边缘视频处理的细粒度划分与重组部署算法

覃 剑[1],石昌伟[1],张 媛[2],贾云健[1],胡浩星[1]
(1. 重庆大学微电子与通信工程学院,重庆 400030; 2. 国网重庆市电力公司电力科学研究院,重庆 401123)

摘 要: 随着视频数据的迅速增长,大规模视频处理业务需求急剧增加.如何及时处理视频数据获取有效信息,进而向用户快速提供视频分析业务是亟待解决的重要问题.针对此问题,提出一种面向大规模视频处理的边缘功能模块化及重组部署方法(EFMR).该方法将视频处理业务下沉到网络边缘,利用网络功能虚拟化,将边缘服务器中的视频业务请求根据其内在相关性进行功能细粒度划分,按需匹配并最大化复用资源,实现重组部署,从而以较小代价实现边缘视频业务处理功能的平滑扩展.实验结果表明,EFMR 方法不仅降低了边缘服务器的接入与响应时延、业务的推理时间,而且还节省了大量的计算资源,提高了视频处理业务部署速度.

关键词: 移动边缘计算; 网络功能虚拟化; 模块化; 重组; 细粒度
中图分类号: TN919.8 **文献标识码:** A **文章编号:** 0372-2112(2021)11-2152-08
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.12263/DZXB.20200767

Fine-Grained Partitioning and Reorganization Deployment Strategy of Edge Video Processing

QIN Jian[1], SHI Chang-wei[1], ZHANG Yuan[2], JIA Yun-jian[1], HU Hao-xing[1]
(1. *School of Microelectronic and Communication Engineering, Chongqing University, Chongqing* 400030, *China*;
2. *State Grid Chongqing Electric Power Corporation, Electric Power Research Institute, Chongqing* 401123, *China*)

Abstract: With the rapid growth of video data, the demand for large-scale video processing tasks increases dramatically. How to process video data in time to obtain effective information and provide users with video analysis services quickly is an important issue to be solved. Aiming at this problem, a new deployment method of Edge Functions Modularized and Reorganized (EFMR) for large-scale video processing is proposed. This method sinks video processing services to the edge of the network. Using network function virtualization, video service requests sent to the edge server are divided fine-grainedly based on their inherent process correlation, and resources are matched and redeployed on demand based on the division results. In this way, we can smoothly expand the edge video service processing capabilities at a small cost. Experimental results show that EFMR method not only greatly reduces the edge server's access and response delay, reduces the inference time, but also saves a lot of computing resources of edge servers and speeds up the deployment of video processing services.

Key words: mobile edge computing; network function virtualization; modular; reorganized; fine-grained

1 引言

随着智能终端及网络摄像头数量的爆发式增长,视频业务需求逐渐多样化、复杂化,随时随地都有庞大的视频数据[1]等待处理,其增长速度远远超过了网络带宽的增速,海量视频的储存、分析以及移动终端需求的多样化对计算能力、网络带宽以及时延等都提出了巨大挑战.为及时处理如此大量视频数据,依靠人工已变得非常困难,所以如何及时处理视频并获取有效信息成为亟待解决的问题.

针对该问题,一种典型解决方案就是丰富前端设备资源,增加设备计算能力[2],使其能够对接收到的视频数据进行及时有效的处理.这就要求每个摄像头或者智能终端都具备视频处理的功能,会使前端设备的成本及功耗大大提高.另一种方案就是将视频数据交由云端[3,4]的数据中心处理,只将结果返回给终端.基于云端数据中心的方式虽然解决了前端设备资源受限的问题,但大量数据(图像和视频数据)通过延迟高、带

收稿日期:2020-07-23;修回日期:2021-03-02;责任编辑:李勇锋
基金项目:国家自然科学基金(No. 61971077);天奥基金(No. 2020154704)

宽波动大的核心网传输到远端,会造成较大的端到端延迟、较高的移动终端通信功耗及加大核心网负担等问题. 移动边缘计算[5,6](Mobile Edge Computing, MEC)位于云端和终端之间,可以在移动网络边缘提供服务环境和计算能力,通过靠近移动用户侧来减少网络操作和服务交付的时延. 如EdgeLearning框架[7]将运算分解到边缘服务器与云端,边缘服务器进行简单的数据预处理,然后发送到远端有强大GPU资源的云端来执行CNN(Convolutional Neural Networks)等处理. Huang[8]等人综合考虑了云-边的数据传输量、学习精度和处理延时等多种因素,对调度算法进行优化. 这种边缘服务器和云端协同工作的方法虽然减轻了云端数据处理压力,但仍需传输大量数据到远端进行处理,依然会加大核心网负载及端到端传输延迟. 为此一些学者又提出端-边协同的方法,尽量减少对云端的访问. Edgent[9]就将运算量最大的DNN部分进行自适应分区,由智能终端和边缘服务器协同处理,以均衡计算负载. Wang[10]则搭建了有36个节点的边缘计算系统CampEdge,对8500个AP、44000个用户进行长时间观测,建立业务量预测模型,将业务分配与卸载视为多目标优化问题,设计了基于ADMM(Alternating Direction Method of Multipliers)的调度算法,将用户服务延迟减少了30%. 这些端-边结合的方法缓解了远距离传输和移动设备资源不足的问题,但由于没有考虑到视频业务本身相似性以及模块化特点,当请求多个视频处理业务时,会进行大量重复计算,在设备和边缘服务器上的总计算量并没有减少,占用了移动设备和边缘服务器的计算资源,大大增加了推理时间,难以满足实时性要求.

故在要求低时延及资源受限的情况下,上述方法均存在一定局限性,无法实现既降低时延又减少业务计算量的目标. 实际上深入分析可发现,视频处理业务具有明显的模块化特点,一个完整的视频处理业务常可分解成多个相关模块,而不同视频处理业务之间的处理模块往往又有重复性,可有效利用多个视频处理业务之间具有重复处理模块的性质,减少业务的计算量. 在边缘计算中,则可以通过虚拟化技术,将计算、存储等资源进行池化,基于业务需求,灵活、按需、智能地提供分布式、低时延、高性能、安全可靠、绿色节能的信息化基础设施,满足业务的需求. 因此,本文提出了一种网络功能虚拟化[11]下,边缘服务器功能模块化并重组(Edge Functions Modularized and Reorganized, EFMR)的方法. EFMR方法的优化策略主要表现在四个方面:一是EFMR通过网络功能虚拟化方式由边缘服务器向用户提供服务,通过网络功能虚拟化结合通信网中边缘服务器的计算资源不仅提高了资源利用率,便于管理,又增加了业务的可靠性. 同时由于靠近用户侧,处理时延低,带宽大,可以满足低时延和实时性的要求;二是功能模块化并按需重组,当用户向边缘服务器请求视频处理业务时,根据其内在处理过程相关性对业务进行细粒度分解,并根据划分结果按需调度边缘服务器中的多个处理模块进行重组部署,实现边缘服务器功能、资源的按需重组及可定制;三是计算量复用,大规模视频处理业务中,用户的多个视频处理业务请求具有很强的相关性,可直接利用相关模块的计算结果,送入后续的计算过程,进而降低推理过程中的计算量和时间;四是边缘功能的细粒度扩展,当边缘服务器中需要增加新的功能时,可以较小代价模块化扩充,节省资源,提高资源利用率和部署速度.

2　EFMR方法原理

目前,基于MEC的大规模视频处理业务中,视频分析扮演着十分重要的角色,包括目标分类、检测、跟踪[12]等,且大量使用基于深度学习的模型和算法. 但深度学习计算资源需求多、计算时间长、算法时间复杂度高. 同时,随着视频处理算法的不断改进与提升,从同一视频来源中可以进行的分析以及获得的信息越来越多,相应的视频分析算法也随之增多且还在不断的扩展之中. 传统的部署方法是用户请求新的视频处理业务时,边缘服务器会针对该业务增加一个处理算法程序,如目标分类程序,车牌检测程序等,或在原来程序基础上增加相应的处理功能,以完成整个的视频处理,这就导致了计算的重复、需占用更多的计算资源以及需重新部署视频处理功能等问题. 而本文提出的EFMR方法,通过将视频处理功能模块化部署于边缘服务器中,并根据业务需求按需调用各功能模块,当有新视频处理类型产生时,独立地增加相应需要的功能模块,具有细粒度扩展性,不仅节省了边缘服务器的计算资源和算力,而且无需重复部署,节约了部署时间. 在传统部署(图1(a))中,前端设备(如监控设备、无人机等)负责采集视频数据并请求业务,由通信链路将数据传给边缘服务器,然后由部署在边缘服务器中的完整独立的视频处理程序对请求进行处理,并将结果反馈给前端. 而EFMR方法(图1(b))主要由用户层和边缘计算层组成,而边缘计算层中又包含分解调度模块和虚拟化模块,分解调度模块的功能是将用户请求的视频业务分解成多个小的模块化任务,然后与部署在边缘服务器中的多个功能模块进行匹配,匹配成功后调度相应功能模块进行重组来完成视频分析功能;虚拟化模块的功能是将部署在边缘服务器中的视频处理功能通过网络功能虚拟化以模块化方式向用户提供业务. 用户层指前端设备,是视频处理业务的来源;边缘计算层负责对用户层的视频处理任务分解、匹配并调

度边缘服务器相关功能模块重组并进行处理，然后将结果返回给前端设备，且多个边缘服务器之间通过网关连接在一起，保证边缘服务器资源的共享. 当用户层向边缘服务器请求视频处理业务时，其处理流程如下：

(1)通过边缘接入点将前端设备产生视频业务数据发送到边缘服务器；

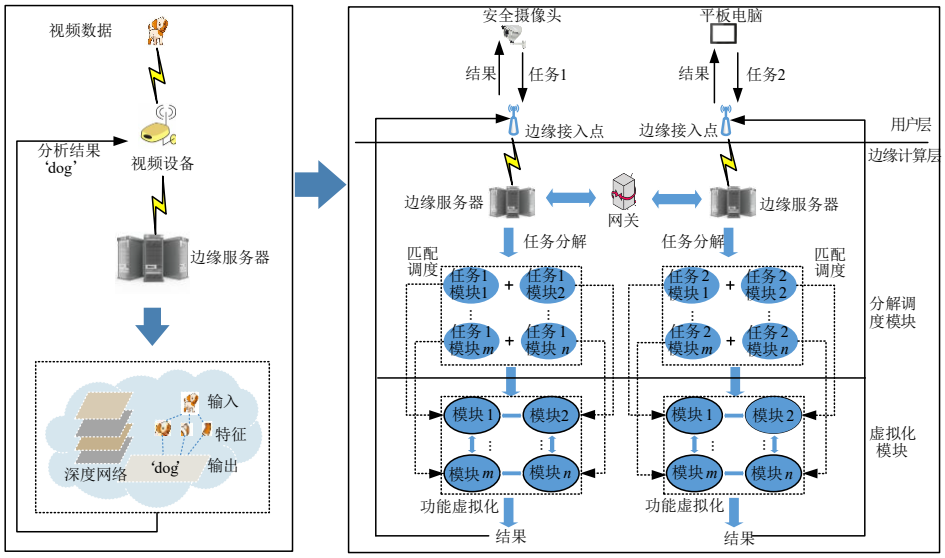
(2)分解调度模块将视频处理业务进行分解，然后与资源虚拟化模块中的多个功能模块进行匹配；

(3)匹配成功后，由分解调度模块对相应功能模块进行调度；

(4)重组模块处理视频业务，得到分析结果；

(5)边缘服务器通过通信链路将结果反馈给用户的前端设备.

基于深度学习的视频分析业务大多要求低时延，需要计算资源较多，因此如何对视频分析业务进行合理有效地分解及分配是本文的主要研究内容. 为评价算法效果，此处先建立时间模型和功能模块化模型. 通过边缘服务器功能模块化并按需重组的方式减少深度模型推理的计算量，加快边缘服务器的处理速度，减少总体时延是本文主要的研究目标.



(a) 传统完整模型部署方法　　(b) EFMR 部署方法

图1　边缘服务器视频处理功能部署方法示意图

2.1　时间模型

通过在靠近 UE(User Equipment)的基站侧部署 MEC 服务器，利用 MEC 存储资源将服务下载到 MEC 服务器上，用户可直接从 MEC 服务器获取所需的服务，这样极大的节省了用户发出请求到被响应之间的等待时间. 设用户从向边缘服务器发送视频业务请求开始，到接收到边缘服务器的计算结果为止所需要的时间为 t. 则对同一业务请求 n 次所需要总时间为 t_{total}，如按传统的边缘服务器视频业务的部署方式，总时间如式(1)所示：

$$t_{\text{total}} = nt = nt_{\text{t}} + nt_{\text{s}} \tag{1}$$

其中，t_{s} 表示边缘服务器处理时间，即处理视频业务并推理出结果所用的时间，t_{t} 表示传输时间，即请求从用户到达边缘服务器以及边缘服务器将结果发送给用户所需的时间.

对 EFMR 方法而言，则是通过在边缘服务器中模块化部署业务，并减少共同计算量的方式，使其总时间得以大大减少，所用总时间如式(2)所示：

$$t_{\text{total}}=nt=nt_{\text{t}}+t_p+n\left(\sum_{i=1}^{N-1} t_i\right) \tag{2}$$

其中，N 表示一次视频业务中所需的边缘服务器中的功能模块数量，t_p 表示边缘服务器中的视频业务所需的共同模块计算所需的时间；t_i 表示边缘服务器中模块 i 进行计算所需的时间.

2.2　功能模块化

视频本身具有模块化以及可组织性的特点，而边缘服务器中所部署的视频处理功能在进行业务推理时也具有模块化的特性，一般主要有特征提取、候选框生成和分类或者检测等几个步骤.

深度学习模型推理时，主要计算量集中在特征提取阶段. 以经典深度模型 AlexNet[13] 为例(表 1)，卷积层大约占了总运算量的 92% 左右. 本文通过网络功能虚拟化，将边缘服务器中提供的功能模块化，进而减少多次视频请求业务共同处理过程的计算量，充分利用计算资源，加快处理速度.

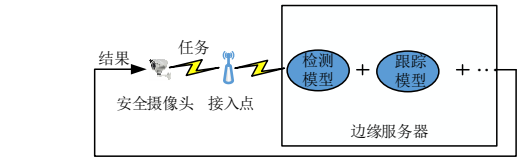
设边缘服务器对用户发起的一次视频分析业务进行处理时所消耗的计算资源为 R. 易见，按传统部署方式，当用户请求 n 次视频分析业务时所需要的总的计算资源如式(3)所示：

表 1　AlexNet 模型浮点运算量

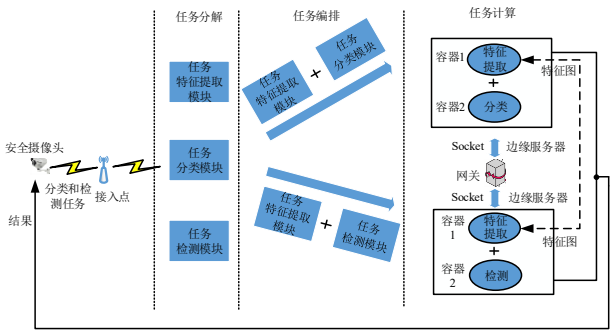
层类型	浮点运算量(FLOPs)
卷积层 1(11×11, 4, 96, ReLU)	105M
卷积层 2(5×5, 1, 256, ReLU)	223M
卷积层 3(3×3, 1, 384, ReLU)	149M
卷积层 4(3×3, 1, 384, ReLU)	112M
卷积层 5(3×3, 1, 256, ReLU)	74M
全连接层 1(4096, ReLU)	37M
全连接层 2(4096, ReLU)	16M
全连接层 3(1000)	4M

$$R_{\text{total}} = nR \tag{3}$$

而对 EFMR 方法,首先对请求的完整视频分析任务进行模块化分解,分解成多个模块化的小业务,然后与边缘服务器中的功能模块进行匹配并调度相关功能模块完成功能重组,边缘服务器通过轻量级容器方式提供模块化功能,同时保留各功能模块的计算结果,当有同源其他业务产生时,直接将计算结果传输到相应的计算模块. 如图 2,以用户请求单个目标分类任务为例,边缘服务器将按需组合 CNN 特征提取模块和分类模块,推理出待分类目标的标签并将 CNN 特征提取模块的结果进行保存. 当该用户继续请求目标检测业务时,则可直接使用 CNN 特征提取模块对分类业务计算的特征图结果,通过 Socket 通信的方式直接传输给检测模块,进行后续的检测工作.



(*a*)　传统部署方式



(*b*)　EFMR 方法部署方式

图 2　两种功能部署方法比较

当用户向边缘服务器请求 m 次分类业务与 n 次检测业务时,EFMR 通过网络虚拟化的方式,调用临近边缘服务器特征提取模块的结果送入后续的检测和分类任务,则总的计算量为:

$$R_{\text{total}} = R_{\text{f}} + mR_{\text{c}} + nR_{\text{d}} \tag{4}$$

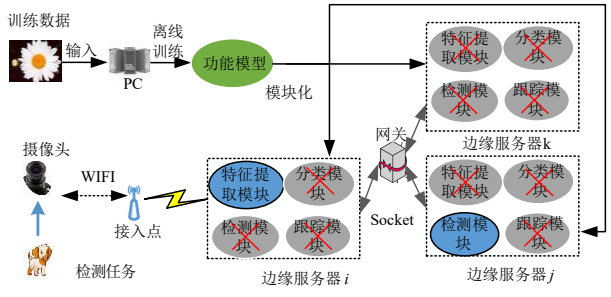
其中,R_{f} 表示 CNN 特征提取模块所需的计算资源,R_{c} 表示分类模块所需的计算资源,R_{d} 表示检测模块所需的计算资源.

3　EFMR 方法

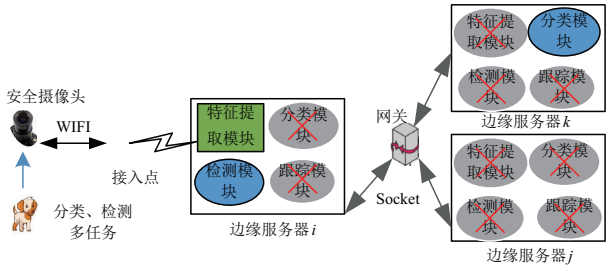
由于边缘服务器不可能像云计算中心一样具有丰富的计算及存储资源,当接入大量的视频分析业务时,由于无法及时增加物理资源,很大可能将会面临资源不足的问题. 同时,当不同类型的视频分析请求接入时,由于需要单独完整扩展视频分析功能,也面临重新部署时间长,资源占用多的问题.

为在降低边缘服务器接入与响应时延的同时,尽可能多地满足任务的资源需求,减少计算量和处理时间,同时支持边缘服务器中视频处理功能的细粒度扩展,EFMR 将通过网络功能虚拟化,将边缘服务器中视频处理功能模块化并按需重组. 当用户向边缘服务器请求单独的检测业务时,边缘服务器模块组合和资源使用情况如图 3(*a*)所示. 图中模块的"×"标志表示边缘服务器因资源不足或被占用,暂不能提供该模块功能;反之则表示可以提供该业务功能模块. 通过网关配合网络虚拟化,可以综合利用其它边缘服务器的算力和存储资源,解决当前边缘服务器算力和存储资源不足的问题. 同时想强调的是,EFMR 不需对特定视频任务单独增加完整的处理算法程序,而是通过模块化方式细粒度增添小功能模块,因此具有平滑扩展性,节省了部署时间和边缘服务器的可利用资源.

当用户再次请求多种视频分析任务时,由于特征提取模块保存有上次计算结果,只需要调度其他的功能模块进行重组即可,因此将特征提取结果送入后续



(*a*)单检测任务模块重组



(*b*)分类和检测多视频分析任务模块重组

图 3　EFMR 方法多次视频分析任务模块资源重组情况

的分类模块即可完成分类任务，而送入检测模块即完成目标检测，图3(*b*)为有多种视频分析业务请求时的资源使用情况，模块为矩形表示仅使用前面业务保留的相关计算结果，本次业务没有再用该模块计算资源. EFMR方法的优化逻辑主要包括模型离线训练、功能模块化、网络功能虚拟化、模块重组.

(1)离线训练. 根据业务需求，制作特定数据集，并选择适合的目标检测网络生成检测模型. 本文实验中以AlexNet为骨干网提取特征，以RCNN[12]进行目标检测，通过组合多个SVM(Support Vector Machine)二分类器实现目标的多分类，以17flowers数据集进行预训练，预训练完成后选择标注好的0类和1类花进行微调，并将模型保存. 然后再使用0类和1类花训练SVM分类器和bbox回归器，供后面分类和检测使用.

(2)功能模块化. 在实验中将AlexNet减去最后一层softmax的输出，作为特征提取模块. SVM分类器则作为目标分类识别模块，bbox回归器负责微调检测框的位置，作为检测模块.

(3)网络功能虚拟化. 对(2)中模块化于边缘服务器中的各功能块，在边缘服务器中建立轻量级Docker容器向用户提供服务，即各功能模块是部署在容器中的. Docker容器之间使用macvlan Docker网络模型，即在边缘服务器中创建macvlan网络，运行多个Docker容器，且保证各容器的网关一致，进而实现不同边缘服务器Docker容器之间的通信，从而确保各边缘服务器之间资源的调度.

(4)模块重组. 在(3)中网络虚拟化的基础上，根据用户请求，选择最佳的功能模块组合. 实验中模块重组是通过遍历方式，从最近的边缘服务器开始检查资源使用情况，若可满足请求，则向用户提供服务，同时特征提取模块会保存本次得到的特征，以便下次执行同源视频业务时使用. 否则则通过距离最近的其他边缘服务器提供服务. 若附近边缘服务器均无法提供请求的功能时，则向云端请求，对边缘服务器中功能进行细粒度的模块化扩展.

4 实验及分析

实验环境：边缘服务器的处理器Intel Xeon(R)，3.40GHz，显卡GeForce GTX 1080，内存32GB，操作系统Ubuntu 14.04 LTS，树莓派作为UE，与边缘服务器处于同一局域网内. 实验以RCNN为主要验证方法，AlexNet和VGG16作为骨干网，用17_Category_Flower数据集训练模型，数据集有17种花，每种花有80张图片，整个数据集有1360张图片，训练框架为TensorFlow.

4.1 时延测试

先将离线训练好的完整功能的RCNN检测网络部署在边缘服务器的Docker容器中，用户请求视频分析业务时，由Docker容器提供服务. 骨干网分别选用AlexNet和VGG16，测试边缘服务器从接收到UE请求到得到结果为止的整个过程时间.

表2 RCNN完整检测过程时间

过程	AlexNet	VGG16
疑似出现物体框	12s	12s
检测出目标	15s	18s

表2为分别用AlexNet和VGG16作为RCNN骨干网的目标检测时间. 其中“疑似出现物体框”指先用selective search[14]得到目标初始候选框，再删除重复、太小及长或宽为0的区域，最后筛选得到候选框的过程.

作为对比，将训练好完整功能的RCNN模型，用EFMR方法模块化成三个子模块，分别为CNN特征提取模块、SVM多分类模块及检测模块，依次编号为1、2、3. 若第一次用户请求为目标分类，则选择模块1和2进行组合. 当用户要对同目标再次请求检测业务时，由EFMR方法选择模块组合1和3，由于前面对目标进行过分类业务，CNN特征提取模块已将得到的特征图进行保存，所以此时可直接将特征图和产生的候选框坐标以Socket的方式传送到检测模块进行相关的检测业务，并推理出最终的结果. 表3是以AlexNet为骨干网时边缘服务器各模块两次业务的执行时间. 可见当复用特征提取结果时，检测时间与表2相比减少了2s，约13.3%. 表4是以VGG16为骨干网时的执行时间，可以看到同样的效果.

表3 多次业务请求执行时间(AlexNet)

模块	第一次业务(分类)	再一次业务(检测)
AlexNet特征提取	2s	0s
分类模块	12s	/
检测模块	/	13s
总时间	14s	13s

表4 多次业务请求执行时间(VGG16)

模块	第一次业务(分类)	再一次业务(检测)
VGG16特征提取	2s	0s
分类模块	15s	/
检测模块	/	16s
总时间	17s	16s

为与实际应用场景接近，减少偶然误差，在实验中我们向边缘服务器产生多批次目标分类和检测业务请求，分别对传统的视频处理业务边缘服务器的部署方式和EMFR部署方式处理时间进行统计，采用AlexNet

和VGG16为骨干网的测试结果分别如图4、图5所示.可见EFMR方法在执行多次目标分类或者目标检测业务时,通过计算复用大大节省了推理时间,且因各功能模块间共享数据传输时间很短,实际所用的时间与式(2)的理论分析基本一致.

无论骨干网是AlexNet还是VGG16,当进行多次分类任务和检测业务,本文提出的EFMR边缘服务器部署方法在所用时间上均优于完整深度模型的部署方式,且随着请求次数的逐渐增多,时间优势越明显.

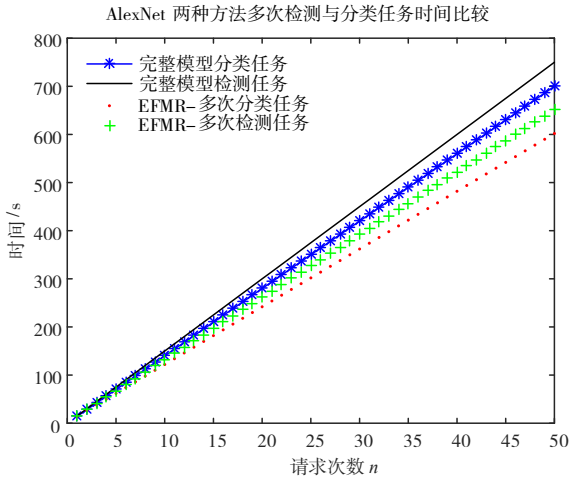


图4　不同部署方式执行时间(Bone:AlexNet)

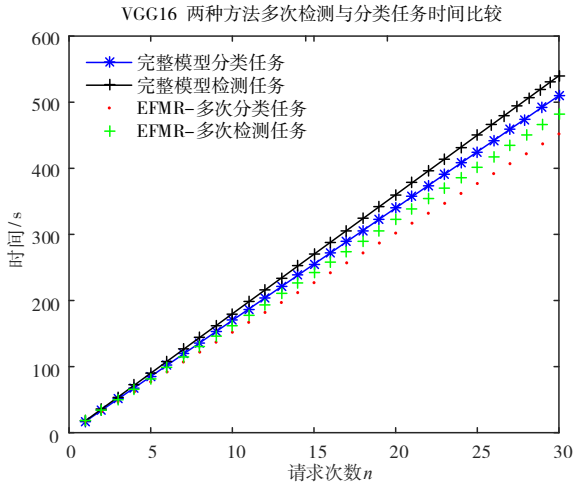


图5　不同部署方式执行时间(Bone:VGG16)

4.2　计算量测试

为比较不同部署方式下的系统运算量,本文在实验中采用FLOPs(floating point operations)浮点运算量作为衡量指标

4.2.1　骨干网为AlexNet的计算量测试

先以AlexNet作为骨干网来进行特征提取,将第二个全连接层之后的输出作为特征,送入后续的分类或者检测模块进行分类或者检测. 表5是实验中AlexNet网络各层参数的设置,输入图片大小为224×224×3,最后一层全连接层的输出类别设置为17,计算可得AlexNet网络的参数数量共有5500W个,五个卷积层所占的计算量为663M,第一个全连接层和第二个全连接层的计算量为53M.

表5　AlexNet网络参数及每层计算量和参数数量

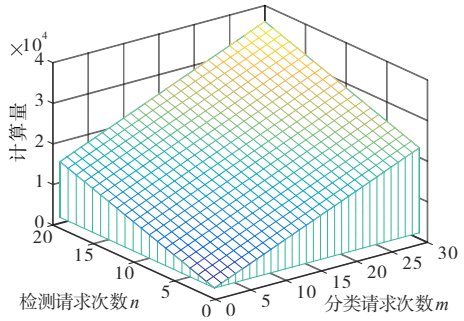
类型	卷积核/步长/通道数	FLOPs	参数量
卷积层(ReLU)	11×11/4/96	105M	35K
最大值池化层	3×3/2	/	/
卷积层(ReLU)	5×5/1/256	223M	307K
最大值池化层	3×3/2	/	/
卷积层(ReLU)	3×3/1/384	149M	884K
卷积层(ReLU)	3×3/1/384	112M	1. 3M
卷积层(ReLU)	3×3/1/256	74M	442K
最大值池化层	3×3/2	/	/
全连接层(ReLU)	4096	37M	37M
全连接层(ReLU)	4096	16M	16M
全连接层(1000)	17	69K	69K

在SVM分类模块中,设定提取2000个候选框,得到2000个4096维度的特征向量,将此矩阵与SVM权值矩阵(4096×3)相乘,进而得到结果. 由于微调网络时为3分类,故SVM分类的计算量为4096×2000×3=24576000,即2. 46M FLOPs.

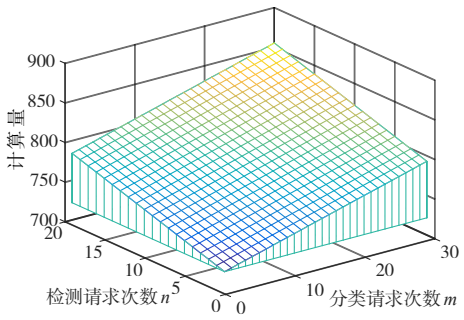
在检测模块中,主要通过边框回归进行位置精修,精确地检测出目标所在的位置. 对每一类目标使用一个线性回归器进行精修,输入为深度网络的全连接层输出的4096维特征,输出为xy方向的缩放和平移,所以检测阶段的主要计算量为4096×2000×4,为3. 28M FLOPs.

故骨干网为AlexNet时,完整的RCNN检测模型的计算量大致为卷积特征提取模块的计算量与分类模块和检测模块的计算量之和,约为721. 74M FLOPs. 在测试时,假设不同用户对同一目标向边缘服务器请求多次分类和检测业务,根据多次测得实际数据进行统计,传统的完整模型部署方法与本文提出的方法边缘服务器的计算量如图6所示.

由图可知, EFMR方法的计算量要远小于传统的部署方式,且EFMR方法实际计算量的消耗与式(4)推导的理论计算值基本一致. 以请求三次分类业务,两次检测业务为例,完整模型部署到边缘服务器的方式,所用的计算量为3608M FLOPs,而按EFMR方法,所用的计算量为730M FLOPs,约为完整部署方法所用计算量的20%左右.



(a) 完整模型部署计算量



(b) EFMR模型部署计算量

图6 两种部署方式边缘服务器计算量(Bone:AlexNet)

4.2.2 骨干网为VGG16的计算量测试

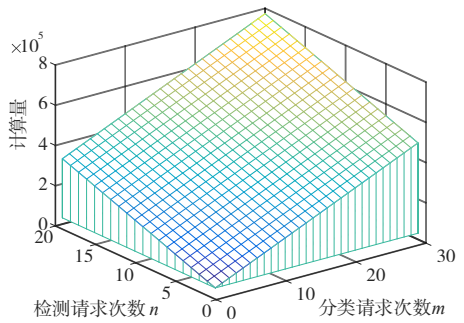
类似地,将特征提取骨干网换成更为复杂的VGG16,各层的参数设置并统计其计算量则如表6,输入图片大小仍然设置为224×224×3,最后一层全连接层的输出类别设置为17. 在前面业务进行推理时,仍然保留最后一个全连接层前各层对图片特征提取的推理结果值,当后续有视频分析的检测业务产生时,直接将保留的特征计算结果送入后续检测任务中,不再进行前面的特征推理.

同理,与4.2.1节过程相同,SVM分类模块所用计算量为2.46M FLOPs,检测过程所用计算量为3.28M FLOPs. 以当骨干网为VGG16时,完整的RCNN检测模型的计算量大致为卷积特征提取模块的计算量与分类模块和检测模块的计算量之和,为15475.74M FLOPs. 用户向边缘服务器请求多次分类和检测任务时,传统的完整模型部署方法与本文EFMR方法边缘服务器的计算量如图7所示.

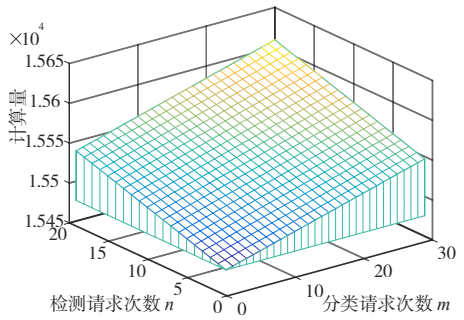
由图可知,EFMR部署方式的计算量同样远小于完整模型部署方法. 且由于VGG16相对AlexNet更为复杂,对边缘服务器计算量节省的比例也更大. 以请求三次分类任务,三次检测任务为例,完整模型部署方式所用的计算量为77378.7M FLOPs,而按本文EFMR方法,所用的计算量为15483.94M FLOPs,约为完整模型部署方法的16.7%左右.

表6 VGG16网络参数及每层计算量和参数数量

类型	卷积核/步长/通道数	FLOPs
卷积层(ReLU)	3×3/1/64	870M
卷积层(ReLU)	3×3/1/64	1849M
最大值池化层	2×2/2	/
卷积层(ReLU)	3×3/1/128	925M
卷积层(ReLU)	3×3/1/128	1850M
最大值池化层	3×3/2	/
卷积层(ReLU)	3×3/1/256	925M
卷积层(ReLU)	3×3/1/256	1850M
卷积层(ReLU)	3×3/1/256	1850M
最大值池化层	3×3/2	/
卷积层(ReLU)	3×3/1/512	925M
卷积层(ReLU)	3×3/1/512	1850M
卷积层(ReLU)	3×3/1/512	1850M
最大值池化层	3×3/2	/
卷积层(ReLU)	3×3/1/512	462M
卷积层(ReLU)	3×3/1/512	462M
卷积层(ReLU)	3×3/1/512	462M
最大值池化层	3×3/2	/
全连接层(ReLU)	4096	103M
全连接层(ReLU)	4096	17M
全连接层	17	0.069M



(a) 完整模型部署计算量



(b) EFMR模型部署计算量

图7 两种部署方式边缘服务器计算量(Bone:VGG16)

可见,EFMR方法不仅降低了边缘服务器的推理时间,并且也节省了整个系统的总计算资源.

5 结束语

本文提出了一种面向大规模视频处理的边缘功能模块化及重组部署方法. 该方法将视频处理业务下沉到网络边缘,利用不同的视频服务请求中的共有属性,结合网络功能虚拟化,将发送到边缘服务器中的视频业务请求根据其内在处理过程相关性进行功能细粒度划分,然后根据划分结果按需匹配资源并与功能模块进行重组,使各模块最大化复用对应的视频分析功能,并使边缘服务器以较小代价扩展新的业务功能. 实验结果表明,EFMR方法相较于传统云端的解决方案,大大的降低了接入与响应时延,同时与传统的边缘服务器部署多个单独完整功能的方法相比,不仅在业务的推理时间上有较明显的减少,而且整个视频推理业务的计算量大大降低,节省了大量边缘服务器的计算资源,提高了视频处理业务的部署速度. 下一步的研究工作会着重研究任务的分解、编排以及调度策略,以便优化边缘服务器的存储以及计算资源的使用等.

参考文献

[1] Abdulsalam Y, Shailendra S, Shamim H M, et al. IoT big data analytics for smart homes with fog and cloud computing[J]. Future Generation Computer Systems, 2019,(2): 563 – 573.

[2] Guo T. Cloud-based or on-device: An empirical study of mobile deep inference[A]. IEEE International Conference on Cloud Engineering (IC2E) [C]. Orlando, USA: IEEE, 2018. 184 – 190.

[3] Zhou Z, Liao H, Gu B, et al. Robust mobile crowd sensing when deep learning meets edge computing[J]. IEEE Network, 2018, 32(4): 54 – 60.

[4] Kang Y, Hauswald J, Gao C, et al. Neurosurgeon: collaborative intelligence between the cloud and mobile edge[J]. ACM SIGPLAN Notices, 2017, 52(4): 615 – 629.

[5] Li H, Shou G, Hu Y, et al. Mobile edge computing: Progress and challenges[A]. IEEE International Conference on Mobile Cloud Computing, Services, and Engineering[C]. Oxford, UK: IEEE, 2016. 83 – 84.

[6] Ahmed A, Ahmed E. A survey on mobile edge computing [A]. International Conference on Intelligent Systems and Control[C]. Coimbatore, India, 2016. 1 – 8.

[7] Huang Y, Ma X, Fan X, et al. When deep learning meets edge computing[A]. IEEE International Conference on Network Protocols[C]. Toronto, Canada: IEEE, 2017. 1 – 2.

[8] Huang Y, Zhu Y, Fan X, et al. Task scheduling with optimized transmission time in collaborative cloud-edge learning[A]. IEEE International Conference on Computer Communication and Networks[C]. Hangzhou, China: IEEE, 2018. 1 – 9.

[9] Li E, Zhou Z, Chen X. Edge intelligence: On-demand deep learning model co-inference with device-edge synergy[A]. Proceedings of the Workshop on Mobile Edge Communications[C]. Budapest, Hungary: ACM, 2018.31 – 36.

[10] Wang Z, Xue G, Qian S, et al. CampEdge: Distributed computation offloading strategy under large-scale Ap-based edge computing system for IoT applications[J]. IEEE Internet of Things Journal, 2020:6733 – 6745.

[11] Campolo C, Iera A, Molinaro A, Ruggeri G. MEC support for 5g-v2x use cases through docker containers[A]. IEEE Wireless Communications and Networking Conference[C]. Marrakesh, Morocco: IEEE, 2019. 1 – 6.

[12] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[A]. IEEE Conference on Computer Vision and Pattern Recognition[C]. Columbus, USA: IEEE, 2014. 580 – 587.

[13] Evgeny A, Denis M, Serge N. Comparison of regularization methods for ImageNet classification with deep convolutional neural networks[J]. AASRI Procedia, 2014,(6): 89 – 94.

[14] Uijlings JRR, van de Sande KEA, Gevers T, et al. Selective search for object recognition[J]. International Journal of Computer Vision, 2013, 104(2): 154 – 171.

作者简介

覃　剑　男,博士,1977年5月生,陕西宝鸡人.重庆大学微电子与通信工程学院副教授,研究方向为视频分析及传输.
E-mail:qinjian@cqu.edu.cn

石昌伟　男,硕士,1993年5月生,山东菏泽人.重庆大学微电子与通信工程学院,研究方向为云计算与图像处理
E-mail:593778745@qq.com