ModelArts

开发环境

文档版本	01
发布日期	2025-01-02

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 开发环境介绍

软件开发的历史，就是一部降低开发者成本，提升开发体验的历史。在AI开发阶段，ModelArts也致力于提升AI开发体验，降低开发门槛。ModelArts开发环境，以云原生的资源使用和开发工具链的集成，目标为不同类型AI开发、探索、教学用户，提供更好云化AI开发体验。

ModelArts Notebook云上云下，无缝协同

- 代码开发与调测。云化JupyterLab使用，本地IDE+ModelArts插件远程开发能力，贴近开发人员使用习惯
- 云上开发环境，包含AI计算资源，云上存储，预置AI引擎
- 运行环境自定义，将开发环境直接保存成为镜像，供训练、推理使用

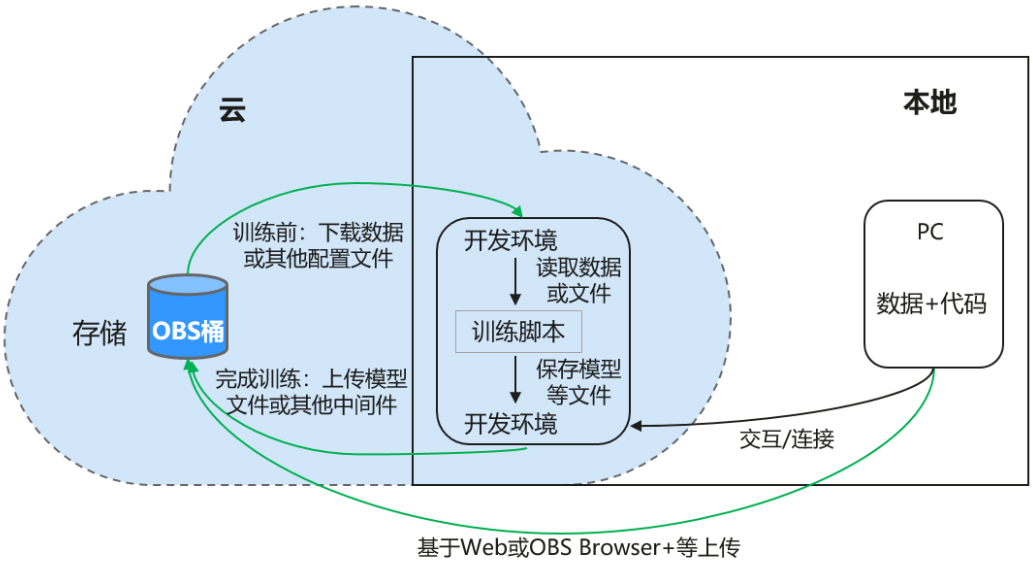
亮点特性 1：远程开发 - 支持本地 IDE 远程访问 Notebook

Notebook提供了远程开发功能，通过开启SSH连接，用户本地IDE可以远程连接到ModelArts的Notebook开发环境中，调试和运行代码。

对于使用本地IDE的开发者，由于本地资源限制，运行和调试环境大多使用团队公共搭建的服务器，并且是多人共用，这带来一定的环境搭建和维护成本。

而ModelArts的Notebook的优势是即开即用，它预先装好了不同的AI引擎，并且提供了非常多的可选规格，用户可以独占一个容器环境，不受其他人的干扰。只需简单配置，用户即可通过本地IDE连接到该环境进行运行和调试。

图 1-1 本地 IDE 远程访问 Notebook 开发环境



ModelArts的Notebook可以视作是本地PC的延伸，均视作本地开发环境，其读取数据、训练、保存文件等操作与常规的本地训练一致。

对于习惯使用本地IDE的开发者，使用远程开发方式，不影响用户的编码习惯，并且可以方便快捷地使用云上的Notebook开发环境。

本地IDE当前支持VS Code、PyCharm、SSH工具。还有专门的插件PyCharm Toolkit和VS Code Toolkit，方便将云上资源作为本地的一个扩展。

亮点特性 2：开发环境保存 - 支持一键镜像保存

ModelArts的Notebook提供了镜像保存功能。支持一键将运行中的Notebook实例保存为镜像，将准备好的环境保存下来，可以作为自定义镜像，方便后续使用，并且方便进行分享。

保存镜像时，安装的依赖包（pip包）不丢失，VS Code远程开发场景下，在Server端安装的插件不丢失。

亮点特性 3：预置镜像 - 即开即用，优化配置，支持主流 AI 引擎

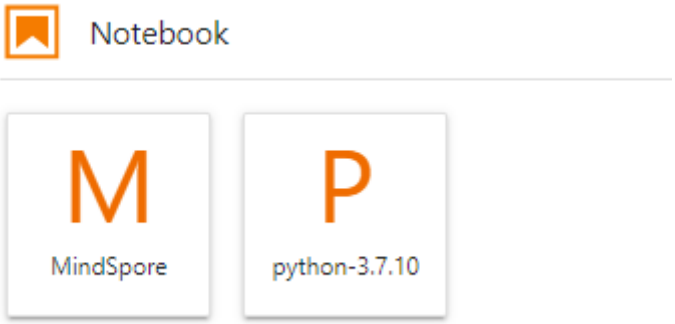
每个镜像预置的AI引擎和版本是固定的，在创建Notebook实例时明确AI引擎和版本，包括适配的芯片。

ModelArts开发环境给用户提供了一组预置镜像，主要包括PyTorch、TensorFlow、MindSpore系列。用户可以直接使用预置镜像启动Notebook实例，在实例中开发完成后，直接提交到ModelArts训练作业进行训练，而不需要做适配。

ModelArts开发环境提供的预置镜像版本是依据用户反馈和版本稳定性决定的。当用户的功能开发基于ModelArts提供的版本能够满足的时候，建议用户使用预置镜像，这些镜像经过充分的功能验证，并且已经预置了很多常用的安装包，用户无需花费过多的时间来配置环境即可使用。

ModelArts开发环境提供的预置镜像主要包含：

- 常用预置包，基于标准的Conda环境，预置了常用的AI引擎，例如PyTorch、MindSpore；常用的数据分析软件包，例如Pandas、Numpy等；常用的工具软件，例如cuda、cudnn等，满足AI开发常用需求。
- 预置Conda环境：每个预置镜像都会创建一个相对应的Conda环境和一个基础Conda环境python（不包含任何AI引擎），如预置MindSpore所对应的Conda环境如下：



  用户可以根据是否使用AI引擎参与功能调试，选择不同的Conda环境。
- Notebook：是一款Web应用，能够使用户在界面编写代码，并且将代码、数学方程和可视化内容组合到一个文档中。
- JupyterLab插件：插件包括规格切换，分享案例到AI Gallery进行交流，停止实例等，提升用户体验。
- 支持SSH远程连接功能，通过SSH连接启动实例，在本地调试就可以操作实例，方便调试。
- ModelArts开发环境提供的预置镜像支持功能开发后，直接提到ModelArts训练作业中进行训练。

说明

- 为了简化操作，ModelArts的Notebook，同一个Notebook实例中支持不同引擎之间的切换。
- 不同Region支持的AI引擎不一样，请以控制台实际界面为准。

亮点特性 4：提供在线的交互式开发调试工具 JupyterLab

ModelArts集成了基于开源的JupyterLab，可为您提供在线的交互式开发调试。您无需关注安装配置，在ModelArts管理控制台直接使用Notebook，编写和调测模型训练代码，然后基于该代码进行模型的训练。

JupyterLab是一个交互式的开发环境，是Jupyter Notebook的下一代产品，可以使用它编写Notebook、操作终端、编辑MarkDown文本、打开交互模式、查看csv文件及图片等功能。

2 使用场景

ModelArts提供灵活开放的开发环境，您可以根据实际情况选择。

- ModelArts提供了云化版本的Notebook，无需关注安装配置，即开即用，具体参见**JupyterLab简介及常用操作**。
- ModelArts也提供了本地IDE的方式开发模型，通过开启SSH连接，用户本地IDE可以远程连接到ModelArts的Notebook开发环境中，调试和运行代码。本地IDE方式不影响用户的编码习惯，并且可以方便快捷地使用云上的Notebook开发环境。

  本地IDE当前支持VS Code、PyCharm、SSH工具。PyCharm和VS Code还分别有专门的插件PyCharm Toolkit、VS Code Toolkit，让远程连接操作更便捷。具体参见**VS Code ToolKit连接Notebook**。

3 管理 Notebook 实例

3.1 创建 Notebook 实例

在开始进行模型开发前，您需要创建Notebook实例，并打开Notebook进行编码。

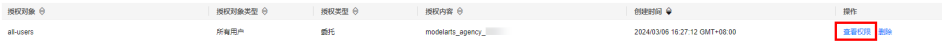
背景信息

- Notebook使用涉及到计费，具体收费项如下：
  - 处于“运行中”状态的Notebook，会消耗资源，产生费用。根据您选择的资源不同，收费标准不同，价格详情请参见**产品价格详情**。当您不需要使用Notebook时，建议停止Notebook，避免产生不必要的费用。
  - 创建Notebook时，如果选择使用云硬盘EVS存储配置，云硬盘EVS会一直收费，建议及时停止并删除Notebook，避免产品不必要的费用。
- 在创建Notebook时，默认会开启自动停止功能，在指定时间内停止运行Notebook，避免资源浪费。
- 只有处于“运行中”状态的Notebook，才可以执行打开、停止操作。
- 一个账户最多创建10个Notebook。

创建 Notebook 实例

1. 登录ModelArts管理控制台，在左侧导航栏中选择“权限管理”，检查是否配置了访问授权。如果未配置，请先配置访问授权。参考**使用委托授权**完成操作。

   图 3-1 查看委托配置信息

2. 登录ModelArts管理控制台，在左侧导航栏中选择“开发空间 > Notebook”，进入“Notebook”页面。
3. 单击右上角“创建”，进入“创建Notebook”页面，请参见如下说明填写参数。
   a. 填写Notebook基本信息，包含名称、描述、是否自动停止，详细参数请参见**表3-1**。

图 3-2 Notebook 基本信息

表 3-1 基本信息的参数描述

参数名称	说明
“名称”	Notebook的名称。系统会自动生成一个名称，您可以根据业务需求重新命名，命名规则如下：只能包含数字、大小写字母、下划线和中划线，长度不能超过128位且不能为空。
“描述”	对Notebook的简要描述。
“自动停止”	默认开启，且默认值为“1小时”，表示该Notebook实例将在运行1小时之后自动停止，即1小时后停止规格资源计费。可选择“1小时”、“2小时”、“4小时”、“6小时”或“自定义”几种模式。选择“自定义”模式时，可指定1~24小时范围内任意整数。 ● 定时停止：开启定时停止功能后，该Notebook实例将在运行时长超出您所选择的时长后，自动停止。 **说明** 出于对用户任务进度的保护，在您设置的自动停止时间到达后，Notebook不会立即自动停止，可能会有2-5分钟的延迟，方便您进行续约。

b. 填写Notebook详细参数，如镜像、资源规格等，详细参数请参见**表3-2**。

表 3-2 Notebook 实例的详细参数说明

参数名称	说明
“镜像”	支持公共镜像和自定义镜像。 ● 公共镜像：即预置在ModelArts内部的AI引擎。 ● 自定义镜像：可以将基于公共镜像创建的实例保存下来，作为自定义镜像使用。自定义镜像的介绍及制作请参考**在Notebook中使用自定义镜像**。 一个镜像对应支持一种AI引擎，创建Notebook实例时选择好了对应AI引擎的镜像。用户可以根据需要选择镜像。在右侧搜索框中输入镜像名称关键字，可快速查找镜像。 Notebook运行停止后，可以在同一个Notebook实例中变更镜像。

参数名称	说明
“资源类型”	支持公共资源池和专属资源池。 “公共资源池”无需单独购买，即开即用，按需付费，即按您的Notebook实例运行时长进行收费。 “专属资源池”按实际情况选择已创建的专属资源池。如果没有专属资源，需要单独购买并创建。 **说明** 如果您购买的专属池是单节点的Tnt004规格：GPU: 1*tnt004 | CPU: 8核 32GiB (modelarts.vm.gpu._tnt004u8)，使用该集群创建Notebook实例时，Tnt004卡空闲但是规格显示售罄或者创建失败显示资源不足的情况，请联系技术支撑。
“类型”	芯片类型包括CPU、GPU和ASCEND类型。 不同的镜像支持的芯片类型不同，根据实际需要选择。 GPU性能更佳，但是相对CPU而言，费用更高。
“规格”	根据选择的芯片类型不同，可选资源规格也不同。请根据界面实际情况和需要选择。 ● CPU规格 “2核8GB”： Intel CPU通用规格，用于快速数据探索和实验 “8核32GB”： Intel CPU算力增强型，适用于密集计算场景下运算 ● GPU规格 “GPU: 1*Vnt1(32GB)|CPU: 8 核 64GB”： GPU单卡规格，32GB显存，适合深度学习场景下的算法训练和调测 “GPU: 1*Tnt004(16GB)|CPU: 8核* 32GB”： GPU单卡规格，16GB显存，推理计算最佳选择，覆盖场景包括计算机视觉、视频处理、NLP等 “GPU: 1*Pnt1(16GB)|CPU: 8核 64GB”： GPU单卡规格，16GB显存，适合深度学习场景下的算法训练和调测

参数名称	说明
“存储配置”	包括“云硬盘EVS”、“弹性文件服务SFS”、“对象存储服务OBS”和“并行文件系统PFS”。请根据界面实际情况和需要选择。 **说明** “对象存储服务OBS”、“并行文件系统PFS”是白名单功能，如果有试用需求，请提工单申请权限。 ● 选择“云硬盘EVS”作为存储位置。 根据实际使用量设置磁盘规格。磁盘规格默认5GB。磁盘规格的最大值请以实际界面显示为准。 从Notebook实例创建成功开始，直至实例删除成功，磁盘每GB按照规定费用收费。 ● 选择“弹性文件服务SFS”作为存储位置。 仅专属资源池支持，并需要在专属资源池对应的网络打通VPC才能生效，具体操作请参见**ModelArts网络**。 **说明** 如果需要设置SFS Turbo的文件夹权限，请参考**权限管理**文档配置。 – “弹性文件服务”：选择已创建的SFS Turbo（在**弹性文件服务控制台**创建SFS Turbo）。 – “云上挂载路径”：默认为/home/ma-user/work/。 – “子目录挂载”：选择SFS Turbo的存储位置。 – “挂载方式”：当用户配置了文件夹控制权限，则显示此参数。根据SFS Turbo存储位置的权限显示“读写”或“只读”。 ● 选择“对象存储服务OBS”或“并行文件系统PFS”作为存储位置。 选择“存储位置”：设置用于存储Notebook数据的OBS路径。如果想直接使用已有的文件或数据，可将数据提前上传至对应的OBS路径下。 选择“凭据”：选择已有的凭据或单击右侧的“立即创建”，跳转至数据加密控制台**创建凭据**，凭据键/值填写用户的AK、SK信息（“键”分别填写“accessKeyId”，“secretAccessKey”；“值”在控制台个人账号下“我的凭证>访问密钥”获取AK、SK）。 **图 3-3** 设置凭据值  “云硬盘EVS”、“弹性文件服务SFS”的存储路径挂载在/home/ma-user/work目录下。 Notebook实例运行中，可以通过**动态挂载OBS并行文件系统**操作来增加数据存储路径。

参数名称	说明
	停止或重启Notebook实例时，存储的内容会被保留，不丢失。 删除Notebook实例时，EVS存储会一起释放，存储的内容不保留。SFS可以重新挂载到新的Notebook，可以保留数据。
“扩展存储配置”	**说明** “扩展存储配置”功能是白名单功能，如果有试用需求，请提工单申请权限。 如果有多个数据存储路径，可以单击“增加扩展存储配置”，增加用户指定的存储挂载目录。支持增加的存储类型有“存储桶OBS”、“并行文件系统PFS”、“弹性文件服务SFS”。 约束限制： ● 每种存储类型最多支持挂载5个。 ● 扩展存储挂载目录不允许重复，不允许挂载到黑名单目录，允许嵌套挂载。不允许挂载的黑名单目录为以下前缀匹配的目录：/data/、/cache/、/dev/、/etc/、/bin/、/lib/、/sbin/、/modelarts/、/train-worker1-log/、/var/、/resource_info/、/usr/、/sys/、/run/、/tmp/、/infer/、/opt/ 添加扩展存储后，可进入Notebook实例详情页，单击“存储配置 > 扩展存储”，查看或编辑扩展存储信息。在存储个数未达到最大个数时，也可在右侧单击“添加扩展存储”。
“SSH远程开发”	● 开启此功能后，用户可以在本地开发环境中远程接入Notebook实例的开发环境。 ● 实例在停止状态时，用户可以在Notebook详情页中更新SSH的配置信息。 **说明** 开启此功能的实例中会预置VS Code插件（python、jupyter等）以及VS Code Server包，会占用约1G左右的持久化存储空间。
“密钥对”	开启“SSH远程开发”功能后，需要设置此参数。 可以选择已有密钥对。 也可以单击密钥对右侧的“立即创建”，跳转到数据加密控制台，在“密钥对管理 > 账号密钥对”页面，单击“创建密钥对”。 创建完Notebook后，可以在Notebook详情页中修改密钥对。 **注意** 创建好的密钥对，请下载并妥善保存，使用本地IDE远程连接云上Notebook开发环境时，需要用到密钥对进行鉴权认证。

参数名称	说明
“远程访问白名单”	可选，开启“SSH远程开发”功能后，可以设置此参数。 设置为允许远程接入访问这个Notebook的IP地址（例如本地PC的IP地址或者访问机器的外网IP地址，最多配置5个，用英文逗号隔开），不设置则表示无接入IP地址限制。 如果用户使用的访问机器和ModelArts服务的网络有隔离，则访问机器的外网地址需要在主流搜索引擎中搜索“IP地址查询”获取，而不是使用**ipconfig**或**ifconfig/ip**命令在本地查询。 **图 3-4** 查询外网 IP 地址  创建完Notebook后，可以在Notebook详情页中修改白名单IP地址。

c. 可选：添加Notebook标签。在标签栏输入“标签键”和“标签值”，单击“添加”。

表 3-3 添加标签

参数名	参数说明
“标签”	ModelArts支持对接标签管理服务TMS，在ModelArts中创建资源消耗性任务（例如：创建Notebook、训练作业、推理在线服务）时，可以为这些任务配置标签，通过标签实现资源的多维分组管理。 标签详细用法请参见**ModelArts如何通过标签实现资源分组管理**。 添加标签后，可在Notebook实例详情页查看标签内容，也可进行修改、删除标签。

说明

可以在标签输入框下拉选择TMS预定义标签，也可以自己输入自定义标签。预定义标签对所有支持标签功能的服务资源可见。租户自定义标签只对自己服务可见。

4. 参数填写完成后，单击“立即创建”进行规格确认。

5. 参数确认无误后，单击“提交”，完成Notebook的创建操作。

   进入Notebook列表，正在创建中的Notebook状态为“创建中”，创建过程需要几分钟，请耐心等待。当Notebook状态变为“运行中”时，表示Notebook已创建并启动完成。

6. 在Notebook列表，单击实例名称，进入实例详情页，查看Notebook实例配置信息。

   “SSH远程开发”功能开启时，在“白名单”右侧单击修改，可以修改允许远程访问的白名单IP地址。实例在停止状态时，在“认证”右侧单击修改，用户可以更新密钥对。

   单击“存储配置”页签的“添加数据存储”，可以挂载OBS并行文件系统，方便读取数据，具体操作参见**动态挂载OBS并行文件系统**。

   如果存储使用的是云硬盘EVS，单击存储容量右侧的“扩容”，可以动态扩充云硬盘EVS的容量，具体操作参见**动态扩充云硬盘EVS容量**。

3.2 打开 Notebook 实例

针对创建好的Notebook实例（即状态为“运行中”的实例），可以打开Notebook并在开发环境中启动编码。

基于不同AI引擎创建的Notebook实例，打开方式不一样。

- 本地IDE使用PyCharm/VS Code/SSH工具，远程连接访问，具体参见**VS Code ToolKit连接Notebook**。
- 在线JupyterLab访问，具体参见**JupyterLab简介及常用操作**。

创建实例，持久化存储挂载路径为/home/ma-user/work目录。

```
sh-4.4$pwd
/home/ma-user
sh-4.4$cd work/
sh-4.4$pwd
/home/ma-user/work
sh-4.4$
```

存放在work目录的内容，在实例停止、重新启动后依然保留，其他目录下的内容不会保留，使用开发环境时建议将需要持久化的数据放在/home/ma-user/work目录。

3.3 查找/启动/停止/删除实例

查找实例

Notebook页面展示了所有创建的实例。如果需要查找特定的实例，可根据筛选条件快速查找。单击搜索框，可按照属性类型选择单个条件来筛选，或者同时选择多个筛选条件筛选。

图 3-5 查找实例

- 打开“查看所有”开关，可以看到IAM项目下所有子用户创建的Notebook实例。
- 按实例名称、实例ID、实例状态、使用的镜像、实例规格、实例描述、创建时间等单个筛选或组合筛选。

表格显示设置

单击设置按钮可自定义设置需要显示在表格中的列。

图 3-6 设置表格列显示

启动/停止实例

由于运行中的Notebook将一直耗费资源，您可以通过停止操作，停止资源消耗。对于停止状态的Notebook，可通过启动操作重新使用Notebook。

1. 登录ModelArts管理控制台，在左侧菜单栏中选择“开发空间 > Notebook”，进入Notebook管理页面。
2. 执行如下操作启动或停止Notebook。
   - **启动Notebook**：单击“操作”列的“启动”。只有处于“停止”状态的Notebook可以执行启动操作。
   - **停止Notebook**：单击“操作”列的“停止”。只有处于“运行中”状态的Notebook可以执行停止操作。

注意

Notebook停止后：

- /home/ma-user/work目录下的数据会保存，其余目录下内容会被清理。例如：用户在开发环境中的其他目录下安装的外部依赖包等，在Notebook停止后会被清理。
- Notebook实例将停止计费，但如有EVS盘挂载，存储部分仍会继续计费。

删除实例

针对不再使用的Notebook实例，可以删除以释放资源。

1. 登录ModelArts管理控制台，在左侧菜单栏中选择“开发空间>Notebook”，进入Notebook页面。
2. 在Notebook列表中，单击操作列的“删除”，在弹出的确认对话框中，确认信息无误，然后输入“DELETE”，单击“确定”，完成删除操作。

⚠ 注意

Notebook删除后不可恢复，请谨慎操作。实例删除后，挂载目录下的数据也将一并删除，请谨慎操作。

3.4 变更 Notebook 实例镜像

ModelArts允许用户在同一个Notebook实例中切换镜像，方便用户灵活调整实例的AI引擎。

约束限制

Notebook实例状态必须在“停止”中。

变更实例镜像操作

1. 登录ModelArts管理控制台，在左侧菜单栏中选择“开发空间 > Notebook”，进入Notebook页面。
2. 在Notebook列表，单击某个Notebook实例操作栏的“更多 > 变更镜像”，在变更镜像窗口选择新的镜像，单击“确定”。
3. 在镜像窗口选择新的镜像，单击“确定”，变更成功后，在Notebook列表页的镜像栏，可以查看到变更后的镜像。

3.5 变更 Notebook 实例运行规格

ModelArts允许用户在同一个Notebook实例中切换节点运行规格，方便用户灵活调整规格资源。

约束限制

只有处于“停止”、“运行中”和“启动失败”的Notebook实例才能变更规格。

变更实例规格操作

1. 登录ModelArts管理控制台，在左侧菜单栏中选择“开发空间 > Notebook”，进入Notebook页面。
2. 在Notebook列表，单击某个Notebook实例操作列的“更多 > 变更规格”，在弹出的“变更规格”对话框中选择对应规格资源。

图 3-7 变更规格

图 3-8 选择规格

说明

规格切换需要该规格所在的集群有其他规格才可以执行，当前上线的部分规格所在集群无其他规格，切换的时候会显示为空，所以不可进行切换。

3.6 开发环境中如何选择存储

不同存储的实现原理都不同，在性能、易用性、成本的权衡中可以有不同的选择，没有一个存储可以覆盖所有场景，了解下云上开发环境中各种存储使用场景说明，更能提高使用效率。

说明

仅支持挂载同一区域下的OBS并行文件系统（PFS）和OBS对象存储。

表 3-4 云上开发环境中各种存储使用场景说明

存储类型	建议使用场景	优点	缺点
云硬盘EVS	比较适合只在开发环境中做数据、算法探索，性能较好。	块存储SSD，可以理解为一个磁盘，整体IO性能比NFS要好，可以动态扩充，最大可以到4096GB。 云硬盘EVS作为持久化存储挂载在/home/ma-user/work目录下，该目录下的内容在实例停止后会被保留，存储支持在线按需扩容。	缺点是只能在单个开发环境中使用 。
并行文件系统PFS	**说明** 并行文件系统PFS为白名单功能，如需使用，请联系华为技术支持开通。 适合直接使用PFS桶作为持久化存储进行AI开发和探索场景。 1. 数据集的存储。将数据集直接挂载到Notebook进行浏览和数据处理，在训练时直接使用。或在实例运行后，将承载数据集的OBS并行文件系统动态挂载至Notebook中，详细操作请参考**动态挂载OBS并行文件系统**。 2. 代码的存储。在Notebook调测完成，可以直接指定对应的对象存储路径作为启动训练的代码路径，方便临时修改。 3. 训练观测。可以将训练日志等输出路径进行挂载，在Notebook中实时查看和观测，特别是利用TensorBoard，Notebook功能完成对训练输出的分析。	PFS是一种经过优化的高性能对象存储文件系统，存储成本低，吞吐量大，能够快速处理高性能计算（HPC）工作负载。在需要使用对象存储服务场景下，推荐使用PFS挂载。 **说明** 建议上传时按照128MB或者64MB打包或者切分，使用时边下载边解压后在本地存储读取，以获取更好的读写与吞吐性能。	缺点是小文件频繁读写性能较差，例如直接作为存储用于模型重型训练，大文件解压等场景慎用。 **说明** PFS挂载需要用户对当前桶授权给ModelArts完整读写权限，Notebook删除后，此权限策略不会被删除。

存储类型	建议使用场景	优点	缺点
对象存储服务OBS	**说明** OBS对象存储为白名单功能，如需使用，请联系华为技术支持开通。 在开发环境中做大规模的数据上传下载时，可以通过OBS桶做中转。	存储成本低，吞吐量大，但是小文件读写较弱。建议上传时按照128MB或者64MB打包或者切分，使用时边下载边解压后在本地读取。	对象存储语义，和Posix语义有区别，需要进一步理解。
弹性文件服务SFS	目前只支持在专属资源池中使用；针对探索、实验等非正式生产场景，建议使用这种。开发环境和训练环境可以同时挂载一块SFS存储，省去了每次训练作业下载数据的要求，一般来说重IO读写模型，超过32卡的大规模训练不适合。	实现为NFS，可以在多个开发环境、开发环境和训练之间共享，如果不需要重型分布式训练作业，特别是启动训练作业时，不需要额外再对数据进行下载，这种存储便利性可以作为首选。	缺点性能比EVS云硬盘块存储低。
本地存储	重型训练任务首选	运行所在虚拟机或者裸金属机器上自带的SSD高性能存储，文件读写的吞吐量大，建议对于重型训练任务先将数据准备到对应目录再启动训练。 默认在容器/cache目录下进行挂载，/cache目录可用空间请参考**开发环境中不同Notebook规格资源“/cache”目录的大小**。	缺点是存储生命周期和容器生命周期绑定，每次训练都要下载数据。

如何使用

1. 在开发环境中如何使用云硬盘EVS块存储?

   例如，在**创建Notebook实例**时选择云硬盘EVS存储小容量，Notebook运行过程中如果发现存储容量不够，可以扩容，请参考**动态扩充云硬盘EVS容量**。

2. 在开发环境中如何使用OBS并行文件系统?

   例如，在Notebook中训练时，可直接使用挂载至Notebook容器中的数据集，在运行过程中可以**动态挂载OBS并行文件系统**。

3.7 动态挂载 OBS 并行文件系统

什么是动态挂载 OBS 并行文件系统

并行文件系统（Parallel File System）是对象存储服务（Object Storage Service，OBS）提供的一种经过优化的高性能文件系统，详细介绍可以参见**并行文件系统**。

在ModelArts运行态的Notebook容器中，采用动态挂载特性，将OBS对象存储模拟成本地文件系统。其本质是通过挂载工具，将对象协议转为POSIX文件协议。挂载后应用层可以在容器中正常操作OBS对象。

动态挂载适用于哪些使用场景

场景1：数据集预览和操作，将承载数据集的OBS挂载至Notebook中，可以像本地文件系统一样操作数据集。

场景2：在Notebook中训练时，可直接使用挂载至Notebook容器中的数据集。

动态挂载 OBS 并行文件系统有什么限制

OBS提供两种桶，对象存储（对象桶）和并行文件系统PFS。

ModelArts的Notebook仅支持挂载OBS的**并行文件系统**，挂载至Notebook容器“/data/”的子目录下。

动态挂载 OBS 并行文件系统操作

方式1：通过ModelArts控制台操作

1. 登录ModelArts管理控制台，在左侧导航栏中选择“开发空间 > Notebook”，进入“Notebook”页面。
2. 选择运行中的Notebook实例，单击实例名称，进入Notebook实例详情页面，在“存储配置”页签，单击“添加数据存储”，设置挂载参数。
   a. 设置本地挂载目录，在“/data/”目录下输入一个文件夹名称，例如：demo。挂载时，后台自动会在Notebook容器的“/data/”目录下创建该文件夹，用来挂载OBS文件系统。
   b. 选择存放OBS并行文件系统下的文件夹，单击“确定”。

图 3-9 动态挂载 OBS 并行文件系统

3. 挂载成功后，可以在Notebook实例详情页查看到挂载结果。

图 3-10 挂载成功

3.8 动态扩充云硬盘 EVS 容量

什么是动态扩容 EVS

存储配置采用云硬盘EVS的Notebook实例， 存储盘是挂载至容器/home/ma-user/work/目录下， 可以在实例运行中的状态下，动态扩充存储盘容量，单次最大动态扩容100GB。

动态扩容 EVS 适用于哪些使用场景

在Notebook开发过程中，初期存储使用量较小时， 创建Notebook可以选择小容量EVS， 比如5G大小； 开发完成后，需要大规模数据集训练，此时再将存储容量扩容至当前阶段所需容量，可以节约成本。

动态扩容 EVS 有什么限制

- Notebook实例的存储配置采用的是云硬盘EVS。

  图 3-11 创建 Notebook 实例时选择云硬盘 EVS 存储

- 单次最大可以扩容100GB，扩容后的总容量不超过4096GB。
- 云硬盘EVS存储容量最大支持4096GB，达到4096GB时，不允许再扩容。
- 实例停止后，扩容后的容量仍然有效。计费也是按照扩容后的云硬盘EVS容量进行计费。
- 云硬盘EVS只要使用就会计费，请在停止Notebook实例后，确认不使用就及时删除数据，释放资源，避免产生费用。

动态扩容 EVS 操作

1. 登录ModelArts管理控制台，在左侧导航栏中选择“开发空间 > Notebook”，进入“Notebook”页面。
2. 选择运行中的Notebook实例，单击实例名称，进入Notebook实例详情页面，单击“扩容”。

图 3-12 Notebook 实例详情页

3. 设置待扩充的存储容量大小，单击“确定”。系统显示“扩容中”，扩容成功后，可以看到扩容后的存储容量。

图 3-13 扩容

图 3-14 扩容中

3.9 修改 Notebook SSH 远程连接配置

ModelArts允许用户在Notebook实例中更改SSH配置信息。

在创建Notebook实例时，如果未配置SSH远程连接，当用户需要开启远程连接时，则可以在Notebook的实例详情页打开SSH的配置信息开关；

在创建Notebook实例时，如果设置了允许远程连接Notebook的白名单IP地址，当用户需要更换一个IP地址远程连接Notebook实例时，则可以在Notebook的实例详情页修改白名单IP地址，也可更换密钥对。

约束限制

Notebook实例状态必须在“停止”中。

修改密钥对和远程连接 IP 地址

1. 登录ModelArts管理控制台，在左侧菜单栏中选择“开发空间 > Notebook”，进入Notebook页面。
2. 在Notebook列表，进入Notebook实例详情页。打开SSH远程开发开关，更新密钥对和白名单。

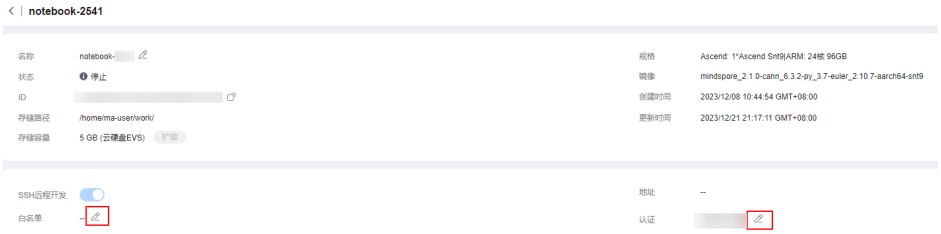
   说明

   “远程SSH开发”开关可以手动打开的场景，请打开远程SSH开发开关，参考图3-15操作。SSH配置信息更新后，“远程SSH开发”开关打开后不可关闭。

   “所选镜像必须配置SSH远程开发”的场景，请参考图3-16操作。

图 3-15 更新 SSH 配置信息

图 3-16 修改白名单和密钥对



- 密钥对可单击 ▼ 选择已有的密钥对或“立即创建”创建新的密钥对。

- 白名单IP地址的设置请参考**设置远程连接IP地址**。修改远程连接的可访问IP地址后， 原来已经建立的链接依然有效， 当链接关闭后失效；新打开建立的链接只允许当前设置的IP进行访问。

设置远程连接 IP 地址

图 3-17 设置远程连接 IP 地址

此处的IP地址，请填写外网IP地址。如果用户使用的访问机器和华为云ModelArts服务的网络有隔离，则访问机器的外网地址需要在主流搜索引擎中搜索“IP地址查询”获取，而不是使用ipconfig或ifconfig/ip命令在本地查询。

图 3-18 查询外网 IP 地址

3.10 查看所有子账号的 Notebook 实例

当子用户被授予“listAllNotebooks”和“listUsers”权限时，在Notebook页面上，单击“查看所有”，可以看到IAM项目下所有子用户创建的Notebook实例 。

说明

配置该权限后，可以查看子用户的Notebook，也可以在Notebook中访问子用户的OBS、SWR等。

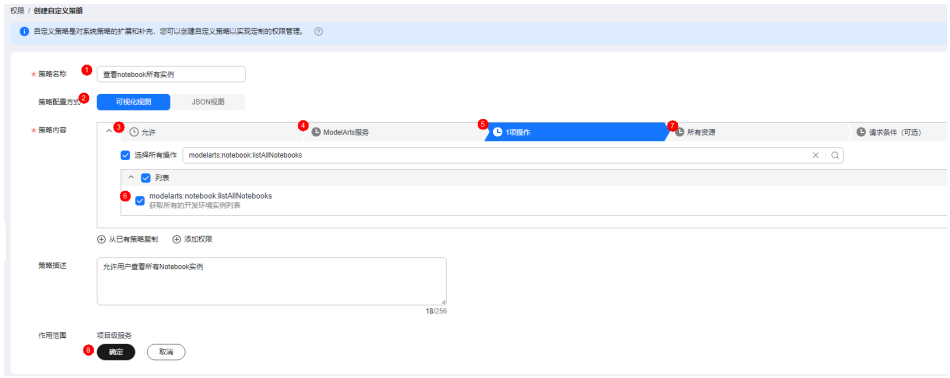
给子账号配置查看所有 Notebook 实例的权限

1. 使用主用户账号登录ModelArts管理控制台，单击右上角用户名，在下拉框中选择“统一身份认证”，进入统一身份认证（IAM）服务。
2. 在统一身份认证服务页面的左侧导航选择“权限管理 > 权限”，单击右上角的“创建自定义策略”，需要设置两条策略。

   策略1：设置查看Notebook所有实例，如图3-19所示，单击“确定”。

   - “策略名称”：设置自定义策略名称，例如：查看Notebook所有实例。
   - “策略配置方式”：选择可视化视图。
   - “策略内容”：允许，云服务中搜索ModelArts服务并选中，操作列中搜索关键词modelarts:notebook:listAllNotebooks并选中，所有资源选择默认值。

   图 3-19 创建自定义策略

   策略2：设置查看Notebook实例创建者信息的策略。

   - “策略名称”：设置自定义策略名称，例如：查看所有子用户信息。
   - “策略配置方式”：选择可视化视图。
   - “策略内容”：允许，云服务中搜索IAM服务并选中，操作列中搜索关键词iam:users:listUsers并选中，所有资源选择默认值。
3. 在统一身份认证服务页面的左侧导航选择“用户组”，在用户组页面查找待授权的用户组名称，在右侧的操作列单击“授权”，勾选步骤2创建的两条自定义策略，单击“下一步”，选择授权范围方案，单击“确定”。

   此时，该用户组下的所有用户均有权限查看该用户组内成员创建的所有Notebook实例。

   如果没有用户组，也可以创建一个新的用户组，并通过“用户组管理”功能添加用户，并配置授权。如果指定的子用户没有在用户组中，也可以通过“用户组管理”功能增加用户。

子用户启动其他用户的 SSH 实例

子用户可以看到所有用户的Notebook实例后，如果要通过SSH方式远程连接其他用户的Notebook实例，需要将SSH密钥对更新成自己的，否则会报错ModelArts.6786。更新密钥对具体操作请参见**修改Notebook SSH远程连接配置**。

具体的错误信息提示：ModelArts.6789: 在ECS密钥对管理中找不到指定的ssh密钥对xxx，请更新密钥对并重试。

3.11 查看 Notebook 实例事件

在Notebook的整个生命周期，包括实例的创建、启动、停止、规格变更等关键操作以及实例的运行状态等在后台都有记录，用户可以在Notebook实例详情页中查看具体的事件，通过实例的事件，从而看到实例的运行或者异常等状态详情。在右侧可以手动刷新事件，也可以设置间隔30秒，1分钟，5分钟自动刷新事件。

图 3-20 查看 Notebook 实例事件并设置自动刷新

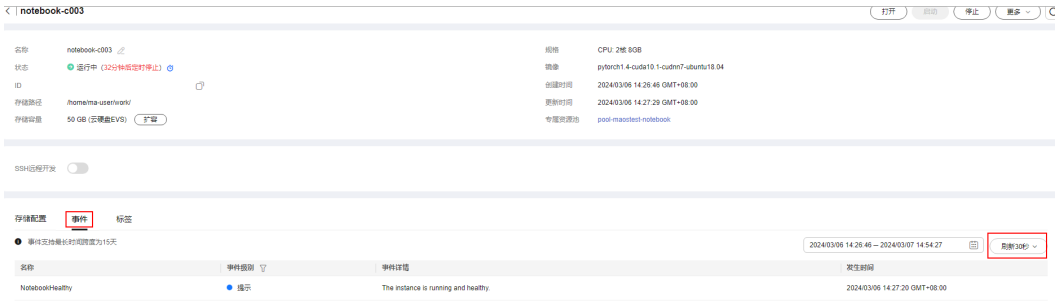


表 3-5 实例创建过程的事件列表

事件名称	事件描述	事件级别
Scheduled	实例被调度成功	提示
PullingImage	正在拉取镜像	提示
PulledImage	镜像拉取完毕	提示
NotebookHealthy	实例运行中，处于健康状态	重要
CreateNotebookFailed	创建实例失败	紧急
PullImageFailed	镜像拉取失败	紧急
FailedCreate	Failed to create notebook container. Please contact SRE to check node {node_name}	紧急
CreateContainerError	Failed to create container. Please contact SRE to check node {node_name}	紧急
FailedAttachVolume	Failed to attach volume. Please contact SRE to check node {node_name}	重要

事件名称	事件描述	事件级别
MountVolumeFailed	Mount volume failed; Check whether the DEW secret is correct if the instance cannot change to running in five minutes	紧急
	Mount volume failed; Check if vpc of sfs-turbo is interconnected if the instance cannot change to running in five minutes	紧急
	Mount volume failed; Please contact SRE to check node {node_name} if the instance cannot change to running in five minutes	紧急

表 3-6 实例启动过程的事件列表

事件名称	事件描述	事件级别
EmptyDirExceeded	Usage of empty-dir volume exceeds its limit. A new container will be scheduled and created automatically soon.	紧急
NodeResourcePressure	Insufficient node resources. A new container will be scheduled and created automatically soon.	紧急
EphemeralStorageExceeded	Local ephemeral storage exceeds its limit. A new container will be scheduled and created automatically soon.	紧急
FailedToStartContainer	Failed to start container. Please contact SRE to check node {node_name}	紧急
Scheduled	实例被调度成功	提示
PullingImage	正在拉取镜像	提示
PulledImage	镜像拉取完毕	提示
NotebookHealthy	实例运行中，处于健康状态	重要
RunHookScript	运行自定义脚本	提示
StartNotebookFailed	实例启动失败	紧急

事件名称	事件描述	事件级别
PullImageFailed	镜像拉取失败	紧急
CreateKernelFailed	conda命令不可用导致创建jupyter kernel失败 （The jupyter launcher page does not contain the kernel due to conda environment issues, please ensure that {conda_env} is available and the command: {conda_cmdt} env list can be run properly）	重要
	权限问题导致创建jupyter kernel失败 （The jupyter launcher page does not contain the kernel due to permission issues, please ensure that the uid {ma_uid} have write permissions to {conda_path}）	重要
ConfigurationError	conda命令不可用导致配置modelarts sdk和ma-cli路径到conda env失败 （The modelarts sdk and cli is unavailable in the conda envs due to conda environment issues, please ensure that the {conda_env} is available and the command: {conda_cmd} env list can be run properly）	重要
	权限问题导致配置modelarts sdk和ma-cli路径到conda env失败 （The modelarts sdk and cli is unavailable in the conda env due to permission issues,please ensure that the uid {ma_uid} have write permissions to {conda_path}）	重要
FailedToPullImageReason	Failed to pull image. Please make sure the image exists in SWR repo, otherwise contact SRE to check node {node_name}	重要
	Failed to pull image. Please contact SRE to check node {node_name} **说明** {node_name}表示节点名称，为可变变量，一般为IP形式，如：192.168.1.1	

表 3-7 实例停止过程的事件列表

事件名称	事件描述	事件级别
StopNotebook	实例停止	重要
StopNotebookResourceIdle	实例因资源空闲即将自动停止或实例因资源空闲自动停止	重要

表 3-8 更新实例过程的事件列表

事件名称	事件描述	事件级别
UpdateName	更新实例名称	提示
UpdateDescription	更新实例描述	提示
UpdateFlavor	更新实例规格	重要
UpdateImage	更新实例镜像	重要
UpdateStorageSize	实例存储正在扩容 （User %s is updating storage size from %sGB to %sGB）	重要
	实例扩容完成 （User %s updated storage size successfully）	重要
UpdateKeyPair	配置实例密钥对 （User %s updated the instance keypair to "{%s}"）	重要
	更新实例密钥对 （User %s updated the instance keypair from %s to %s）	重要
UpdateWhitelist	更新实例访问白名单	重要
UpdateHook	更新自定义脚本	重要
UpdateStorageSizeFailed	资源售罄引起的实例存储扩容失败 （The EVS disk is sold out）	紧急
	内部错误引起的实例扩容失败 （The EVS disk size updated failed. Operations and maintenance personnel are handling the problem）	紧急

表 3-9 镜像保存过程中的事件列表

事件名称	事件描述	事件级别
SaveImage	保存镜像成功	重要

事件名称	事件描述	事件级别
SavedImageFailed	D进程引起的保存镜像失败 （There are processes in 'D' status, please check process status using 'ps -aux' and kill all the 'D' status processes）	紧急
	镜像大小引起的保存镜像失败 （Container size %dG is greater than threshold %dG）	紧急
	层数限制引起的保存镜像失败 （Too many layers in your image）	紧急
	任务超时引起的保存镜像失败 （Operations personnel are handling the problem）	紧急
	SWR故障引起的保存镜像失败 （Failed to save the image because the SWR service is faulty）	紧急

表 3-10 实例运行过程的事件列表

事件名称	事件描述	事件级别
NotebookUnhealthy	实例处于不健康状态	紧急
OutOfMemory	实例被OOM掉了	紧急
JupyterProcessKilled	jupyter进程被killed掉了	紧急
CacheVolumeExceedQuota	/cache目录文件大小超过最大限制	紧急
NotebookHealthy	实例从不健康恢复到了健康状态	重要
EVSSoldOut	EVS存储售罄	紧急

表 3-11 OBS 动态挂载产生的事件列表

事件名称	事件描述	事件级别
DynamicMountStorage	挂载OBS存储	重要
DynamicUnmountStorage	卸载OBS存储	重要

表 3-12 用户侧触发的事件

事件名称	事件描述	事件级别
RefreshCredentialsFailed	用户鉴权失败	紧急

3.12 Notebook cache 盘告警上报

创建Notebook时，可以根据业务数据量的大小选择CPU、GPU或者Ascend资源，对GPU或Ascend类型的资源，ModelArts会挂载硬盘至“/cache”目录，用户可以使用此目录来储存临时文件。

当前开发环境的cache盘使用时，没有容量告警，在使用时很容易超过限制，并直接重启Notebook实例。重启后多种配置重置，会导致用户数据丢弃，环境丢失，造成很不好的使用体验。因此需要提供cache盘使用情况的监控和告警，并将数据上报至AOM平台。

配置流程

1. 填写告警基本信息
2. **设置告警规则**
   a. 监控对象指标配置
   b. 告警触发条件设置
3. **告警通知设置**
   a. 创建主题、设置主题策略、订阅主题
   b. 创建告警行动规则
   c. 选择已创建的行动规则

告警上报配置方法

1. 登录AOM控制台。
2. 单击“告警 > 告警规则”，在“告警规则”界面，单击“添加告警”。
3. 填写告警基本信息。

4. 设置告警规则。

   “规则类型”选择“阈值规则”。

   “监控对象”：选择“选择资源对象”。单击选择资源对象，弹出新窗口。

   - 添加方式：选择“按指标维度添加”。
   - 指标名称：选择“全量指标”，搜索需要监控的cache指标名称然后选中。例如：ma_container_notebook_cache_dir_size_bytes（cache目录的总大小）、ma_container_notebook_cache_dir_util（cache目录的利用率）
   - 指标维度：根据实际需求选择相应的指标维度。例如service_id:xxx，然后单击“确定”。

   监控对象设置完成后，选择“统计方式”和“统计周期”。

   “告警条件设置”：触发条件根据实际需求设置。

图 3-21 监控对象指标设置

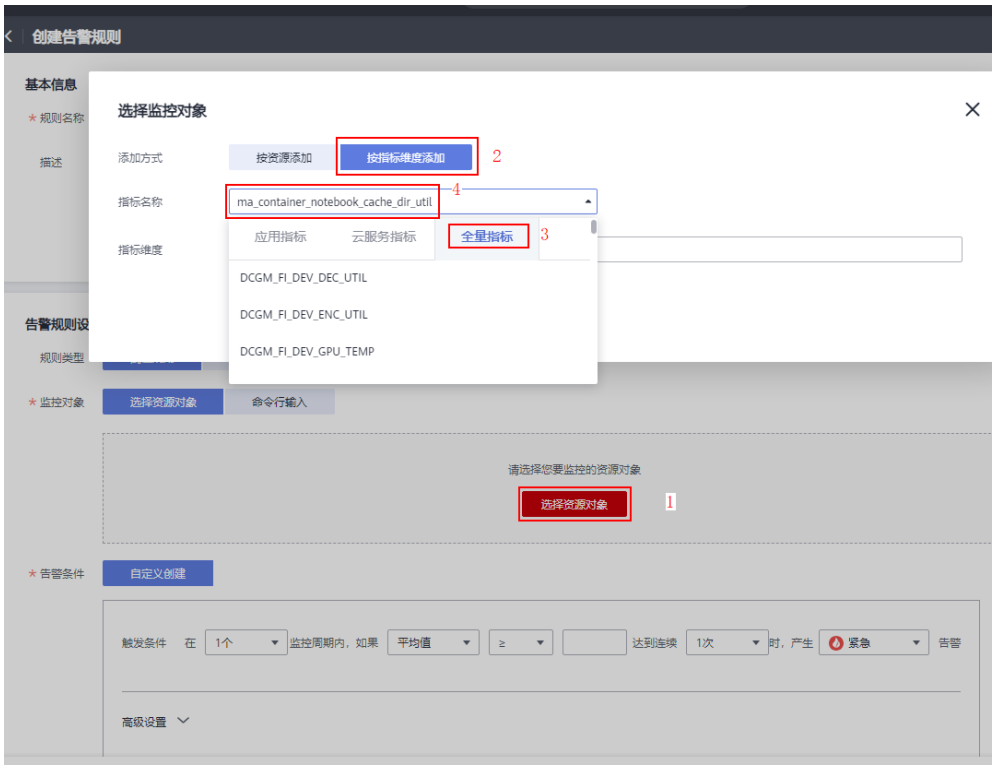


图 3-22 设置指标统计方式

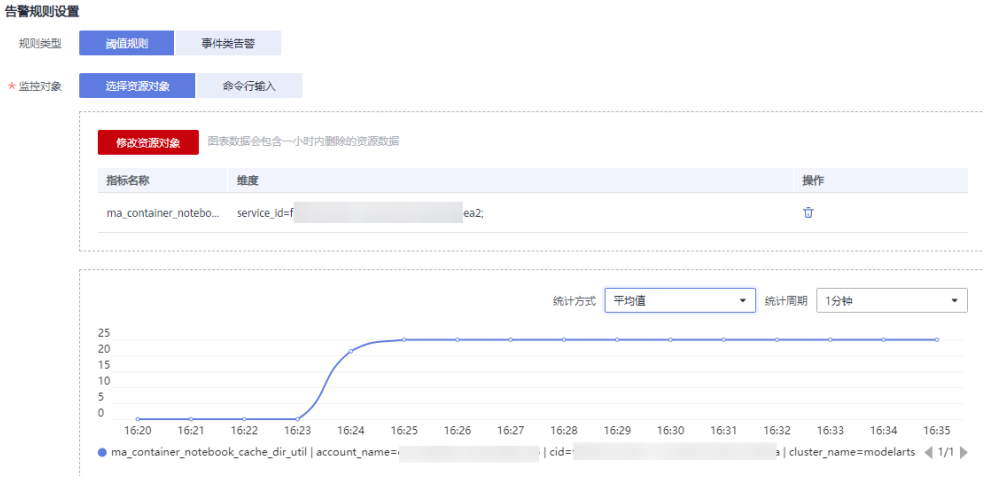


图 3-23 告警条件设置

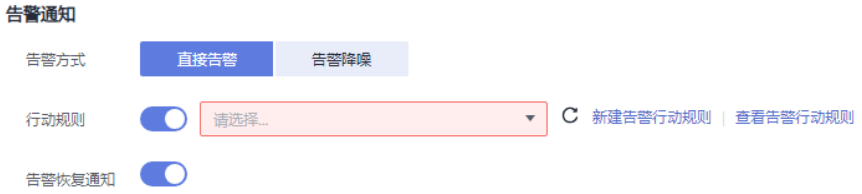
5. 设置告警通知，单击“立即创建”。

   “告警方式”：选择“直接告警”

   “行动规则”：开启开关，选择已创建的行动规则。如果现有列表中的告警行动规则无法满足需要，可单击“新建告警行动规则”添加，详细操作请参考**创建告警行动规则**。

   “告警恢复通知”：开启开关

图 3-24 设置告警通知



先在SMN创建一个主题，用于配置告警通知规则。

- **创建主题**

  i. 进入“消息通知服务”控制台，单击“主题管理 > 主题”，进入“主题”页面。

ii. 单击“创建主题”填写主题名称，选择企业项目后，单击确定即可创建一个主题。

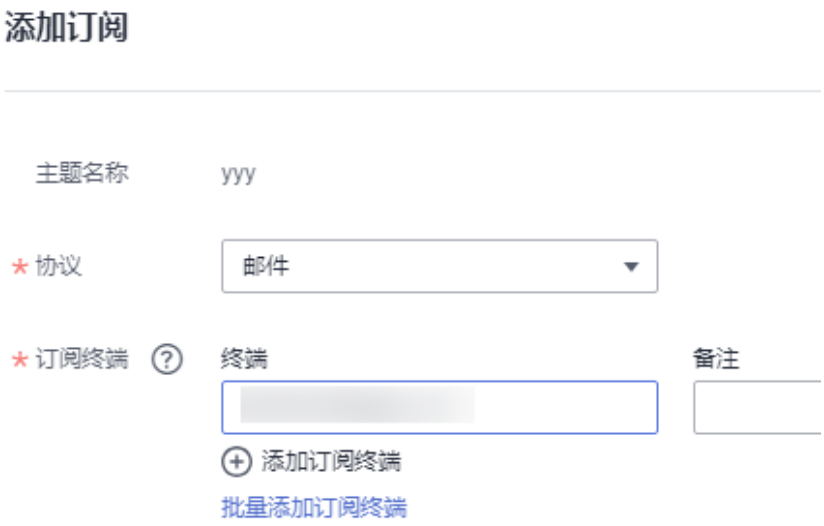
iii. 单击主题名称“操作”列的“更多 > 设置主题策略”。

选择APM，即允许AOM的告警触发SMN服务。

图 3-25 设置主题策略

iv. 单击主题名称“操作”列的“添加订阅”。订阅成功后，一旦满足告警条件，那么就会收到通知。

选择合适的协议，如邮件，短信等，并填写终端，如邮件地址，手机号等。单击确认。



此时订阅总数中会出现一条记录，但是处于未确认的状态。

收到邮件后单击“订阅确认”。

此时该订阅记录将处于已确认的状态。

- **创建告警行动规则**

  行动规则即为告警触发时，AOM以怎样的方式来告知用户。启用告警行动规则后，系统根据关联SMN主题与消息模板来发送告警通知。

  根据界面提示填写行动规则名称，选择行动规则类型，选择上一步创建的主题，选择消息模板，然后单击“确定”。

  图 3-26 新建告警行动规则

在之前打开的“创建告警规则”页面的告警通知区域，“行动规则”选择新创建的告警行动规则，单击“立即创建”。

至此，整个告警流程配置完成，一旦满足告警条件，那么就会收到邮件通知。

4 JupyterLab

4.1 JupyterLab 操作流程

ModelArts支持通过JupyterLab工具在线打开Notebook，开发基于PyTorch、TensorFlow和MindSpore引擎的AI模型。具体操作流程如下图所示。

图 4-1 使用 JupyterLab 在线开发调试代码

1. 创建Notebook实例。

   在ModelArts控制台创建一个Notebook开发环境实例，选择要使用的AI框架。具体参见**创建Notebook实例**。
2. 使用JupyterLab打开Notebook实例。具体参见**打开JupyterLab**。
3. 准备训练数据和代码文件，上传到JupyterLab中。具体参见**上传本地文件至JupyterLab**。
4. 在JupyterLab中编写代码文件，并运行调试。具体参见**JupyterLab简介及常用操作**。
5. 在JupyterLab中直接调用ModelArts提供的SDK，创建训练作业，上云训练。

   调用SDK创建训练作业的操作请参见**调用SDK创建训练作业**。

4.2 JupyterLab 简介及常用操作

JupyterLab是一个交互式的开发环境，是Jupyter Notebook的下一代产品，可以使用它编写Notebook、操作终端、编辑MarkDown文本、打开交互模式、查看csv文件及图片等功能。

可以说，JupyterLab是开发者们下一阶段更主流的开发环境。JupyterLab具有和Jupyter Notebook一样的组件，但支持更加灵活和更加强大的项目操作方式。

打开 JupyterLab

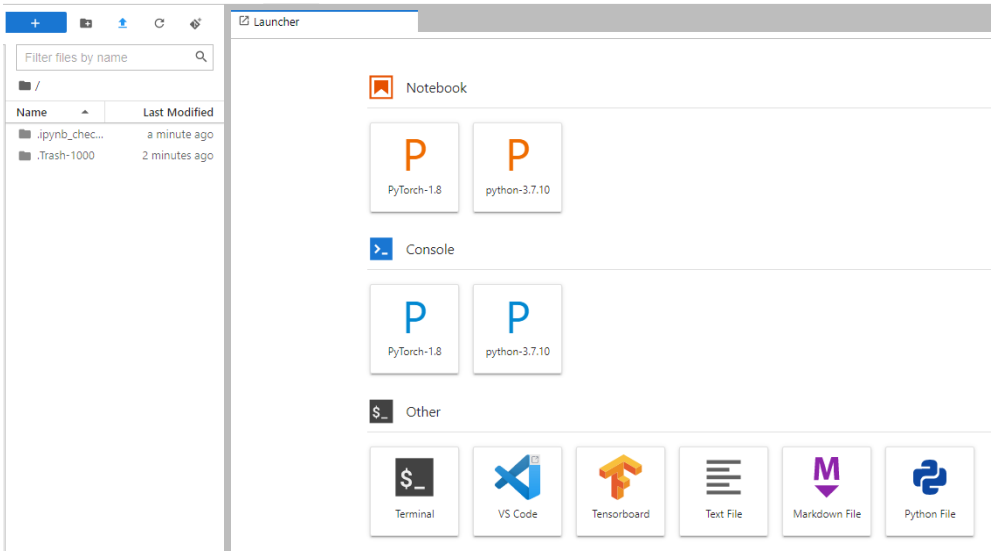
下面介绍如何从运行中的Notebook实例打开JupyterLab。

1. 登录ModelArts管理控制台，在左侧菜单栏中选择“开发空间 > Notebook”，进入Notebook页面。
2. 选择状态为“运行中”的Notebook实例，单击操作列的“打开”，访问JupyterLab。

图 4-2 打开 Notebook 实例

3. 进入JupyterLab页面后，自动打开Launcher页面，如下图所示。您可以使用开源支持的所有功能，详细操作指导可参见**JupyterLab官网文档**。

图 4-3 JupyterLab 主页



说明

不同AI引擎的Notebook，打开后Launcher页面呈现的Notebook和Console内核及版本均不同，**图4-3**仅作为示例，请以实际控制台为准。

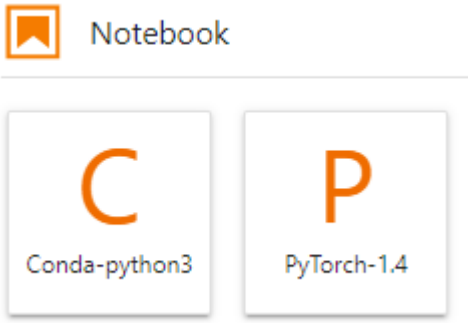
- Notebook：选择运行Notebook的一个内核，例如TensorFlow、python
- Console：可调出终端进行命令控制
- Other：可编辑其他文件

在 JupyterLab 中新建 ipynb 文件

进入JupyterLab主页后，可在“Notebook”区域下，选择适用的AI引擎，单击后将新建一个对应框架的ipynb文件。

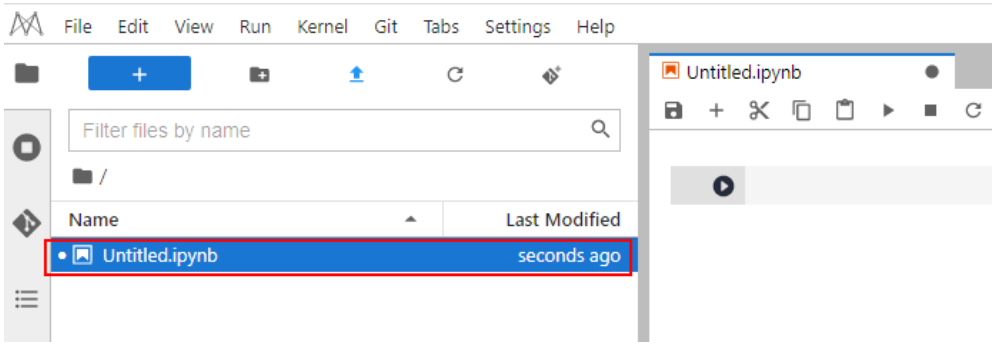
由于每个Notebook实例选择的工作环境不同，其支持的AI框架也不同，下图仅为示例，请根据实际显示界面选择AI框架。

图 4-4 选择 AI 引擎并新建一个 ipynb 文件



新建的ipynb文件将呈现在左侧菜单栏中。

图 4-5 新建文件



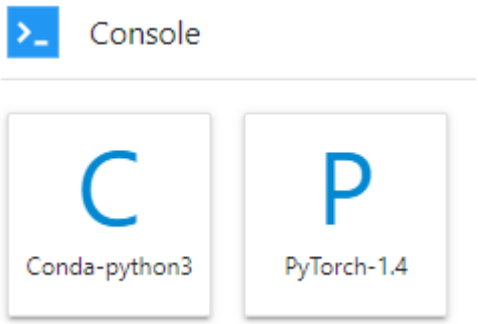
新建文件并打开 Console

Console的本质为Python终端，输入一条语句就会给出相应的输出，类似于Python原生的IDE。

进入JupyterLab主页后，可在“Console”区域下，选择适用的AI引擎，单击后将新建一个对应框架的Notebook文件。

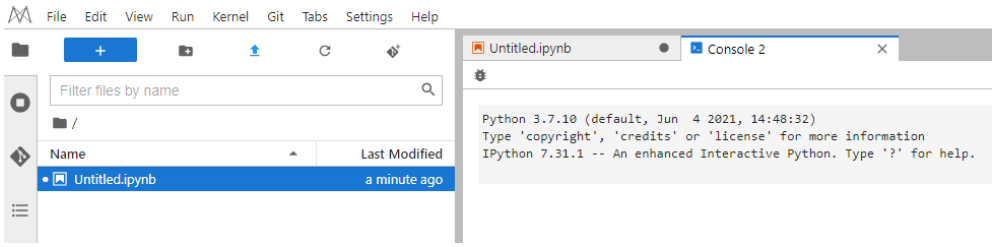
由于每个Notebook实例选择的工作环境不同，其支持的AI框架也不同，下图仅为示例，请根据实际显示界面选择AI框架。

图 4-6 选择 AI 引擎并新建一个 Console



文件创建成功后，将直接呈现Console页面。

图 4-7 新建文件（Console）

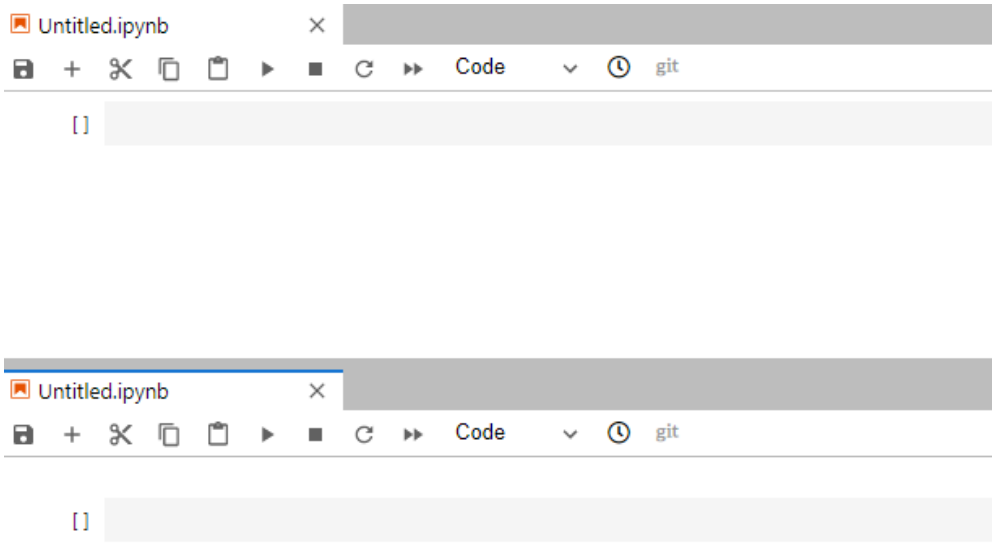


在 JupyterLab 中编辑文件

JupyterLab可以在同一个窗口同时打开几个Notebook或文件（如HTML、TXT、Markdown等），以页签形式展示。

JupyterLab的一大优点是，可以任意排版多个文件。在右侧文件展示区，您可以拖动打开文件，随意调整文件展示位置，可以同时打开多个文件。

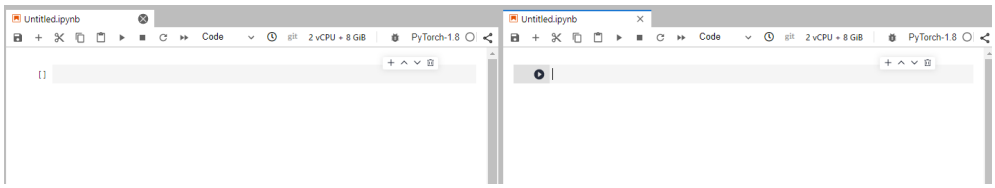
图 4-8 多文件任意编排



当在一个Notebook中写代码时，如果需要实时同步编辑文件并查看执行结果，可以新建该文件的多个视图。

打开ipynb文件，然后单击菜单栏“File > New View for Notebook”，即可打开多个视图。

图 4-9 同一个文件的多个视图



JupyterLab的ipynb文件代码栏中输入代码，需要在代码前加!符号。

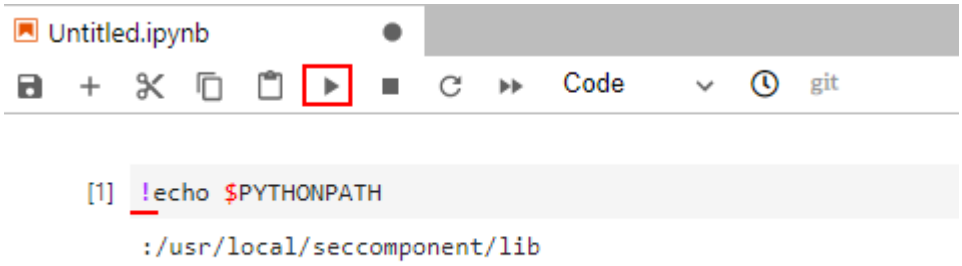
例如：安装外部库Shapely

```
!pip install Shapely
```

例如：查看PythonPath

```
!echo $PYTHONPATH
```

图 4-10 运行代码

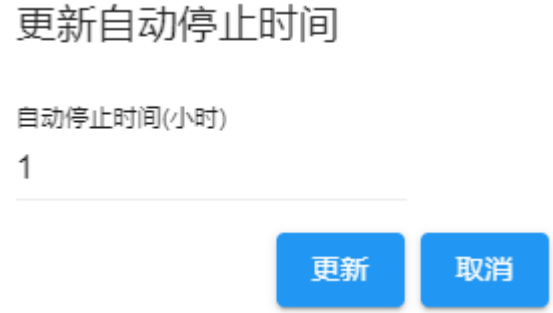


自动停止及续期

在创建或启动Notebook时，如果启用了自动停止功能，则在JupyterLab的右上角会显示当前实例停止的剩余时长，在计时结束前可以单击剩余时间进行续期。

图 4-11 自动停止

图 4-12 续期



JupyterLab 常用快捷键和插件栏

图 4-13 JupyterLab 常用快捷键和插件栏

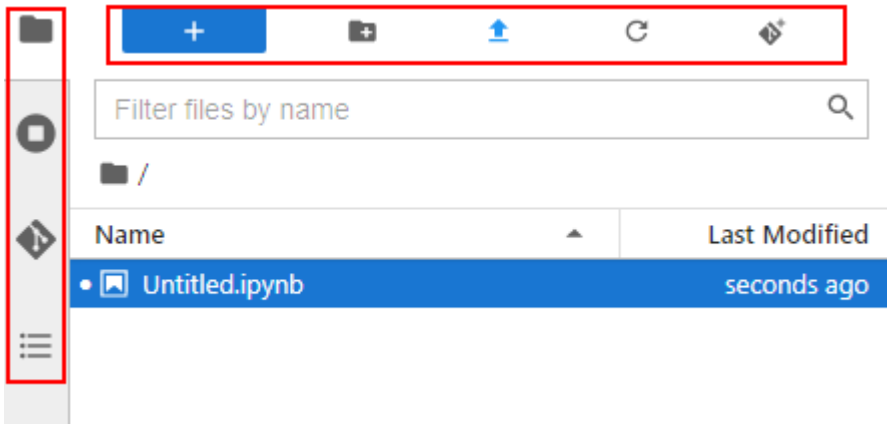


表 4-1 快捷键说明

快捷键	说明
+	快速打开Notebook、Terminal。或打开Launcher页面，可快速创建新的Notebook、Console或其他文件。
	创建文件夹。
	上传文件。
	刷新文件目录。
	Git插件，可连接此Notebook实例关联的Github代码库。

表 4-2 插件栏常用插件说明

插件	说明
	文件列表。单击此处，将展示此Notebook实例下的所有文件列表。
	当前实例中正在运行的Terminal和Kernel。
	Git插件，可以方便快捷地使用Github代码库。
	属性检查器。

插件	说明
	文档结构图。

图 4-14 导航栏按钮

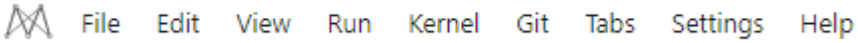


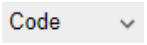
表 4-3 导航栏按钮介绍

按钮	说明
File	新建、关闭、保存、重新加载、重命名、导出、打印Notebook等功能。
Edit	编辑ipynb文件中代码块的相关操作，包括撤销、重做、剪切、复制、粘贴、选择、移动、合并、清除、查找代码块等。
View	查看视图相关操作。
Run	运行代码块相关操作，例如：运行选中代码块、一键运行所有代码块等。
Kernel	中断、重启、关闭、改变Kernel相关操作。
Git	Git插件相关操作，可以方便快捷地使用Github代码库。
Tabs	同时打开多个ipynb文件时，通过Tabs激活或选择文件。
Settings	JupyterLab工具系统设置。
Help	JupyterLab工具自带的帮助参考。

图 4-15 ipynb 文件菜单栏中的快捷键

表 4-4 ipynb 文件菜单栏中的快捷键

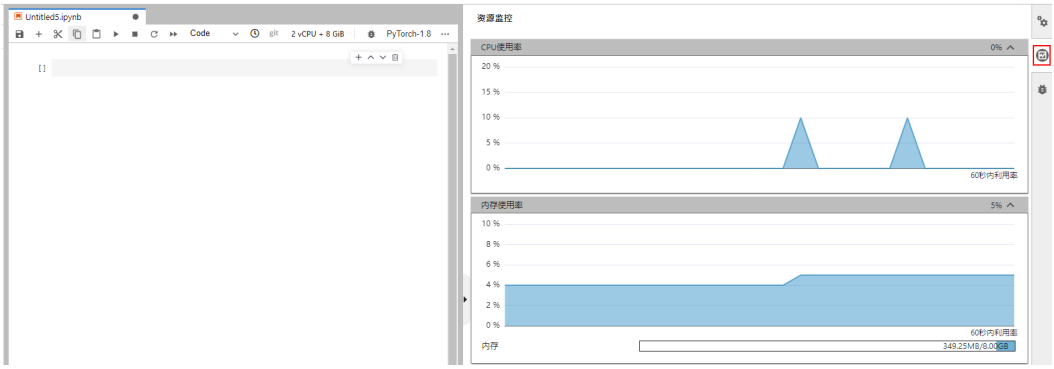
快捷键	说明
	保存文件。
+	添加新代码块。

快捷键	说明
	剪切选中的代码块。
	复制选中的代码块。
	粘贴选中的代码块。
	执行选中的代码块。
	终止kernel。
	重启kernel。
	重启kernel，然后重新运行当前Notebook的所有代码。
	此处下拉框有4个选项，分别是： Code（写python代码），Markdown（写Markdown代码，通常用于注释），Raw（一个转换工具），-（不修改）。
	查看代码历史版本。
	git插件，图标显示灰色表示当前Region不支持。
2 vCPU + 4 GiB	当前的资源规格。
PyTorch-1.4	单击可以选择Kernel。
	表示代码运行状态，变为实心圆  时，表示代码在运行中。

资源监控

在使用过程中，如果想了解资源使用情况，可在右侧区域选择“Resource Monitor”，展示“CPU使用率”和“内存使用率”。

图 4-16 资源监控



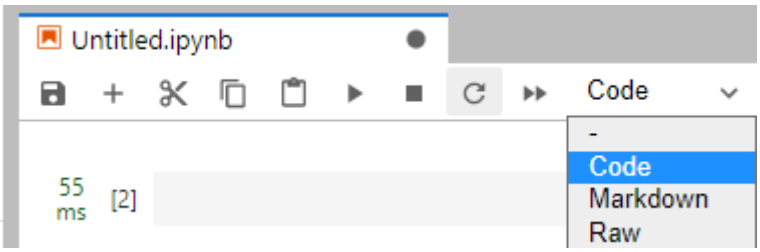
4.3 代码参数化插件

代码参数化插件可以降低Notebook案例的复杂度，用户无需感知复杂的源码，按需调整参数快速进行案例复现、模型训练等。该插件可用于定制Notebook案例，适用于比赛、教学等场景。

使用介绍

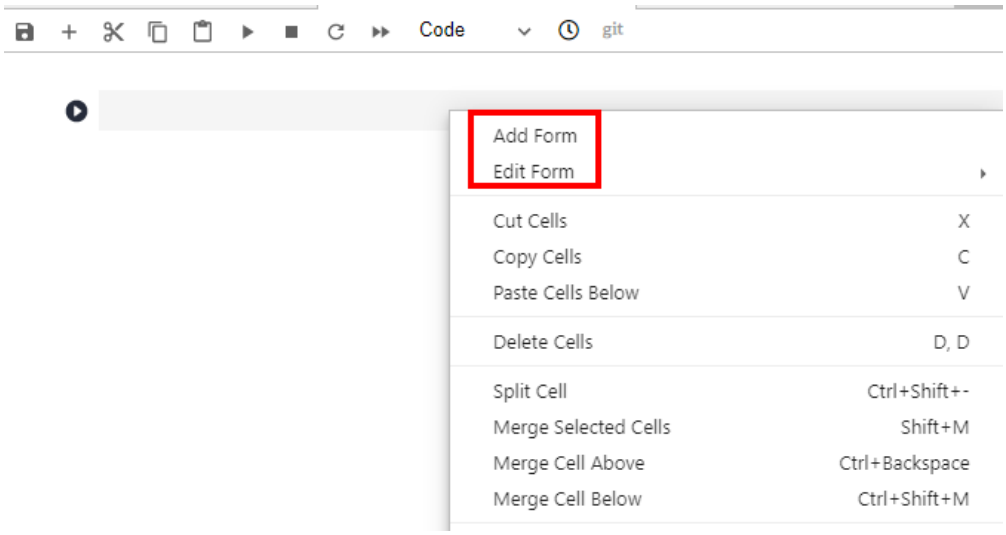
- 仅对Code cell类型新增了Edit Form和Add Form功能，如果cell类型是Markdown或者Raw类型则不支持。如下图所示：

  图 4-17 查看 Code cell

- 打开新的代码后，需先Add Form，再Edit Form。

  图 4-18 Code 类型的 cell 右键选项

Add Form 介绍

“Add Form”按钮会将Code cell水平拆分为两种编辑区域，左侧为代码区域，右侧为表单区域。单击表单右侧的“Edit”可修改默认标题。

图 4-19 两种编辑区域

Edit Form 介绍

“Edit Form”按钮有四个子选项，分别是“Add new form field”、“Hide code”、“Hide form”和“show all”四个按钮，下文介绍这四个选项的功能。

- “Add new form field”按钮支持新增“dropdown”、“input”和“slider”类型的表单。如图4-20所示。每新增一个字段，会分别在代码和表单区域中增加对应的变量，修改表单区域的值也会同时修改代码变量值。

  说明

  创建dropdown类型的表单时，“ADD Item”至少创建2项。如图4-21所示。

图 4-20 “dropdown”，“input”，“slider”的表单样式

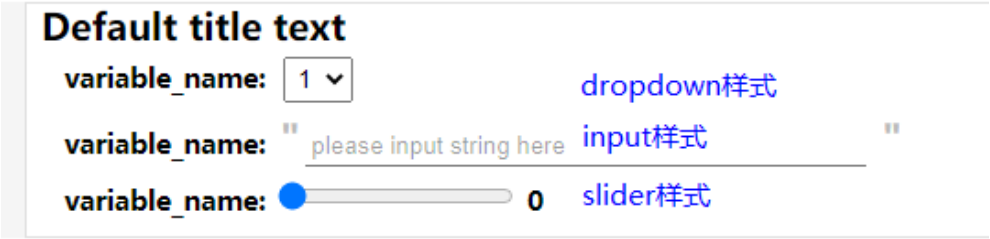


图 4-21 创建“dropdown”类型的表单

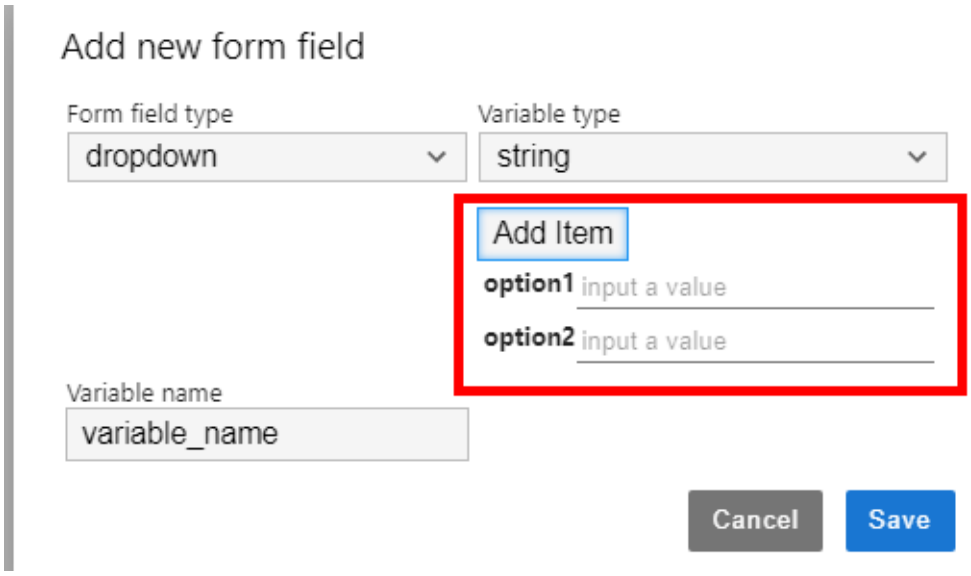


图 4-22 删除表单

  - 表单字段类型为“dropdown”时，支持的变量类型为“raw”和“string”。
  - 表单字段类型为“input”时，支持的变量类型有“boolean”、“date”、“integer”、“number”、“raw”和“string”。
  - 表单字段类型为“slider”时，支持输入滑动条的最小值、最大值和步长。
- “Hide code”按钮会隐藏代码区域。
- “Hide form”按钮会隐藏表单区域。
- “Show all”按钮会同时展示code和form区域。

4.4 使用 ModelArts SDK

在Notebook中，通过使用ModelArts SDK，可以完成OBS管理、训练作业管理、模型管理以及在线服务管理。

在Notebook中，已承载了登录用户的鉴权信息（AK/SK）和区域信息，因此SDK session鉴权时，无需输入参数即可完成session鉴权。

示例代码

- 创建训练作业

```
from modelarts.session import Session
from modelarts.estimator import Estimator
session = Session()
estimator = Estimator(
                    modelarts_session=session,
                    framework_type='PyTorch',                                   # AI引擎名称
                    framework_version='PyTorch-1.0.0-python3.6',                # AI引擎版本
                    code_dir='/obs-bucket-name/src/',                           # 训练脚本目录
                    boot_file='/obs-bucket-name/src/pytorch_sentiment.py',      # 训练启动脚本目录
                    log_url='/obs-bucket-name/log/',                            # 训练日志目录
                    hyperparameters=[
                                {"label":"classes",
                                 "value": "10"},
                                {"label":"lr",
                                 "value": "0.001"}
                                ],
                    output_path='/obs-bucket-name/output/',                     # 训练输出目录
                    train_instance_type='modelarts.vm.xxx.xxx',                 # 训练环境规格
                    train_instance_count=1,                                     # 训练节点个数
                    job_description='pytorch-sentiment with ModelArts SDK')     # 训练作业描述
job_instance = estimator.fit(inputs='/obs-bucket-name/data/train/', wait=False,
job_name='my_training_job')
```

- 查询模型列表

```
from modelarts.session import Session
from modelarts.model import Model
session = Session()
model_list_resp = Model.get_model_list(session, model_status="published", model_name="digit",
order="desc")
```

- 查询服务详情

```
from modelarts.session import Session
from modelarts.model import Predictor
session = Session()
predictor_instance = Predictor(session, service_id="input your service_id")
predictor_info_resp = predictor_instance.get_service_info()
```

4.5 使用 Git 插件

在JupyterLab中使用Git插件可以克隆GitHub开源代码仓库，快速查看及编辑内容，并提交修改后的内容。

前提条件

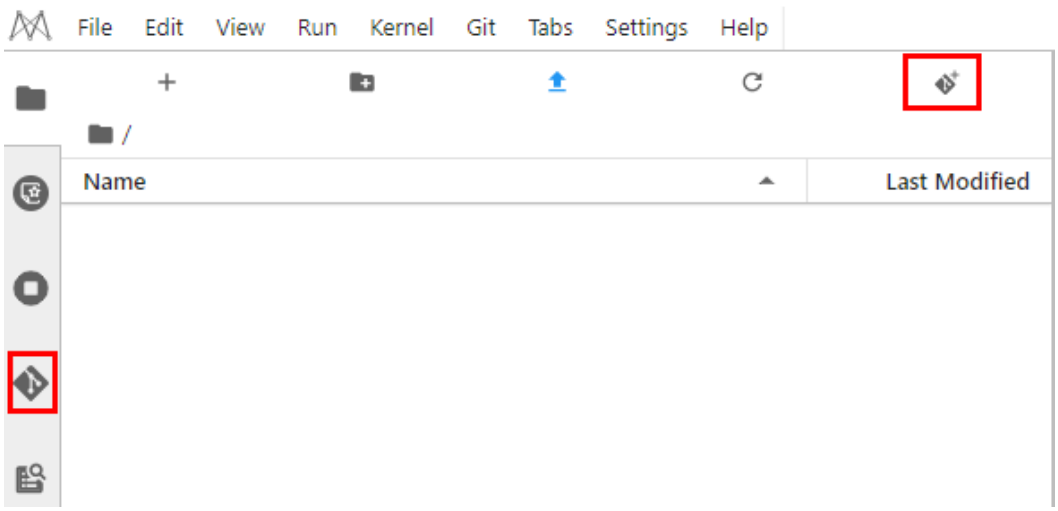
Notebook处于运行中状态。

打开 JupyterLab 的 git 插件

在Notebook列表中，选择一个实例，单击右侧的打开进入“JupyterLab”页面。

图4-23所示图标，为JupyterLab的Git插件。

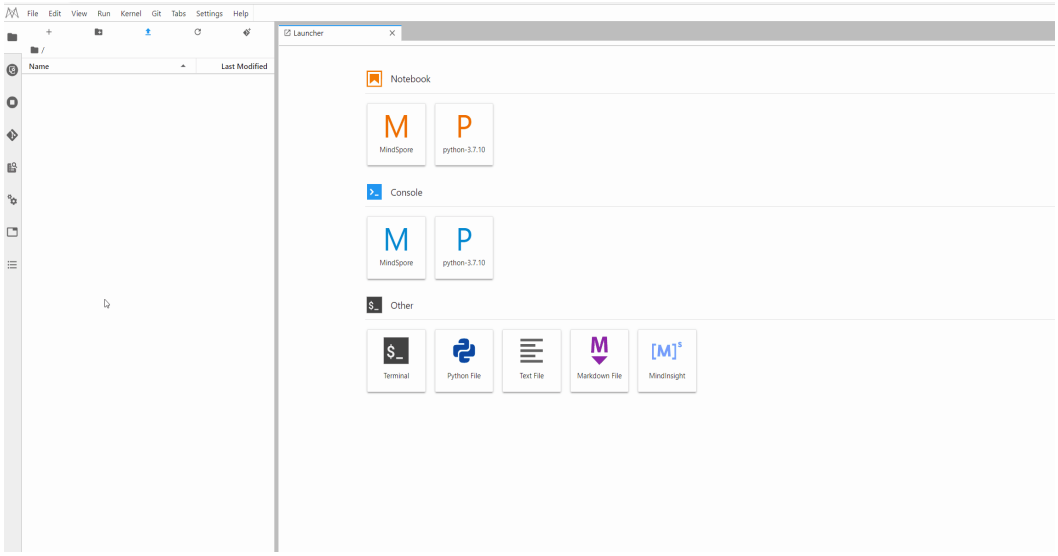
图 4-23 Git 插件



克隆 GitHub 的开源代码仓库

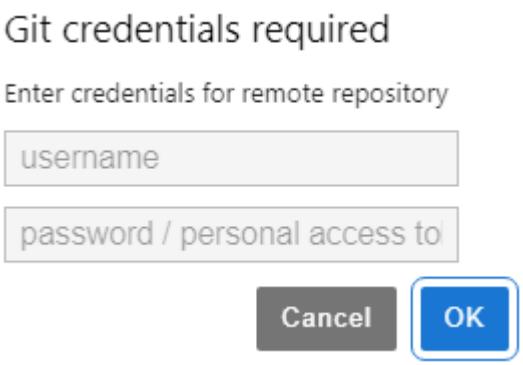
GitHub开源仓库地址：https://github.com/jupyterlab/extension-examplesitHub，单击 ，输入仓库地址，单击确定后即开始克隆，克隆完成后，JupyterLab左侧导航出现代码库文件夹。

图 4-24 使用 git 插件克隆 GitHub 的开源代码仓库



克隆 GitHub 的私有仓库

克隆GitHub私有仓库时，会弹出输入个人凭证的对话框，如下图。此时需要输入GitHub中Personal Access Token信息。



查看Personal Access Token步骤如下：

1. 登录**Github**，打开设置页面。
2. 单击“Developer settings”。
3. 单击“Personal access tokens > Generate new token”。
4. 验证登录账号。
5. 填写Token描述并选择权限，选择私有仓库访问权限，单击“Generate token”生成Token。
6. 复制生成的Token到编译构建服务即可。

须知

- Token生成后，请及时保存，下次刷新页面将无法读取，需要重新生成新Token。
- 注意填写有效的Token描述信息，避免误删除导致构建失败。
- 无需使用时及时删除Token，避免信息泄露。

图 4-25 克隆 GitHub 的私有仓库（目前只支持 Personal Access Token 授权）

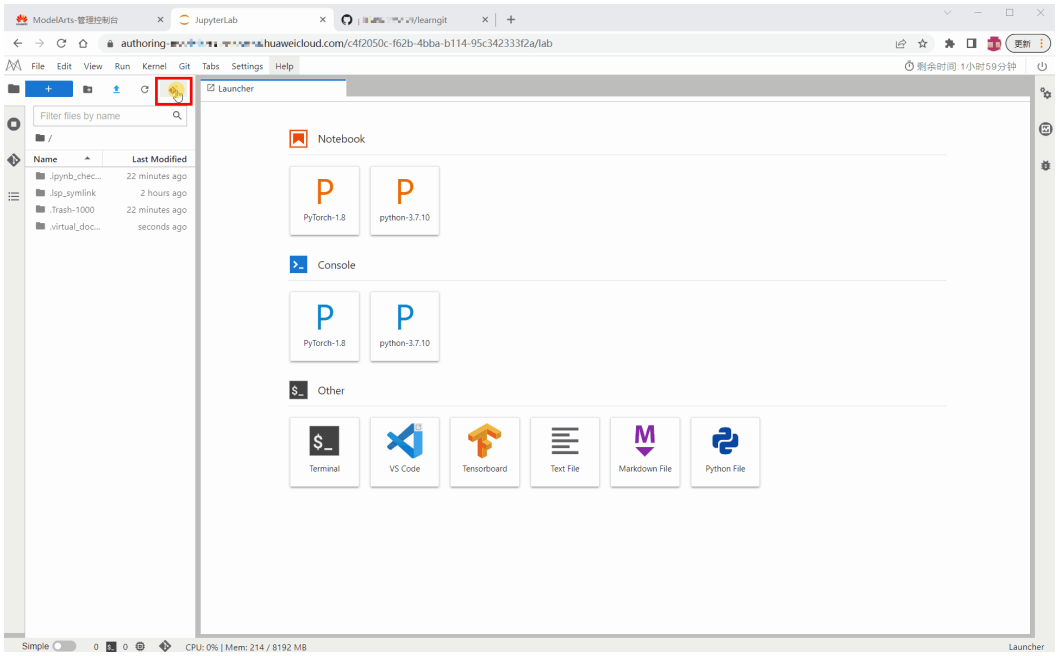
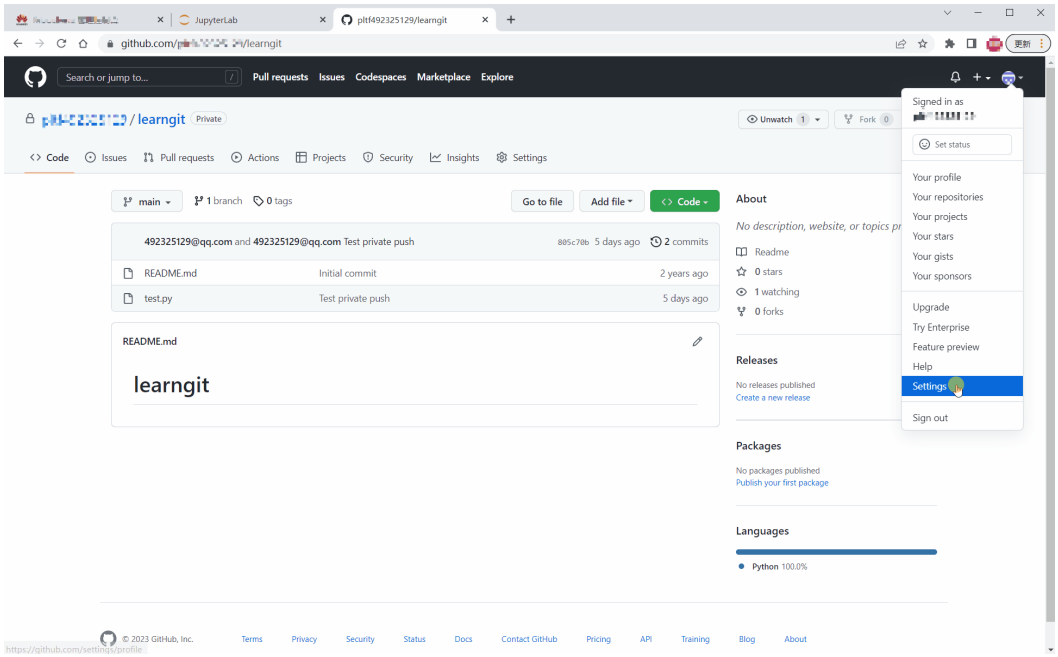


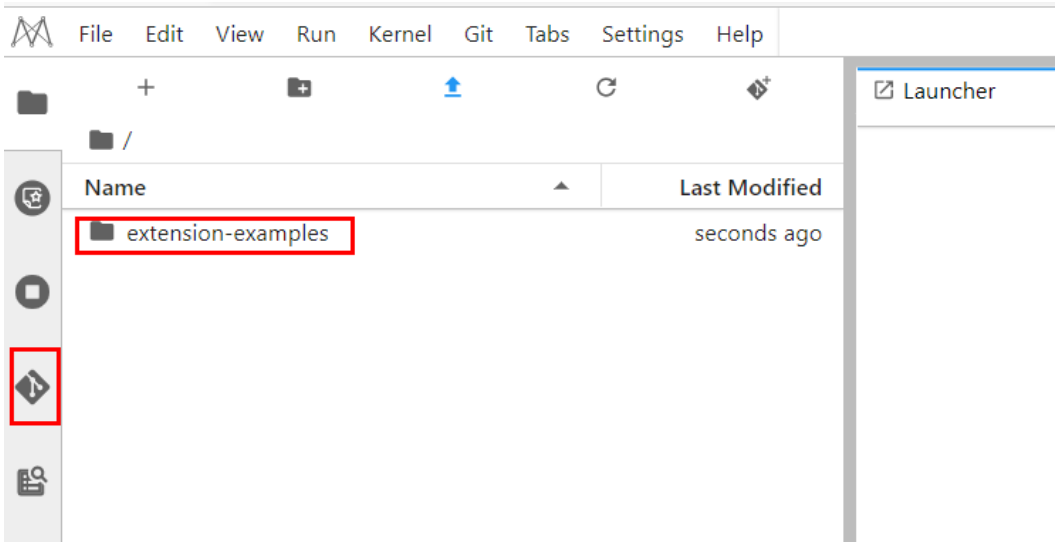
图 4-26 获取 Personal Access Token



查看代码库信息

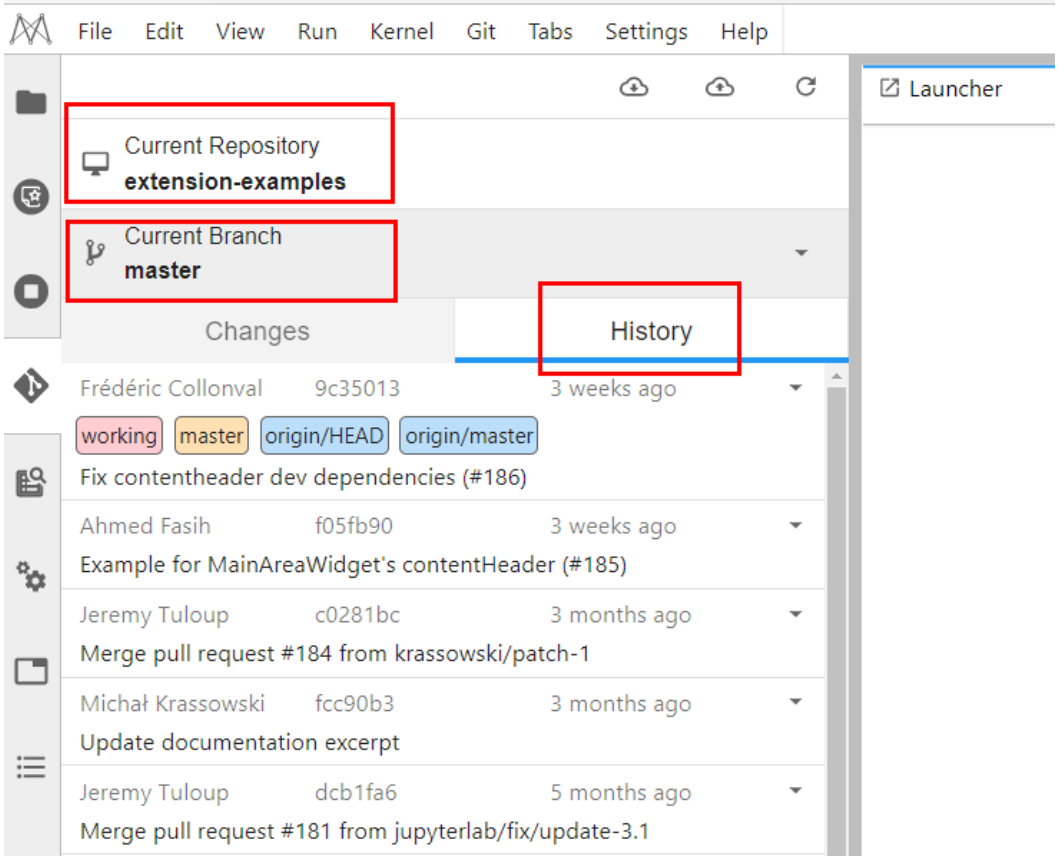
在Name下方列表中，选中您希望使用的文件夹，双击打开，然后单击左侧git插件图标进入此文件夹对应的代码库。

图 4-27 打开文件夹后打开 git 插件



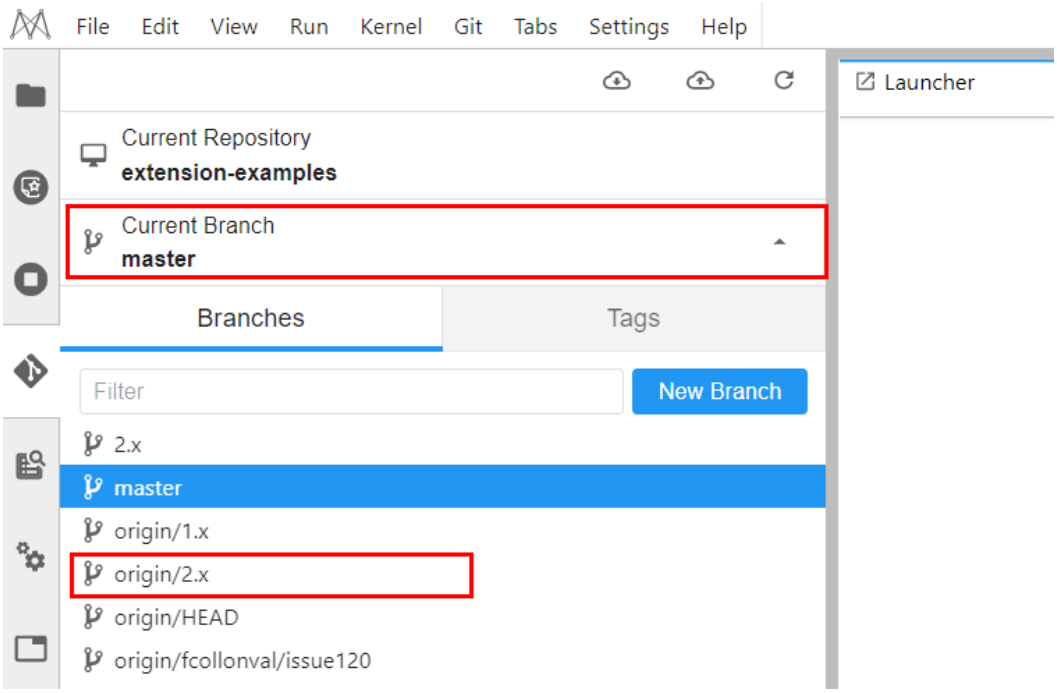
即可看到当前代码库的信息，如仓库名称、分支、历史提交记录等。

图 4-28 查看代码库信息



说明

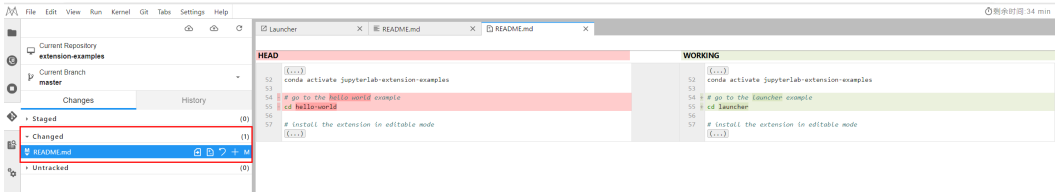
Git插件一般默认克隆master分支，如果要切换分支可单击Current Branch展开所有分支，单击相应分支名称可完成切换。



查看修改的内容

如果修改代码库中的某个文件，在“Changes”页签的“Changed”下可以看到修改的文件，并单击修改文件名称右侧的“Diff this file”，可以看到修改的内容。

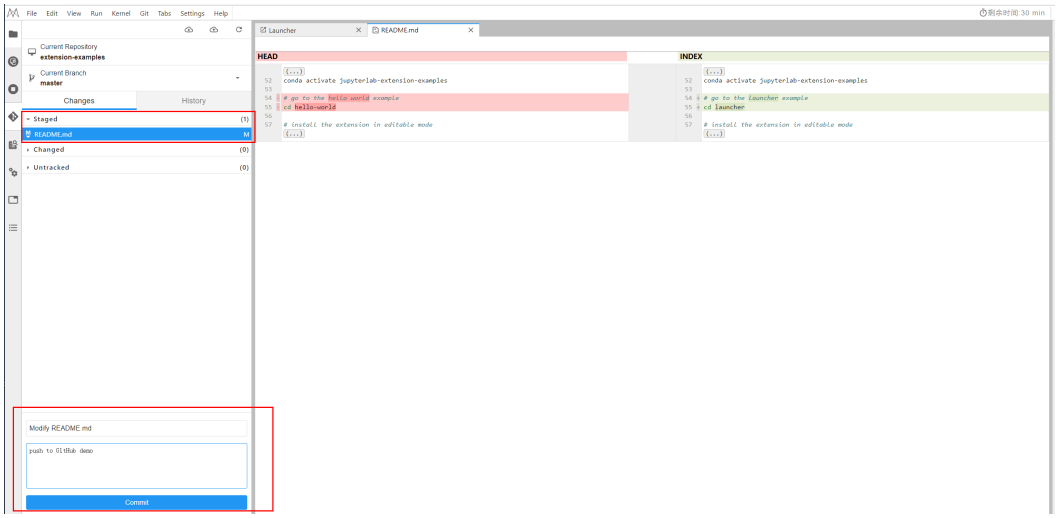
图 4-29 查看修改的内容



提交修改的内容

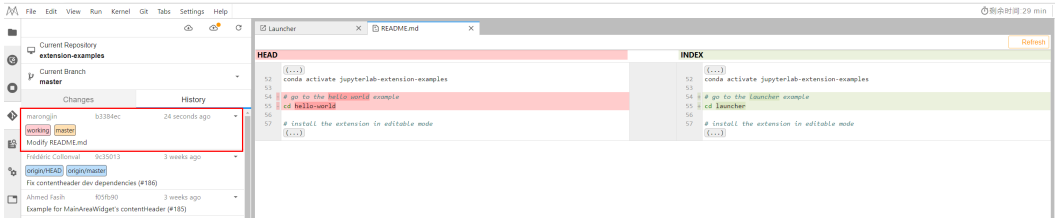
确认修改无误后，单击修改文件名称右侧的“Stage this change”，文件将进入Staged状态，相当于执行了git add命令。在左下方输入本次提交的Message，单击“Commit”，相当于执行了git commit命令。

图 4-30 提交修改内容



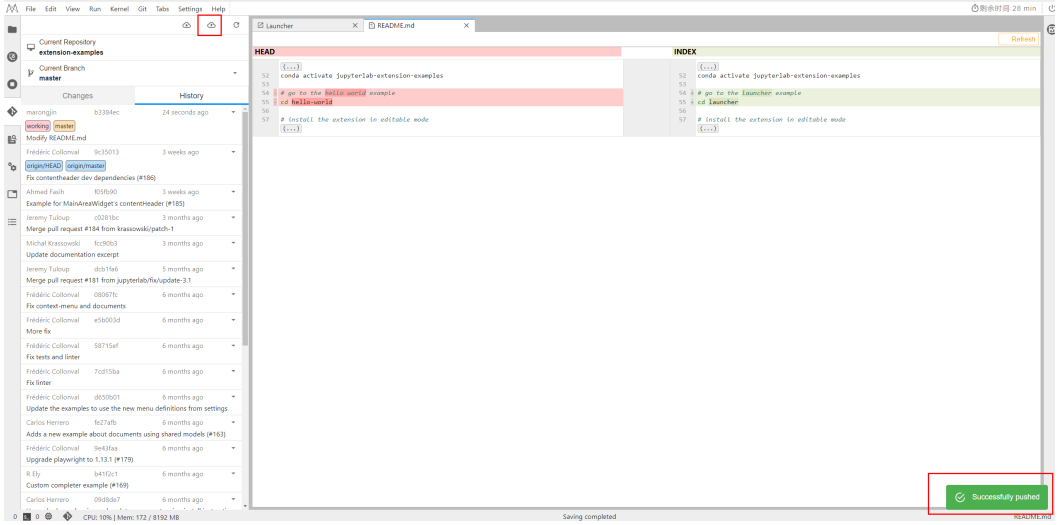
此时，可以在“History”页签下看到本地提交已成功。

图 4-31 查看是否提交成功



单击“push”按钮，相当于执行git push命令，即可提交代码到GitHub仓库中。提交成功后会提示“Successfully completed ”。如果OAuth鉴权的token过期，则此时再push会弹框让输入用户的token或者账户信息，按照提示输入即可。这里推荐使用Personal Access Token授权方式，如果出现密码失效报错请参考**git插件密码失效如何解决?**

图 4-32 提交代码至 GitHub 仓库



完成上述操作后，可以在JupyterLab的git插件页面的History页签，看到“origin/HEAD”和“orgin/master”已指向最新一次的提交。同时在GitHub对应仓库的commit记录中也可以查找到对应的信息。

4.6 可视化训练作业

4.6.1 可视化训练作业介绍

ModelArts支持在开发环境中开启TensorBoard和MindInsight可视化工具。在开发环境中通过小数据集训练调试算法，主要目的是验证算法收敛性、检查是否有训练过程中的问题，方便用户调测。

ModelArts可视化作业支持创建TensorBoard类型和MindInsight两种类型。

TensorBoard和MindInsight能够有效地展示训练作业在运行过程中的变化趋势以及训练中使用到的数据信息。

- TensorBoard

  TensorBoard是一个可视化工具，能够有效地展示TensorFlow在运行过程中的计算图、各种指标随着时间的变化趋势以及训练中使用到的数据信息。TensorBoard相关概念请参考**TensorBoard官网**。

  TensorBoard可视化训练作业，当前仅支持基于TensorFlow2.1、Pytorch1.4/1.8版本镜像，CPU/GPU规格的资源类型。请根据实际局点支持的镜像和资源规格选择使用。

- MindInsight

  MindInsight能可视化展现出训练过程中的标量、图像、计算图以及模型超参等信息，同时提供训练看板、模型溯源、数据溯源、性能调试等功能，帮助您在更高效地训练调试模型。MindInsight当前支持基于MindSpore引擎的训练作业。MindInsight相关概念请参考**MindSpore官网**。

  MindInsight可视化训练作业，当前支持的镜像如下，请根据实际局点支持的镜像和资源规格选择使用。

  - mindspore1.2.0版本，CPU/GPU规格的资源类型。

您可以使用模型训练时产生的Summary文件在开发环境Notebook中创建可视化作业。

- 在开发环境中创建MindInsight可视化作业，请参见**MindInsight可视化作业**。
- 在开发环境中创建TensorBoard可视化作业，请参见**TensorBoard可视化作业**。

4.6.2 MindInsight 可视化作业

ModelArts支持在开发环境中开启MindInsight可视化工具。在开发环境中通过小数据集训练调试算法，主要目的是验证算法收敛性、检查是否有训练过程中的问题，方便用户调测。

MindInsight能可视化展现出训练过程中的标量、图像、计算图以及模型超参等信息，同时提供训练看板、模型溯源、数据溯源、性能调试等功能，帮助您在更高效地训练调试模型。MindInsight当前支持基于MindSpore引擎的训练作业。MindInsight相关概念请参考**MindSpore官网**。

MindSpore支持将数据信息保存到Summary日志文件中，并通过可视化界面MindInsight进行展示。

前提条件

使用MindSpore引擎编写训练脚本时，为了保证训练结果中输出Summary文件，您需要在脚本中添加收集Summary相关代码。

将数据记录到Summary日志文件中的具体方式请参考**收集Summary数据**。

注意事项

- 在开发环境跑训练任务，在开发环境使用MindInsight，要求先启动MindInsight，后启动训练进程。
- 仅支持单机单卡训练。
- 运行中的可视化作业不单独计费，当停止Notebook实例时，计费停止。
- Summary文件如果存放在OBS中，由OBS单独收费。任务完成后请及时停止Notebook实例，清理OBS数据，避免产生不必要的费用。

在开发环境中创建 MindInsight 可视化作业流程

Step1 创建开发环境并在线打开

Step2 上传Summary数据

Step3 启动MindInsight

Step4 查看训练看板中的可视化数据

Step1 创建开发环境并在线打开

在ModelArts控制台，进入“开发空间> Notebook”页面，创建MindSpore引擎的开发环境实例。创建成功后，单击开发环境实例操作栏右侧的“打开 ”，在线打开运行中的开发环境。

MindInsight可视化训练作业，当前支持的镜像如下，请根据实际局点支持的镜像和资源规格选择使用。

- mindspore1.2.0版本，CPU/GPU规格的资源类型。
- mindspore1.5.x以上版本，Ascend规格的资源类型。

Step2 上传 Summary 数据

在开发环境中使用MindInsight可视化功能，需要用到Summary数据。

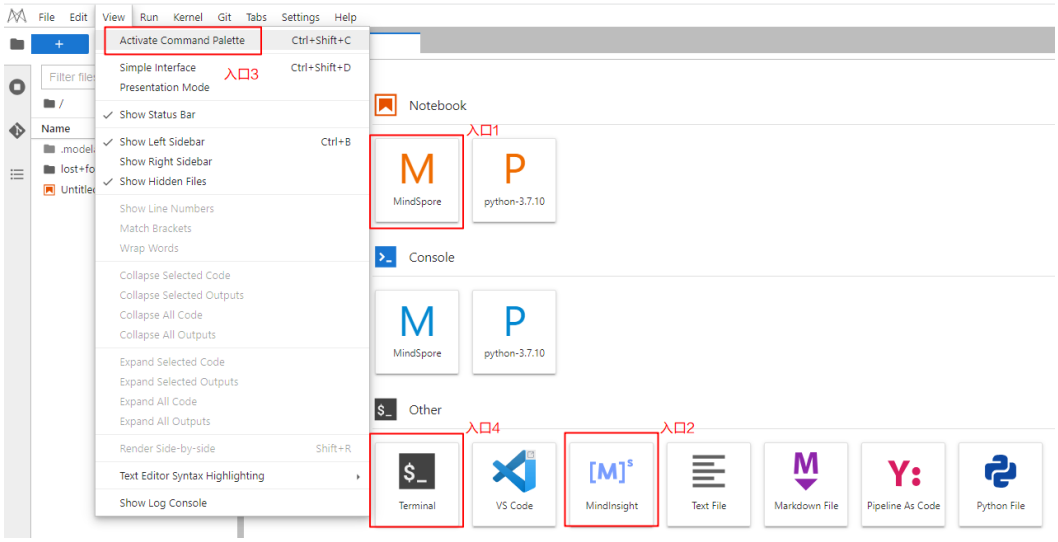
Summary数据可以直接传到开发环境的这个路径下/home/ma-user/work/，也可以放到OBS并行文件系统中。

- Summary数据上传到Notebook路径/home/ma-user/work/下的方式，请参见**上传数据至Notebook**。
- Summary数据如果是通过OBS并行文件系统挂载到Notebook中，请将模型训练时产生的Summary文件先上传到OBS并行文件系统，并确保OBS并行文件系统与ModelArts在同一区域。在Notebook中启动MindInsight时，Notebook会自动从挂载的OBS并行文件系统目录中读取Summary数据。

Step3 启动 MindInsight

在开发环境的JupyterLab中打开MindInsight有多种方法。可根据使用习惯选择。

图 4-33 JupyterLab 中打开 MindInsight 的方法



方式1：

1. 单击入口1 

，进入JupyterLab开发环境，并自动创建“.ipynb”文件。
2. 在对话框中输入MindInsight相应命令，即可展示界面。

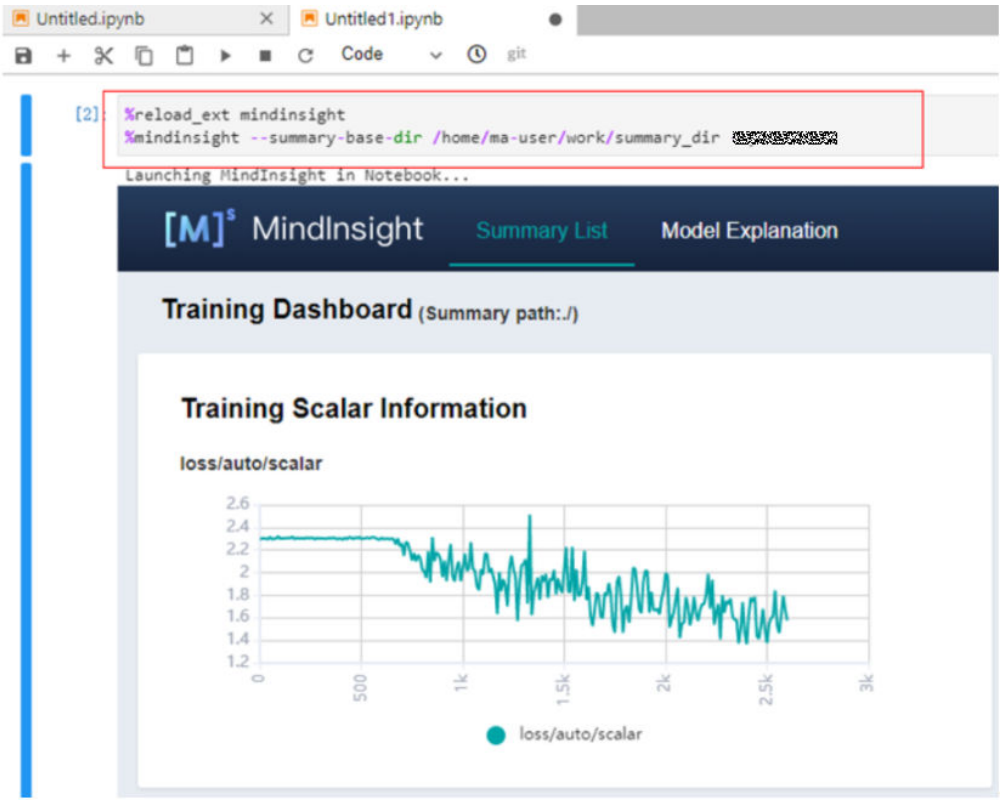
```
%reload_ext mindinsight
%mindinsight --port {PORT} --summary-base-dir {SUMMARY_BASE_DIR}
```

参数解释：

- --port {PORT}：指定Web可视化服务端口。可以不设置，默认使用8080端口。如果8080端口被占用了，需要在1~65535任意指定一个端口。
- --summary-base-dir {SUMMARY_BASE_DIR}：表示数据在开发环境中的存储路径。
  - 开发环境本地路径：“./work/xxx”（相对路径）或/home/ma-user/work/xxx（绝对路径）
  - OBS并行文件系统桶的路径：obs://xxx/

```
例如：
#Summary数据如果是在开发环境的这个路径下/home/ma-user/work/，执行下面这条命令
%mindinsight --summary-base-dir /home/ma-user/work/xxx
或者
#Summary数据如果是存在OBS并行文件系统中，执行下面这条命令，开发环境会自动挂载该OBS并行文件系统路径并读取数据
%mindinsight --summary-base-dir obs://xxx/
```

图 4-34 MindInsight 界面（1）



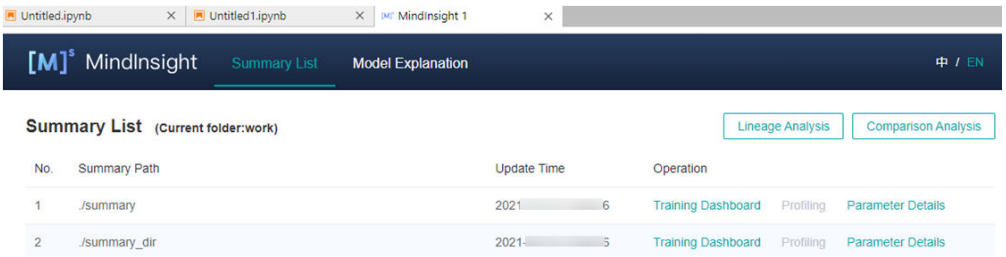
方式2：

单击入口2，直接进入MindInsight可视化界面。

默认读取路径/home/ma-user/work/

当存在两个以及以上工程的log时，界面如下。通过Runs下选择查看相对应的log。

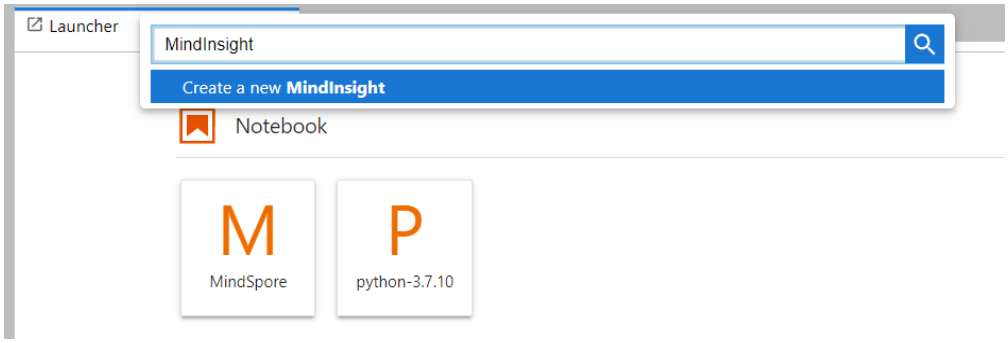
图 4-35 MindInsight 界面（2）



方式3：

1. 单击入口3 “View > Activate Command Palette”，在搜索框中输入“MindInsight”，再单击“Create a new MindInsight”。

图 4-36 Create a new MindInsight



2. 填写需要查看的Summary数据路径，或者OBS并行文件系统桶的路径，单击CREATE。
   - 开发环境本地路径：./summary（相对路径）或/home/ma-user/work/summary（绝对路径）
   - OBS并行文件系统的路径：obs://xxx/

图 4-37 输入 Summary 数据路径

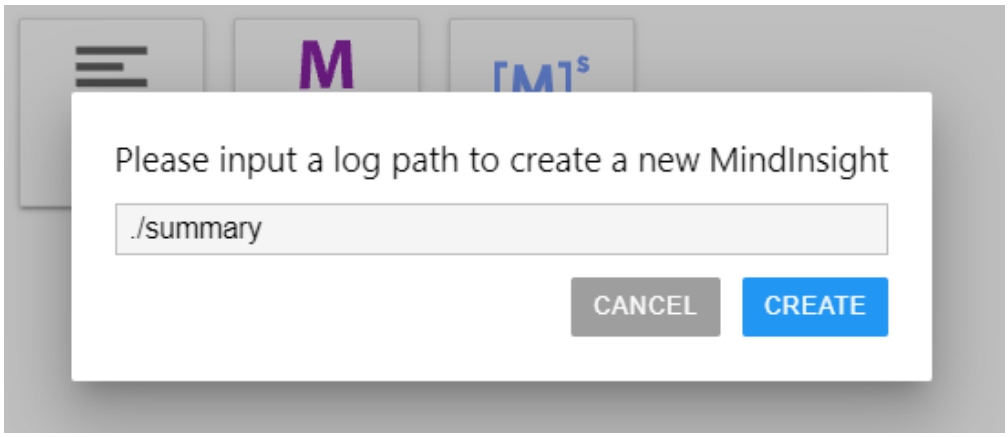
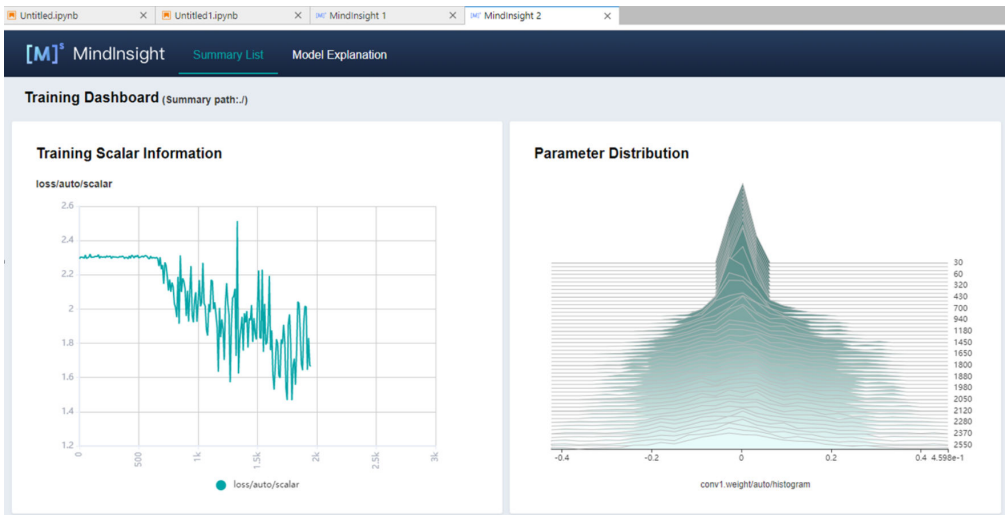


图 4-38 MindInsight 界面（3）



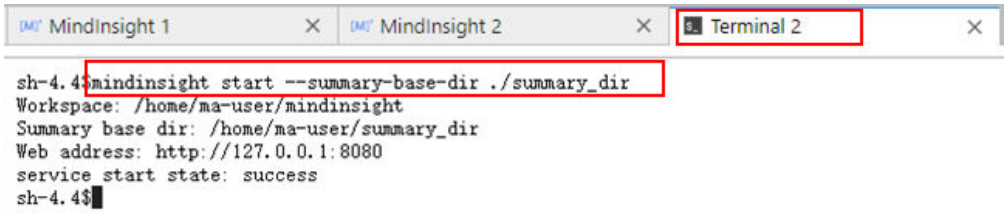
> **说明**
>
> 方式2及方式3最多可以启动10个MindInsight实例。

方式4：

单击入口4 Terminal ，输入并执行以下命令，但启动后不能显示UI界面。

```
mindinsight start --summary-base-dir ./summary_dir
```

图 4-39 Terminal 方式打开 MindInsight



Step4 查看训练看板中的可视化数据

训练看板是MindInsight的可视化组件的重要组成部分，而训练看板的标签包含：标量可视化、参数分布图可视化、计算图可视化、数据图可视化、图像可视化和张量可视化等。

更多功能介绍请参见MindSpore官网资料：**查看训练看板中可视的数据**。

相关操作

关闭MindInsight方式如下：

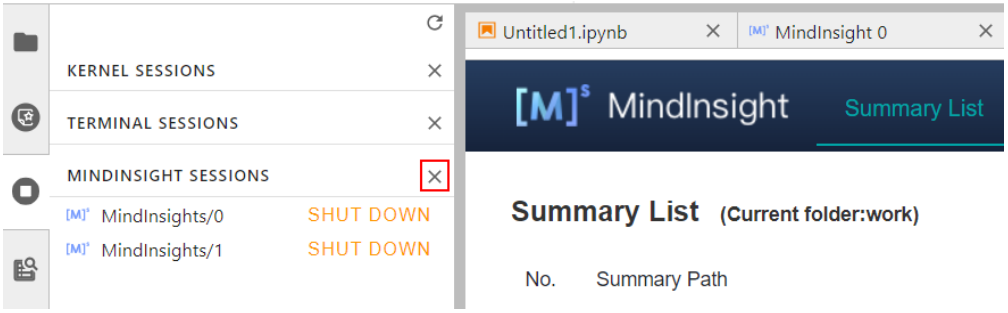
- 方式1：在开发环境JupyterLab中的“.ipynb”文件窗口中输入命令，关闭MindInsight。端口号在**启动MindInsight**中设置，默认使用8080，需要替换为实际开启MindInsight时的端口。
  ```
  !mindinsight stop --port 8080
  ```
- 方式2：单击下方 按钮进入MindInsight实例管理界面，该界面记录了所有启动的MindInsight实例，单击对应实例后面的SHUT DOWN即可停止该实例。

  图 4-40 单击 SHUT DOWN 停止实例

- 方式3：单击下方红框中的  按钮可以关闭所有启动的MindInsight实例。

  图 4-41 关闭所有启动的 MindInsight 实例

- 方式4（不推荐）：直接在JupyterLab中上关闭MindInsight窗口，此方式仅是关闭MindInsight可视化窗口，并未关闭后台。

4.6.3 TensorBoard 可视化作业

ModelArts支持在开发环境中开启TensorBoard可视化工具。TensorBoard是TensorFlow的可视化工具包，提供机器学习实验所需的可视化功能和工具。

TensorBoard能够有效地展示TensorFlow在运行过程中的计算图、各种指标随着时间的变化趋势以及训练中使用到的数据信息。

前提条件

为了保证训练结果中输出Summary文件，在编写训练脚本时，您需要在脚本中添加收集Summary相关代码。

TensorFlow引擎的训练脚本中添加Summary代码，具体方式请参见**TensorFlow官方网站**。

注意事项

- 运行中的可视化作业不单独计费，当停止Notebook实例时，计费停止。
- Summary文件数据如果存放在OBS中，由OBS单独收费。任务完成后请及时停止Notebook实例，清理OBS数据，避免产生不必要的费用。

在开发环境中创建 TensorBoard 可视化作业流程

Step1 创建开发环境并在线打开

Step2 上传Summary数据

Step3 启动TensorBoard

Step4 查看训练看板中的可视化数据

Step1 创建开发环境并在线打开

在ModelArts控制台，进入“开发空间 > Notebook”页面，创建TensorFlow或者PyTorch镜像的开发环境实例。创建成功后，单击开发环境实例操作栏右侧的“打开”，在线打开运行中的开发环境。

TensorBoard可视化训练作业，当前仅支持基于TensorFlow2.1、Pytorch1.4/1.8以上版本镜像，CPU/GPU规格的资源类型。请根据实际局点支持的镜像和资源规格选择使用。

Step2 上传 Summary 数据

在开发环境中使用TensorBoard可视化功能，需要用到Summary数据。

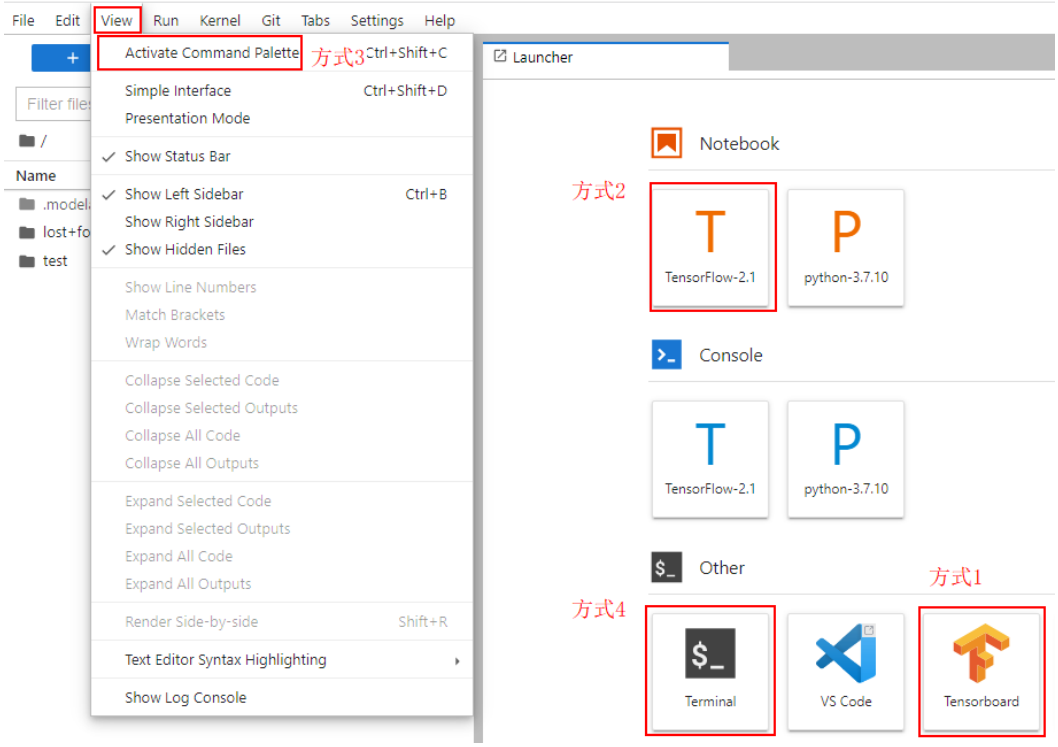
Summary数据可以直接传到开发环境的这个路径下/home/ma-user/work/，也可以放到OBS并行文件系统中。

- Summary数据上传到Notebook路径/home/ma-user/work/下的方式，请参见**上传数据至Notebook**。
- Summary数据如果是通过OBS并行文件系统挂载到Notebook中，请将模型训练时产生的Summary文件先上传到OBS并行文件系统，并确保OBS并行文件系统与ModelArts在同一区域。在Notebook中启动TensorBoard时，Notebook会自动从挂载的OBS并行文件系统目录中读取Summary数据。

Step3 启动 TensorBoard

在开发环境的JupyterLab中打开TensorBoard有多种方法。可根据使用习惯选择。

图 4-42 JupyterLab 中打开 TensorBoard 的方法

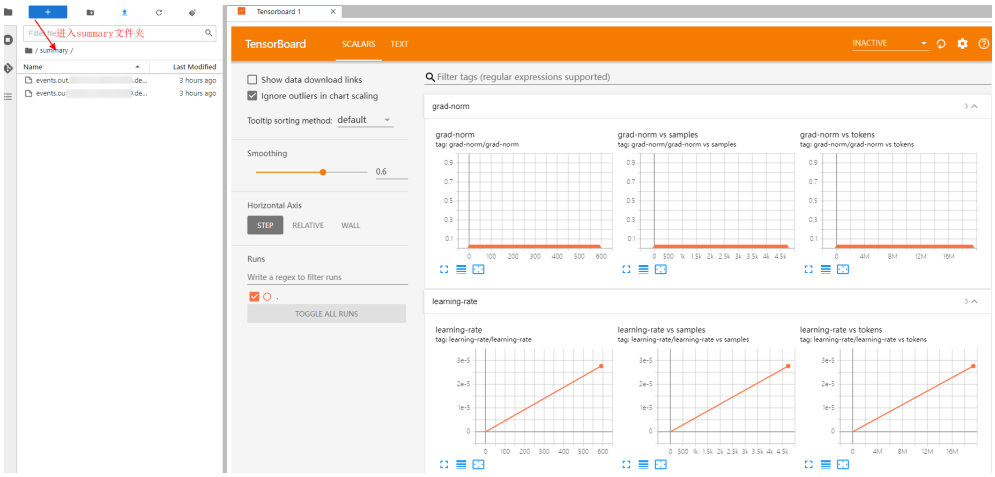


方式1（推荐）：

1. 在JupyterLab左侧导航创建名为“summary”的文件夹，将数据上传到“/home/ma-user/work/summary”路径。注：文件夹命名只能为summary否则无法使用。

2. 进入“summary”文件夹，单击方式1，直接进入TensorBoard可视化界面。如图4-43所示。

图 4-43 TensorBoard 界面（1）



方式2：

须知

用户可以自行升级除2.4.0之外的TensorBoard，但需注意升级后只有方式2使用新的TensorBoard，其余方式保持TensorBoard2.1.1不变。

1. 单击方式2，进入JupyterLab开发环境中，并自动创建“.ipynb”文件。
2. 在对话框中输入TensorBoard相应命令，即可展示界面。

```
%reload_ext ma_tensorboard
%ma_tensorboard  --port {PORT} --logdir {BASE_DIR}
```

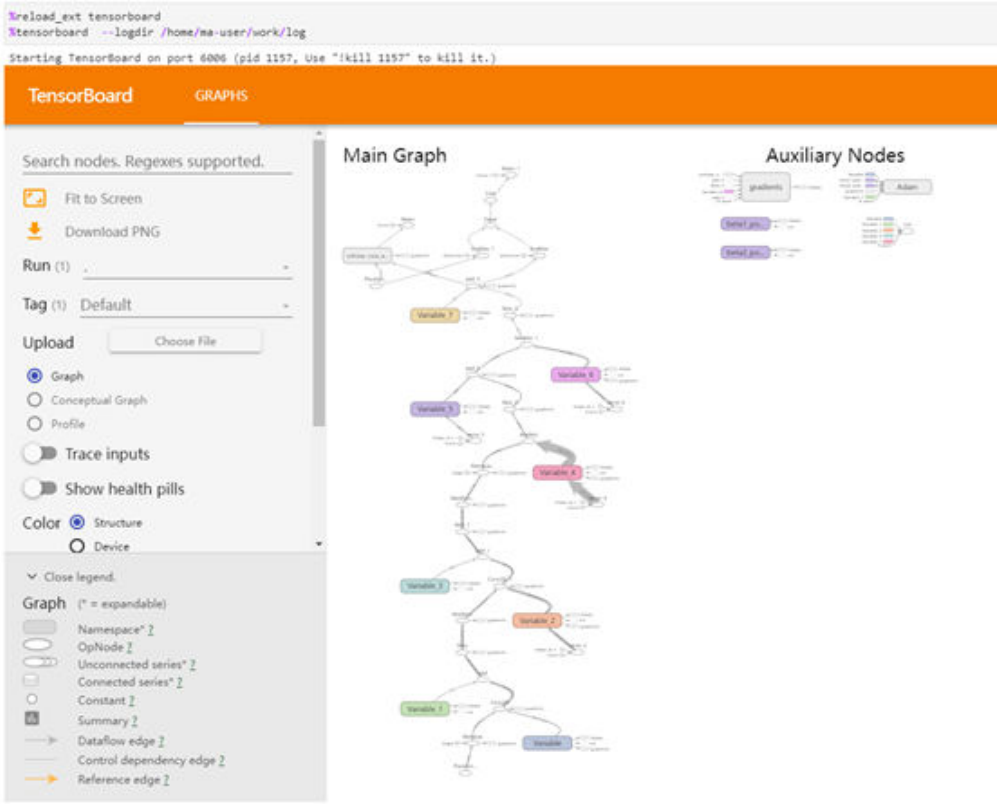
参数解释：

- --port {PORT}：指定Web可视化服务端口。可以不设置，默认使用8080端口。如果8080端口被占用了，需要在1~65535任意指定一个端口。
- --logdir {BASE_DIR}：表示数据在开发环境中的存储路径
  - 开发环境本地路径：“./work/xxx”（相对路径）或/home/ma-user/work/xxx（绝对路径）

- OBS并行文件系统的路径：obs://xxx/

例如：
#Summary数据如果是在开发环境的这个路径下/home/ma-user/work/，执行下面这条命令。
%ma_tensorboard --port {PORT} --logdir /home/ma-user/work/xxx
或者
#Summary数据如果是存在OBS并行文件系统中，执行下面这条命令，开发环境会自动挂载该OBS并行文件系统路径并读取数据。
%ma_tensorboard --port {PORT} --logdir obs://xxx/

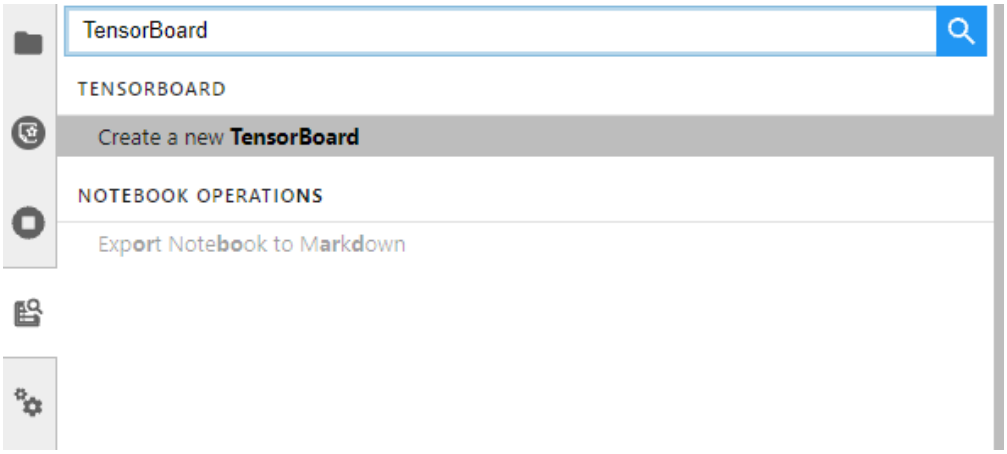
图 4-44 TensorBoard 界面（2）



方式3：

1. 单击方式3 “View > Activate Command Palette”，在搜索框中输入“TensorBoard”，再单击“Create a new TensorBoard”。

图 4-45 Create a new TensorBoard



2. 填写需要查看的Summary数据路径，或者OBS并行文件系统桶的路径。
   - 开发环境本地路径：./summary（相对路径）或/home/ma-user/work/summary（绝对路径）
   - OBS并行文件系统桶的路径：obs://xxx/

图 4-46 输入 Summary 数据路径

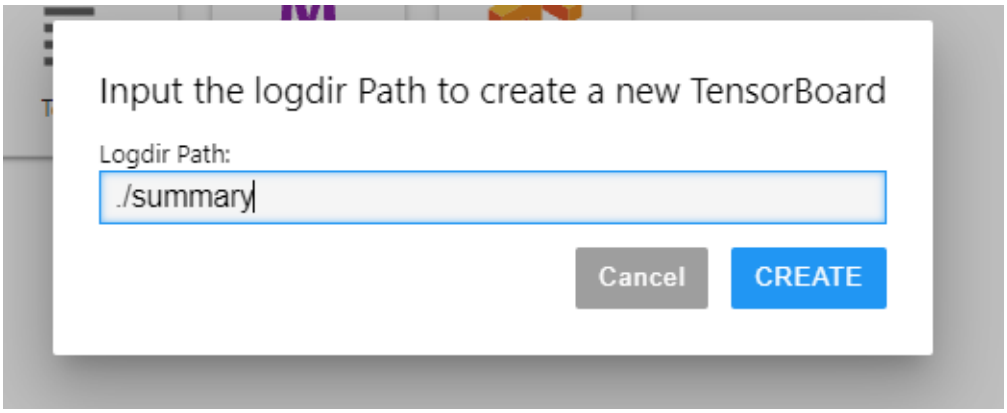
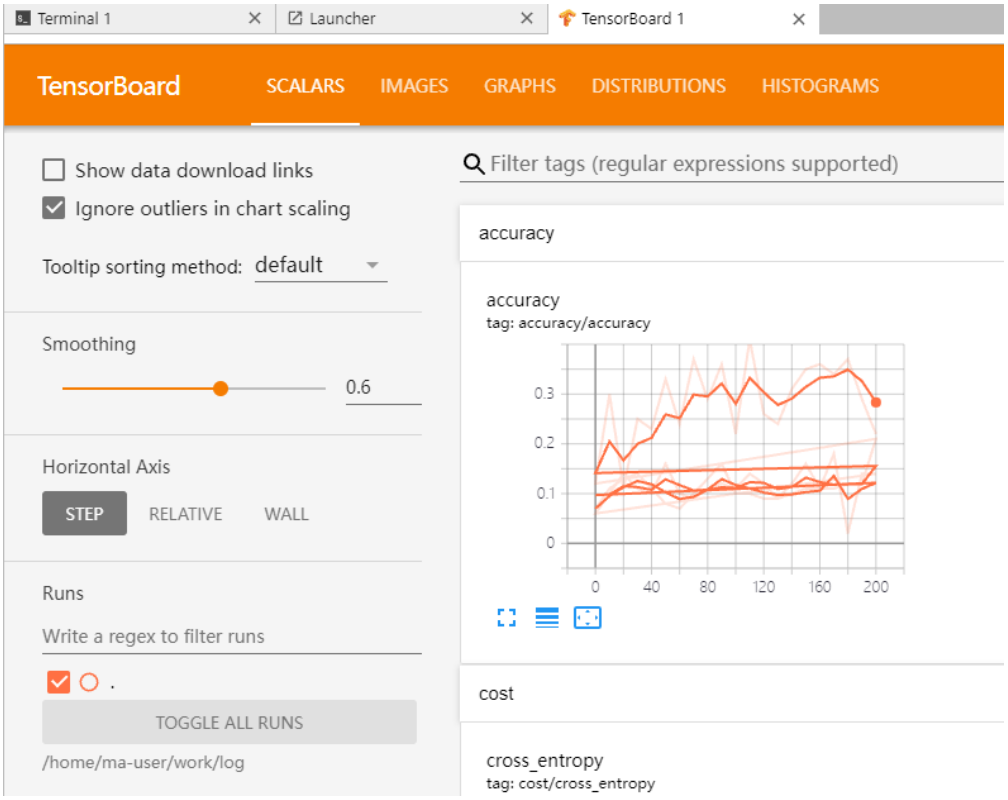


图 4-47 TensorBoard 界面（3）



方式4：

单击方式4 Terminal ，输入命令执行，但启动后不能显示UI界面。

```
tensorboard --logdir ./log
```

图 4-48 Terminal 方式打开 TensorBoard

Step4 查看训练看板中的可视化数据

训练看板是TensorBoard的可视化组件的重要组成部分，而训练看板的标签包含：标量可视化、图像可视化和计算图可视化等。

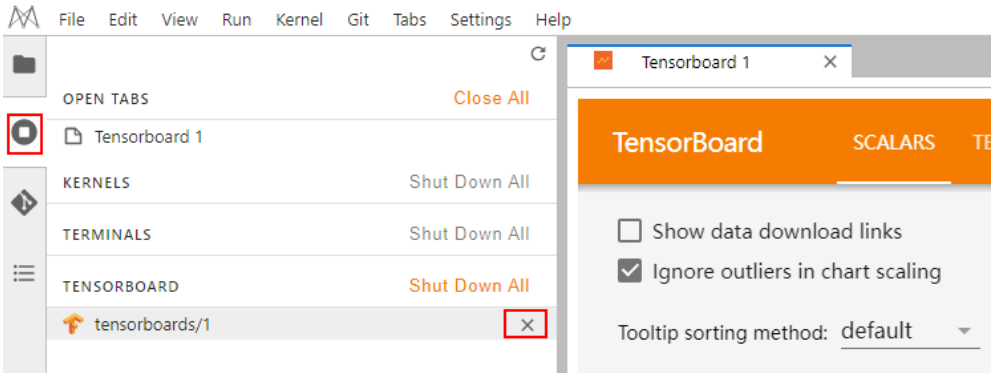
更多功能介绍请参见**TensorBoard官网资料**。

相关操作

关闭TensorBoard方式如下：

- 方式1：单击下图所示的，进入TensorBoard实例管理界面，该界面记录了所有启动的TensorBoard实例，单击对应实例后面的SHUT DOWN即可停止该实例。

图 4-49 单击 SHUT DOWN 停该实例

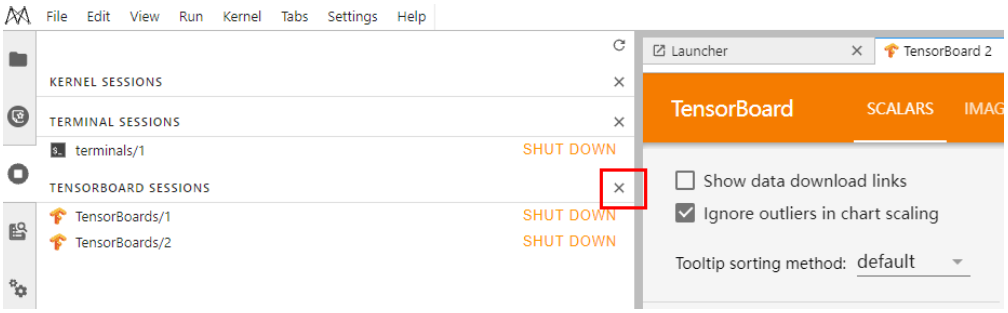


- 方式2：在开发环境JupyterLab中的“.ipynb”文件窗口中输入命令，关闭TensorBoard。PID在启动界面有提示或者通过ps -ef | grep tensorboard查看。

```
!kill PID
```

- 方式3：单击下方红框中的 按钮可以关闭所有启动的TensorBoard实例。

图 4-50 关闭所有启动的 TensorBoard 实例



- 方式4（不推荐）：直接在JupyterLab中上关闭TensorBoard窗口，此方式仅关闭可视化窗口，并未关闭后台。

4.7 Notebook 中的数据上传下载

4.7.1 上传文件至 JupyterLab

4.7.1.1 场景介绍

在AI开发过程中，如何将文件方便快速地上传到Notebook几乎是每个开发者都会遇到的问题。

ModelArts之前对文件直接上传到Notebook的大小限制是100MB，超过限制的文件无法直接上传；其次需要上传的文件并不都在本地，可能是GitHub的开源仓库，可能是类似开源数据集（**https://nodejs.org/dist/v12.4.0/node-v12.4.0-linux-x64.tar.xz**）

这样的远端文件，也可能是存放在OBS中的文件，ModelArts之前无法将这些文件直接上传到Notebook中；在文件上传过程中，用户无法获得更多的信息，例如上传进度和速度。

ModelArts上传文件特性主要解决了以上三个问题，不仅提供了更多上传文件的功能满足用户需求，而且展示了更多文件上传的细节，提升了用户的体验。

当前的文件上传功能：

- 支持上传本地文件；
- 支持Clone GitHub开源仓库；
- 支持上传OBS文件；
- 支持上传远端文件；
- 将文件上传详情可视化。

4.7.1.2 上传本地文件至 JupyterLab

4.7.1.2.1 上传场景和入口介绍

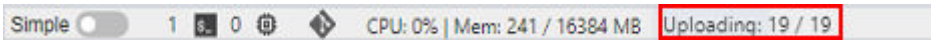
Notebook的JupyterLab中提供了多种方式上传文件。

上传文件要求

- 对于大小不超过100MB的文件直接上传，并展示文件大小、上传进度及速度等详细信息。
- 对于大小超过100MB不超过5GB的文件可以使用OBS中转，系统先将文件上传OBS（对象桶或并行文件系统），然后从OBS下载到Notebook，上传完成后，会将文件从OBS中删除。
- 5GB以上的文件上传通过调用ModelArts SDK或者Moxing完成。
- 对于Notebook当前目录下已经有同文件名称的文件，可以覆盖继续上传，也可以取消。
- 支持10个文件同时上传，其余文件显示“等待上传”。不支持上传文件夹，可以将文件夹压缩成压缩包上传至Notebook后， 在Terminal中解压压缩包。

  ```
  unzip xxx.zip  #在xxx.zip压缩包所在路径直接解压
  ```

  解压命令的更多使用说明可以在主流搜索引擎中查找Linux解压命令操作。
- 多个文件同时上传时，JupyterLab窗口最下面会显示上传文件总数和已上传文件数。

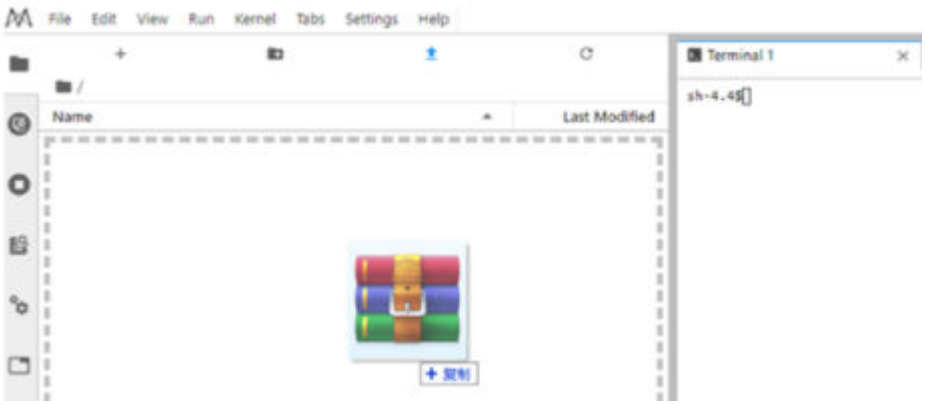


前提条件

使用JupyterLab打开一个运行中的Notebook环境。

上传文件入口 1：将文件直接拖拽至浏览器窗口中

直接将文件拖拽到JupyterLab窗口左边的空白处上传。



上传文件入口 2：通过上传图标传文件

单击窗口上方导航栏的 ModelArts Upload Files按钮，打开文件上传界面，拖拽或者选择本地文件上传。

图 4-51 上传文件图标

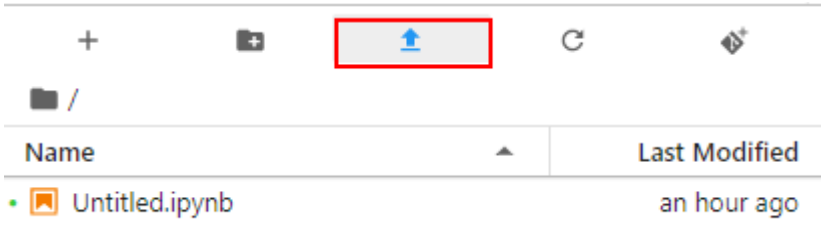


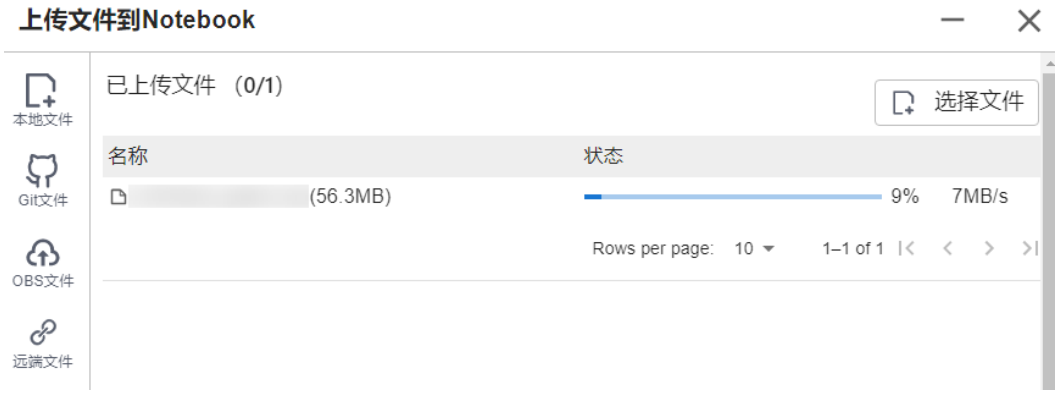
图 4-52 上传文件窗口

4.7.1.2.2 上传本地小文件（100MB 以内）至 JupyterLab

对于大小不超过100MB的文件，您可以直接上传。

“已上传文件”页面会展示目标文件大小、上传进度及速度等详细信息。

图 4-53 上传 100MB 以下小文件



文件上传完成后给出提示，显示“文件上传成功”。

图 4-54 上传成功

4.7.1.2.3 上传本地大文件（100MB~5GB）至 JupyterLab

对于大小超过100MB不超过5GB的文件可以使用OBS中转，系统先将文件上传至OBS（对象桶或并行文件系统），然后从OBS下载到Notebook。下载完成后，ModelArts会将文件自动从OBS中删除。

例如，对于下面这种情况，可以通过“OBS中转”上传。

图 4-55 通过 OBS 中转上传大文件

如果使用OBS中转需要提供一个OBS中转路径，可以通过以下三种方式提供：

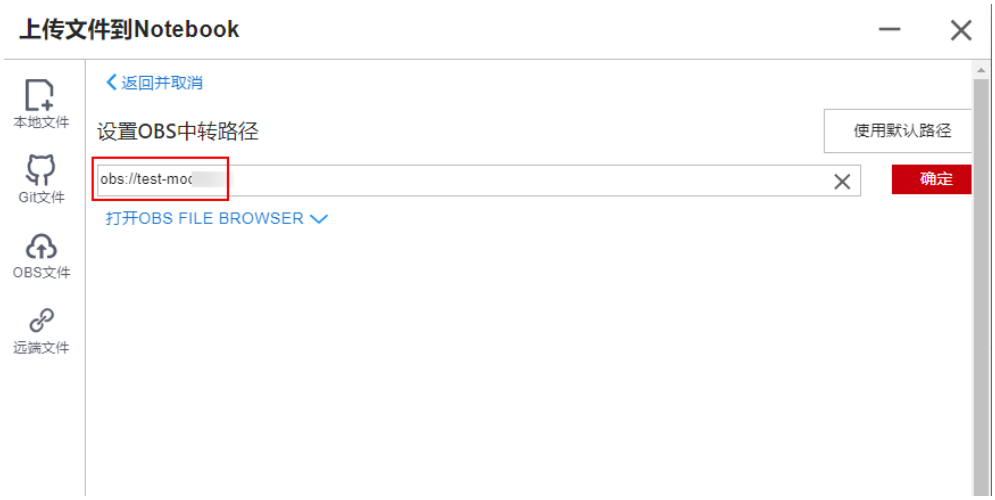
图 4-56 通过 OBS 中转路径上传

说明

仅第一次单击“OBS中转”需要提供OBS中转路径，以后默认使用该路径直接上传，可以通过上传文件窗口左下角的设置按钮 更新OBS中转路径。如图4-60所示。

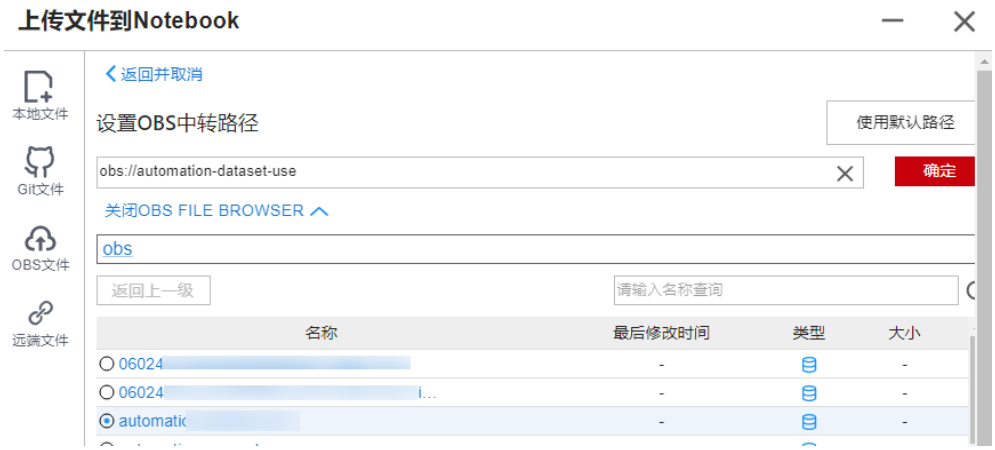
- 方式一：在输入框中直接输入有效的OBS中转路径，然后单击“确定”完成。

  图 4-57 输入有效的 OBS 中转路径



- 方式二：打开OBS File Browser选择一个OBS中转路径，然后单击“确定”完成。

图 4-58 打开 OBS File Browser



- 方式三：单击“使用默认路径”完成。

图 4-59 使用默认路径上传文件

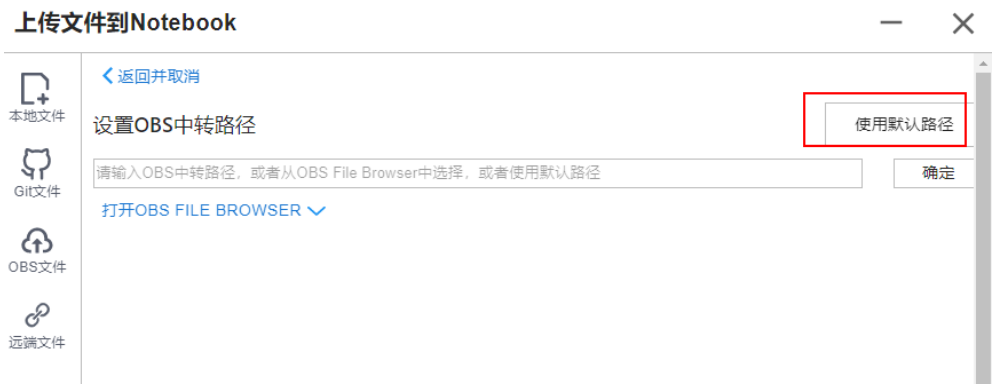
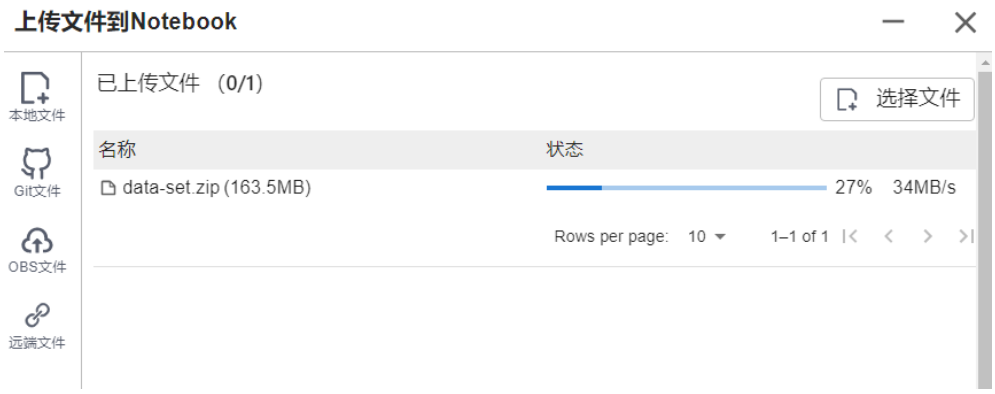


图 4-60 设置本地文件 OBS 中转路径

完成OBS中转路径设置后，开始上传文件。

图 4-61 上传文件



解压缩文件包

将文件以压缩包形式上传至Notebook JupyterLab后， 可在Terminal中解压缩文件包。

```
unzip xxx.zip  #在xxx.zip压缩包所在路径直接解压
```

解压命令的更多使用说明可以在主流搜索引擎中查找Linux解压命令操作。

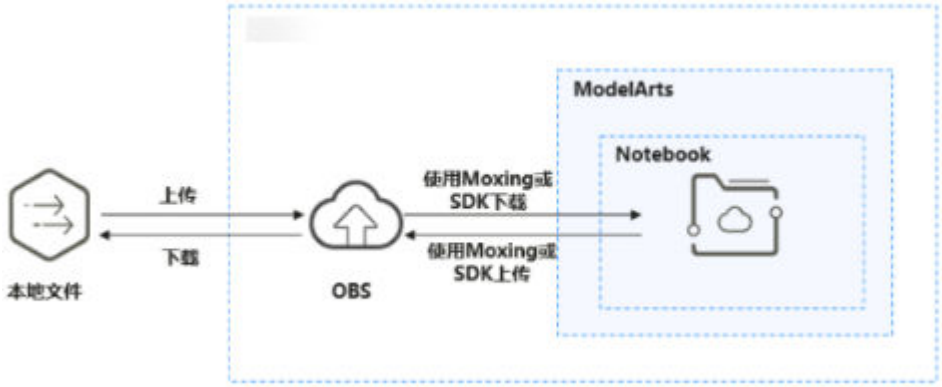
4.7.1.2.4 上传本地超大文件（5GB 以上）至 JupyterLab

不支持在Notebook的JupyterLab中直接上传大小超过5GB的文件。

图 4-62 不支持直接上传大小超过 5GB 的文件

5GB以上的文件需要先从本地上传到OBS中，再在Notebook中调用ModelArts的Moxing接口或者SDK接口读写OBS中的文件。

图 4-63 在 Notebook 中上传下载大文件



具体操作如下：

1. 从本地上传文件至OBS。具体操作请参见**上传文件至OBS桶**。
2. 将OBS中的文件下载到Notebook，可以通过在Notebook中运行代码的方式完成数据下载，具体方式有2种，ModelArts的SDK接口或者调用MoXing接口。
   - 方法一：使用ModelArts SDK接口将OBS中的文件下载到Notebook后进行操作。

     示例代码：

     ```
     from modelarts.session import Session
     session = Session()
     session.obs.copy("obs://bucket-name/obs_file.txt","/home/ma-user/work/")
     ```

   - 方法二：使用如下Moxing接口将OBS中的文件同步到Notebook后进行操作。

     ```
     import moxing as mox

     #下载一个OBS文件夹sub_dir_0，从OBS下载至Notebook
     mox.file.copy_parallel('obs://bucket_name/sub_dir_0', '/home/ma-user/work/sub_dir_0')
     #下载一个OBS文件obs_file.txt，从OBS下载至Notebook
     mox.file.copy('obs://bucket_name/obs_file.txt', '/home/ma-user/work/obs_file.txt')
     ```

     如果下载到Notebook中的是zip文件，在Terminal中执行下列命令，解压压缩包。

     ```
     unzip xxx.zip  #在xxx.zip压缩包所在路径直接解压
     ```

     代码执行完成后，参考**图4-64**打开Terminal后执行**ls /home/ma-user/work**命令查看下载到Notebook中的文件。或者在Jupyter左侧导航中显示下载的文件，如果没有显示，请刷新后查看，如**图4-65**所示。

图 4-64 打开 Terminal

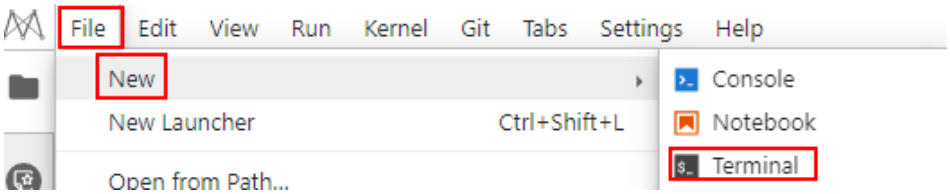
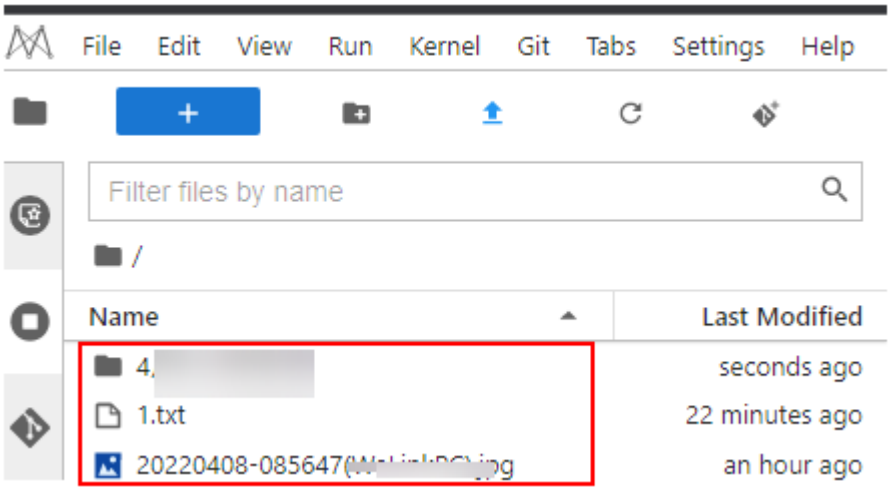


图 4-65 查看下载到 Notebook 中的文件



异常处理

通过OBS下载文件到Notebook中时，提示Permission denied。请依次排查：

- 请确保读取的OBS桶和Notebook处于同一站点区域。不支持跨站点访问OBS桶。
- 请确认操作Notebook的账号有权限读取OBS桶中的数据。

具体请参见**ModelArts中提示OBS路径错误**。

4.7.1.3 GitHub 开源仓库 Clone

在Notebook的JupyterLab中，支持从GitHub开源仓库Clone文件。

1. 通过JupyterLab打开一个运行中的Notebook。
2. 单击JupyterLab窗口上方导航栏的 ModelArts Upload Files按钮，打开文件上传窗口，选择左侧的 Git文件 进入GitHub开源仓库Clone界面。

图 4-66 上传文件图标

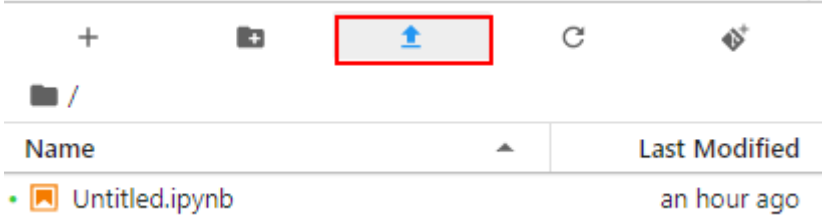
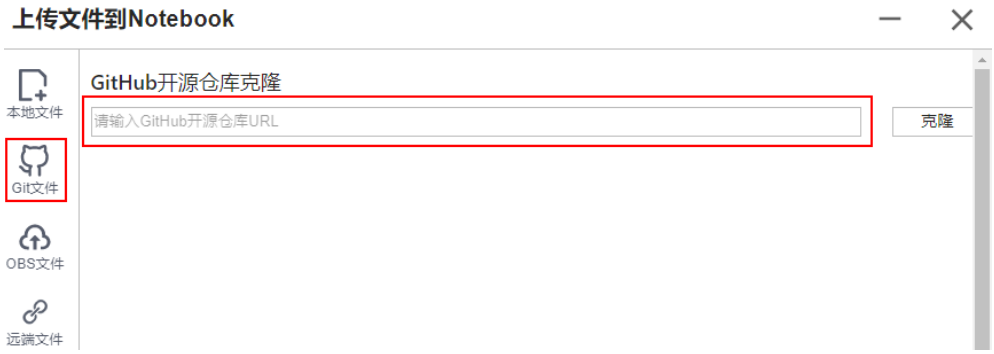


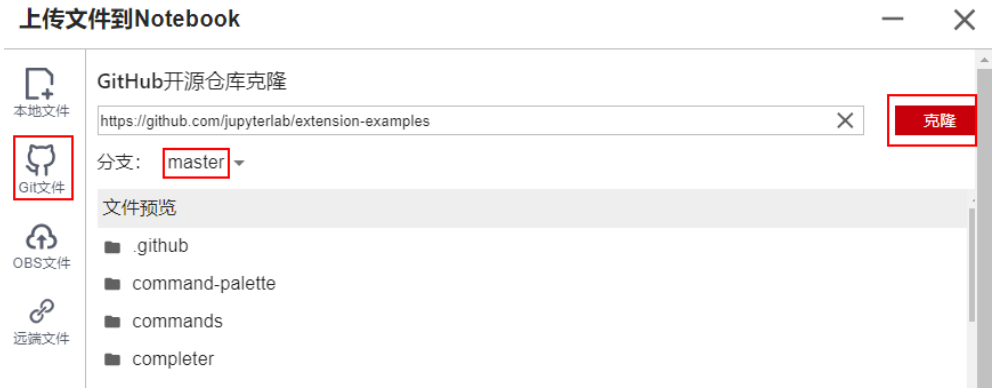
图 4-67 进入 GitHub 开源仓库 Clone 界面



3. 输入有效的GitHub开源仓库地址后会展示该仓库下的文件及文件夹，说明用户输入了有效的仓库地址，同时给出该仓库下所有的分支供选择，选择完成后单击“克隆”开始Clone仓库。

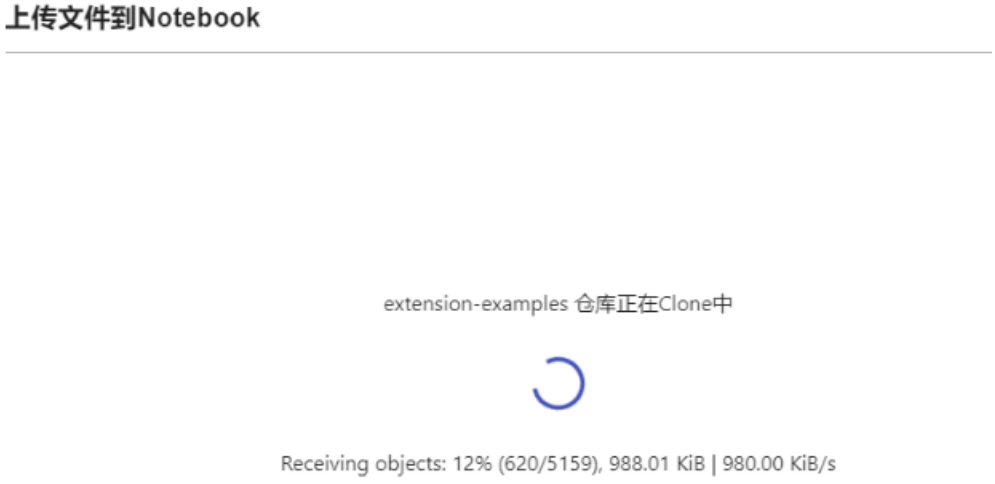
   GitHub开源仓库地址：https://github.com/jupyterlab/extension-examples

图 4-68 输入有效的 GitHub 开源仓库地址



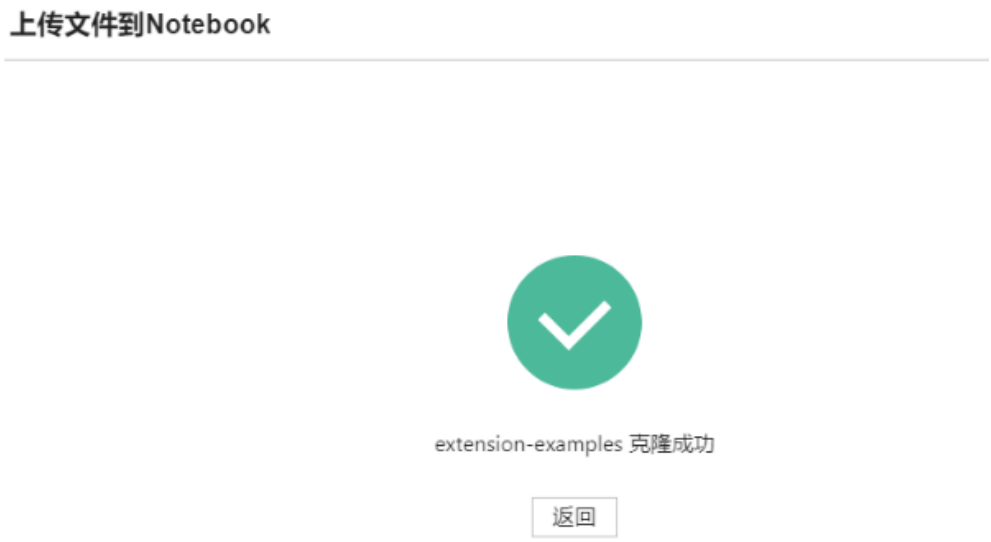
4. Clone仓库的过程中会将进度展示出来。

图 4-69 Clone 仓库的过程



5. Clone仓库成功。

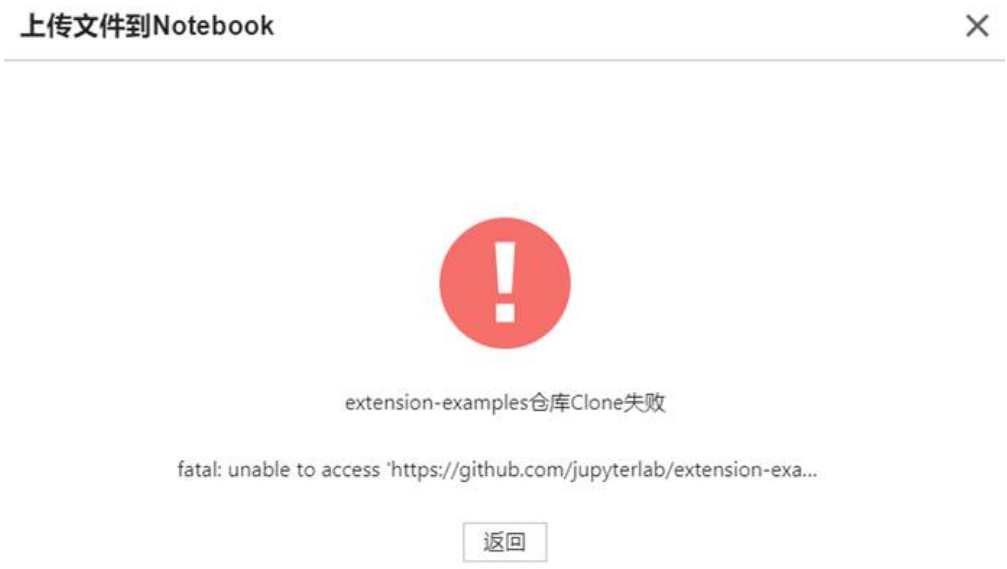
图 4-70 Clone 仓库成功



异常处理

- Clone仓库失败。可能是网络原因问题。可以在JupyterLab的Terminal中通过执行git clone https://github.com/jupyterlab/extension-examples.git测试网络连通情况。

  图 4-71 Clone 仓库失败

- 如果克隆时遇到Notebook当前目录下已有该仓库，系统给出提示仓库名称重复，此时可以单击“覆盖”继续克隆仓库，也可以单击 取消。

4.7.1.4 上传 OBS 文件到 JupyterLab

在Notebook的JupyterLab中，支持将OBS中的文件下载到Notebook。注意：文件大小不能超过10GB，否则会上传失败。

1. 通过JupyterLab打开一个运行中的Notebook。
2. 单击JupyterLab窗口上方导航栏的 ModelArts Upload Files按钮，打开文件上传窗口，选择左侧的 进入OBS文件上传界面。

图 4-72 上传文件图标

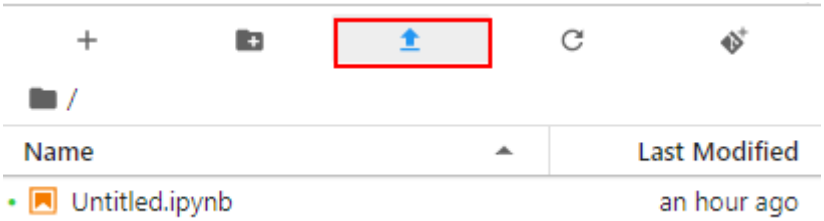
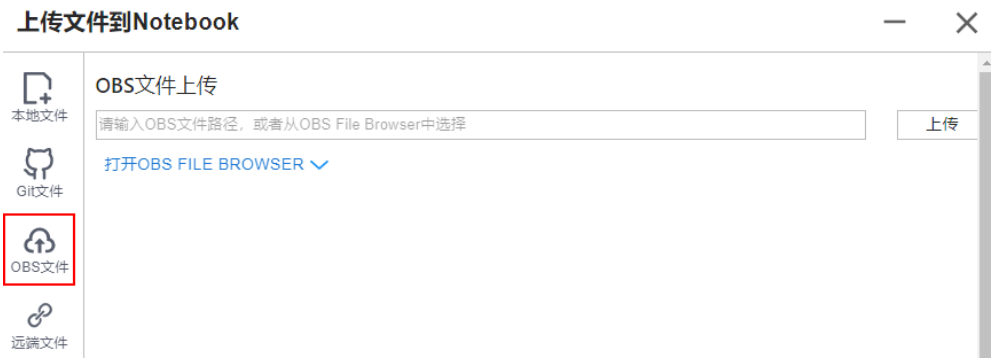
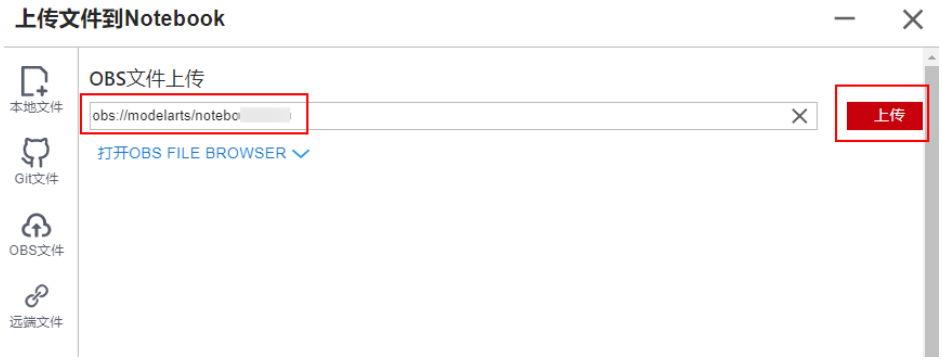


图 4-73 OBS 文件上传界面



3. 需要提供OBS文件路径，可以通过以下两种方式提供：
   - 方式一：在输入框中直接输入有效的OBS文件路径，然后单击“上传”开始传文件。

图 4-74 输入有效的 OBS 文件路径



说明

此处输入的是具体的OBS文件路径，不是文件夹的路径，否则会导致上传失败。

- 方式二：打开OBS File Browser选择OBS文件路径，然后单击“上传”，开始上传文件。

图 4-75 上传 OBS 文件

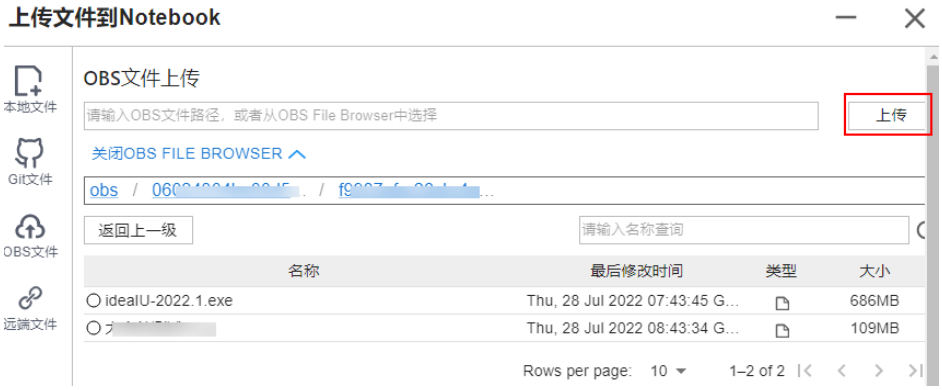
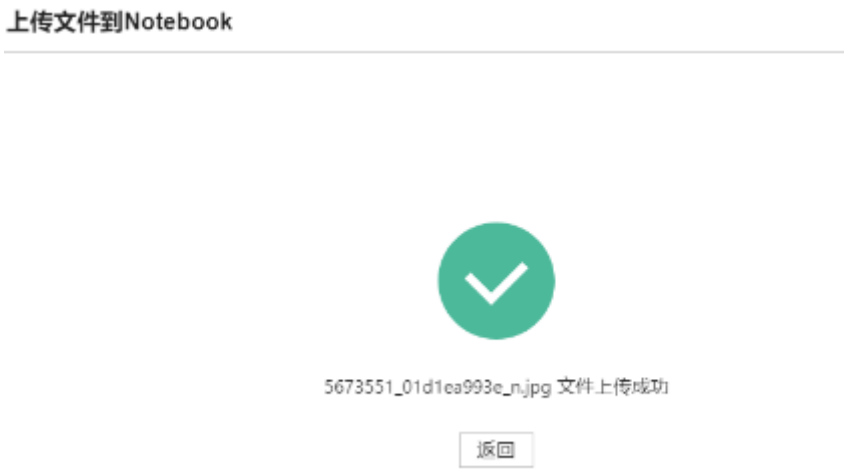


图 4-76 文件上传成功



异常处理

提示文件上传失败，有以下三种常见场景。

- **异常场景1**

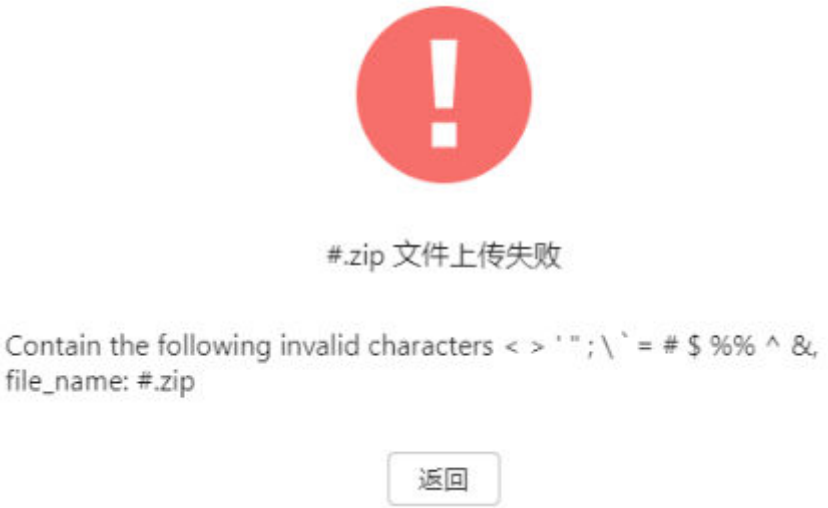
图 4-77 文件上传失败

可能原因：

- OBS路径没有设置为具体的文件路径，设置成了文件夹。
- OBS中的文件设置了加密。请前往OBS控制台查看，确保该文件未加密。
- OBS桶和Notebook不在同一个区域。请确保读取的OBS桶和Notebook处于同一站点区域，不支持跨站点访问OBS桶。具体操作请参见**如何查看OBS桶与ModelArts是否在同一区域**。
- 没有该OBS桶的访问权限。请确认操作Notebook的账号有权限读取OBS桶中的数据。具体操作请参见**检查您的账号是否有该OBS桶的访问权限**。
- OBS文件被删除。请确认待上传的OBS文件是否存在。

- **异常场景2**

图 4-78 文件上传失败



可能原因：
文件名包含<>'";\`=#$%^&等特殊字符。

- **异常场景3**

图 4-79 文件上传失败

可能原因：
文件大小超过10GB导致上传失败。

4.7.1.5 上传远端文件至 JupyterLab

在Notebook的JupyterLab中，支持通过远端文件地址下载文件。

要求：远端文件的URL粘贴在浏览器的输入框中时，可以直接下载该文件。

1. 通过JupyterLab打开一个运行中的Notebook。
2. 单击JupyterLab窗口上方导航栏的 ModelArts Upload Files按钮，打开文件上传窗口，选择左侧的 进入远端文件上传界面。

图 4-80 上传文件图标

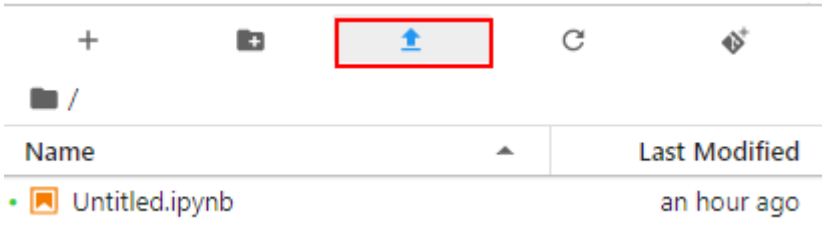


图 4-81 进入远端文件上传界面

3. 输入有效的远端文件URL后，系统会自动识别上传文件名称，单击“上传”，开始上传文件。

图 4-82 输入有效的远端文件 URL

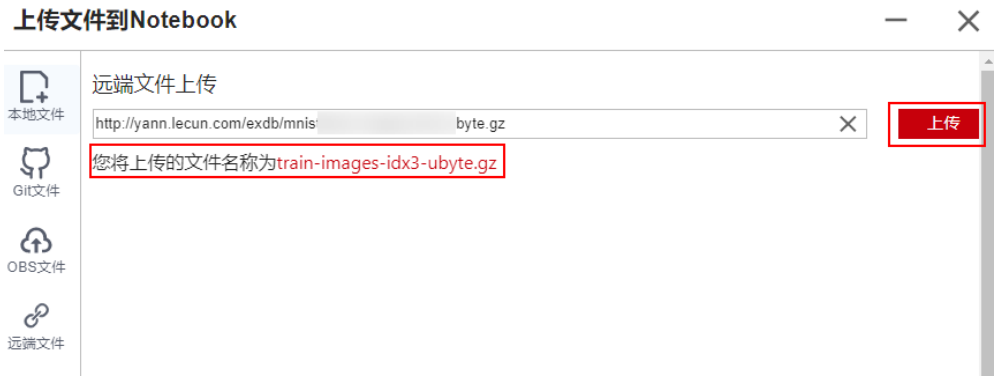
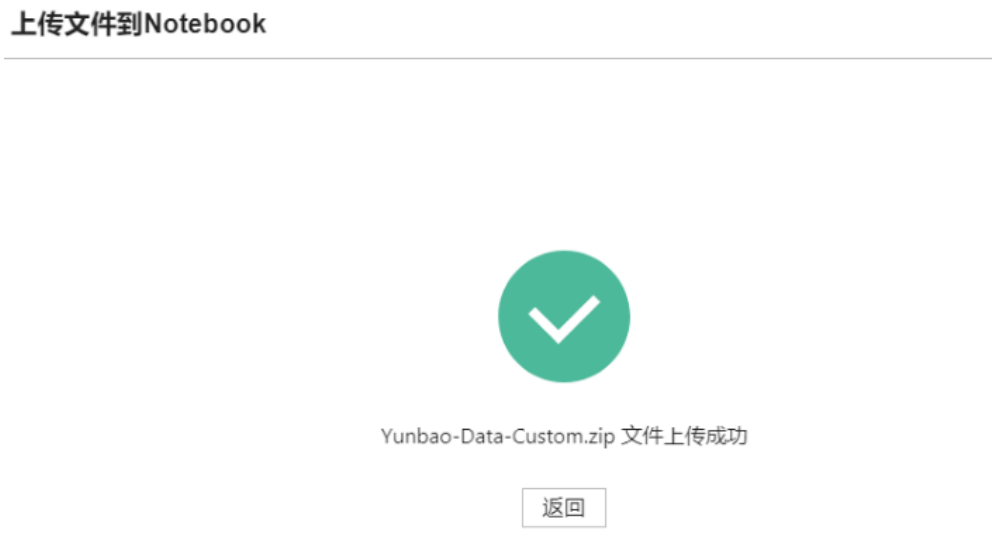


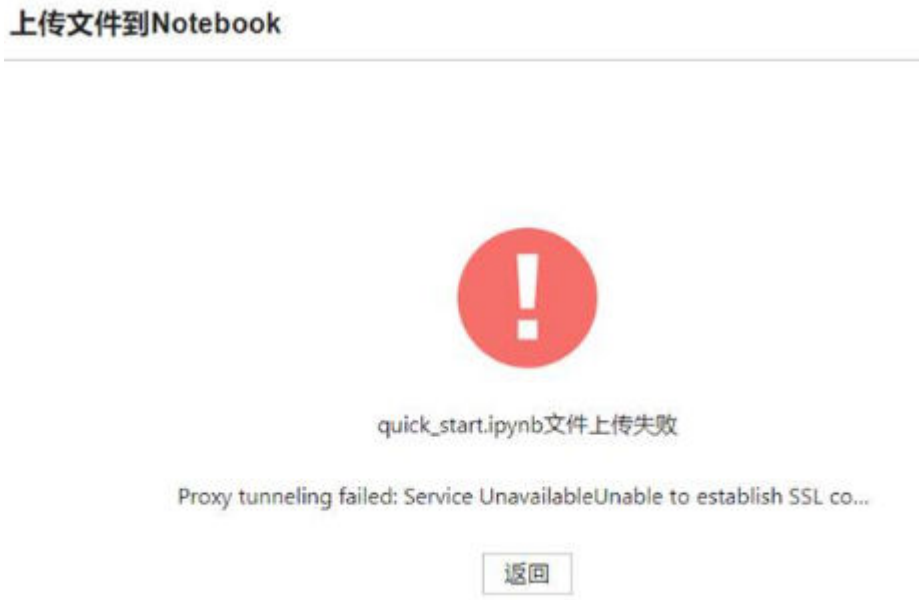
图 4-83 远端文件上传成功



异常处理

远端文件上传失败。可能是网络原因。请先在浏览器中输入该远端文件的URL地址，测试该文件是否能下载。

图 4-84 远端文件上传失败



4.7.2 从 JupyterLab 下载文件至本地

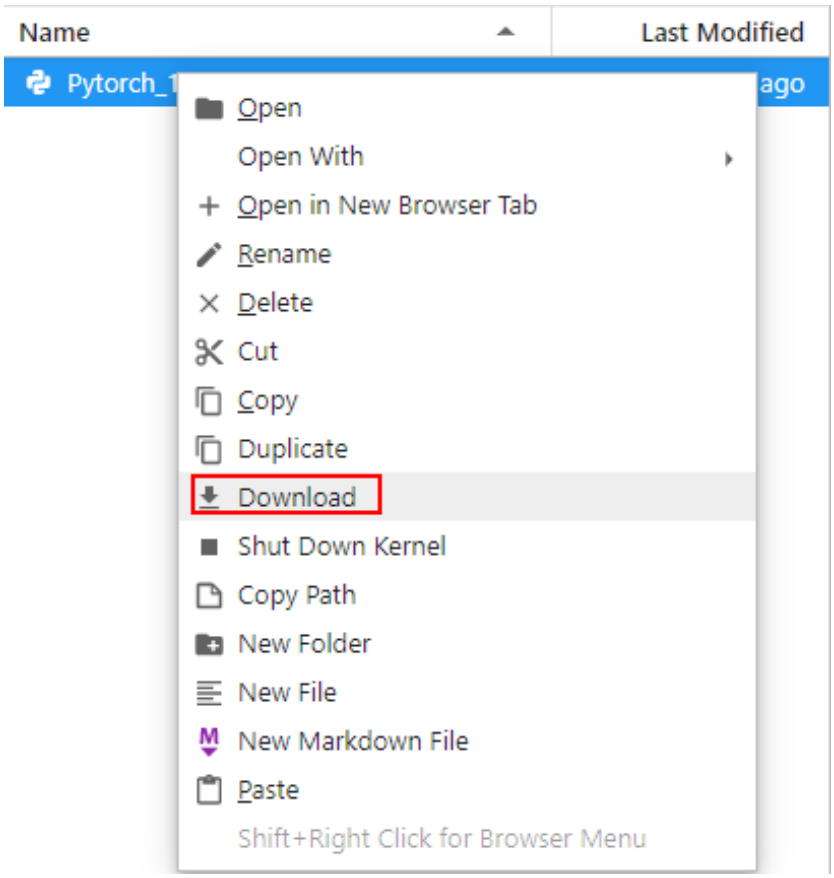
在JupyterLab中开发的文件，可以下载至本地。

- 不大于100MB的文件，可以直接从JupyterLab中下载到本地，具体操作请参见**从JupyterLab中下载不大于100MB的文件至本地**。
- 大于100MB的文件，需要先从JupyterLab上传到OBS，再通过OBS下载到本地，具体操作请参见**从JupyterLab中下载大于100MB的文件到本地**。

从 JupyterLab 中下载不大于 100MB 的文件至本地

在JupyterLab文件列表中，选择需要下载的文件，单击右键，在操作菜单中选择“Download”下载至本地。下载的目的路径，为您本地浏览器设置的下载目录。

图 4-85 下载文件



从 JupyterLab 中下载大于 100MB 的文件到本地

大于100MB的文件需要先从Notebook中上传到OBS，再从OBS下载到本地，具体操作如下：

1. 在Notebook中，新建一个大于100MB的“ipynb”文件，使用MoXing先将该文件从Notebook上传到OBS中，示例代码如下：

   ```
   import moxing as mox
   mox.file.copy('/home/ma-user/work/obs_file.txt', 'obs://bucket_name/obs_file.txt')
   ```

   其中“/home/ma-user/work/obs_file.txt”为文件在Notebook中的存储路径，“obs://bucket_name/obs_file.txt”为该文件上传到OBS的存储路径，其中“bucket_name”为OBS中创建的桶的名称，“obs_file.txt”为上传的文件。
2. 使用OBS或ModelArts SDK将OBS中的文件下载到本地。
   - 方式一：使用OBS进行下载
   - 在OBS中，可以将样例中的“obs_file.txt”下载到本地。如果您的数据较多，推荐OBS Browser+下载数据或文件夹。使用OBS下载文件的操作指导参见**下载文件**
   - 方式二：使用ModelArts SDK进行下载
     i. 在您的本地环境**下载并安装ModelArts SDK**。
     ii. 完成ModelArts SDK的**Session鉴权**。
     iii. 将OBS中的文件下载到本地，详请参见**从OBS下载数据**。示例代码如下：

     ```
     from modelarts.session import Session
     ```

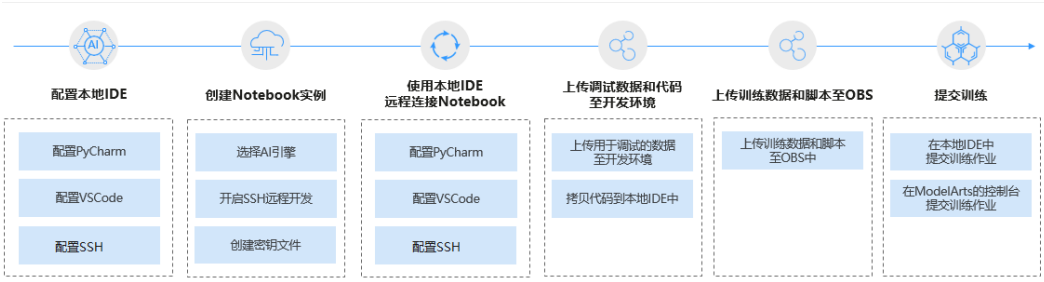
```
# 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；
# 本示例以ak和sk保存在环境变量中来实现身份验证为例，运行本示例前请先在本地环境中设置环境变量HUAWEICLOUD_SDK_AK和HUAWEICLOUD_SDK_SK。
__AK = os.environ["HUAWEICLOUD_SDK_AK"]
__SK = os.environ["HUAWEICLOUD_SDK_SK"]
# 如果进行了加密还需要进行解密操作
session = Session(access_key=__AK,secret_key=__SK, project_id='***', region_name='***')

session.download_data(bucket_path="/bucket_name/obs_file.txt",path="/home/user/obs_file.txt")
```

5 本地 IDE

5.1 本地 IDE 操作流程

ModelArts支持通过本地IDE环境远程连接到Notebook中，开发基于PyTorch、TensorFlow和MindSpore引擎的AI模型。具体操作流程如下图所示。

图 5-1 使用本地 IDE 开发流程



1. 配置本地IDE

   在用户的PC端配置本地IDE环境。

   支持通过**PyCharm**、**VS Code**、**SSH工具**本地IDE连接云上Notebook。PyCharm和VS Code可以使用插件自动化配置，也可以手工配置。

2. **创建Notebook实例**

   在ModelArts控制台上创建一个Notebook开发环境实例，选择要使用的AI框架，并开启SSH远程开发功能。

3. 使用本地IDE远程连接到ModelArts的开发环境中。
4. **上传数据和代码至开发环境中**，进行代码调试。
   - 代码直接复制至本地IDE中即可，本地IDE中会自动同步至云上开发环境。
   - 不大于500MB数据量直接复制至本地IDE中即可。
   - 创建训练作业大于500MB数据量请先上传到OBS中，从OBS上传到云硬盘EVS。
5. 将调试好的训练脚本和用于训练的数据集上传至OBS目录。
6. 提交训练作业。提交训练作业方式如下：

- 在本地IDE中提交训练作业
  可以通过调用ModelArts提供的SDK，创建训练作业，上云训练，调用SDK创建训练作业的操作请参见**调用SDK创建训练作业**。
- 在ModelArts的Console控制台页面中提交训练作业。

5.2 本地 IDE（PyCharm）

5.2.1 PyCharm Toolkit 插件连接 Notebook

5.2.1.1 PyCharm ToolKit 介绍

由于AI开发者会使用PyCharm工具开发算法或模型，为方便快速将本地代码提交到ModelArts的训练环境，ModelArts提供了一个PyCharm插件工具PyCharm ToolKit（插件下载请参见**通过Marketplace安装**），协助用户完成代码上传、提交训练作业、将训练日志获取到本地展示等，用户只需要专注于本地的代码开发即可。

使用限制

- 当前仅支持PyCharm 2019.2及以上版本，包括社区版和专业版。
- 使用PyCharm ToolKit远程连接Notebook开发环境，仅限PyCharm专业版。
- 使用PyCharm ToolKit提交训练作业，社区版和专业版都支持。
- PyCharm ToolKit工具仅支持Windows版本的PyCharm。

支持的功能

表 5-1 ToolKit（latest）功能列表

支持的功能	说明	对应操作指导
SSH远程连接	支持SSH远程连接ModelArts的Notebook开发环境。	**配置PyCharm ToolKit远程连接Notebook**
训练模型	支持将本地开发的代码，快速提交至ModelArts并自动创建训练作业，在训练作业运行期间获取训练日志并展示到本地。	• **提交训练作业** • **停止训练作业** • **查看训练日志**
OBS上传下载	上传本地文件或文件夹至OBS，从OBS下载文件或文件夹到本地。	**在PyCharm中上传数据至Notebook**

5.2.1.2 下载并安装 ToolKit 工具

在使用PyCharm ToolKit之前，您需要根据如下操作指导完成在PyCharm中的安装配置。

前提条件

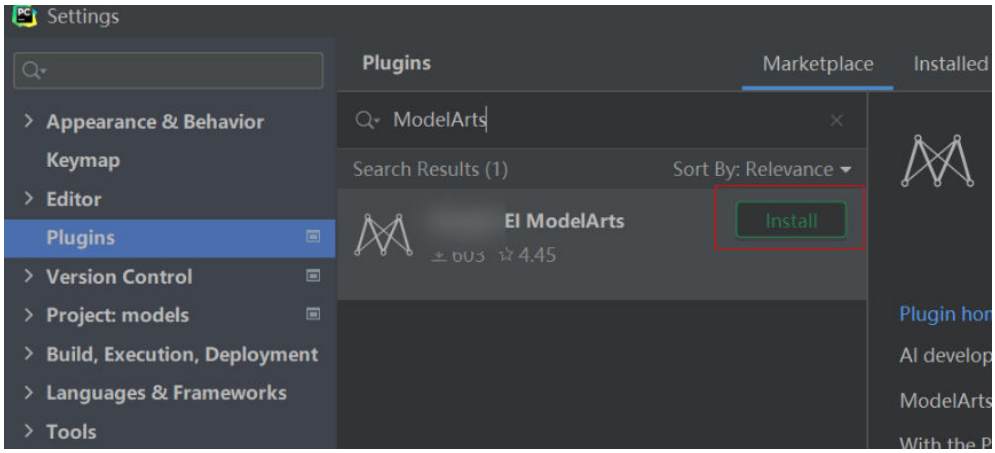
本地已安装2019.2及以上版本的PyCharm社区版或专业版。

- 使用PyCharm ToolKit远程连接Notebook开发环境，仅限PyCharm专业版。
- 使用PyCharm ToolKit提交训练作业，社区版和专业版都支持。

通过 Marketplace 安装

在PyCharm中选择“File > Settings > Plugins”，在Marketplace里搜索“ModelArts”，单击“Install”即可完成安装。

图 5-2 通过 Marketplace 安装



说明

- 通过该方式安装的PyCharm ToolKit为latest版本。
- 如果Marketplace里搜索不到ModelArts，可能是用户网络限制原因，请确保可以正常访问外网。

5.2.1.3 PyCharm ToolKit 连接 Notebook

ModelArts提供了一个PyCharm插件工具PyCharm ToolKit，协助用户完成SSH远程连接Notebook、代码上传、提交训练作业、将训练日志获取到本地展示等，用户只需要专注于本地的代码开发即可。

前提条件

本地已安装2019.2及以上版本的PyCharm专业版。SSH远程开发功能只限PyCharm专业版。单击**PyCharm工具下载地址**下载并完成安装。

说明

请下载**2023.2**或之前版本的PyCharm专业版工具。PyCharm toolkit未适配2023.2之后版本的PyCharm专业版工具。

Step1 创建 Notebook 实例

创建一个Notebook实例，并开启远程SSH开发，配置远程访问IP白名单。该实例状态必须处于“运行中”，具体参见**创建Notebook实例**章节。

Step2 下载并安装 PyCharm ToolKit

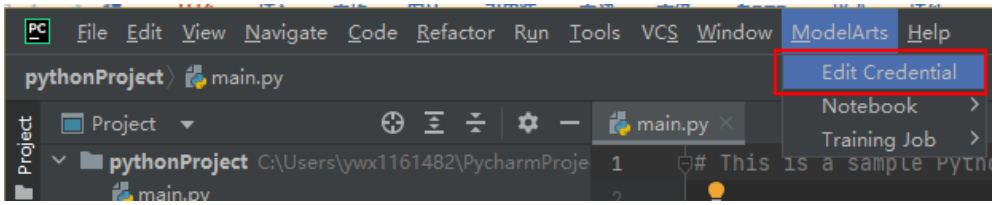
在PyCharm中选择“File > Settings > Plugins”，在Marketplace里搜索“ModelArts”，单击“Install”即可完成安装。具体下载和安装过程请参见**下载并安装ToolKit工具**。

Step3 登录插件

使用访问密钥完成登录认证操作如下：

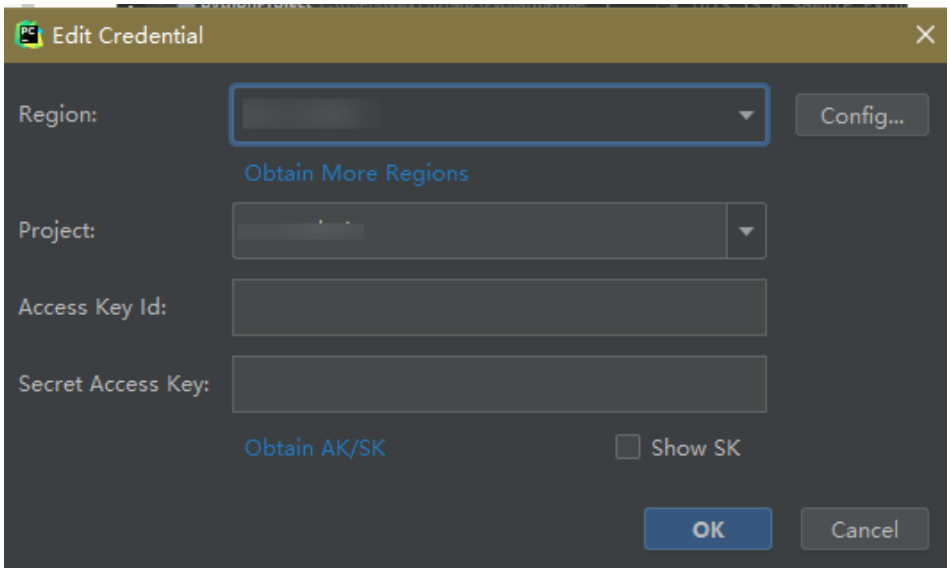
1. 打开已安装ToolKit工具的PyCharm，在菜单栏中选择“ModelArts > Edit Credential”。

   图 5-3 Edit Credential

2. 在弹出的对话框中，选择您使用的ModelArts所在区域、填写AK、SK（获取方式**参考链接**），然后单击“OK”完成登录。
   - “Region”：从下拉框中选择区域。必须与ModelArts管理控制台在同一区域。
   - “Project”：Region选择后，Project自动填充为Region对应的项目。
   - “Access Key ID”：填写访问密钥的AK。
   - “Secret Access Key”：填写访问密钥的SK。

   图 5-4 填写区域和访问密钥

3. 查看认证结果。

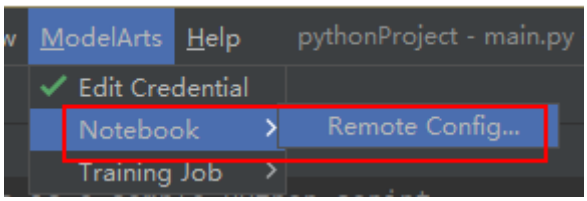
   在Event Log区域中，当提示如下类似信息时，表示访问密钥添加成功。

   ```
   16:01Validate Credential Success: The HUAWEI CLOUDcredential is valid.
   ```

Step4 插件自动化配置

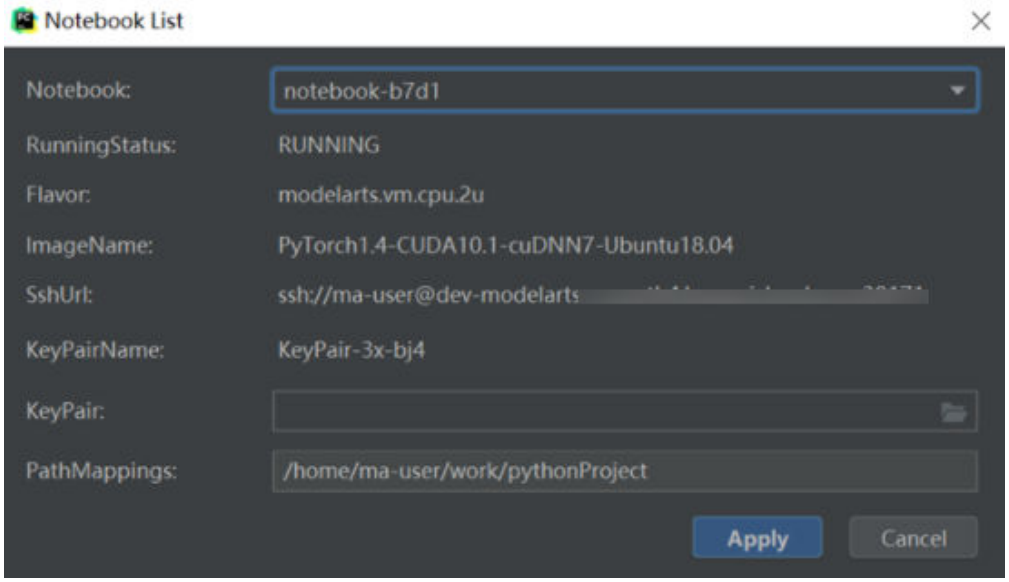
1. 在本地的PyCharm开发环境中，单击“ModelArts > Notebook > Remote Config...”，配置插件。

   图 5-5 配置插件

2. 此时，会出现该账号已创建的所有包含SSH功能的Notebook列表，下拉进行选择对应Notebook。

   图 5-6 Notebook 列表

   - KeyPair: 需要选择保存在本地的Notebook对应的keypair认证。即创建Notebook时创建的密钥对文件，创建时会直接保存到浏览器默认的下载文件夹中。
   - PathMappings: 该参数为本地IDE项目和Notebook对应的同步目录，默认为/home/ma-user/work/project名称，可根据自己实际情况更改。

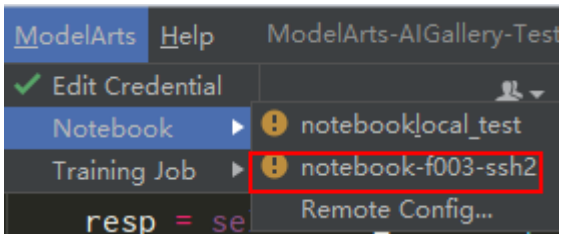
3. 单击“Apply”，配置完成后，重启IDE生效。

   重启后初次进行update python interpreter需要耗费20分钟左右。

Step5 使用插件连接云上 Notebook

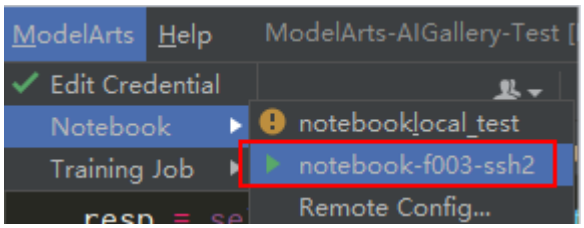
与Notebook断开连接的状态下，单击Notebook名称，根据提示启动本地IDE与Notebook的连接(默认启动时间4小时)。

图 5-7 启动连接 Notebook



连接状态下，单击Notebook名称，根据提示断开本地IDE与云上Notebook的连接。

图 5-8 停止连接 Notebook



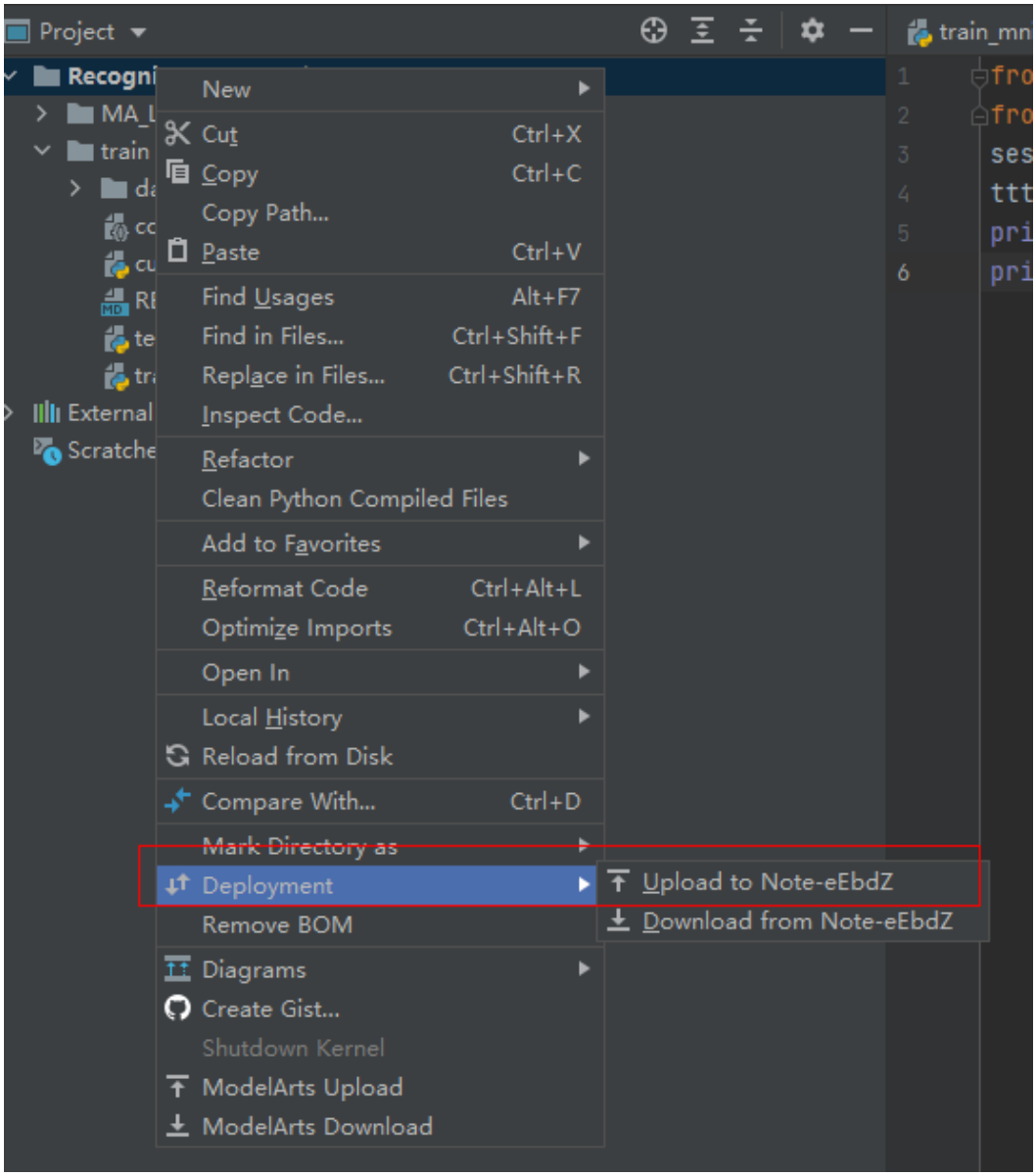
Step6 同步上传本地文件至 Notebook

本地文件中的代码直接复制至本地IDE中即可，本地IDE中会自动同步至云上开发环境。

初始化同步：

在本地IDE的Project目录下，单击右键，选择“Deployment”，单击“Upload to xxx”（Notebook名称），将本地工程文件上传至指定的Notebook。

图 5-9 同步本地文件至 Notebook

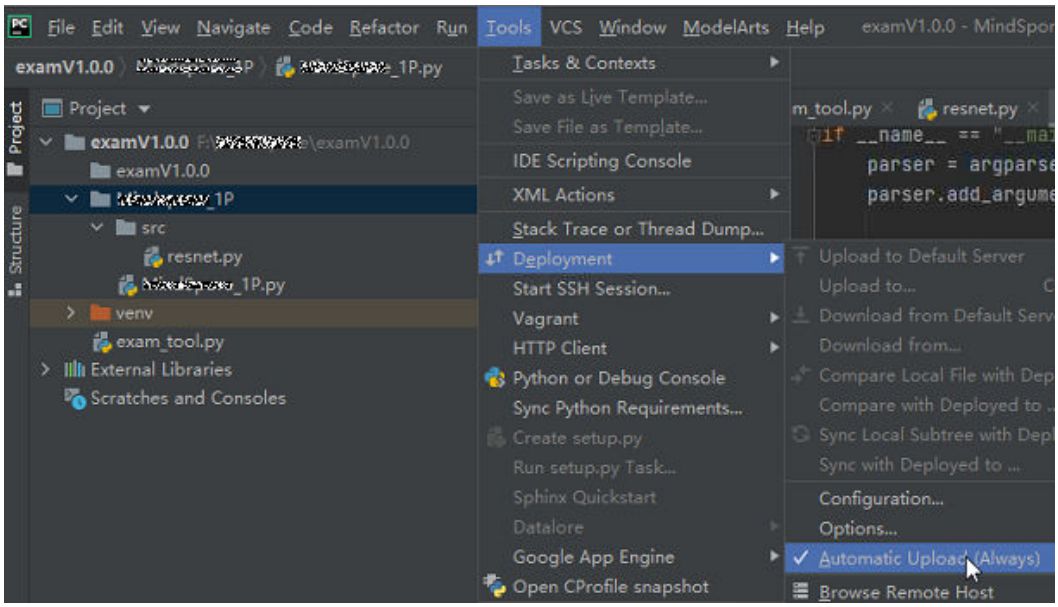


后续同步:

只需修改代码后保存（ctrl+s），即可进行自动同步。

插件安装完成后在本地IDE中开启了“Automatic Upload”，本地目录中的文件会自动上传至云端开发环境Notebook。如果未开启，请参考下图开启自动上传。

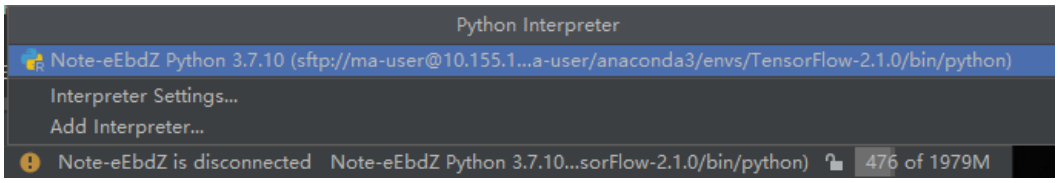
图 5-10 开启自动上传



Step7 远程调试

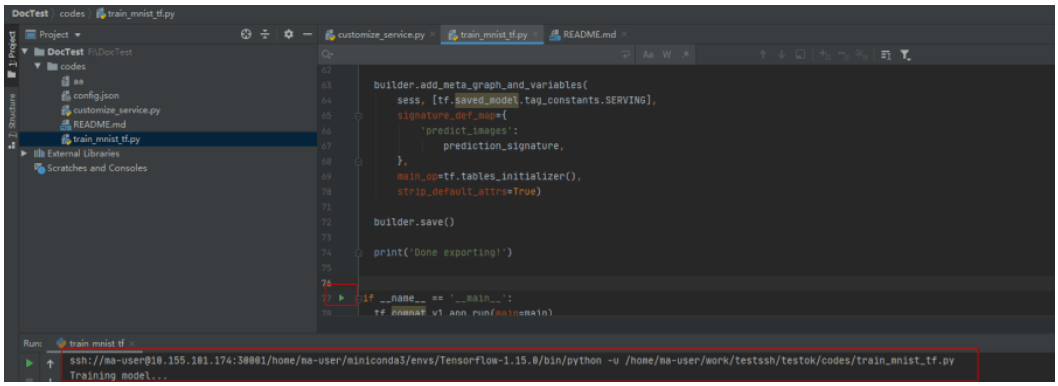
单击本地IDE右下角interpreter，选择Notebook的python解释器。

图 5-11 选择 Python 解释器



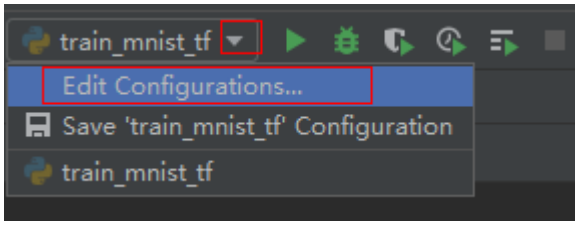
像本地运行代码一样，直接单击运行按钮运行代码即可，此时虽然是在本地IDE点的运行按钮，实际上运行的是云端Notebook里的代码，日志可以回显在本地的日志窗口。

图 5-12 查看运行日志



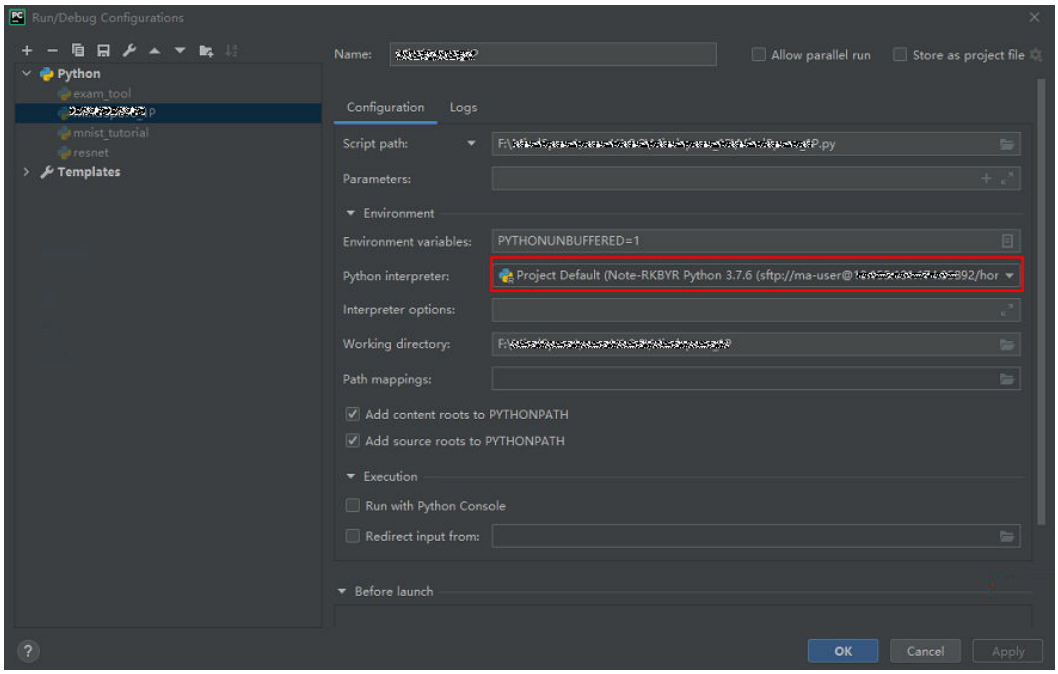
也可以单击本地IDE右上角的Run/Debug Configuration按钮来设置运行参数。

图 5-13 设置运行参数（1）



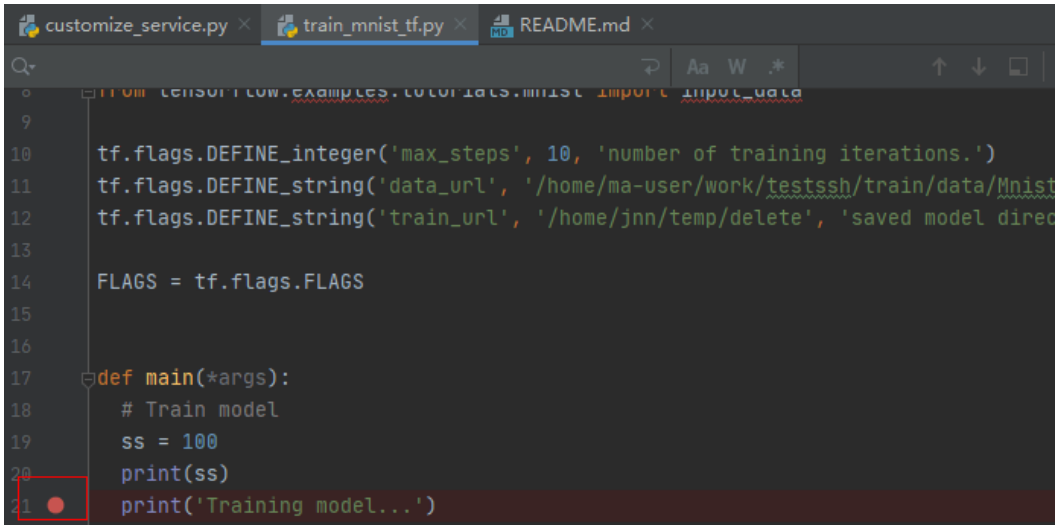
选择远程连接到云上开发环境实例对应的Python解释器。

图 5-14 设置运行参数（2）



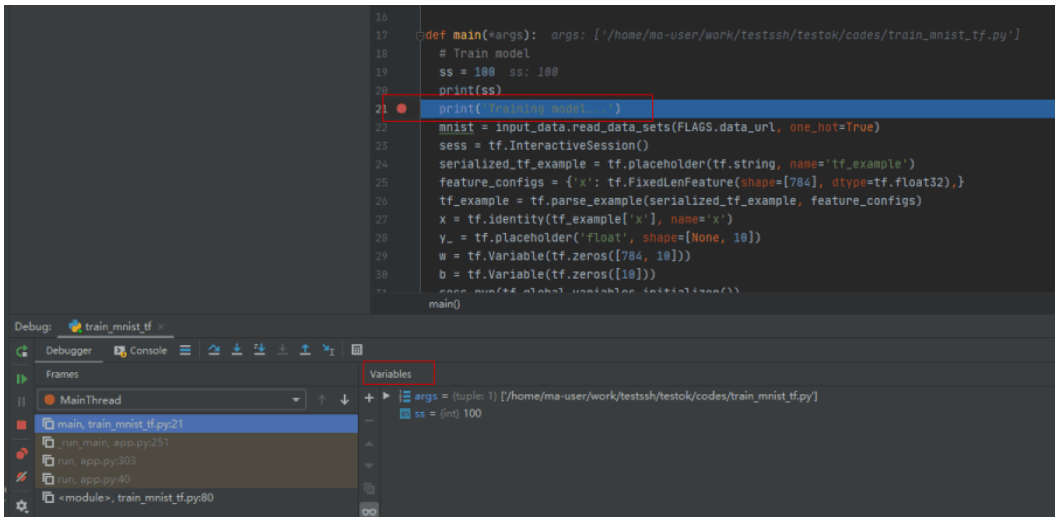
当需要调试代码时，可以直接打断点，然后使用debug方式运行程序。

图 5-15 使用 debug 方式运行程序



此时可以进入debug模式，代码运行暂停在该行，且可以查看变量的值。

图 5-16 Debug 模式下查看变量值



5.2.2 PyCharm 手动连接 Notebook

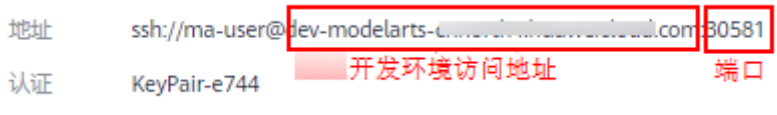
本地IDE环境支持Pycharm和VS Code。通过简单配置，即可用本地IDE远程连接到ModelArts的Notebook开发环境中，调试和运行代码。

本章节介绍基于PyCharm环境访问Notebook的方式。

前提条件

- 本地已安装2019.2及以上版本的PyCharm专业版。SSH远程调试功能只限PyCharm专业版。
- 创建一个Notebook实例，并开启远程SSH开发。该实例状态必须处于“运行中”，具体参见**创建Notebook实例**章节。
- 在Notebook实例详情页面获取开发环境IP地址和端口号。

  图 5-17 Notebook 实例详情页面

- 准备好密钥对。

  密钥对在用户第一次创建时，自动下载，之后使用相同的密钥时不会再有下载界面（用户一定要保存好），或者每次都使用新的密钥对。

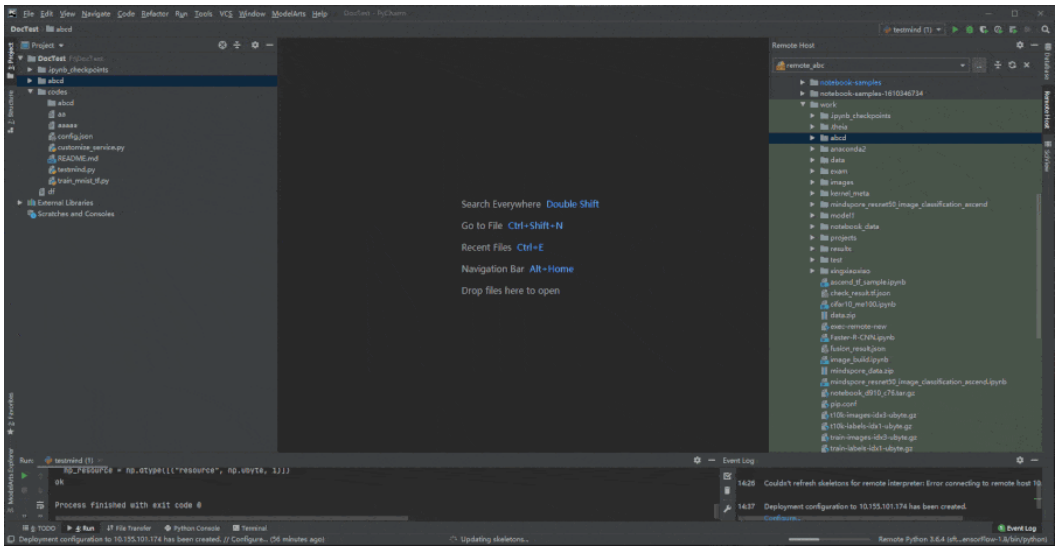
Step1 配置 SSH

1. 在本地的PyCharm开发环境中，单击File -> Settings -> Tools -> SSH Configurations，单击+号，增加一个SSH连接配置。
   - Host: 云上开发环境的IP地址，即在开发环境实例页面远程访问模块获取的IP地址。

- Port: 云上开发环境的端口，即在开发环境实例页面远程访问模块获取的端口号。
- User name: 固定为ma-user。
- Authentication type: Key pair方式。
- Private key file: 存放在本地的云上开发环境私钥文件，即在创建开发环境实例时创建并保存的密钥对文件。

2. 单击  将连接重命名，可以自定义一个便于识别的名字，单击OK。
3. 配置完成后，单击Test Connection测试连通性。
4. 选择Yes，显示Successfully connected表示网络可以连通，单击OK。
5. 在最下方再单击OK保存配置。

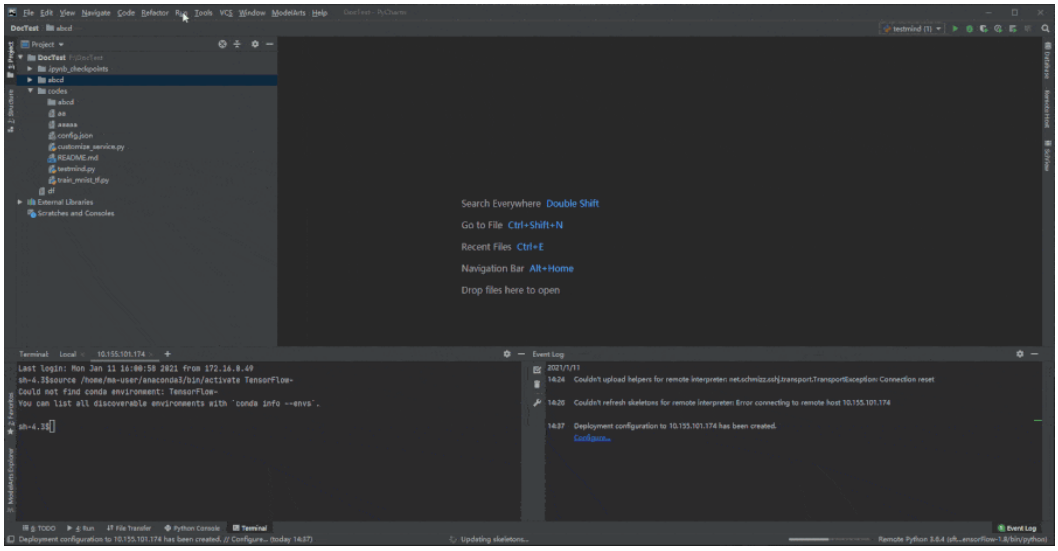
图 5-18 配置 SSH



Step2 获取开发环境预置虚拟环境路径

1. 单击“Tools > Start SSH Session”，则可连接到云端开发环境内。
2. 执行如下命令可在/home/ma-user/下面的README文件查看当前环境内置的Python虚拟环境。
   ```
   cat /home/ma-user/README
   ```
3. 执行source命令可以切换到具体的Python环境中。
4. 执行which python查看python路径并复制出来，以备后续配置云上Python Interpreter使用。

图 5-19 获取开发环境预置虚拟环境路径



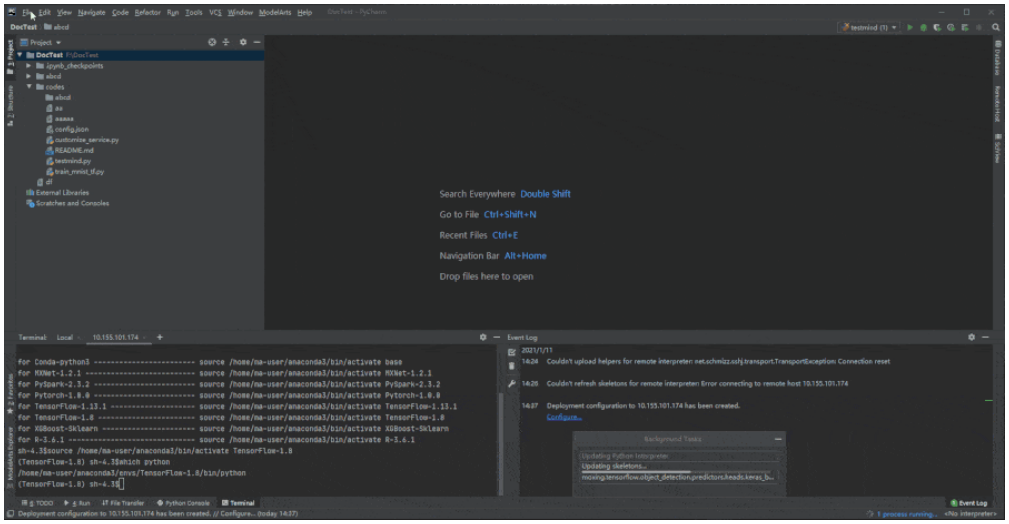
Step3 配置云上 Python Interpreter

1. 单击“File > Settings > Project：PythonProject > Python Interpreter”，单击设置图标，再单击“Add”，添加一个新的interpreter。
2. 选择“Existing server configuration”，在下拉菜单中选择上一步配置好的SSH configuration，单击“Next”。
3. 配置Python Interpreter
   - Interpreter：填写第一步复制的python路径，例如：/home/ma-user/anaconda3/envs/Pytorch-1.0.0/bin/python
     如果路径为~/anaconda3/envs/Pytorch-1.0.0/bin/python 把~替换为/home/ma-user即可。
   - Sync folders: 需要配置本地的工程目录文件同步到云上开发环境中的某个目录，推荐配置为/home/ma-user下的某个目录中（其他目录可能没有访问权限），例如/home/ma-user/work/projects。
4. 单击右侧文件夹图标，勾选上“Automatically upload”选项，以便于本地修改的文件自动上传到容器环境中。
5. 单击“Finish”，结束配置。

   可以看到本地的工程文件已经自动往云上环境上传了。后续本地的文件每修改一次，都会自动的同步到云上的环境中。

   右下角可以看到当前的Interpreter为Remote Interpreter。

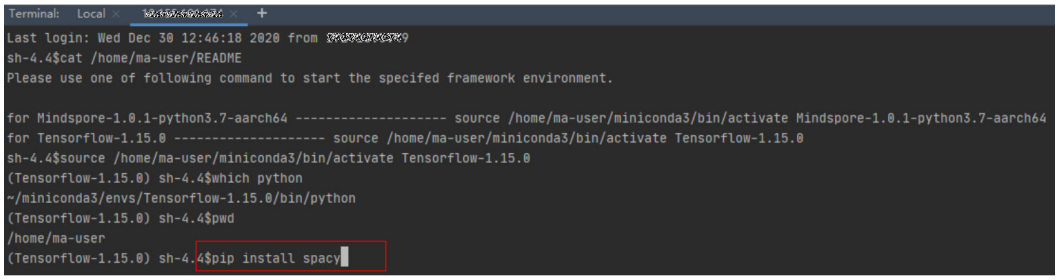
图 5-20 配置云上 Python Interpreter



Step4 云上环境依赖库安装

在进入开发环境后，可以使用不同的虚拟环境，例如TensorFlow、PyTorch等，但是实际开发中，通常还需要安装其他依赖包，此时可以通过Terminal连接到环境里操作。

单击工具栏“Tools >Start SSH session”，选择SSH Configuration中配置的开发环境。可以执行pip install安装所需要的包。

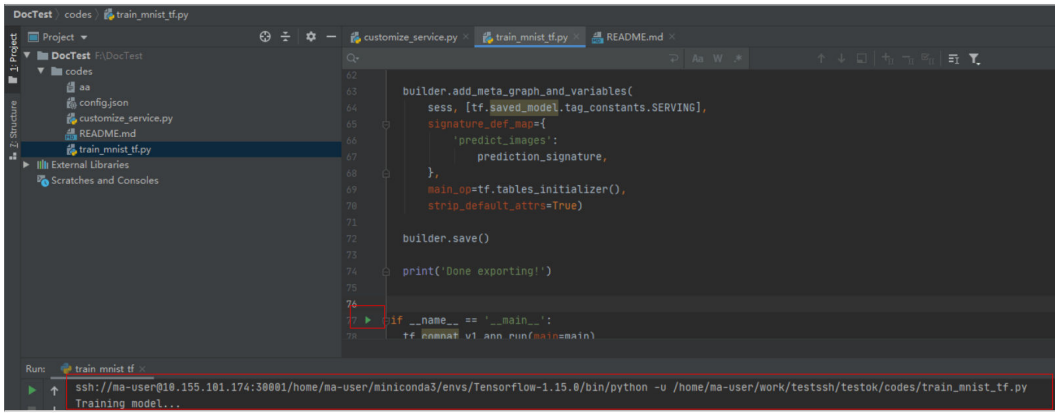


Step5 在开发环境中调试代码

由于已经连接至云端开发环境，此时可以方便地在本地PyCharm中编码、调测并运行。运行实际环境为云上开发环境，资源为云上昇腾AI处理器资源。可以做到本地编写修改代码，直接在云上环境运行。

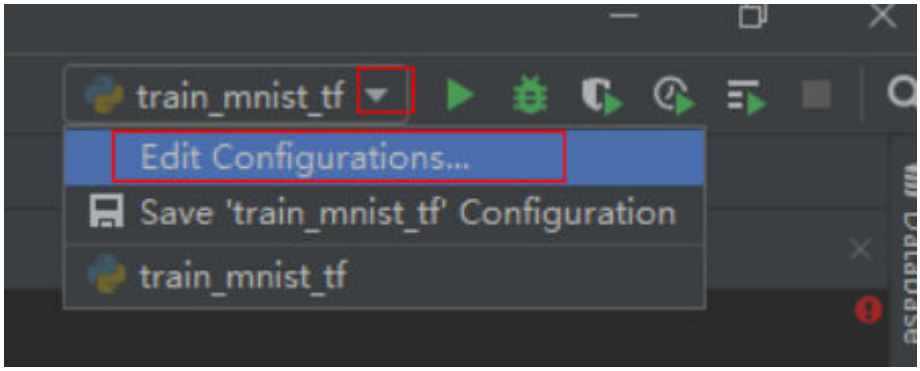
像本地运行代码一样，直接单击运行按钮运行代码即可，此时虽然是在本地IDE单击的运行按钮，实际上运行的是云端开发环境里的代码，日志可以回显在本地的日志窗口。

图 5-21 调试代码



也可以单击右上角的Run/Debug Configuration来设置运行的参数。

图 5-22 设置运行参数



当需要调试代码时，可以直接打断点，然后使用debug方式运行程序。

图 5-23 代码打断点

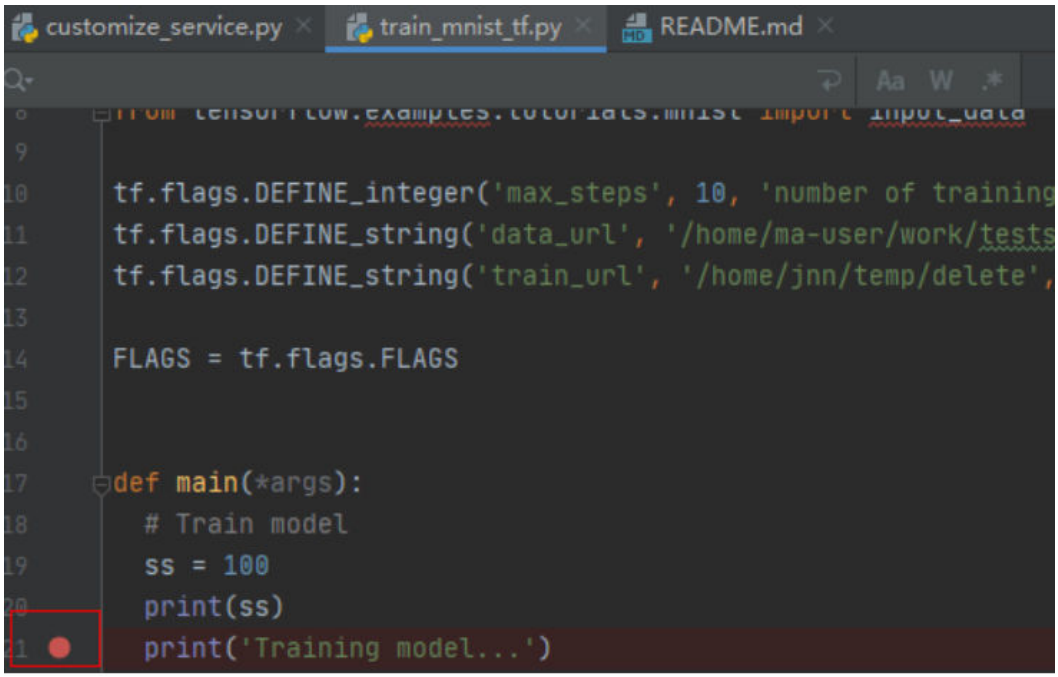
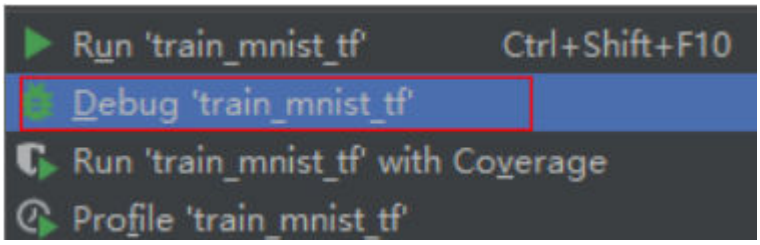
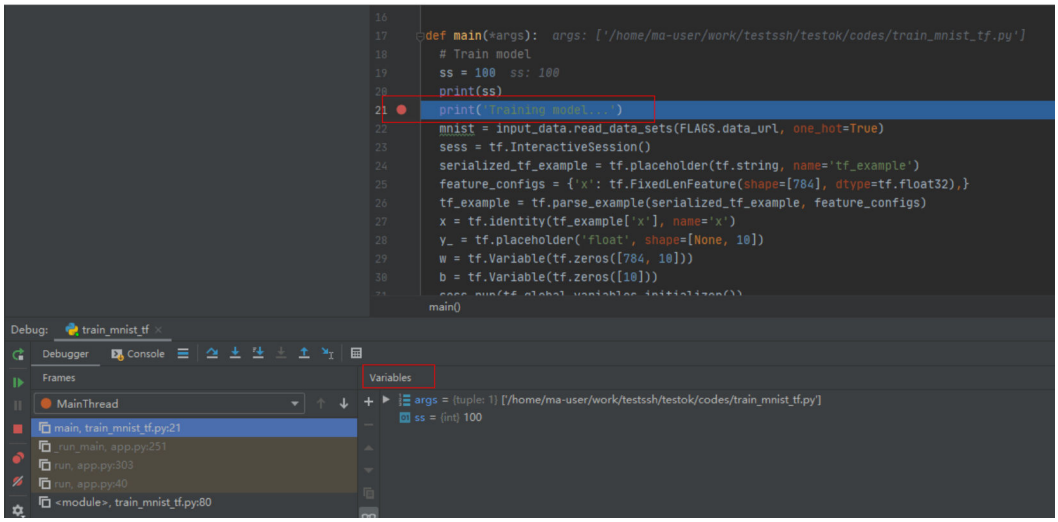


图 5-24 Debug 方式调试



此时可以进入debug模式，代码运行暂停在该行，且可以查看变量的值。

图 5-25 Debug 模式



使用debug方式调试代码的前提是本地的代码和云端的代码是完全一致的，如果不一致可能会导致在本地打断点的行和实际运行时该行的代码并不一样，会出现意想不到的错误。

因此在配置云上Python Interpreter时，推荐选择Automatically upload选项，以保证本地的文件修改能自动上传到云端。如果没有选择自动上传，则本地代码修改完后，也可以参考**Step6 同步上传本地文件至Notebook**手动上传目录或代码。

5.2.3 PyCharm Toolkit 提交训练作业

5.2.3.1 提交训练作业

使用PyCharm ToolKit（latest版本）工具，可以快速将本地开发的训练代码，提交至ModelArts侧进行训练。

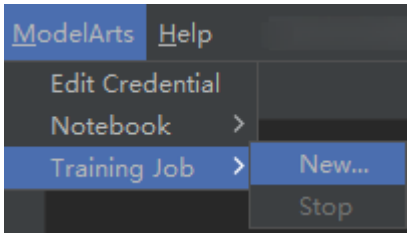
前提条件

- 在本地PyCharm中已有训练代码工程。
- 已在OBS中创建桶和文件夹，用于存放数据集和训练输出模型。 训练作业使用的数据已上传至OBS。
- 已配置credential，详细请参考**使用访问密钥登录**。

配置训练作业参数

1. 在PyCharm中，打开训练代码工程和训练启动文件，然后在菜单栏中选择“ModelArts > Training Job > New...”。

   图 5-26 选择作业配置

2. 在弹出的对话框中，设置训练作业相关参数，详细参数说明请参见**表5-2**。

   表 5-2 训练作业配置参数说明

参数	说明
Job Name	训练作业的名称。 系统会自动生成一个名称，您可以根据业务需求重新命名，命名规则如下： • 支持1~64位字符。 • 并包含大小写字母、数字、中划线（-）或下划线（_）。
Job Description	训练作业的简要描述。

参数	说明
Algorithm Source	训练算法来源，分为“常用框架”和“自定义镜像”两种，二者选一项即可。 常用框架指使用ModelArts训练管理中支持的常用AI引擎。 如果您使用的AI引擎为支持列表之外的，建议使用自定义镜像的方式创建训练作业。
AI Engine	选择代码使用的AI引擎及其版本。
Boot File Path	训练启动文件，所选启动文件必须是当前PyCharm训练工程中的文件。当“Algorithm source”选“Frequently-used”时，显示此参数。
Code Directory	训练代码目录，系统会自动填写为训练启动文件所在的目录，用户可根据需要修改，所选目录必须是当前工程中的目录且包含启动文件。 当算法来源为自定义镜像，训练代码已预置在镜像中时，该参数可以为空。
Image Path(optional)	SWR镜像的URL地址。
Boot Command	启动本次训练作业的运行命令。例如“bash /home/work/run_train.sh python {python启动文件及参数}”。当“Algorithm source”选“Custom”时，显示此参数。 当用户输入的命令中不包含“--data_url”和“--train_url”参数时，工具在提交训练作业时会在命令后面自动添加这两个参数，分别对应存储训练数据的OBS路径和存放训练输出的OBS路径。
Data OBS Path	设置为存储训练数据的OBS路径，例如“/test-modelarts2/mnist/dataset-mnist/”，其中“test-modelarts2”为桶名称。
Training OBS Path	设置OBS路径，该路径下会自动创建用于存放训练输出模型和训练日志的目录。
Running Parameters	运行参数。如果您的代码需要添加一些运行参数，可以在此处添加，多个运行参数使用英文分号隔开， 例如"key1=value1;key2=value2"。此参数也可以不设置，即保持为空。
Specifications	训练使用资源类型。目前支持公共资源池和专属资源池两种类型。 专属资源池规格以“Dedicated Resource Pool”标识。只有购买了专属资源池的用户才会显示专属资源池规格。
Compute Nodes	计算资源节点个数。数量设置为1时，表示单机运行；数量设置大于1时，表示后台的计算模式为分布式。
Available/Total Nodes	当“Specifications”选择专属资源池规格时，显示专属资源池的可用节点数和总节点数，用户选择“Compute Nodes”的个数不要超过可用节点数。

图 5-27 配置训练作业参数（公共资源池）

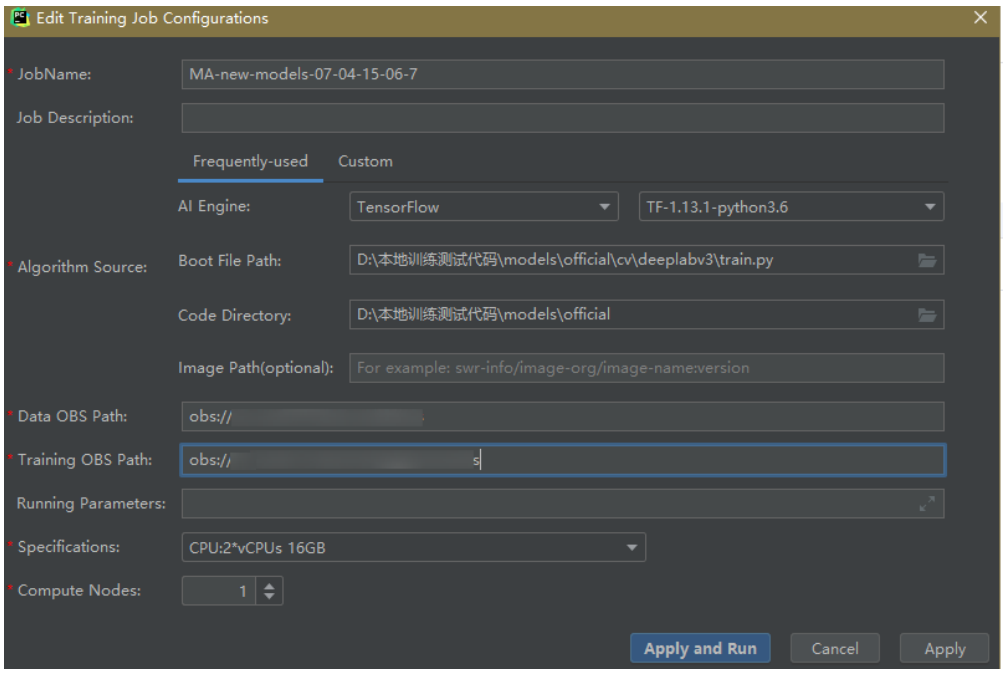


图 5-28 配置训练作业参数（专属资源池）

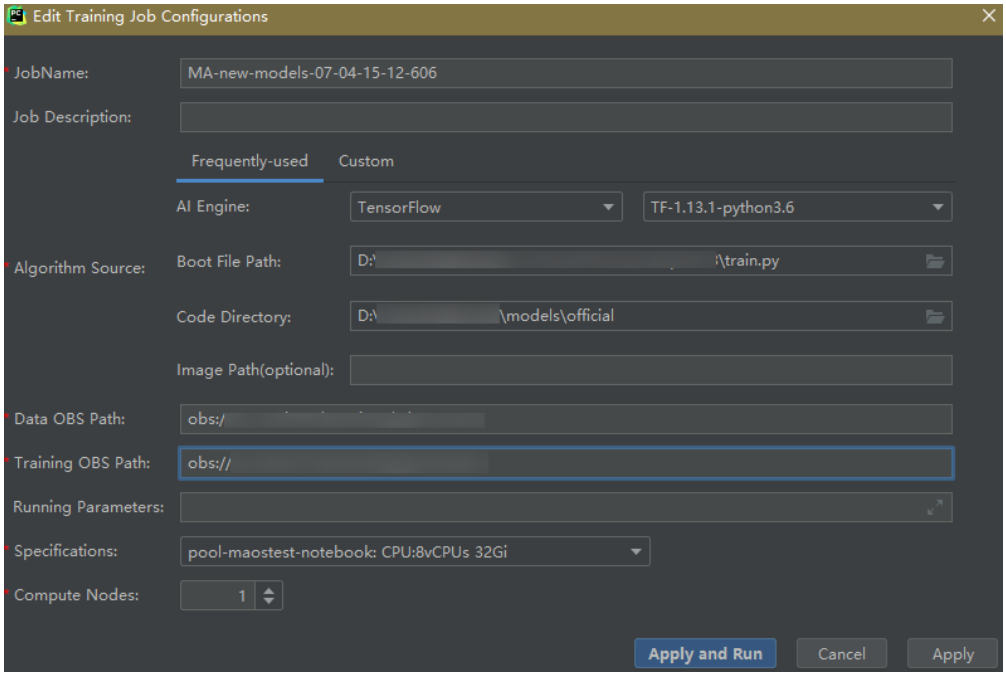
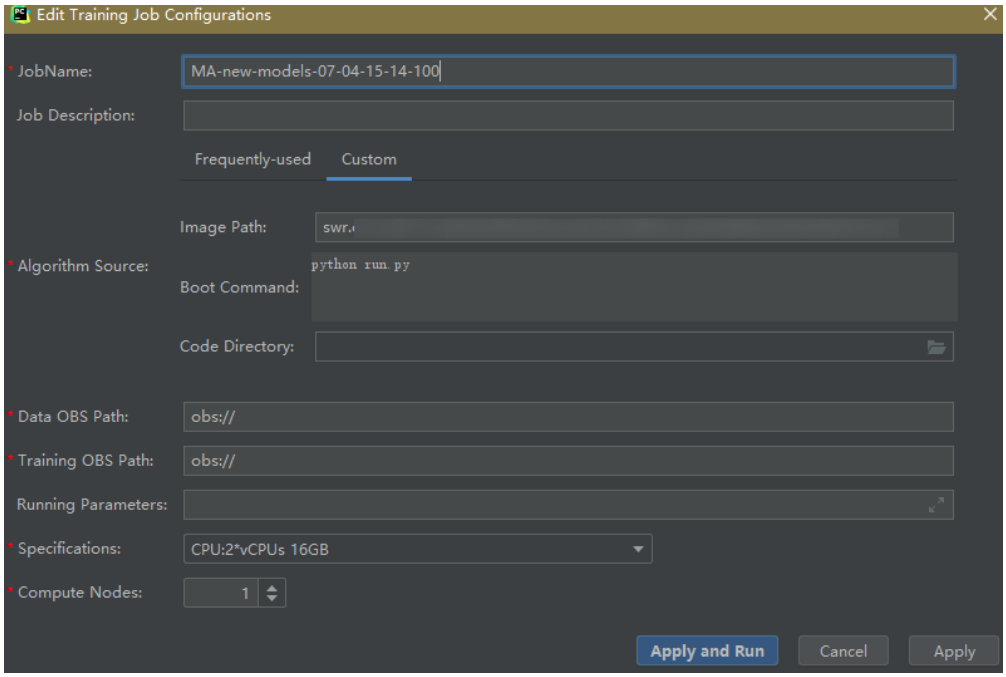


图 5-29 配置训练作业参数（自定义镜像）

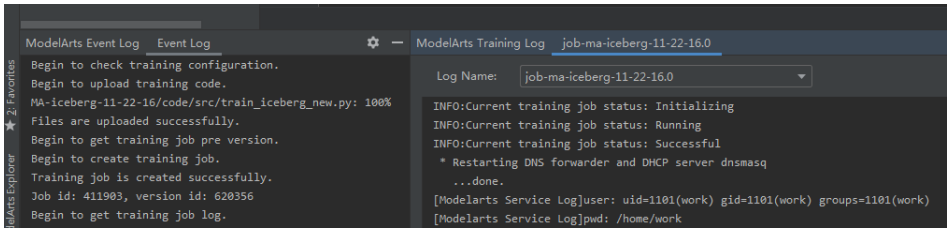


3. 参数填写完成后，单击“Apply and Run”，即自动上传本地代码至云端并启动训练，在工具下方的Training Log区域，会实时展示训练作业运行情况。当训练日志中出现“Current training job status: Successful”类似信息时，表示训练作业运行成功。

说明

- 在单击“Apply and Run”按钮后，系统将自动开始执行训练作业。如果您想停止此作业，可以选择菜单栏中的“ModelArts > Training Job >Stop”停止此作业。
- 如果单击“Apply”，不会直接启动运行，只是保存训练作业的设置，如果需要启动作业，可以单击“Apply and Run”。

图 5-30 训练日志展示样例



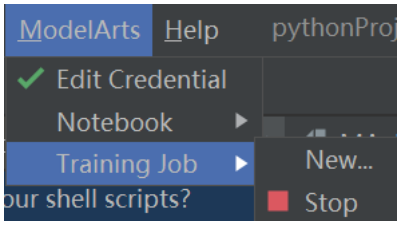
5.2.3.2 停止训练作业

当训练作业在运行过程中时，您可以执行停止作业的操作。

停止作业

1. 选择正在运行的训练作业。
2. 在PyCharm菜单栏中，选择“ModelArts >Training Job > Stop ”停止此作业。

图 5-31 停止作业



5.2.3.3 查看训练日志

本章节介绍如何查看训练作业产生的日志。

在 OBS 中查看

提交训练作业时，系统将自动在您配置的OBS Path中，使用作业名称创建一个新的文件夹，用于存储训练输出的模型、日志和代码。

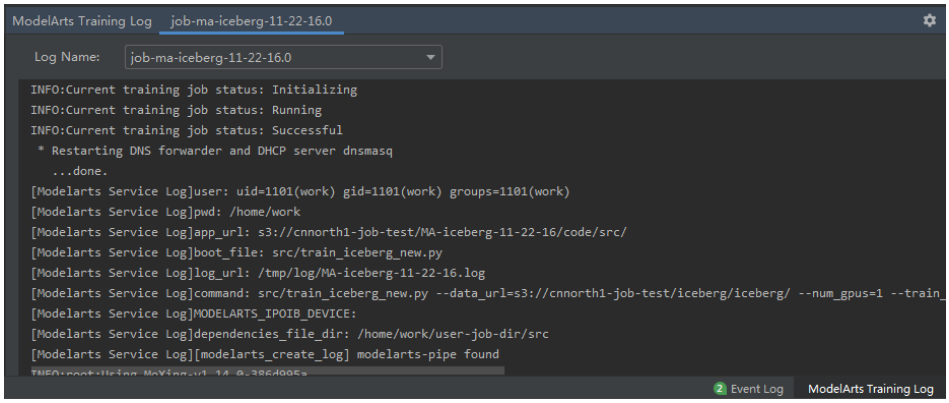
例如“train-job-01”作业，提交作业时会在“test-modelarts2”桶下创建一个命名为“train-job-01”的文件夹，且此文件夹下分别新建了三个文件夹“output”、“log”、“code”，分别用于存储输出模型、日志和训练代码。“output”文件夹还会根据您的训练作业版本再创建子文件夹，结构示例如下。

```
test-modelarts2
  |---train-job-01
      |---output
      |---log
      |---code
```

在 ToolKit 工具中查看

在PyCharm工具中，单击页面右下角的ModelArts Training Log，展示训练日志。

图 5-32 查看训练日志



5.2.4 在 PyCharm 中上传数据至 Notebook

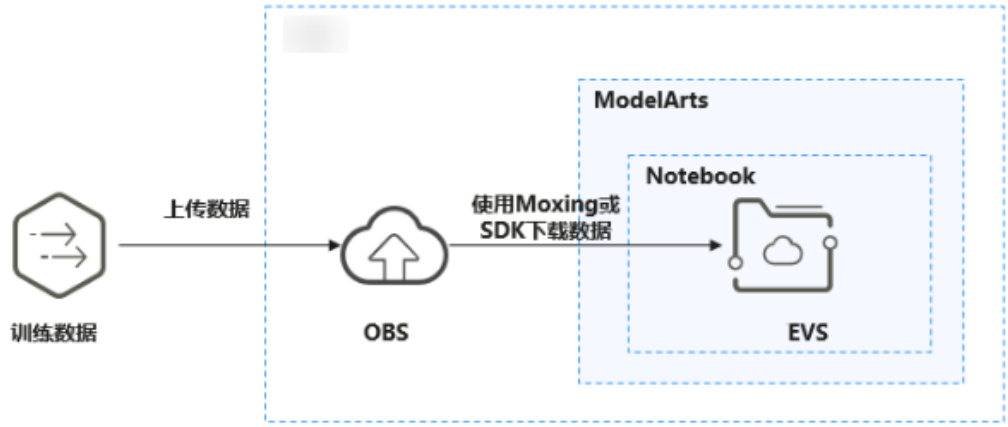
不大于500MB数据量，直接复制至本地IDE中即可。

大于500MB数据量，请先上传到OBS中，再从OBS上传到云上开发环境。

1. 上传数据至OBS，具体操作请参见**上传文件至OBS桶**。

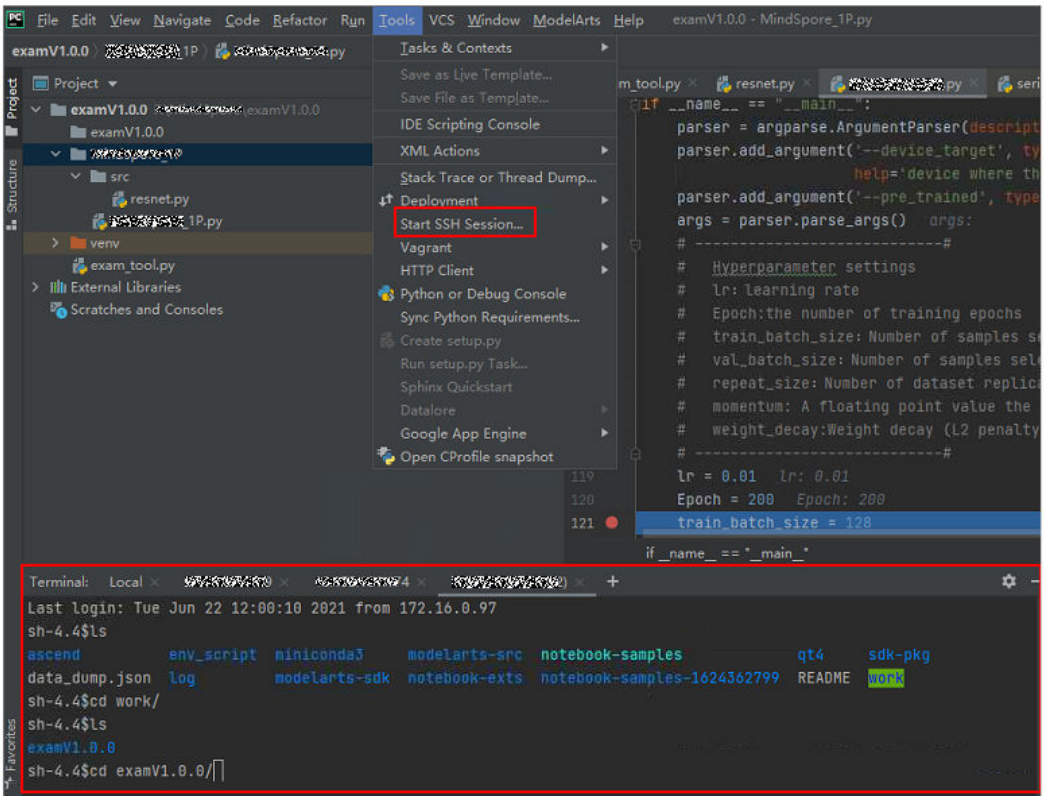
2. 将OBS中的数据传至Notebook中，通过在本地IDE的Terminal中使用ModelArts提供的Moxing库的文件操作API（mox.file.copy_parallel）完成。

图 5-33 数据通过 OBS 中转上传到 Notebook



下图以PyCharm环境中开启Terminal为例，VS Code中操作类似。

图 5-34 PyCharm 环境开启 Terminal



在本地IDE的Terminal中使用Moxing下载OBS文件到开发环境的操作示例如下：

```
#手动source进入开发环境
cat /home/ma-user/README
#然后选择要source的环境
source /home/ma-user/miniconda3/bin/activate MindSpore-python3.7-aarch64
#输入python并回车，进入python环境
```

```
python
#使用moxing
import moxing as mox
#下载一个OBS文件夹，从OBS下载至EVS（OBS -> EVS）
mox.file.copy_parallel('obs://bucket_name/sub_dir_0', '/tmp/sub_dir_0')
```

5.3 本地 IDE（VS Code）

5.3.1 VS Code 连接 Notebook 方式介绍

当用户创建完成支持SSH的Notebook实例后，使用VS Code的开发者可以通过以下两种方式连接到开发环境中：

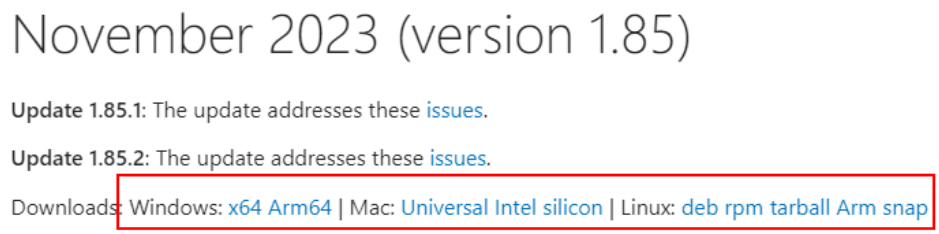
- **VS Code ToolKit连接Notebook**（推荐）
  该方式是指用户在VS Code上使用ModelArts VS Code Toolkit插件提供的登录和连接按钮，连接云上实例。
- **VS Code手动连接Notebook**
  该方式是指用户使用VS Code Remote SSH插件手工配置连接信息，连接云上实例。

5.3.2 安装 VS Code 软件

VS Code下载方式：

- 下载地址: **https://code.visualstudio.com/updates/v1_85**

  图 5-35 VS Code 的下载位置



VS Code版本要求：

建议用户使用VS Code 1.85.2版本或者最新版本进行远程连接。

VS Code安装指导如下：

Windows系统下，下载后直接双击安装包完成安装。

Linux系统下，执行命令**sudo dpkg -i code_1.85.2-1705561292_amd64.deb**安装。

说明

Linux系统用户，需要在非root用户进行VS Code安装。

5.3.3 VS Code ToolKit 连接 Notebook

本节介绍如何在本地使用ModelArts提供的VS Code插件工具VS Code ToolKit，协助用户完成SSH远程连接Notebook。

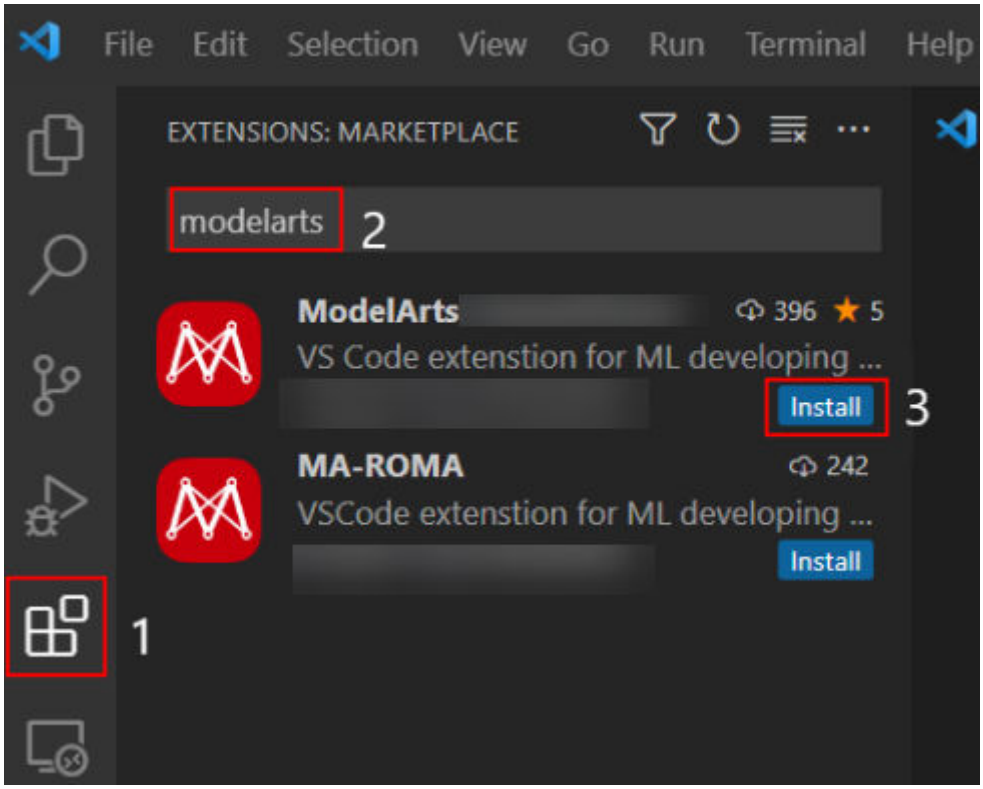
前提条件

已下载并安装VS Code。详细操作请参考**安装VS Code软件**。

Step1 安装 VS Code 插件

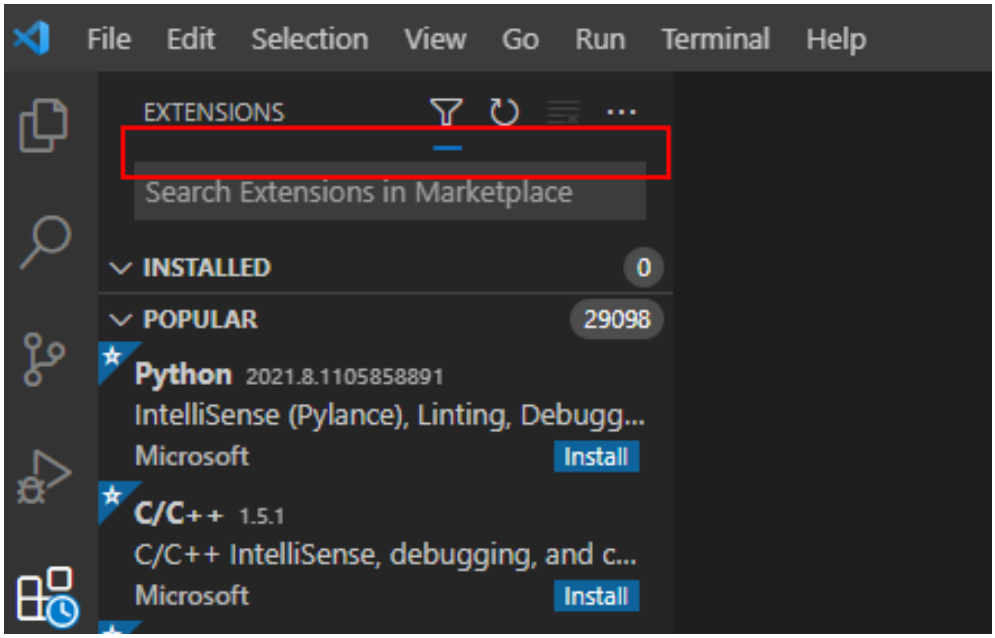
1. 在本地的VS Code开发环境中，如**图5-36**所示，在VS Code扩展中搜索“ModelArts-HuaweiCloud”并单击“安装”。

图 5-36 安装 VS Code 插件



2. 安装过程预计1~2分钟，如**图5-37**所示，请耐心等待。

图 5-37 安装过程



3. 安装完成后，系统右下角提示安装完成，导航左侧出现ModelArts图标和SSH远程连接图标，表示VS Code插件安装完成。

图 5-38 安装完成提示

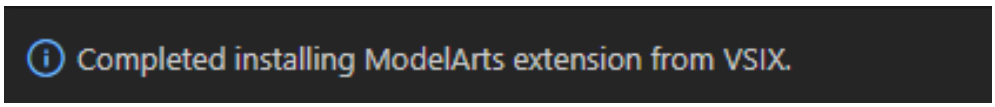
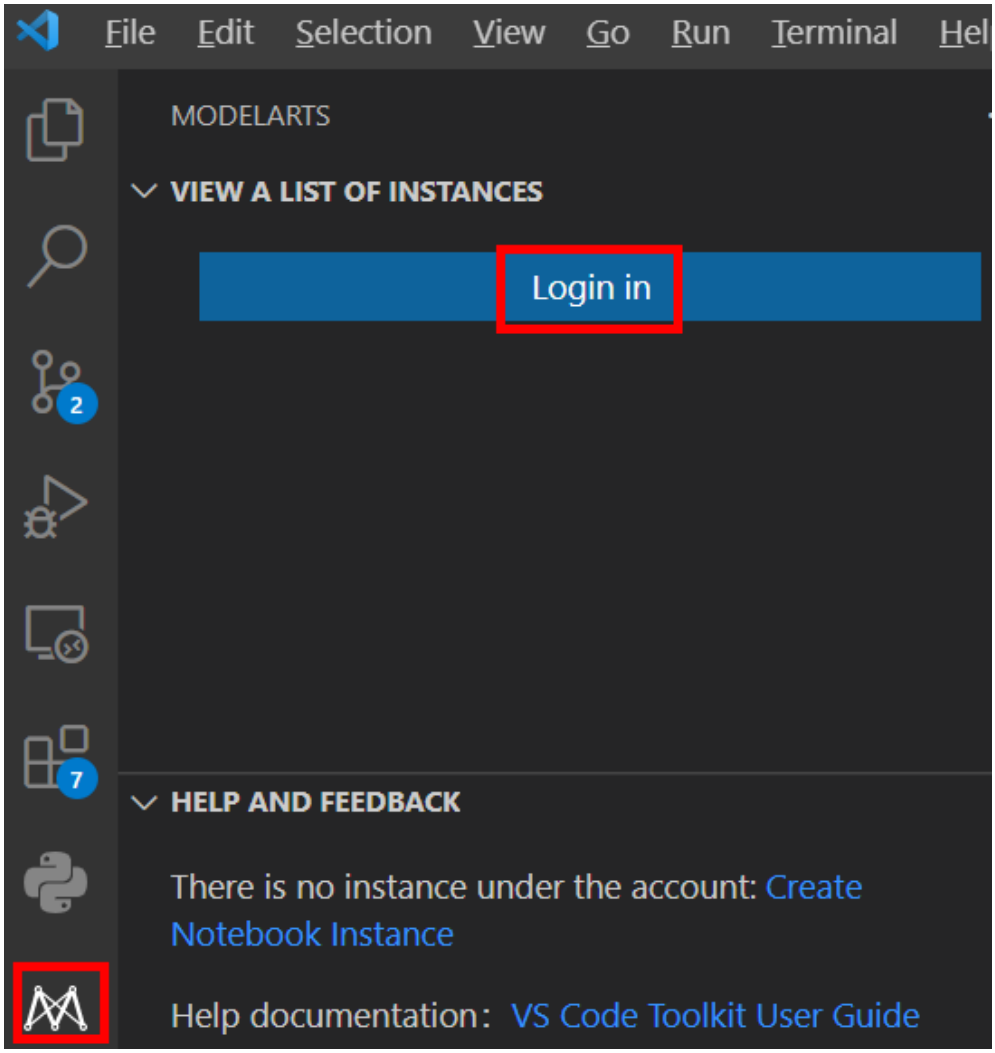
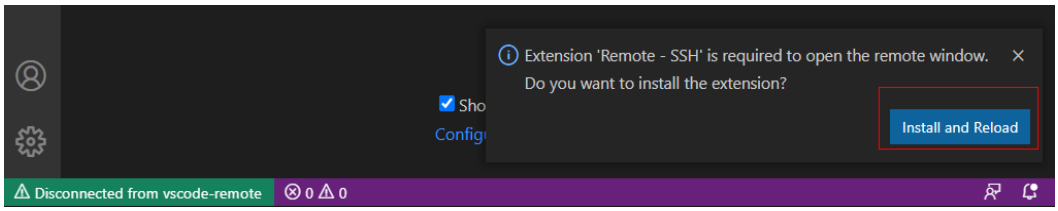


图 5-39 安装完成



当前网络不佳时SSH远程连接插件可能未安装成功，此时无需操作，在**Step4 连接Notebook实例**的**1**之后，会弹出如下图对话框，单击Install and Reload即可。

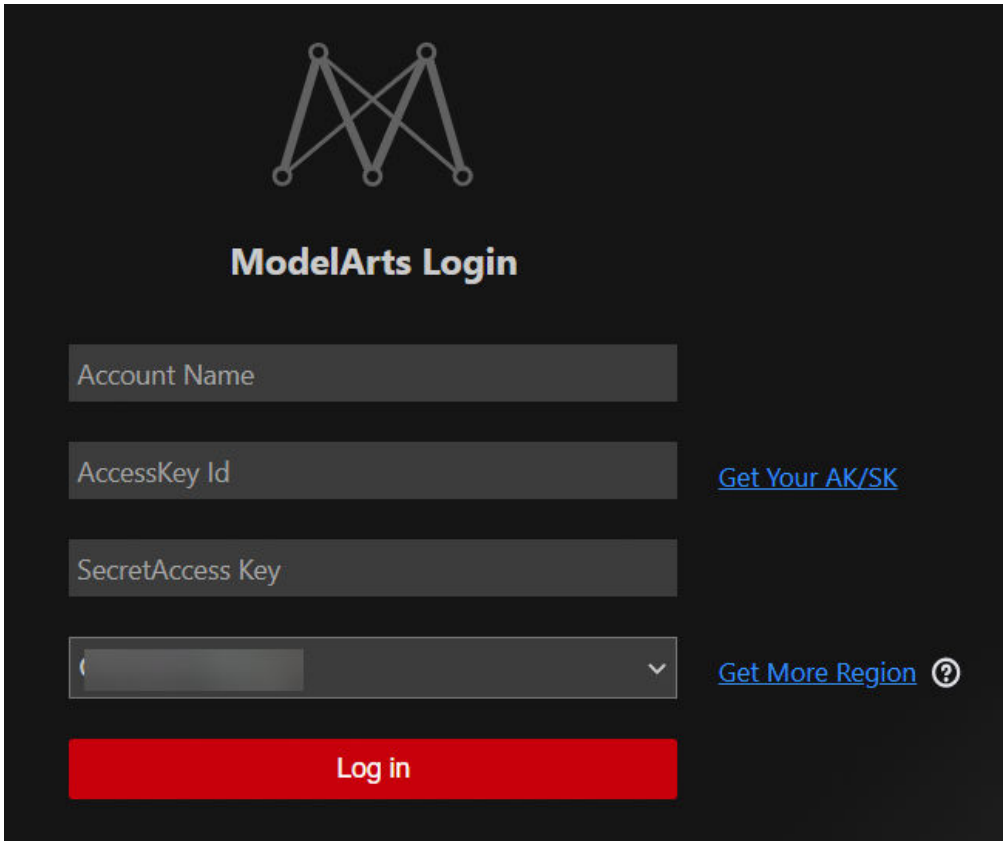
图 5-40 重新连接远程 SSH



Step2 登录 VS Code 插件

1. 在本地的VS Code开发环境中，单击ModelArts图标  ，单击“User Settings”，配置用户登录信息。

图 5-41 登录插件



输入如下用户登录信息，单击“登录”。

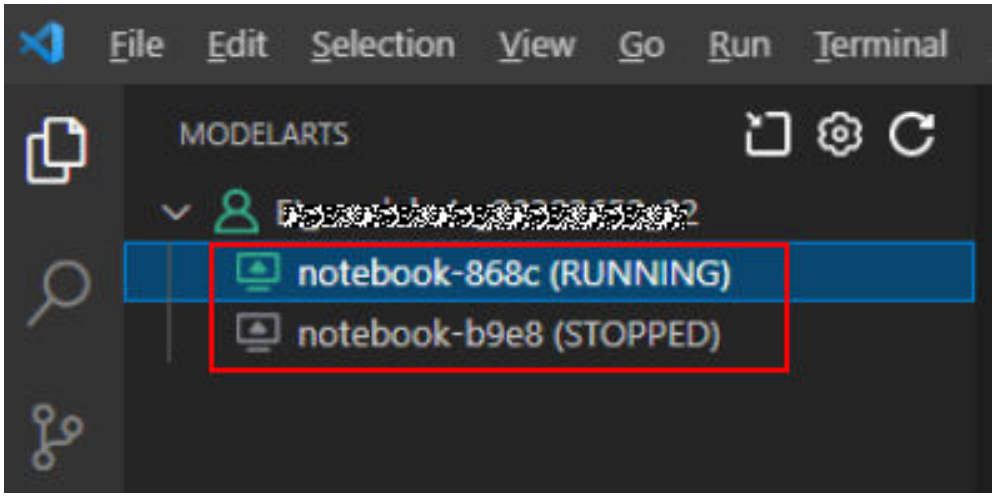
- Name：自定义用户名，仅用于VS Code页面展示，不与任何华为云用户关联。
- AK、SK：在“账号中心 > 我的凭证 > 访问密钥”中创建访问密钥，获取AK、SK（**参考链接**）。
- 选择站点：此处的站点必须和远程连接的Notebook在同一个站点，否则会导致连接失败。

2. 登录成功后显示Notebook实例列表。

说明

此处仅显示ModelArts控制台default工作空间下的Notebook实例。

图 5-42 登录成功



Step3 创建 Notebook 实例

注意

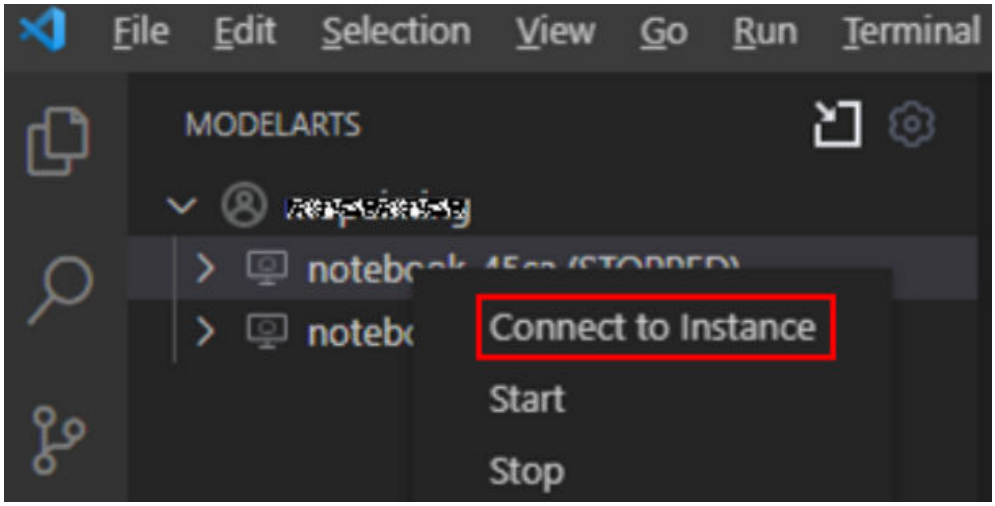
- 创建实例时，需开启“SSH远程开发”，并下载保存密钥对至本地如下目录。
  Windows：C:\Users\{{user}}
  macOS/Linux: Users/{{user}}
- 密钥对在用户第一次创建时自动下载，之后使用相同的密钥时不会再有下载界面（请妥善保管），或者每次都使用新的密钥对。

创建一个Notebook实例，并开启远程SSH开发，具体参见**创建Notebook实例**。

Step4 连接 Notebook 实例

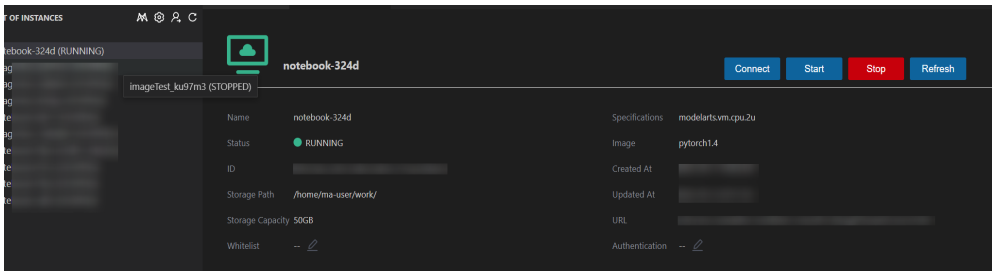
1. 在本地的VS Code开发环境中，右键单击实例名称，单击“Connect to Instance”，启动并连接Notebook实例。
   Notebook实例状态处于“运行中”或“停止”状态都可以，如果Notebook实例是停止状态，连接Notebook时，VS Code插件会先启动实例再去连接。

图 5-43 连接 Notebook 实例



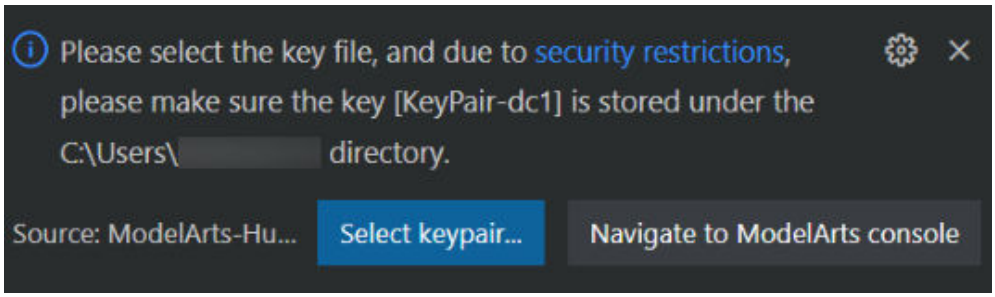
或者单击实例名称，在VS Code开发环境中显示Notebook实例详情页，单击“连接”，系统自动启动该Notebook实例并进行远程连接。

图 5-44 查看 Notebook 实例详情页



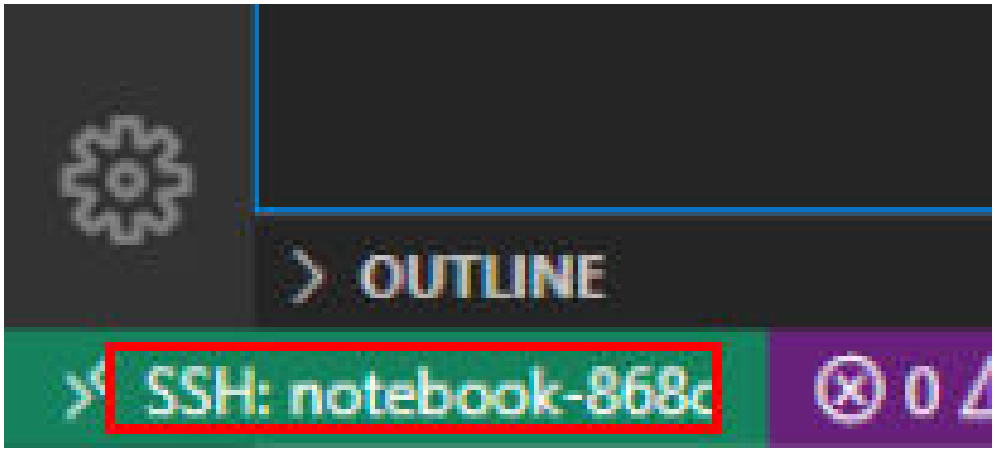
2. 第一次连接Notebook时，系统右下角会提示需要先配置密钥文件。选择本地密钥pem文件，根据系统提示单击“OK”。

图 5-45 配置密钥文件



3. 单击“确定”后，插件自动连接远端Notebook实例。首次连接大约耗时1~2分钟，取决于本地的网络情况。VS Code环境左下角显示类似下图即为连接成功。

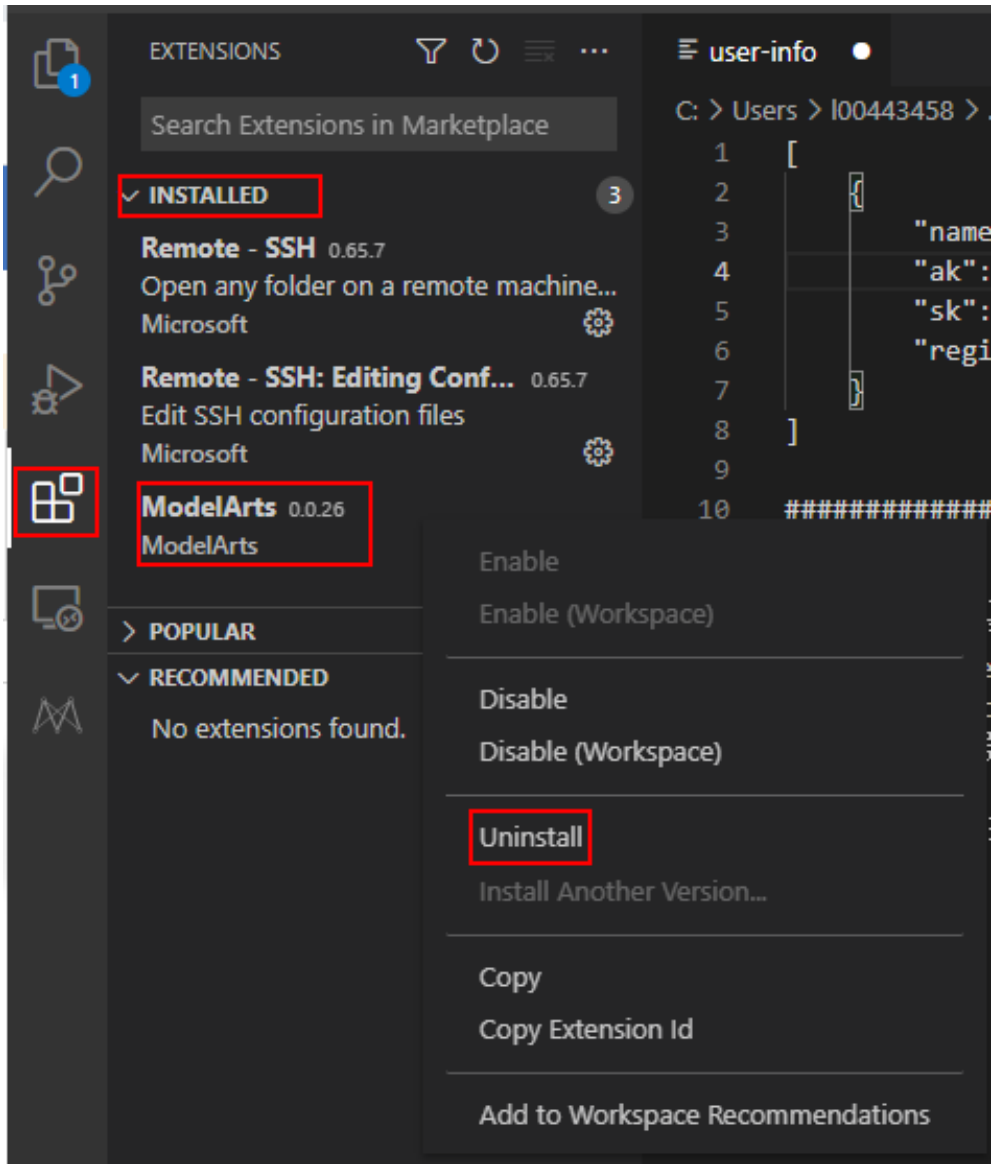
图 5-46 连接成功



相关操作

卸载VS Code插件操作如图5-47所示。

图 5-47 卸载 VS Code 插件



5.3.4 VS Code 手动连接 Notebook

本地IDE环境支持PyCharm和VS Code。通过简单配置，即可用本地IDE远程连接到ModelArts的Notebook开发环境中，调试和运行代码。

本章节介绍基于VS Code环境访问Notebook的方式。

前提条件

- 已下载并安装VS Code。详细操作请参考**安装VS Code软件**。
- 用户本地PC或服务器的操作系统中建议先安装Python环境，详见**VSCode官方指导**。
- 创建一个Notebook实例，并开启远程SSH开发。该实例状态必须处于“运行中”，具体参见**创建Notebook实例**章节。
- 在Notebook实例详情页面获取开发环境访问地址和端口号。

图 5-48 Notebook 实例详情页面

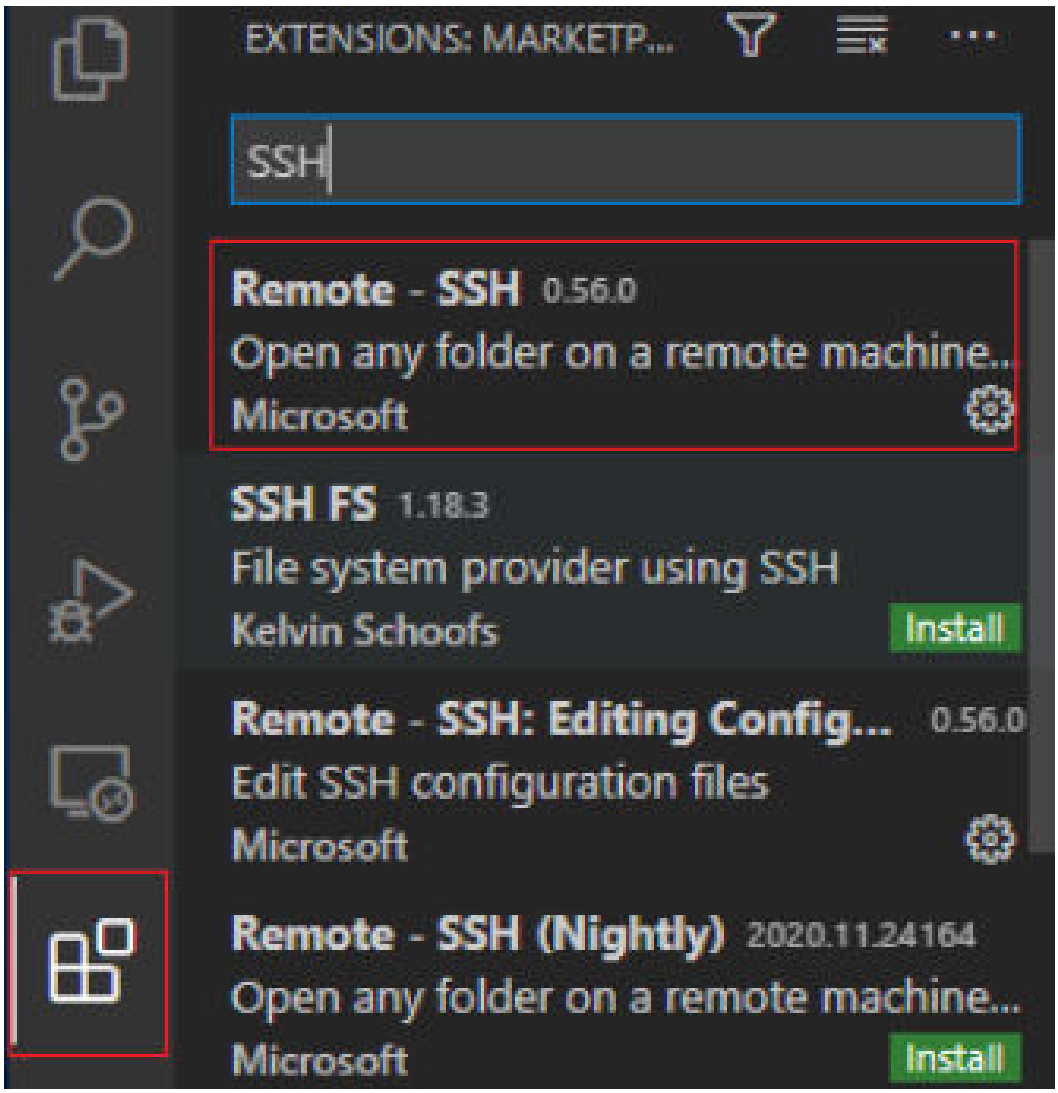
- 准备好密钥对。

  密钥对在用户第一次创建时，自动下载，之后使用相同的密钥时不会再有下载界面（用户一定要保存好），或者每次都使用新的密钥对。

Step1 添加 Remote-SSH 插件

在本地的VS Code开发环境中，单击左侧列表的Extensions图标选项  ，在搜索框中输入SSH，单击Remote-SSH插件的install按钮，完成插件安装。

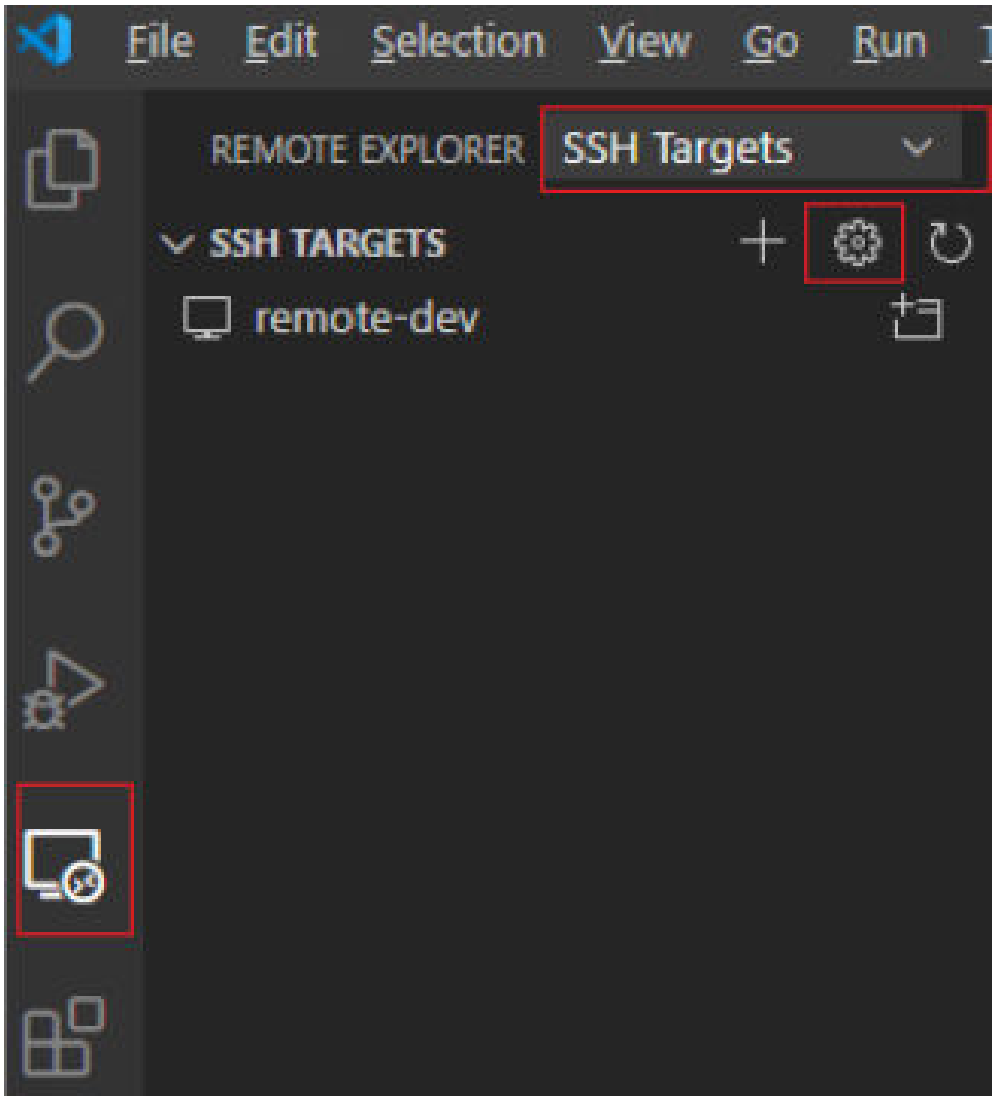
图 5-49 添加 Remote-SSH 插件



Step2 配置 SSH

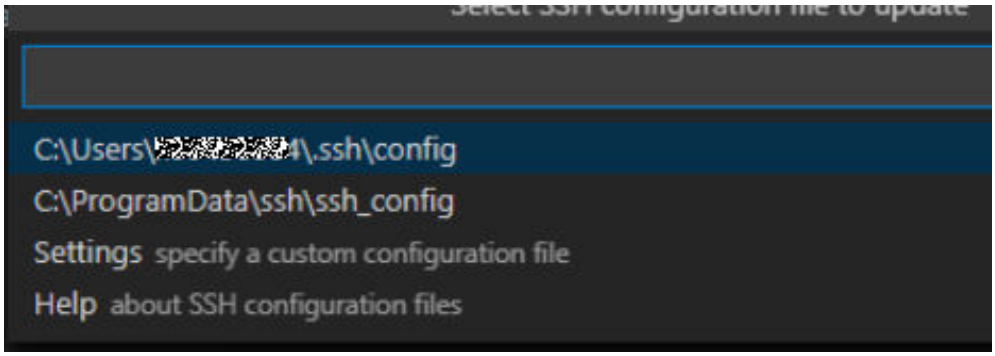
1. 在本地的VS Code开发环境中，单击左侧Remote Explorer按钮，在上方的下拉列表中选择“SSH Target”，再单击页面上的设置按钮，此时会出现SSH配置文件路径。

   图 5-50 配置 SSH Targets 页面

2. 单击列表中出现的SSH路径按钮，打开config文件，进行配置。

图 5-51 配置 SSH Config 文件

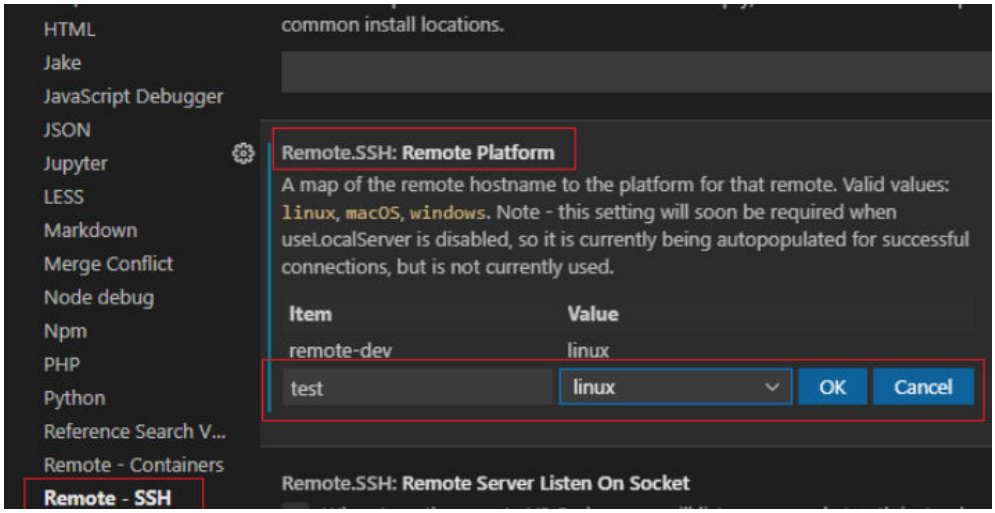



```
HOST remote-dev
    hostname <instance connection host>
    port <instance connection port>
    user ma-user
    IdentityFile ~/.ssh/test.pem
    UserKnownHostsFile=/dev/null
    StrictHostKeyChecking no
```

- Host: 自定义设置的云上开发环境名称。
- HostName: 云上开发环境的访问地址，即在开发环境实例页面远程访问模块获取的访问地址。
- Port: 云上开发环境的端口，即在开发环境实例页面远程访问模块获取的端口号。
- User：登录用户只支持ma-user进行登录。
- IdentityFile: 存放在本地的云上开发环境私钥文件，即前提条件**准备好密钥对**中准备的密钥对。

3. 配置云上开发环境系统平台，单击“File > Preference > Settings > Extensions > Remote-SSH”，在“Remote Platform”中，单击“Add Item”选项，设置“Item”和“Value”，配置完成后，单击“OK”。

图 5-52 配置云上开发环境系统平台

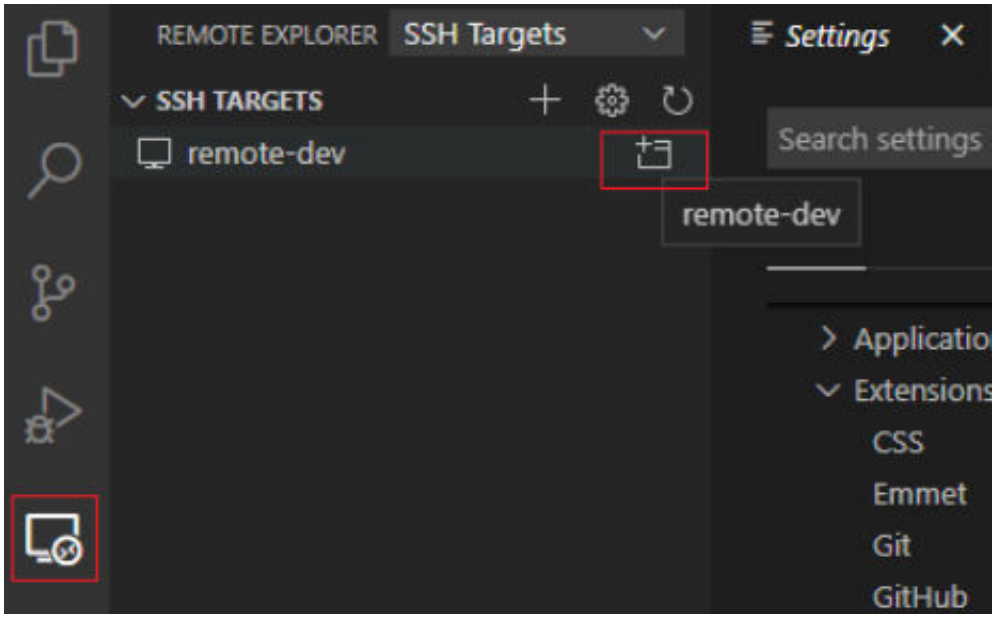


“Item”： 在SSH Config中配置的Host的名称。

“Value”： 在下拉选择框中选择远端开发环境平台。

4. 再回到SSH Targets页面，单击右侧的Connect to Host in New Window按钮 ，该按钮会显示远程开发环境名称，选中并打开。

图 5-53 打开开发环境



在新打开的页面中，看到下图所示界面，即表示连接成功。

图 5-54 开发环境远程连接成功

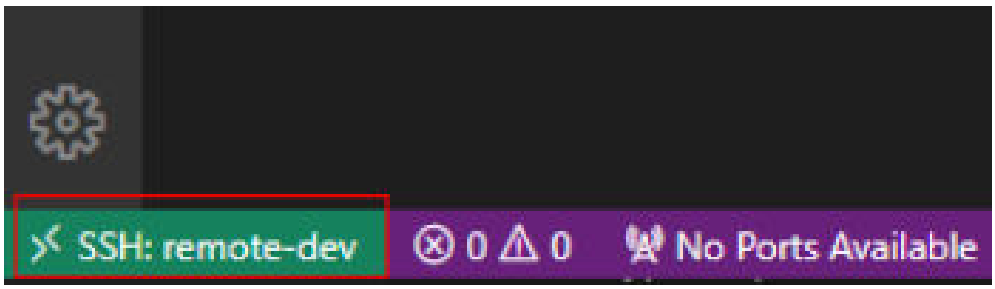
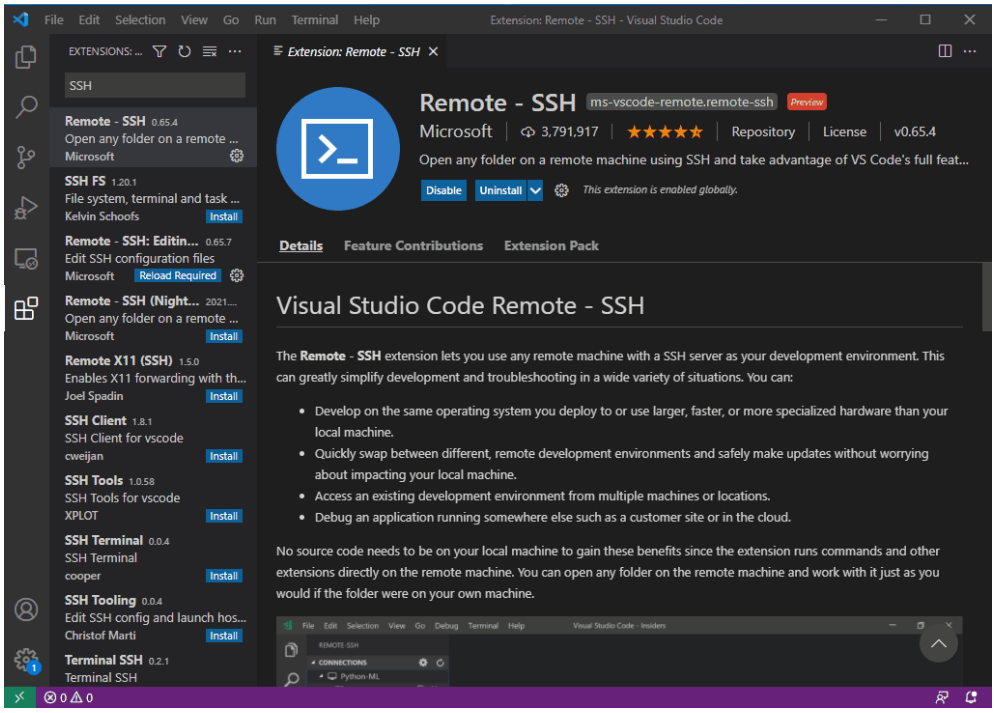


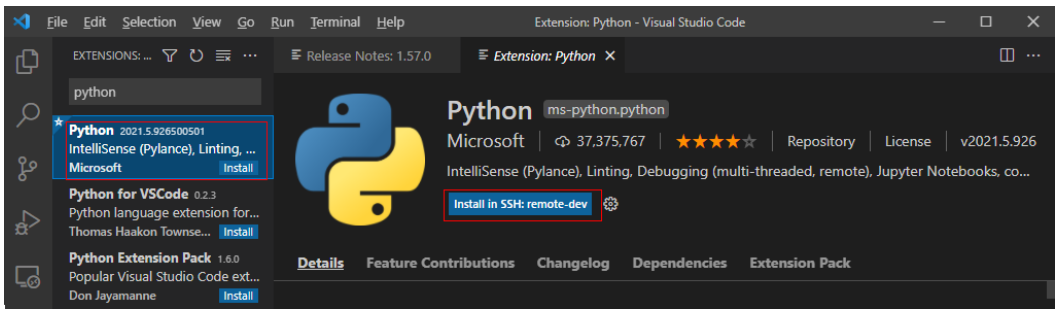
图 5-55 完整配置示例



Step3 安装云端 Python 插件

在新打开的VS Code界面，单击左侧列表的Extensions选项  ，在搜索框中输入Python，在下拉列表中单击“Install”进行安装。

图 5-56 安装云端 Python 插件



如果安装云端的Python插件不成功时，建议通过离线包的方式安装。

Step4 云上环境依赖库安装

在进入容器环境后，可以使用不同的虚拟环境，例如TensorFlow、PyTorch等，但是实际开发中，通常还需要安装其他依赖包，此时可以通过Terminal连接到环境里操作。

1. 在VS Code环境中，执行Ctrl+Shift+P。
2. 搜Python: Select Interpreter，选择对应的Python环境。
3. 单击页面上方的“Terminal > New Terminal”，此时打开的命令行界面即为远端容器环境命令行。

4. 进入引擎后，通过执行如下命令安装依赖包。
```
pip install spacy
```

5.3.5 在 VS Code 中远程调试代码

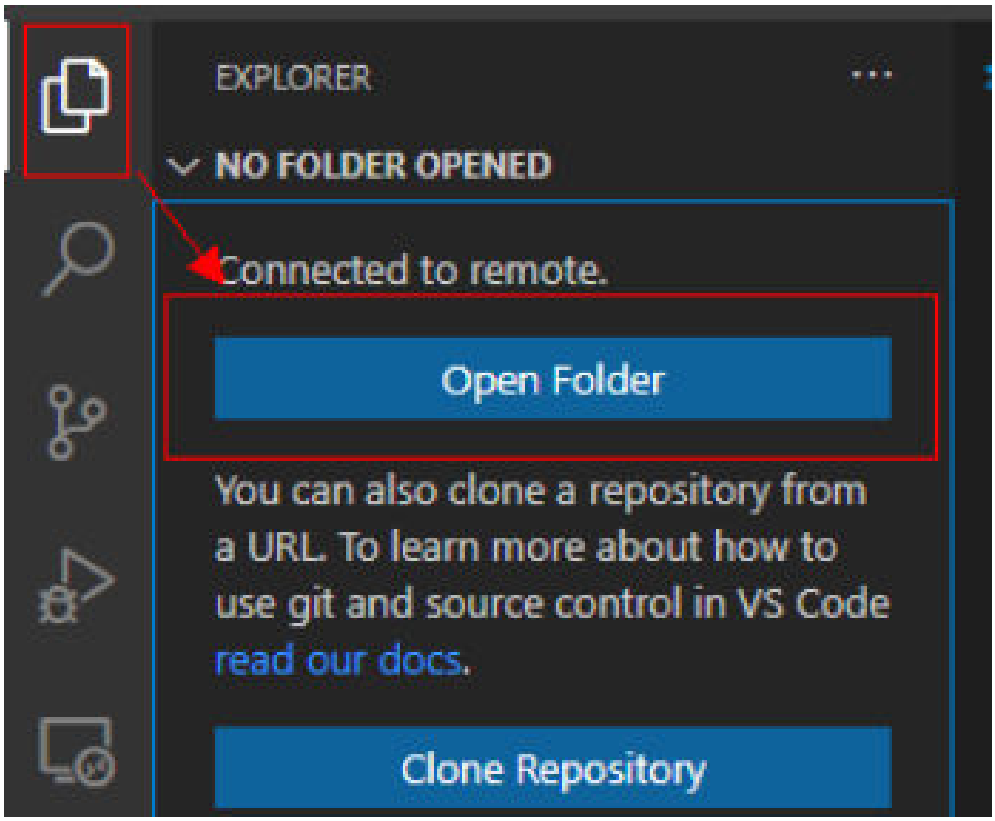
前提条件

VS Code已连接到Notebook。

Step1 上传本地代码到云端开发环境

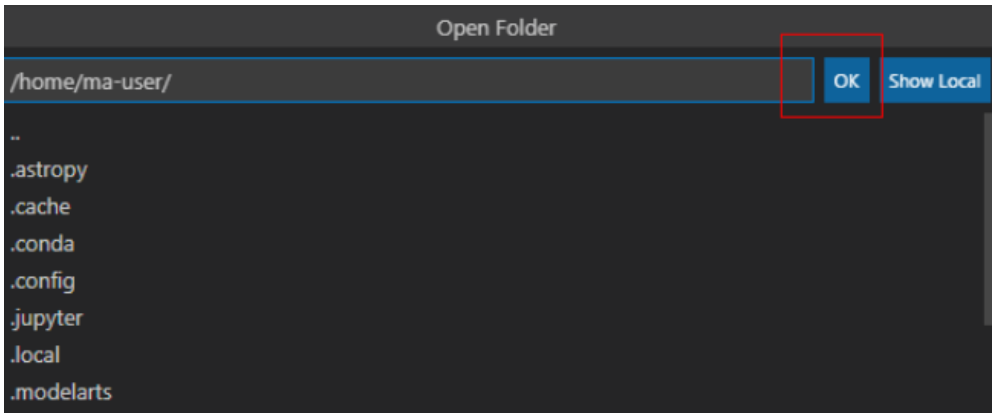
1. 在VS Code界面，单击“File > OpenFolder”打开云端路径。

图 5-57 Open Folder



2. 选择要打开的路径，单击“OK”。

图 5-58 选择文件路径

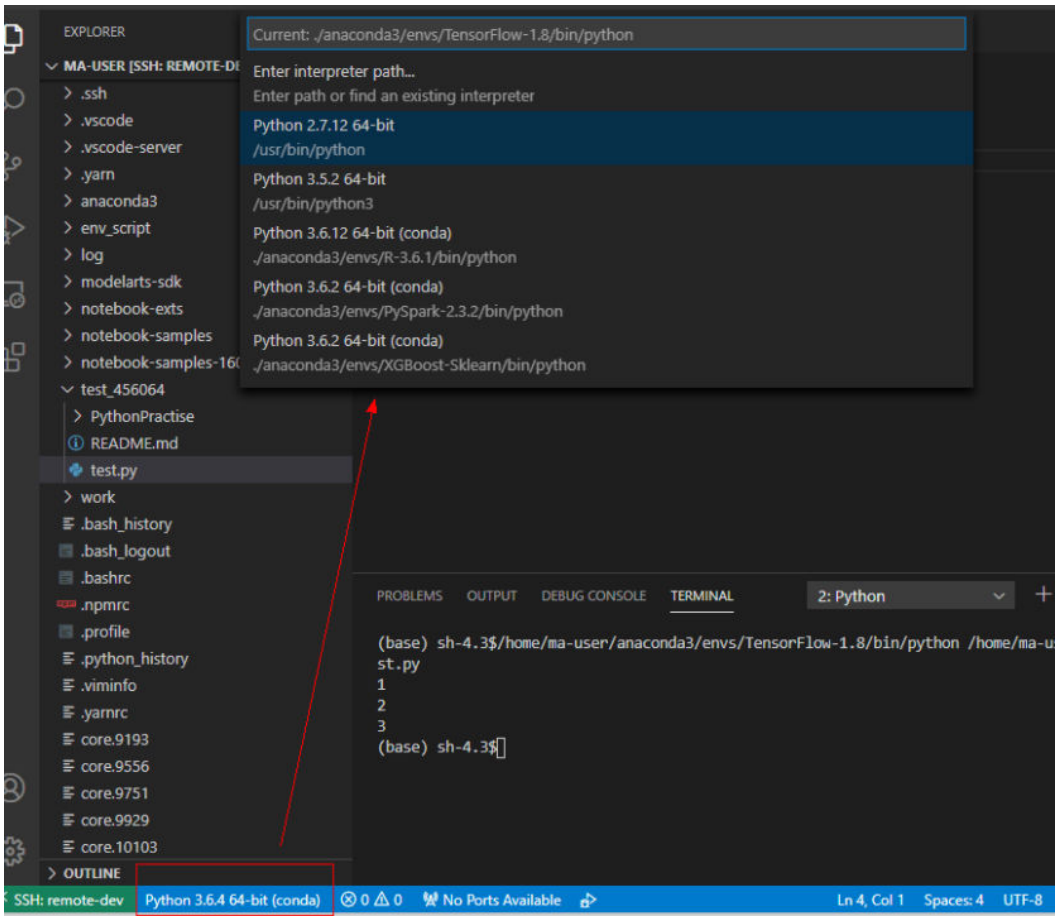


3. 此时，会在IDE左侧出现该开发环境下的目录结构，把想要上传的代码及其他文件直接拖拽至对应的文件夹内即完成本地代码上传至云端。

Step2 远程调试代码

在VS Code中打开要执行的代码文件，在执行代码之前需要选择合适的Python版本路径，单击下方默认的Python版本路径，此时在上方会出现该远程环境上所有的python版本，选择自己需要的版本即可。

图 5-59 选择 Python 版本



- 对于打开的代码文件，单击run按钮，即可执行，可以在下方的Terminal中看到代码输出信息。
- 如果执行较长时间的训练任务，建议使用nohup命令后台运行，否则SSH窗口关闭或者网络断连会影响正在运行的训练任务，命令参考：
  ```
  nohup your_train_job.sh > output.log 2>&1  &  tail -f  output.log
  ```
- 如果要对代码进行debug调试，步骤如下：
  a. 单击左侧“Run > Run and Debug”。
  b. 选择当前打开的默认的python代码文件进行调试。
  c. 对当前代码进行打断点，即在代码左侧进行单击，就会出现小红点。
  d. 此时，即可按照正常的代码调试步骤对代码调试，在界面左边会显示debug信息，代码上方有相应的调试步骤。

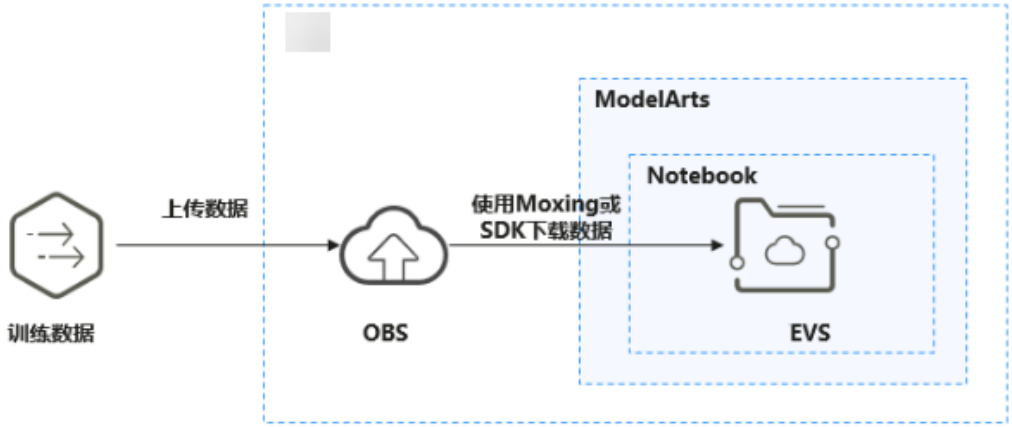
5.3.6 在 VS Code 中上传下载文件

在本地 IDE 中上传数据至 Notebook

不大于500MB数据量，直接复制至本地IDE中即可。

大于500MB数据量，请先上传到OBS中，再从OBS上传到云上开发环境。

图 5-60 数据通过 OBS 中转上传到 Notebook

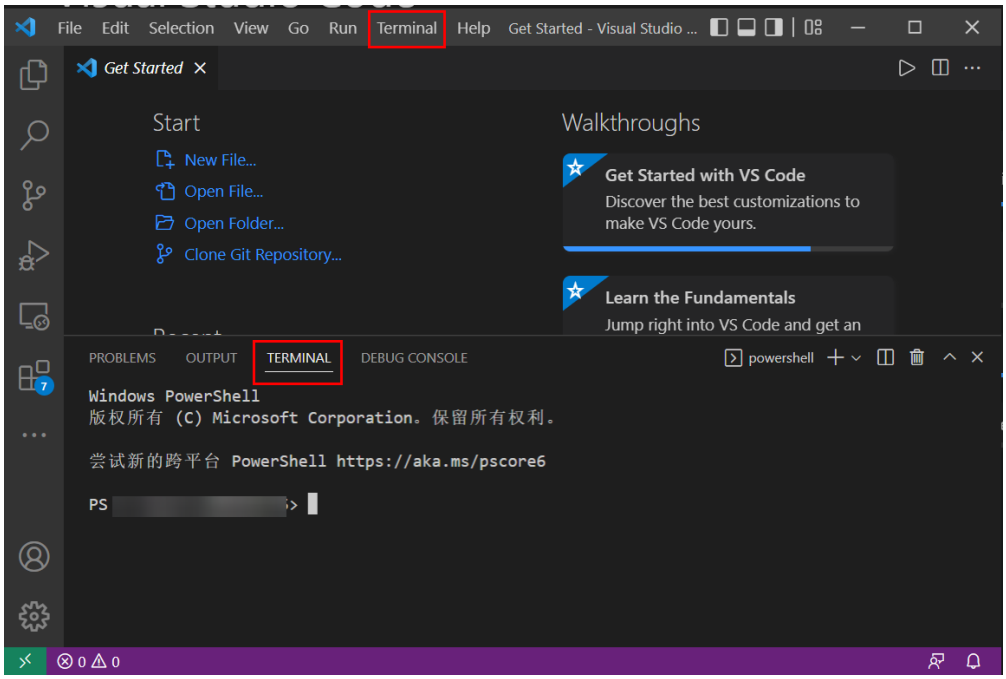


操作步骤

1. 上传数据至OBS。具体操作请参见**上传文件至OBS桶**。或者在本地VS Code的Terminal中使用ModelArts SDK完成数据上传至OBS。

   首先在本地VS Code环境中开启Terminal。

图 5-61 本地 VS Code 环境开启 Terminal



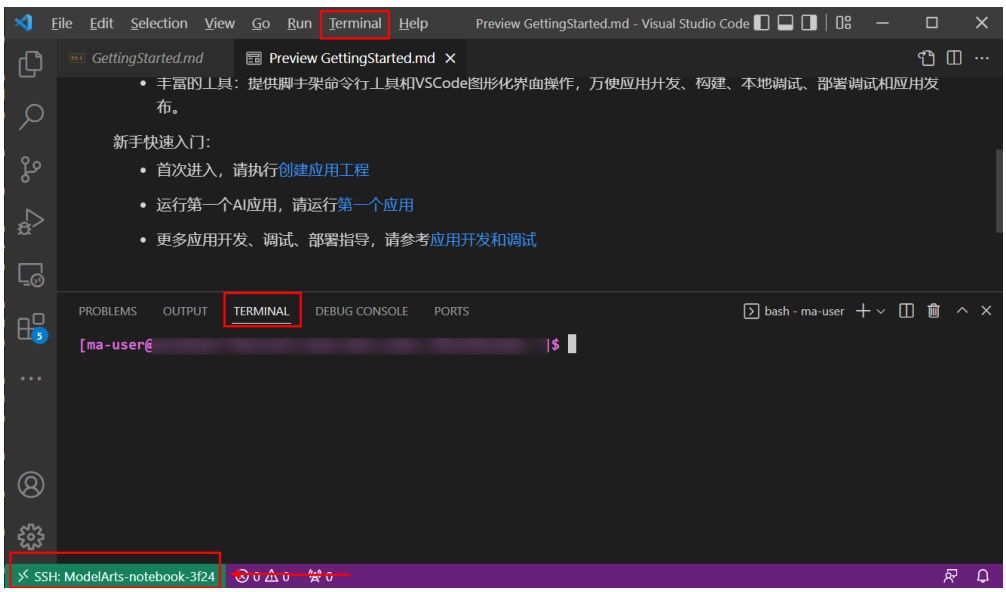
输入python并回车，进入python环境。

```
python
```

在本地VS Code的Terminal中使用ModelArts SDK上传本地文件至OBS，详情请参考**文件传输**进行OBS传输操作。

2. 在远程连接VS Code的Terminal中使用ModelArts SDK下载OBS文件到开发环境的操作示例如下：

图 5-62 远程连接 VS Code 环境开启 Terminal




```
#手动source进入开发环境
cat /home/ma-user/README
#然后选择要source的环境
```



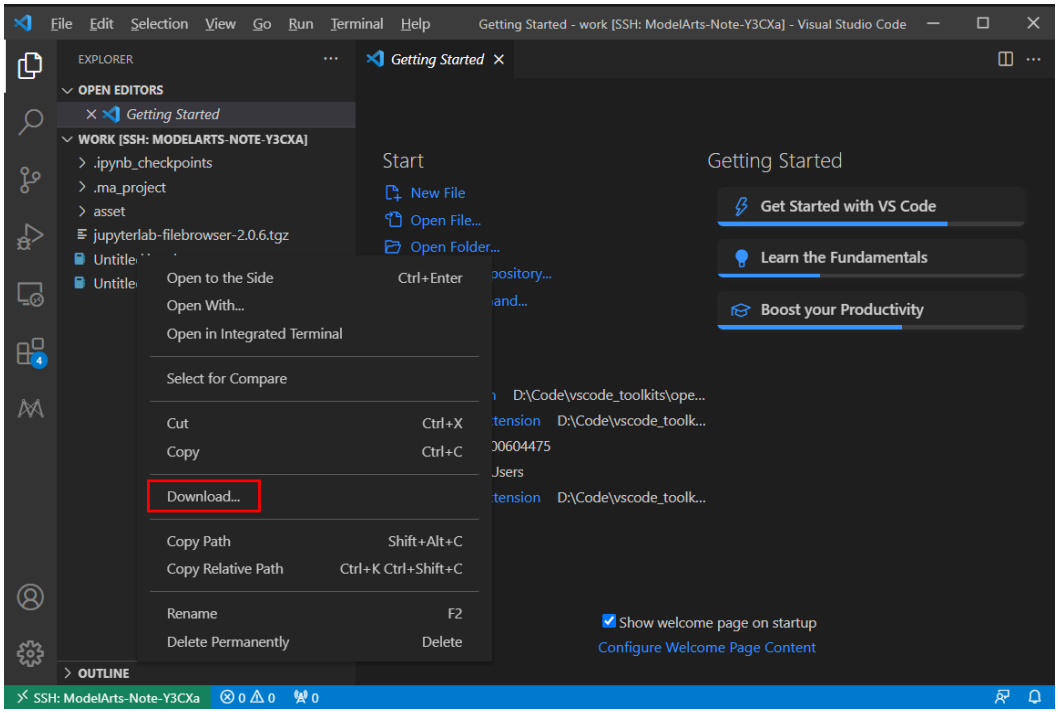
```
source /home/ma-user/miniconda3/bin/activate MindSpore-python3.7-aarch64
#输入python并回车，进入python环境
python
```

然后参考**上传文件至OBS**进行OBS传输操作。

下载 Notebook 中的文件至本地

在Notebook中开发的文件，可以下载至本地。在本地IDE的Project目录下的Notebook2.0工程单击右键，单击“Download...”将文件下载到本地。

图 5-63 VS Code 环境下载 Notebook 中的文件至本地



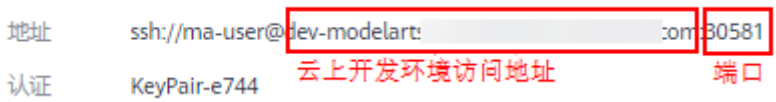
5.4 本地 IDE（SSH 工具连接）

本节操作介绍在Windows环境中使用PuTTY SSH远程登录云上Notebook实例的操作步骤。

前提条件

- 创建一个Notebook实例，并开启远程SSH开发，配置远程访问IP白名单。该实例状态必须处于“运行中”，具体参见**创建Notebook实例**章节。
- 在Notebook实例详情页面获取开发环境访问地址和端口号。

图 5-64 Notebook 实例详情页面



- 准备好密钥对文件。

  密钥对在用户第一次创建时，自动下载，之后使用相同的密钥时不会再有下载界面（用户一定要保存好），或者每次都使用新的密钥对。

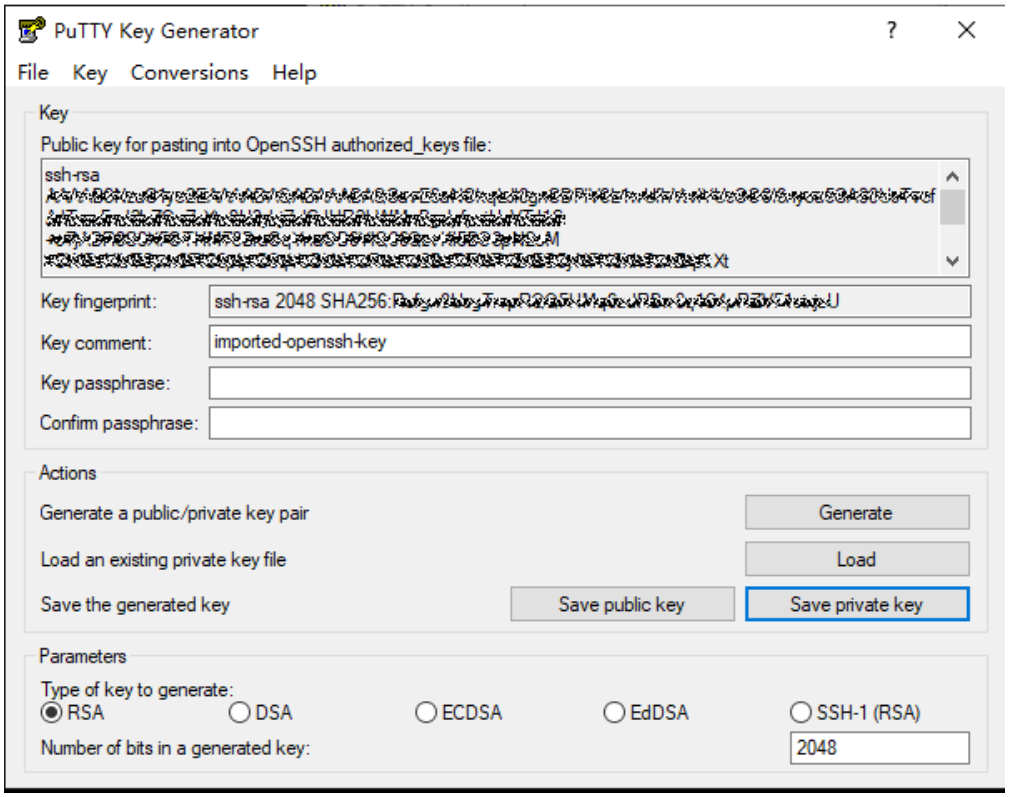
Step1 安装 SSH 工具

下载并安装SSH远程连接工具，以PuTTY为例，**下载链接**。

Step2 使用 puttygen 将密钥对.pem 文件转成.ppk 文件

1. **下载puttygen**，并双击运行puttygen。
2. 单击“Load”，上传.pem密钥（即在创建Notebook实例时创建并保存的密钥对文件）。
3. 单击“Save private key”，保存生成的.ppk文件。.ppk文件的名字可以自定义，例如key.ppk。

图 5-65 将密钥对.pem 文件转成.ppk 文件



Step3 使用 SSH 工具连接云上 Notebook 实例

1. 运行PuTTY。
2. 单击“Session”，填写以下参数。

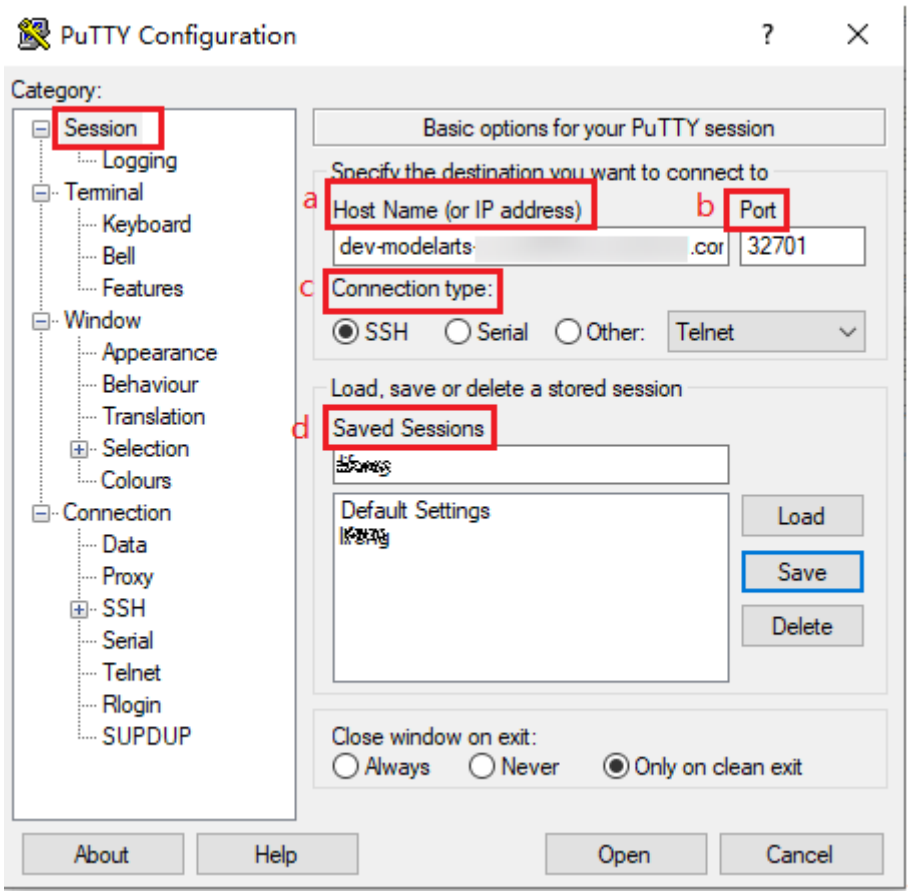
   a. Host Name (or IP address)：云上开发环境Notebook实例的访问地址，即在Notebook实例详情页获取的地址。

   b. Port：云上Notebook实例的端口，即在Notebook实例详情页获取的端口号。例如：32701。

c. Connection Type：选择SSH。

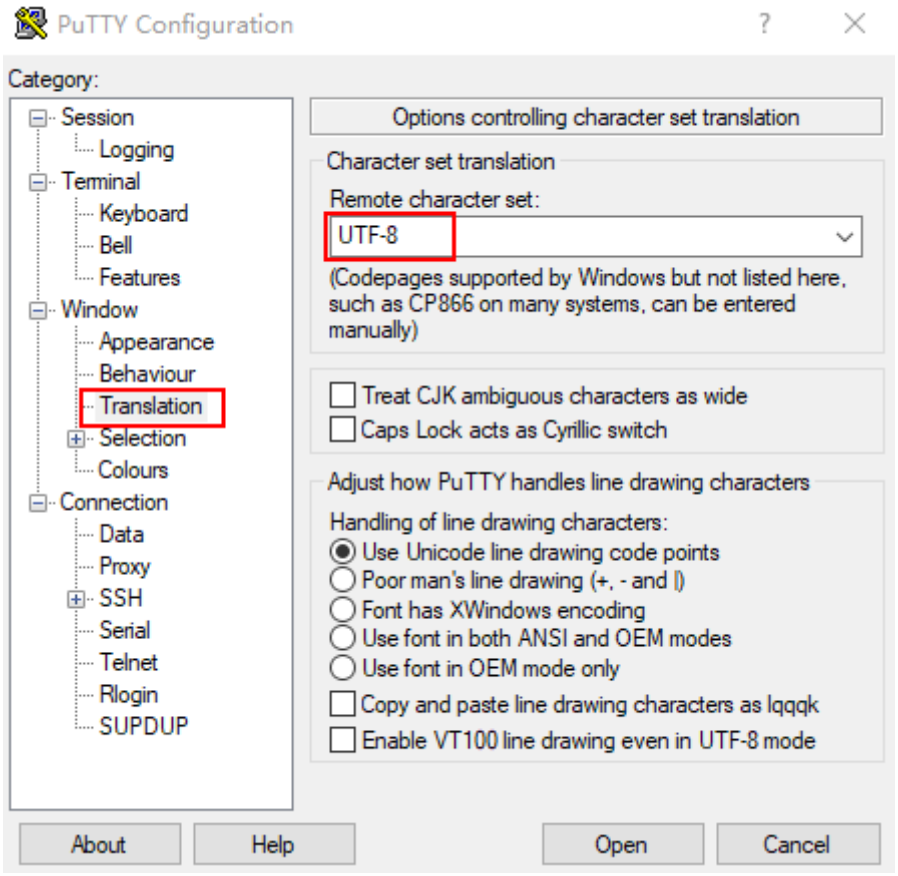
d. Saved Sessions：任务名称，在下一次使用PuTTY时就可以单击保存的任务名称，即可打开远程连接。

图 5-66 设置 Session



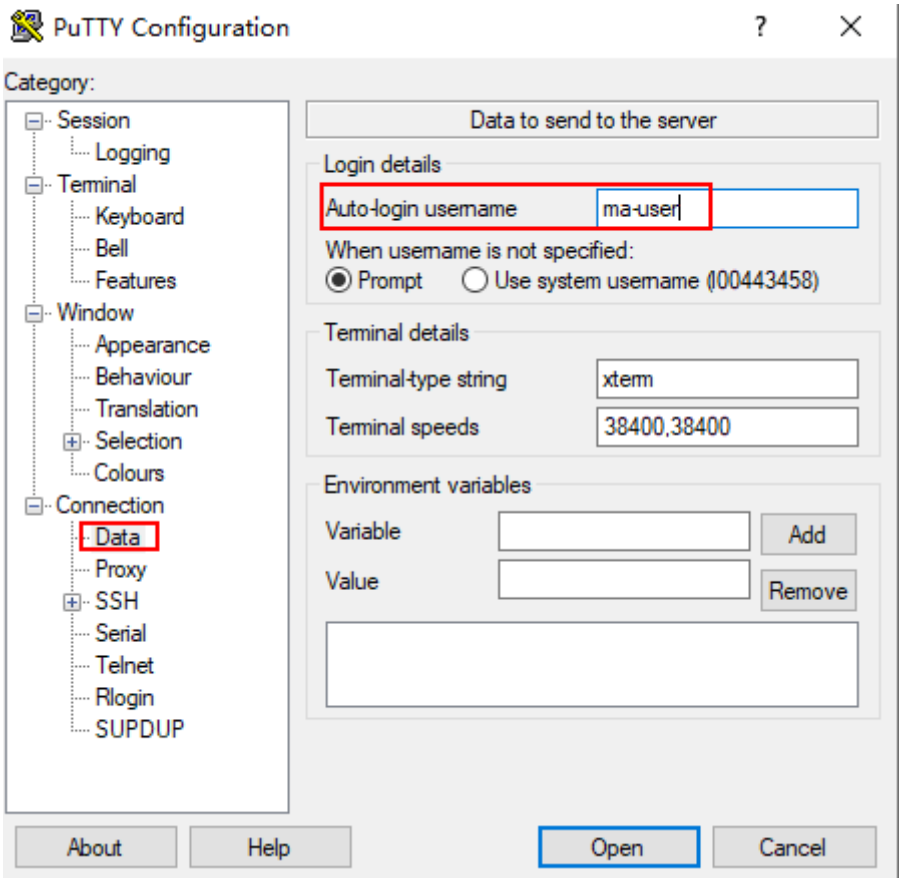
3. 选择“Window > Translation”，在“Remote character set:”中选择“UTF-8”。

图 5-67 设置字符格式

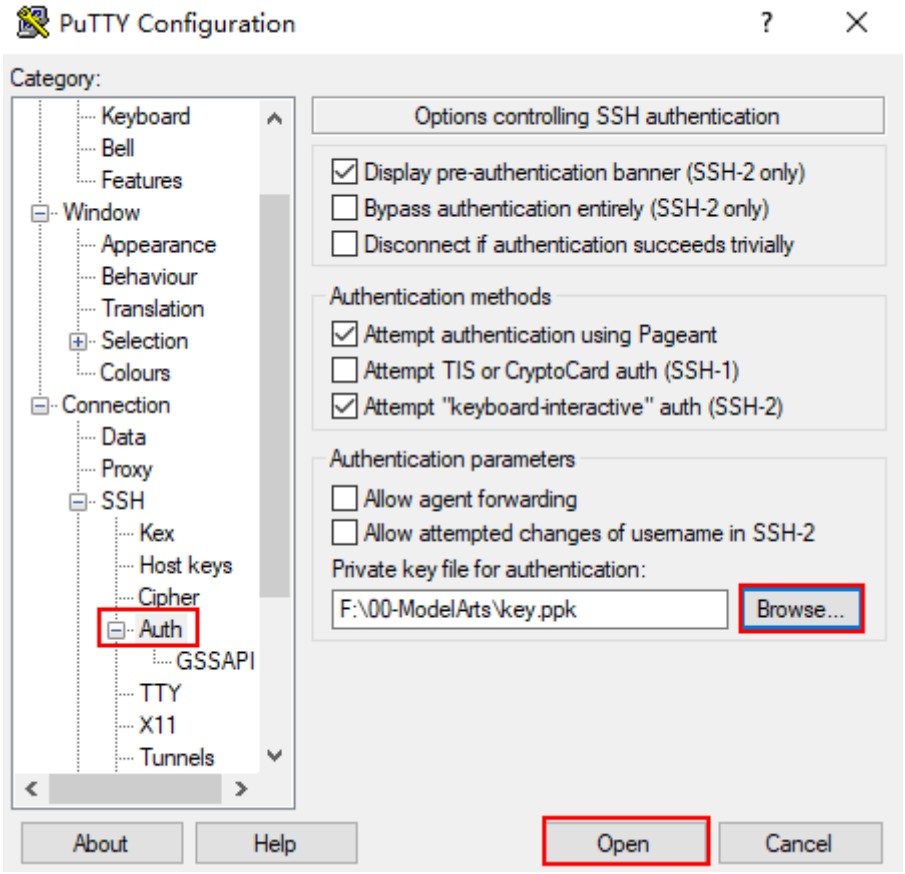


4. 选择“Connection > Data”， 在“Auto-login username” 中填写用户名“ma-user”。

图 5-68 填写用户名

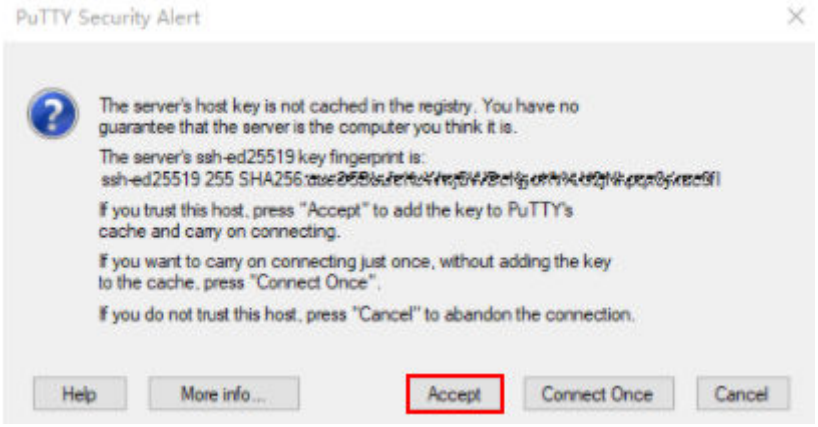


5. 选择“Connection > SSH > Auth”，单击“Browse”，选择“.ppk文件”（由**Step2**密钥对.pem文件生成）。



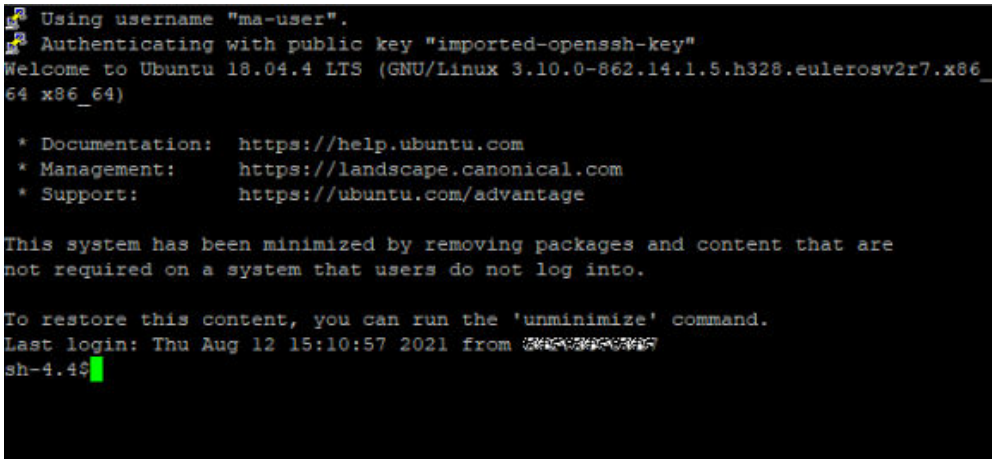
6. 单击“Open”。如果首次登录，PuTTY会显示安全警告对话框，询问是否接受服务器的安全证书。单击“Accept”将证书保存到本地注册表中。

图 5-69 询问是否接受服务器的安全证书



7. 成功连接到云上Notebook实例。

图 5-70 连接到云上 Notebook 实例



6 ModelArts CLI 命令参考

6.1 ModelArts CLI 命令功能介绍

功能介绍

ModelArts CLI，即ModelArts命令行工具，是一个跨平台命令行工具，用于连接ModelArts服务并在ModelArts资源上执行管理命令。用户可以使用交互式命令行提示符或脚本通过终端执行命令。为了方便理解，下面将ModelArts CLI统称为ma-cli。ma-cli支持用户在ModelArts Notebook及线下虚拟机中与云端服务交互，使用ma-cli命令可以实现命令自动补全、鉴权、镜像构建、提交ModelArts训练作业、提交DLI Spark作业、OBS数据复制等。

使用场景

- ma-cli已经集成在ModelArts开发环境Notebook中，可以直接使用。

  登录ModelArts控制台，在“开发环境 > Notebook”中创建Notebook实例，打开Terminal，使用ma-cli命令。
- ma-cli在本地Windows/Linux环境中需要安装后在本地Terminal中使用。安装步骤具体可参考**（可选）本地安装ma-cli**。

  说明

  - ma-cli不支持在git-bash上使用。
  - 推荐使用Linux Bash、ZSH、Fish，WSL或PowerShell等Terminal。在使用过程中，注意您的敏感信息数据保护，避免敏感信息泄露。

命令预览

```
$ ma-cli -h
Usage: ma-cli [OPTIONS] COMMAND [ARGS]...

Options:
  -V, -v, --version       1.2.1
  -C, --config-file TEXT  Configure file path for authorization.
  -D, --debug             Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
  -h, -H, --help          Show this message and exit.

Commands:
```



```
  configure        Configures authentication and endpoints info for the CLI.
  image            Support get registered image list、register or unregister image、debug image, build image
in Notebook.
  obs-copy         Copy file or directory between OBS and local path.
  ma-job           ModelArts job submission and query job details.
  dli-job          DLI spark job submission and query job details.
  auto-completion  Auto complete ma-cli command in terminal, support "bash(default)/zsh/fish".
```

其中，-C、-D、-P，-h参数属于全局可选参数。

- -C表示在执行此命令时可以手动指定鉴权配置文件，默认使用~/.modelarts/ma-cli-profile.yaml配置文件；
- -P表示鉴权文件中的某一组鉴权信息，默认是DEFAULT；
- -D表示是否开启debug模式（默认关闭），当开启debug模式后，命令的报错堆栈信息将会打印出来，否则只会打印报错信息；
- -h表示显示命令的帮助提示信息。

命令说明

表 6-1 ma-cli 支持的命令

命令	命令详情
configure	ma-cli鉴权命令，支持用户名密码、AK/SK
image	ModelArts镜像构建、镜像注册、查询已注册镜像信息等
obs-copy	本地和OBS文件/文件夹间的相互复制
ma-job	ModelArts训练作业管理，包含作业提交、资源查询等
dli-job	DLI Spark任务提交及资源管理
auto-completion	命令自动补全

6.2 （可选）本地安装 ma-cli

使用场景

本文以Windows系统为例，介绍如何在Windows环境中安装ma-cli。

Step1：安装 ModelArts SDK

参考 **本地安装ModelArts SDK**完成SDK的安装。

Step2：下载 ma-cli

1. **下载ma-cli软件包**。
2. 完成软件包签名校验。
   a. **下载软件包签名校验文件**。

b. 安装openssl并执行如下命令进行签名校验。

```
openssl cms -verify -binary -in D:\ma_cli-latest-py3-none-any.whl.cms -inform DER -content
D:\ma_cli-latest-py3-none-any.whl  -noverify > ./test
```

说明

本示例以软件包在D:\举例，请根据软件包实际路径修改。

Step3：安装 ma-cli

1. 在本地环境cmd中执行命令**python --version**，确认环境已经安装完成Python。（Python版本需大于3.7.x且小于3.10.x版本，推荐使用3.7.x版本）

```
C:\Users\xxx>python --version
Python *.*.*
```

2. 执行命令**pip --version**，确认Python通用包管理工具pip已经存在。

```
C:\Users\xxx>pip --version
pip **.*.* from c:\users\xxx\appdata\local\programs\python\python**\lib\site-packages\pip (python *.*)
```

3. 执行如下命令，安装ma-cli。

pip install *{ma-cli软件包路径}*\ma_cli-latest-py3-none-any.whl

```
C:\Users\xxx>pip install C:\Users\xxx\Downloads\ma_cli-latest-py3-none-any.whl
......
Successfully installed ma_cli.*.*.*
```

在安装ma-cli时会默认同时安装所需的依赖包。当显示“Successfully installed”时，表示ma-cli安装完成。

说明

如果在安装过程中报错提示缺少相应的依赖包，请根据报错提示执行如下命令进行依赖包安装。

pip install xxxx

其中，xxxx为依赖包的名称。

6.3 ma-cli auto-completion 自动补全命令

命令行自动补全是指用户可以在Terminal中输入命令前缀通过Tab键自动提示支持的ma-cli命令。ma-cli自动补全功能需要手动在Terminal中激活。执行**ma-cli auto-completion**命令，用户根据提示的补全命令，复制并在当前Terminal中执行，就可以自动补全ma-cli的命令。目前支持Bash、Fish及Zsh三种Shell，默认是Bash。

以Bash命令为例：在Terminal中执行**eval "$(_MA_CLI_COMPLETE=bash_source ma-cli)"**激活自动补全功能。

```
eval "$(_MA_CLI_COMPLETE=bash_source ma-cli)"
```

此外，可以通过“ma-cli auto-completion Fish”或“ma-cli auto-completion Fish”命令查看“Zsh”、“Fish”中的自动补全命令。

命令概览

```
$ ma-cli auto-completion -h
Usage: ma-cli auto-completion [OPTIONS] [[Bash|Zsh|Fish]]

  Auto complete ma-cli command in terminal.

  Example:
```

```
  # print bash auto complete command to terminal
  ma-cli auto-completion Bash

Options:
  -H, -h, --help  Show this message and exit.
```

```
# 默认显示Bash Shell自动补全命令

$ ma-cli auto-completion

Tips: please paste following shell command to your terminal to activate auto complation.

[ OK ] eval "$(_MA_CLI_COMPLETE=bash_source ma-cli)"

# 执行上述命令，此时Terminal已经支持自动补全

$ eval "$(_MA_CLI_COMPLETE=bash_source ma-cli)"
```

```
# 显示Fish Shell自动补全命令
$ ma-cli auto-completion Fish
Tips: please paste following shell command to your terminal to activate auto complation.

[ OK ] eval (env _MA_CLI_COMPLETE=fish_source ma-cli)
```

6.4 ma-cli configure 鉴权命令

鉴权信息说明

- 在虚拟机及个人PC场景，需要配置鉴权信息，目前支持用户名密码鉴权（默认）和AK/SK鉴权；
- 在使用账号认证时，需要指定username和password；在使用IAM用户认证时，需要指定account、username和password；
- 在ModelArts Notebook中可以不用执行鉴权命令，默认使用委托信息，不需要手动进行鉴权操作；
- 如果用户在ModelArts Notebook中也配置了鉴权信息，那么将会优先使用用户指定的鉴权信息。

  说明

  在鉴权时，注意您的敏感信息数据保护，避免敏感信息泄露。

命令参数总览

```
$ ma-cli configure -h
Usage: ma-cli configure [OPTIONS]

Options:
  -auth, --auth [PWD|AKSK|ROMA]  Authentication type.
  -rp, --region-profile PATH      ModelArts region file path.
  -a, --account TEXT              Account of an IAM user.
  -u, --username TEXT             Username of an IAM user.
  -p, --password TEXT             Password of an IAM user
  -ak, --access-key TEXT          User access key.
  -sk, --secret-key TEXT          User secret key.
  -r, --region TEXT               The region you want to visit.
  -pi, --project-id TEXT          User project id.
  -C, --config-file TEXT          Configure file path for authorization.
  -D, --debug                     Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT              CLI connection profile to use. The default profile is "DEFAULT".
  -h, -H, --help                  Show this message and exit.
```

表 6-2 鉴权命令参数说明

参数名	参数类型	是否必选	参数说明
-auth / --auth	String	否	鉴权方式，支持PWD（用户名密码）、AKSK（access key和secret key），默认是PWD。
-rp / --region-profile	String	否	指定ModelArts region配置文件信息。
-a / --account	String	否	IAM租户账号，在使用IAM用户认证场景时需要指定，属于PWD鉴权的一部分。
-u / --username	String	否	用户名，在使用账号认证时表示账号名，IAM认证时表示IAM用户名，在云星账号场景不需要指定，属于PWD鉴权的一部分。
-p / --password	String	否	密码，属于PWD鉴权的一部分。
-ak / --access-key	String	否	access key，属于AKSK鉴权的一部分。
-sk / --secret-key	String	否	secret key，属于AKSK鉴权的一部分。
-r / --region	String	否	region名称，如果不填会默认使用REGION_NAME环境变量的值。
-pi / --project-id	String	否	项目ID，如果不填会默认使用对应region的值，或者使用PROJECT_ID环境变量。
-P / --profile	String	否	鉴权配置项，默认是DEFAULT。
-C / --config-file	String	否	配置文件本地路径，默认路径为~/.modelarts/ma-cli-profile.yaml。

配置用户名密码鉴权

以在虚拟机上使用**ma-cli configure**为例，介绍如何配置用户名密码进行鉴权。

说明

以下样例中所有以${}装饰的字符串都代表一个变量，用户可以根据实际情况指定对应的值。

比如${your_password}表示输入用户自己的密码信息。

```
# 默认使用DEFAULT鉴权配置项，默认提示账号、用户名及密码（其中账号和用户名如果不需要填写可以使用Enter跳过）
$ ma-cli configure --auth PWD --region ${your_region}
account: ${your_account}
username: ${your_username}
password: ${your_password}  # 输入在控制台不会回显
```

AKSK 鉴权

如下命令表示使用AKSK进行鉴权，需要交互式输入AK及SK信息。默认提示AK和SK，且输入在控制台不会回显。

⚠ 注意

以下样例中所有以${}装饰的字符串都代表一个变量，用户可以根据实际情况指定对应的值。

比如${access key}表示输入用户自己的access key。

```
ma-cli configure --auth AKSK
access key [***]: ${access key}
secret key [***]: ${secret key}
```

执行完鉴权命令后，将会在~/.modelarts/ma-cli-profile.yaml配置文件中保存相应的鉴权信息。

6.5 使用 ma-cli image 构建镜像

6.5.1 ma-cli image 镜像构建支持的命令

ma-cli image命令支持：查询用户已注册的镜像、查询/加载镜像构建模板、Dockerfile镜像构建、查询/清理镜像构建缓存、注册/取消注册镜像、调试镜像是否可以在Notebook中使用等。具体命令及功能可执行**ma-cli image -h**命令查看。

镜像构建命令总览

```
$ ma-cli image -h
Usage: ma-cli image [OPTIONS] COMMAND [ARGS]...
  Support get registered image list, register or unregister image, debug image, build image in Notebook.

Options:
  -H, -h, --help  Show this message and exit.

Commands:
  add-template, at  List build-in dockerfile templates.
  build             Build docker image in Notebook.
  debug             Debug SWR image as a Notebook in ECS.
  df                Query disk usage.
  get-image, gi     Query registered image in ModelArts.
  get-template, gt  List build-in dockerfile templates.
  prune             Prune image build cache.
  register          Register image to ModelArts.
  unregister        Unregister image from ModelArts.
```

表 6-3 镜像构建支持的命令

命令	命令详情
get-template	查询镜像构建模板。

命令	命令详情
add-template	加载镜像构建模板。
get-image	查询ModelArts已注册镜像。
register	注册SWR镜像到ModelArts镜像管理。
unregister	取消注册ModelArts镜像管理中的已注册镜像。
build	基于指定的Dockerfile构建镜像 （只支持ModelArts Notebook里使用）。
df	查询镜像构建缓存（只支持ModelArts Notebook里使用）。
prune	清理镜像构建缓存 （只支持ModelArts Notebook里使用）。
debug	在ECS上调试SWR镜像是否能在ModelArts Notebook中使用 （只支持已安装docker环境的ECS）。

6.5.2 使用 ma-cli image get-template 命令查询镜像构建模板

ma-cli提供了一些常用的镜像构建模板，模板中包含了在ModelArts Notebook上进行Dockerfile开发的牵引指导。

```
$ ma-cli image get-template -h
Usage: ma-cli image get-template [OPTIONS]

  List build-in dockerfile templates.

  Example:

  # List build-in dockerfile templates
  ma-cli image get-template [--filer <filter_info>]  [--page-num <yourPageNum>] [--page-size
<yourPageSize>]

Options:
  --filter TEXT                   filter by keyword.
  -pn, --page-num INTEGER RANGE   Specify which page to query.  [x>=1]
  -ps, --page-size INTEGER RANGE  The maximum number of results for this query.  [x>=1]
  -D, --debug                     Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT              CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help                  Show this message and exit.
(PyTorch-1.4) [ma-user work]$
```

表 6-4 参数说明

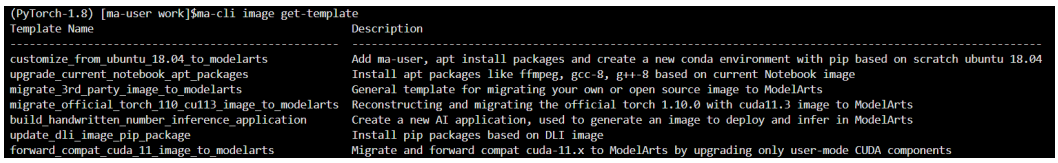
参数名	参数类型	是否必选	参数说明
--filter	String	否	根据模板名称关键字过滤模板列表。
-pn / --page-num	Int	否	镜像页索引，默认是第1页。

参数名	参数类型	是否必选	参数说明
-ps / --page-size	Int	否	每页显示的镜像数量，默认是20。

示例

查看镜像构建模板。

```
ma-cli image get-template
```



6.5.3 使用 ma-cli image add-template 命令加载镜像构建模板

ma-cli可以使用**add-template**命令将镜像模板加载到指定文件夹下，默认路径为当前命令所在的路径。

比如${current_dir}/.ma/${template_name}/。也可以通过--dest命令指定保存的路径。当保存的路径已经有同名的模板文件夹时，可以使用--force | -f参数进行强制覆盖。

```
$ ma-cli image add-template -h
Usage: ma-cli image add-template [OPTIONS] TEMPLATE_NAME

  Add buildin dockerfile templates into disk.

  Example:

  # List build-in dockerfile templates
  ma-cli image add-template customize_from_ubuntu_18.04_to_modelarts --force

Options:
  --dst TEXT          target save path.
  -f, --force         Override templates that has been installed.
  -D, --debug         Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT  CLI connection profile to use. The default profile is "DEFAULT".
  -h, -H, --help      Show this message and exit.
```

表 6-5 参数说明

参数名	参数类型	是否必选	参数说明
--dst	String	否	加载模板到指定路径，默认是当前路径。
-f / --force	Bool	否	是否强制覆盖已存在的同名模板，默认不覆盖。

示例

加载**customize_from_ubuntu_18.04_to_modelarts**镜像构建模板。

```
ma-cli image add-template customize_from_ubuntu_18.04_to_modelarts
```

```
(PyTorch-1.8) [ma-user work]$ma-cli image add-template customize_from_ubuntu_18.04_to_modelarts
[ OK ] Successfully add configuration template [ customize_from_ubuntu_18.04_to_modelarts ] under folder [ /home/ma-user/work/.ma/customize_from_ubuntu_18.04_to_modelarts ]
```

6.5.4 使用 ma-cli image get-image 查询 ModelArts 已注册镜像

Dockerfile一般需要提供一个基础镜像的地址，目前支持从docker hub等开源镜像仓拉取公开镜像，以及SWR的公开或私有镜像。其中ma-cli提供了查询ModelArts预置镜像和用户已注册镜像列表及SWR地址。

```
$ma-cli image get-image -h
Usage: ma-cli image get-image [OPTIONS]

  Get registered image list.

  Example:

  # Query images by image type and only image id, show name and swr_path
  ma-cli image get-image --type=DEDICATED

  # Query images by image id
  ma-cli image get-image --image-id ${image_id}

  # Query images by image type and show more information
  ma-cli image get-image --type=DEDICATED -v

  # Query images by image name
  ma-cli image get-image --filter=torch

Options:
  -t, --type [BUILD_IN|DEDICATED|ALL]
                                  Image type(default ALL)
  -f, --filter TEXT               Image name to filter
  -v, --verbose                   Show detailed information on image.
  -i, --image-id TEXT             Get image details by image id
  -n, --image-name TEXT           Get image details by image name
  -wi, --workspace-id TEXT        The workspace where you want to query image(default "0")
  -pn, --page-num INTEGER RANGE   Specify which page to query  [x>=1]
  -ps, --page-size INTEGER RANGE  The maximum number of results for this query  [x>=1]
  -C, --config-file PATH          Configure file path for authorization.
  -D, --debug                     Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT              CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help                  Show this message and exit.
```

表 6-6 参数说明

参数名	参数类型	是否必选	参数说明
-t / --type	String	否	查询的镜像类型，支持BUILD_IN、DEDICATED和ALL三种查询类型。 ● BUILD_IN：预置镜像 ● DEDICATED：用户已注册的自定义镜像 ● ALL：所有镜像

参数名	参数类型	是否必选	参数说明
-f / --filter	String，	否	镜像名关键字。根据镜像名关键字过滤镜像列表。
-v / --verbose	Bool	否	显示详细的信息开关，默认关闭。
-i / --image-id	String	否	查询指定镜像ID的镜像详情。
-n / --image-name	String	否	查询指定镜像名称的镜像详情。
-wi / --workspace-id	String	否	查询指定工作空间下的镜像信息。
-pn / --page-num	Int	否	镜像页索引，默认是第1页。
-ps / --page-size	Int	否	每页显示的镜像数量，默认是20。

示例

查询ModelArts已注册的自定义镜像。

```
ma-cli image get-image --type=DEDICATED
```

```
(PyTorch-1.8) [ma-user work]$ma-cli image get-image --type=DEDICATED
+-------+------------------------------------+-------------+------------------------------------------------------+
| INDEX |              IMAGE ID              |    NAME     |                       SWR PATH                       |
+-------+------------------------------------+-------------+------------------------------------------------------+
|   1   | c857e5a8[redacted]fc5e3d002f | 0314test    | [redacted]/notebook_test/0314test:1.0.0 |
|   2   | 193b2557[redacted]d39093a811 | 0328        | [redacted].com/notebook_test/0328:1 |
|   3   | 171fe036[redacted]3b37e9aa7c | 0926        | [redacted]i_modelarts_y00218826_05/0926:1 |
|   4   | 1b48bb0a[redacted]689b0a7267 | 0926        | sw[redacted]_modelarts_y00218826_05/0926:111 |
|   5   | c8667cf0[redacted]d2e3563107 | 1           | [redacted]/ei_modelarts_y00218826_05/1:6 |
|   6   | 3e6cda6a[redacted]a360eea80e | 1           | [redacted]/ei_modelarts_y00218826_05/1:1 |
|   7   | 42e86ca5[redacted]ec198be968 | 111         | [redacted]com/notebook_test/111:1227 |
|   8   | 0f349cef[redacted]c411011ef2 | 11111110801 | [redacted]otebook_test/11111110801:111111 |
|   9   | 3a082e32[redacted]4f485aadb6 | 112121      | swr[redacted]modelarts_y00218826_05/112121:123 |
|  10   | db0d02f6[redacted]74eb00e1ce | 1203        | [redacted]om/notebook_test/1203:1.2.3 |
|  11   | 031dc02e[redacted]d92cd457d8 | 1227        | [redacted]com/notebook_test/1227:111 |
|  12   | f7d95648[redacted]7aaec8b1cc | 1227        | [redacted]com/notebook_test/1227:888 |
|  13   | 2f720610[redacted]a1d1db9d7d | 1227        | [redacted]com/notebook_test/1227:6666 |
|  14   | 42221bf2[redacted]22d726d270 | 1229        | [redacted]com/notebook_test/1229:123 |
|  15   | 70deea1e[redacted]70b2414ae7 | 123         | [redacted]om/mindspore-dis-train/123:2 |
|  16   | e6cc5414[redacted]ce318069f4 | 123         | [redacted]com/notebook_test/123:45678 |
|  17   | 6e7a86c9[redacted]319fb3bb28 | 1234        | [redacted]com/notebook_test/1234:666 |
|  18   | ec036306[redacted]8c9dc6b391 | 1234        | [redacted].com/notebook_test/1234:1 |
|  19   | b37f8f3b[redacted]7a9941c978 | 441211      | [redacted]com/notebook_test/441211:11 |
|  20   | d5acd51b[redacted]ef16534d68 | aaa         | [redacted]com/notebook_test/aaa:1.1.1 |
```

6.5.5 使用 ma-cli image build 命令在 ModelArts Notebook 中进行镜像构建

使用**ma-cli image build**命令基于指定的Dockerfile进行镜像构建，仅支持在ModelArts Notebook里使用该命令。

```
$ ma-cli image build -h
Usage: ma-cli image build [OPTIONS] FILE_PATH

  Build docker image in Notebook.

  Example:

  # Build a image and push to SWR
  ma-cli image build .ma/customize_from_ubuntu_18.04_to_modelarts/Dockerfile -swr my_organization/
my_image:0.0.1

  # Build a image and push to SWR, dockerfile context path is current dir
  ma-cli image build .ma/customize_from_ubuntu_18.04_to_modelarts/Dockerfile -swr my_organization/
my_image:0.0.1 -context .

  # Build a local image and save to local path and OBS
  ma-cli image build .ma/customize_from_ubuntu_18.04_to_modelarts/Dockerfile --target ./build.tar --
obs_path obs://bucket/object --swr-path my_organization/my_image:0.0.1

Options:
  -t, --target TEXT        Name and optionally a tag in the 'name:tag' format.
  -swr, --swr-path TEXT    SWR path without swr endpoint, eg:organization/image:tag.  [required]
  --context DIRECTORY      build context path.
  -arg, --build-arg TEXT   build arg for Dockerfile.
  -obs, --obs-path TEXT    OBS path to save local built image.
  -f, --force              Force to overwrite the existing swr image with the same name and tag.
  -C, --config-file PATH   Configure file path for authorization.
  -D, --debug              Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT       CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help           Show this message and exit.
```

表 6-7 参数说明

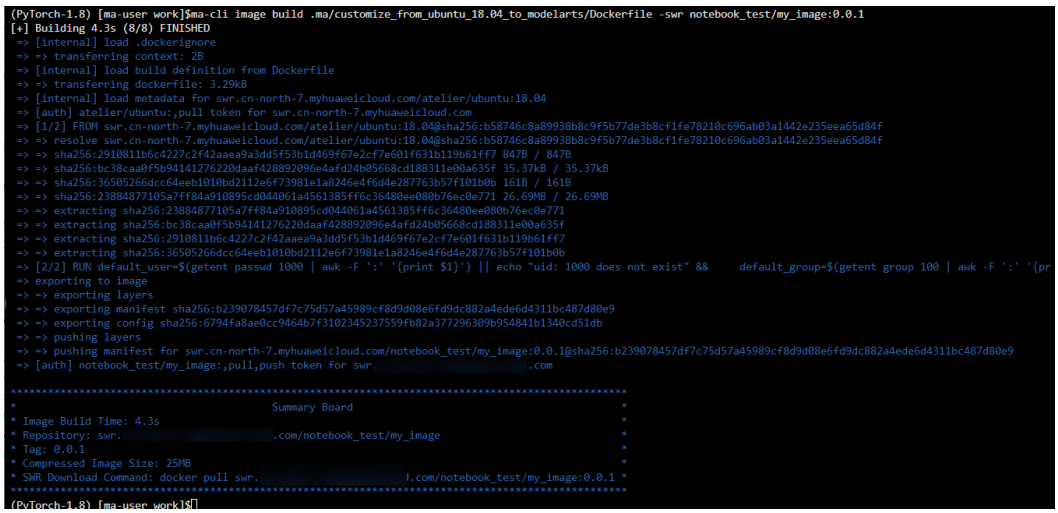
参数名	参数类型	是否必选	参数说明
FILE_PATH	String	是	Dockerfile文件所在的路径。
-t / --target	String	否	表示构建生成的tar包保存在本地的路径，默认是当前文件夹目录。
-swr / --swr-path	String	是	SWR镜像名称，遵循organization/image_name:tag格式，针对于构建保存tar包场景可以省略。
--context	String	否	Dockerfile构建时的上下文信息路径，主要用于数据复制。
-arg / --build-arg	String	否	指定构建参数，多个构建参数可以使用--build-arg VERSION=18.04 --build-arg ARCH=X86_64
-obs / --obs-path	String	否	将生成的tar包自动上传到OBS中。
-f / --force	Bool	否	是否强制覆盖已存在的SWR镜像，默认不覆盖。

示例

在ModelArts Notebook里进行镜像构建。

```
ma-cli image build .ma/customize_from_ubuntu_18.04_to_modelarts/Dockerfile -swr
notebook_test/my_image:0.0.1
```

其中“.ma/customize_from_ubuntu_18.04_to_modelarts/Dockerfile”为Dockerfile文件所在路径，“notebook_test/my_image:0.0.1”为构建的新镜像的SWR路径。



6.5.6 使用 ma-cli image df 命令在 ModelArts Notebook 中查询镜像构建缓存

使用**ma-cli image df**命令查询镜像构建缓存，仅支持在ModelArts Notebook里使用该命令。

```
$ ma-cli image df -h
Usage: ma-cli image df [OPTIONS]

  Query disk usage used by image-building in Notebook.

  Example:

  # Query image disk usage
  ma-cli image df

Options:
  -v, --verbose      Show detailed information on disk usage.
  -D, --debug        Debug Mode. Shows full stack trace when error occurs.

  -h, -H, --help     Show this message and exit.
```

表 6-8 参数说明

参数名	参数类型	是否必选	参数说明
-v / --verbose	Bool	否	显示详细的信息开关，默认关闭。

示例

- 在ModelArts Notebook里查看所有镜像缓存。
  ma-cli image df

```
(PyTorch-1.8) [ma-user work]$ma-cli image df
ID                                                                RECLAIMABLE     SIZE            LAST ACCESSED
iwrrwsi9pdcjafe1ij6d0r918                                         true            98.50MB
cp52c4q81ud2abu2vp7sj5vyt                                         true            1.04MB
4jbo6v06r2w1575ddq3w8g12e                                         true            139.68kB
ojdjw5mok71s1nh2cauant051                                         true            86.86kB
k2jm6g061n5twmz7gmonmqjsh                                         true            16.55kB
efu5kwgig1ve44fe7smbrncnh*                                        true            8.19kB
uzikwqk5taxnslvajm14jrbje*                                        true            4.10kB
2g8p0qcb014g3qva7ucawkv87*                                        true            4.10kB
Reclaimable:    99.80MB
Total:          99.80MB
```

- 显示镜像的详细信息。
  ma-cli image df --verbose

```
(PyTorch-1.8) [ma-user work]$ma-cli image df --verbose
ID:             iwrrwsi9pdcjafe1ij6d0r918
Created at:     2023-03-28 12:23:28.353759532 +0000 UTC
Mutable:        false
Reclaimable:    true
Shared:         false
Size:           98.50MB
Description:    pulled from swr                          .com/atelier/ubuntu:18.04@sha256:b5874                                                                ea65d84f
Usage count:    1
Last used:      2023-03-28 12:23:28.37337776 +0000 UTC
Type:           regular

ID:             cp52c4q81ud2abu2vp7sj5vyt
Parents:        iwrrwsi9pdcjafe1ij6d0r918
Created at:     2023-03-28 12:23:28.366910223 +0000 UTC
Mutable:        false
Reclaimable:    true
Shared:         false
Size:           1.04MB
Description:    pulled from swr.                         .com/atelier/ubuntu:18.04@sha256:b5874                                                                ?e235eea65d84f
Usage count:    1
Last used:      2023-03-28 12:23:28.38560437 +0000 UTC
Type:           regular

ID:             4jbo6v06r2w1575ddq3w8g12e
Parents:        k2jm6g061n5twmz7gmonmqjsh
Created at:     2023-03-28 12:23:30.681643727 +0000 UTC
Mutable:        false
Reclaimable:    true
Shared:         false
Size:           139.68kB
Description:    mount / from exec /bin/sh -c default_user=$(getent passwd 1000 | awk -F ':' '{print $1}') || echo "uid: 1000 does not exist" &&     default_group=$(getent
up 100 | awk -F ':' '{print $1}') || echo "gid: 100 does not exist" &&     if [ ! -z ${default_user} ] && [ ${default_user} != "ma-user" ]; then         userdel -r ${defau
user};     fi &&     if [ ! -z ${default_group} ] && [ ${default_group} != "ma-group" ]; then         groupdel -f ${default_group};     fi &&     groupadd -g 100 ma-group
useradd -d /home/ma-user -m -u 1000 -g 100 -s /bin/bash ma-user &&     chmod -R 750 /home/ma-user
Usage count:    2
Last used:      2023-03-28 12:25:39.149080471 +0000 UTC
Type:           regular
```

6.5.7 使用 ma-cli image prune 命令在 ModelArts Notebook 中清理镜像构建缓存

使用**ma-cli image prune**命令清理镜像构建缓存，仅支持在ModelArts Notebook里使用该命令。

```
$ ma-cli image prune -h
Usage: ma-cli image prune [OPTIONS]

  Prune image build cache by image-building in Notebook.

  Example:

  # Prune image build cache
  ma-cli image prune

Options:
  -ks, --keep-storage INTEGER  Amount of disk space to keep for cache below this limit (in MB) (default: 0).
  -kd, --keep-duration TEXT    Keep cache newer than this limit, support second(s), minute(m) and hour(h)
(default:   0s).
  -v, --verbose                Show more verbose output.
  -D, --debug                  Debug Mode. Shows full stack trace when error occurs.
  -h, -H, --help               Show this message and exit.
```

表 6-9 参数说明

参数名	参数类型	是否必选	参数说明
-ks / --keep-storage	Int	否	清理缓存时保留的缓存大小，单位是MB，默认是0，表示全部清理。
-kd / --keep-duration	String	否	清理缓存时保留较新的缓存，只清除历史缓存，单位为s（秒）、m（分钟）、h（小时），默认是0s，表示全部清理。
-v / --verbose	Bool	否	显示详细的信息开关，默认关闭。

示例

清理保留1MB镜像缓存。

```
ma-cli image prune -ks 1
```

```
(PyTorch-1.8) [ma-user work]$ma-cli image prune -ks 1
ID                                                              RECLAIMABLE     SIZE    LAST ACCESSED
uzikwqk5taxnslvajm14jrbje*                                      true            4.10kB
4jbo6v06r2wl575ddq3w8g12e                                       true            139.68kB
k2jm6g061n5twmz7gmonmqjsh                                       true            16.55kB
ojdjw5mok71s1nh2cauant051                                       true            86.86kB
cp52c4q81ud2abu2vp7sj5vyt                                       true            1.04MB
iwrrwsi9pdcjafe1ij6d0r918                                       true            98.50MB
Total:  99.79MB
```

6.5.8 使用 ma-cli image register 命令注册 SWR 镜像到 ModelArts 镜像管理

调试完成后，使用**ma-cli image register**命令将新镜像注册到ModelArts镜像管理服务中，进而在能够在ModelArts中使用该镜像。

```
$ma-cli image register -h
Usage: ma-cli image register [OPTIONS]

  Register image to ModelArts.

  Example:

  # Register image into ModelArts service
  ma-cli image register --swr-path=xx

  # Share SWR image to DLI service
  ma-cli image register -swr xx -td

  # Register image into ModelArts service and specify architecture to be 'AARCH64'
  ma-cli image register --swr-path=xx --arch AARCH64

Options:
  -swr, --swr-path TEXT          SWR path without swr endpoint, eg:organization/image:tag.  [required]
  -a, --arch [X86_64|AARCH64]    Image architecture (default: X86_64).
  -s, --service [NOTEBOOK|MODELBOX]
                                 Services supported by this image(default NOTEBOOK).
  -rs, --resource-category [CPU|GPU|ASCEND]
                                 The resource category supported by this image (default: CPU and GPU).
  -wi, --workspace-id TEXT       The workspace to register this image (default: "0").
  -v, --visibility [PUBLIC|PRIVATE]
```

```
                        PUBLIC: every user can use this image. PRIVATE: only image owner can use this
image (Default: PRIVATE).
  -td, --to-dli                 Register swr image to DLI, which will share SWR image to DLI service.
  -d, --description TEXT            Image description (default: "").
  -C, --config-file PATH            Configure file path for authorization.
  -D, --debug                   Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT             CLI connection profile to use. The default profile is "DEFAULT".
  -h, -H, --help                Show this message and exit.
```

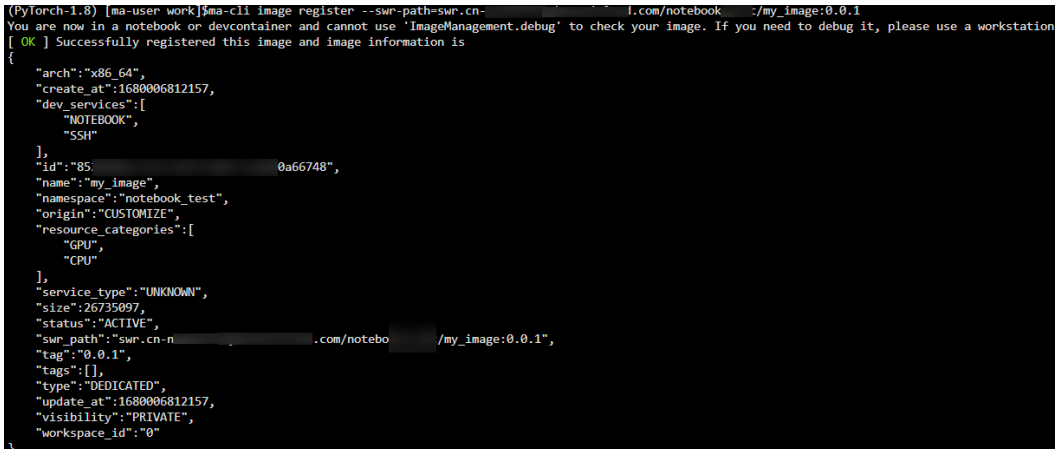
表 6-10 参数说明

参数名	参数类型	是否必选	参数说明
-swr / --swr-path	String	是	需要注册的镜像的SWR路径。
-a / --arch	Sring	否	注册镜像的架构，X86_64或者AARCH64，默认是X86_64。
-s / --service	String	否	注册镜像的服务类型，NOTEBOOK或者MODELBOX，默认是NOTEBOOK。 可以输入多个值，如-s NOTEBOOK -s MODELBOX。
-rs / --resource-category	String	否	注册镜像能够使用的资源类型，默认是CPU和GPU。
-wi / --workspace-id	String	否	注册镜像到指定的工作空间，workspace ID默认是0。
-v / --visibility	Bool	否	注册的镜像可见性，PRIVATE（仅自己可见）或者PUBLIC（所有用户可见），默认是PRIVATE。
-td / --to-dli	Bool	否	注册镜像到DLI服务。
-d/ --description	String	否	填写镜像描述，默认为空。

示例

注册SWR镜像到ModelArts。

```
ma-cli image register --swr-path=xx
```



6.5.9 使用 ma-cli image unregister 命令取消 ModelArts 中的已注册镜像

使用**ma-cli image unregister**命令将注册的镜像从ModelArts中删除。

```
$ ma-cli image unregister -h
Usage: ma-cli image unregister [OPTIONS]

  Unregister image from ModelArts.

  Example:

  # Unregister image
  ma-cli image unregister --image-id=xx

  # Unregister image and delete it from swr
  ma-cli image unregister --image-id=xx -d

Options:
  -i, --image-id TEXT     Unregister image details by image id.  [required]
  -d, --delete-swr-image  Delete the image from swr.
  -C, --config-file PATH  Configure file path for authorization.
  -D, --debug             Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
  -h, -H, --help          Show this message and exit.
```

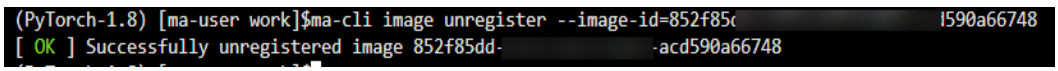
表 6-11 参数说明

参数名	参数类型	是否必选	参数说明
-i / -image-id	String	是	需要取消注册的镜像ID。
-d / --delete-swr-image	Bool	否	取消注册后同步删除SWR镜像开关，默认关闭。

示例

取消ModelArts镜像管理中已注册镜像。

```
ma-cli image unregister --image-id=xx
```



6.5.10 在 ECS 上调试 SWR 镜像是否能在 ModelArts Notebook 中使用

ma-cli支持在ECS上调试SWR镜像是否可以在ModelArts开发环境中运行，发现镜像中可能存在的问题。

```
ma-cli image debug -h
Usage: ma-cli image debug [OPTIONS]

  Debug SWR image as a Notebook in ECS.

  Example:

  # Debug cpu notebook image
  ma-cli image debug --swr-path=xx --service=NOTEBOOK --region=

  # Debug gpu notebook image
  ma-cli image debug --swr-path=xx --service=NOTEBOOK --region= --gpu

Options:
  -swr, --swr-path TEXT           SWR path without SWR endpoint, eg:organization/image:tag.  [required]
  -r, --region TEXT               Region name.  [required]
  -s, --service [NOTEBOOK|MODELBOX]
                                  Services supported by this image(default NOTEBOOK).
  -a, --arch [X86_64|AARCH64]     Image architecture(default X86_64).
  -g, --gpu                       Use all gpus to debug.
  -D, --debug                     Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT              CLI connection profile to use. The default profile is "DEFAULT".
  -h, -H, --help                  Show this message and exit.
```

表 6-12 参数说明

参数名	参数类型	是否必选	参数说明
-swr / --swr-path	String	是	需要调试的镜像的SWR路径。
-r / --region	String	是	需要调试的镜像所在的区域。
-s / --service	String	否	调试镜像的服务类型，NOTEBOOK或者MODELBOX，默认是NOTEBOOK。
-a / --arch	Sring	否	调试镜像的架构，X86_64或者AARCH64，默认是X86_64。
-g / --gpu	Bool	否	使用GPU进行调试开关，默认关闭。

6.6 使用 ma-cli ma-job 命令提交 ModelArts 训练作业

6.6.1 ma-cli ma-job 训练作业支持的命令

使用**ma-cli ma-job**命令可以提交训练作业，查询训练作业日志、事件、使用的AI引擎、资源规格及停止训练作业等。

```
$ ma-cli ma-job -h
Usage: ma-cli ma-job [OPTIONS] COMMAND [ARGS]...

  ModelArts job submission and query jod details.

Options:
  -h, -H, --help  Show this message and exit.

Commands:
  delete      Delete training job by job id.
  get-engine  Get job engines.
  get-event   Get job running event.
  get-flavor  Get job flavors.
  get-job     Get job details.
  get-log     Get job log details.
  get-pool    Get job engines.
  stop        Stop training job by job id.
  submit      Submit training job.
```

表 6-13 训练作业支持的命令

命令	命令详情
get-job	查询ModelArts训练作业列表及详情。
get-log	查询ModelArts训练作业运行日志。
get-engine	查询ModelArts训练AI引擎。
get-event	查询ModelArts训练作业事件。
get-flavor	查询ModelArts训练资源规格。
get-pool	查询ModelArts训练专属池。
stop	停止ModelArts训练作业。
submit	提交ModelArts训练作业。
delete	删除指定作业id的训练作业。

6.6.2 使用 ma-cli ma-job get-job 命令查询 ModelArts 训练作业

使用**ma-cli ma-job get-job**命令可以查看训练作业列表或某个作业详情。

```
$ ma-cli ma-job get-job -h
Usage: ma-cli ma-job get-job [OPTIONS]

  Get job details.

  Example:

  # Get train job details by job name
  ma-cli ma-job get-job -n ${job_name}

  # Get train job details by job id
```

```
ma-cli ma-job get-job -i ${job_id}

# Get train job list
ma-cli ma-job get-job --page-size 5 --page-num 1

Options:

-i, --job-id TEXT              Get training job details by job id.
-n, --job-name TEXT            Get training job details by job name.
-pn, --page-num INTEGER        Specify which page to query.  [x>=1]
-ps, --page-size INTEGER RANGE  The maximum number of results for this query.  [1<=x<=50]
-v, --verbose                  Show detailed information about training job details.
-C, --config-file TEXT         Configure file path for authorization.
-D, --debug                    Debug Mode. Shows full stack trace when error occurs.
-P, --profile TEXT             CLI connection profile to use. The default profile is "DEFAULT".
-h, -H, --help                 Show this message and exit.
```

表 6-14 参数说明

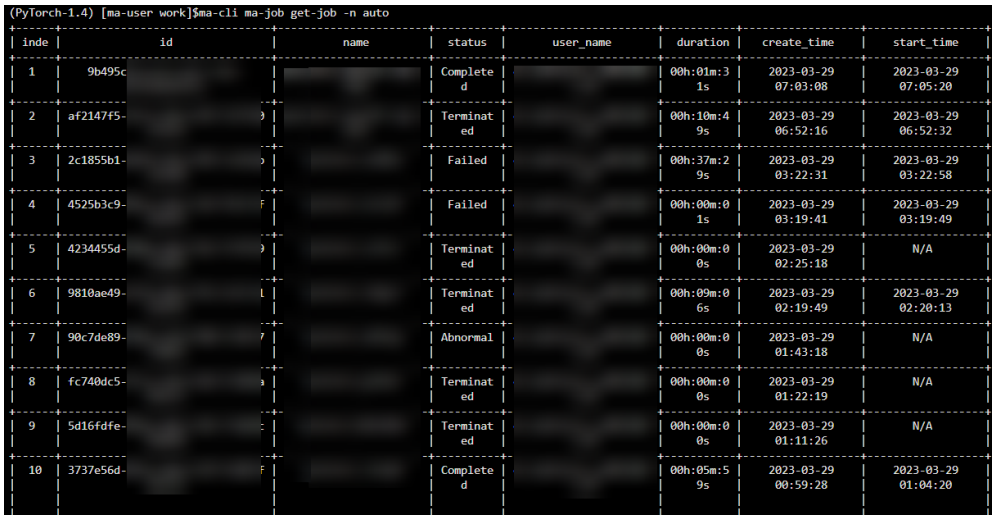
参数名	参数类型	是否必选	参数说明
-i / --job-id	String	否	查询指定训练任务ID的任务详情。
-n / --job-name	String	否	查询指定任务名称的训练任务或根据任务名称关键字过滤训练任务。
-pn / --page-num	Int	否	页面索引，默认是第1页。
-ps / --page-size	Int	否	每页显示的训练作业数量，默认是10。
-v / --verbose	Bool	否	显示详细的信息开关，默认关闭。

示例

- 查询指定任务ID的训练任务。
  ma-cli ma-job get-job -i *b63e90xxx*

```
(PyTorch-1.4) [ma-user work]$ma-cli ma-job get-job -i b63e90ba-91
+--------------+--------------------------------------+---------+---------------+----------+-------------+-------------+-----------+
|      id      |                 name                 | status  |   user_name   | duration |  create_time |  start_time | descripti |
+--------------+--------------------------------------+---------+---------------+----------+-------------+-------------+-----------+
| b63e90ba-    | workflow_created_job_ed3a963f-5438-4a99-9a19-c97ce | Complet | ei_modela     | 00h:01m:1 | 2023-03-29  | 2023-03-29  |           |
|              |                 88c48b8              |   ed    |               |    6s    |  03:41:21   |  03:41:30   |           |
+--------------+--------------------------------------+---------+---------------+----------+-------------+-------------+-----------+
```

- 根据任务名称关键字“auto”过滤训练任务。
  ma-cli ma-job get-job -n *auto*



6.6.3 使用 ma-cli ma-job submit 命令提交 ModelArts 训练作业

执行**ma-cli ma-job submit**命令提交ModelArts训练作业。

ma-cli ma-job submit命令需要指定一个位置参数YAML_FILE表示作业的配置文件路径，如果不指定该参数，则表示配置文件为空。配置文件是一个YAML格式的文件，里面的参数就是命令的option参数。此外，如果用户在命令行中同时指定YAML_FILE配置文件和option参数，命令行中指定的option参数的值将会覆盖配置文件相同的值。

```
$ma-cli ma-job submit -h
Usage: ma-cli ma-job submit [OPTIONS] [YAML_FILE]...

  Submit training job.

  Example:

  ma-cli ma-job submit --code-dir obs://your_bucket/code/
                       --boot-file main.py
                       --framework-type PyTorch
                       --working-dir /home/ma-user/modelarts/user-job-dir/code
                       --framework-version pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-x86_64
                       --data-url obs://your_bucket/dataset/
                       --log-url obs://your_bucket/logs/
                       --train-instance-type modelarts.vm.cpu.8u
                       --train-instance-count 1

Options:
  --name TEXT                   Job name.
  --description TEXT            Job description.
  --image-url TEXT              Full swr custom image path.
  --uid TEXT                    Uid for custom image (default: 1000).
  --working-dir TEXT            ModelArts training job working directory.
  --local-code-dir TEXT         ModelArts training job local code directory.
  --user-command TEXT           Execution command for custom image.
  --pool-id TEXT                Dedicated pool id.
  --train-instance-type TEXT    Train worker specification.
  --train-instance-count INTEGER  Number of workers.
  --data-url TEXT               OBS path for training data.
  --log-url TEXT                OBS path for training log.
  --code-dir TEXT               OBS path for source code.
  --output TEXT                 Training output parameter with OBS path.
  --input TEXT                  Training input parameter with OBS path.
  --env-variables TEXT          Env variables for training job.
  --parameters TEXT             Training job parameters (only keyword parameters are supported).
  --boot-file TEXT              Training job boot file path behinds `code_dir`.
```

```
--framework-type TEXT           Training job framework type.
--framework-version TEXT        Training job framework version.
--workspace-id TEXT             The workspace where you submit training job(default "0")
--policy [regular|economic|turbo|auto]
                                Training job policy, default is regular.
--volumes TEXT                  Information about the volumes attached to the training job.
-q, --quiet                     Exit without waiting after submit successfully.
-C, --config-file PATH          Configure file path for authorization.
-D, --debug                     Debug Mode. Shows full stack trace when error occurs.
-P, --profile TEXT              CLI connection profile to use. The default profile is "DEFAULT".
-H, -h, --help                  Show this message and exit.
```

表 6-15 参数说明

参数名	参数类型	是否必选	参数说明
YAML_FILE	String	否	表示训练作业的配置文件，如果不传则表示配置文件为空。
--code-dir	String	是	训练源代码的OBS路径。
--data-url	String	是	训练数据的OBS路径。
--log-url	String	是	存放训练生成日志的OBS路径。
--train-instance-count	String	是	训练作业计算节点个数，默认是1，表示单节点。
--boot-file	String	否	当使用自定义镜像或自定义命令时可以省略，当使用预置命令提交训练作业时需要指定该参数。
--name	String	否	训练任务名称。
--description	String	否	训练任务描述信息。
--image-url	String	否	自定义镜像SWR地址，遵循organization/image_name:tag
--uid	String	否	自定义镜像运行的UID，默认值1000。
--working-dir	String	否	运行算法时所在的工作目录。
--local-code-dir	String	否	算法的代码目录下载到训练容器内的本地路径。
--user-command	String	否	自定义镜像执行命令。需为/home下的目录。 当code-dir以file://为前缀时，当前字段不生效。

参数名	参数类型	是否必选	参数说明
--pool-id	String	否	训练作业选择的资源池ID。可在ModelArts管理控制台，单击左侧“专属资源池”，在专属资源池列表中查看资源池ID。
--train-instance-type	String	否	训练作业选择的资源规格。
--output	String	否	训练的输出信息，指定后，训练任务将会把训练脚本中指定输出参数对应训练容器的输出目录上传到指定的OBS路径。如果需要指定多个参数，可以使用--output output1=obs://bucket/output1 --output output2=obs://bucket/output2
--input	String	否	训练的输入信息，指定后，训练任务将会把对应OBS上的数据下载到训练容器，并将数据存储路径通过指定的参数传递给训练脚本。如果需要指定多个参数，可以使用--input data_path1=obs://bucket/data1 --input data_path2=obs://bucket/data2
--env-variables	String	否	训练时传入的环境变量，如果需要指定多个参数，可以使用--env-variables ENV1=env1 --env-variables ENV2=env2
--parameters	String	否	训练入参，可以通过--parameters "--epoch 0 --pretrained"指定多个参数。
--framework-type	String	否	训练作业选择的引擎规格。
--framework-version	String	否	训练作业选择的引擎版本。
-q / --quiet	Bool	否	提交训练任务成功后直接退出，不再同步打印作业状态。
--workspace-id	String	否	作业所处的工作空间，默认值为“0”。
--policy	String	否	训练资源规格模式，可选值regular、economic、turbo、auto。

参数名	参数类型	是否必选	参数说明
--volumes	String	否	挂载EFS，如果需要指定多个参数，可以使用--volumes。 "local_path=/xx/yy/zz;read_only=false;nfs_server_path=xxx.xxx.xxx.xxx:/" -volumes "local_path=/xxx/yyy/zzz;read_only=false;nfs_server_path=xxx.xxx.xxx.xxx:/"

基于 ModelArts 预置镜像提交训练作业

指定命令行options参数提交训练作业

```
ma-cli ma-job submit --code-dir obs://your-bucket/mnist/code/ \
                --boot-file main.py \
                --framework-type PyTorch \
                --working-dir /home/ma-user/modelarts/user-job-dir/code \
                --framework-version pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-x86_64 \
                --data-url obs://your-bucket/mnist/dataset/MNIST/ \
                --log-url obs://your-bucket/mnist/logs/ \
                --train-instance-type modelarts.vm.cpu.8u \
                --train-instance-count 1  \
                -q
```

使用预置镜像的train.yaml样例：

```
# .ma/train.yaml样例（预置镜像）
# pool_id: pool_xxxx
train-instance-type: modelarts.vm.cpu.8u
train-instance-count: 1
data-url: obs://your-bucket/mnist/dataset/MNIST/
code-dir: obs://your-bucket/mnist/code/
working-dir: /home/ma-user/modelarts/user-job-dir/code
framework-type: PyTorch
framework-version: pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-x86_64
boot-file: main.py
log-url: obs://your-bucket/mnist/logs/

##[Optional] Uncomment to set uid when use custom image mode
uid: 1000

##[Optional] Uncomment to upload output file/dir to OBS from training platform
output:
   - name: output_dir
     obs_path: obs://your-bucket/mnist/output1/

##[Optional] Uncomment to download input file/dir from OBS to training platform
input:
   - name: data_url
     obs_path: obs://your-bucket/mnist/dataset/MNIST/

##[Optional] Uncomment pass hyperparameters
parameters:
   - epoch: 10
   - learning_rate: 0.01
   - pretrained:
```

```
##[Optional] Uncomment to use dedicated pool
pool_id: pool_xxxx

##[Optional] Uncomment to use volumes attached to the training job
volumes:
  - efs:
      local_path: /xx/yy/zz
      read_only: false
      nfs_server_path: xxx.xxx.xxx.xxx:/
```

基于自定义镜像创建训练作业

指定命令行options参数提交训练作业

```
ma-cli ma-job submit --image-url atelier/pytorch_1_8:pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-
x86_64-20220926104358-041ba2e \
                --code-dir obs://your-bucket/mnist/code/ \
                --user-command "export LD_LIBRARY_PATH=/usr/local/cuda/compat:$LD_LIBRARY_PATH &&
cd /home/ma-user/modelarts/user-job-dir/code && /home/ma-user/anaconda3/envs/PyTorch-1.8/bin/
python main.py" \
                --data-url obs://your-bucket/mnist/dataset/MNIST/ \
                --log-url obs://your-bucket/mnist/logs/ \
                --train-instance-type modelarts.vm.cpu.8u \
                --train-instance-count 1  \
                -q
```

使用自定义镜像的train.yaml样例：

```
# .ma/train.yaml样例（自定义镜像）
image-url: atelier/pytorch_1_8:pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-
x86_64-20220926104358-041ba2e
user-command: export LD_LIBRARY_PATH=/usr/local/cuda/compat:$LD_LIBRARY_PATH && cd /home/ma-
user/modelarts/user-job-dir/code && /home/ma-user/anaconda3/envs/PyTorch-1.8/bin/python main.py
train-instance-type: modelarts.vm.cpu.8u
train-instance-count: 1
data-url: obs://your-bucket/mnist/dataset/MNIST/
code-dir: obs://your-bucket/mnist/code/
log-url: obs://your-bucket/mnist/logs/

##[Optional] Uncomment to set uid when use custom image mode
uid: 1000

##[Optional] Uncomment to upload output file/dir to OBS from training platform
output:
   - name: output_dir
     obs_path: obs://your-bucket/mnist/output1/

##[Optional] Uncomment to download input file/dir from OBS to training platform
input:
   - name: data_url
     obs_path: obs://your-bucket/mnist/dataset/MNIST/

##[Optional] Uncomment pass hyperparameters
parameters:
   - epoch: 10
   - learning_rate: 0.01
   - pretrained:

##[Optional] Uncomment to use dedicated pool
pool_id: pool_xxxx

##[Optional] Uncomment to use volumes attached to the training job
volumes:
  - efs:
      local_path: /xx/yy/zz
      read_only: false
      nfs_server_path: xxx.xxx.xxx.xxx:/
```

示例

- 基于yaml文件提交训练作业
  ma-cli ma-job submit ./train-job.yaml

```
(PyTorch-1.4) [ma-user work]$ma-cli ma-job submit ./train_job.yaml
[ OK ] Current training job id is: 4d7c8584-b213-4f88-9833-d3e6a82f9e42
[ OK ] Creating
[ OK ] Running
```

- 基于命令行和预置镜像pytorch1.8-cuda10.2-cudnn7-ubuntu18.04提交训练作业。

```
ma-cli ma-job submit --code-dir obs://automation-use-only/Original/TrainJob/TrainJob-v2/pytorch1.8.0_cuda10.2/code/ \
                     --boot-file test-pytorch.py \
                     --framework-type PyTorch \
                     --working-dir /home/ma-user/modelarts/user-job-dir/code \
                     --framework-version pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-x86_64 \
                     --data-url obs://automation-use-only/Original/TrainJob/TrainJob-v2/pytorch1.8.0_cuda10.2/data/ \
                     --log-url obs://automation-use-only/Original/TrainJob/TrainJob-v2/pytorch1.8.0_cuda10.2/data/logs/ \
                     --train-instance-type modelarts.vm.cpu.8u \
                     --train-instance-count 1 \
```

```
(PyTorch-1.4) [ma-user work]$ma-cli ma-job submit --code-dir obs://au                    /Original/TrainJob/T                 ?/pytorch1.8.0_cuda10.2/code/ \
>                       --boot-file  test-pytorch.py \
>                       --framework-type PyTorch \
>                       --working-dir /home/ma-user/modelarts/user-job-dir/code \
>                       --framework-version pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-x86_64 \
>                       --data-url obs://autom                    ly/Original/TrainJob/TrainJob-v2/pytorch1.8.0_cuda10.2/data/ \
>                       --log-url obs://automa                    /Original/TrainJob/TrainJob-v2/pytorch1.8.0_cuda10.2/data/logs/ \
>                       --train-instance-type modelarts.vm.cpu.8u \
>                       --train-instance-count 1 \
>
[ OK ] Current training job id is: 7db3e6f9-181d-4142-ba13-235213499430
[ OK ] Creating
[ OK ] Running
```

6.6.4 使用 ma-cli ma-job get-log 命令查询 ModelArts 训练作业日志

执行**ma-cli ma-job get-log**命令查询ModelArts训练作业日志。

```
$ ma-cli ma-job get-log -h
Usage: ma-cli ma-job get-log [OPTIONS]

  Get job log details.

  Example:

  # Get job log by job id
  ma-cli ma-job get-log --job-id ${job_id}

Options:
  -i, --job-id TEXT       Get training job details by job id.  [required]
  -t, --task-id TEXT      Get training job details by task id (default "worker-0").
  -C, --config-file TEXT  Configure file path for authorization.
  -D, --debug             Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
  -h, -H, --help          Show this message and exit.
```

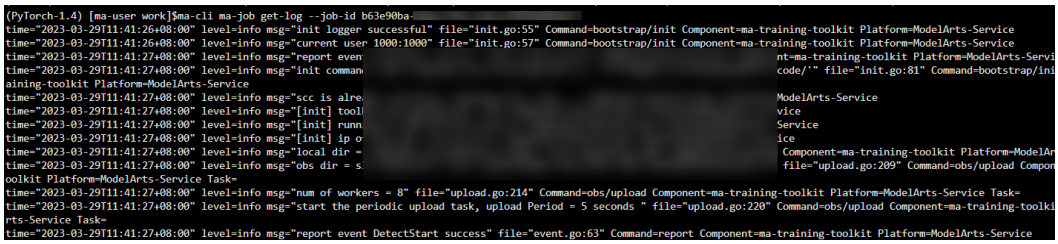
参数名	参数类型	是否必选	参数说明
-i / --job-id	String	是	查询指定训练任务ID的任务日志。

参数名	参数类型	是否必选	参数说明
-t / --task-id	String	否	查询指定task的日志，默认是work-0。

示例

查询指定训练任务ID的作业日志。

```
ma-cli ma-job get-log --job-id b63e90baxxx
```



6.6.5 使用 ma-cli ma-job get-event 命令查询 ModelArts 训练作业事件

执行**ma-cli ma-job get-event**命令查看ModelArts训练作业事件。

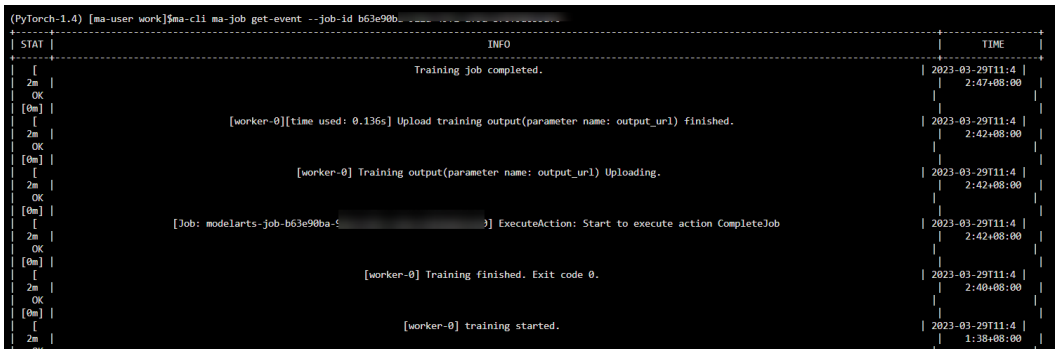
```
$ ma-cli ma-job get-event -h
Usage: ma-cli ma-job get-event [OPTIONS]

  Get job running event.

  Example:

  # Get training job running event
  ma-cli ma-job get-event --job-id ${job_id}

Options:
  -i, --job-id TEXT       Get training job event by job id.  [required]
  -C, --config-file TEXT  Configure file path for authorization.
  -D, --debug             Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help          Show this message and exit.
```

参数名	参数类型	是否必选	参数说明
-i / --job-id	String	是	查询指定训练任务ID的事件。

示例

查看指定ID的训练作业的事件详情等。

```
ma-cli ma-job get-event --job-id b63e90baxxx
```



6.6.6 使用 ma-cli ma-job get-engine 命令查询 ModelArts 训练 AI 引擎

执行**ma-cli ma-job get-engine**命令查询ModelArts训练使用的AI引擎。

```
$ ma-cli ma-job get-engine -h
Usage: ma-cli ma-job get-engine [OPTIONS]

  Get job engine info.

  Example:

  # Get training job engines
  ma-cli ma-job get-engine

Options:
  -v, --verbose           Show detailed information about training engines.
  -C, --config-file TEXT  Configure file path for authorization.
  -D, --debug             Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help          Show this message and exit.
```

表 6-16 参数说明

参数名	参数类型	是否必选	参数说明
-v / --verbose	Bool	否	显示详细的信息开关，默认关闭。

示例

查看训练作业的AI引擎。

```
ma-cli ma-job get-engine
```

```
(PyTorch-1.4) [ma-user work]$ma-cli ma-job get-engine
+-------+-----------------------------------------------------------+-----------------------+----------+
| index |                         engine id                         |      engine name      | run user |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   1   |                    caffe-1.0.0-python2.7                  |         Caffe         |          |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   2   |                    horovod-cp36-tf-1.16.2                 |        Horovod        |          |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   3   | horovod_0.20.0-pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-x86_64 |        Horovod        |   1102   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   4   | horovod_0.20.0-tensorflow_2.1.0-cuda_10.1-py_3.7-ubuntu_18.04-x86_64 |        Horovod        |   1102   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   5   |                kungfu-0.2.2-tf-1.13.1-python3.6           |         KungFu        |          |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   6   |        mindspore_1.3.0-cuda_10.1-py_3.7-ubuntu_1804-x86_64        |          MPI          |   1102   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   7   |       mindspore_1.7.0-cann_5.1.0-py_3.7-euler_2.8.3-aarch64       | Ascend-Powered-Engine |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   8   |       mindspore_1.8.0-cann_5.1.2-py_3.7-euler_2.8.3-aarch64       | Ascend-Powered-Engine |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   9   |       mindspore_1.9.0-cann_6.0.0-py_3.7-euler_2.8.3-aarch64       | Ascend-Powered-Engine |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   10  |                    mxnet-1.2.1-python3.6                  |         MXNet         |          |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   11  |    optverse_0.2.0-pygrassland_1.1.0-py_3.7-ubuntu_18.04-x86_64    |           OR          |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   12  |                      pytorch-cp36-1.0.0                   |        PyTorch        |          |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   13  |                      pytorch-cp36-1.3.0                   |        PyTorch        |          |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   14  |                      pytorch-cp36-1.4.0                   |        PyTorch        |          |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   15  |       pytorch_1.8.0-cann_5.1.0-py_3.7-euler_2.8.3-aarch64         | Ascend-Powered-Engine |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   16  |        pytorch_1.8.0-cuda_10.2-py_3.7-ubuntu_18.04-x86_64         |        PyTorch        |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   17  |       pytorch_1.8.1-cann_5.1.2-py_3.7-euler_2.8.3-aarch64         | Ascend-Powered-Engine |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   18  |       pytorch_1.8.1-cann_6.0.0-py_3.7-euler_2.8.3-aarch64         | Ascend-Powered-Engine |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   19  |        pytorch_1.8.1-cuda_11.1-py_3.7-ubuntu_18.04-x86_64         |        PyTorch        |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   20  |        pytorch_1.8.2-cuda_10.2-py_3.7-ubuntu_18.04-x86_64         |        PyTorch        |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   21  |        pytorch_1.9.1-cuda_11.1-py_3.7-ubuntu_18.04-x86_64         |        PyTorch        |   1000   |
+-------+-----------------------------------------------------------+-----------------------+----------+
|   22  |                       ray-cp36-0.7.4                      |          Ray          |          |
+-------+-----------------------------------------------------------+-----------------------+----------+
```

6.6.7 使用 ma-cli ma-job get-flavor 命令查询 ModelArts 训练资源规格

执行**ma-cli ma-job get-flavor**命令查询ModelArts训练的资源规格。

```
$ ma-cli ma-job get-flavor -h
Usage: ma-cli ma-job get-flavor [OPTIONS]

  Get job flavor info.

  Example:

  # Get training job flavors
  ma-cli ma-job get-flavor

Options:
  -t, --flavor-type [CPU|GPU|Ascend]
                                  Type of training job flavor.
  -v, --verbose                   Show detailed information about training flavors.
  -C, --config-file TEXT          Configure file path for authorization.
  -D, --debug                     Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT              CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help                  Show this message and exit.
```

表 6-17 参数说明

参数名	参数类型	是否必选	参数说明
-t / --flavor-type	String	否	资源规格类型，如果不指定默认返回所有的资源规格。
-v / --verbose	Bool	否	显示详细的信息开关，默认关闭。

示例

查看训练作业的资源规格及类型。

```
ma-cli ma-job get-flavor
```

```
(PyTorch-1.4) [ma-user work]$ma-cli ma-job get-flavor
+-------+-----------------------------+-----------------------------------+-------------+
| index |          flavor id          |            flavor name            | flavor type |
+-------+-----------------------------+-----------------------------------+-------------+
|   1   |    modelarts.kat1.8xlarge   | Computing NPU(8*Ascend) instance  |    Ascend   |
+-------+-----------------------------+-----------------------------------+-------------+
|   2   |    modelarts.kat1.xlarge    |  Computing NPU(Ascend) instance   |    Ascend   |
+-------+-----------------------------+-----------------------------------+-------------+
|   3   |     modelarts.vm.cpu.2u     |    Computing CPU(2U) instance     |     CPU     |
+-------+-----------------------------+-----------------------------------+-------------+
|   4   |     modelarts.vm.cpu.8u     |    Computing CPU(8U) instance     |     CPU     |
+-------+-----------------------------+-----------------------------------+-------------+
|   5   | modelarts.vm.cpu.8u16g.119  |    Computing CPU(8U) instance     |     CPU     |
+-------+-----------------------------+-----------------------------------+-------------+
|   6   |   modelarts.vm.v100.large   |   Computing GPU(V100) instance    |     GPU     |
+-------+-----------------------------+-----------------------------------+-------------+
|   7   | modelarts.vm.v100.large.free |   Computing GPU(V100) instance    |     GPU     |
+-------+-----------------------------+-----------------------------------+-------------+
```

6.6.8 使用 ma-cli ma-job stop 命令停止 ModelArts 训练作业

执行**ma-cli ma-job stop**命令，可停止指定作业id的训练作业。

```
$ ma-cli ma-job stop -h
Usage: ma-cli ma-job stop [OPTIONS]

  Stop training job by job id.

  Example:

  Stop training job by job id
  ma-cli ma-job stop --job-id ${job_id}

Options:
  -i, --job-id TEXT       Get training job event by job id.  [required]
```

```
-y, --yes               Confirm stop operation.
-C, --config-file TEXT  Configure file path for authorization.
-D, --debug             Debug Mode. Shows full stack trace when error occurs.
-P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
-H, -h, --help          Show this message and exit.
```

表 6-18 参数说明

参数名	参数类型	是否必选	参数说明
-i / --job-id	String	是	ModelArts训练作业ID。
-y / --yes	Bool	否	强制关闭指定训练作业。

示例

停止运行中的训练作业。

```
ma-cli ma-job stop --job-id efd3e2f8xxx
```

6.7 使用 ma-cli dli-job 命令提交 DLI Spark 作业

6.7.1 ma-cli dli-job 提交 DLI Spark 作业支持的命令

```
$ma-cli dli-job -h
Usage: ma-cli dli-job [OPTIONS] COMMAND [ARGS]...

  DLI spark job submission and query jod details.

Options:
  -h, -H, --help  Show this message and exit.

Commands:
  get-job       Get DLI spark job details.
  get-log       Get DLI spark log details.
  get-queue     Get DLI spark queues info.
  get-resource  Get DLI resources info.
  stop          Stop DLI spark job by job id.
  submit        Submit dli spark batch job.
  upload        Upload local file or OBS object to DLI resources.
```

表 6-19 提交 DLI Spark 作业命令总览

命令	命令详情
get-job	查询DLI Spark作业列表及详情。
get-log	查询DLI Spark运行日志。
get-queue	查询DLI 队列。
get-resource	查询DLI 分组资源。
stop	停止DLI Spark作业。

命令	命令详情
submit	提交DLI Spark作业。
upload	上传本地文件或OBS文件到DLI分组资源。

6.7.2 使用 ma-cli dli-job get-job 命令查询 DLI Spark 作业

执行**ma-cli dli-job get-job**查询DLI Spark作业列表或单个作业详情。

```
ma-cli dli-job get-job -h
Usage: ma-cli dli-job get-job [OPTIONS]

  Get DLI Spark details.

  Example:

  # Get DLI Spark job details by job name
  ma-cli dli-job get-job -n ${job_name}

  # Get DLI Spark job details by job id
  ma-cli dli-job get-job -i ${job_id}

  # Get DLI Spark job list
  ma-cli dli-job get-job --page-size 5 --page-num 1

Options:
  -i, --job-id TEXT              Get DLI Spark job details by job id.
  -n, --job-name TEXT            Get DLI Spark job details by job name.
  -pn, --page-num INTEGER RANGE   Specify which page to query.  [x>=1]
  -ps, --page-size INTEGER RANGE  The maximum number of results for this query.  [x>=1]
  -v, --verbose                  Show detailed information about DLI Spark job details.
  -C, --config-file PATH         Configure file path for authorization.
  -D, --debug                    Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT             CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help                 Show this message and exit.
```

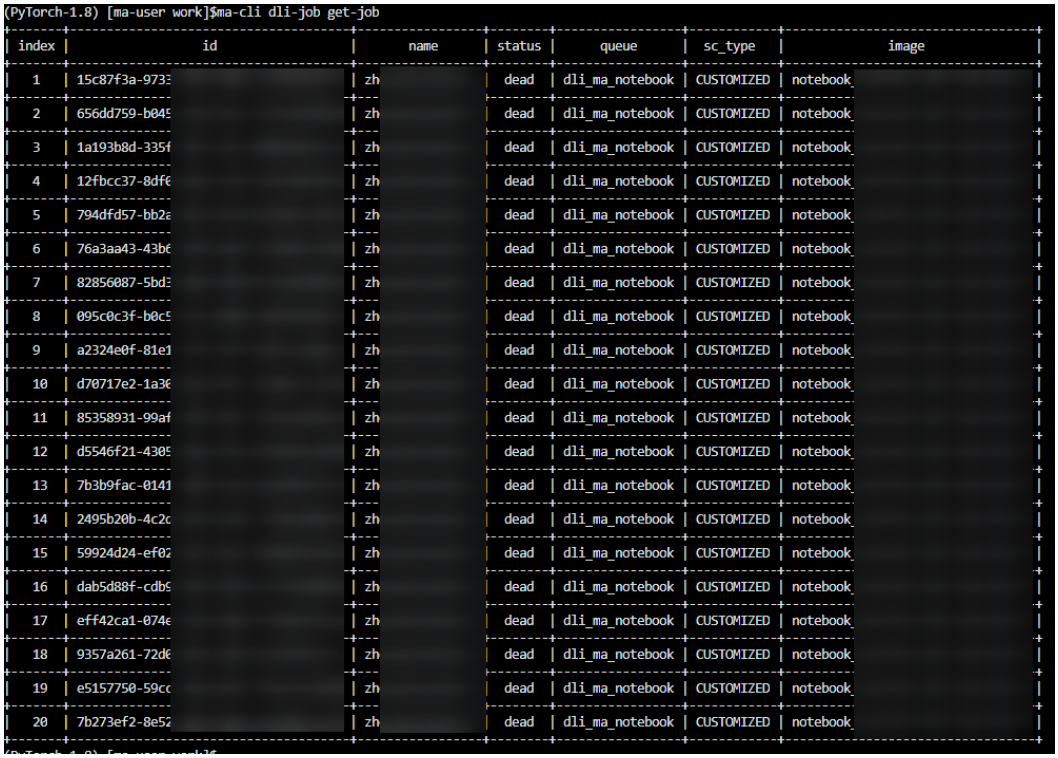
表 6-20 参数说明

参数名	参数类型	是否必选	参数说明
-i / --job-id	String	否	查询指定DLI Spark作业ID的任务详情。
-n / --job-name	String	否	查询指定作业名称的DLI Spark作业或根据作业名称关键字过滤DLI Spark作业。
-pn / --page-num	Int	否	作业索引页，默认是第1页。
-ps / --page-size	Int	否	每页显示的作业数量，默认是20。
-v / --verbose	Bool	否	显示详细的信息开关，默认关闭。

示例

查询DLI Spark所有作业。

```
ma-cli dli-job get-job
```



```
(PyTorch-1.8) [ma-user work]$ma-cli dli-job get-job
+-------+--------------------------------------+------+--------+-----------------+-----------+-----------------------------+
| index |                  id                  | name | status |      queue      |  sc_type  |            image            |
+-------+--------------------------------------+------+--------+-----------------+-----------+-----------------------------+
|   1   | 15c87f3a-9733-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   2   | 656dd759-b045-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   3   | 1a193b8d-335f-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   4   | 12fbcc37-8df6-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   5   | 794dfd57-bb2a-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   6   | 76a3aa43-43b6-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   7   | 82856087-5bd3-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   8   | 095c0c3f-b0c5-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   9   | a2324e0f-81e1-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   10  | d70717e2-1a36-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   11  | 85358931-99af-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   12  | d5546f21-4305-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   13  | 7b3b9fac-0141-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   14  | 2495b20b-4c2c-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   15  | 59924d24-ef02-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   16  | dab5d88f-cdb9-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   17  | eff42ca1-074e-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   18  | 9357a261-72d6-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   19  | e5157750-59cc-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
|   20  | 7b273ef2-8e52-                       | zh   |  dead  | dli_ma_notebook | CUSTOMIZED | notebook                   |
+-------+--------------------------------------+------+--------+-----------------+-----------+-----------------------------+
```

6.7.3 使用 ma-cli dli-job submit 命令提交 DLI Spark 作业

执行**ma-cli dli-job submit**命令提交DLI Spark作业。

ma-cli dli-job submit命令需要指定一个位置参数YAML_FILE表示作业的配置文件路径，如果不指定该参数，则表示配置文件为空。配置文件是一个YAML格式的文件，里面的参数就是命令的option参数。此外，如果用户在命令行中同时指定YAML_FILE配置文件和option参数，命令行中指定的option参数的值将会覆盖配置文件相同的值。

命令参数预览

```
ma-cli dli-job submit -h
Usage: ma-cli dli-job submit [OPTIONS] [YAML_FILE]...

  Submit DLI Spark job.

  Example:

  ma-cli dli-job submit  --name test-spark-from-sdk
                         --file test/sub_dli_task.py
                         --obs-bucket dli-bucket
                         --queue dli_test
                         --spark-version 2.4.5
                         --driver-cores 1
                         --driver-memory 1G
                         --executor-cores 1
                         --executor-memory 1G
                         --num-executors 1

Options:
  --file TEXT                 Python file or app jar.
  -cn, --class-name TEXT        Your application's main class (for Java / Scala apps).
  --name TEXT                 Job name.
  --image TEXT                Full swr custom image path.
  --queue TEXT                Execute queue name.
```

```
-obs, --obs-bucket TEXT        DLI obs bucket to save logs.
-sv, --spark-version TEXT      Spark version.
-st, --sc-type [A|B|C]         Compute resource type.
--feature [basic|custom|ai]    Type of the Spark image used by a job (default: basic).
-ec, --executor-cores INTEGER  Executor cores.
-em, --executor-memory TEXT    Executor memory (eg. 2G/2048MB).
-ne, --num-executors INTEGER   Executor number.
-dc, --driver-cores INTEGER    Driver cores.
-dm, --driver-memory TEXT      Driver memory (eg. 2G/2048MB).
--conf TEXT                    Arbitrary Spark configuration property (eg. <PROP=VALUE>).
--resources TEXT               Resources package path.
--files TEXT                   Files to be placed in the working directory of each executor.
--jars TEXT                    Jars to include on the driver and executor class paths.
-pf, --py-files TEXT           Python files to place on the PYTHONPATH for Python apps.
--groups TEXT                  User group resources.
--args TEXT                    Spark batch job parameter args.
-q, --quiet                    Exit without waiting after submit successfully.
-C, --config-file PATH         Configure file path for authorization.
-D, --debug                    Debug Mode. Shows full stack trace when error occurs.
-P, --profile TEXT             CLI connection profile to use. The default profile is "DEFAULT".
-H, -h, --help                 Show this message and exit.
```

yaml文件预览

```
# dli-demo.yaml
name: test-spark-from-sdk
file: test/sub_dli_task.py
obs-bucket: ${your_bucket}
queue: dli_notebook
spark-version: 2.4.5
driver-cores: 1
driver-memory: 1G
executor-cores: 1
executor-memory: 1G
num-executors: 1

## [Optional]
jars:
  - ./test.jar
  - obs://your-bucket/jars/test.jar
  - your_group/test.jar

## [Optional]
files:
  - ./test.csv
  - obs://your-bucket/files/test.csv
  - your_group/test.csv

## [Optional]
python-files:
  - ./test.py
  - obs://your-bucket/files/test.py
  - your_group/test.py

## [Optional]
resources:
  - name: your_group/test.py
    type: pyFile
  - name: your_group/test.csv
    type: file
  - name: your_group/test.jar
    type: jar
  - name: ./test.py
    type: pyFile
  - name: obs://your-bucket/files/test.py
    type: pyFile

## [Optional]
groups:
```

```
- group1
- group2
```

指定options参数提交DLI Spark作业示例：

```
$ ma-cli dli-job submit --name test-spark-from-sdk \
                        --file test/sub_dli_task.py \
                        --obs-bucket ${your_bucket} \
                        --queue dli_test \
                        --spark-version 2.4.5 \
                        --driver-cores 1 \
                        --driver-memory 1G \
                        --executor-cores 1 \
                        --executor-memory 1G \
                        --num-executors 1
```

表 6-21 参数说明

参数名	参数类型	是否必选	参数说明
YAML_FILE	String，本地文件路径	否	DLI Spark作业的配置文件，如果不传则表示配置文件为空。
--file	String	是	程序运行入口文件，支持本地文件路径、OBS路径或者用户已上传到DLI资源管理系统的类型为jar或pyFile的程序包名。
-cn / --class_name	String	是	批处理作业的Java/Spark主类。
--name	String	否	创建时用户指定的作业名称，不能超过128个字符。
--image	String	否	自定义镜像路径,格式为：组织名/镜像名:镜像版本。当用户设置“**feature**”为“custom”时，该参数生效。用户可通过与“**feature**”参数配合使用，指定作业运行使用自定义的Spark镜像。
-obs / --obs-bucket	String	否	保存Spark作业的obs桶，需要保存作业时配置该参数。同时也可作为提交本地文件到resource的中转站。
-sv/ --spark-version	String	否	作业使用Spark组件的版本号。
-st / `--sc-type	String	否	如果当前Spark组件版本为2.3.2，则不填写该参数。 如果当前Spark组件版本为2.3.3，则在“**feature**”为“basic”或“ai”时填写。如果不填写，则使用默认的Spark组件版本号2.3.2。

参数名	参数类型	是否必选	参数说明
--feature	String	否	作业特性。表示用户作业使用的Spark镜像类型，默认值为basic。 ● basic：表示使用DLI提供的基础Spark镜像。 ● custom：表示使用用户自定义的Spark镜像。 ● ai：表示使用DLI提供的AI镜像。
--queue	String	否	用于指定队列，填写已创建DLI的队列名。必须为通用类型的队列。队列名称的获取请参考**表6-23**。
-ec / --executor-cores	String	否	Spark应用每个Executor的CPU核数。该配置项会替换sc_type中对应的默认参数。
-em / --executor-memory	String	否	Spark应用的Executor内存，参数配置例如2G, 2048M。该配置项会替换“**sc_type**”中对应的默认参数，使用时必须带单位，否则会启动失败。
-ne / --num-executors	String	否	Spark应用Executor的个数。该配置项会替换sc_type中对应的默认参数。
-dc / --driver-cores	String	否	Spark应用Driver的CPU核数。该配置项会替换sc_type中对应的默认参数。
-dm / --driver-memory	String	否	Spark应用的Driver内存，参数配置例如2G, 2048M。该配置项会替换“**sc_type**”中对应的默认参数，使用时必须带单位，否则会启动失败。
--conf	Array of String	否	batch配置项，参考**Spark Configuration**。如果需要指定多个参数，可以使用--conf conf1 --conf conf2。
--resources	Array of String	否	资源包名称。支持本地文件，OBS路径及用户已上传到DLI资源管理系统的文件。如果需要指定多个参数，可以使用--resources resource1 --resources resource2。
--files	Array of String	否	用户已上传到DLI资源管理系统的类型为file的资源包名。也支持指定OBS路径，例如：obs://桶名/包名。同时也支持本地文件。如果需要指定多个参数，可以使用--files file1 --files file2。
--jars	Array of String	否	用户已上传到DLI资源管理系统的类型为jar的程序包名。也支持指定OBS路径，例如：obs://桶名/包名。也支持本地文件。如果需要指定多个参数，可以使用--jars jar1 --jars jar2。

参数名	参数类型	是否必选	参数说明
-pf /--python-files	Array of String	否	用户已上传到DLI资源管理系统的类型为pyFile的资源包名。也支持指定OBS路径，例如：obs://桶名/包名。也支持本地文件。如果需要指定多个参数，可以使用--python-files py1 --python-files py2。
--groups	Array of String	否	资源分组名称，如果需要指定多个参数，可以使用--groups group1 --groups group2。
--args	Array of String	否	传入主类的参数，即应用程序参数。如果需要指定多个参数，可以使用--args arg1 --args arg2。
-q / --quiet	Bool	否	提交DLI Spark作业成功后直接退出，不再同步打印任务状态。

示例

- 通过YAML_FILE文件提交DLI Spark作业。

```
$ma-cli dli-job submit dli_job.yaml
```

```
(PyTorch-1.4) [ma-user work]$ma-cli dli-job submit ./dli-job.yaml
[ OK ] Current DLI job id is: 01b698b8-9fd6-4a8e-bc3c-6821c6405b14
[ OK ] starting
[ OK ] running
[ OK ] success
[ OK ] Successfully submit DLI spark job [ 01b698b8-9fd6-4a8e-bc3c-6821c6405b14 ].
```

- 指定命令行options参数提交DLI Spark作业。

```
$ma-cli dli-job submit --name test-spark-from-sdk \
>                      --file test/jumpstart-trainingjob-gallery-pytorch-sample.ipynb \
>                      --queue dli_ma_notebook \
>                      --spark-version 2.4.5 \
>                      --driver-cores 1 \
>                      --driver-memory 1G \
>                      --executor-cores 1 \
>                      --executor-memory 1G \
>                      --num-executors 1
```

```
(PyTorch-1.4) [ma-user work]$ma-cli dli-job submit --name test-spark-from-sdk \
>                      --file test/jumpstart-trainingjob-gallery-pytorch-sample.ipynb \
>                      --queue dli_ma_notebook \
>                      --spark-version 2.4.5 \
>                      --driver-cores 1 \
>                      --driver-memory 1G \
>                      --executor-cores 1 \
>                      --executor-memory 1G \
>                      --num-executors 1
[ OK ] Current DLI job id is: ae856c20-e9ae-49ca-8409-7a02652297b8
[ OK ] starting
```

6.7.4 使用 ma-cli dli-job get-log 命令查询 DLI Spark 运行日志

执行**ma-cli dli-job get-log**命令查询DLI Spark作业后台的日志。

```
$ ma-cli dli-job get-log -h
Usage: ma-cli dli-job get-log [OPTIONS]
```

```
Get DLI spark job log details.

Example:

# Get job log by job id
ma-cli dli-job get-log --job-id ${job_id}

Options:
  -i, --job-id TEXT       Get DLI spark job details by job id.  [required]
  -C, --config-file TEXT  Configure file path for authorization.
  -D, --debug             Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help          Show this message and exit.
```

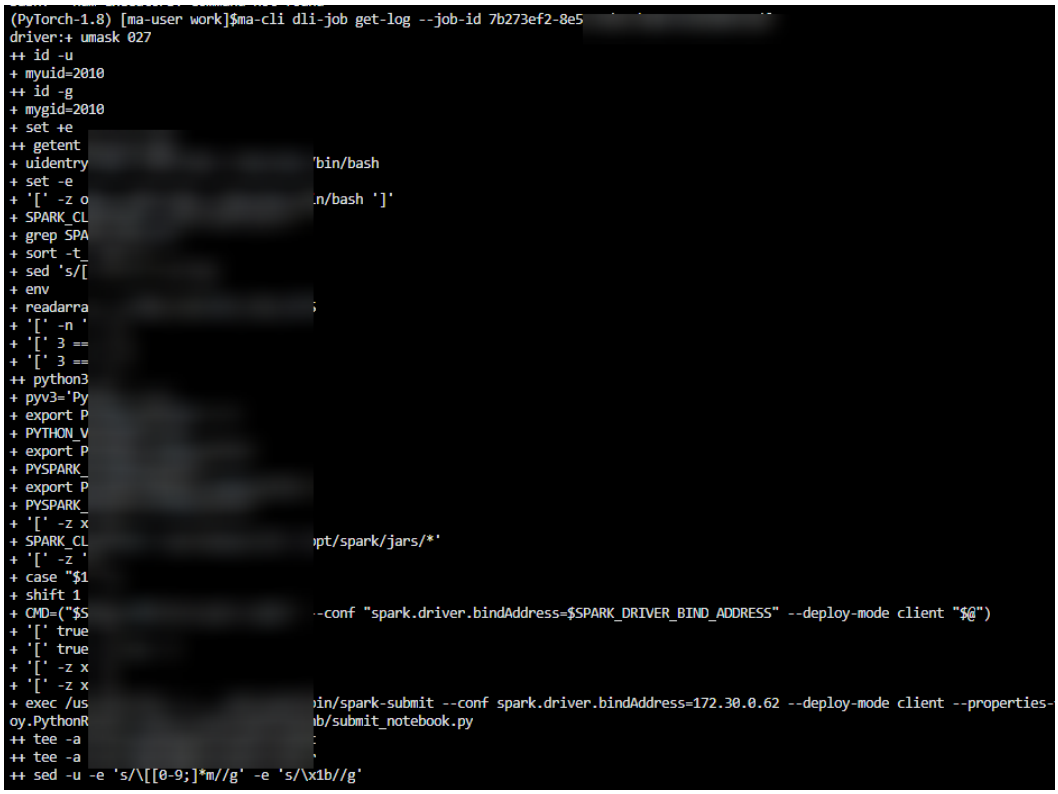
表 6-22 参数说明

参数名	参数类型	是否必选	参数说明
-i / --job-id	String	是	查询指定DLI Spark作业ID的任务日志。

示例

查询指定作业ID的DLI Spark作业运行日志。

```
ma-cli dli-job get-log --job-id ${your_job_id}
```



6.7.5 使用 ma-cli dli-job get-queue 命令查询 DLI 队列

执行**ma-cli dli-job get-queue**命令查询DLI队列。

```
ma-cli dli-job get-queue -h
Usage: ma-cli dli-job get-queue [OPTIONS]

  Get DLI queues info.

  Example:

  # Get DLI queue details by queue name
  ma-cli dli-job get-queue --queue-name $queue_name}

Options:
  -pn, --page-num INTEGER RANGE   Specify which page to query.  [x>=1]
  -ps, --page-size INTEGER RANGE  The maximum number of results for this query.  [x>=1]
  -n, --queue-name TEXT           Get DLI queue details by queue name.
  -t, --queue-type [sql|general|all]
                                  DLI queue type (default "all").
  -tags, --tags TEXT              Get DLI queues by tags.
  -C, --config-file PATH          Configure file path for authorization.
  -D, --debug                     Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT              CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help                  Show this message and exit.
```

表 6-23 参数说明

参数名	参数类型	是否必选	参数说明
-n / --queue-name	String	否	指定需要查询的DLI队列名称。
-t / --queue-type	String	否	指定查询的DLI队列类型，支持sql、general和all，默认是all。
-tags / --tags	String	否	指定查询的DLI队列tags。
-pn / --page-num	Int	否	DLI队列页索引，默认是第1页。
-ps / --page-size	Int	否	每页显示的DLI队列数量，默认是20。

示例

查询队列名为“dli_ma_notebook”的队列信息。

```
ma-cli dli-job get-queue --queue-name dli_ma_notebook
```

```
(PyTorch-1.8) [ma-user work]$ ma-cli dli-job get-queue --queue-name dli_ma_notebook
{'chargingMode': 1,
 'create_time': 1668585417422,
 'cuCount': 16,
 'cu_spec': 16,
 'description': '',
 'enterprise_project_id': '0',
 'is_success': True,
 'message': '',
 'owner': '[redacted]',
 'queueName': 'dli_ma_notebook',
 'queueType': 'general',
 'queue_id': 8[redacted],
 'resource_id': '242e7af8-c9d1[redacted]',
 'resource_mode': 1,
 'resource_type': 'container',
 'support_spark_versions': ['2.3.2', '2.4.5', '3.1.1']}
```

6.7.6 使用 ma-cli dli-job get-resource 命令查询 DLI 分组资源

执行**ma-cli dli-job get-resource**命令获取DLI资源详细信息，如资源名称，资源类型等。

```
$ ma-cli dli-job get-resource -h
Usage: ma-cli dli-job get-resource [OPTIONS]

  Get DLI resource info.

  Example:

  # Get DLI resource details by resource name
  ma-cli dli-job get-resource --resource-name ${resource_name}

Options:
  -n, --resource-name TEXT        Get DLI resource details by resource name.
  -k, --kind [jar|pyFile|file|modelFile]
                                  DLI resources type.
  -g, --group TEXT                Get DLI resources by group.
  -tags, --tags TEXT              Get DLI resources by tags.
  -C, --config-file TEXT          Configure file path for authorization.
  -D, --debug                     Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT              CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help                  Show this message and exit.
```

表 6-24 参数说明

参数名	参数类型	是否必选	参数说明
-n / --resource-name	String	否	按DLI分组资源名称查询DLI资源详细信息。
-k / --kind	String	否	按DLI分组资源类型查询DLI资源详细信息，支持jar、pyFile、file和modelFile。
-g / --group	String	否	按DLI分组资源组名查询DLI资源组详细信息。
-tags / --tags	String	否	通过DLI分组资源tags获取DLI资源详细信息。

示例

查询所有DLI分组资源信息。

```
ma-cli dli-job get-resource
```

6.7.7 使用 ma-cli dli-job upload 命令上传文件到 DLI 分组资源

ma-cli dli-job upload命令支持将本地文件或OBS文件上传到DLI资源组。

```
$ ma-cli dli-job upload -h
Usage: ma-cli dli-job upload [OPTIONS] PATHS...

  Upload DLI resource.

  Tips: --obs-path is need when upload local file.

  Example:

  # Upload an OBS path to DLI resource
  ma-cli dli-job upload obs://your-bucket/test.py -g test-group --kind pyFile

  # Upload a local path to DLI resource
  ma-cli dli-job upload ./test.py -g test-group -obs ${your-bucket} --kind pyFile

  # Upload local path and OBS path to DLI resource
  ma-cli dli-job upload ./test.py obs://your-bucket/test.py -g test-group -obs ${your-bucket}
```

```
Options:
  -k, --kind [jar|pyFile|file]  DLI resources type.
  -g, --group TEXT              DLI resources group.
  -tags, --tags TEXT            DLI resources tags, follow --tags `key1`=`value1`.
  -obs, --obs-bucket TEXT       OBS bucket for upload local file.
  -async, --is-async            whether to upload resource packages in asynchronous mode. The default value is
False.
  -C, --config-file TEXT        Configure file path for authorization.
  -D, --debug                   Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT            CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help                Show this message and exit.
```

表 6-25 参数说明

参数名	参数类型	是否必选	参数说明
PATHS	String	是	需要上传到DLI分组资源的本地文件路径或者obs路径，支持同时传入多个路径。
-k / --kind	String	否	上传文件的类型，支持jar、pyFile和file。
-g / --group	String	否	上传文件的DLI分组名。
-tags / --tags	String	否	上传文件的tag。
-obs / --obs-bucket	String	否	如果上传文件包含本地路径，则需要指定一个OBS桶作为中转。
-async / --is-async	Bool	否	异步上传文件，推荐使用。

示例

- 上传本地文件到DLI分组资源
  ma-cli dli-job upload ./test.py -obs ${your-bucket} --kind pyFile

  ```
  (PyTorch-1.8) [ma-user work]$ma-cli dli-job upload ./test.py -obs obs://[redacted] --kind pyFile
  [ OK ] Upload ['test.py']  successfully.
  ```

- 上传OBS文件到DLI分组资源
  ma-cli dli-job upload obs://your-bucket/test.py --kind pyFile

  ```
  (PyTorch-1.8) [ma-user work]$ma-cli dli-job upload obs://[redacted]/test.py --kind pyFile
  [ OK ] Upload ['test.py']  successfully.
  ```

6.7.8 使用 ma-cli dli-job stop 命令停止 DLI Spark 作业

执行**ma-cli dli-job stop**命令停止DLI Spark作业。

```
$ ma-cli dli-job stop -h
Usage: ma-cli dli-job stop [OPTIONS]

  Stop DLI spark job by job id.

  Example:

  Stop training job by job id
  ma-cli dli-job stop --job-id ${job_id}

Options:
  -i, --job-id TEXT       Get DLI spark job event by job id.  [required]
```

```
-y, --yes               Confirm stop operation.
-C, --config-file TEXT  Configure file path for authorization.
-D, --debug             Debug Mode. Shows full stack trace when error occurs.
-P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
-H, -h, --help          Show this message and exit.
```

表 6-26 参数说明

参数名	参数类型	是否必选	参数说明
-i / --job-id	String	是	DLI Spark作业ID。
-y / --yes	Bool	否	强制关闭指定DLI Spark作业。

示例

```
ma-cli dli-job stop -i ${your_job_id}
```

6.8 使用 ma-cli obs-copy 命令复制 OBS 数据

使用**ma-cli obs-copy [SRC] [DST]**可以实现本地和OBS文件或文件夹的相互复制。

```
$ma-cli obs-copy -h
Usage: ma-cli obs-copy [OPTIONS     ] SRC DST

  Copy file or directory to between OBS and local path. Example:

  # Upload local file to OBS path
  ma-cli obs-copy ./test.zip obs://your-bucket/copy-data/

  # Upload local directory to OBS path
  ma-cli obs-copy ./test/ obs://your-bucket/copy-data/

  # Download OBS file to local path
  ma-cli obs-copy obs://your-bucket/copy-data/test.zip ./test.zip

  # Download OBS directory to local path
  ma-cli obs-copy obs://your-bucket/copy-data/ ./test/

Options:
  -d, --drop-last-dir     Whether to drop last directory when copy folder. if True, the last directory of the
source folder will not copy to the destination folder.  [default: False]
  -C, --config-file PATH  Configure file path for authorization.
  -D, --debug             Debug Mode. Shows full stack trace when error occurs.
  -P, --profile TEXT      CLI connection profile to use. The default profile is "DEFAULT".
  -H, -h, --help          Show this message and exit.
```

表 6-27 参数说明

参数名	参数类型	是否必选	参数说明
-d / --drop-last-dir	Bool	否	如果指定，在复制文件夹时不会将源文件夹最后一级目录复制至目的文件夹下，仅对文件夹复制有效。

命令示例

上传文件到OBS中

```
$ ma-cli obs-copy ./test.csv obs://${your_bucket}/test-copy/
[ OK ] local src path: [ /home/ma-user/work/test.csv ]
[ OK ] obs dst path: [ obs://${your_bucket}/test-copy/ ]
```

上传文件夹到OBS中，对应上传到OBS的目录为obs://${your_bucket}/test-copy/data/

```
$ ma-cli obs-copy /home/ma-user/work/data/ obs://${your_bucket}/test-copy/
[ OK ] local src path: [ /home/ma-user/work/data/ ]
[ OK ] obs dst path: [ obs://${your_bucket}/test-copy/ ]
```

上传文件夹到OBS中，并指定--drop-last-dir，对应上传到OBS的目录为obs://${your_bucket}/test-copy/

```
$ ma-cli obs-copy /home/ma-user/work/data/ obs://${your_bucket}/test-copy/ --drop-last-dir
[ OK ] local src path: [ /home/ma-user/work/data ]
[ OK ] obs dst path: [ obs://${your_bucket}/test-copy/ ]
```

从OBS下载文件夹到本地磁盘中

```
$ ma-cli obs-copy obs://${your_bucket}/test-copy/ ~/work/test-data/
[ OK ] obs src path: [ obs://${your_bucket}/test-copy/ ]
[ OK ] local dst path: [ /home/ma-user/work/test-data/ ]
```