doi:10.3772/j.issn.1002-0470.2024.03.001

Cloudless-Training:基于 serverless 的高效跨地域分布式 ML 训练框架①

谭文婷②*** 吕存驰*** 史 骁③**** 赵晓芳*****

(*中国科学院计算技术研究所 北京 100190)

(**中国科学院大学 北京 100049)

(***中科南京信息高铁研究院 南京 211135)

(****中科苏州智能计算技术研究院 苏州 215028)

摘 要 跨地域分布式机器学习(ML)训练能够联合多区域的云资源协作训练,可满足许多新兴 ML 场景(比如大型模型训练、联邦学习)的训练需求。但其训练效率仍受 2 方面挑战的制约。首先,多区域云资源缺乏有效的弹性调度,这会影响训练的资源利用率和性能;其次,模型跨地域同步需要在广域网(WAN)上高频通信,受 WAN 的低带宽和高波动的影响,会产生巨大通信开销。本文提出 Cloudless-Training,从 3 个方面实现高效的跨地域分布式 ML 训练。首先,它基于 serverless 计算模式实现,使用控制层和训练执行层的 2 层架构,支持多云区域的弹性调度和通信。其次,它提供一种弹性调度策略,根据可用云资源的异构性和训练数据集的分布自适应地部署训练工作流。最后,它提供了 2 种高效的跨云同步策略,包括基于梯度累积的异步随机梯度下降(ASGD-GA)和跨云参数服务器(PS)间的模型平均(MA)。Cloudless-Training 是基于 OpenFaaS 实现的,并被部署在腾讯云上评估,实验结果表明 Cloudless-Training 可显著地提高跨地域分布式 ML 训练的资源利用率(训练成本降低了 9.2% ~24.0%)和同步效率(训练速度最多比基线快 1.7 倍),并能保证模型的收敛精度。

关键词 跨地域分布式机器学习(ML)训练;跨云 ML 训练;分布式训练框架;serverless;跨云模型同步

跨地域分布式机器学习(machine learning,ML)训练系统能够支持大规模训练,以及多地域弹性云资源和分散数据集的联合训练。例如,单个云中的当前可用资源可能无法满足大规模训练;多模态的训练数据集(如用户行为日志、图像、视频等)以极快的速率生成并存储于在世界各地的云中,而出于商业或者隐私保护原因无法迁移到一起。这些都促使了在多区域上进行跨地域的分布式 ML 训练。目前,它已被应用到许多新兴的 ML 场景中,例如大型模型训练[1]、联邦学习[2]以及边缘云协同[3]或混合云[4]环境中的 ML 训练。

然而,实现高效的跨地域分布式 ML 训练并非易事。首先,跨地域环境中存在资源异构和数据集分布不均,且多区域云资源缺乏有效弹性调度易造成云之间负载不平衡,会直接减少训练的资源利用率、速度与效果。例如,在腾讯云[5]的 2 个云区域(例如上海和重庆)上训练模型 LeNet[6],2 个云区域的中央处理器(central processing unit, CPU)核数相同,但是分布的数据集大小不等,这导致一个云区域中 25% 的资源被过度调配。其次,跨地域间的模

① 国家重点研发计划(2021YFF0703800)和光合基金 B 类(202302028357)资助项目。

② 女,1988 年生,博士生;研究方向:云计算,serverless 计算,分布式训练;E-mail: tanwenting@ict.ac.cn。

③ 通信作者,E-mail: shixiao@ict.ac.cn。

(收稿日期:2023-02-18)

型同步需在低带宽的广域网(wide area network,WAN)上高频通信,造成大量的时间和经济负担。在基于参数服务器(parameter server, PS)架构[7]的跨地域分布式 ML 训练中,各云的 PS 节点之间需要定期通信以保持模型参数的一致性,通信开销取决于模型大小和网络带宽,而网络带宽在云上却是非常稀缺和昂贵的资源。在腾讯云的 2 个云区域(它们间的通信带宽最大为 100 Mbps)上使用图形处理器(graphics processing unit, GPU)训练 ResNet 18[8],WAN 上的通信时间就占总了时间的 98% 以上。

现有跨地域分布式 ML 训练系统的相关研究[8-9]提出了改善 WAN 上同步效率的方案,但是并没有解决上述负载失衡问题。在同步效率优化问题上,其模型具有启发性和深入研究的价值。文献[9]将云内和云间的同步模式解耦,并提出了一种新的 ML 同步模型:近似同步并行(approximate synchronous parallel,ASP),对同步信息设定重要性判断阈值,过滤掉不重要的同步操作,以减少通信次数。文献[10]采用自适应通信加速技术以及数据压缩方法,以减少同步操作的通信数据量。

为提高跨地域分布式 ML 训练中的资源利用率和同步效率,本文提出一种基于 serverless 计算[11]模式构建的训练框架:Cloudless-Training。该框架可支持跨地域的分布式 ML 训练,并提供一种面向负载均衡的弹性调度算法以解决跨地域训练中负载不均的问题,同时提出了 2 种跨地域模型同步策略以降低通信开销。

本文的主要贡献总结为以下 4 点。

(1) 采用 serverless 计算模式构建了 2 层的基于 PS 架构的跨地域分布式 ML 训练框架,可显著提高训练的资源利用率和同步效率;

(2)通过对可用云资源的异构性和训练数据集的分布进行负载建模,提供了一种可保障云间负载平衡的资源弹性调度策略;

(3)提供了 2 种同步策略支持多区域云间的模型同步,包括基于梯度累积的异步随机梯度下降(asynchronous stochastic gradient descent (SGD) with gradient accumulation,ASGD-GA)和模型平均(model average,MA),两者都可显著降低 WAN 的通信开销;

(4)将 Cloudless-Training 部署到具有多区域的公有云上并对其进行评估,结果表明它可以有效减少跨地域分布式 ML 训练的训练开销(例如,训练成本降低了 9.2% ~24.0%)和提高同步效率(例如,训练速度最多比基线快 1.7 倍),并保证模型准确性。

本文的组织结构如下。第 1 节介绍跨地域分布式 ML 训练的挑战和目标;第 2 节阐述 Cloudless-Training 的系统设计;第 3 节简述其实现方式;第 4 节实验验证 Cloudless-Training 的有效性;第 5 节总结全文。

1 背 景

1.1 跨地域分布式 ML 训练

相对于常规 ML 应用场景,很多新兴 ML 应用场景都对训练系统提出了全新要求,这成为发展高性能、大规模或协作式训练的跨地域分布式 ML 训练的极大推力。本文总结了 2 个基本要求。

(1) 要求 1:灵活获取比单个云更多的云资源。随着模型和数据集大小的不断增长,单个云中的实时可用资源可能无法满足大型模型训练的计算需求。人工智能(artificial intelligence,AI)应用不断要求更高数量和质量的计算资源[12-13],例如 Nvidia 开发的 Megatron LM 模型具有 83 亿个参数,需在 512 个 Nvidia Tesla V100 上对 174 GB 大小语料库训练[14],这使得小规模的云数据中心有时不能满足大模型训练的计算资源需求。因此,可将训练任务划分到更多的云上,以利用更多的云资源,跨地域分布式 ML 训练正是基于这种思想,可用于支持边缘云[15]、混合云[4]或多云环境中各种模型的训练。

(2) 要求 2:利用分布在多区域云上的数据进行训练。考虑到延迟或者服务质量(quality of service, QoS),很多应用分布于多个云区域部署,以便尽可能接近终端用户。用户数据(例如,用户行为日志、图像和视频)可以在世界各地以极高的速率产生并被保存[16-17],这些数据对于构建 AI 服务很有价值,但是将所有数据收集在一起面临着很大的

实现挑战,如 WAN 上的低带宽和高成本传输,以及严格的数据隐私规则。因此,跨地域分布式 ML 训练需要支持跨多个云协作训练。

为满足上述需要,构造高效的跨地域分布式 ML 训练框架十分必要,训练框架如图 1 所示。同时,该框架要求可扩展性和启发性,以便进一步地研究与实践。分布式 ML 训练中常见的并行训练模式包括数据并行[18]、模型并行[19]和流水线并行[20],这项工作专注基于 PS 架构[7]的数据并行训练。针对基于 PS 架构的跨地域分布式 ML 训练,虽然已有研究[9-10]给出了很多解决方案,但其工作效率还比较有限。本工作认为 serverless 计算[11]模式可以简化训练框架构建,并且其自动可扩展和细粒度云函数的资源调配方式有助于更好地满足上述需求。Serverless 计算框架如 Kubeless[21]、OpenWhisk[22]、OpenFaas[23]和 Lambda[24],以事件驱动方式运行,可实现弹性计算,吸引了许多应用场景基于其实施。诸多研究利用 serverless 来提高分布式 ML 训练的弹性效率,如 Cirrus[25]和 LambdaML[26]。另外本文观察到负载不平衡以及通信开销是造成跨地域分布式 ML 训练实施困难的两大原因。

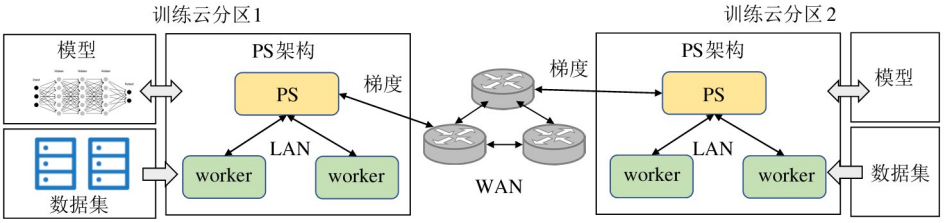

图 1　跨地域分布式 ML 训练系统的架构

1.2　挑战 1:跨地域弹性调度

跨地域分布式 ML 训练过程中易出现云间负载不平衡,需要高效的资源弹性调度。

多区域云间的负载不平衡。各云中训练任务的资源分配方案通常基于经验性的或粗略的(例如,使用贪婪策略)。多区域云环境中资源的异构性和数据分布的不均匀性,导致难以得到适当的资源分配方案,易造成负载失衡。该问题会显著降低训练的资源利用率、速度和效果,如图2所示,随着数据分布比率与可用计算资源的类型和数量的变化,出现负载失衡,训练速度完全由负载最大的云区域一方决定。负载较小的云区域一方在等待掉队者赶上的期间也需要保留计算资源,从而导致不必要的资源消耗。这同时也放大了云之间训练步调的不一致性,影响模型的收敛。由于会从掉队者接收到严重过时梯度或者参数,AdamLike[27]表明梯度滞后严重阻碍了模型收敛。

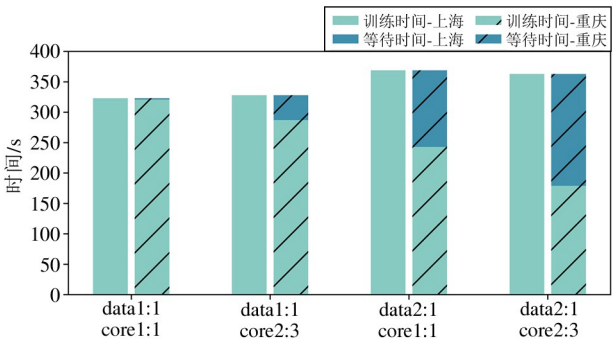

图 2　在腾讯云的上海和重庆地区,在 4 类不同的异构资源分配方案和不均匀数据分布的情况下训练 LeNet 的时间比例

为提高跨地域分布式 ML 训练的资源利用率、速度和效果,需平衡云之间的相对负载,以便各云中的训练呈现出更协调的处理模式。但以往研究一般假定训练中云之间利用同构资源以及数据分布均匀这类负载平衡的环境,如文献[9]。为动态地为每个训练任务匹配负载均衡的调度,本文构建的模型能够根据资源的异构和数据的分布,对负载状况进行定量的描述。另外,为提高资源扩展效率,本文主张采用 serverless 模式,采用按需的方式分配与回收 worker 云函数,减少等待掉队者的时间。

1.3　挑战 2:减少 WAN 上的同步开销

在跨地域分布式 ML 训练中,在 WAN 上同步训练中间结果(如梯度或者模型参数)的开销要比在

局域网(local area network,LAN)上大得多,而且同步操作是周期性的(训练是反复迭代的过程),因此同步开销不容忽视。

WAN 上的同步开销。在 ML 训练中,worker 需要定期与 PS 或其他 worker 通信,以保持模型一致性。跨地域分布式 ML 训练是通过 WAN 进行云间通信,而 WAN 通常只能提供数百 Mbps(例如公有云中是 100 Mbps)的带宽,比云内 LAN 的带宽最少低 20 倍[28],这极大增加了训练中的通信开销比例。如图 3 所示,在腾讯云的重庆和上海地区训练 ResNet 18[8](模型大小 48 MB),用 CPU 和 GPU 时云间通信时间分别占总训练时间的 64.9% 和 98.4%,这不仅减缓了训练过程,还给用户造成了更多的经济成本。

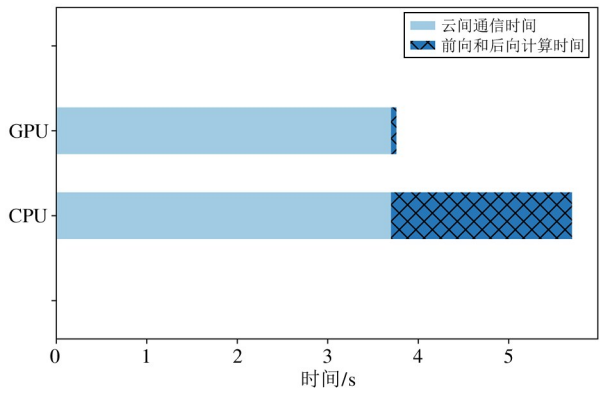

图 3　在腾讯云的上海和重庆 2 个地区使用 CPU 或 GPU 跨域训练 ResNet 18 的时间比例

降低 WAN 上的同步开销可以得到更好的训练速度以及更低的开销。在对通用 ML 训练系统的同步优化问题进行研究时,常用方法有 2 种:压缩同步数据的大小(如 DGC (deep gradient compression)[29]、top-K[30])和降低同步的频率(如梯度累积[31]、模型平均[32])。梯度累积策略中,计算梯度后并不立即同步,只在到达同步阈值时才执行同步操作;模型平均策略中,worker 仅需要周期性地与并行 worker 执行模型平均操作而不需要每次迭代均通信。Gaia[9]采用降低同步频率的方式减少通信开销,其提出的近似同步并行(approximate synchronous parallel,ASP)策略,设置参数更新的重要性判断阈值,以过滤掉不重要的同步操作。然而细微的参数更新也可能较大程度影响模型收敛,因此参数更新重要性阈值难以精确界定。本文一方面实现了服务于云函数(Cloudless Training 的 serverless 架构)的通信寻址机制;另一方面采用了 2 种策略降低同步频率,包括基于梯度累积的异步 SGD 和跨云 PS 间的模型平均。

2　Cloudless-Training 设计

2.1　框架概述

Cloudless-Training 提供了基于 PS 架构的跨地域分布式 ML 训练的整体视图。它采用统一的逻辑视图管理所有参与训练的云区域,该视图分为 2 层,由控制层与训练执行层组成,如图 4 所示。相对于传统的跨地域分布式 ML 训练框架,Cloudless-Training 是基于 serverless 计算模式构建的,训练工作流由一组云函数组成。框架的控制层主要负责调度训练任务并支持所有云区域之间的通信寻址,各云区域训练分区可以并行执行,并可获取同伴云区域的 PScommunicator 云函数的全局通信地址。控制层可部署在任意云区域上,用户通过控制层提交训练任务,任务的设置包括模型定义和训练配置。训练执行层是由 serverless 训练工作流组建的训练分区,承担特定训练任务,需部署在每一个云区域上。训练分区间并行执行任务,训练工作流通过 communicator 功能来与其他云中的训练分区通信,以完成全局模型同步。

训练工作流的跨地域支持。相对于基础的 serverless 训练工作流[33],本文主要作了 2 方面的扩展以支持跨地域的分布式 ML 训练。首先,在控制层中插入调度云函数和全局 communicator 寻址云函数以支持跨云协作工作,它们会在任务启动阶段被触发;其次,扩展 PS 云函数的功能以支持在 WAN 上同步,各云的 PS 云函数可通过 communicator 功能在 WAN 上暴露身份信息并与其他云区域的 PS 云函数共享中间状态。

框架中的训练工作流以 serverless 模式运行。当训练请求到达时,调度云函数先加载调度策略,为各训练云区域生成训练资源分配方案,并在各云中调用 serverless 训练工作流。之后全局 communicator

寻址云函数等待各云中的 PS 云函数就绪,并为每个 PS 云函数的 communicator 分配通信地址,将其 serverless 身份映射到 WAN 上的 < IP,Port > 。

在执行完以上初始化工作之后,在各训练云区域部署相应的训练云函数,执行本地训练任务,包括加载本地训练数据、计算梯度和更新本地模型参数。当达到同步状态时(例如,每 10 个迭代),PS 云函数的 communicator 将本地模型状态(梯度或者模型参数)发送给另一个云区域的 PS 云函数。当 PS 云函数接收到远程模型状态后,会用其来更新本地的模型参数,在本地训练完成后(例如,当训练轮次(epoch)数或损失值(loss)达到阈值时),为减少资源消耗,会立即回收所有云中的 worker 函数。

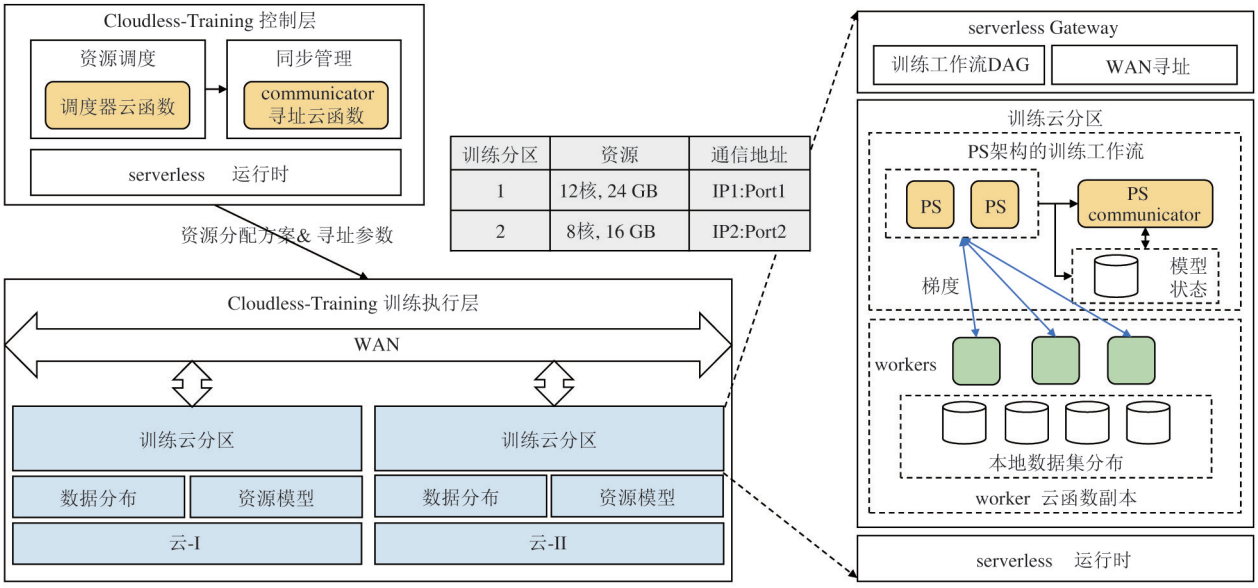

图 4　Cloudless-Training 系统架构图

调度机制。框架的调度机制能支持中心式调度策略,还可通过更新调度云函数的实现来扩展更多策略。本文提出了一种弹性调度策略来实现云间负载平衡的资源分配,具体步骤将在 2.2 节详细介绍。该策略可根据检测或手动设置的可用云资源的异构性和数据分布的情况,利用负载模型评估云的负载能力,然后根据弹性调度算法生成适当的资源分配方案。

同步支持。在传统的 ML 训练中,PS 云函数负责管理模型状态,所以每个 PS 在其本地 serverless 运行时都有一个地址,以便本地 worker 云函数访问。对于 WAN 通信,全局 communicator 寻址云函数需要为每个用于 WAN 通信的 PScommunicator 分配唯一标识。此外,它提供了 2 种同步策略:基于梯度累积的异步 SGD 和跨云 PS 间的模型平均策略,可以在保证训练正确性的情况下减少云间的同步频率,策略将在 2.3 节详细介绍。

2.2　弹性调度策略

弹性调度策略受不同训练阶段的时间组成启发,它定量地描述了相对计算负载,以便均衡关键计算阶段的时间消耗,进而构成启发式调度算法。

调度洞察和负载建模。达到每次 WAN 上同步状态前的这段时间被视为本地云内计算周期($T_{process}$)。在这段时间内,各训练分区会完成模型加载(T_{load})、本地训练(T_{train},包括前向和后向计算、云内通信以及其他训练分区完成的等待时间)。由于训练过程是有状态的,在训练期间不会释放云资源,所以等待时间会造成不必要的资源消耗。它们的关系可以表示为 $T_{process} = T_{load} + T_{train}$,如前文所述,各云之间的 $T_{process}$ 差异较大会导致训练步调不一致,可能无法保障模型最终收敛。从文献[34]和训练经验可知,模型训练中,T_{train} 占 $T_{process}$ 的主要部分直接影响负载均衡状态。

负载均衡状态可粗略根据云资源分配与数据分

配情况预估出来。通过对云中的一些计算设备(包括CPU和GPU)采样,收集了它们的规格信息(例如CPU和CUDA的型号、数量和TFLOPS)以量化其计算能力,并观察它们在实际训练中的性能(例如在各种设备类型上用 CIFAR-10[35] 来训练 ResNet 18 的单次迭代时间),结果如表1所示。从表格中可知,一次训练迭代时间 T_{train}(IN 为量化指标)与设备的计算能力 P_{device}(TN 为量化指标)成反比,又因为 T_{train} 与数据集大小 S_{data} 成正比,因此可以得到它们三者的关系为 $T_{\text{train}} \propto \frac{S_{\text{data}}}{P_{\text{device}}}$。对于云资源设备 P_{device} 可以基于它们的 TFLOPS 值(TN)来进行粗略的预估。

表1 云资源的训练速度量化

CPU/GPU	核数	*TFLOPS/TN*	迭代时间(s)/*IN*	*IN/TN* 比例
Intel Xeon IceLake (baseline)	2	0.096/1.000	3.697/1.000	1.000
Intel Xeon Cascade Lake	2	0.090/0.938	5.549/0.666	0.710
Intel Xeon Skylake	2	0.112/1.167	3.800/0.973	0.834
Nvidia T4	2 560	5.554/57.854	0.062/59.629	1.031
Nvidia V100	5 120	13.345/139.010	0.024/154.042	1.108

基于上述基础,可以定义云区域 i(跨域中的某个云区域)的训练负载能力(load power, LP),它可描述云区域 i 使用特定资源处理特定负载的能力,可粗略地量化为现有可用设备(例如,CPU、GPU)的计算能力总和与本地数据集大小上的比例,如式(1)所示。

$$LP_i = \frac{\sum_{m=1}^{M} N_{\text{cpu},m} \cdot P_m + \sum_{n=1}^{N} N_{\text{gpu},n} \cdot P_n}{S_{\text{data}}} \tag{1}$$

式中,M 和 N 表示可分配的 CPU 和 GPU 的类型个数,可被保留或使用;$N_{\text{cpu},m}$ 代表第 m 种类型 CPU 的数量;P_m 表示第 m 种类型的 CPU 的计算能力;$N_{\text{gpu},n}$ 代表第 n 种类型 GPU 的数量;P_n 表示第 n 种类型的 GPU 的计算能力;S_{data} 表示数据集的大小。公式可用于预估各种资源分配方案下训练的相对负载状态,是调度算法的一个重要指标。

弹性调度算法。本文还提出了一种弹性调度算法来规划各云区域的训练资源分配方案,以获得负载平衡的弹性调度,如算法 1 所示。算法的关键思想是对比各云区域的负载能力单位 (LP_i),找到最小的负载单位,这可能是训练过程中最慢的掉队者。并将其作为参考对象,通过暴力穷举的方式遍历各云区域的可用资源并计算出与掉队者相匹配的实际所需的资源数量(search _ optimal _ plan( ))。

算法 1 最优匹配算法

```
输入:
    N:参与训练的云区域的个数
    Res[N]:各云的可用计算资源
    S_data[N]:各云的数据集分布大小
输出:
    ResPlan[N]:各云的资源分配方案

LP[N] = {} //训练负载能力列表,初始值为空列表
MinLP = +∞ //初始值为无重大
ResPlan[N] = {} //初始值为空列表
for i, res in Res[N] do
    根据式(1)计算云区域 i 的算力负载能力
LP[i]
    找到最小的负载单位 MinLP: min (MinLP,
LP[i])
end for
for i, res in Res[N] do
    计算出与最小负载单位相匹配的资源分配
方案 ResPlan[i]:search _ optimal _ plan(res)
end for
return ResPlan[N]
```

2.3 同步优化

Cloudless-Training 支持 WAN 通信寻址,为训练工作流提供 WAN 上的基础同步机制,并提供 2 种同步策略,即基于梯度累积的异步 SGD(ASGD-GA)和跨云 PS 间的模型平均方法(MA)。这 2 种策略可在不损害模型收敛精度的情况下降低同步频率,

从而减少通信开销,分别支持传输梯度和模型参数。基于梯度累积的异步 SGD 为训练分区提供了异步协调策略,这在以往的工作中通常是同步进行的,并集成了梯度累积以降低信息损失,而 MA 是一种面向传输模型参数的方法。

在 WAN 上的同步机制中,各云的 PS 云函数会周期性地同步模型状态,它们的 communicator 互为通信的发送方和接收方,同步流程如下:(1)worker 云函数从 Local PS 云函数中拉取最新模型参数,计算梯度,并将梯度推送到 Local PS 云函数;(2)Local PS 云函数接收到梯度后,更新模型;(3)Local PS 云函数判断是否需要跨云同步,如果不需要本轮迭代完成;(4)如果需要,根据同步策略执行基于梯度累积的异步 SGD 或者 MA 同步策略;(5)Local PS 云函数的 communicator 根据同步策略,将同步信息发送给接收方 PS 云函数;(6)接收方 PS 云函数根据同步策略更新模型参数。

基于梯度累积的异步 SGD 和 MA 2 种策略的同步流程中同步条件、发送的状态(梯度或模型参数)、通信模式(所有训练分区之间的同步或异步)、更新算法(SGD 或模型平均)都有所不同。

基于梯度累积的异步 SGD 策略专注于传输梯度来同步模型,如图 5 所示。其同步条件被定义为同步频率变量,当达到同步状态时,发送本地的累积梯度;如果未达到同步状态,云内每次迭代的梯度会被累积保存。其通信模式是异步的,因此无需阻塞训练进程以等待同伴。当接收方 PS 函数接收到同步请求,使用 SGD 算法更新模型参数。

Inter-PS 模型平均(MA)策略。在分布式 ML 训练系统中,模型平均是一种周期性的同步策略,可减少 worker 之间的通信频率。本文利用该机制来降低跨地域 PS 之间的通信频率,当达到同步状态时,发送方 PS 云函数的 communicator sender 会发送模型参数,接收方 PS 云函数 communicator receiver 接收到请求后执行平均运算。MA 同步模式支持异步和同步 2 种通信,同步的通信模式基于同步屏障为基础实现。如图 6 所示。

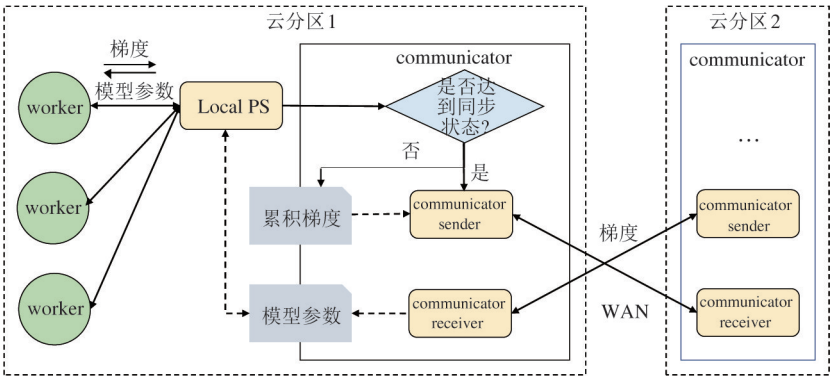

图 5　基于梯度累积的异步 SGD 同步策略的同步流程

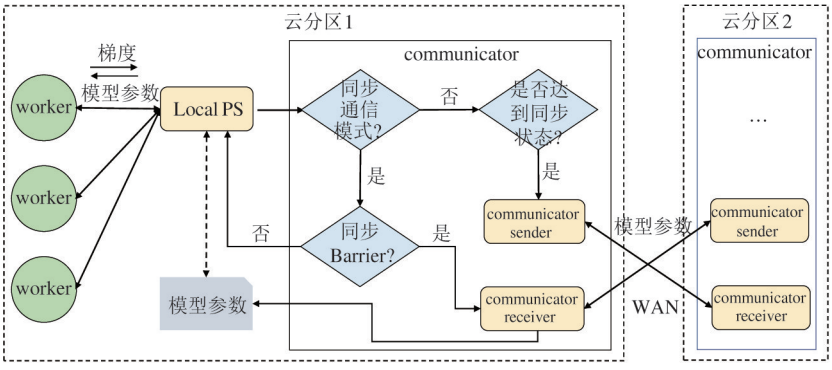

图 6　MA 同步策略的同步流程

3 实 现

Cloudless-Training 是基于 serverless 基础计算框架 OpenFaaS 实现的。具体实施分两部分,一部分是 OpenFaaS 跨云功能定制化开发,另一部分是训练工作流的跨云功能开发。

针对 OpenFaaS 的定制化开发,本文做了 3 方面的扩展。首先,在 OpenFaaS 中添加工作流类型的新实体,支持工作流有向无环图(directed acyclic graph,DAG)的定义,以支持 serverless 工作流的部署和调用。其次,扩展 OpenFaaS 的 gateway 组件以支持工作流和工作流内部云函数的调用。最后,支持云函数的寻址和身份识别,本文在 OpenFaas 中维护一个云函数寻址表,它存储云函数的各副本的标识、名称、命名空间和 endpoint。此处难点在于函数副本的 endpoint 是动态变化的,寻址表需要实时更新。

训练工作流的开发包括控制层工作流和训练执行层工作流。在 Cloudless-Training 2 层的基于 serverless 的分布式训练框架中,控制层与训练执行层由 serverless 工作流组成。工作流是包含一组云函数的集合,云函数都是基于 OpenFaaS 提供的 Python 云函数模板构建的。其中,控制层工作流主要由调度云函数和 communicator 寻址云函数组成,它们都被实现成具有后端内存存储的有状态云函数。训练执行层工作流采用 PS 架构,包括 PS 云函数和 worker 云函数,其中 PS 云函数负责参数管理,worker 云函数负责具体的梯度计算。本文迁移了 ElasticDL[36] 的 PS 训练模块到 serverless 模式中,基于 PS 和 worker 的实现代码构建了对应的云函数。另外 PS 云函数的 communicator 实现为 gRPC 的 sender 和 receiver,用于支持同步策略的实现。本文利用 TensorFlow 来提供具体的训练计算接口。

4 实验与分析

本文在公有云供应商腾讯云上,对 Cloudless-Training 进行了可行性、弹性调度策略和同步策略性能的评估。实验结果表明,Cloudless-Training 降低了跨地域分布式 ML 训练的训练开销(例如,降低了 9.2% ~24.0% 的训练成本)和提升了通信性能(例如,训练速度提升了 1.7 倍),并保障了模型的准确性。

在腾讯云的上海和重庆地区(分别位于中国东部和西部)搭建了 2 个 serverless 集群,在集群上部署 Cloudless-Training。集群之间的 WAN 带宽为 100 Mbps,这是腾讯云能提供的最大带宽。训练使用了 2 种类型的 CPU:Intel Xeon Cascade Lake(Cascade)和 Intel Xeon Skylake(Sky)。

如表 2 所示,实验使用 3 个 ML 模型对 Cloudless-Training 进行了评估,包括 LeNet[6]、ResNet[8] 和 DeepFM[37]。为降低在腾讯云上实验的经济成本,ResNet 是 ResNet 18 模型的变体,其过滤器减少了 4 倍。ResNet 训练过程中学习率(learning rate,lr)每隔 10 个轮次衰减 0.10。

表 2 实验中使用的模型和数据集说明

模型	梯度大小/MB	数据集和大小	训练设置
LeNet	0.4	MNIST(60×10^3)	Epoch = 10,lr = 0.01
ResNet	0.6	CIFAR-10(50×10^3)	Epoch = 50,lr = 0.01
DeepFM	2.4	Frappe(200×10^3)	Epoch = 20,lr = 0.10

实验中收集了训练时间、模型准确性、训练成本和 WAN 上的通信时间作为评价指标,具体说明见表 3。

表 3 实验中所用指标说明

指标名称	指标描述	计算方式
训练时间	评价训练性能/速度	完成指定轮次数量的时间,单位:s
WAN 上通信时间	评价同步的性能/速度	通信时间 = 单次模型同步时间 × 总同步次数,单位:s
精度(accuracy)	评价训练的模型效率	通用精度计算公式所得
损失值(loss)	评价训练的模型效率	通用损失值计算公式所得
训练成本	评价训练的资源利用率	训练成本 = CPU 单价 × CPU 数量 × 训练时间。为了方便对比,图表绘制时基于 baseline 做了归一化处理

4.1 可行性验证实验

为评估 Cloudless-Training 的可行性,将其与传统的单云分布式 ML 训练框架进行了对比,评价指标是模型训练后的精度(accuracy)和损失值(loss)。首先在上海地区部署一个基于 PS 架构的传统 ML 训练框架,可用计算资源包括 24 个核的 CPU(Cascade)和 48 GB RAM。Cloudless-Training 部署在分别配置 12 个核的 CPU(Cascade)和 24 GB RAM 的 2 个云区域上(上海和重庆)。训练每个模型时,单云和跨地域训练使用相同的资源配置。

如图 7 所示,Cloudless-Training 能成功训练所有的模型,且在所有模型训练中,其表现了与单云训练相近的精度收敛趋势和 loss 下降趋势。模型 LeNet、ResNet 和 DeepFM 在 Cloudless-Training 上的训练精度为 0.986 4、0.790 0 和 0.880 0,和在单云框架上的训练精度 0.985 1、0.780 0 和 0.840 0 非常接近,详见表 4。

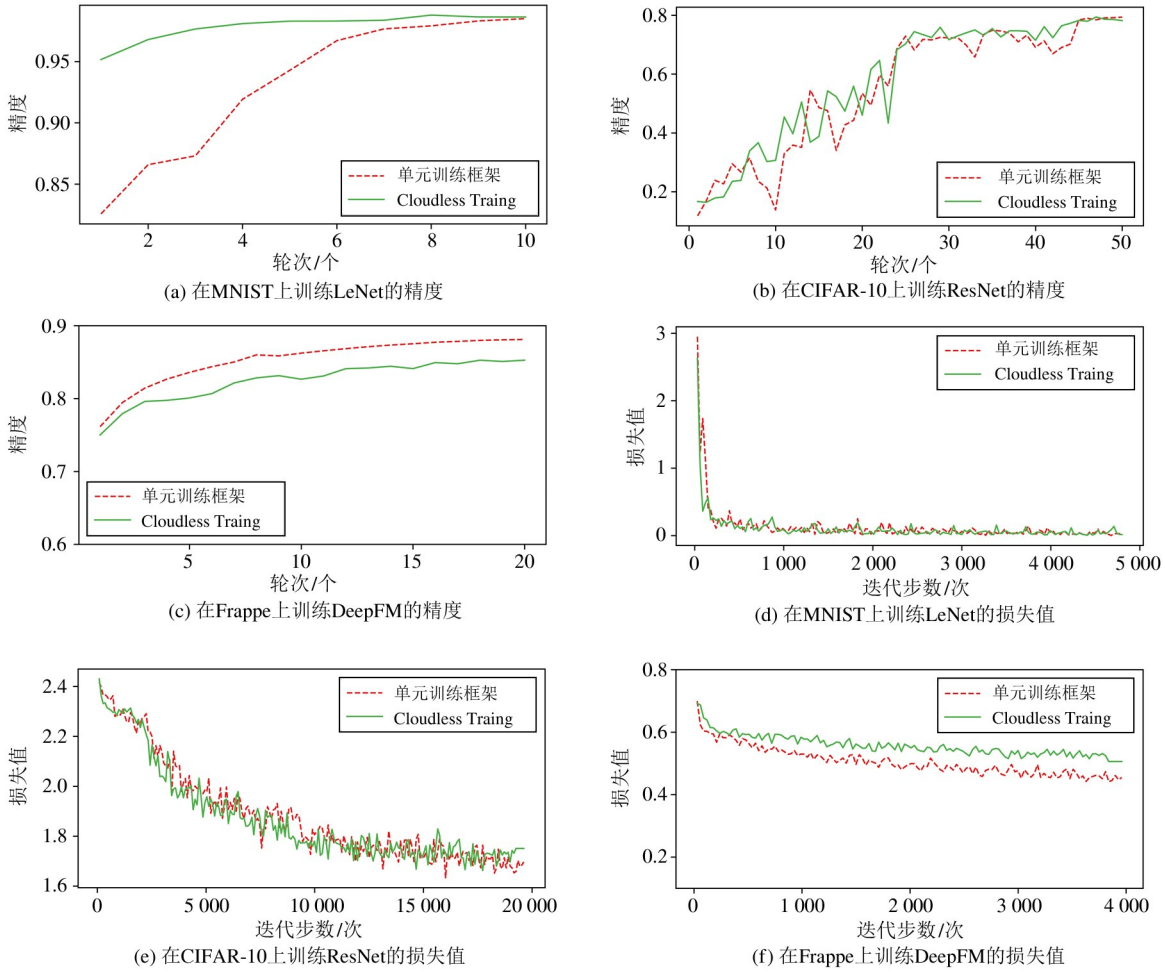

图 7 LeNet、ResNet、DeepFM 模型在 Cloudless-Training 上跨地域训练(腾讯云上海和重庆地区)与常规的单云分布式训练(腾讯云的上海地区)的精度和 loss 值对比

表 4 不同模型在 2 种实验对象上的精度对比

框架	LeNet 精度	ResNet 精度	DeepFM 精度
单云训练框架	0.985 1	0.780 0	0.880 0
Cloudless-Training	0.986 4	0.790 0	0.840 0

4.2 弹性调度策略验证实验

为了评估弹性调度策略的性能,本文在上海(SH)和重庆(CQ)2 个地区分别使用 Cascade 和 Sky 这 2 类不同的 CPU,每个地区最多可以提供 12 个核。根据上述的训练负载公式,这 2 种类型的 CPU 负载能力比约为 2:3。通过调整数据分布比例和资

源分配方案中 CPU 类型以及核数,实验可以分为 3 组,如表 5 所示。Baseline 实验的资源分配方案是基于贪婪策略,即使用所有可用的 24 个 CPU 核,每个区域 12 个。基于弹性调度策略生成的资源分配方案会使用掉队者云区域的所有 CPU 核数,并减少了其他云区域的实际分配 CPU 核数。

表 5　基于弹性调度算法的资源分配表

ID	数据分布比例	Baseline 的资源分配方案(SH/CQ)	弹性策略算法生成的资源分配方案(SH/CQ)
案例 1	1:1	Cascade/Sky 12:12	Cascade/Sky 12:8
案例 2	2:1	Cascade/Cascade 12:12	Cascade/Cascade 12:6
案例 3	2:1	Cascade/Sky 12:12	Cascade/Sky 12:4

实验结果如图 8 和图 9 所示,Cloudless-Training 在使用弹性调度策略的资源分配方案后,它的资源利用率被显著提高,取得更优的模型收敛精度。

首先,如图 8 所示,训练的资源利用率被显著提升,同时训练经济成本被明显减少。图 8(a)~(c)所示为使用弹性调度策略前后的训练时间占比(包括掉队者的有效执行时间和等待时间)。从图中可以看到,每种案例的总训练时间基本与 baseline 相等,而它们的等待时间被显著缩短(由于 WAN 上的网络波动,总的训练用时有时可能会略高于 baseline)。表 6 列出了使用弹性调度策略后的指标优化程度,LeNet 的等待时间被缩短了 46.0%~82.6%,ResNet 的等待时间被缩短了 82.3%~94.6%,DeepFM 的等待时间被缩短了 6.8%~26.0%。由于 DeepFM 比其他 2 种模型更具通信密集性,因此其改进不如其他模型明显。从图 8(d)~(f)可以看出,在提高资源利用率的基础上,使用弹性调度策略降低了因等待掉队者而产生的训练成本。例如,LeNet 的培训成本降低了 13.8%~16.0%,ResNet 降低了 9.2%~15.7%,DeepFM 降低了 13.4%~24.0%。

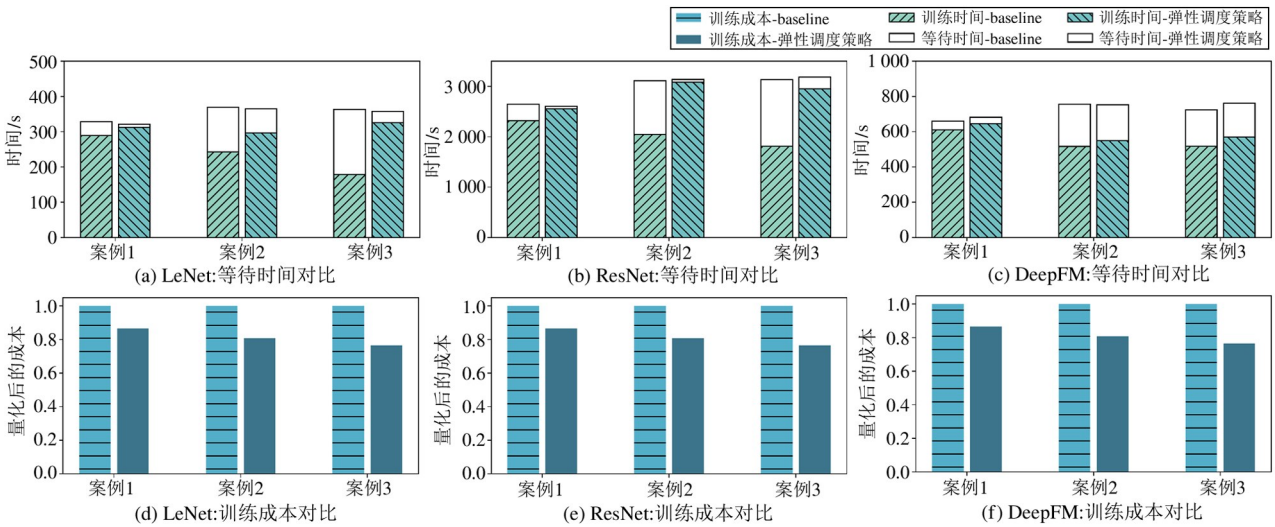

图 8　在腾讯云上海和重庆地区不同数据分布(1:1、1:2、2:1)和 CPU 核数分配的 3 种情况下,使用和不使用弹性调度的培训时间和成本比较

其次,从图 9 可以看到,模型收敛精度和收敛速度得到了提升。Cloudless-Training 大多数精度收敛曲线都略高于 baseline,且收敛速度快于 baseline,振动也少。这得益于调度算法有助于平衡云之间的训练步调,从而减少过时梯度的影响。

4.3　同步策略验证实验

为了评估基于梯度累积的异步 SGD(ASGD-GA)和跨云 PS 间的模型平均(MA)2 种同步策略的效果,研究对比了 baseline、ASGD-GA、异步通信下的 MA(AMA)和同步通信下的 MA(SMA)4 种同步策略下的跨地域训练效果。Baseline 的跨云同步策

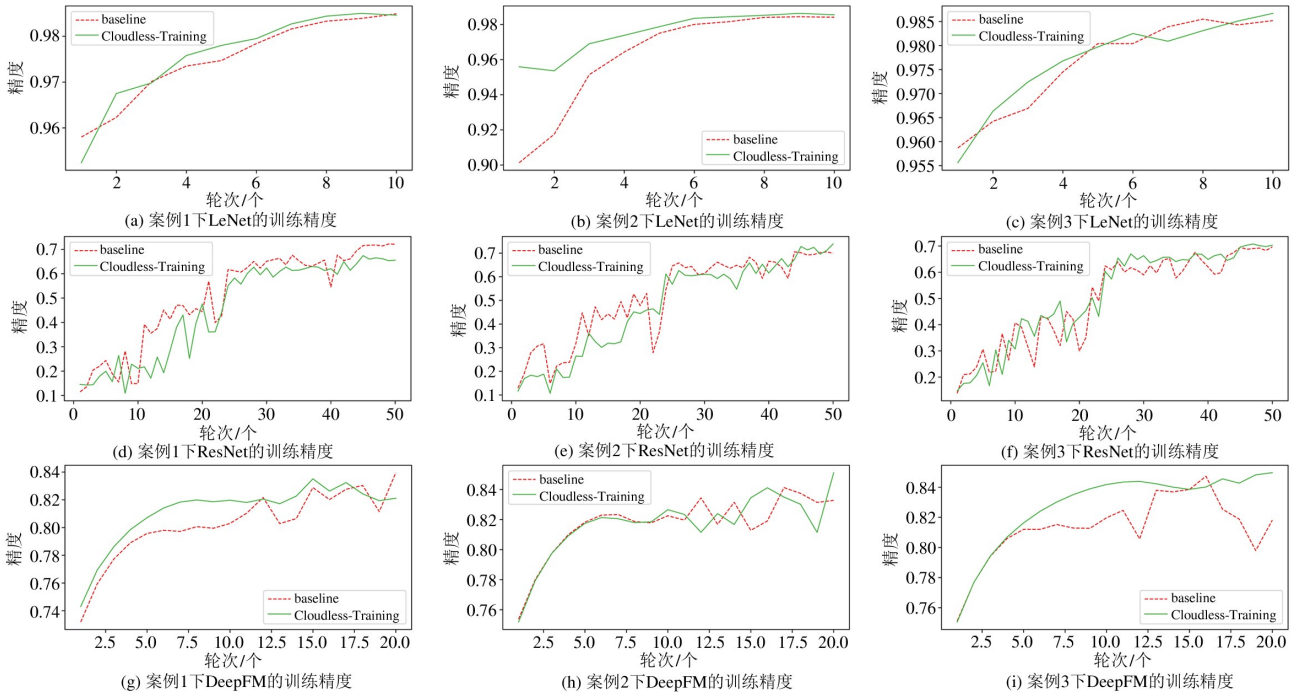

图 9　腾讯云上海和重庆地区不同数据分布(1:1、1:2、2:1)和不同 CPU 核数分配的 3 种情况下,使用和不使用弹性调度的训练精度比较

表 6　使用弹性调度策略后的指标优化程度

指标	LeNet	ResNet	DeepFM
等待时间减少率/%	46.0 ~ 82. 6	82.3 ~ 94.6	6.8 ~ 26.0
训练成本减少率/%	13.8 ~ 16.0	9.2 ~ 15.7	13.4 ~ 24.0

略是简单的异步 SGD 策略,同步频率是 1。ASGD-GA 和 AMA 策略下,同步频率分别为 4 和 8。除 SMA 外,其他的几种策略都部署在上述腾讯云上,由于 SMA 的训练成本过高,它的评估实验在北京和上海的自托管云集群中完成。

实验结果如图 10 和图 11 所示,本文的同步策略可显著提高跨地域分布式训练的性能。

首先,ASGD-GA 和 AMA 同步策略都可有效提升跨地域分布式训练的性能,如图 10(a) ~ (c)所示。LeNet、ResNet 和 DeepFM 的训练速度分别提高了 1.2 倍、1.2 倍和 1.7 倍。训练速度的提升主要收益于通信效率的提高,而通信效率的提高主要来自云间同步频率的增加,从 1 到 4 再到 8,它将总通信量减少了几倍。从表 7 可见,ASGD-GA 和 AMA 的加速效果非常相似,当同步频率为 4 时,使用 ASGD-GA 同步策略训练 LeNet、ResNet 和 DeepFM 的通信时间比 baseline 减少了 48.3%、52.7%和 58.4%,AMA 同步策略的通信时间分别减少了 46.0%、52.7%和 58.5%。当同步频率为 8 时,ASGD-GA 同步策略的通信时间比 baseline 减少了 64.3%、70.1%和72.9%,AMA 同步策略的通信时间分别减少了 60.8%、56.7%和71.0%。由于 WAN 中的网络波动,下降的幅度没有达到理论上预期的 2 倍。

其次,ASGD-GA 和 AMA 同步策略都可以保证跨地域训练的模型收敛精度,如图 10(d) ~ (f)所示。多数情况下,ASGD-GA 和 AMA 的精度收敛趋势与 baseline 相似。周期性的模型同步有时可能会减缓全局同步,但最终结果仍能达到甚至超过 baseline。

最后,如图 11 所示,使用 SMA 同步策略的训练精度高于其他同步策略。在自托管环境中比较 baseline、ASGD-GA、AMA 和 SMA 这 4 种同步策略,使用 SMA 同步策略的训练速度明显慢于 ASGD-GA 和 AMA 的,它的训练用时与 baseline 相似。但其最终精度明显优于其他所有同步策略,因此,该策略可用于提高训练的准确性,值得进一步研究。

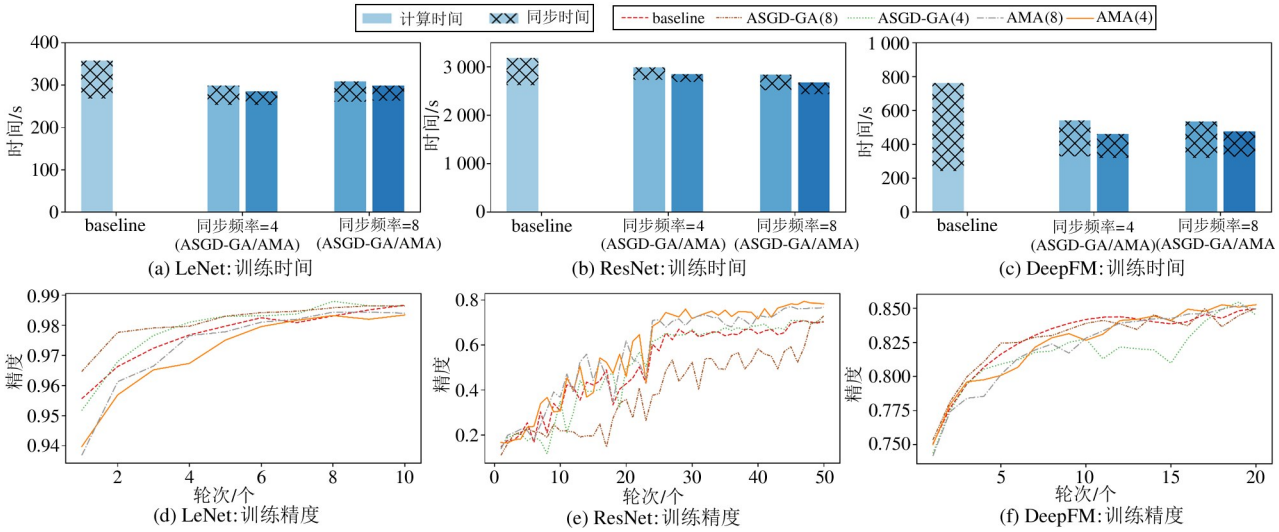

图 10　在腾讯云上海和重庆地区，ResNet 在 3 种模型同步策略（ASGD、ASGD-GA、AMA）下的训练时间和精度收敛比较

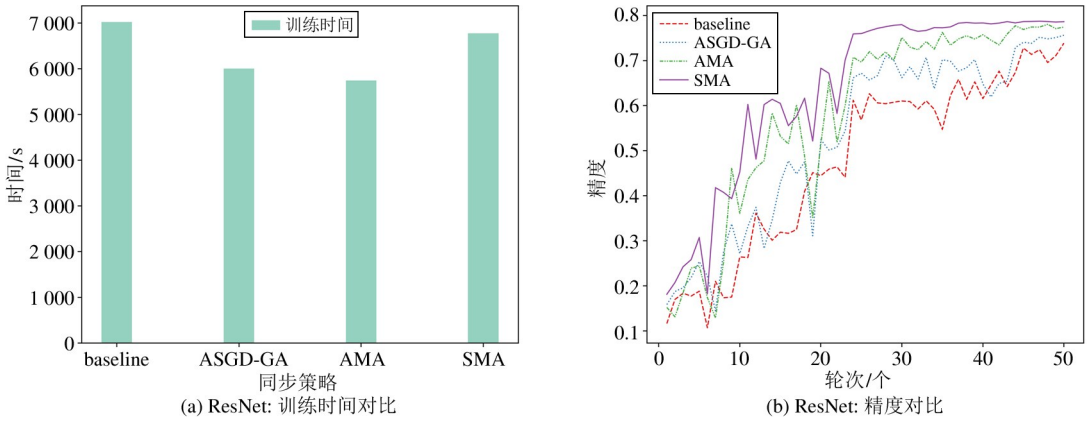

图 11　在自托管环境中，ResNet 在 4 种模型同步策略（ASGD、ASGD-GA、AMA、SMA）下的训练时间和精度收敛比较

表 7　各模型在 2 种同步策略的通信时间下降率

同步策略	LeNet/%	ResNet/%	DeepFM/%
ASGD-GA（同步频率 =4）	48.3	52.7	58.4
AMA（同步频率 =4）	46.0	52.7	58.5
ASGD-GA（同步频率 =8）	64.3	70.1	72.9
AMA（同步频率 =8）	60.8	56.7	71.0

5　结　论

针对跨地域分布式 ML 训练中负载不均和模型同步通信开销大的问题，本文提出了一种基于 serverless 技术的分布式 ML 训练框架：Cloudless-Training。在基础架构方面，采用 2 层架构对训练控制层和训练执行层进行解耦，简化多云区域下训练管理的逻辑视图。在资源调度方面，为了解决多地域间的负载失衡问题，提出了可保障云间负载平衡的资源弹性调度策略，为训练任务制定负载均衡的资源分配方案。在同步优化方面，提出了可设定同步频率的基于梯度累积的异步 SGD（ASGD-GA）和模型平均（MA）2 种模型同步策略，以减少在低带宽 WAN 上的通信开销。实验结果表明，Cloudless-Training 提高了跨地域分布式 ML 训练的资源利用率（例如，降低了 9.2% ~24.0% 的训练成本）和通信性能（例如，训练速度提升了 1.7 倍）。

参考文献

[1] BROWNT, MANN B, RYDERN, et al. Language models are few-shot learners[J]. Advances in Neural Information Processing Systems, 2020,33:1877-1901.

[2] LI L, FAN Y, TSE M, et al. A review of applications in

federated learning[J]. Computers & Industrial Engineering, 2020,149:106854.
[3] ZHANG J, QU Z, CHEN C, et al. Edge learning: the enabling technology for distributed big data analytics in the edge[J]. ACM Computing Surveys, 2022,54(7):1-36.
[4] ZHANG M, KRINTZ C, WOLSKI R. Stoic: Serverless teleoperable hybrid cloud for machine learning applications on edge device[C]//2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). Texas, USA: IEEE, 2020:1-6.
[5] Tenceng. Tencent cloud [EN/OL]. [2023-05-12]. https://cloud.tencent.com/?mobile&lang=en.
[6] CUN Y L, BOSER B, DENKERJ S, et al. Handwritten digit recognition with a back-propagation network[C]// Proceedings of the 2nd International Conference on Neural Information Processing Systems. Cambridge, USA: MIT Press,1989:396-404.
[7] LI M, ZHOU L, YANG Z, et al. Parameter server for distributed machine learning[C]// Big Learning NIPS Workshop. Lake Tahoe, USA: NIPS, 2013:1-10.
[8] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA:IEEE, 2016:770-778.
[9] HSIEH K, HARLAP A, VIJAYKUMAR N, et al. Gaia: geo-distributed machine learning approaching LAN speeds [C]// Proceedings of the 14th USENIX Conference on Networked Systems Design and Implementation. Boston, USA: USENIX Association, 2017:629-647.
[10] XIANG Y, WU Z, GONG W, et al. Nebula-I:a general framework for collaboratively training deep learning models on low-bandwidth cloud clusters[EB/OL]. (2022-05-19)[2023-05-12]. https://arxiv.org/pdf/2205.09470.pdf.
[11] MCGRATH G, BRENNER P R. Serverless computing: design, implementation, and performance[C]// 2017 IEEE 37th International Conference on Distributed Computing Systems Workshops (ICDCSW). Atlanta, USA: IEEE, 2017:405-410.
[12] JORDAN M I, MITCHELL T M. Machine learning: trends, perspectives, and prospects[J]. Science, 2015, 349(6245):255-260.
[13] SZE V, CHEN Y J, YANG H T, et al. Efficient processing of deep neural networks: a tutorial and survey[J]. Proceedings of the IEEE, 2017,105(12):2295-2329.
[14] SHOEYBI M, PATWARY M, PURI R, et al. Megatron-lm:training multi-billion parameter language models using model parallelism [EB/OL]. (2019-09-17)[2023-05-12]. https://arxiv.org/pdf/1909.08053.pdf.
[15] HIND B, RAKRAK S, RAGHAY S, et al. Moving to the edge-cloud-of-things: recent advances and future research directions[J]. Electronics, 2018,7(11):309.
[16] KHAN N, YAQOOB I, HASHEM I A T, et al. Big data: survey, technologies, opportunities, and challenges[J]. The Scientific World Journal, 2014,2014:1-19.
[17] MISHRA S, TYAGI A K. The role of machine learning techniques in Internet of things-based cloud applications [J]. Artificial Intelligence-based Internet of Things Systems, 2022, 2022:105-135.
[18] ZHANG H, ZHENG Z, XU S, et al. Poseidon:an efficient communication architecture for distributed deep learning on GPU clusters[C]// USENIX Annual Technical Conference. Santa Clara, USA: USENIX Association, 2017:181-193.
[19] SHAZEER N, CHENG Y, PARMAR N, et al. Mesh-tensorflow:deep learning for supercomputers[C]// Proceedings of the 32nd International Conference on Neural Information Processing Systems. Montréal, Canada: Curran Associates Inc., 2018:10435-10444.
[20] NARAYANAN D, PHANISHAYEE A, SHI K, et al. Memory-efficient pipeline-parallel DNN training[C]// International Conference on Machine Learning. Online: ICML, 2021:7937-7947.
[21] KUBELESS. Kubeless: the Kubernetes native serverless framework[EB/OL]. [2023-05-12]. https://kubeless.io.
[22] OPENWHISK. Apache OpenWhisk[EB/OL]. [2023-05-12]. http://openwhisk.org/.
[23] Github. OpenFaaS-serverless functions made simple[EB/OL]. [2023-05-12]. https://github.com/openfaas/faas.
[24] Amazon. Lambda[EB/OL]. [2023-05-12]. https://aws.amazon.com/lambda/.
[25] CARREIRA J, FONSECA P, TUMANOV A, et al. Cirrus: a serverless framework for end-to-end ML workflows [C]// Proceedings of the ACM Symposium on Cloud Computing. Santa Cruz,USA: ACM, 2019:13-24.
[26] JIANG J, GAN S, LIU Y, et al. Towards demystifying serverless machine learning training[C]// Proceedings of the 2021 International Conference on Management of Data. Online: Association for Computing Machinery, 2021: 857-871.
[27] CUI H, ZHANG H, GANGER G R, et al. GeePS: scalable deep learning on distributed GPUs with a GPU-specialized parameter server[C]// Proceedings of the 11th European Conference on Computer Systems. London, UK: ACM, 2016:1-16.
[28] Amazon. Docs[EB/OL]. [2023-05-12]. https://docs.aws.amazon.com/zh_cn/AWSEC2/latest/UserGuide/ec2-instance-network-bandwidth.html.
[29] LIN Y, HAN S, MAO H, et al. Deep gradient compres-

sion: reducing the communication bandwidth for distributed training[EB/OL]. (2017-12-05)[2023-05-12]. https://arxiv.org/pdf/1712.01887.pdf.
[30] FIKRIA A, HEAFIELD K. Sparse communication for distributed gradient descent [EB/OL]. (2017-04-17)[2023-05-12]. https://arxiv.org/pdf/1704.05021.pdf.
[31] RUDER S. An overview of gradient descent optimization algorithms[EB/OL]. (2016-09-15)[2023-05-12]. https://arxiv.org/pdf/1609.04747.pdf.
[32] YU H, YANG S, ZHU S. Parallel restarted SGD with faster convergence and less communication: demystifying why model averaging works for deep learning[C]// Proceedings of the AAAI Conference on Artificial Intelligence. Hawaii, USA: AAAI Press, 2019:5693-5700.
[33] TAN W T, SHI X, LEI Z Y, et al. TrainFlow: a lightweight, programmable ML training framework via serverless paradigm[C]// Network and Parallel Computing: 19th Annual IFIP International Conference. Jinan, China: IFIP, 2022:155-167.
[34] SHI S, ZHOU X, SONG S, et al. Towards scalable distributed training of deep learning on public cloud clusters [J]. Proceedings of Machine Learning and Systems, 2021,3:401-412.
[35] ALEX K, ANDHINTON G. Learning multiple layers of features from tiny images [R]. Toronto :University of Toronto, 2009.
[36] Github. Kubernetes-native deep learning framework [EB/OL]. [2023-05-12]. https://github.com/sql-machine-learning/elasticdl.
[37] GUO H, TANG R, YE Y, et al. DeepFM: a factorization-machine based neural network for CTR prediction [EB/OL]. (2017-03-13)[2023-05-12]. https://arxiv.org/pdf/1703.04247.pdf.

Cloudless-Training: a framework to improve efficiency of geo-distributed ML training based on serverless

TAN Wenting[*,**], LV Cunchi[*,**], SHI Xiao[*,***], ZHAO Xiaofang[*,****]
([*] Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)
([**] University of Chinese Academy of Sciences, Beijing 100049)
([***] Nanjing Institute of Information SuperBahn, Nanjing 211135)
([****] Institute of Intelligent Computing Technology, Suzhou, Chinese Academy of Sciences, Suzhou 215028)

Abstract

Geo-distributed machine learning (ML) training can benefit many emerging ML scenarios (e.g., large model training, federated learning) with multi-regional cloud resources and wide area network. However, its efficiency is limited due to two challenges. First, efficient elastic scheduling of multi-regional cloud resources is usually missing, affecting resource utilization and performance of training. Second, training communication on wide area network (WAN) is still the main overhead, easily subjected to low bandwidth and high fluctuations of WAN. In this paper, a framework Cloudless-Training is proposed to realize efficient geo-distributed ML training in 3 aspects. First, it uses a two-layer architecture with control and physical training planes to support elastic scheduling and communication for multi-regional clouds in a serverless manner. Second, it provides an elastic scheduling strategy that can deploy training workflows adaptively according to the heterogeneity of available cloud resources and distribution of pre-existing training datasets. Third, it provides two new synchronization strategies for training partitions among clouds, including asynchronous stochastic gradient descent with gradient accumulation (ASGD-GA) and inter-parameter server (PS) model averaging (MA). It is implemented with OpenFaaS and evaluated on Tencent Cloud. Experimental results show that Cloudless-Training can support general ML training in a geo-distributed way, and greatly improve resource utilization (e.g., 9.2% −24.0% training cost reduction) and synchronization efficiency (e.g., 1.7 times speedup of training over baseline at most) with model correctness guarantees.

Key words: geo-distributed machine learning(ML) training, cross cloud ML training, distributed training framework, serverless, cross cloud model synchronization