于其数据存取特点和显存分配规律，我们利用基于视频数据的自适应的数据压缩技术对部署在 GPU 的神经网络模型的输入数据进行一定处理：在全局模型完成对终端模型上传的视频帧推理标注之后，若视频帧不存在小目标物体，则使视频帧通过池化层，对视频帧进行压缩；若视频帧存在小物体，为了保证演化效果，则不压缩视频帧. 在这种优化方案下，模型的精度并不会产生明显的下降，而其显存占用明显减少. 并且该方法减少了模型运行过程中的中间结果，降低了演化的运算量，从而加快了演化任务的训练过程，缩减演化延迟，优化系统的性能.

对于存内计算加速器而言，其实现的高效矩阵乘法如图 6 所示. 输入 $\boldsymbol{x}$ 矩阵与权重矩阵 $\boldsymbol{g}$ 相乘，得到结果 $\boldsymbol{y}$ 矩阵. 在存内计算加速器中，输入 $\boldsymbol{x}$ 矩阵转变为模拟量电压 $\boldsymbol{V}_n^{\mathrm{T}}$，而权重则以电导的形式存储在忆阻器 $G_{m\times n}$ 中，由安培定律和基尔霍夫定律可以得出：

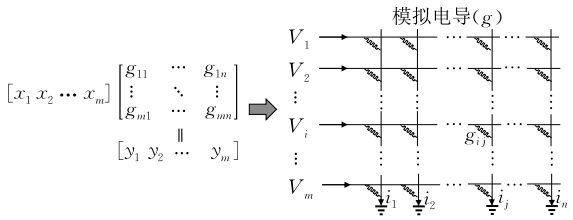

图 6　模拟加速器示意图

$$i_j=\sum_{i=1}^{M} g_{i,j}\cdot v_i \tag{16}$$

$$\boldsymbol{I}_n^{\mathrm{T}}=\boldsymbol{V}_n^{\mathrm{T}}\times G_{m\times n} \tag{17}$$

可以看到，模拟电流量 $\boldsymbol{I}_n^{\mathrm{T}}$ 即为我们所需得到的结果 $\boldsymbol{y}$ 矩阵. 之后，通过模数转换，并加上偏置，代入激活函数后，即可得到所需的最终结果. 由此可见，存内计算加速器的推理速度比 GPU 中重复进行的乘加操作快很多. 因此，存内计算加速器对于仅推理的深度学习应用具有巨大的吸引力，可以加快数倍模型推理速度.

而在模型演化系统中，全局模型主要进行推理工作，因此，将全局模型部署在存内加速器上可以使其更快为互学习和知识蒸馏过程提供监督信号，加速模型演化速度. 但是，存内计算加速器中新型存储器产生的电导噪声会影响模型的推理精度，从而影响全局模型提供的监督信号质量，损害终端模型的演化学习效果. 因此，我们从模型训练过程入手，对全局模型进行优化，增加其对于权重噪声的鲁棒性，保证其推理结果的稳定性和准确性. 一方面，我们采用的互学习训练方式以及数据生成可以增强模型的泛化能力，使模型收敛到更为平缓的极小值点，从而加强模型的噪声鲁棒性，维持原本的推理精度；另一方面，我们在终端模型和中间模型的互学习过程中注入与存内计算加速器电导噪声相似的权重噪声，使中间模型适应噪声的影响. 具体而言，由于新型存储器中的电导噪声基本服从于正态分布，我们在互学习向前传播过程中，令第 l 层的权重满足：

$$W_l\in N(W_{l_0},\sigma_{N,l}^2) \tag{18}$$

$$\sigma_{N,l}=\mu\times(W_{l_{\max}}-W_{l_{\min}}) \tag{19}$$

为了防止权重选择幅度过大，影响模型训练效率和精度，我们对于权重进行了进一步的约束：

$$W_{l_{\min}}\leq W_l\leq W_{l_{\max}} \tag{20}$$

其中，$N(\ )$ 表示正态分布，W_{l_0} 为注入噪声之前第 l 层权重，$\sigma_{N,L}$ 为注入的噪声，μ 为噪声系数，$W_{l_{\max}}$ 和 $W_{l_{\min}}$ 分别为第 l 层的最大权重和最小权重. 通过该训练方式可以加强聚合后全局模型对于权重噪声的鲁棒性，减少在存内计算加速器上进行推理预测时的精度损失，保证推理输出产生的监督型号的稳定性和准确性，从而确保终端模型的演化效果，提升终端模型的精度增益.

7　实验验证

本节首先介绍实验设置，并分别对多终端互学习共进演化方法进行实验验证和基于存内计算的模型演化训练加速方案进行模拟计算验证，并结合实验结果进行分析.

7.1　实验设置

本文采用 BDD100K[78] 数据集作为终端模型和全局模型的预训练集，并使用在 VOC[79] 数据集下精度超过 80% 的 Faster-RCNN[80] 网络对终端捕捉到的视频帧进行标注. 我们利用终端设备拍摄了五组不同帧率、场景下时长为 20 min 的视频，作为真实数据对系统进行性能测试和在线演化，如图 7 所示. 视频集的详细信息如表 1 所示.

图 7　原型系统原理演示图

表 1　移动终端采集视频数据详细信息

视频	帧率	时长/min	场景
A	30	10	白天校园
B	20	10	白天校园
C	10	10	白天校园
D	30	10	白天乡村
E	30	10	夜晚校园

本文在实验过程中，终端模型、全局模型和中间模型我们均使用以 ResNet-50 为骨干网络的 Faster-RCNN 模型，而终端模型我们使用不同剪枝率的模型剪枝进行模型压缩处理. 除仿真外，所有实验都是在 RTX3080 10 GB 上进行.

本文采用以下四种基准算法进行对比实验：

(1)域自适应(A1)[81]:利用基于对抗训练的域自适应方法对终端相同配置的 Faster-RCNN 模型进行训练更新.

(2)无数据生成的多终端视频流智能识别模型共进演化方法(A2):相比于 NestEvo,该方法去除了数据生成,直接使用特征分布异构数据进行模型训练聚合生成全局模型,并利用该全局模型对终端模型进行知识蒸馏.

(3)原始终端模型(A3):采用由 BDD100K 预训练得到的剪枝后的 Faster-RCNN,并不进行任何演化更新操作.

(4)独立的多终端模型演化(A4):利用原始的全局模型对终端 Faster-RCNN 模型直接进行知识蒸馏,实现终端 Faster-RCNN 的更新.

对于边缘服务器 GPU 模块,本文采用两种任务执行方案与 NestEvo 进行对比实验:

(1)GPU 并行执行(B1):随机选取演化任务放入 GPU 中进行服务,当有任务完成后,再次随机选取任务进行服务,直到所有等待任务完成.

(2)GPU 串行执行(B2):GPU 按照演化任务到达边缘服务器的顺序串行执行演化任务服务.

对于终端模型,本文使用三种不同剪枝率的 Faster-RCNN 进行部署:

(1)剪枝率 30%的 Faster-RCNN(M1);

(2)剪枝率 50%的 Faster-RCNN(M2);

(3)剪枝率 70%的 Faster-RCNN(M3).

7.2 多终端互学习共进演化方法精度增益性能

多终端互学习共进演化方法的优化目标是最大化终端模型的推理精度,同时提升部署在边缘服务器上的全局模型的推理精度.因此,本次实验首先检验了多终端视频推理中,数据漂移对于终端模型和全局模型的精度影响以及多终端数据异构对于模型演化的影响.最后,通过实验对比分析了 NestEvo 与四种基准算法的性能.

7.2.1 数据漂移对压缩模型的影响

多终端模型演化场景中的数据漂移对部署于终端设备的压缩终端模型和位于边缘服务器端的全局模型都有着较为严重的影响.本实验主要验证数据漂移问题对于深度模型精度的影响,首先终端部署的终端模型和边缘服务器端部署的全局模型用 BDD100K 数据集进行预训练并进行精度测试,之后,我们利用多终端采集到的真实视频数据对终端模型和全局模型的精度再次进行测试并进行两次测试的精度对比.在该实验中使用的训练方法为默认训练方法.实验结果如图 8 所示.

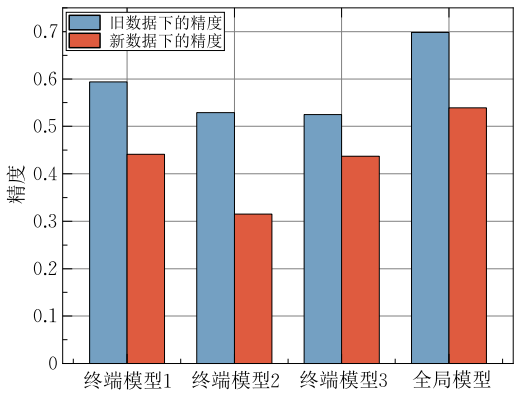

图 8 数据漂移对模型精度的影响

从图 8 中可以看到预训练后的全局模型在 BDD-100K 数据下的精度可以达到 69.8%,而在受到数据漂移影响后,其精度下降到了 53.9%;三个终端模型在 BDD100K 数据集下的精度均超过 50%,最高为 59.4%,而其在真实视频数据下的精度则均不足 50%,最低为 31.5%.由此可见数据漂移对于深度模型精度的危害是致命的.

7.2.2 多终端模型在不同数据下的演化表现

多终端模型异构是提升聚合后全局模型的精度以及演化系统整体性能的重要挑战.本实验探究了不同压缩程度的终端模型(M1、M2 和 M3)在不同视频数据下(A、B、C、D 和 E)的演化性能表现以及不同终端模型组合聚合后全局模型的推理精度性能.在此实验中,我们利用五个不同的视频数据集分别对三个不同压缩程度的终端模型进行基于知识蒸馏的演化更新,并测试了演化后终端模型的精度.之后,将基于不同视频数据训练的不同终端模型组合利用 FedAvg 算法进行聚合,得到新的全局模型,并基于终端数据测试聚合后全局模型精度并进行对比分析.实验结果如表 2 所示.

表 2 多终端模型在不同移动数据上的演化精度表现

实验编号	终端模型			数据			演化后终端模型精度($IoU=0.50$)			聚合全局模型精度($IoU=0.50$)
	终端 1	终端 2	终端 3	终端 1	终端 2	终端 3	终端 1	终端 2	终端 3	
1	M1	M1	M1	A	A	A	0.504	0.504	0.504	0.504
2	M1	M1	M1	A	B	C	0.504	0.498	0.494	0.497
3	M1	M1	M1	A	D	E	0.504	0.720	0.308	0.538
4	M2	M2	M2	A	D	E	0.400	0.540	0.205	0.317
5	M3	M3	M3	A	D	E	0.285	0.465	0.132	0.237
6	M1	M2	M3	A	A	A	0.504	0.400	0.285	0
7	M1	M2	M3	A	D	E	0.504	0.540	0.132	0
8	M1	M2	M3	D	E	A	0.720	0.205	0.285	0

通过实验 1 和实验 2 可以看出，视频的帧率（视频帧数量）对于模型演化精度增益会产生一定的影响，一般来说，视频帧数量越大，模型获得的精度增益越大，但与此同时会引入较大的演化延迟. 实验 3～5 展示了不同压缩程度的终端模型在不同场景数据下会呈现出不同的推理精度性能. 可以看到，随着剪枝率的提升，模型在不同场景下的推理精度都有着不同程度的下降，例如在 A 场景（白天校园环境）中，M3 模型相比于 M1 模型推理精度下降了 43.45%，这是模型压缩带来的固有缺陷. 并且，随着剪枝率的上升，模型在不同场景下的推理精度波动幅度更大，M1、M2 和 M3 在不同场景下的最大推理精度波动分别为 57.22%、62.03%和 71.6%，这是压缩模型或轻量化模型的参数稀疏性导致的模型泛化能力下降. 因此，面对数据分布存在未知性和动态性的移动场景中，终端稀疏模型需要通过边缘辅助的在线模型演化维持高精度推理状态，满足用户需求. 而实验 6～7 通过基于不同数据集训练得到的终端模型组合进行聚合，表现出由不同终端模型直接聚合更新全局模型是不现实的，即多终端模型异构给终端模型和全局模型的协同在线演化带来了不可忽视的困难和挑战.

7.2.3　多终端的数据分布异构性的影响

多终端的数据分布异构对于聚合后全局模型的精度损失同样有着重要影响，也是实现终端模型和全局模型协同演化面临的难题. 本实验进一步探究不同终端实时采集的视频数据之间的数据特征分布差异以及该差异对于聚合生成全局模型精度的影响. 首先，我们将四个移动终端放置于白天校园场景中，并对它们采集到的视频数据进行特征提取，相同时间段内，四个终端对应的数据特征分布如图 9 所示. 从图 9 中可以看到，实时视频数据特征分布大多呈现正态分布，但是其均值和方差各不相同，之后，我们计算两两终端之间数据特征分布的 KL 散度以显示不同终端数据特征分布之间的差异，结果如图 10 所示.

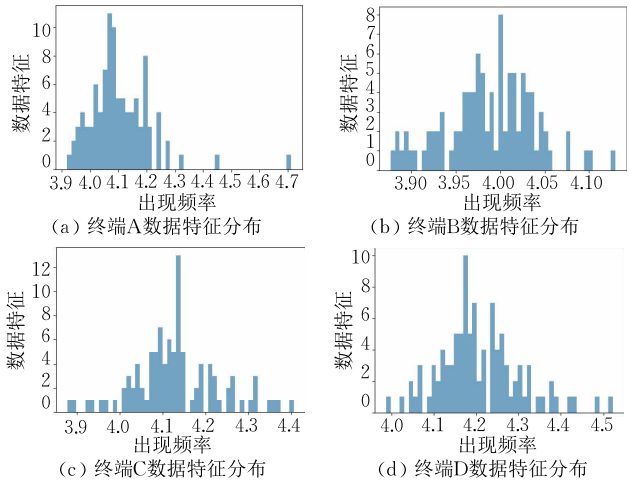

图 9　四个移动终端实时数据特征

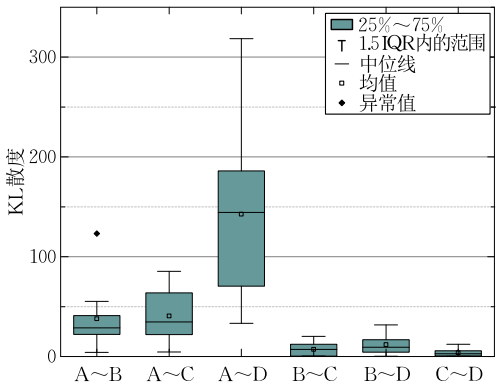

图 10　不同终端数据分布之间的 KL 散度

图 10 显示，不同终端采集的数据之间 KL 散度有着明显的差异，如终端 A 与终端 D 之间的 KL 散度最高时超过 300，而终端 C 和终端 D 之间的 KL 散度均接近 0，四个终端采集的数据之间特征分布差异难以预测. 然后我们在三个不同的时间点，将四个终端模型用对应终端采集到的视频数据进行模型训练并聚合生成全局模型，得到聚合后全局模型的精度并计算全局模型的精度损失，如表 3 所示.

表 3　聚合全局模型精度以及全局模型精度损失

模型	时间点	聚合后全局模型精度			全局模型精度损失		
M1	t_1	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$
		0.364	0.538	0.408	0.181	0.195	0.203
	t_2	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$
		0.400	0.582	0.472	0.225	0.193	0.258
	t_3	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$
		0.317	0.492	0.345	0.251	0.226	0.295
M2	t_1	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$
		0.256	0.413	0.293	0.142	0.163	0.175
	t_2	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$
		0.283	0.437	0.332	0.178	0.158	0.193
	t_3	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$	$IoU=0.50:0.95$	$IoU=0.50$	$IoU=0.75$
		0.224	0.361	0.255	0.191	0.187	0.204

表 3 说明,在多终端实时数据特征分布异构的情况下,模型聚合带来的全局模型精度损失是不可忽视的,最高损失甚至接近 30%.因此,解决数据分布异构性是提升聚合模型的关键,也是实现终端模型和全局模型协同高效演化的保证.

7.2.4 NestEvo 与其他基准算法的演化性能对比

本实验对比了多终端互学习共进演化方法(NestEvo)与四种不同基准算法在真实世界视频数据下的性能,在此实验中,我们对三个精度分别为 0.441,0.315,0.437(基于其对应终端采集到的视频数据)的剪枝后的 Faster-RCNN 作为终端模型进行演化更新,A2、A4 和本文算法采用了精度为 0.624(基于三个终端的联合数据)的 Faster-RCNN 作为原始全局模型.本实验具体对比了五种算法演化前后终端模型和全局模型的精度以及其对应的演化增益,实验结果如表 4 所示.

表 4 模型在线演化方法性能对比

方法	演化前全局模型精度	演化后全局模型精度	全局模型的演化增益	演化后终端模型精度				终端模型的演化增益			
				$IoU=0.50$				$IoU=0.50$			
	$IoU=0.50$	$IoU=0.50$	$IoU=0.50$	终端模型A	终端模型B	终端模型C	平均	终端模型A	终端模型B	终端模型C	平均
A1		None		0.504	0.469	0.497	0.490	14.3%	48.9%	13.7%	23.4%
A2	0.624	0.632	1.3%	0.504	0.475	0.501	0.505	14.3%	50.8%	15.5%	27.2%
A3		None		0.441	0.315	0.437	0.397	None			
A4	0.624	0.624	None	0.501	0.478	0.493	0.491	13.6%	51.7%	12.8%	23.7%
NestEvo	0.624	0.681	9.13%	0.571	0.543	0.584	0.566	29.5%	72.4%	33.6%	42.6%

根据表 4 显示,NestEvo 在全局模型演化增益和终端模型演化增益取得了最优性能.具体地,NestEvo 实现了 9.13%的全局模型精度增益,由于全局模型本身精度较高,提升困难较大,因此,该提升幅度较为明显.相比于同样进行全局模型演化的算法 A4,本文算法实现了全局模型精度 7.8%的增长,原因在于 NestEvo 利用数据生成,减少了不同终端之间的数据特征分布差异,减少了模型聚合损失.而在终端模型演化方面,NestEvo 实现了 42.6%的终端模型演化增益,相较于其他四种基准有着明显优势.因此,NestEvo 实现了终端模型和全局模型协同演化的最优性能.

7.2.5 数据偏差阈值的影响

本实验利用 7 个部署了 M1 模型的终端放置在白天校园环境中测试 30 min,对于不同数据偏差阈值下,7 个终端模型的平均推理精度和全局模型的演化次数进行了统计,在此过程中,我们对于阈值的设置为 4,即当超过一半终端的实时数据与全局模型训练数据产生较大偏差时,触发全局模型演化,实验结果如图 11 所示.

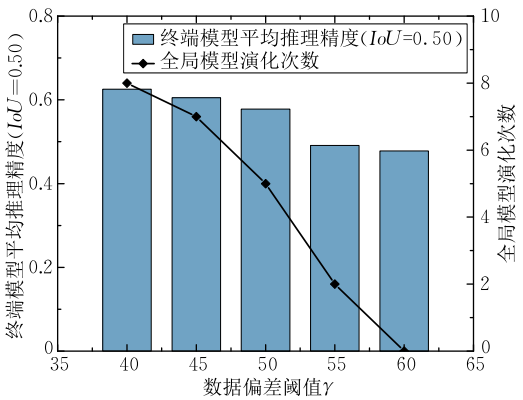

图 11 数据偏差阈值的影响

可以看到,随着数据偏差阈值的增加,全局模型演化次数逐渐减少,但终端模型的平均推理精度不断下降.为了在保证较高终端模型推理精度以及用户体验的同时,降低系统演化的开销,因此,我们令数据偏差阈值 $\gamma=50$.当 $\gamma<50$ 时,终端模型推理精度提升并不明显,而全局模型演化次数减少了接近一半;而当 $\gamma>50$ 时,虽然全局模型的演化次数下降明显,但终端推理精度下降至 0.5 以下,影响了用户体验.

7.3 基于存内计算的模型演化训练加速方案测试

在本实验中,我们对基于存内计算的模型演化训练加速方案进行了模拟实验和计算仿真验证.

7.3.1 GPU 加速的验证与分析

在本实验中我们以 RTX3080 10 GB 作为 GPU,对终端模型和中间模型的互学习过程进行加速优化.首先,本实验通过将 RTX3080 10 GB 作为单一共享边缘服务器对四个终端进行模型在线演化服务,每个终端模型进行 10 次演化,统计对比不同 GPU 调度方案下(B1、B2 和 NestEvo),四个终端每次演化任务的平均演化延迟,其中本次实验以带标签的真实场景视频数据作为训练集进行常规模型训练,实验结果如图 12 所示.

从图 12 中可以看到,NestEvo 在四个终端均获得了最短的演化延迟,这是因为 NestEvo 在并行执行模型演化服务,充分利用 GPU 吞吐量的同时,借助短作业优先原则,进一步缩短了所有演化任务的

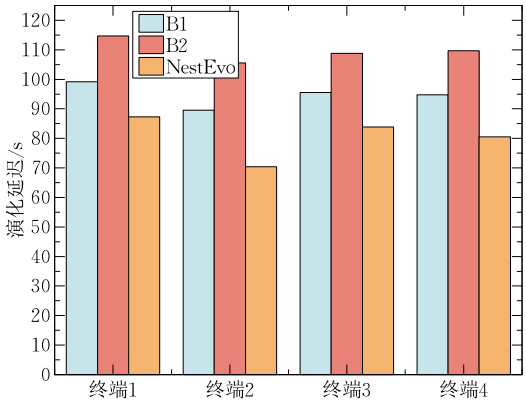

图 12　GPU 加速训练不同调度方式下的演化延迟对比

平均等待时间，从而取得了最短的演化延迟.

之后，本实验通过分析不同 batch size 下互学习的模型显存占用、互学习训练时间以及自适应数据压缩引入的终端模型精度变化，对 GPU 加速训练以及自适应数据压缩机制进行进一步验证. 实验结果如图 13、表 5 所示.

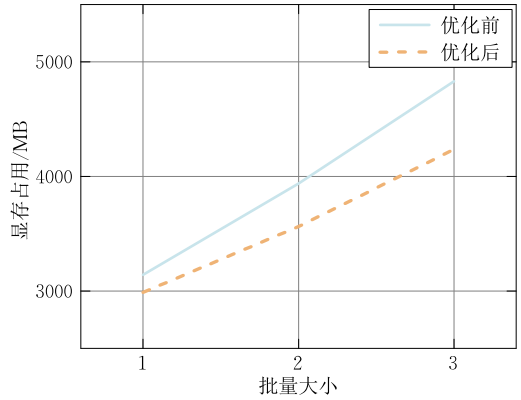

图 13　自适应数据压缩优化前后模型演化显存占用的对比

由图 13 可以看到，自适应数据压缩后的模型的显存占用总是低于优化之前，并且，随着 batch size 的增加，优化后模型显存占用的降低愈发明显. 当 batch size 为 1 时，显存占用从 3141 MB 降低到 2989 MB，降低了 5.09%，当 batch size 为 3 时，显存占用从 4831 MB 降低到了 4236 MB，降低了 12.32%. 可以看出，自适应数据压缩可以减少终端模型在互学习和知识蒸馏过程中的中间结果，从而降低模型的显存占用，节约硬件资源.

由表 5 可见，优化之后的终端模型和中间模型互学习的运行时间明显缩短，当 batch size 为 1 时，优化前两种模型互学习时间为 453.46 s，而优化之后，此过程耗时 321.33 s，优化前后，时间减少 29.14%，而 batch size 为 2 或 3 时，优化前后耗时也均降低了 30%左右. 由此可见，针对 GPU 的优化可以有效地降低终端模型和中间模型互学习过程的运行时间，大幅度加快整个系统的演化速度.

从表 5 中我们还可以发现，优化前后，batch size 为 3 或 2 时，模型的运行时间变化很小，但当 batch size 从 2 降低到 1 时，模型的运行时间均有明显的上升，即互学习的速度降低了：优化前耗时从 359.76 s 增加到了 453.46 s，耗时增加了 26.05%，优化后从 248.23 s 增加到了 321.33 s，耗时增加了 29.45%. 该过程运行时间的突然大幅度增加与 GPU 固有的“内存墙”问题有关：当 batch size 不低于 2 时，此时 GPU 运行速度的限制在于计算单元的计算能力，如 roofline 模型所示，此时已经到达“屋顶”，即计算强度已经到达最大计算强度，硬件性能的瓶颈为硬件算力；而当 batch size 降低至 1 时，GPU 吞吐量的限制在于计算平台的带宽，由于带宽的限制，此时并没有办法发挥出硬件的全部算力，导致计算资源的浪费. 而通过提高 batch size 来弥补“内存墙”的手段会导致整个系统的显存占用随之快速提升，如图 13 所示. 显存是 GPU 非常重要的硬件资源，决定着进程是否可以正常稳定运行，在运行程序时宁可牺牲计算速度，也不能无限制地增加系统的显存占用，导致进程无法运行. 这也是存内计算技术存在和发展的必要性.

表 5　自适应数据压缩优化前后演化时间的对比

批量大小	运行时间（优化前）/s	运行时间（优化后）/s	优化性能	
			缩减时间/s	缩减比例/%
1	453.46	321.33	132.13	29.14
2	359.76	248.23	111.53	31.00
3	327.35	227.53	99.82	30.49

由图 14 可以看出，在经过自适应数据压缩前后的深度模型精度的差别很小，在误差允许的范围内基本可以忽略不计. 因此，GPU 加速训练的优化方案并不会给模型精度增益带来明显的负面影响，可

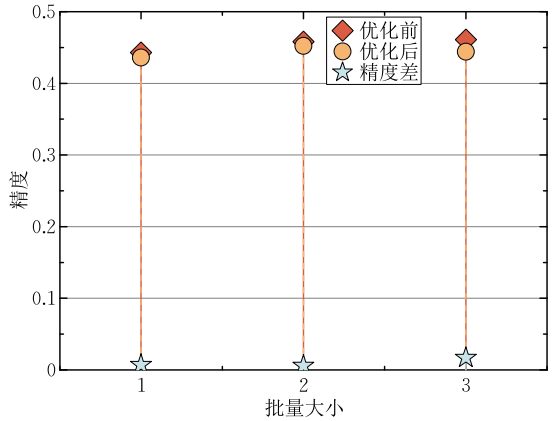

图 14　自适应数据压缩训练优化前后模型精度的对比

以稳定加速终端模型和中间模型的学习演化过程.

7.3.2 存内计算加速训练的模拟验证与分析

在本实验中,我们对存内计算加速训练的加速效果和优化方案进行了模拟实验和计算仿真验证.具体来说,我们在终端模型和中间模型的互学习过程中注入噪声,在中间模型聚合生成全局模型后,我们对全局模型的精度进行测试,并在测试过程中向全局模型注入权重噪声,验证全局模型对于权重噪声的鲁棒性.实验结果如图 15 所示.

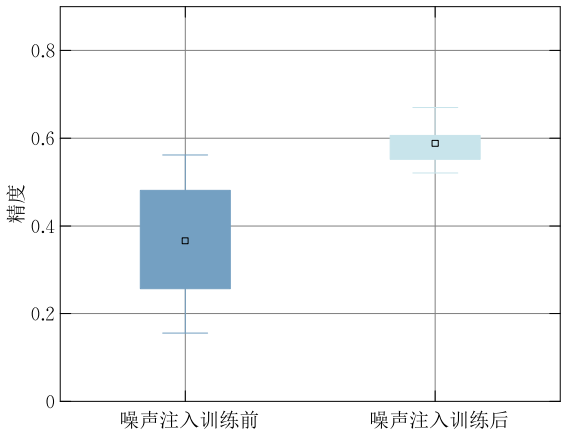

图 15 噪声注入模型训练优化前后鲁棒性对比

之后,我们以存算一体芯片 ISAAC[52] 为例,通过其吞吐量,模拟仿真计算本系统中 Faster-RCNN 部署在ISAAC的推理时间,与Faster-RCNN在RTX3080 10 GB的运行时间进行对比分析.在本系统中,存内计算加速用于全局模型的推理加速,全局模型在本系统中的作用是为各个终端模型上传至边缘服务器的视频帧数据进行标注,并为每个终端模型的知识蒸馏过程提供相应的 soft target,假设每次模型演化时,终端设备向边缘服务器上传 1200 张视频帧,通过模拟实验和吞吐量计算,Faster-RCNN 在 RTX3080 10 GB 和 ISAAC 上的推理时间如表 6 所示.

由表 6 中可见,使用存内计算加速其 ISAAC 可以大幅度缩短全局模型的推理时间,使其可以更快地为终端模型上传的视频帧集进行数据标注和为终端模型知识蒸馏过程提供 soft target,从而加快整个模型在线演化系统的演化速度,缩短演化延迟.并且,随着边缘服务器服务的终端设备数量以及终端上传至边缘的视频帧数量的增加,存内计算加速器的加速效果会更明显.

表 6 不同硬件设备推理时间对比

数据量	1200 张视频帧						
分辨率	1280×720						
模型类型	FASTER-RCNN						
网络复杂度	11.72MFLOPS						
硬件设备	RTX 3080 10GB					ISAAC	
时间点/s	0～2.1	1200 张视频帧重复进行				0～2.1	2.1～20.83
		2.1～2.112	2.112～2.218	…	143.59～143.7		
事件	加载模型	外部数据迁移	GPU 内部数据迁移和模型推理	…	GPU 内部数据迁移和模型推理	加载模型	外部数据迁移和模型推理
显存占用/MB	1675	0.239	1466	…	1466	None	None
总时间/s	143.7					20.83	
时间缩减/s	122.87						

8 未来展望

本文提出了多终端视频流智能识别模型共进演化方法,旨在提升受到数据漂移影响的终端模型推理精度,本文通过互学习实现了多终端模型的知识交互和共进学习,完善了现有的演化系统,实现性能突破;同时,利用基于存内计算的训练加速加快了演化速度,进一步提高了多终端模型全生命周期推理性能,为未来模型在线演化系统中边缘服务器的异构加速器设计提供指导.而边缘辅助的模型演化仍存在需要深入研究的场景和任务.首先,目前绝大部分的模型演化相关工作(包括本文)都是基于视觉模型,然而,如今多模态信号的应用越来越高,例如智能个人助理和智能交通信号等需要处理语音、文本甚至无线感知等信号.并且,多模态信号之间信息的互补和利用可以为模型演化性能增益的提升提供解决思路.因此,多模态数据处理模型的协同演化与多模态数据的交互方法可能会成为未来的研究重点.此外,频繁的演化请求会造成大量不必要的通信开销.我们可以从两方面入手:一方面,我们可以通过设计仅依靠终端资源的终端模型推理精度评估指标,自适应地触发模型演化;另一方面,由于大部分工作只注重终端模型的精度提升,而使模型过度拟

合训练数据，导致其泛化能力的下降，从而增加了模型演化频率，因此，如何平衡精度与泛化能力也将成为未来的研究重点.

9　结束语

本文针对移动场景中的轻量化深度模型精度易受数据漂移影响的现象提出了一种多终端视频流智能识别模型共进演化方法，旨在提升终端模型的推理精度. 本文以软硬件结合的思想为基础，设计了新颖的多终端互学习共进演化方法，实现了终端模型和全局模型的自适应在线协同演化，并且提出了支撑终端互学习共进演化方法的基于存内计算的训练加速方案来加快系统的演化速度，从而提高终端模型全生命周期推理性能. 最后我们在 RTX 3080 10 GB 上进行实验验证了多终端互学习共进演化方法以及 GPU 加速训练优化方案的优越性，并通过仿真计算验证存内计算加速训练的有效性. 经过一次完整的模型在线演化后，演化延迟减少了 51.98%，终端模型平均推理精度提升了 42.6%，并且模型演化时间和 GPU 显存占用都得到了大幅度缩小. 通过训练加速方案的模拟验证指导了未来在模型演化系统中边缘服务器异构加速器的设计优化.

参　考　文　献

[1] Denil M, Shakibi B, Dinh L, et al. Predicting parameters in deep learning. Advances in Neural Information Processing Systems, 2013, 26: 2148-2156

[2] Liu S, Guo B, Ma K, et al. AdaSpring: Context-adaptive and runtime-evolutionary deep model compression for mobile applications. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2021, 5(1): 1-22

[3] Gama J, Žliobaitė I, Bifet A, et al. A survey on concept drift adaptation. ACM Computing Surveys, 2014, 46(4): 1-37

[4] Lu J, Liu A, Dong F, et al. Learning under concept drift: A review. IEEE Transactions on Knowledge and Data Engineering, 2018, 31(12): 2346-2363

[5] Zhang L, Gao G, Zhang H. Towards data-efficient continuous learning for edge video analytics via smart caching//Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems. Boston, USA, 2022: 1136-1140

[6] Jia L, Zhou Z, Xu F, et al. Cost-efficient continuous edge learning for artificial intelligence of things. IEEE Internet of Things Journal, 2021, 9(10): 7325-7337

[7] Heo B, Lee M, Yun S, et al. Knowledge distillation with adversarial samples supporting decision boundary//Proceedings of the AAAI Conference on Artificial Intelligence. Washington, USA, 2019, 33(1): 3771-3778

[8] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015

[9] Han S, Pool J, Tran J, et al. Learning both weights and connections for efficient neural network. Advances in Neural Information Processing Systems, 2015, 28: 1135-1143

[10] Jiang N F, Zhao X, Zhao C Y, et al. Pruning-aware sparse regularization for network pruning. Machine Intelligence Research, 2023, 20(1): 109-120

[11] Fang G, Ma X, Song M, et al. DepGraph: Towards any structural pruning//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Vancouver, Canada, 2023: 16091-16101

[12] Ma X, Fang G, Wang X. LLM-Pruner: On the structural pruning of large language models. arXiv preprint arXiv:2305.11627, 2023

[13] Yang W, Zhi X, Tong W. Optimization of linear quantization for general and effective low bit-width network compression. Algorithms, 2023, 16(1): 31

[14] Wu X, Wu Y, Zhao Y. Binarized neural networks on the ImageNet classification task. arXiv preprint arXiv:1604.03058, 2016

[15] Liu Z, Cheng K T, Huang D, et al. Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. New Orleans, USA, 2022: 4942-4952

[16] Choi K, Lee H Y, Hong D, et al. It's all in the teacher: Zero-shot quantization brought closer to the teacher//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. New Orleans, USA, 2022: 8311-8321

[17] Liu Z, Li J, Shen Z, et al. Learning efficient convolutional networks through network slimming//Proceedings of the IEEE International Conference on Computer Vision. Venice, Italy, 2017: 2736-2744

[18] Li H, Kadav A, Durdanovic I, et al. Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710, 2016

[19] Huang Z, Wang N. Data-driven sparse structure selection for deep neural networks//Proceedings of the European Conference on Computer Vision (ECCV). Munich, Germany, 2018: 304-320

[20] Han S, Mao H, Dally W J. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. arXiv preprint arXiv:1510.00149, 2015

[21] Elthakeb A T, Pilligundla P, Mireshghallah F, et al. WaveQ: Gradient-based deep quantization of neural networks through sinusoidal adaptive regularization. arXiv preprint arXiv:2003.00146, 2020

[22] Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017

[23] Sandler M, Howard A, Zhu M, et al. MobileNetV2: Inverted residuals and linear bottlenecks//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 4510-4520

[24] Howard A, Sandler M, Chu G, et al. Searching for MobileNetV3//Proceedings of the IEEE/CVF International Conference on Computer Vision. Seoul, Korea (South), 2019: 1314-1324

[25] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA, 2016: 770-778

[26] Mullapudi R T, Chen S, Zhang K, et al. Online model distillation for efficient video inference//Proceedings of the IEEE/CVF International Conference on Computer Vision. Seoul, South Korea, 2019: 3573-3582

[27] Bhardwaj R, Xia Z, Ananthanarayanan G, et al. Ekya: Continuous learning of video analytics models on edge compute servers//Proceedings of the 19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22). Seattle, USA, 2022: 119-135

[28] Tiezzi M, Marullo S, Faggi L, et al. Stochastic coherence over attention trajectory for continuous learning in video streams. arXiv preprint arXiv:2204.12193, 2022

[29] Khani M, Ananthanarayanan G, Hsieh K, et al. RECL: Responsive resource-efficient continuous learning for video analytics//Proceedings of the 20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23). Boston, USA, 2023: 917-932

[30] Zhang Y, Xiang T, Hospedales T M, et al. Deep mutual learning//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 4320-4328

[31] Nie X, Feng J, Yan S. Mutual learning to adapt for joint human parsing and pose estimation//Proceedings of the European Conference on Computer Vision (ECCV). Munich, Germany, 2018: 502-517

[32] Zhao H, Yang G, Wang D, et al. Deep mutual learning for visual object tracking. Pattern Recognition, 2021, 112: 107796

[33] Zhou Z, Qiu X, Xie J, et al. Binocular mutual learning for improving few-shot classification//Proceedings of the IEEE/CVF International Conference on Computer Vision. Montreal, Canada, 2021: 8402-8411

[34] Xu Y, Kan M, Shan S, et al. Mutual learning of joint and separate domain alignments for multi-source domain adaptation //Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. Waikoloa, USA, 2022: 1890-1899

[35] McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data//Proceedings of the Artificial Intelligence and Statistics. Fort Lauderdale, USA, 2017: 1273-1282

[36] Wu Q, He K, Chen X. Personalized federated learning for intelligent IoT applications: A cloud-edge based framework. IEEE Open Journal of the Computer Society, 2020, 1: 35-44

[37] Guo B, Mei Y, Xiao D, et al. PFL-MoE: Personalized federated learning based on mixture of experts. arXiv preprint arXiv: 2012.15589, 2020

[38] Balakrishnan R, Li T, Zhou T, et al. Diverse client selection for federated learning via submodular maximization//Proceedings of the International Conference on Learning Representations. 2022

[39] Huang T, Lin W, Shen L, et al. Stochastic client selection for federated learning with volatile clients. IEEE Internet of Things Journal, 2022, 9(20): 20055-20070

[40] Ruan Y, Joe-Wong C. FedSoft: Soft clustered federated learning with proximal local updating. Proceedings of the AAAI Conference on Artificial Intelligence, 2022, 36(7): 8124-8131

[41] Sattler F, Müller K R, Samek W. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. IEEE Transactions on Neural Networks and Learning Systems, 2020, 32(8): 3710-3722

[42] Tang M, Ning X, Wang Y, et al. FedCor: Correlation-based active client selection strategy for heterogeneous federated learning//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. New Orleans, USA, 2022: 10102-10111

[43] Qin Z, Yang L, Wang Q, et al. Reliable and interpretable personalized federated learning//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Vancouver, Canada, 2023: 20422-20431

[44] Xie M, Long G, Shen T, et al. Multi-center federated learning. arXiv preprint arXiv:2108.08647, 2021

[45] Demler M. Mythic multiplies in a flash. Microprocesser Report, 2018

[46] Klein J, Boybat I, Qureshi Y, et al. ALPINE: Analog in-memory acceleration with tight processor integration for deep learning. IEEE Transactions on Computers, 2022, 72(7): 1958-1998

[47] Dalgaty T, Castellani N, Turck C, et al. In situ learning using intrinsic memristor variability via Markov chain Monte Carlo sampling. Nature Electronics, 2021, 4(2): 151-161

[48] Ankit A, Sengupta A, Panda P, et al. RESPARC: A reconfigurable and energy-efficient architecture with memristive crossbars for deep spiking neural networks//Proceedings of the 54th Annual Design Automation Conference 2017. Austin, USA, 2017: 1-6

[49] Chi P, Li S, Xu C, et al. PRIME: A novel processing-in-memory architecture for neural network computation in reram-based main memory. ACM SIGARCH Computer Architecture News, 2016, 44(3): 27-39

[50] Hu M, Li H, Wu Q, et al. Hardware realization of BSB recall function using memristor crossbar arrays//Proceedings of the 49th Annual Design Automation Conference. San Francisco, USA, 2012: 498-503

[51] Liu X, Mao M, Liu B, et al. RENO: A high-efficient reconfigurable neuromorphic computing accelerator design//Proceedings of the 52nd Annual Design Automation Conference. San Francisco, USA, 2015: 1-6

[52] Shafiee A, Nag A, Muralimanohar N, et al. ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars. ACM SIGARCH Computer Architecture News, 2016, 44(3): 14-26

[53] Ankit A, Hajj I E, Chalamalasetti S R, et al. PUMA: A programmable ultra-efficient memristor-based accelerator for machine learning inference//Proceedings of the 24th International Conference on Architectural Support for Programming Languages and Operating Systems. Providence, USA, 2019: 715-731

[54] "A Mythic Approach to Deep Learning Inference". https://www.nextplatform.com/2018/08/23/a-mythic-approach-to-deep-learning-inference/

[55] "Mythic@ Hot Chips 2018". https://medium.com/mythic-ai/mythic-hot-chips-2018-637dfb9e38b7

[56] Doevenspeck J, Vrancx P, Laubeuf N, et al. Noise tolerant ternary weight deep neural networks for analog in-memory inference//Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN). 2021: 1-8

[57] Gokmen T, Rasch M J, Haensch W. The marriage of training and inference for scaled deep learning analog hardware//Proceedings of the 2019 IEEE International Electron Devices Meeting (IEDM). San Francisco, USA, 2019: 22.3.1-22.3.4

[58] He Z, Lin J, Ewetz R, et al. Noise injection adaption: End-to-end ReRAM crossbar non-ideal effect adaption for neural network mapping//Proceedings of the 56th Annual Design Automation Conference 2019. Las Vegas, USA, 2019: 1-6

[59] Joshi V, Le Gallo M, Haefeli S, et al. Accurate deep neural network inference using computational phase-change memory. Nature Communications, 2020, 11(1): 2473

[60] Zhou C, Kadambi P, Mattina M, et al. Noisy machines: Understanding noisy neural networks and enhancing robustness to analog hardware errors using distillation. arXiv preprint arXiv:2001.04974, 2020

[61] Moon S, Shin K, Jeon D. Enhancing reliability of analog neural network processors. IEEE Transactions on Very Large Scale Integration Systems, 2019, 27(6): 1455-1459

[62] Du W, Xu D, Wu X, et al. Fairness-aware agnostic federated learning//Proceedings of the 2021 SIAM International Conference on Data Mining (SDM). Virtual, Society for Industrial and Applied Mathematics, 2021: 181-189

[63] Wang C, Chen X, Wang J, et al. ATPFL: Automatic trajectory prediction model design under federated learning framework//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. New Orleans, USA, 2022: 6563-6572

[64] Li Q, Diao Y, Chen Q, et al. Federated learning on non-iid data silos: An experimental study//Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE). 2022: 965-978

[65] Kairouz P, McMahan H B, Avent B, et al. Advances and open problems in federated learning. Foundations and Trends in Machine Learning, 2021, 14(1-2): 1-210

[66] Zhao Y, Li M, Lai L, et al. Federated learning with non-iid data. arXiv preprint arXiv:1806.00582, 2018

[67] Goyal P, Dollár P, Girshick R, et al. Accurate, large minibatch SGD: Training ImageNet in 1 hour. arXiv preprint arXiv:1706.02677, 2017

[68] Smith S L, Kindermans P J, Ying C, et al. Don't decay the learning rate, increase the batch size. arXiv preprint arXiv:1711.00489, 2017

[69] Kanev S, Darago J P, Hazelwood K, et al. Profiling a warehouse-scale computer//Proceedings of the 42nd Annual International Symposium on Computer Architecture. Portland Oregon, USA, 2015: 158-169

[70] Yu Y, Zhao T, Wang K, et al. Light-OPU: An FPGA-based overlay processor for lightweight convolutional neural networks//Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Seaside, USA, 2020: 122-132

[71] Ke R, Li Z, Tang J, et al. Real-time traffic flow parameter estimation from UAV video based on ensemble classifier and optical flow. IEEE Transactions on Intelligent Transportation Systems, 2018, 20(1): 54-64

[72] Shi S, Gupta V, Hwang M, et al. Mobile VR on edge cloud: A latency-driven design//Proceedings of the 10th ACM Multimedia Systems Conference. Amherst, USA, 2019: 222-231

[73] Khani M, Hamadanian P, Nasr-Esfahany A, et al. Real-time video inference on edge devices via adaptive model streaming//Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021: 4572-4582

[74] Zhu Z, Hong J, Zhou J. Data-free knowledge distillation for heterogeneous federated learning//Proceedings of the International Conference on Machine Learning. 2021: 12878-12889

[75] Shen T, Zhang J, Jia X, et al. Federated mutual learning. arXiv preprint arXiv:2006.16765, 2020

[76] Jain A, Phanishayee A, Mars J, et al. Gist: Efficient data encoding for deep neural network training//Proceedings of the 2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA). Los Angeles, USA, 2018: 776-789

[77] Salkin H M, De Kluyver C A. The knapsack problem: A survey. Naval Research Logistics Quarterly, 1975, 22(1): 127-144

[78] Yu F, Chen H, Wang X, et al. BDD100K: A diverse driving dataset for heterogeneous multitask learning//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 2636-2645

[79] Everingham M, Eslami S M A, Van Gool L, et al. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision, 2015, 111: 98-136

[80] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards real-time object detection with region proposal networks. Advances in Neural Information Processing Systems, 2015, 28: 91-99

[81] Chen Y, Li W, Sakaridis C, et al. Domain adaptive faster R-CNN for object detection in the wild//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 3339-3348

WANG Le-Hao, Ph. D. candidate. His research interest is mobile embedded computing.

LIU Si-Cong, Ph. D., associate professor. Her research interests are Smart IoT and mobile embedded computing.

YU Zhi-Wen, Ph. D., professor. His research interests are ubiquitous computing and Smart IoT.

YU Hao-Yi, M. S. His research interests are smart IoT and mobile computing.

GUO Bin, Ph. D., professor. His research interests are ubiquitous computing, crowd intelligence and mobile crowd perception.

Background

Data drift, which means that the distribution of real data and model training data is different, will lead to a decrease in model accuracy, and this problem is more serious for mobile models. There are roughly two reasons: on the one hand, the data distribution captured by the mobile terminal will change faster. On the other hand, the compressed model deployed on the mobile terminal has a simple structure and poor generalization ability which leads to low target recognition accuracy in new scenarios. Continuous Model Evolution is an ideal solution to the problem of terminal data drift.

Existing work includes cloud-based online training and edge-based online training. Mullapudi et al. make the terminal device intermittently send video frames to the server, and perform knowledge distillation on the cloud server. Afterwards, the evolved model is reloaded to the terminal, so as to realize the online evolution of the terminal model. However, placing data and models in the cloud creates problems such as user privacy and model evolution delays. Bhardwaj et al. implemented both model inference and regular model retraining on edge node with limited resources to solve the data drift problem of edge video analysis systems, and designed a resource scheduling algorithm at the edge to balance the resource occupy between inference and model retraining. However, the teacher model may be also affected by the data drift to a extent. Therefore, in order to ensure the effect of evolving the terminal model, we need to evolve the global model as well. The evolution of the global model will face multiple challenges of terminal model heterogeneity and data distribution heterogeneity. Moreover, the existing research has not yet realized the cooperative work of the model retraining algorithm and the hardware accelerator. This paper overcomes the problem of heterogeneity for the first time, realizes the online co-evolution of the terminal model and the global model, and accelerates the retraining from the hardware level which provides guidance for the design of future edge heterogeneous accelerators, and optimizes the adaptation of software and hardware. In addition, we demonstrate the effectiveness of the proposed system by experiments.

This work is supported in part by the Key Program of National Natural Science Foundation of China (61960206008), the National Science Fund for Distinguished Young Scholars (62025205), and the General Program of National Natural Science Foundation of China (62102317).