阿里云 容器服务

深度学习解决方案

文档版本:20180808



法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读 或使用本文档,您的阅读或使用行为将被视为对本声明全部内容的认可。

- 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档,且仅能用于自身的合法 合规的业务活动。本文档的内容视为阿里云的保密信息,您应当严格遵守保密义务;未经阿里云 事先书面同意,您不得向任何第三方披露本手册内容或提供给任何第三方使用。
- 未经阿里云事先书面许可,任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分 或全部,不得以任何方式或途径进行传播和宣传。
- 由于产品版本升级、调整或其他原因,本文档内容有可能变更。阿里云保留在没有任何通知或者 提示下对本文档的内容进行修改的权利,并在阿里云授权通道中不时发布更新后的用户文档。您 应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
- 4. 本文档仅作为用户使用阿里云产品及服务的参考性指引,阿里云以产品及服务的"现状"、"有缺陷"和"当前功能"的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引,但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的,阿里云不承担任何法律责任。在任何情况下,阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害,包括用户使用或信赖本文档而遭受的利润损失,承担责任(即使阿里云已被告知该等损失的可能性)。
- 5. 阿里云网站上所有内容,包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站 画面的安排、网页设计,均由阿里云和/或其关联公司依法拥有其知识产权,包括但不限于商标 权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意,任何人不得擅自使 用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此 外,未经阿里云事先书面同意,任何人不得为了任何营销、广告、促销或其他目的使用、公布或 复制阿里云的名称(包括但不限于单独为或以组合形式包含"阿里云"、Aliyun"、"万网"等阿里云 和/或其关联公司品牌,上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或 服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联 公司)。
- 6. 如若发现本文档存在任何错误,请与阿里云取得直接联系。

通用约定

格式	说明	样例
•	该类警示信息将导致系统重大变更甚至 故障,或者导致人身伤害等结果。	禁止: 重置操作将丢失用户配置数据。
A	该类警示信息可能导致系统重大变更甚 至故障,或者导致人身伤害等结果。	▲ 警告: 重启操作将导致业务中断,恢复业务所需 时间约10分钟。
	用于补充说明、最佳实践、窍门等,不是用户必须了解的内容。	送 说明: 您也可以通过按 Ctrl + A 选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 确定。
courier 字体	命令。	执行 cd /d C:/windows 命令,进 入Windows系统文件夹。
斜体	表示参数、变量。	bae log listinstanceid Instance_ID
[]或者[a b]	表示可选项,至多选择一个。	ipconfig[-all/-t]
{}或者{a b}	表示必选项,至多选择一个。	<pre>swich {stand slave}</pre>

目录

法律声明	I
通用约定	I
1 Swarm 集群	1
1.1 概述	1
1.2 环境准备	1
1.2.1 创建数据卷	1
1.2.2 创建容器集群	5
1.3 模型开发	9
1.3.1 创建 Jupyter 环境	9
1.3.2 在 Jupyter 环境使用 Git 管理代码	12
1.3.3 通过 SSH 访问 Jupyter 服务	13
1.3.4 通过端口转发方式访问 Jupyter 服务	17
1.4 模型训练	
1.4.1 单机模型训练	24
1.4.2 分布式模型训练	
1.5 预测服务	
1.5.1 使用 TensorFlow Serving	36
1.6 在GPU 集群上实现Caffe模型训练	41
1.6.1 环境准备	41
1.6.2 构建和推送自定义 caffe 镜像	44
1.7 利用 TFRecord 和 HDFS 准备 TensorFlow 训练数据	47

1 Swarm 集群

1.1 概述

基于阿里云强大计算能力的深度学习解决方案,为您提供一个低门槛、开放、端到端的深度学习服 务平台。方便数据科学家和算法工程师快速开始利用阿里云的资源(包括 ECS 云服务器、GPU 云 服务器、高性能计算 HPC、对象存储 OSS、Elastic MapReduce、负载均衡等服务)执行数据准 备、模型开发、模型训练、评估和预测等任务。并能够方便地将深度学习能力转化为服务 API,加 速与业务应用的集成。

具体而言,该深度学习解决方案具备以下特性:

- 简单:降低构建和管理深度学习平台的门槛。
- 高效:提升 CPU、GPU 等异构计算资源的使用效率,提供统一的用户体验。
- 开放:支持 TensorFlow、Keras、MXNet 等多种主流深度学习框架,同时用户也可使用定制的环境。
- 全周期:提供基于阿里云强大服务体系构建端到端深度学习任务流程的最佳实践。
- 服务化:支持深度学习能力服务化,与云上应用的轻松集成。

开始使用

1. 环境准备。

创建容器集群。如果您需要使用 OSS 数据卷存储数据,您还需要 创建 OSSFS 数据卷。

- 2. 创建 Jupyter 环境,并在 Jupyter 环境使用 Git 管理代码。
- 3. 运行 单机模型训练 或者 分布式模型训练,导出模型。
- 4. 利用导出的模型,执行使用 TensorFlow Serving。

1.2 环境准备

1.2.1 创建数据卷

OSSFS 是阿里云官方提供的基于 FUSE 的文件系统 (项目主页见 https://github.com/aliyun/ossfs

)。OSSFS 数据卷可以将 OSS 的 Bucket 包装成数据卷。

由于数据需要经过网络同步到云端,OSSFS 在性能和功能上与本地文件系统有差距。请不要把数 据库等重 IO 应用、日志等需要不断改写文件的应用运行在 OSSFS 上。OSSFS 比较适合多容器之 间共享配置文件,或者附件上传等没有改写操作的场景。 OSSFS 和本地文件系统具体差异如下所示:

- 随机或者追加写文件会导致整个文件的重写。
- 因为需要远程访问 OSS 服务器,元数据操作(例如 list directory)性能较差。
- 文件/文件夹的重命名操作不是原子的。
- 多个客户端挂载同一个 OSS Bucket 时,需要您自行协调各个客户端的行为。例如,避免多个客户端写同一个文件等等。
- 不支持硬链接(hard link)。

前提条件

您的集群必须满足以下两个条件,才可以开通数据卷功能:

• 集群 Agent 的版本为 0.6 或更高。

您可以在集群列表页面查看您的 Agent 的版本。选择所需的集群,单击右侧的 更多 > 升 级Agent。



如果您的 Agent 版本过低,请先升级您的 Agent。有关如何升级 Agent,参见 升级 Agent。

集群里部署了 acsvolumedriver 应用。建议您将 acsvolumedriver 升级到最新版本。

您可以通过升级系统服务部署和升级 acsvolumedriver 应用。详细操作参见 升级系统服务。



升级或重启 acsvolumedriver 时,使用 OSSFS 数据卷的容器会重启,您的服务也会重启。

操作步骤

步骤 1 创建 OSS Bucket

登录 对象存储管理控制台,创建一个 Bucket,参见 创建存储空间。

本示例创建了一个位于华南1地域的 Bucket。

tensorflow-samples 区域: 华南 1 创建时间: 2017-04-20 14:42 删除						
☞ 概览 団 Object管理	⑧ 基础设置 域名管理 图片处理 事件通知	▶ 基础数据 热点统计 API统计 Obj	ect统计			
基础数据						
存储用量	本月流量 流入 流出 CDN回源	本月请求次数 PUT GET	Object数量	文件碎片		
0 в	0 в	0	0	0		
月同比 日环比	上月流量总量 0.00B	上月请求次数总量 0				
访问域名						
VPC内网域名: tensorflow-	and a state of the state of processing	外网域名: tensorflow				
经典网络内网域名: tensorflow-						

步骤 2 创建 OSSFS 数据卷

- 1. 登录 容器服务管理控制台。
- 2. 单击左侧导航栏中的数据卷。
- 3. 选择要创建数据卷的集群 (本示例中为 tfoss) 并单击页面右上角的 创建。

容器服务		数据卷列表				刷新创建		
Kubernetes	Swarm							
概览		常见问题:						
服务		 市点 	卷名	驱动	挂载点	引用容器	卷参数	操作
集群		ce33a1ac76fbe41488341891	b724636e4a04a7392c307bfb	本地磁盘	/var/lib/docker/volumes/			删除所有同名卷
节点		ce33a1ac76fbe41488341891	e7f7d735c1970272080eeca1	本地磁盘	/var/lib/docker/volumes/			删除所有同名卷
网络		ce33a1ac76fbe41488341891	e95498f84d31ac57cce8011e	本地磁盘	/var/lib/docker/volumes/			删除所有同名卷
数据卷	1	出生肥除						
配置项								

在弹出的对话框中,选择数据卷类型为OSS,设置数据卷参数并单击创建。容器服务会在集群的所有节点上创建名称相同的数据卷。

创建数据卷		\times
数据卷类型:	● OSS ◎ 云盘	
数据卷名:		
AccessKey ID :		
AccessKey Secret :		
可选参数:	🗌 allow_other 🕢 🔲 noxattr 🖉	
更多参数:	更多参数的填写格式可以参考该文档 注意:只有volume driver在0.7版本及以上的集群 才支持该参数,您可以到"应用"列表中找到 acsvolumedriver应用,然后在其详情页的服务列 表中查看volumedriver服务的镜像版本,如果是 0.7以下的话,请升级系统服务	
Bucket ID :	选择Bucket	
访问域名:	🔍 内网域名 🔍 外网域名 🔍 vpc域名 🖉	
文件缓存:	◎ 打开 💿 关闭 🛛 🕢	
		创建取消

- 数据卷名:数据卷的名称。数据卷名在集群内必须唯一。
- AccessKey ID、AccessKey Secret:访问 OSS 所需的 AccessKey。您可以从 阿里云账号 AccessKey 控制台 获取。
- Bucket ID: 您要使用的 OSS bucket 的名称。单击 选择Bucket,在弹出的对话框中选择所 需的 bucket (本示例中为 tensorflow-samples)并单击 选择。
- 访问域名:选择 vpc域名。
- 文件缓存:如果需要在不同机器间同步同一个文件的修改(比如在机器 A 中修改文件,在机器 B 中读取修改后的内容),请关闭文件缓存。

📃 说明:

关闭文件缓存将导致 ls 文件夹变得很缓慢,尤其是同一个文件夹下文件比较多时。因此,没 有上述需求时,请打开文件缓存,提高 ls 的速度。

后续操作

创建数据卷之后,您可以在您的应用中使用创建的数据卷。有关如何在应用中使用数据卷,参见使用第三方数据卷。

1.2.2 创建容器集群

深度学习解决方案支持使用云服务器 ECS 容器集群或者 GPU 服务器容器集群。本文档以 GPU 服务器容器集群为例进行说明。



有关如何创建 ECS 容器集群,参见 创建集群。

使用限制

- 目前,容器服务仅支持在华南1、华东2、华北2和美西1地域创建 GN4型 GPU 云服务器集群。
- 目前, GN4型 GPU 云服务器只支持专有网络(VPC)。

提前准备

目前,按量付费的 GPU 计算型 GN4 云服务器需要申请工单开通。如果您需要使用按量付费的 GPU 计算型 GN4 云服务器,请按照如下内容 提交 ECS 工单。

我需要申请按量付费的GPU计算型gn4,请帮忙开通,谢谢。

操作流程

- 1. 登录 容器服务管理控制台。
- 2. 在 Swarm 菜单下,单击左侧导航中的 集群,单击右上角的 创建Swarm集群。

容器服务		集群列表		您最多可以创建 5 个集群,每1	个集群最多可以添加 20 个节点 刷新	创建Swarm集群 🗸
Swarm	Kubernetes					
概览		常见问题: Ø 如何创建集群	Ø如何添加已有云服务器 Ø跨可用区节点管理 Ø	♀集成日志服务		
应用		名称 ▼				
服务		集群名称/ID	集群英型 地域 网络类型	集群状态 节点状态 🚱 节点个数 创建时	间 Docker版本	操作
集群 节点	1					

3. 设置集群的基本信息。本示例中创建一个位于华南1地域名为 EGS-cluster 的集群。

* 集群名称:	EGS-cluster					
	名称为1-63个字符,	可包含数字、汉字、	英文字符,或"-"			
地域:	华北1(青岛)	华北 2 (北京)	华东1(杭州)	华东 2 (上海)	华南1(深圳)	亚太东北1(东京)
				欧洲中部 1 (法兰克	美国东部1(弗吉尼	
	美国西部1(硅谷)	亚太东南 1 (新加坡)	亚太东南 2 (悉尼)	福)	亚)	香港
	华北3(张家口)	亚太东南 3 (吉隆坡)				
可用区:	华南 1 可用区 B	•				

 集群名称:要创建的集群的名称。可以包含 1~64 个字符,包括数字、中文字符、英文字符 和连字符(-)。

☐ 说明:	
集群名称在同一个用户和同一个地域下必须唯一。	
地域:所创建集群将要部署到的地域。选择华南1、华东2、华北2或美国西部1。	
〕 说明 :	

目前, 仅支持在华南1、华东2、华北2和美国西部1地域创建 GN4型 GPU 云服务器集群。

• 可用区:集群的可用区。

门 说明:

您可以根据您的服务器分布情况,选择不同的地域和可用区。

4. 设置集群的网络类型为 专有网络 并配置相关信息。

网络类型:	经典网络	专有网络		
	vpc-bp1anu8b	or4av7beb2w 🔻	defaultyswitch	•
	the oblighted	/ 10// 002 M	ucidultyswitch	
容器起始网段:	172.18.0.0/24	4	查看已有网段 ②	

专有网络 VPC 支持您基于阿里云构建一个隔离的网络环境,您可以完全掌控自己的虚拟网络,包括自由 IP 地址范围、划分网段、配置路由表和网关等。

专有网络需要您指定一个 VPC、一个 VSwitchld 和容器的起始网段(Docker 容器所属的子网 网段,为了便于 IP 管理,每个虚拟机的容器属于不同网段,容器子网网段不能和虚拟机网段冲 突)。

为了防止网络冲突等问题,建议您为容器集群建立属于自己的 VPC/VSwitchld。

5. 添加节点。

是否新增节点: 创建节点 添加已有节点

您可以在创建集群的同时创建若干个节点,或者创建一个零节点集群并添加已有云服务器。有关 如何添加已有云服务器的详细信息,参见添加已有云服务器。

- 创建节点
 - 1. 设置节点的操作系统。

|--|

目前支持的操作系统包括 Ubuntu 14.04 64 位和 CentOS 7.4 64 位。

- 2. 设置云服务器的实例规格。
 - 实例系列选择系列III。
 - 实例规格选择 32核 48GB(ecs.gn4.8xlarge)或 56核 96GB(ecs.gn4.14xlarge)。



如果您已经通过了 GN4 型 GPU 云服务器的使用申请,但是未找到这两种实例规格,说明目前这两种规格的实例没有资源,建议晚些时候或者次日再尝试购买。

案列 II 案列 III ⑦
系列 II 采用 Intel Haswell CPU、DDR4 内存,拥有更好的内存计算能力;默认为 I/O 优化实例,搭配 SSD 云盘可获得更好的存储性能。
1/0 优化实例
373#2 48/GR (ers nn4 8vlame) *
U 5台 10台 20台 2 台 ≑
每个集群可以创建20台 云服务器
高效云盘 SSD 云盘
SSD 云盘 高效云盘 普通云盘
挂載数輝盘
· ⑦
8 - 30 个字符,且同时包含三项(大、小写字母,数字和特殊符号)

您可选择实例的数量,并指定数据盘的容量(云服务器默认带有 20G 大小的系统盘)和登录密码。

📃 说明:

- 如果您选择了数据盘,它会被挂载到 /var/lib/docker 目录,用于 Docker 镜像和 容器的存储。
- 从性能和管理考虑,建议您在宿主机挂载独立的数据盘,并利用 Docker 的 volume 对 容器的持久化数据进行管理。

• 添加已有节点

您可以单击下边的选择已有实例将已有的云服务器添加到集群中,或者直接单击创建集群 等集群创建完成后再通过集群列表页面添加已有云服务器。

6. 配置 EIP。

当您将网络类型设置为 VPC 时,容器服务会默认给每一个专有网络下的云服务器配置一个 *EIP*。如果不需要,您可以勾选 不配置公网EIP 复选框,但是需要额外配置 SNAT 网关。

```
EIP: 不配置公网EIP
不配置公网EIP,可以使用阿里云提供的 NAT网关产品 实现VPC安全访问公网环境,您也可以自行配置SNAT(请参考以下文档)。未配置SNAT会导致VPC不能正常访问公网,会影响集群创建和应用部署等。请参考:
VPC网络环境下Linux系统配置SNAT实现无公网ECS通过有EIP的服务器代理上网
```

7. 创建一个负载均衡实例。

负载均衡:	🔽 自动创建负载均衡
	创建集群会默认创建一个公网负载均衡实例,计费类型为 <mark>按量付费</mark>

目前创建集群会默认创建一个负载均衡实例。您可以通过这个负载均衡实例访问集群内的容器应 用。所创建的负载均衡实例为按量付费实例。

8. 单击 创建集群。

后续操作

您可以查看集群创建日志。在集群列表页面,选择所创建的集群并单击查看日志。

集群列表									您最多	可以创建 5	个集群,每个集群最多可以	以添加 20 个节点	子账号授权	刷新	创建集群
小助手:如	口何创建集群	如何添加已有云	服务器	跨可用区节	京管理	集成日志服务	通过	Docker客户望	连接集群						
名称 ▼															
集群名称/ID		纬	朝鮮美型	地域	网络类型	핀		集群状态	节点状态 🕜	节点个数	创建时间	Docker版本			操作
EGS-cluster c78210ff0b		ß	可里云集群	华南1	虚拟专有 Vpc·	有网络		● 就绪	健康 ℃	1	2017-04-19 13:23:50	17.03.1-ce	管理	查看日起 监	た

1.3 模型开发

1.3.1 创建 Jupyter 环境

准备工作

在运行模型训练任务之前,请确认以下工作已经完成:

- 创建包含适当数量弹性计算资源(ECS或 GPU 服务器)的容器集群。参见创建容器集群。
- 如果您需要使用对象存储服务(OSS)保存用于模型训练的数据,您需要使用相同账号创建
 OSS Bucket;然后在上面的容器集群中创建数据卷,用于将 OSS Bucket 作为本地目录挂载到
 执行训练任务的容器内。参见创建数据卷。

约定

为了方便您的应用代码读取训练数据,输出训练日志,训练卷中的数据会存放在 / input 目录,用 户代码需要从该目录中读取数据。

操作流程

- 1. 登录 容器服务管理控制台。
- 2. 在 Swarm 菜单下,单击左侧导航栏中的 镜像与方案 > 解决方案。
- 3. 在模型开发框中单击创建。

容器服务		机器学习		
Swarm	Kubernetes			
概览		▶ 模型开发		良力 模型预测
应用		使用 Jupyter 快速开发、调试模型代码,并使用	使用CPU, GPU集群训练模型。支持TensorFlow,	使用CPU, GPU进行模型预测。支持TensorFlow
服务		TensorBoard 监控训练过程。支持TensorFlow, Keras等。	Keras等,及分布式训练方法。支持TensorBoard 监控训练过程。	Serving服务端。支持服务的负载均衡和弹性伸缩。
集群		创建 指南	创建 指南 历史记录	创建 指南
节点		区块链 3		
网络				
数据卷		Hyperledger Fabric		
配置项		支持 Hyperledger Fabric v1.0 的区块链网络自动		
1 镜像与方言	2 1	化配置和部署		
镜像		创建 指南		
编排模板	2 1			
解决方案	2			

- 4. 设置创建 Jupyter 环境的基本信息。
 - 集群:所创建模型开发应用将要部署到的集群。本示例中为 EGS-cluster。
 - 应用名:所创建应用的名称。名称可以包含 1~64 个字符,包括数字、英文字符和连字符(-),且不能以-开头。
 - 训练框架:所支持的训练框架包括 TensorFlow, Keras 以及不同 Python 版本。
 - GPU数量:所使用的 GPU 数量,如果为 0 表示不使用 GPU。
 - 数据卷名:指定为用于存储训练数据的数据来源,可以选择对象存储服务在该集群中创建的数据卷的名称;也可以选择本地目录,但要求填写绝对路径;或者选择不使用数据卷。本示例中使用名为 tfoss 的数据卷。
 - Jupyter密码:登录 Jupyter 所用的密码。
 - 训练监控:是否使用 TensorBoard 监控训练状态;一旦选择监控,请指定训练日志的路径,并保证与训练代码中日志输出的路径一致。
 - · 启用SSH登录:选择是否启用 SSH 方法访问服务。勾选此选项后,您需要填写您的 SSH密码。

有关如何通过 SSH 方法访问服务,参见 通过 SSH 访问 Jupyter 服务。

株型工作	
候坐开友	
集群	EGS-cluster 🔻
应用名	mydevbox
训练框架	tensorflow:1.0.0
GPU数量	0
数据卷名	tfoss 🔻
Jupyter密码	******
	☑ 训练监控
训练日志路径	/output/training_logs
	明味证与则\$15时中口志利口的增化——3
	☑ 启用SSH登录
CC1 1582277	
2245619	

- 5. 设置完毕后,单击确定。
- 6. 在应用列表页面,选择创建的应用,单击应用名称进去。

集群: EGS-cli	uster	▼ 🗷 隐藏系统应用 🔲 隐藏	离线应用 🔲 隐藏	在线应用		名称 ▼
应用名称	描述	状态	容器状态	创建时间 🔺	更新时间 ▲	操作
mydevbox		● 就绪	就绪:0 停止:0	2017-04-19 14:01:31	2017-04-19 14:01:31	亭止 変更配置 删除 重新部署 事件

7. 单击 路由列表,可以看到两个链接,分别是以 jupyter 和 tensorboard 开头的链接。

服务列表	容器列表	路由列表	日志	事件			
路由地址(集群绑定SLB后路由地址才能被访问)							
tensorboard-							
jupyter-:				-		设置服务权重	

- 8. 单击 jupyter 开头的链接,并且输入 jupyter 的密码,就能进入 jupyter 环境。
- 9. 单击 tensorboard 开头的链接,查看训练结果。

TensorBoard	SCALARS IMAGES AUDIO GRAPHS DISTRIBU	TION
Write a regex to create a tag group × Split on underscores Data download links Tooltip sorting method: default • Smoothing 0.6	accuracy accuracy 0.900 0.900 0.700 0.600 0.600 0.000 4.000k 8.000k 12.00k C3	
Horizontal Axis STEP RELATIVE WALL	loss	
Runs Write a regex to filter runs	2.20 1.80 1.40	

→ C ③ tensorboard-20170405155352.c28cec27bad6c403ba7fb5fb045321592.cn-shenzhen.alice

10.分布式存储中的训练数据都存储在本地的 /input 文件夹下,您可以从 /input 下读取数据。

1.3.2 在 Jupyter 环境使用 Git 管理代码

操作步骤

1. 在 Jupyter 主页面创建 Terminal。

💭 Jupyter	Logout
Files Running Clusters	
Select items to perform actions on them.	Upload New - 2
• •	Text File
Note	book list empty.
	Notebooks
	Python 3

2. 在 Terminal 内执行利用 git clone 下载应用代码。

```
git clone https://{id}:{password}@github.com/{id}/test.git
Cloning into 'test'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

```
Checking connectivity... done.
```

其中, {id} 为 GitHub 用户名, {password} 为 GitHub 的密码。

- **3.** 切回到 Jupyter 主页面,可以看到应用代码已经在列,这样就可以利用 Jupyter 开发相应的代码。
- 4. 可以回到 Terminal,利用 Git 提交代码。

💭 jupyter

```
test
            config
t confi
                                 global user.email "eml@example.com"
-global user.name "eml"
                 8d272d3] commit
changed, 0 insertions(+), 0 deletions(-)
mode 100644 test2
          : push
ing: push.default is unset; its implicit value is changing in
2.0 from 'matching' to 'simple'. To squelch this message
maintain the current behavior after the default changes, use:
     git config --global push.default matching
To squelch this message and adopt the new behavior now, use:
    git config --global push.default simple
When push.default is set to 'matching', git will push local branches to the remote branches that already exist with the same name.
In Git 2.0, Git will default to the more conservative 'simple'
behavior, which only pushes the current branch to the corresponding
remote branch that 'git pull' uses to update the current branch.
                                          g' and search for 'push.default' for further information.
was introduced in Git 1.7.11. Use the similar mode
of 'simple' if you sometimes use older versions of Git)
           git help config'
                       instead of
   ounting objects: 3, done.
elta compression using up to 4 threads.
ompressing objects: 100% (2/2), done.
riting objects: 100% (2/2), 257 bytes |
otal 2 (delta 0), reused 0 (delta 0)
o https://cheyang:tang3zang@github.com/
lcb83db..8d272d3 master -> master
                                                                                es | 0 bytes/s, done.
                                                                                      /cheyang/test.git
```

1.3.3 通过 SSH 访问 Jupyter 服务

如果您在 创建 Jupyter 环境 环境时选择了 启用SSH登录,您可以通过本文档中介绍的方法 SSH 访问您的 Jupyter 服务。

本示例中, SSH 访问的端口映射为 192.***.*.32775->22/tcp。其中, 192.***.* 是 ECS 实例的 私网 IP, 您无法通过此 IP 访问您的 Jupyter 服务。您需要通过 ECS 实例的弹性公网 IP 从外部 SSH 访问 Jupyter 服务。

刷新

| 应用:mydevbox

基本信息												
应用名称:	Z用名称: mydevbox			创建时间: 20	017-05-16	更新时间	1 : 2017-05-16		所在集群: EGS			
触发器 1.	触发器 1.每件类型的缺发器只能创建1个◎											
目前没有任何能	前没有任何触发器,点击右上角按钮创建触发器											
服务列表	容器列表	路由列表	日志	新 件								
名称/ID		状态	健康检测	镜像		端口		容器IP	节点IP			操作
mydevbox_ju 718623ea4de	pyter () 94f4a	running	正常	registry-vp sha256:21	c.cn 3c1c3ad	6006/tcp 192. :32775->22/tc 192. :32774->8888	p /tcp	172.	192.	删除 停止	监控 日志	远程终端
mydevbox_te 5846e35b8d8	nsorb () 31f82e	running	正常	registry-vp sha256:2b	c.cn 4c70877	192.168		172.	192.	删除 停止	监控 日志	远程终端

您可以登录到 ECS 管理控制台 查看该 ECS 实例绑定的弹性公网 IP。本示例中,弹性公网 IP 为 39.***.**.236。

说明 : 如果 ECS 实例没有	有绑定引	単性公网	IP,您需	需要先	为其绑定	至弹性公网 IP,参	见 管理弹性	公网	IP.	
实例名称 ▼ 输入实例名称模糊查询		搜索	》 标签					高级搜索	<u>₹</u> ¢ ?	
□ 实例ID/名称	监控 所在可	可用区 IP地址	: ¥	犬态(全部) 👻	网络类型(全部) 🔻	配置	付费方式(全部) ▼		撮作	
☐ i-wz9eqt798e c7e8e1064cd	▶ 华南 1	1 可用区 B 39. 192.	.236(弹性) (私有)	」运行中	专有网络	CPU: 56核 内存:96 GB (I/O优化) GPU:NVIDIA M40*2 100Mbps(峰值)	按量 17-05-09 09:42 创建	1	管理 远程连接 升降配 更多▼	

此外,为了通过 ECS 实例的弹性公网 IP 从外部 SSH 访问您的 Jupyter 服务,您还需要配置安全 组规则开放 32775 端口。

配置安全组规则

- 1. 登录 ECS 管理控制台。
- 单击左侧导航栏中的 实例,选择地域(本示例中为 华南 1),选择容器服务应用 mydevbox 对 应的 ECS 实例。

云服务器 ECS	实例列表 华南1 亚太东南1 (新加坡) 华北	1 华北 2 华北 3 华东 2 美国东部 1 (弗吉尼亚	香港 中东东部1(迪拜	2) 亚太东南 2 (悉尼) 华东 1		2 创建实例
概览	1007年前11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日)11月1日(11月1日)11月1日(11月1日)11月1日)11月1日)11月1日)11月1日)11月1日)11月1日)11月1日)11月1日)11月1日)11月1日)11月1日)11月1日)	乐京) 美国四部 1 (住台)				
实例	实例名称 ▼ 輸入实例名称模糊查询	搜索				高级搜索 💆 🌣 ?
 存储 	□ 实例ID/名称 监控	所在可用区 IP地址 3 志(全)	部) ▼ 网络类型(全部) ▼	配置	付盡方式(全部) ▼	操作
 快照和鏡像 	i-wz9ec	() 39236(译性)		CPU: 56核 内存: 96 GB (I/O优化)	按量	管理 远程连接
▶ 网络和安全	└ c7e8e1064cdf2480888551 ♥Q №	陸南 I 可用区 B 192. (私有)	十 专有网络	GPO:NVIDIA M40-2 100Mbps(峰值)	17-05-09 09:42 创建	升降配 更多▼
标签管理 操作日志	i-wz9i2 cef28696cdfb245ed9801e< ♥ 爻 ⊵	华南 1 可用区 B 4.75(3钟性) 6 运行	中 专有网络	CPU: 1核 内存:2 GB (I/O优化) 100Mbps (峰值)	按量 17-04-17 18:21 创建	管理 远程连接 升降配 更多▼

- 3. 单击右侧的 更多 > 安全组配置。
- 4. 选择容器服务集群对应的安全组并单击 配置规则。

<	o c7e8e1064cdf2480888551	o c7e8e1064cdf2480888551					
实例详情	安全组列表			加入安全組			
本实例磁盘	安全组ID/名称	描述	所履专有网络	操作			
本实例快照							
本实例安全组	sg-wz9ha54179m5cmhroysw alicloud-cs-auto-creat	security group of ACS	vpc-	配置规则 移出			
本实例安全防护							

5. 单击 添加安全组规则,填写规则信息并单击确定。

编辑安	安全组规则					×
	网卡类型:	内网				
	规则方向:	入方向	•			
	授权策略:	允许	•			
	协议类型:	自定义 TCP	•			
	* 端口范围:	32775/32775		0		
	授权类型:	地址段访问	•			
	* 授权对象:	0.0.0/0		0 ∌ 置	女我 设	
	优先级:	1		0		
	描述:					
		L 长度为2-256个字符,不能	能以htt	p://或hi	ttps://开头。	



如果您使用的是 Linux 机器,你只需要运行以下命令即可 SSH 访问您的 Jupyter 服务。

ssh -p 32775 root@39.***.**.236

其中,32775 为 SSH 访问的端口;39.***.**.236 是您的 ECS 实例绑定的弹性公网 IP。

Windows 环境下访问

1. 运行 PuTTY, 配置 Session。

设置 IP 地址(ECS 实例的弹性公网 IP)、端口(SSH 访问的端口,本示例为 32775)和连接 方式(SSH),并单击 Open。

🕵 PuTTY Configuration	8
Category:	
Session Logging Creminal Keyboard Bell Features Window Appearance Behaviour Translation Selection Colours Connection Data Proxy Telnet Rlogin SSH Serial	Basic options for your PuTTY session Specify the destination you want to connect to Host Name (or IP address) 39.1044226 32775 Connection type: Raw Ielnet Rlogin SSH Serial Load, save or delete a stored session Saved Sessions Default Settings Load Save Default Settings Load Save Delete
	Always Never Only on clean exit
About	Open Cancel

2. 在弹出的会话窗口登录并访问服务。

输入登录账号 root 并输入您在 创建 Jupyter 环境 时 启用SSH登录 所输入的 SSH密码。

可以看到您成功访问到 Jupyter 服务。

🛃 root@c7e8e1064cdf2480888551710e43a0c11-node1: ~		×
login as: root root@39.10236's password:		^
Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 4.4.0-63-generic x86_64)		
* Documentation: https://help.ubuntu.com/ New release '16.04.2 LTS' available.		
Run 'do-release-upgrade' to upgrade to it.		
Welcome to Alibaba Cloud Elastic Compute Service !		
Last login: Wed May 17 16:22:42 2017 from 106.1		

1.3.4 通过端口转发方式访问 Jupyter 服务

通过端口转发方式访问 Jupyter 服务具有以下优点,但是设置较复杂。

- 您不需要购买负载均衡实例,可以节省您的费用。
- 您不需要在官网上开放端口即可从外部访问 Jupyter 服务。

下面的示例假设通过本地的 12345 端口进行转发 SSH 访问您的 Jupyter 服务。

应用 : myde	evbox									刷新
基本信息										
应用名称:	mydevbox				创建时间:	2017-05-16	更新时间	: 2017-05-16		所在集群: EGS
触发器 1.	每种类型的触发	器只能创建1个	0							创建触发器 ^
目前没有任何的	度发器,点击右。	上角按钮创建触	发器							
服务列表	容器列表	路由列表	日志	事件						
名称/ID		状态	健康检测	则領	總	端口		容器IP	节点IP	操作
mydevbox_ju 718623ea4d	ıpyter () e94f4a	running	正常	re sł	agistry-vpc.cn na256:213c1c3ad	6006/tcp 192. :32775->22/td 192. :32774->8888	cp B/tcp	172.	192.	· 翻除 停止 监控 日志 远程终端
mydevbox_te 5846e35b8d	ensorb () 81f82e	running	正常	re st	egistry-vpc.cn na256:2b4c70877	192.168		172.	192.	删除 停止 监控 日志 远程终端

步骤 1 打通 SSH Tunnel

Mac OS X 和 Linux 环境下

运行以下命令打通本地端口和 ECS 实例的连接。

```
ssh -ND 12345 root@39.***.**.236
```

12345 是您要使用的本地端口,您可以自定义。39.***.**.236 是您的 ECS 实例绑定的弹性公 网 IP。



您可以登录 ECS 管理控制台 查看 ECS 实例绑定的弹性公网 IP。如果您的实例未绑定弹性公网 IP,您需要先为其绑定弹性公网 IP,参见 管理弹性公网 IP。

实例名	3称 🔻	输入实例名称模糊查试	1		搜索					高级搜索	2	¢	?
3	e例ID/名称		监控	所在可用区	IP地址	状态(全部) 👻	网络类型(全部) 👻	配置	付费方式(全部) 👻			ł	操作
i c	-wz9eqt798e 7e8e1064cd	•	2 ⊾	华南 1 可用区 B	39236(弹性) 192. (私有)		专有网络	CPU: 56核 内存:96 GB (I/O优化) GPU:NVIDIA M40*2 100Mbps(峰值)	按量 17-05-09 09:42 创建	ŧ	管理 ; 升降配	远程连 更	接 多▼

Windows 环境下

1. 配置本地端口跳转。

运行 PuTTY 并配置 SSH Tunnels。

- a. 填写 Source Port。本示例中为 12345。
- b. 选择 Dynamic。
- **C.** 单击 Add。



2. 登录 ECS 实例。

a. 运行 PuTTY, 配置 Session 并单击 Open。

填写 IP 地址,即 ECS 实例的弹性公网 IP,本示例中为 39.***.**.236。

🕵 PuTTY Configuration	Σ	X
Category:		
Session Logging Terminal Keyboard Bell Features Window Appearance Behaviour Translation Selection Colours	Basic options for your PuTTY session Specify the destination you want to connect to Host Name (or IP address) 39 236 Connection type: Raw Telnet Rlogin SSH Serial Load, save or delete a stored session Saved Sessions	
E Connection Data Proxy Telnet Rlogin SSH Serial	Default Settings Load Save Delete Delete Delete Close window on exit Image: Close window on exit Always Never Image: Only on clean exit	
About	Open Cancel	

b. 在弹出的会话框里输入您的登录账号和 ECS 实例的登录密码。

您成功登录到 ECS 实例上。

🚰 root@c7e8e1064cdf2480888551710e43a0c11-node1: ~	•	×
login as: root root@39.10 .4 .236's password: Welcome to Ubuntu 14.04.5 LTS (GNU/Linux 4.4.0-63-generic x86_64)		^
* Documentation: https://help.ubuntu.com/ New release '16.04.2 LTS' available. Run 'do-release-upgrade' to upgrade to it.		
Welcome to Alibaba Cloud Elastic Compute Service !		
Last login: Wed May 17 16:22:42 2017 from 106.		

步骤 2 配置浏览器的网络连接

Firefox 浏览器

如果您使用的是 Firefox 浏览器,打开浏览器,单击 工具 > 选项 > 高级 > 网络 选项卡 > 单击 连接 对应的 设置 按钮 > 设置 SOCKS 主机。

文件(E) 编辑(E)	查看(<u>V</u>) 历史(<u>S</u>) 书签(<u>B</u>) 工具(<u>T</u>) 帮助(<u>H</u>)	
🗱 选项	× +	
Firefox	about:preferences#advanced	▼ C Q 百度 <ctrl+k></ctrl+k>
	repairing and the second	
0 常规	高级	
Q 搜索	常规 反馈 网络 更新 证书	
● 内容		
🛕 应用程序	连接	
∞ 隐私	配置 Firefox 如何连接至国际互联网	设置(E)
🖴 安全	网络内容缓存	
🚺 同步	您的网络内容缓存当前已使用 197 MB 磁盘空间	立即清除(<u>C</u>)
👌 高级	无视自动缓存管理(Q) 缓存最大为(L) 350 MB磁盘空间	

连接设置	2 1			×				
一配置)	访问国际互联网的	代理						
07	下使用代理(<u>Y</u>)							
ÓÉ	◎ 自动检测此网络的代理设置(W)							
◎ 使用系统代理设置(U)								
 ● 長方が代生業(位) ● 手动配置代理:(M) 								
	HTTP 代理:(X)	-	端口:(<u>P</u>)	0				
	-	🔲 为所有协议使用相同代理	(<u>S</u>)	¥				
	SS <u>L</u> 代理:		端口:(<u>O</u>)	0				
	<u>E</u> TP 代理:		端口:(<u>R</u>)	0				
	SOCKS 主机:	localhost	⊯□・(T)	12345				
		localitost	утаці · (<u>т</u>)	12343				
		SOCKS v4 ● SOCKS	ynu · (⊥) v5	<u>D</u> NS				
5	不使用代理:(<u>N</u>)	SOCKS v4 ● SOCKS	ynu · (1) ⊻5	DNS				
2	不使用代理:(<u>N</u>) localhost, 127.0.	○ SOC <u>K</u> S v4 ● SOCKS 0.1	ynu · (1) ⊻5	<u>D</u> NS				
	不使用代理:(<u>N</u>) localhost, 127.0.0	○ SOC <u>K</u> S v4	ynu · (⊥) ⊻5	DNS				
	不使用代理:(<u>N</u>) localhost, 127.0.0 例如:.mozilla.org	© SOC <u>K</u> S v4 SOCKS 0.1 g, .net.nz, 192.168.1.0/24	ynu · (⊥) ⊻5	<u>DNS</u>				
	不使用代理:(<u>N</u>) localhost, 127.0. 例如:.mozilla.org 自动代理配置(P <u>A</u>	© SOC <u>K</u> S v4 SOCKS 0.1 g, .net.nz, 192.168.1.0/24 <u>A</u> C) :	yn⊔ · (⊥) ⊻5	DNS				
; () () () () () () () () () () () () ()	不使用代理:(<u>N</u>) localhost, 127.0. 例如:.mozilla.org 自动代理配置(P <u>A</u>	 SOC<u>K</u>S v4 ● SOCKS 0.1 g, .net.nz, 192.168.1.0/24 (C) : 	yīni · (1) w5 □ 远程	<u>125+5</u> <u>■</u> DNS 新载入(<u>E</u>)				
	不使用代理:(<u>N</u>) localhost, 127.0.0 列如:.mozilla.org 自动代理配置(P <u>A</u>	 SOCKS v4 ● SOCKS 0.1 g, .net.nz, 192.168.1.0/24 	ynu · (1) w5 □ 远程	<u>12545</u> <u>■</u> DNS 新载入(E)				
。 () () () () () () () () () () () () ()	不使用代理:(<u>N</u>) localhost, 127.0.0 例如:.mozilla.org 自动代理配置(P <u>A</u> 果密码已保存,不	 ○ SOCKS v4 ● SOCKS 0.1 g, .net.nz, 192.168.1.0/24 	yīni · (1) ⊻5 远程	<u>12545</u> <u>■</u> <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> ■ <u>12545</u> <u>12545</u> <u>12545</u> <u>12545</u> <u>12545</u> <u>12545</u> <u>12545</u> <u>12545</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>12555</u> <u>125555</u> <u>125555</u> <u>125555</u> <u>1255555</u> <u>12555555555555555555555555555555555555</u>				
。 (((一 (一 () (不使用代理:(№) localhost, 127.0.0 例如:.mozilla.org 自动代理配置(PA	 SOCKS v4 ● SOCKS 0.1 g, .net.nz, 192.168.1.0/24 .C): □ 	ym□ · (1) v5 □ 远程	±Bab(4.1)				
。 () () () () () () () () () () () () ()	不使用代理:(<u>N</u>) localhost, 127.0.0 例如:.mozilla.org 自动代理配置(P <u>A</u>	◎ SOC <u>K</u> S v4 ④ SOCKS 0.1 g, .net.nz, 192.168.1.0/24 <u>(</u> C) : 張示身份验证(I)	yīni · (1) w5 □ 远程	IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII				

Chrome 浏览器

运行以下命令。

Chrome --proxy-server="socks5://localhost:12345" --host-resolver-rules ="MAP * 0.0.0.0 , EXCLUDE localhost" --user-data-dir=/tmppath/

其中:

- 12345 是您所使用的本地端口。
- 若是 Windows 系统,这里的 /tmppath/可以写成类似 d:/tmppath;若是 Linux 或者 Mac OS X,可以直接写成 /tmp/。

在不同的操作系统中, Chrome 的位置不同,请参见下表:

操作系统	Chrome 的位置
Mac OS X	/Applications/Google\ Chromapp/ Contents/MacOS/Google\ Chrome
Linux	/usr/bin/google-chrome
Windows	<i>C:\Program Files (x86)\Google\</i> <i>Chrome\Application\chrome.exe</i>

以 Windows 系统为例:

C:\windows\system32\cmd.exe	
Microsoft Windows [版本 6.1.7601] 版权所有 (c) 2009 Microsoft Corporation。保留所有权利。	
C:\Users >"C:\Program Files (x86)\Google\Chrome\Application e"proxy-server="socks5://localhost:12345"host-resolver-rules=" .0 , EXCLUDE localhost"user-data-dir=d:/tmppath	n∖chrome.ex MAP × 0.0.0
C:\Users\ >	

步骤 3 访问 Jupyter 服务

在您的浏览器中输入 Jupyter 服务的访问地址。本示例中为 192.***.*.83:32774。即可成功地从外部 SSH 访问到 Jupyter 服务。

Firefox :

● 火狐主页 × C Jupyter Notebook	× +		
(33:32774/login?next=%2Ftree%3F		🦁 器 🔻 🧉 🔍 百度 <ctrl+k></ctrl+k>	☆自
and a second statement of the second se			
💭 jupyter			
	Deseward	Login	
	rasswolu.	Log III	
Chrome :			

Log in

C Jupyter	Notebook	×	
$\leftrightarrow \ \ni \ \mathbf{G}$	① 不安全	192. 32774/login?next=%2Ftree%3F	
	💭 Jupyter		

Password:

1.4 模型训练

1.4.1 单机模型训练

利用阿里云提供的弹性计算资源和存储服务,执行您的模型训练代码,快速开始进行单机版训练迭代。训练过程中,您可以随时查看日志和监控训练状态。

准备工作

在运行模型训练任务之前,请确认以下工作已经完成:

- 创建包含适当数量弹性计算资源 (ECS 或 EGS) 的容器集群。创建步骤请参考 创建容器集群。
- 如果您需要使用对象存储服务(OSS)保存用于模型训练的数据,您需要使用相同账号创建
 OSS Bucket;然后在上面的容器集群中创建数据卷,用于将 OSS Bucket 作为本地目录挂载到
 执行训练任务的容器内。参见创建数据卷。
- 将模型训练代码同步到有效的 GitHub 代码仓库中。

约定

为了方便您的训练代码读取训练数据,输出训练日志,保存训练迭代状态数据(checkpoint),请 注意以下约定。

- 训练数据将会自动存放在 / input 目录,且保持与 OSS 中路径一致。用户代码需要从该目录中 读取训练数据。
- 代码输出的所有数据都应保存到 /output 目录,包括日志, checkpoint 文件等。所有保存在 / output 目录中的文件将以同样的目录结构被自动同步到用户的 OSS Bucket。
- 如果训练代码中需要使用特别的 Python 依赖库,请将所有依赖写入名为 requirements.txt 的配置文件,并保存在 GitHub 仓库中代码根目录下。

操作流程

- 1. 登录 容器服务管理控制台。
- 2. 在 Swarm 菜单下,单击左侧导航栏中的 镜像与方案 > 解决方案。

3. 在模型训练框中单击创建。

容器服务		11	机器学习		
Swarm	Kubernetes				
概览			レ 模型开发	「模型训练	P 模型预测
应用			使用 Jupyter 快速开发、调试模型代码,并使用	使用CPU, GPU集群训练模型。支持TensorFlow,	使用CPU, GPU进行模型预测。支持TensorFlow
服务			TensorBoard 监控训练过程。支持TensorFlow, Keras等。	Keras等,及分布式训练方法。支持TensorBoard 监控训练过程。	Serving服务端。支持服务的负载均衡和弹性伸缩。 缩。
集群			创建 指南	创建 指南 历史记录	创建 指南
节点		[区块链	3	
网络					
数据卷			Hyperledger Fabric		
配置项			支持 Hyperledger Fabric v1.0 的区块链网络自动 化配置和部署		
▼ 镜像与方到					
镜像	t		创建 指南		
编排模板	Ē				
解决方案	2				

4. 设置单机训练任务的基本信息。

进入模型训练页面,填写以下信息用于配置训练任务。

模型训练		
集群	EGS-cluster	
应用者	tf-train-standalone	
训练框架	tensorflow:1.1.0	
	□ 分布式训练	
单Worker使用GPU数量	0	
数据卷名	tfoss 🔻	
Git地址	https://code.aliyun.com/deeplearning/mnist-example:	
	☑ 私有代码仓库	
Git用户名	t	
GIt密码	••••••	
执行命令	python mnist_save.pytraining_iteration=1000	
	log_dir=/output/training_logs	
	2. 训练听统	11
	□ WISAmlı	
训练日志路径	/output/training_logs	
	请保证与训练代码中日志输出的路径一致	

- 集群:选择将要运行单机模型训练任务的容器集群。
- 应用名:为运行单机模型训练任务所创建的应用名称,例如 tf-train-standalone。名称可以包含 1~64 个字符,包括数字、英文字母和连字符(-)。
- 训练框架:选择用来进行模型训练的框架。目前支持的框架包括 TensorFlow、 Keras、Python 以及自定义镜像,选择自定义镜像要求您输入合法的镜像地址。

- 分布式训练:选择是否执行分布式训练任务。这里不要勾选,则执行单机训练。
- 单Worker使用GPU数量:指定所使用的 GPU 数量,如果为 0 表示不使用 GPU,那么将使用 CPU 训练。
- 数据卷名:指定为用于存储训练数据的数据来源,可以选择对象存储服务在该集群中创建的数据卷的名称;也可以选择本地目录,但要求填写绝对路径;或者选择不使用数据卷。本示例中使用名为 tfoss 的数据卷。
- Git地址:指定训练代码所处的 GitHub 代码仓库地址。

📋 说明:

目前仅支持 HTTP 和 HTTPS 协议,不支持 SSH 协议。

- 私有代码仓库:如果您使用私有代码仓库,请勾选此选项并设置 Git用户名(您的 GitHub 账号)和 Git密码。
- 执行命令:指定如何执行上述代码的命令,以进行模型训练。

说明	:

如果你选择的是支持 Python3 的框架,请在命令行中直接调用 python3,而不是 python。

 训练监控:是否使用 TensorBoard 监控训练状态;一旦选择监控,请指定合法的训练日志路径,并确保与训练代码中日志输出的路径一致。例如,指定训练日志路径为/output/ training_logs,则用户代码也应该将日志输出到同样的路径下。图中示例是通过指定用户代码命令行参数 --log_dir 保证这一点的。

🗐 说明:

这里的训练日志指的是使用 TensorFlow API 输出的供 Tensorboard 读取的事件文件,以及 保存了模型状态的 checkpoint 文件。

- 5. 设置完毕后,单击确定,创建执行模型训练任务的应用。
- 6. 单击左侧导航栏中的 应用,单击上面创建的应用 tf-train-standalone。

集群: EGS-cluster	▼ 🕑 隐藏系统应用 🔲 隐藏器	离线应用 🔲 隐藏在线的	如用		名称 ▼
应用名称	描述 状态	容器状态	创建时间 🔺	更新时间 🔺	操作
tf-train-standalone	● 就绪	就绪:0 停止:0	2017-04-19 14:25:36	2017-04-19 14:25:36	停止 变更配置 删除 重新部署 事件

7. 单击路由列表,可以看到一个以 tensorboard 开头的链接,用来打开 TensorBoard 监控页面。

应用:tf-tra	in-standalone									刷新
基本信息										
应用名称:	tf-train-standa	lone			创建时间: 2017-04-1	19	更新时间: 2017-04-19	所在集群: EGS-cluster	÷	
触发器 1.4	每种类型的触发	器只能创建1个	0							创建触发器 ^
目前没有任何意	触发器,点击右	上角按钮创建的	被发器							
服务列表	容器列表	路由列表	日志	事件						
路由地址 (集	路由地址(集群绑定SLB后路由地址才能被访问) 攝代							操作		
tensorboard	tensorboard 设置服务权重									

单击这个链接,可以查看训练监控数据。

\rightarrow C \square tensorboard-201704141	00823.c288	914c0cf6340	dcb980086	e691e3ada8.c	n-shenzhen.aliconta	iner.com/?spm=51	76.2020520152.221.16.4Dx1HQ			S) :
TensorBoard	SCALARS	IMAGES						G	۵	0
Write a regex to create a tag group	×	accuracy_1								1
Split on underscores		accuracy_1								
Data download links		0.900								
Tooltip sorting method: default	*	0.700								
Smoothing 0.6		0.300) 1.000k 2.00	0k 3.000k 4.000	< 5.000k					
Horizontal Axis										
STEP RELATIVE WALL		cross_entropy	/_1							1
		global_step								1
Runs		layer_linear								8
Write a regex to filter runs										
✓ ○ .										
✓ ○ test										
🗹 🔿 train										
TOGGLE ALL RUNS										
/output/training_logs										

8. 单击 日志,查看执行训练任务过程中,用户代码输出到标准输出/标准错误的日志内容。



为了运行训练任务应用,一般会自动创建多个服务容器,分别运行不同的程序分支。比如 worker 容器一般用来运行用户的模型代码,tensorboard 容器用来运行 TensorBoard 训练监 控。具体生成的服务 / 容器,可以单击 服务列表,或 容器列表 查看。

可以按服务容器名称筛选某个容器,查看该容器运行程序输出到 stdout/stderr 的详细日志,比如 tf-train-standalone_worker_1。也可以按时间和显示数量,过滤要查看的日志内容。同样,也可以选择下载日志文件到本地。

送 说明:

这里提到的标准输出/标准错误日志,请区别于上述 Tensorboard 事件文件和 checkpoint 文

件。

服务列表 容函列表 路由列表 日志 事件 每个容易查看条目: 100余 100余 皮容易名 按容易名為 按目本 按容易名為為 按目本 丁和日本 每个容易查看条目: 100余 2017-64-17705:46:82.6538534892 Cloning training code from http://www.github.com/wsxiaozhang/tf-mnist.git tf-train-standalone_worker_1 2017-64-17705:46:82.6523634892 Cloning into 'f-mnist' tf-train-standalone_worker_1 2017-64-17705:46:82.6524726122 Cloning into 't-mnist' tf-train-standalone_worker_1 2017-64-17705:46:82.652745122 Done Cloning code tf-train-standalone_worker_1 2017-64-17705:46:82.652765431022 Done Cloning code tf-train-standalone_worker_1 2017-64-17705:46:82.65272572 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE1 instructions, but these are available on your machine and could speed up CPU computations. tf-train-standalone_worker_1 2017-64-17705:46:11.6552374927 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE1 instructions, but these are available on your machine and could speed up CPU computations. tf-train-standalone_worker_1 2017-64-17705:46:11.6552394282 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SK21 instructions, but these are available on your mach					
中个容易重泰引: 100条 ・	服务列表 容器列表 路由列表	日志 事件			
<pre>tf-train-standalone_worker_1 2017-04-17T05:46:02.658363499Z Cloning training code from http://www.github.com/wsxiaozhang/tf-mnist.git tf-train-standalone_worker_1 2017-04-17T05:46:02.6582720 Zone cloning code. tf-train-standalone_worker_1 2017-04-17T05:46:07.67265115822 Run training code under /starter/tf-mnist as: python mnist_save.py —training_ite ration=1000 —data_dir=/input/tensorflow/mnist/data —log_dir=/output/training_logs tf-train-standalone_worker_1 2017-04-17T05:46:11.0552196122 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE3 instructions, but these are available on your machine and could speed up CPU computations. tf-train-standalone_worker_1 2017-04-17T05:46:11.05527972 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations. tf-train-standalone_worker_1 2017-04-17T05:46:11.05527972 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations. tf-train-standalone_worker_1 2017-04-17T05:46:11.055232727 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SXE4.2 instructions, but these are available on your machine and could speed up CPU computations. tf-train-standalone_worker_1 2017-04-17T05:46:11.0552321732 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SXE4.2 instructions, but these are available on your machine and could speed up CPU computations. tf-train-standalone_worker_1 2017-04-17T05:46:11.0553231732 W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SXE4.2 instructions, but these are available on your machine and could speed up CPU computations. tf-train-standalone_worker_1 2017-04-17T05:46:11.056322172</pre>	每个容器查看条目: 100条 🛟	_	按容器名称筛选: tf-train	in-stanc 🗧 按日志起始时间筛选:	下载日志
	tf-train-standalone_worker_1 201 tf-train-standalone_worker_1 201 tf-train-standalone_worker_1 201 tf-train-standalone_worker_1 201 ration=1000data_dir=/input/tens tf-train-standalone_worker_1 201 compiled to use SSE3 instructions, tf-train-standalone_worker_1 201 compiled to use SSE4.1 instruction tf-train-standalone_worker_1 201 compiled to use SSE4.2 instruction tf-train-standalone_worker_1 201 compiled to use SSE4.2 instructions, tf-train-standalone_worker_1 201 compiled to use AVX instructions, tf-train-standalone_worker_1 201 compiled to use AVX instructions, tf-train-standalone_worker_1 201 compiled to use AVX instructions, tf-train-standalone_worker_1 201 topuled to use AVX instructions, tf-train-standalone_worker_1 201 tf-train-standalone_worker_1 201	<pre>7-04-17T05:46:02.6583634892 Clo 7-04-17T05:46:02.6624726192 Clo 7-04-17T05:46:07.6714531022 Don 7-04-17T05:46:07.6726518502 Run orflow/mnist/datalog_dir=/ou 7-04-17T05:46:11.0552196182 W t but these are available on you 7-04-17T05:46:11.0552707972 W t is, but these are available on you 7-04-17T05:46:11.0552854032 W t but these are available on you 7-04-17T05:46:11.0552834032 W t but these are available on you 7-04-17T05:46:11.0553133202 W t but these are available on you 7-04-17T05:46:11.05531372 W t but these are available on you 7-04-17T05:46:11.063234392 WAR ated and will be removed after 7-04-17T05:46:11.1053927302 Tns 7-04-17T05:46:11.1053927302 Tn 100kup method of the returned 7-04-17T05:46:11.18169347232 Tr 7-04-17T05:46:11.8169347232 Tr 7-04-17T05:46:11.8169347232 Tr 7-04-17T05:46:11.8169347232 Tr 7-04-17T05:46:11.8169347232 Tr 7-04-17T05:46:11.8169347232 Tr 7-04-17T05:46:11.8169347232 Tr 7-04-17T05:46:11.8169347232 Tr 7-04-17T05:46:11.8169725322 Ext 7-04-17T05:46:11.816972532 Ext 7-04-17</pre>	<pre>ixteac.commodel. In train ning training code from h ning into 'tf-mnist' e cloning code. training code under /star tput/training_logs ensorflow/core/platform/cp our machine and could speed ensorflow/core/platform/cp our machine and could speed ensorflow/core/platform/cp machine and could speed u ensorflow/core/platform/cp machine and could speed u ensorflow/core/platform/cp machine and could speed u ensorflow/core/platform/cp machine and could speed u ensorflow/core/platform/cp machine and could speed u NINGitensorflow:From mnist 2017-03-02. tructions for updating: `tf.global_variables_init NING:tensorflow:From mnist -01-07. tructions for updating: s op will be removed after table. ining model racting /input/tensorflow// racting /input/t</pre>	<pre>interanc ; XCH askader y laymode. http://www.github.com/wsxiaozhan rter/tf-mnist as: python mnist_s pu_feature_guard.cc:45] The Tens up CPU computations. pu_feature_guard.cc:45] The Tens ed up CPU computations. pu_feature_guard.cc:45] The Tens up CPU computations. pu_feature_guard.cc:45] The Tens up CPU computations. pu_feature_guard.cc:45] The Tens up CPU computations. pu_feature_guard.cc:45] The Tens up CPU computations. t_save.py:43: initialize_all_var tializer` instead. t_save.py:49: index_to_string (f r the deprecation date. Please sr /mnist/data/train-images-idx3-ub /mnist/data/t10k-labels-idx1-ub /mnist/data/t10k-labels-idx1-ub /mnist/data/t10k-labels-idx1-ub</pre>	<pre>P3(E)25 g/tf-mnist.git ave.pytraining_ite orFlow library wasn't orFlow library wasn't orFlow library wasn't orFlow library wasn't iables (from tensorfl rom tensorflow.contri witch to index_to_str tyte.gz te.gz te.gz te.gz</pre>

9. 单击 服务列表,如果看到 worker 服务的状态为"已停止",说明此次训练任务已经执行结束。

服务列表	容器列表	路由列表	日志	事件				
服务名称			所属应用		服务状态	容器状态	镜像	撮作
tf-train-stand	alone		tf-train-sta	ndalone	● 就绪	就绪:1 停止:0	1210	停止 重治 重新调度 交更配置 删除 事件
worker		ł	tf-train-stan	Idalone	●巳停止	就绪:0 停止:1		启动 重启 重新调度 交更配置 删除 事件

10.通过 OSS 客户端可以看到训练结果自动保存在 OSS Bucket 中了。

训练结束后,系统自动将所有保存在本地 /output 路径中的文件,复制到用户给定数据卷所对 应的对象存储服务(OSS)Bucket 中。通过 OSS 客户端,可以看到本示例中用户代码写入 / output 的事件文件和 checkpoint 文件都已经备份到 OSS Bucket 中了。

OSS-1.2	.0		
tensorflow-samples → training_logs			٩
上传 新建文件夹 下载 复制	删除 获取地址	设置HTTP头 粘贴	RAM授权
文件名	大小	创建时间	
test		-	
train	-	-	
checkpoint	109B	2017-04	-17 13:46:11
ckpt.data-00000-of-00001	30.66K	B 2017-04	-17 13:46:12
ckpt.index	159B	2017-04	-17 13:46:12
ckpt.meta	16.87K	B 2017-04	-17 13:46:12

11.管理训练任务。

如果需要停止、重启、删除(比如为了释放计算资源给新的训练任务)某个训练任务,只需回到应用>应用列表页面。找到对应的应用,在右侧进行操作即可。

集群: EGS-cluster	▼ 🖉 🎘	藏系统应用 🔲 隐藏离线应用	月 🔲 隐藏在线应	用		名称 ▼
应用名称	描述	状态	容器状态	创建时间 🔺	更新时间 🔺	操作
tf-train-standalone		●就绪	就绪:1 停止:1	2017-04-19 14:25:36	2017-04-19 15:33:40	停止 安更配置 删除 重新部署 事件

■ 说明:

利用本节描述的模型训练服务,您不仅可以从零开始训练一个模型,同样也可以在一个已有模型的基础(checkpoint)之上,使用新的数据继续训练(比如 fine tuning)。利用已创建的应用,可以不断通过更新配置的方式调整超参数,进行迭代训练。

1.4.2 分布式模型训练

利用阿里云提供的弹性计算资源和存储服务,执行用户的模型训练代码,快速开始进行分布式训练。训练过程中,您可以控制如何分配计算资源(CPU,GPU),随时查看日志和监控训练状态,并将训练结果备份到存储服务中。

利用本文档描述的模型训练服务,您不仅可以从零开始训练一个模型,同样也可以在一个已有模型的基础(checkpoint)之上,使用新的数据继续训练(比如 fine tuning)。利用已创建的应用,可以不断通过更新配置的方式调整超参数,进行迭代训练。

准备工作

在运行模型训练任务之前,请确认以下工作已经完成:

- 创建包含适当数量弹性计算资源(ECS或EGS)的容器集群。创建步骤请参考创建容器集群。
- 如果您需要使用对象存储服务(OSS)保存用于模型训练的数据,您需要使用相同账号创建
 OSS Bucket;然后在上面的容器集群中创建数据卷,用于将 OSS Bucket 作为本地目录挂载到
 执行训练任务的容器内。参见 创建数据卷。
- 将模型训练代码同步到有效的 GitHub 代码仓库中。

约定

为了方便用户的训练代码读取训练数据,输出训练日志,保存训练迭代状态数据(checkpoint

-),请注意以下约定。
- 训练数据将会自动存放在 / input 目录, 且保持与 OSS 中路径一致。用户代码需要从该目录中 读取训练数据。
- 代码输出的所有数据都应保存到 /output 目录,包括日志,checkpoint 文件等。所有保存在 / output 目录中的文件将以同样的目录结构被自动同步到用户的 OSS Bucket。如果训练代码中需要使用特别的 Python 依赖库,请将所有依赖写入名为 requirements.txt 的配置文件,并保存在 GitHub 仓库中代码根目录下。

操作步骤

- 1. 登录 容器服务管理控制台。
- 2. 在 Swarm 菜单下,单击左侧导航栏中的 镜像与方案 > 解决方案。
- 3. 在模型训练框中单击创建。

容器服务		11	机器学习		
Swarm	Kubernetes				
概览			〕模型开发	▲ 【 横型川族	P 模型预测
应用 服务			使用 Jupyter 快速开发、调试模型代码,并使用 TensorBoard 监控训练过程。支持TensorFlow, Keras等.	使用CPU,GPU集群训练模型。支持TensorFlow, Keras等,及分布式训练方法。支持TensorBoard 监控训练过程。	使用CPU, GPU进行模型预测。支持TensorFlow Serving服务端。支持服务的负载均衡和弹性伸缩。
集群			创建 指南	创建 指南 历史记录	创建 指南
节点			区块链	3	
网络					
数据卷			Hyperledger Fabric		
配置项			支持 Hyperledger Fabric v1.0 的区块链网络自动 化配置和部署		
▼ 镜像与方言					
镜像	÷		创建 指南		
编排模板	ž				
解决方案	2				

4. 设置分布式训练任务的基本信息。

进入模型训练页面,填写以下信息用于配置训练任务。

模型训练		
集群	EGS-cluster	¥
应用名	tf-train-distributed	
训练框架	tensorflow:1.1.0	v
	☑ 分布式训练	
	Parameter Server数量	1
	Worker数量	2
单Worker使用GPU数量	1	
数据卷名	tfoss	v
Git地址	https://code.aliyun.com/deeplearning,	/mnist-example:
	☑ 私有代码仓库	
Git用户名	te	
Git密码	••••••	
执行命令	python mnist_dist_train.py data_dir=/input/tensorflow/mnist/data log_device_placement=False log_dir=/output/mnist/dist/logbatch learning_rate=0.01sync_replicas=Fa	a/train_steps=5000 1_size=100 Ise <u>nym</u> opus=2
	✔ 训练监控	
训练日志路径	/output/training_logs 请保证与训练代码中日志输出的路径一到	ζ

- 集群:选择将要运行分布式模型训练任务的容器集群。
- 应用名:为运行训练任务所创建的应用名称,例如 tf-train-distributed。名称可以包含 1~64
 个字符,包括数字、英文字母和连字符(-)。

- 训练框架:选择用来进行模型训练的框架。目前支持的框架包括
 TensorFlow、Keras、Python 以及自定义镜像,选择自定义镜像要求您输入合法的镜像地址。
- 分布式训练:选择是否执行分布式训练任务。这里要勾选,则执行分布式训练。会出现具体的配置内容,包括 Parameter Server 数量和 Worker 数量,用于指定分布式训练时参数 服务器节点的数量和进行具体计算的 Worker 节点数量。示例中设置了1个参数服务器和2 个Worker。
- 单Worker使用GPU数量:指定每个 Worker 在执行训练任务时所使用的 GPU 数量,示例中选择每个 Worker 使用 1 块 GPU 卡。如果为 0 表示不使用 GPU,那么将使用 CPU 训练。
- 数据卷名:指定为用于存储训练数据的数据来源,可以选择对象存储服务在该集群中创建的数据卷的名称;也可以选择本地目录,但要求填写绝对路径;或者选择不使用数据卷。本示例中使用名为 tfoss 的数据卷。
- Git地址:指定训练代码所处的 GitHub 代码仓库地址。



目前仅支持 HTTP 和 HTTPS 协议,不支持 SSH 协议。

- 私有代码仓库:如果您使用私有代码仓库,请勾选此选项并设置 Git用户名 (您的 GitHub 账 号)和 Git密码。
- 执行命令:指定如何执行上述代码的命令,以进行模型训练。

如果你选择的是支持 Python3 的框架,请在命令行中直接调用 python3,而不是 python。

· 训练监控:是否使用 TensorBoard 监控训练状态;一旦选择监控,请指定合法的训练日志
 路径,并确保与训练代码中日志输出的路径一致。例如,指定 训练日志路径为 /output/
 training_logs,则用户代码也应该将日志输出到同样的路径下。图中示例是通过指定用户
 代码命令行参数 --log_dir 保证这一点的。

📕 说明:

这里的训练日志指的是使用 TensorFlow API 输出的供 Tensorboard 读取的事件文件,以及 保存了模型状态的 checkpoint 文件 (使用 Tensorflow 进行分布式训练时,一般由担任 Chief 角色的 Worker 负责保存 checkpoint)。

- 5. 设置完毕后,单击确定,创建执行模型训练任务的应用。
- 6. 单击左侧导航栏中的 应用,单击上述创建的应用 tf-train-distributed。

集群:	EGS-cluster	•	隐藏系统应用 🔲 隐藏离线应用	□ 隐藏在线应用	ŧ		名称 ▼	
应用名	称	描述	状态	容器状态	创建时间 🔺	更新时间 ▲		操作
tf-train	-distributed		●就绪	就绪:3 停止:2	2017-04-19 15:53:28	2017-04-19 15:53:34	停止 变更配置 删除 重	新部署 事件

 7. 单击 路由列表,可以看到 2 个以 tensorboard 开头的链接,每个链接用于打开一个 Worker 节点 对应的 TensorBoard 监控页面。tensorboard0 开头的链接代表 Worker 0, tensorboard1 开头 的链接代表 Worker 1。

服务列表	容器列表	路由列表	日志	事件			
路由地址(集群绑定SLB后路由地址才能被访问) 操作							
tensorboard1-2017041							
tensorboard0-2017041- 设置服务权重							

单击链接,可以查看训练监控数据。

C 🕑 C tensorboard0-20170414100823.c288914c0cf6340dcb98008e691e3ada8.cn-shenzhen.alicontainer.com/?spm=5176.2020520152.221.17.VdajsK									☆	1
TensorBoard	SCALARS	IMAGES		GRAPHS			EMBEDDINGS	G	¢	
Write a regex to create a tag group	×	accuracy_1								
Split on underscores		accuracy_1								
Data download links		0.900								
Tooltip sorting method: default	*	0.700								
Smoothing 0.6	5	0.500	00 1.000k 2	.000k 3.000k 4.0	00k 5.000k					
Horizontal Axis		cross_entrop	oy_1							
STEP RELATIVE WAI		layer_linear								

8. 单击 日志,查看执行训练任务过程中,用户代码输出到标准输出/标准错误的日志内容。

📕 说明:

为了运行训练任务应用,一般会自动创建多个服务容器,分别运行不同的程序分支。比如 ps 容器一般用来运行参数服务器代码,worker 容器一般用来运行模型计算代码,tensorboard 容器 用来运行 TensorBoard 训练监控。具体生成的服务 / 容器,可以单击 服务列表 或 容器列表 查看。)

可以按服务容器名称筛选某个容器,查看该容器运行程序输出到 stdout/stderr 的详细日志,比如 worker0,如图所示。也可以按时间和显示数量,过滤要查看的日志内容。同样,也可以选择下 载日志文件到本地。在本例中,如果查看 worker0 和 worker1 的日志,可以发现它们以数据并 行的方式,针对部分训练数据分别计算同样的模型,并通过 ps0 同步梯度更新。



这里提到的标准输出/标准错误日志,请区别于上述 Tensorboard 事件文件和 checkpoint 文件。

worker0 日志

服务列表 容器列表 路由列表 日志 事件
每个容器查看条目: 200条 \$ 按容器名称筛选: worker0 \$ 按日志起始时间筛选: 下载日志
<pre>worker0 2017-04-14T02:09:29.868822722Z 1492135769.378905: Worker 0: training step 3007 done (global step: 4756) worker0 2017-04-14T02:09:29.868828676Z 1492135769.387526: Worker 0: training step 3008 done (global step: 4758) worker0 2017-04-14T02:09:29.868828876Z 1492135769.396051: Worker 0: training step 3009 done (global step: 4759) worker0 2017-04-14T02:09:29.868831831Z 1492135769.406528: Worker 0: training step 3010 done (global step: 4750) worker0 2017-04-14T02:09:29.868834832Z Accuracy at local step 3011: 0.9641 worker0 2017-04-14T02:09:29.868837574Z 1492135769.498207: Worker 0: training step 3012 done (global step: 4767) worker0 2017-04-14T02:09:29.86884761Z 1492135769.508580: Worker 0: training step 3013 done (global step: 4767) worker0 2017-04-14T02:09:29.86884761Z 1492135769.517708: Worker 0: training step 3014 done (global step: 4769) worker0 2017-04-14T02:09:29.868849758Z 1492135769.527830: Worker 0: training step 3015 done (global step: 4771) worker0 2017-04-14T02:09:29.868849758Z 1492135769.536226: Worker 0: training step 3016 done (global step: 4773) worker0 2017-04-14T02:09:29.868849758Z 1492135769.536226: Worker 0: training step 3016 done (global step: 4773) worker0 2017-04-14T02:09:29.86885714Z 1492135769.5536226: Worker 0: training step 3016 done (global step: 4775) worker0 2017-04-14T02:09:29.86885714Z 1492135769.5536226: Worker 0: training step 3017 done (global step: 4775) worker0 2017-04-14T02:09:29.86885714Z 1492135769.553621: Worker 0: training step 3017 done (global step: 4775) worker0 2017-04-14T02:09:29.86885714Z 1492135769.553961: Worker 0: training step 3017 done (global step: 4776) worker0 2017-04-14T02:09:29.86885714Z 1492135769.553961: Worker 0: training step 3019 done (global step: 4776) worker0 2017-04-14T02:09:29.86885714Z 1492135769.552931: Worker 0: training step 3019 done (global step: 4776) worker0 2017-04-14T02:09:29.86885714Z 1492135769.552931: Worker 0: training step 3020 done (global step: 4778) worker0 2017-04-14T02:09:29.868</pre>
<pre>worker0 2017-04-14102:09:29.8688702382 Accuracy at local step 3021: 0.9615 worker0 2017-04-14T02:09:29.8688731722 1492135769.654884: Worker 0: training step 3022 done (global step: 4784) worker0 2017-04-14T02:09:29.8688751912 1492135769.664437: Worker 0: training step 3024 done (global step: 4786) worker0 2017-04-14T02:09:29.8688751912 1492135769.674020: Worker 0: training step 3025 done (global step: 4786) worker0 2017-04-14T02:09:29.8688821592 1492135769.674020: Worker 0: training step 3025 done (global step: 4786) worker0 2017-04-14T02:09:29.8688851532 1492135769.61998: Worker 0: training step 3025 done (global step: 4798) worker0 2017-04-14T02:09:29.8688851532 1492135769.700654: Worker 0: training step 3027 done (global step: 4790) worker0 2017-04-14T02:09:29.8688851532 1492135769.700654: Worker 0: training step 3027 done (global step: 4792) worker0 2017-04-14T02:09:29.86888949742 1492135769.718128: Worker 0: training step 3028 done (global step: 4794) worker0 2017-04-14T02:09:29.868890262 1492135769.718128: Worker 0: training step 3029 done (global step: 4795) worker0 2017-04-14T02:09:29.86889040782 Accuracy at local step 3031: 0.9566</pre>

worker1 日志

服务列表	容器列表	路由列表	日志	事件									
每个容器查看	f条目: 200	条 💠					按容	器名称	筛选:	worker1	按日志起	已始时间筛选:	下载日志
worker1	2017-04-	14T02:09:29	.0476053	16Z A	ccuracy at local step	2231: 0	0.9722						
worker1	2017-04-	14T02:09:29	.0476081	91Z 1	192135768.632527: Worl	ker 1: 1	training	step 2	232 d	one (glob	al step:	4671)	
worker1	2017-04-	14T02:09:29	.0476111	21Z 1	192135768.642804: Worl	ker 1: 1	training	step 2	233 d	one (glob	al step:	4672)	
worker1	2017-04-	14T02:09:29	0476141	.89Z 1	192135768.652608: Worl	ker 1: 1	training	step 2	234 d	one (glob	al step:	4673)	
worker1	2017-04-	14T02:09:29	0476171	.38Z 1	192135768.661336: Worl	ker 1: 1	training	step 2	235 d	one (glob	al step:	4674)	
worker1	2017-04-	14T02:09:29	0476200	99Z 1	192135768.670677: Worl	ker 1: 1	training	step 2	236 d	one (glob	l step:	4675)	
worker1	2017-04-	14T02:09:29	0476230	53Z 1	192135768.680458: Worl	ker 1: 1	training	step 2	237 d	one (glob	il step:	4676)	
worker1	2017-04-	14T02:09:29	0476260	04Z 1	192135768.690011: Worl	ker 1: 1	training	step 2	238 d	one (glob	al step:	4677)	
worker1	2017-04-	14T02:09:29	0476289	61Z 1	192135768.699700: Worl	ker 1: 1	training	step 2	239 d	one (glob	al step:	4678)	
worker1	2017-04-	14T02:09:29	0.0476318	70Z 1	192135768.709331: Worl	ker 1: 1	training	step 2	240 d	one (glob	il step:	4680)	
worker1	2017-04-	14T02:09:29	0476348	54Z A	ccuracy at local step	2241: 0	0.9738						
worker1	2017-04-	14T02:09:29	0.0476377	65Z 1	192135768.798582: Worl	ker 1: 1	training	step 2	242 d	one (glob	l step:	4689)	
worker1	2017-04-	14T02:09:29	0.0476407	27Z 1	192135768.808421: Worl	ker 1: 1	training	step 2	243 d	one (glob	il step:	4690)	
worker1	2017-04-	14T02:09:29	0476436	80Z 1	192135768.817869: Worl	ker 1: 1	training	step 2	244 d	one (glob	il step:	4691)	
worker1	2017-04-	14T02:09:29	0476466	66Z 1	192135768.827370: Worl	ker 1: 1	training	step 2	245 d	one (glob	il step:	4692)	
worker1	2017-04-	14T02:09:29	0.0476496	18Z 1	192135768.836649: Worl	(er 1: 1	training	step 2	2246 d	one (glob	il step:	4693)	
worker1	2017-04-	14T02:09:29	0.0476587	21Z 1	192135768.846184: Worl	(er 1: 1	training	step 2	2247 d	one (glob	il step:	4694)	
worker1	2017-04-	14102:09:29	0.0476618	59Z 1	192135768.856308: Worl	(er 1: 1	training	step 2	2248 d	one (glob	il step:	4695)	
worker1	2017-04-	14102:09:29	0476648	52Z 1	192135768.867560: Worl	(er 1: 1	training	step 2	2249 d	one (glob	il step:	4696)	
worker1	2017-04-	14102:09:29	0476678	23Z 1	192135768.878451: Worl	(er 1: 1	training	step 2	250 d	one (glob	il step:	4699)	
worker1	2017-04-	14102:09:29	0476708	59Z A	ccuracy at local step	2251: 6	0.9701	- + 7		(-1-h	1	(707)	
workeri	2017-04-	14102:09:29	0476737	462 1	192135768.967434: Worl	(er 1: 1	craining	step 2	252 0	one (glob	it step:	4707)	
worker1	2017-04-	14102:09:29	0.04/0/0/	322 1	192135768.978180: Worl	(er 1: 1	training	step 2	253 0	one (glob	it step:	4708)	
worker1	2017-04-	14102:09:29	0.0476026	232 1	192135768.988919; WOF	(er 1: 1	training	step 2	254 0	one (glob	it step:	4709)	
worker1	2017-04-	14102:09:29	0476826	90Z 1	192135768.998069: Worl	(er 1: 1	training	step 2	255 0	one (glob	il step:	4/10)	
worker1	2017-04-	14102:09:29	0476005	25Z 1	492135/09.00/3//: Worl	(er 1: 1	craining	step 2	250 0	one (glob	it step:	4711)	
worker1	2017-04-	14102:09:29	0476040	027 1	+92133/09.01/31/: WOrl	(er 1: 1	training	step 2	25/ 0	one (glob	it step:	4712)	
worker1	2017-04-	14102:09:29	0476940	017 1	+92133709.02/302: WOF	(er 1: 1	training	step 2	250 0	one (glob	it step:	4714)	
worker1	2017-04-	14102:09:29	0240971	507 1	192133709.037144: WOF	(er 1: 1	training	step 2	259 0	one (glob	it step:	4710)	
worker1	2017-04-	14102:09:30	0248931	.JOZ 1	+92133/09.04/080: WOF	2261. 0		step 2	200 0	one (glob)	it step:	4/10)	
worker1	201/-04-	14102109136	.0249203	1002 A	curacy at total step	2201: 6	0.9/02						

9. 单击 服务列表,如果看到所有 worker 服务的状态为"已停止",说明此次训练任务已经执行结

束。

服务列表	容器列表	路由列表	日志	事件				
服务名称	所属。	立用		服务状态	容器状态	镜像		操作
ps0	tf-trai	n-distributed		●就绪	就绪:1 停止:0	registry.cn-shenzhen.aliyuncs.com/tensorflow-sam	停止 重启	重新调度 变更配置 删除 事件
tensorboard0	tf-trai	n-distributed		●就绪	就绪:1 停止:0	registry.cn-shenzhen.aliyuncs.com/tensorflow-sam	停止 重启	重新调度 变更配置 删除 事件
tensorboard1	tf-trai	n-distributed	_	●就绪	就绪:1 停止:0	registry.cn-shenzhen.aliyuncs.com/tensorflow-sam	停止 重启	重新调度 变更配置 删除 事件
worker0	tf-trai	in-distributed		●巳停止	就绪:0 停止:1	registry.cn-shenzhen.aliyuncs.com/tensorflow-sam	启动 重启	重新调度 变更配置 删除 事件
worker1	tf-trai	n-distributed	Ŀ	●已停止	就绪:0 停止:1	registry.cn-shenzhen.aliyuncs.com/tensorflow-sam	启动 重启	重新调度 变更配置 删除 事件

10.通过 OSS 客户端您可以看到训练结果自动保存在了 OSS Bucket 中。

训练结束后,系统自动将所有保存在 Worker 容器本地 /output 路径中的文件,复制到用户给 定数据卷所对应的对象存储服务(OSS)Bucket 中。通过 OSS 客户端,可以看到本示例中用户 代码写入 /output 的事件文件和 checkpoint 文件,都已经备份到用户的 OSS Bucket 中了。

00	tensorflow-samples > training_logs			Q
上传	新建文件夹 下载 复制 删除 获取	7地址 设置HTTP头	粘贴 RAM授权	
文件名	大小	创建时间		深圳
	test	-	-	
	train	-	-	
	events.out.tfevents.1492135720.d19b802f8····	163.42KB	2017-04-14 10:09:50	
	graph.pbtxt	195.14KB	2017-04-14 10:09:33	
	model.ckpt-0.data-00000-of-00001	931.77KB	2017-04-14 10:09:33	
	model.ckpt-0.index	569B	2017-04-14 10:09:48	
	model.ckpt-0.meta	85.35KB	2017-04-14 10:09:48	

11.管理训练任务。

如果需要停止、重启、删除(比如为了释放计算资源给新的训练任务)某个训练任务,只需回到 应用>应用列表页面。找到对应的应用,在右侧进行操作即可。

1.5 预测服务

1.5.1 使用 TensorFlow Serving

利用阿里云弹性计算资源(ECS或EGS),负载均衡和对象存储服务,对于 TensorFlow 的导出模型,快速部署服务上线,用于其他应用的访问。

准备工作

在运行模型训练任务之前,请确认以下工作已经完成:

- 创建包含适当数量弹性计算资源 (ECS 或 EGS) 的容器集群。参见 创建容器集群。
- 使用相同账号创建对象存储服务(OSS)Bucket,使用它保存用于模型训练的数据。
- 为上面的容器集群创建数据卷,用于将 OSS Bucket 作为本地目录挂载到执行训练任务的容器
 内。创建方法请参考 创建数据卷。
- 了解 TensorFlow Serving 的基本概念和工作流程。参见 Serving a TensorFlow Model。

约定

为了简化您对于模型预测的使用,请注意以下约定。

• 在 OSS 的根目录以模型名创建文件夹。

• 在模型预测中,负载均衡的前、后端端口和应用端口必须保持一致。

操作步骤

1. 将预测模型保存在共享存储中。

可以通过在 OSS 客户端上传模型文件夹完成。

a. 在 OSS Bucket 的根目录以模型名创建文件夹,在本例子中使用 mnist。

• • •	OSS-1.1.9.0			
↔ OSS	Contensorflow-samples		Q	0
🕀 新建 🌣 设置 🗲 刷新	上传 新建文件夹 下载 复制 删除 获	取地址 设置HTTP头	RAM授权	
eml	●小技巧:使用Shift和Ctrl键(Mac下Command键)可以实现多选操作。	不再提示		
fast-style	文件名	大小	创建时间	
fast-style-model	mnist	-		
fast-style-transfer-input				
input-data				
into-billing-measure-data-p				
mywrfoutput				
neural-art	上传队列(5/5) 下载队列(5/5) 错误日志 最新资	н.	上传速度: - 下载速度: - >>	4
seowhitelist	暂停 取消 全部开始 全部暂停 全部取消	清空已完成		
sgp-intl-billing	mnist/2/	0	完成 🗙	¢
tensorflow-samples	mnist/2/saved_model.pb	85.44KB	完成 🗙	¢
tianyiweiqi-test	mnist/2/variables/	0	完成 🗙	¢
	mnist/2/variables/variables.data-00000-of-00001	931.77KB	完成 🗙	¢
POWERED BY 驻云科技	mnist/2/variables/variables.index	569B	完成 🗙	¢

b. 将带版本号的 Tensorflow Serving 模型文件夹上传到 mnist 下。

	OSS-1.1.9.0			
→ OSS	S ≥ tensorflow-samples → mnist			Q
🕀 新建 🌣 设置 🗲 刷新	上传 新建文件夹 下载 复制 删除 获	取地址 设置HTTP头	RAM授权	
eml	❶小技巧:使用Shift和Ctrl键(Mac下Command键)可以实现多选操作。	不再提示		
fast-style	文件名	大小	创建时间	
fast-style-model	1	-	-	
fast-style-transfer-input				
input-data				
into-billing-measure-data-p				
mywrfoutput			- 上生油座 · 下井油座 ·	×
neural-art	上传队列(5/5) 卜载队列(5/5) 错误日志 最新资	H	上17还反: - 「私还反: -	*
seowhitelist	暂停 取消 全部开始 全部暂停 全部取消	清空已完成		
sgp-intl-billing	mnist/2/	0	完成	×
tensorflow-samples	mnist/2/saved_model.pb	85.44KB	完成	×
tianyiweiqi-test	mnist/2/variables/	0	完成	×
	mnist/2/variables/variables.data-00000-of-00001	931.77KB	完成	×
POWERED BY 驻云科技	mnist/2/variables/variables.index	569B	完成	×

- 2. 配置负载均衡的监听端口。
 - a. 在负载均衡管理控制台(单击页面右上角的创建负载均衡)创建一个用于路由的负载均衡
 实例。

本示例中选择的是公网实例,您可以根据自己的需要选择公网或者私网。

道 说明:

由于负载均衡不支持跨地域(Region)部署,因此应选择与您所使用容器服务集群相同的地域。

b. 返回负载均衡管理控制台,将购买创建的负载均衡实例命名为 TensorFlow-serving。容器服务会通过该名称来引用这个负载均衡实例。

单击左侧导航栏中的 实例管理 > 选择实例所在的地域 > 选择所需实例 > 编辑实例的名称并单击 确定。

实例管理	亚太东南 1 (新加坡)	欧洲中部 1 (法兰克福)	华北1	美国东部 1 (弗吉尼亚) 华北2 华东2	2	
	亚太东北 1 (东京) :	华南1 香港 亚太东	南 2 (悉尼)	美国西部 1 (硅谷)	中东东部 1 (迪拜)	华北3 华东1	
负载均衡名和	你 \$		提	捜索 ●标签			
□ 负载均衡	ID/名称 可用区	服务地址(全部	▼ 状态	网络类型(全部) 🔻	端口/健康检查	后端服务器	标签
Dewz918 TensorFic	0660 华南 1 可用 ow-serving 华南 1 可用	区 B(主) 119.23.119.77 区 A(备) 公网	❷ 运行中	中 经典网络	TCP: 9000 I	iZwz9bcdf E常 c288914c0	zd5lol 未绑定 0cf6340
编辑句	5载均衡名称: orFlow-serving						
长度网	限制为1-80个字符,允许 定 取消	包含字母、数字、'-'、'/	、皇、皇这幽	些字符			

C. 创建监听端口。

单击实例右侧的 管理 > 单击左侧导航栏中的 监听 > 单击 添加监听 > 设置监听配置。协议为 TCP,端口映射为 8000:8000。

添加监听		\times
1.基本配置	2.健康检查配置 2.健康检查配置 3.配置成功	
前端协议[端口]:★	TCP ▼ : 8000 端口输入范围为1-65535。	
后端协议[端口]: 🛊	TCP : 8000 端口输入范围为1-65535。	
带宽峰值:	不限制 配置 使用流量计费方式的实例默认不限制带宽峰值;峰值输入范围1-1000M	
调度算法:	加权轮询	
使用服务器组: 🖉	0	
创建完毕自动启动监 听:	● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	
→ 展开高级配置		
	下一步取	消

3. 启动模型预测。

- a. 登录 容器服务管理控制台。
- b. 单击左侧导航栏中的 镜像与方案 > 解决方案。
- C. 在模型预测框中单击创建。

容器服务	机器学习		
概览			
应用			使空观测
服务	TensorBoard 监控训练过程。支持TensorFlow, Keras 等。	Keras等,及分布式训练方法。支持TensorBoard监控 训练过程。	Serving服务端。支持服务的负载均衡和理性伸缩。
集群	创建	创建	创建
节点			3
数据卷			
配置项			
▼ 镜像与方案 1			
镜像			
编排模板			
解决方案 2			

d. 设置模型预测任务的基本信息。

模型预测	
集群	EGS-cluster
应用名	tf-serving-dist
训练框架	tensorflow-serving:0.5.1
模型名	mnist
实例数量	1
单实例使用GPU数量	0
数据卷名	tfoss
负载均衡实例	TensorFlow-serving
负载均衡端口	8000
	负载均衡实例的前、后端监听都必须设置为该端口

- 集群:选择将要运行模型预测的集群。
- 应用名:为运行单机模型预测的应用名称。名称可以包含 1~64 个字符,包括数字、英文 字母和连字符(-)。
- 训练框架:选择用来进行模型训练的框架,包括 TensorFlow Serving 和自定义镜像。这里选择 TensorFlow Serving。
- 模型名称:这里的模型名称要与步骤1中创建的模型文件夹一致。
- 实例数量: TensorFlow Serving 实例的数量,这里不能超过集群中节点数。

- GPU数量:指定所使用的 GPU 数量,如果为 0 表示不使用 GPU (将使用 CPU 预测)。
- 数据卷名:指定为用于存储预测模型的对象存储服务(OSS)Bucket 在该集群中创建的数据卷的名称。数据卷的创建方法,请参考创建数据卷。
- 负载均衡实例:请选择步骤2负载均衡实例名称。
- 负载均衡端口:请将步骤2设置的端口号填入。
- 4. 设置完毕后,单击确定。
- 5. 在应用列表页面中,单击上述创建的应用。
- 6. 单击 路由列表,可以看到负载均衡提供的终端地址。这样您就可以利用 grpc 的客户端访问负载 均衡地址:负载均衡端口,在本例子中为 101.37.193.36:8000。

基本信息										
应用名称:tf-serving-dist 创建时间:2018-01-19 更新时间:2018-01-19 所在集群:EGS-duster										
触发器 1.4 目前没有任何創	專种类型的触发 触发器,点击右	諸只能创 i上角按钮(<mark>建1个</mark> ❼ 创建触发器	2				创建触发器	^	
服务列表	容器列表	日志	事件	路由列表						
路由地址(集群绑定5LB后路由地址才能被访问) 攝作										
101.37.193.3	101.37.193.36 设置服务权重									

1.6 在GPU 集群上实现Caffe模型训练

1.6.1 环境准备

目前,阿里云容器服务提供的深度学习解决方案内置了 Tensorflow、Keras、MXnet 框架的镜像,并支持基于它们的模型开发、训练和预测。同时,对于模型训练和预测,也可以通过指定自定 义容器镜像,来支持用户使用其他框架。

本文将描述如何通过自定义镜像的方式,实现使用 Caffe 框架在 GPU 容器集群上进行模型训练。 使用容器服务的深度学习解决方案,主要的工作包括:

1. 准备计算集群。

- 购买ECS计算资源,可以包括 CPU 和 GPU。
- 创建容器集群管理上述 ECS 节点。
- 2. 准备数据存储,用于保存和共享训练数据集、训练日志和结果模型。
 - 创建阿里云存储服务实例。
 - 为上述数据存储创建数据卷,用于将共享存储挂载入容器内部,方便训练、预测代码本地读写。

3. 在解决方案页面配置任务参数,启动模型训练任务。

前提须知

- 1. 准备一个 GPU 类型的容器集群。详情参见 创建容器集群。
- 2. 不同 ECS 服务地域提供的 GPU 实例类型可能不同。您需要提前确认。
- 请在与上述 ECS 节点的相同阿里云服务区域,创建 OSS 或 NAS 存储实例。否则,运行在 ECS上的容器将无法访问它们。
- 4. 首次使用需要进行角色授权,开通容器服务。请参见角色授权。

操作步骤

创建容器服务集群

- 1. 登录 容器服务控制台。
- 2. 创建 GPU 型的容器集群,本示例创建 GN5 型的容器集群。详情请参见 创建容器集群。
- 3. 在容器服务控制台查看容器集群的详情,本例中创建的是位于华东1的容器集群 swarmcluster。

容器服务		集群列表		您最多可以1	创建 5 个集群,每个集群最多可以涨	泰加 20 个节点 刷新	创建Swarm集群 🚽
Swarm	Kubernetes						
概览		常见问题: ③如何创建集群 ③如何	添加已有云服务器 🔗 跨可用区节点管理		站在接集群		
应用		名称 ▼					
服务		集群名称/ID	集群类型 地域 网络类型	集群状态 节点状态 🔮	节点个数 创建时间	Docker版本	損作
集群		swarmcluster	阿里云集群 华东1 虚拟专有网络	●运行中 健康 😂	1 2018-01-17 09:41:13	17.06.2-ce 管理	! · 查看日志 删除 监控 更多★ ·

创建共享数据存储

容器服务可以通过数据卷挂载的方式支持阿里云 OSS 对象存储和 NAS 文件存储。本示例选择 OSS 数据存储类型。

- 1. 登录 对象存储 OSS 管理控制台。
- 2. 在左侧导航栏,单击新建 Bucket。

对会存储	11 ruz 200						
7.5.80(1) (M	季则形耳			新手入门	开发者指南	SDK 下戴	更多 ▼
概览	安全令牌	安全扫描	内容安全				
我的访问路径 +	通过 RAM 和 STS 为子账号授 予临时的访问权限	可提供一键扫描 APP 漏洞和恶意代码服务	智能内容识别服务,快速识别违 规风险,如色情、暴恐、垃圾广	26 个 Bucket	t		
点击上方按钮添加您已授 权的 OSS 访问路径。			告等	新建 Buck	et		
存储空间 🕂 🕇 🕄	数据处理			0 条报警规则	, 0 已报警 , 0 预	谱状态 , 0 已禁用	1 0
٩	图片服务	视频点播	媒体转码	设置报警规	UQ5		
 apsaradoc 	支持在 OSS 端对图片文件进行	OSS 可配合视频点播(VOD)	OSS 可配合媒体转码服务,将				
 bptest-cjl 	编放、截路、水印等处理	服务,实现一站式自悦观层借	多採体叙描转的成多种终端描放 格式				

3. 进行 Bucket 参数配置,注意选择和容器集群一样的地域。 本例中在华东 1 区创建 OSS bucket caffe-bucket,可以查看其内外网访问地址。

caffe-bucket		举型 标准存储 区域 华东	1 创建时间 2018-01-16 16:20 删除 Bu	lucket							
概览 文件管理 基础设置 域名管理 图片处理 事件通	□ 函数计算 │ 基础数据 热点统计 API统计 文件访问统计										
基础数据											
① 总概范及 Bucket 概范基础政振都非实时数据,数据延迟 2-3 个小时。子账号若著不	到数据,需要主账号赋予云监控的权限。										
存錄用量 シ 本月外网流量	流入 V 本月请求次数	PUT V 文件数量	文件碎片 ②								
155 Byte 3.02	^B 43	6	0								
月同比 日环比 上月外网流量 08	te 上月请求次数 0										
访问继名											
	EndPoint 💿	访问城名 ⑦	HTTPS								
外网访问 ②	oss-cn-hangzhou.aliyuncs.com	caffe-bucket.oss-cn-hangzhou.aliyuncs.com	支持								
ECS 的经费网络访问(内网) ⑦	oss-cn-hangzhou-internal.aliyuncs.com	caffe-bucket.oss-cn-hangzhou-internal.aliyuncs.com	支持								
ECS 的 VPC 网络访问(内网) ②	oss-cn-hangzhou-internal.aliyuncs.com	caffe-bucket.oss-cn-hangzhou-internal.aliyuncs.com	支持								

您也可以创建 NAS 文件存储实例, NAS 文件实例创建过程需要两个步骤, 详见 创建文件系统 和 添加挂载点。

创建数据卷

创建好数据存储实例后,需要在容器集群中创建对应的数据卷。本例中使用 OSS 作为训练数据和 日志存储,可以创建 OSS 数据卷,详情参见 创建 OSSFS 数据卷。

- 1. 登录 容器服务管理控制台。
- 2. 单击左侧导航栏中的数据卷,再单击右上角 创建,开始创建数据卷。
- 3. 本例中创建 OSS 数据卷 caffe-volume,用于连接前一步创建的 OSS bucket caffe-bucket。

创建数据卷	×
数据卷类型:	● OSS ◎ NAS ◎ 云盘
数据卷名:	caffe-volume
AccessKey ID :	
AccessKey Secret :	
可选参数:	🗆 allow_other 📀 🔲 noxattr 📀
更多参数:	
	更多参数的填写格式可以 参考该文档
	注意:只有volume driver在0.7版本及以上的 集群才支持该参数,您可以到 "应用"列表中 找到 acsvolumedriver 应用,然后在其详情页 的服务列表中查看volumedriver服务的镜像版 本,如果是0.7以下的话,请升级系统服务
Bucket ID :	caffe-bucket 选择Bucket
访问域名:	◎ 内网域名 ◎ 外网域名 ◎ vpc域名 Ø
文件缓存:	◎ 打开 ⑧ 关闭 🛛 🕢
	创建取消

创建NAS数据卷的过程与OSS类似。参见 #unique_30。

1.6.2 构建和推送自定义 caffe 镜像

目前,深度学习解决方案还未内置对 caffe 框架的支持,但是训练框架支持自定义镜像,提供了灵活性。您可以通过指定自定义镜像的方式,使用您自己的 caffe 框架来进行模型训练。

您可以使用指定的 caffe 框架制作成 Docker 镜像,推送到容器镜像仓库中,如阿里云容器镜像仓库。以后在该集群部署的训练任务就可以使用这个 caffe 镜像。在开通容器服务的同时,也会开通 阿里云容器镜像仓库服务。

前提条件

- 需要准备一个容器镜像镜像仓库,来推送生成的镜像。可以使用与容器集群相同地域的阿里云容器镜像仓库,或者私有的容器镜像仓库。本例中使用阿里云容器镜像仓库服务。
- 需要一个可以构建 Docker 镜像的环境,且这个环境需要能 push 镜像到镜像仓库。本例中以前 面创建的 swarmcluster 容器集群的一个 ECS 节点作为构建 Docker 镜像的环境。

构建容器镜像仓库

- 1. 登录 容器镜像服务管理控制台。
- 2. 单击右上角 创建镜像仓库。进行相关参数配置,详情请参见 #unique_32 中关于创建镜像仓库的 内容。
- 本示例里,在华东1区创建镜像仓库 registry.cn-hangzhou.aliyuncs.com/dl-framework/acscaffe。

容器镜像服务	镜像仓库列表	华北1 华北2	华北 3 华东	1 华东 2	华南 1	亚太东北1(东京)	亚太东南1(新加坡)	亚太东南 2 (悉尼)	美国东部 1 (弗吉尼亚)	
镜像列表		美国四部 1 (姓谷)	欧洲中部 1	(法三宂福)						
命名空间管理									修改Registry登录密码	创建镜像仓库
▼ 镜像库	命名空间筛选:	全部 🔻								♀刷新
镜像搜索										
	仓库名称	命名空间	仓库状态	性质 も	风限仓	库地址			创建时间	操作
我的收藏	仓库名称 acs-caffe	命名空间 dl-framework	仓库状态 正常	性质相公有を	Q限 仓 管理 re	库地址 gistry.cn-hangzhou.ali	yuncs.com/dl-framewo	rk/acs-caffe	创建时间 2018-01-17 14:23:07	操作 管理 删除

在节点上构建 dockerfile 和脚本文件

选择容器集群中的一台 ECS 实例,登录到该节点上。在该节点上创建一个目录,用于存放将要创建的 dockerfile 文件和脚本文件。

创建dockerfile文件

您可以在集群中的一个 ECS 节点上创建 custom_train_caffe.dockerfile 文件。

登录到 ECS 节点上,依次执行以下命令,开始创建 dockerfile。

```
mkdir caffe-file
cd caffe-file
touch custom_train_caffe.dockerfile ## 创建 dockerfile 空文件
vim custom_train_caffe.dockerfile ##编辑 dockerfile 文件
```

然后输入如下的示例内容。

```
FROM bvlc/caffe:gpu
RUN apt-get update && apt-get install -y git vim
RUN mkdir /starter
COPY ./custom_train_helper.sh /starter
WORKDIR /starter
```

```
RUN chmod +x custom_train_helper.sh
ENTRYPOINT ["./custom_train_helper.sh"]
```

该镜像基于 caffe 官方基础镜像 bvlc/caffe:gpu,并使用一个自定义的脚本 custom_tra

in_helper.sh 作为用镜像启动容器时的入口进程。

保存后,成功创建 dockerfile 文件。

创建脚本文件

在相同目录下创建 custom_train_helper.sh 文件供 dockerfile 文件里构建镜像时使用, 创建脚本文

件与上面的方式类似,脚本的内容如下:

```
#!/bin/bash
default_output_path='/output'
default_remote_volume_path=$DEFAULT_REMOTE_VOLUME_PATH
default_input_path='/input'
# default input dir
if [ -d "$default_remote_volume_path" ] && [ ! -z $default_re
mote volume path ]; then
 ln -s $default_remote_volume_path $default_input_path
fi
# default output dir
if [ ! -d "$default_output_path" ]; then
 mkdir -p $default_output_path
fi
# exec user's command
eval "$@"
echo "Done training."
```

脚本逻辑主要是在执行具体训练命令的前后期,设置工作目录,和训练日志、结果的备份工作。

创建完毕后,执行下面的命令查看是否创建成功。

```
[root@xxx]# ls -l
-rw-r--r-- 1 root root 396 1月 15 11:22 custom_train_caffe.dockerfile
-rw-r--r-- 1 root root 952 1月 15 11:16 custom_train_helper.sh
```

构建镜像和推送镜像到镜像仓库中

构建镜像

在同级目录下构建自定义镜像。

```
docker build -t registry-vpc.cn-hangzhou.aliyuncs.com/dl-framework/acs
-caffe:gpu -f custom_train_caffe.dockerfile .
```

执行命令 docker images 查看是否成功构建镜像。

[root@xxx]#	docker	images			
REPOSITORY					
TAG		IMAGE	ID	CREATED	SIZE

```
registry-vpc.cn-hangzhou.aliyuncs.com/dl-framework/acs-caffe
gpu 7a2b58e3c5fb 39 seconds ago 3.49GB
```

将镜像推送到镜像仓库中

接下来将构建好的镜像 registry.cn-shanghai.aliyuncs.com/dl-framework/acs-caffe:gpu 推送到之前 在华东1区创建的镜像仓库中去。可以参考镜像基本操作。

首先登录到镜像仓库,如下:

[root@xxx]# docker login --username=user@aliyun.com registry-vpc.cnhangzhou.aliyuncs.com ##注意替换为用户登录账号 Password: ##注意密码是镜像仓库的独立登录密码 Login Succeeded

然后将镜像 push 到镜像仓库。

```
docker push registry-vpc.cn-hangzhou.aliyuncs.com/dl-framework/acs-
caffe:gpu
```

您可以在 容器镜像服务控制台 查看该镜像仓库。可以查看到刚刚推送的 caffe 镜像的公网、内网地 址。

<	acs-caffe
基本信息	部署应用 こ 別所
构建 仓库授权	注意:摄像域名地址已经变更,需要重新进行docker login. 另外您可以通过访问私网镜像仓库地址的域名,未加快您镜像下载速度并减少公网流量开锅,如果您的机器在VPC网络中,请访问VPC网络地址
webhook	基本信息 ^
镜像版本	● 報酬を応答: acs-caffe 報酬性质: 公开 公司地址: docker pull registry.cn-hangzhou.aliyuncs.com/dl-framework/acs-caffe 经患内网: docker pull registry.internal.cn-hangzhou.aliyuncs.com/dl-framework/acs-caffe VPC网络: docker pull registry.vpc.cn-hangzhou.aliyuncs.com/dl-framework/acs-caffe 代码合库: https://code.aliyun.com/dl-framework/acs-caffe 镜像地域: 华东 1

您可以单击镜像版本,查看镜像的基本信息。

<	acs-c	affe				
基本信息	镜像版	本				こ 刷新
仓库授权	版本	Image ID 😰	Digest 💿	镜像大小 😰	最后更新时间	操作
webhook	gpu	7a2b58a3c5fb	4w35b06083c8909b4be5b109w88f0307e2b05e097ww2cfd65w3ecwewee70f50d	1.755G	2018-01-17 15:55:11	安全扫描 层信息
79rd 158r0.0X -4->					共有1条 ,每页显示:20条	« < <u>1</u> > »

1.7 利用 TFRecord 和 HDFS 准备 TensorFlow 训练数据

数据准备和预处理在一个深度学习训练过程中扮演着非常重要的角色,它影响着模型训练的速度和 质量。 而 TensorFlow 对于 HDFS 的支持,将大数据与深度学习相集成,完善了从数据准备到模型训练的 完整链条。在阿里云容器服务深度学习解决方案中,为 TensoFlow 提供了 OSS、NAS 和 HDFS 三 种分布式存储后端的支持。

本文介绍如何将数据转化为 TFRecord 格式,并且将生成的 TFRecord 文件保存到 HDFS 中,本示 例使用阿里云 Elastic MapReduce (E-MapReduce)的 HDFS 服务。

为什么要使用 TFRecord

TFRecord 是 TensorFlow 内定的统一标准数据格式,可以支持多线程数据读取,并且可以通过 batch size 和 epoch 参数来控制训练时单次 batch 的大小和样本文件迭次数,同时能更好的利用内 存和方便数据的复制和移动,所以是利用 TensorFlow 进行大规模深度学习训练的首选。

步骤 1 创建 E-MapReduce 集群

E-MapReduce 是运行在阿里云平台上的一种大数据处理的系统解决方案。更多详细信息参见 *E-MapReduce* 概述。

登录 *E-MapReduce* 管理控制台 创建一个 E-MapReduce 集群。有关创建过程,参见 创建 *E-MapReduce* 集群。

本示例创建了一个位于华南1地域的集群,且网络类型为 VPC。

🔅 myEMR 🔹 返回集群列表	调整规模 释放
集群信息	^
ID/名称 C-E8C	付费类型 按量付费
地域 cn-shenzhen	当前状态 空闲
开始时间 2017/05/24 16:14:27	运行时间 8分16秒
日志功能 打开	日志路径 oss://testvolume
软件配置	引导操作/软件配置 正常
高可用 否	ECS应用角色 AliyunEmrEcsDefaultRole
软件信息	^
产品版本 EMR-3.0.2	集群类型 HADOOP
软件信息 hive 2.0.1, phoenix 4.7.0, nginx 1.10.2, spark 2.0.2, ganglia 3.7.2, tez 0.8.4, hue 3.11.0, zeppelin 0.	6.2, sqoop 1.4.6, yam 2.7.2, pig 0.14.0
网络信息	^
网络类型 专有网络	选择安全组 emr-default-securitygroup(sg-wz96oh4)mjkdfiitya13)
可用区 cn-shenzhen-b	专有网络/交换机 vpc-wz96ybdsjjior29djdf4h / vsw-wz96z71x9k43b1z64mw8d

步骤2创建容器集群,并且打通两个集群间的网络

1. 登录 容器服务管理控制台, 在同一个 VPC 下创建 GPU 容器集群。

集醇列表					您	最多可以创建	5 个集群,每个集群最多可	以添加 20 个节点	子账号授权	刷新	创建集群
小助手: 如何创建集群 如何添加已有云服	资器 跨可用[区节点管理	集成日志服务 通过Docke	客户端连接集	Ħ						
名称 ▼											
集群名称/ID	集群类型	地域	网络类型	集群状态	节点状态 🖉	节点个数	创建时间	Docker版本			攝作
ElasticGPUService c7e8e1064cdf2480888551710e43a0c11	阿里云集群	华南1	虚拟专有网络 vpc-wz96ybdsjjior29djdf4h	● 就绪	健康 ℃	1	2017-05-09 09:42:16	17.03.1-ce	管理	查看日調	志 删除 控 更多▼

- 2. 登录 ECS 管理控制台 将容器服务集群的节点加入到 E-MapReduce 集群对应的安全组。
 - a. 选择安全组所在的地域(本示例为华南1),选择安全组并单击右侧的管理实例。

安全组列表 华北 1	华北 2 华北 3	华东 1 华东 2	华南1 香港	亚太东北 1 (东京)	亚太东南 1 (新加坡)	亚太东南 2 (悉尼)	美国东部 1 (弗吉尼亚)		c	创建安全组
美国西	部 1 (硅谷) 中东东	辰部 1 (迪拜) 欧洲	中部 1 (法兰克福)						
安全组ID ▼ 輸入	、安全组ID精确查询,	, 多个用" , "隔开	搜索	》 标签						<u>×</u> ?
□ 安全组ID/名称		所属专有网络		相关实例	网络类型 创建时	间	描述	标签		操作
sg-wz96oh4jmjkdf emr-default-securi	filtya13 tygr	vpc-wz96ybdsjji	ior29djdf4h	2	专有网络 2017-0	05-24 16:14:28	-		修改 売隆 管理实例	还原规则 配置规则

b. 单击 添加实例,选择容器集群中的节点并单击 确定。

添加实例)	\times
*实例ID:	i-wz9eqt798ezd7xar6qjp	•	
		确定 取消	

步骤 3 生成 TFRecord 数据

本示例利用模型训练服务提供运行环境执行 *convert_to_records.py*, 生成 **TFRecord** 数据, 保存到 HDFS 中。

- 1. 登录 容器服务管理控制台。
- 2. 单击左侧导航栏中的 镜像与方案 > 解决方案。
- 3. 在模型训练中单击创建。

机器学习

し模型开发	1 模型训练	良 模型预测
使用 Jupyter 快速开发,调试撞型代码,并使用 TensorBoard 监控训练过程。支持TensorFlow, Keras等。	使用CPU,GPU集群训练模型。支持TensorFlow, Keras 等,及分布式训练方法。支持TensorBoard监控训练过 程。	使用CPU, GPU进行模型预测。支持TensorFlow Serving服务端。支持服务的负载均衡和弹性伸缩。
创建 指南	创建 指南	创建 指南

4. 填写模型训练的配置信息并单击确定。

本示例的具体配置如下所示:

- 训练框架:tensorflow:1.0.0。
- 单Worker使用GPU数量:0。
- 数据卷名:不使用数据卷。
- Git地址: https://code.aliyun.com/deeplearning/mnist-examples.git。
- 执行命令:

模型训练		
集群	ElasticGPUService 🔻	
应用名	prepare-data	
训练框架	tensorflow:1.0.0	
	□ 分布式训练	
单Worker使用GPU数量	0	
数据举名	不使用教授業	
	1.00.09096	
Git地址	https://code.aliyun.com/deeplearning/mnist-example	
执行命令	python convert_to_records.pydirectory hdfs://192.168.100.206:9000/mnist-tfrecord	
		1.
	□ 训练监控	
	- WINDER	
	确定	

运行成功后,可以查看执行的日志,显示 TFRecord 文件已经保存到了 HDFS。

服务列表	容器列表	路由列表	日志	事件		
每个容器查看务	※目: 100条	\$			按容器名称筛选: prepare-data- \$ 按日志起始时间筛选: 下载日表	5
prepare-d it	lata–12_worl	ker_1 201	17-05-23	T11:33:0	02.718478793Z Cloning training code from https://code.aliyun.com/deeplearning/mnist-examples.	g
prepare-d	lata-12_worl	ker_1 201	17-05-23	T11:33:0	02.720364084Z Cloning into 'mnist-examples'	
prepare-d	lata-12 worl	ker 1 201	17-05-23	T11:33:0	07.340480568Z Done cloning code.	
prepare-d	lata-12 worl	ker 1 201	17-05-23	T11:33:0	07.340598411Z Run training code under /starter/mnist-examples as: python convert to records.py	,
direct	orv hdfs:/	/192.168.10	00.206:9	000/mni	st-tfrecord	
prepare-d	ata-12 worl	ker 1 201	17-05-23	T11:34:0	05.508679080Z Extracting MNIST data/train-images-idx3-ubvte.gz	
prepare-d	ata-12 wor	ker 1 201	17-05-23	T11:34:0	05.508706069Z Extracting MNIST data/train-labels-idx1-ubyte.gz	
prepare-d	lata-12 worl	ker 1 201	17-05-23	T11:34:0	05.5087108567 Extracting MNIST data/t10k-images-idx3-ubyte.gz	
prepare-d	lata-12 worl	ker 1 201	17-05-23	T11:34:0	05.508714238Z Extracting MNIST data/t10k-labels-idx1-ubyte.gz	
prepare-d	lata-12 worl	ker 1 201	17-05-23	T11:34:0	05.5087175902 Writing hdfs://192.168.100.206:9000/mnist-tfrecord/train.tfrecords	
prepare-d	lata-12 worl	ker 1 201	17-05-23	T11:34:0	05.5087208747 Writing hdfs://192.168.100.206:9000/mpist-tfrecord/validation.tfrecords	
prepare-d	lata_12_worl	ker 1 201	17-05-23	T11.34.0	05 500/2017 Writing hdfs://102 168 100 206:0000/mnist_tfrerord/test tfrerords	
prepare-d	lata_12_worl	ker 1 201	17-05-23	T11.34.0		
prepare-d	lata_12_worl	kor 1 201	17_05_22	T11.24.0	05.577279272 Done running craining couch	
prepare-d	lata 12_worl	ker_1 201	17 05 23	T11.34.0	05.57431324007 Date persisting checkpoints to rende starses	
prepare-d	ara-12_WOL	vel_t 201	17-05-23	111:34:0	03.3743234302 Done beistsithd checkhothis in iemote storage.	

登录到 E-MapReduce 机器上查看产生的 TFRecord 文件。

```
# hdfs dfs -ls /mnist-tfrecord
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/opt/apps/hadoop-2.7.2/share
/hadoop/common/lib/slf4j-log4j12-1.7.10.jar!/org/slf4j/impl/
StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/opt/apps/tez-0.8.4/lib/slf4j-
log4j12-1.7.10.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an
explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
Found 3 items
-rw-r--r-- 3 root hadoop
                              8910000 2017-05-23 19:34 /mnist-
tfrecord/test.tfrecords
                             49005000 2017-05-23 19:33 /mnist-
-rw-r--r-- 3 root hadoop
tfrecord/train.tfrecords
                              4455000 2017-05-23 19:33 /mnist-
-rw-r--r-- 3 root hadoop
tfrecord/validation.tfrecords
```