ModelArts

故障排除

文档版本 01

发布日期 2025-01-09

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 通用问题

1.1 ModelArts 中提示 OBS 相关错误

问题现象

- 在ModelArts中引用OBS桶路径时，提示找不到用户创建的OBS桶或提示ModelArts.2791：非法的OBS路径。
- 在对OBS桶操作时，出现Error: stat:403错误。
- Notebook中下载OBS文件时提示Permission denied。

原因分析

- OBS桶与ModelArts不在同一个区域导致。
- 没有他人OBS桶的访问权限。
- ModelArts上没有配置委托授权。
- OBS文件加密上传导致。ModelArts不支持OBS加密文件。
- OBS桶的权限和访问ACL设置不正确导致。
- 创建训练作业时，代码目录和启动文件设置有误。

处理办法

查看OBS桶与ModelArts是否在同一个区域

1. 查看创建的OBS桶所在区域。
   a. 登录**OBS管理控制台**。
   b. 进入“对象存储”界面，可在搜索框中输入已经创建的桶名称或者桶名称列表栏，找到您创建的OBS桶。
      在“区域栏”可查看创建的OBS桶的所在区域。
2. 查看ModelArts所在区域。
   登录ModelArts控制台，在控制台左上角可查看ModelArts所在区域。
3. 比对您创建的OBS桶所在区域与ModelArts所在区域是否一致。务必保证OBS桶与ModelArts所在区域一致。

检查您的账号是否有该OBS桶的访问权限

如果在使用Notebook时，需要访问其他账号的OBS桶，请查看您的账号是否有该OBS桶的访问权限。

检查委托授权

请前往权限管理，查看是否具有OBS访问授权。如果没有，请参考**配置访问授权（全局配置）**。

检查OBS桶是否为非加密桶

1. 进入OBS管理控制台，单击桶名称进入概览页。
2. 确保此OBS桶的加密功能关闭。如果此OBS桶为加密桶，可单击“默认加密”选项进行修改。

说明

创建OBS桶时，桶的存储类别请勿选择“归档存储”和“深度归档存储”，归档存储的OBS桶会导致模型训练失败。

图 1-1 查看 OBS 桶是否加密

检查OBS文件是否为加密文件

1. 进入OBS管理控制台，单击桶名称进入概览页。
2. 单击左侧菜单栏对象，进入对象列表。单击存放文件的对象名称，并找到具体的文件，可在文件列表的“加密状态”列查看文件是否加密。文件加密无法取消，请先解除桶加密，重新上传图片或文件。

检查OBS桶的ACLs设置

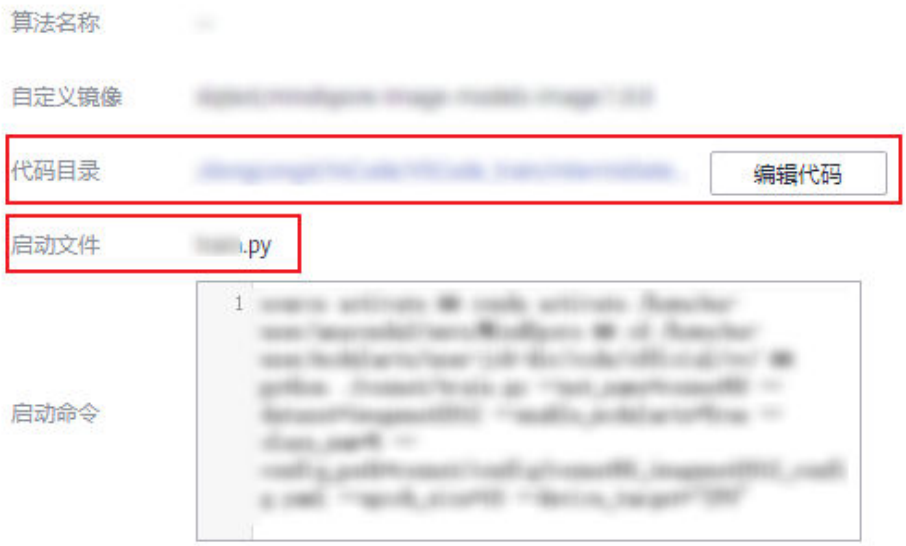
1. 进入OBS管理控制台，查找对应的OBS桶，单击桶名称进入概览页。
2. 在左侧菜单栏选择“访问权限控制>桶ACLs”，检查当前账号是否具备读写权限，如果没有权限，请联系桶的拥有者配置权限。
3. 在左侧菜单栏选择“访问权限控制>桶策略”，检查当前OBS桶是否允许子用户访问。

检查训练作业的代码目录和启动文件地址

1. 进入ModelArts管理控制台，在“作业管理 > 训练作业”中查找到对应的“运行失败”的训练作业，单击作业“名称/ID”进入详情页。
2. 在详情页左侧栏中，查看代码目录和启动文件选择是否正确，且OBS文件名称中不能有空格。
   - 代码目录：需要选择到OBS目录。如果选择了文件，会提示非法的OBS路径。

- 启动文件：需要选择以“.py”结尾的文件。如果选择的文件不是以“.py”结尾，会提示非法的OBS路径。

图 1-2 查看训练作业的代码目录和启动文件



如果还不能解决问题，请参考案例**已配置OBS权限，仍然无法访问OBS（403 AccessDenied）**进行进一步排查。

2 自动学习

2.1 准备数据

2.1.1 数据集版本发布失败

出现此问题时，表示数据不满足数据管理模块的要求，导致数据集发布失败，无法执行自动学习的下一步流程。

请根据如下几个要求，检查您的数据，将不符合要求的数据排除后再重新启动自动学习的训练任务。

ModelArts.4710 OBS 权限问题

ModelArts在跟OBS交互时，由于权限相关的问题导致。当界面提示“OBS service Error Message”信息时，表示是由于OBS权限导致的问题，请参考如下步骤排除故障。如果界面错误提示不包含此信息，则是因为后台服务故障导致，建议**联系华为云技术支持**。

1. 检查当前账号是否具备OBS权限。

   如果当前账号是个IAM用户（即子账号），需确认当前账号是否具备OBS服务操作权限。

   请参考**OBS权限管理**，为当前IAM用户配置“作用范围”为“全局级服务”的“Tenant Administrator”策略，即拥有OBS服务所有操作权限。

   如果需要限制此IAM用户操作，仅为此用户配置OBS相关的最小化权限项，具体操作请参见**创建ModelArts自定义策略**。

2. 检查OBS桶是否具备权限。

   说明

   下方步骤描述中所指的OBS桶，指创建自动学习项目时，指定的OBS桶，或者是创建项目时选择的数据集，其数据存储所在的OBS桶。

   - 检查当前账号具备OBS桶的读写权限（桶ACLs）
     - 进入OBS管理控制台，选择当前自动学习项目使用的OBS桶，单击桶名称进入概览页。

- 在左侧菜单栏选择“访问权限控制>桶ACL”，检查当前账号是否具备读写权限，如果没有权限，请联系桶的拥有者配置权限。

– 确保此OBS桶是非加密桶

i. 进入OBS管理控制台，选择当前自动学习项目使用的OBS桶，单击桶名称进入概览页。

ii. 确保此OBS桶的加密功能关闭。如果此OBS桶为加密桶，可单击“默认加密”选项进行修改。

图 2-1 OBS 桶是否加密

– 确保归档数据直读功能关闭

i. 进入OBS管理控制台，选择当前自动学习项目使用的OBS桶，单击桶名称进入概览页。

ii. 确保此OBS桶的归档数据直读功能关闭。如果此功能开启，可单击“归档数据直读”选项进行修改。

图 2-2 关闭归档数据直读功能

ModelArts.4711 数据集标注样本数满足算法要求

每个类别至少包含5张以上图片。

ModelArts.4342 标注信息不满足切分条件

出现此故障时，建议根据如下建议，修改标注数据后重试。

- 多标签的样本（即一张图片包含多个标签），至少需要有2张。如果启动训练时，设置了数据集切分功能，如果多标签的数据少于2张，会导致数据集切分失败。建议检查您的标注信息，保证标注多标签的图片，超过2张。
- 数据集切分后，训练集和验证集包含的标签类别不一样。出现这种情况的原因：多标签场景下时，做随机数据切分后，包含某一类标签的样本均被划分到训练集，导致验证集无该标签样本。由于这种情况出现的概率比较小，可尝试重新发布版本来解决。

ModelArts.4371 数据集版本已存在

出现此错误码时，表示数据集版本已存在，请重新发布数据集版本。

ModelArts.4712 数据集正在执行导入或同步等其他任务

如果自动学习中使用的数据集，正在执行导入或同步数据的任务时，此时进行训练将出现此错误。建议等待其他任务完成后，再启动自动学习的训练任务。

2.1.2 数据集版本不合格

出现此问题时，表示数据集版本发布成功，但是不满足自动学习训练作业要求，因此出现数据集版本不合格的错误提示。

标注信息不满足训练要求

针对不同类型的自动学习项目，训练作业对数据集的要求如下。

- 图像分类：用于训练的图片，至少有2种以上的分类（即2种以上的标签），每种分类的图片数不少于5张。
- 物体检测：用于训练的图片，至少有1种以上的分类（即1种以上的标签），每种分类的图片数不少于5张。
- 预测分析：由于预测分析任务的数据集不在数据管理中进行统一管理，即使数据不满足要求，不在此环节出现故障信息。
- 声音分类：用于训练的音频，至少有2种以上的分类（即2种以上的标签），每种分类的音频数不少于5个。
- 文本分类：用于训练的文本，至少有2种以上的分类（即2种以上的标签），每种分类的文本数不少于20个。

2.2 模型训练

2.2.1 自动学习训练作业失败

自动学习训练作业创建失败，一般是因为后台服务故障导致的，建议稍等片刻，然后重新创建训练作业。如果重试超过3次仍无法解决，请**联系华为云技术支持**。

自动学习训练作业创建成功，但是在运行过程中，由于一些故障导致作业运行失败，排查方式如下：

首次出现请检查您的账户是否欠费。如果账号状态正常。请针对不同类型的作业进行排查。

- 针对**图像分类**、**声音分类**、**文本分类**的作业，排查思路请参见**确保OBS中的数据存在**、**检查OBS的访问权限**、**检查图片是否符合要求**。
- 针对**物体检测**作业，排查思路请参见**确保OBS中的数据存在**、**检查OBS的访问权限**、**检查图片是否符合要求**、**检查标注框是否符合要求（物体检测）**。
- 针对**预测分析**作业，排查思路请参见**确保OBS中的数据存在**、**检查OBS的访问权限**、**预测分析作业失败的排查思路**。

确保 OBS 中的数据存在

如果存储在OBS中的图片或数据被删除，且未同步至ModelArts自动学习或数据集中，则会导致任务失败。

建议前往OBS检查，确保数据存在。针对图像分类、声音分类、文本分类、物体检测等类型，可在自动学习的数据标注页面，单击“同步数据源”，将OBS中的数据重新同步至ModelArts中。

检查 OBS 的访问权限

如果OBS桶的访问权限设置无法满足训练要求时，将会出现训练失败。请排查如下几个OBS的权限设置。

- 当前账号具备OBS桶的读写权限（桶ACLs）
  a. 进入OBS管理控制台，选择当前自动学习项目使用的OBS桶，单击桶名称进入概览页。
  b. 在左侧菜单栏选择“访问权限控制>桶ACLs”，检查当前账号是否具备读写权限，如果没有权限，请联系桶的拥有者配置权限。
- 确保此OBS桶是非加密桶
  a. 进入OBS管理控制台，选择当前自动学习项目使用的OBS桶，单击桶名称进入概览页。
  b. 确保此OBS桶的加密功能关闭。如果此OBS桶为加密桶，可单击“默认加密”选项进行修改。

图 2-3 OBS 桶是否加密

- 确保归档数据直读功能关闭
  a. 进入OBS管理控制台，选择当前自动学习项目使用的OBS桶，单击桶名称进入概览页。
  b. 确保此OBS桶的归档数据直读功能关闭。如果此功能开启，可单击“归档数据直读”选项进行修改。

图 2-4 关闭归档数据直读功能

- 确保OBS中的文件是非加密状态
  上传图片或文件时不要选择KMS加密，否则会导致数据集读取失败。文件加密无法取消，请先解除桶加密，重新上传图片或文件。

图 2-5 OBS 桶中的文件未加密

检查图片是否符合要求

目前自动学习不支持四通道格式的图片。请检查您的数据，排除或删除四通道格式的图片。

检查标注框是否符合要求（物体检测）

目前物体检测仅支持矩形标注框。请确保所有图片的标注框为矩形框。

如果使用非矩形框，可能存在以下报错：

```
Error bandbox.
```

针对其他类型的项目（图像分类、声音分类等），无需关注此问题。

预测分析作业失败的排查思路

1. 检查用于预测分析的数据是否满足要求。

   由于预测分析任务未使用数据管理的功能发布数据集，因此当数据不满足训练作业要求时，会出现训练作业运行失败的错误。

   建议检查用于训练的数据，是否满足预测分析作业的要求。要求如下所示，如果数据满足要求，执行下一步检查。如果不满足要求，请根据要求进行数据调整后再重新训练。

   - 文件规范：名称由以字母数字及中划线下划线组成，以'.csv'结尾，且文件不能直接放在OBS桶的根目录下，应该存放在OBS桶的文件夹内。如：“/obs-xxx/data/input.csv”。
   - 文件内容：文件保存为“csv”文件格式，文件内容以换行符（即字符“\n”，或称为LF）分隔各行，行内容以英文逗号（即字符“,”）分隔各列。文件内容不能包含中文字符，列内容不应包含英文逗号、换行符等特殊字符，不支持引号语法，建议尽量以字母及数字字符组成。
   - 训练数据：训练数据列数一致，总数据量不少于100条不同数据（有一个特征取值不同，即视为不同数据）。训练数据列内容不能有时间戳格式（如：yy-mm-dd、yyyy-mm-dd等）的数据。确保指定标签列的取值至少有两个且无数据缺失，除标签列外数据集中至少还应包含两个有效特征列（列的取值至少有两个且数据缺失比例低于10%）。训练数据的csv文件不能包含表头，否则会导致训练失败。当前由于特征筛选算法限制，标签列建议放在数据集最后一列，否则可能导致训练失败。

2. 由于ModelArts会自动对数据进行一些过滤，过滤后再启动训练作业。当预处理后的数据不满足训练要求时，也会导致训练作业运行失败。

   对于数据集中列的过滤策略如下所示：

   - 如果某一列空缺的比例大于系统设定的阈值（0.9），此列数据在训练时将被剔除。
   - 如果某一列只有一种取值（即每一行的数据都是一样的），此列数据在训练时将被剔除。
   - 对于非纯数值列，如果此列的取值个数等于行数（即每一行的数值都是不一样的），此列数据在训练时将被剔除。

   经过上述过滤后，如果数据集不再满足第一点中关于训练数据的要求，则会导致训练失败或无法进行。建议完善数据后，再启动训练。

3. 数据集文件有以下限制：

a. 如果您使用2u8g规格，测试建议数据集文件应小于10MB。当文件大小符合限制要求，如果存在极端的数据规模（行数列数之积）时，仍可能会导致训练失败，建议的数据规模低于10000。

   如果您使用8u32g规格，测试建议数据集文件应小于100MB。当文件大小符合限制要求，如果存在极端的数据规模（行数列数之积）时，仍可能会导致训练失败，建议的数据规模低于1000000。

4. 如果上述排查操作仍无法解决，请**联系华为云技术支持**。

2.2.2 训练模型失败，报错：KMS.0314

问题现象

用户在使用自动学习项目，模型训练阶段，提示模型训练失败。

说明

该问题仅适用于国际站用户。

原因分析

出现该问题，是由于海外用户在购买或使用中国大陆云服务区的资源时，需要实名认证。如果在未进行实名认证的情况下使用中国大陆云服务区的资源实例训练模型时就会报错。

处理方法

为确保可以正常使用自动学习完成您的AI开发项目，请**完成账号的实名认证**后，再进行训练模型等其他操作。

2.3 部署上线

2.3.1 部署上线失败

出现此问题，一般是因为后台服务故障导致的，建议稍等片刻，然后重新部署在线服务。如果重试超过3次仍无法解决，请获取如下信息，并**联系华为云技术支持**协助解决故障。

- 获取服务ID。

  进入“部署上线>在线服务”页面，在服务列表中找到自动学习任务中部署的在线服务，自动学习部署的服务都是以“exeML-”开头的。单击服务名称进入服务详情页面，在“基本信息”区域，获取“服务ID”的值。

- 获取在线服务事件信息。

  进入服务详情页面后，单击“事件”页签，将事件信息表截图后反馈给技术支持人员。

2.4 模型发布

2.4.1 模型发布失败

模型发布任务提交失败和模型发布失败问题，一般是因为后台服务故障导致的，建议稍等片刻，然后重新创建训练作业。如果重试超过3次仍无法解决，请获取如下信息，并**联系华为云技术支持**协助解决故障。

- 获取模型ID。

  进入“模型管理”页面，在模型管理页面找到自动学习任务中自动创建的模型，自动学习产生的模型都是以“exeML-”开头的。单击模型名称进入模型详情页面，在“基本信息”区域，获取“ID”的值。

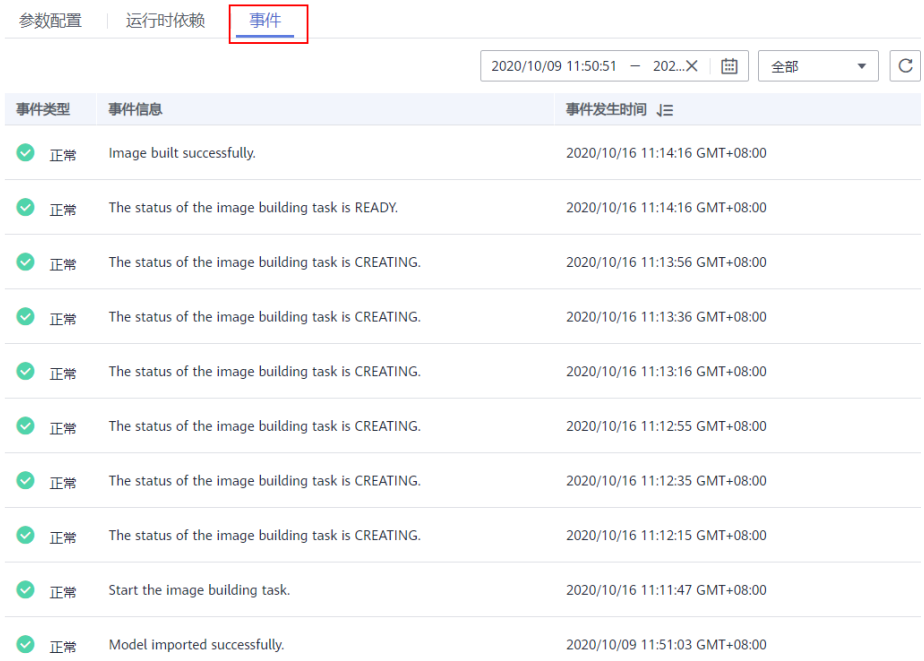
  图 2-6 获取模型 ID

- 获取模型事件信息。

  进入模型详情页面后，单击“事件”页签，将事件信息表截图后反馈给技术支持人员。

  图 2-7 获取事件信息

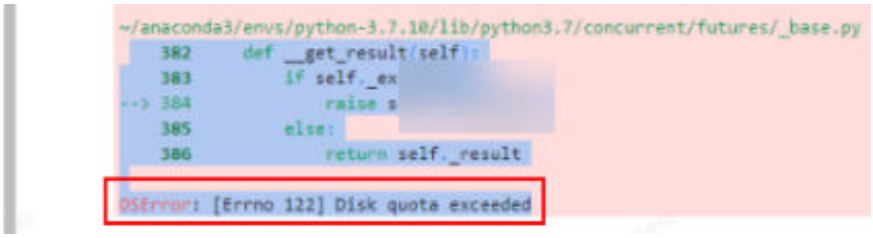
3 开发环境

3.1 环境配置故障

3.1.1 Notebook 提示磁盘空间已满

问题现象

- 在使用Notebook时，提示磁盘空间已满：No Space left on Device。
- 在Notebook执行代码时，出现如下报错，提示：Disk quota exceeded。



原因分析

- 在JupyterLab浏览器左侧导航删除文件后，会默认放入回收站占用内存，导致磁盘空间不足。
- 磁盘配额不足。

处理方法

查看虚拟机所使用的存储空间，再查看回收站文件占用内存，根据实际删除回收站里不需要的大文件。

1. 在Notebook实例详情页，查看实例的存储容量。
2. 执行如下命令，排查虚拟机所使用的存储空间，一般接近存储容量，请排查回收站占用内存。

```
cd /home/ma-user/work
du -h --max-depth 0
```

```
(PyTorch-1.4) [ma-user work]$cd /home/ma-user/work
(PyTorch-1.4) [ma-user work]$du -h --max-depth 0

23G     .
(PyTorch-1.4) [ma-user work]$
```

3. 执行如下命令，排查回收站占用内存（回收站文件默认在/home/ma-user/work/.Trash-1000/files下）。

```
cd /home/ma-user/work/.Trash-1000/
du -ah
```

```
(PyTorch-1.4) [ma-user work]$cd /home/ma-user/work/.Trash-1000/
(PyTorch-1.4) [ma-user .Trash-1000]$du -ah
2.0K    ./files/Untitled.ipynb
1000M   ./files/bigFile-Copy1.txt
977K    ./files/bigFile.txt
512     ./files/bigFile1.txt
9.8G    ./files/bigFile10.txt
9.8G    ./files/bigFile11.txt
21G     ./files
512     ./info/Untitled.ipynb.trashinfo
512     ./info/bigFile-Copy1.txt.trashinfo
512     ./info/bigFile.txt.trashinfo
512     ./info/bigFile1.txt.trashinfo
512     ./info/bigFile10.txt.trashinfo
512     ./info/bigFile11.txt.trashinfo
512     .
512     .
512     .
512     .
512     .
512     .
10K     ./info
21G     .
(PyTorch-1.4) [ma-user .Trash-1000]$
```

4. 根据实际删除回收站不需要的大文件。（注：请谨慎操作，文件删除后不可恢复）

```
rm {文件路径}
```

```
(PyTorch-1.4) [ma-user .Trash-1000]$pwd
/home/ma-user/work/.Trash-1000
(PyTorch-1.4) [ma-user .Trash-1000]$rm /home/ma-user/work/.Trash-1000/files/bigFile10.txt
(PyTorch-1.4) [ma-user .Trash-1000]$rm /home/ma-user/work/.Trash-1000/files/bigFile11.txt
```

说明

如果删除的文件夹或者文件中带有空格，需要给文件夹或文件加上单引号。如图示例：

```
(PyTorch-1.8) [ma-user files]$rm -rf ./'1 6'
(PyTorch-1.8) [ma-user files]$ll
```

5. 执行如下命令，再次检查虚拟机所使用的存储空间。

```
cd /home/ma-user/work
du -h --max-depth 0
```

6. 如果Notebook实例的存储配置采用的是云硬盘EVS，可在Notebook详情页申请扩容磁盘。

建议与总结

建议在使用Notebook时注意磁盘空间大小，随时删除不需要的文件。以免因磁盘空间问题导致训练失败。

3.1.2 Notebook 中使用 Conda 安装 Keras 2.3.1 报错

问题现象

使用Conda安装Keras 2.3.1版本报错。

```
conda install keras=2.3.1
```

```
/home/ma-user/anaconda3/lib/python3.7/site-packages/requests/__init__.py:91: RequestsDependencyWarning: urllib3 (1.26.12)
  RequestsDependencyWarning)
Collecting package metadata (current_repodata.json): done
Solving environment: failed with initial frozen solve. Retrying with flexible solve.
Collecting package metadata (repodata.json): done
Solving environment: failed with initial frozen solve. Retrying with flexible solve.
Solving environment: -
Found conflicts! Looking for incompatible packages.
This can take several minutes.  Press CTRL-C to abort.
                                                                                failed

# >>>>>>>>>>>>>>>>>>>>>> ERROR REPORT <<<<<<<<<<<<<<<<<<<<<<

    Traceback (most recent call last):
      File "/home/ma-user/anaconda3/lib/python3.7/site-packages/conda/cli/install.py", line 265, in install
        should_retry_solve=(_should_retry_unfrozen or repodata_fn != repodata_fns[-1]),
      File "/home/ma-user/anaconda3/lib/python3.7/site-packages/conda/core/solve.py", line 117, in solve_for_transaction
        should_retry_solve)
      File "/home/ma-user/anaconda3/lib/python3.7/site-packages/conda/core/solve.py", line 158, in solve_for_diff
        force_remove, should_retry_solve)
      File "/home/ma-user/anaconda3/lib/python3.7/site-packages/conda/core/solve.py", line 275, in solve_final_state
        ssc = self._add_specs(ssc)
      File "/home/ma-user/anaconda3/lib/python3.7/site-packages/conda/core/solve.py", line 696, in _add_specs
        raise UnsatisfiableError({})
    conda.exceptions.UnsatisfiableError:
    Did not find conflicting dependencies. If you would like to know which
    packages conflict ensure that you have enabled unsatisfiable hints.

    conda config --set unsatisfiable_hints True
```

原因分析

可能是Conda网络不通，请使用**pip install**命令安装。

解决方法

执行 **!pip install keras==2.3.1**命令安装Keras。

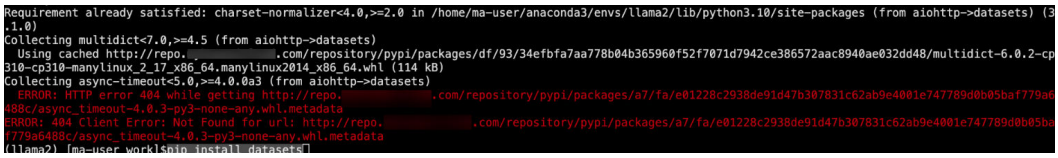
```
!pip install keras==2.3.1
```

```
Looking in indexes: http://repo.                !.com/repository/pypi/simple
Collecting keras==2.3.1
  Using cached http://repo.                .com/repository/pypi/packages/ad/fd/6bfe87920d7f4fd475acd28500a42482b6b84479832bdc0fe9e589a60ceb/Keras-2.3.1-py2.py3-none-any.whl
Requirement already satisfied: keras-applications>=1.0.6 in /home/ma-user/anaconda3/envs/TensorFlow-1.13-gpu/lib/python3.7/site-packages (from keras==2.3.1) (1.0.8)
Requirement already satisfied: keras-preprocessing>=1.0.5 in /home/ma-user/anaconda3/envs/TensorFlow-1.13-gpu/lib/python3.7/site-packages (from keras==2.3.1) (1.1.2)
Requirement already satisfied: numpy>=1.9.1 in /home/ma-user/anaconda3/envs/TensorFlow-1.13-gpu/lib/python3.7/site-packages (from keras==2.3.1) (1.21.5)
Requirement already satisfied: pyyaml in /home/ma-user/anaconda3/envs/TensorFlow-1.13-gpu/lib/python3.7/site-packages (from keras==2.3.1) (5.1)
Requirement already satisfied: scipy>=0.14 in /home/ma-user/anaconda3/envs/TensorFlow-1.13-gpu/lib/python3.7/site-packages (from keras==2.3.1) (1.7.3)
Requirement already satisfied: h5py in /home/ma-user/anaconda3/envs/TensorFlow-1.13-gpu/lib/python3.7/site-packages (from keras==2.3.1) (2.8.0)
Requirement already satisfied: six>=1.9.0 in /home/ma-user/anaconda3/envs/TensorFlow-1.13-gpu/lib/python3.7/site-packages (from keras==2.3.1) (1.16.0)
Installing collected packages: keras
  Attempting uninstall: keras
    Found existing installation: Keras 2.2.4
    Uninstalling Keras-2.2.4:
      Successfully uninstalled Keras-2.2.4
Successfully installed keras-2.3.1
```

3.1.3 Notebook 中安装依赖包报错 ERROR: HTTP error 404 while getting xxx

问题现象

在Notebook中安装依赖包时报错，报错截图如下：



原因分析

pypi源没有这个包或源不可用。

解决方案

使用别的源下载。

```
pip install -i 源地址 包名
```

3.1.4 Notebook 中已安装对应库，仍报错 import numba ModuleNotFoundError: No module named 'numba'

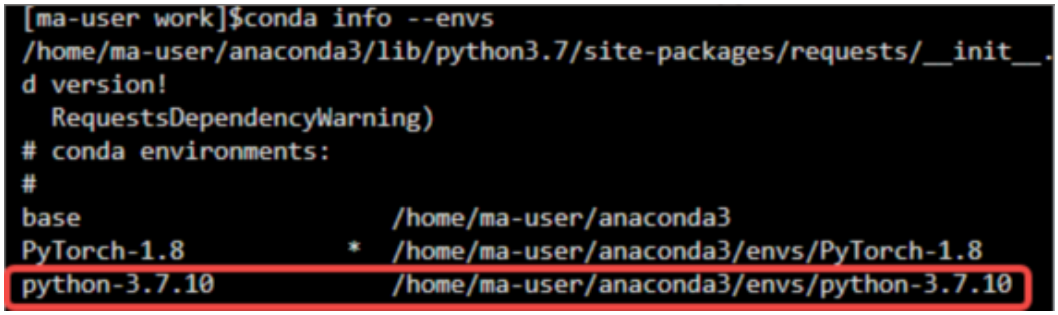
问题现象

在Notebook中使用**!pip install numba**命令安装了numba库且运行正常（且已保存为自定义镜像）， 然后使用DataArts执行此脚本的任务时提示没有这个库。

原因分析

客户创建了多个虚拟环境，numba库安装在了python-3.7.10中，如图3-1所示。

图 3-1 查询创建的虚拟环境



解决方案

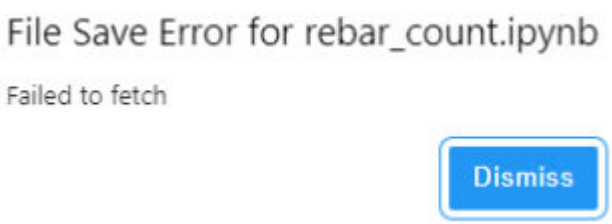
在Terminal中执行**conda deactivate**命令退出当前虚拟环境，默认进入base环境。执行**pip list**命令查询已安装的包，然后安装需要的依赖进行保存，最后切换至指定的虚拟环境后再运行脚本。

```
Using user ma-user
Ubuntu 18.04.6 LTS, CUDA-10.2
Tips:
1) Navigate to the target conda environment. For details, see /home/ma-user/README.
2) Copy (Ctrl+C) and paste (Ctrl+V) on the jupyter terminal.
3) Store your data in /home/ma-user/work, to which a persistent volume is mounted.
(PyTorch-1.8) [ma-user work]$conda deactivate
(base) [ma-user work]$conda deactivate
[ma-user work]$pip list
Package                       Version
----------------------------- ------------------------
absl-py                       1.3.0
addict                        2.4.0
APScheduler                   3.9.1
```

3.1.5 JupyterLab 中文件保存失败，如何解决?

问题现象

JupyterLab中保存文件时报错如下：



原因分析

- 浏览器安装了第三方插件proxy进行了拦截，导致无法进行保存。
- 在Notebook中的运行文件超过指定大小就会提示此报错。
- jupyter页面打开时间太长。
- 网络环境原因，是否有连接网络代理。

解决方法

- 关掉插件然后重新保存。
- 减少文件大小。
- 重新打开jupyter页面。
- 请检查网络。

3.1.6 用户结束 kernelgateway 进程后报错 Server Connection Error，如何恢复?

问题现象

当kernelgateway进程被结束后，出现如下报错，以及选不到Kernel。

图 3-2 报错 Server Connection Error 截图

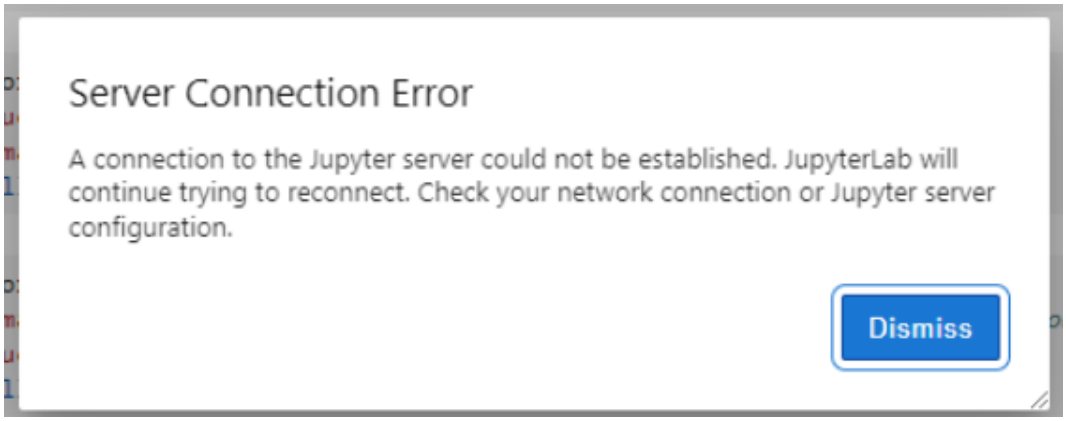
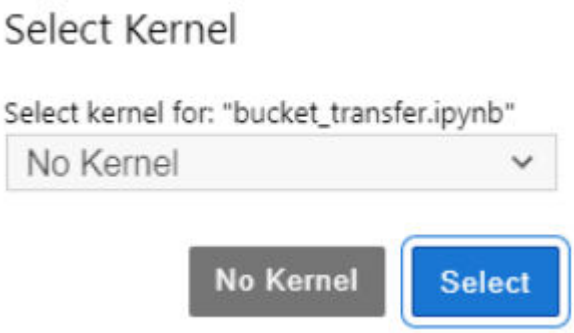


图 3-3 选不到 Kernel



原因分析

用户误操作引起的。

解决方案

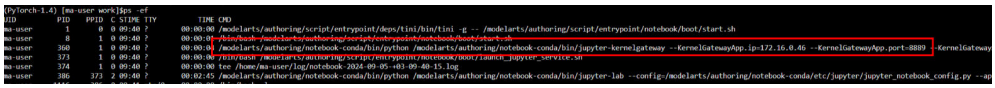
1. 打开Terminal窗口，执行以下命令启动kernelgateway服务。

```
API_TYPE=kernel_gateway.jupyter_websocket
LOG_DIR="/home/ma-user/log"
mkdir -p ${LOG_DIR}
KERNEL_GATEWAY_LOG_FILE="${LOG_DIR}/kernelgateway-`date +"%Y-%m-%d-%Z-%H-%M-%S"`.log"

jupyter kernelgateway --KernelGatewayApp.ip=${HOST_IP} --KernelGatewayApp.port=8889 --KernelGatewayApp.api=${API_TYPE} --KernelGatewayApp.auth_token=${JPY_AUTH_TOKEN} --JupyterWebsocketPersonality.list_kernels=True --debug >> "${KERNEL_GATEWAY_LOG_FILE}" 2>&1 &
chmod 640 ${KERNEL_GATEWAY_LOG_FILE}
```

2. 执行命令**ps -ef**检查进程是否启动。

图 3-4 检查进程是否启动



3.2 实例故障

3.2.1 创建 Notebook 失败，查看事件显示 JupyterProcessKilled

问题现象

创建Notebook失败，查看事件显示JupyterProcessKilled。

原因分析

出现此故障是因为Jupyter进程被清理掉了，一般情况Notebook会自动重启的，如果没有自动重启，创建一直失败，请确认是否是自定义镜像的问题。

解决方案

排查是否是自定义镜像的问题。

自定义镜像构建完成，在ModelArts镜像管理注册时，“架构”和“类型”需要和源镜像保持一致。

图 3-5 注册镜像

3.2.2 创建 Notebook 实例后无法打开页面，如何处理?

如果您在创建Notebook实例之后，打开Notebook时，因报错导致无法打开页面，您可以根据以下对应的错误码来排查解决。

打开 Notebook 显示黑屏

Notebook打开后黑屏，由于代理问题导致，切换代理。

打开 Notebook 显示空白

- 打开Notebook时显示空白，请清理浏览器缓存后尝试重新打开。
- 检查浏览器是否安装了过滤广告组件，如果是，请关闭该组件。

报错 404

如果是IAM用户在创建实例时出现此错误，表示此IAM用户不具备对应存储位置（OBS桶）的操作权限。

解决方法：

1. 使用账号登录OBS，并将对应OBS桶的访问权限授予该IAM用户。详细操作指导请参见：**被授权用户**。
2. IAM用户获得权限后，登录ModelArts管理控制台，删除该实例，然后重新使用此OBS路径创建Notebook实例。

报错 503

如果出现503错误，可能是由于该实例运行代码时比较耗费资源。建议先停止当前Notebook实例，然后重新启动。

报错 504

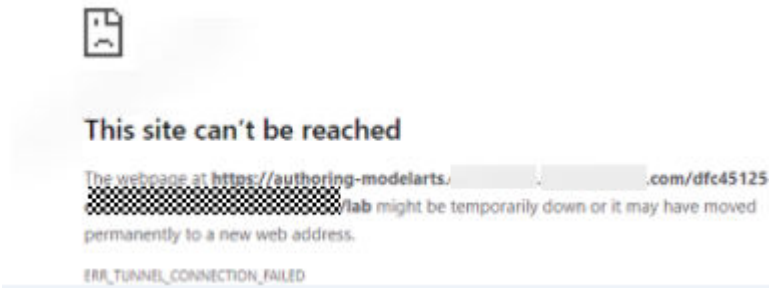
如果报此错误时，请**提工单**或拨打热线电话协助解决。

报错 500

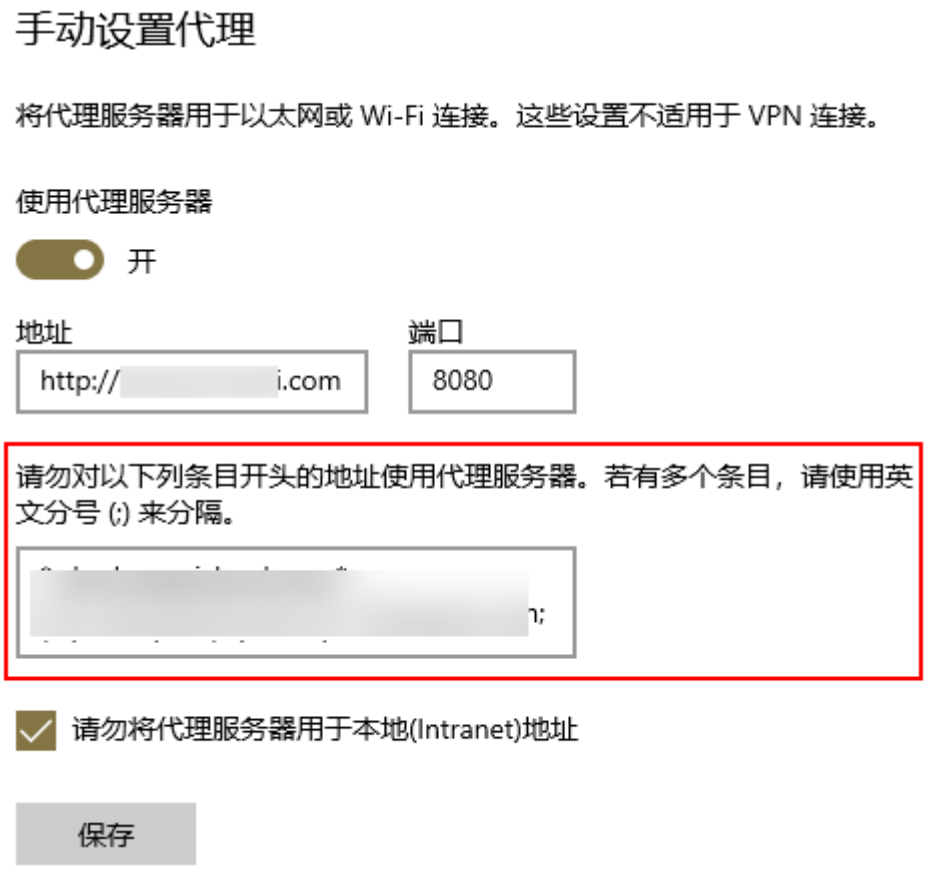
Notebook JupyterLab页面无法打开，报错500，可能是工作目录work下的磁盘空间满了，请参考**Notebook提示磁盘空间已满**排查并清理磁盘空间。

报错 This site can't be reached

创建完Notebook后，单击操作列的“打开”，报错如下：



解决方案：复制页面的域名，添加到windows代理“请勿对以下列条目开头的地址使用代理服务器”中，然后保存就可以正常打开。



3.2.3 使用 pip install 时出现“没有空间”的错误

问题现象

在Notebook实例中，使用**pip install**时，出现“No Space left...”的错误。

解决办法

建议使用**pip install --no-cache **** 命令安装，而不是使用**pip install ****。

加上“--no-cache”参数，可以解决很多此类报错。

3.2.4 出现“save error”错误，可以运行代码，但是无法保存

如果当前Notebook还可以运行代码，但是无法保存，保存时会提示“save error”错误。

大多数原因是华为云WAF安全拦截导致的。当前页面，即用户的输入或者代码运行的输出有一些字符被华为云拦截，认为有安全风险。

出现此问题时，请提交工单，联系专业的工程师帮您核对并处理问题。

3.2.5 出现 ModelArts.6333 错误，如何处理?

问题现象

在使用Notebook过程中，界面出现“ModelArts.6333”报错信息。

原因分析

可能由于实例过负载引起故障，Notebook正在自动恢复中，请刷新页面并等待几分钟。常见原因是内存占用满。

处理方法

当出现此错误时，Notebook会自动恢复，您可以刷新页面，等待几分钟。

由于出现此错误，常见原因是内存占用满导致的，您可以尝试使用如下方法，从根本上解决错误。

- 方法1：将Notebook更换为更高规格的资源。
- 方法2：可以参考如下方法调整代码中的参数，减少内存占用。如果代码调整后仍然出现内存不足的情况，请使用方法1。
  a. 调用**sklearn**方法**silhouette_score(addr_1,siteskmeans.labels)**，可以指定参数**sample_size**来减少内存占用。
  b. 调用**train**方法的时候可以尝试减少**batch_size**等参数。

3.2.6 打开 Notebook 实例提示 token 不存在或者 token 丢失如何处理?

问题现象

把已打开的Notebook url发送给他人使用，他人无法打开，报错“……lost token or incorrect token……”。

原因分析

原因是由于其他人没有此账号的令牌导致。

解决方案

在此url后面加上Notebook拥有者的token。

3.3 代码运行故障

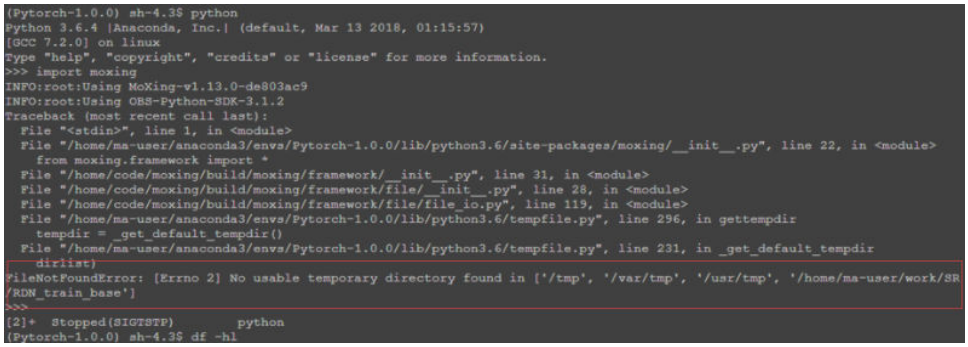
3.3.1 Notebook 运行代码报错，在'/tmp'中到不到文件

问题现象

使用Notebook运行代码，报错：

```
FileNotFoundError: [Error 2] No usable temporary directory found in ['/tmp', '/var/tmp', '/usr/tmp', 'home/ma-user/work/SR/RDN_train_base']
```

图 3-6 运行代码报错



原因分析

根据报错提示，需要排查是否将大量数据被保存在“/tmp”中。

处理方法

1. 进入到“Terminal”界面。在“/tmp”目录下，执行命令**du -sh ***，查看该目录下的空间占用情况。

   ```
   sh-4.3$cd /tmp
   sh-4.3$du -sh *
   4.0K    core-js-banners
   0       npm-19-41ed4c62
   6.7M    v8-compile-cache-1000
   ```

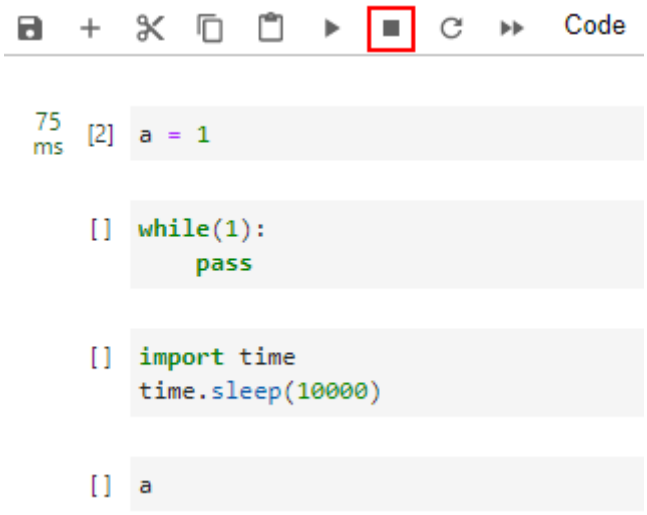
2. 请删除不用的大文件。
   a. 删除示例文件“test.txt”： **rm -f /home/ma-user/work/data/test.txt**
   b. 删除示例文件夹“data”： **rm -rf /home/ma-user/work/data/**

3.3.2 Notebook 无法执行代码，如何处理?

当Notebook出现无法执行时，您可以根据如下几种情况判断并处理。

1. 如果只是Cell的执行过程卡死或执行时间过长，如图3-7中的第2个和第3个Cell，导致第4个Cell无法执行，但整个Notebook页面还有反应，其他Cell也还可以单击，则直接单击下图中红色方框处的“interrupt the kernel”，停止所有Cell的执行，同时会保留当前Notebook中的所有变量空间。

图 3-7 停止所有 Cell



2. 如果整个Notebook页面也已经无法使用，单击任何地方都无反应，则关闭Notebook页面，关闭ModelArts管理控制台页面。然后，重新打开管理控制台，打开之前无法使用的Notebook，此时的Notebook仍会保留无法使用之前的所有变量空间。
3. 如果重新打开的Notebook仍然无法使用，则进入ModelArts管理控制台页面的Notebook列表页面，“停止”此无法使用的Notebook。待Notebook处于“停止”状态后，再单击“启动”，重新启动此Notebook，并打开Notebook。此时，Notebook仍会保留无法使用之前的所有变量空间。

3.3.3 运行训练代码，出现 dead kernel，并导致实例崩溃

在Notebook实例中运行训练代码，如果数据量太大或者训练层数太多，亦或者其他原因，导致出现“内存不够”问题，最终导致该容器实例崩溃。

出现此问题后，系统将自动重启Notebook，来修复实例崩溃的问题。此时只是解决了崩溃问题，如果重新运行训练代码仍将失败。

如果您需要解决“内存不够”的问题，建议您创建一个新的Notebook，使用更高规格的资源池，比如专属资源池来运行此训练代码。

已经创建成功的Notebook不支持选用更高规格的资源规格进行扩容。

3.3.4 如何解决训练过程中出现的 cudaCheckError 错误?

问题现象

Notebook中，运行训练代码出现如下错误。

```
cudaCheckError() failed : no kernel image is available for execution on the device
```

原因分析

因为编译的时候需要设置setup.py中编译的参数arch和code和电脑的显卡匹配。

解决方法

对于GP Vnt1的显卡，GPU算力为**-gencode arch=compute_70,code=[sm_70,compute_70]**，设置setup.py中的编译参数即可解决。

3.3.5 如何处理使用 opencv.imshow 造成的内核崩溃?

问题现象

当在Notebook中使用opencv.imshow后，会造成Notebook崩溃。

原因分析

opencv的cv2.imshow在jupyter这样的client/server环境下存在问题。 而matplotlib不存在这个问题。

解决方法

参考如下示例进行图片显示。注意opencv加载的是BGR格式， 而matplotlib显示的是RGB格式。

Python语言：

```
from matplotlib import pyplot as plt
import cv2
img = cv2.imread('图片路径')
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.title('my picture')
plt.show()
```

3.3.6 使用 Windows 下生成的文本文件时报错找不到路径?

问题现象

当在Notebook中使用Windows下生成的文本文件时，文本内容无法正确读取，可能报错找不到路径。

原因分析

Notebook是Linux环境，和Windows环境下的换行格式不同，Windows下是CRLF，而Linux下是LF。

解决方法

可以在Notebook中转换文件格式为Linux格式。

shell语言：

```
dos2unix  文件名
```

3.3.7 创建 Notebook 文件后，右上角的 Kernel 状态为“No Kernel”如何处理?

问题现象

现象：创建Notebook文件后，右上角的Kernel状态为“No Kernel”。

原因分析

可能因为用户工作目录下的code.py和创建kernel依赖的import code文件名称冲突。

解决方案

1. 查看“/home/ma-user/log/”下以“kernelgateway”开头的最新日志文件，搜索“Starting kernel”附近的日志。如果看到如下类似的堆栈，可看到是因为用户工作目录下的“code.py”和创建kernel依赖的import code文件名冲突：

```
[KernelGatewayApp] Starting kernel: ['/home/ma-user/anaconda3/envs/PyTorch-1.8/bin/python', '-m', 'ipykernel', '-f', '/home/ma-user/.local/share/jupyter/runtime/kernel-6df62665-ebde-4dff-8d3a-bd22ef8a17c3.json']
[KernelGatewayApp] Connecting to: tcp://127.0.0.1:52075
Traceback (most recent call last):
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/runpy.py", line 193, in _run_module_as_main
    "__main__", mod_spec)
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/runpy.py", line 85, in _run_code
    exec(code, run_globals)
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/ipykernel/__main__.py", line 2, in <module>
    from ipykernel import kernelapp as app
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/ipykernel/kernelapp.py", line 42, in <module>
    from .ipkernel import IPythonKernel
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/ipykernel/ipkernel.py", line 38, in <module>
    from .debugger import Debugger
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/ipykernel/debugger.py", line 21, in <module>
    from debugpy.server import api  # noqa
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/debugpy/server/__init__.py", line 7, in <module>
    import debugpy._vendored.force_pydevd  # noqa
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/debugpy/_vendored/force_pydevd.py", line 28, in <module>
    pydevd_constants = import_module('_pydevd_bundle.pydevd_constants')
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/importlib/__init__.py", line 127, in import_module
    return _bootstrap._gcd_import(name[level:], package, level)
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/debugpy/_vendored/pydevd/_pydevd_bundle/pydevd_constants.py", line 379, in <module>
    from _pydev_bundle._pydev_saved_modules import thread, threading
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/debugpy/_vendored/pydevd/_pydev_bundle/_pydev_saved_modules.py", line 91, in <module>
    import code as _code;    verify_shadowed.check(_code, ['compile_command', 'InteractiveInterpreter'])
  File "/home/ma-user/anaconda3/envs/PyTorch-1.8/lib/python3.7/site-packages/debugpy/_vendored/pydevd/_pydev_bundle/_pydev_saved_modules.py", line 75, in check
    raise DebuggerInitializationError(msg)
_pydev_bundle._pydev_saved_modules.DebuggerInitializationError: It was not possible to initialize the debugger due to a module name conflict.

i.e.: the module "code" could not be imported because it is shadowed by:
/home/ma-user/work/test1/code.py
Please rename this file/folder so that the original module from the standard library can be imported.
```

2. 重命名当前工作目录下和创建kernel依赖的库文件冲突的文件名称。

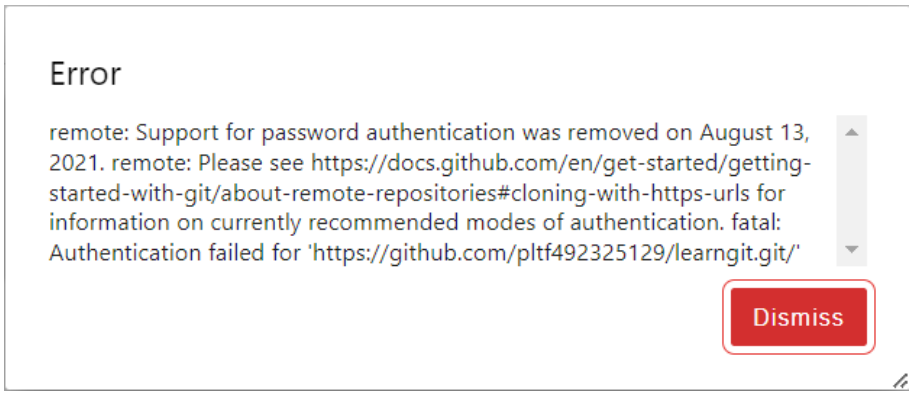
   常见容易冲突的文件：code.py、select.py。

3.4 JupyterLab 插件故障

3.4.1 git 插件密码失效如何解决?

问题现象

在JupyterLab中使用git插件时，当git clone私有仓库和git push文件时会出现如下报错：

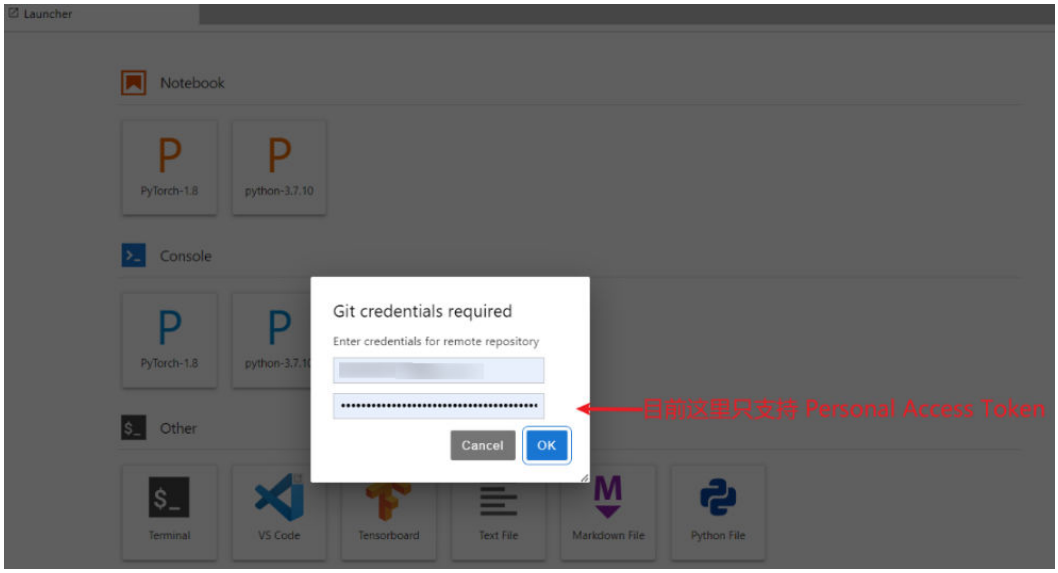


原因分析

原因为Github已取消密码授权方式，此时在git clone私有仓库和git push文件时需要在授权方式框中输入token。

解决方案

使用token替换原先的密码授权方式，在git clone私有仓库和git push文件时，需要在授权方式框中输入token（见下图）；具体获取token方式请参考**查看GitHub中Personal Access Token信息**。



3.5 VS Code 连接开发环境失败故障处理

3.5.1 在 ModelArts 控制台界面上单击 VS Code 接入并在新界面单击打开，未弹出 VS Code 窗口

原因分析

未安装VS Code或者安装版本过低。

解决方法

下载并安装VS Code（Windows用户请单击“Win”，其他用户请单击“其他”下载），安装完成后单击“刷新”完成连接。

3.5.2 在 ModelArts 控制台界面上单击 VS Code 接入并在新界面单击打开，VS Code 打开后未进行远程连接

须知

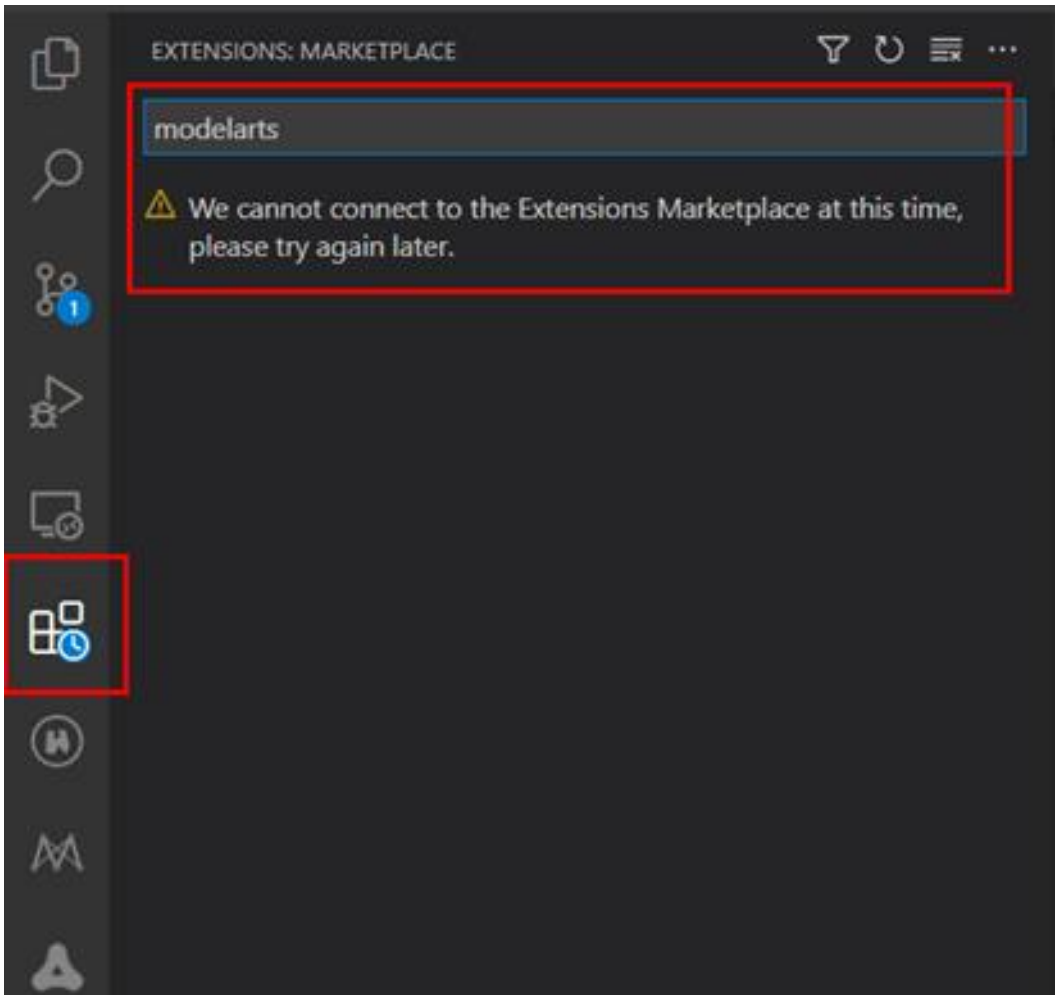
如果本地为Linux系统，见原因分析二。

原因分析一

自动安装VS Code插件ModelArts-HuaweiCloud失败。

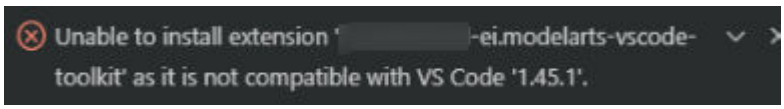
解决方法一

方法一：检查VS Code网络是否正常。在VS Code插件市场上搜索ModelArts-HuaweiCloud，如果显示如下则网络异常，请切换代理或使用其他网络。



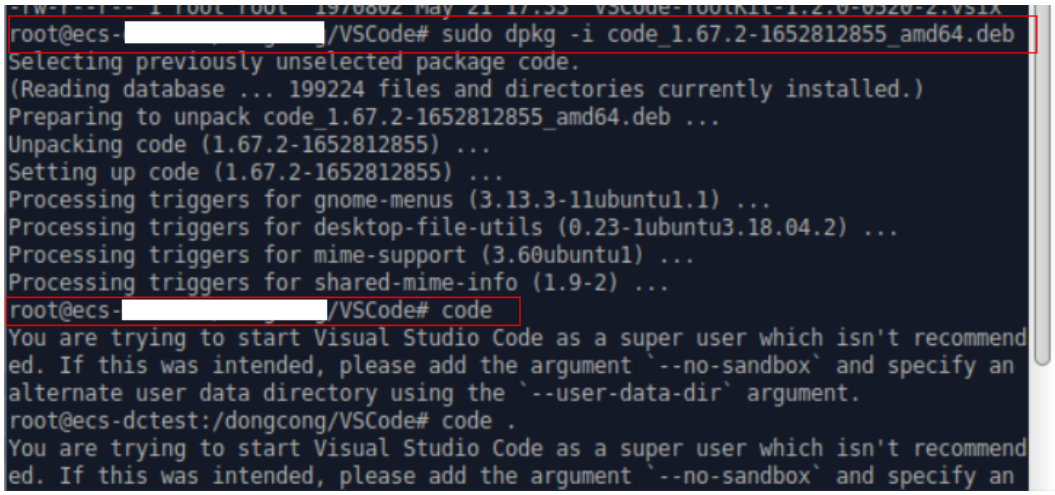
操作完成后再次执行搜索，如果显示如下则网络正常，请回到ModelArts控制台界面再次单击界面上的“VS Code接入”按钮。

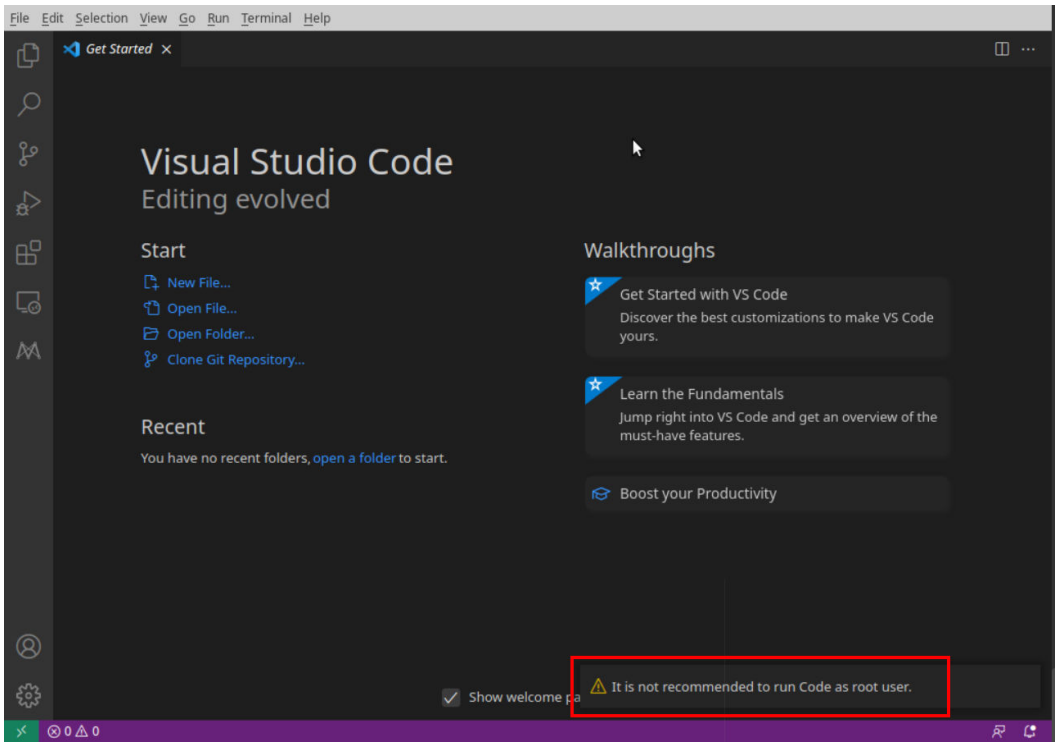
方法二：出现如下图报错，是由于VS Code版本过低，建议升级VS Code版本为1.57.1或者最新版。



原因分析二

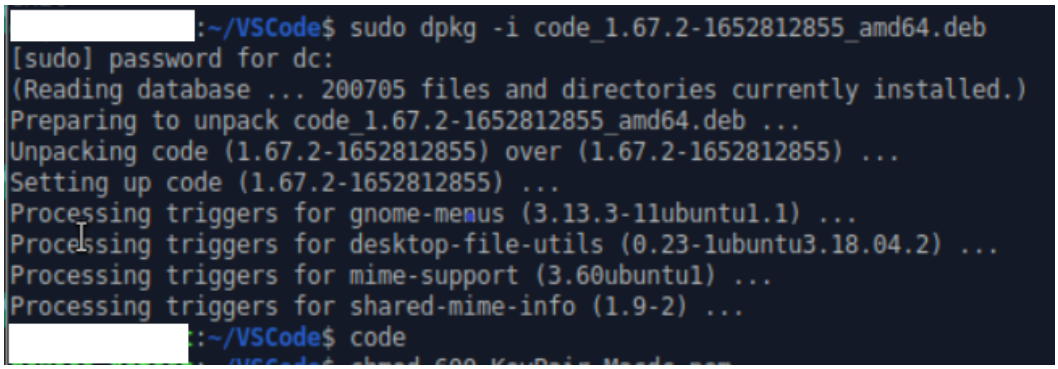
本地系统为Linux，由于使用root用户安装VS Code，打开VS Code显示信息It is not recommended to run Code as root user

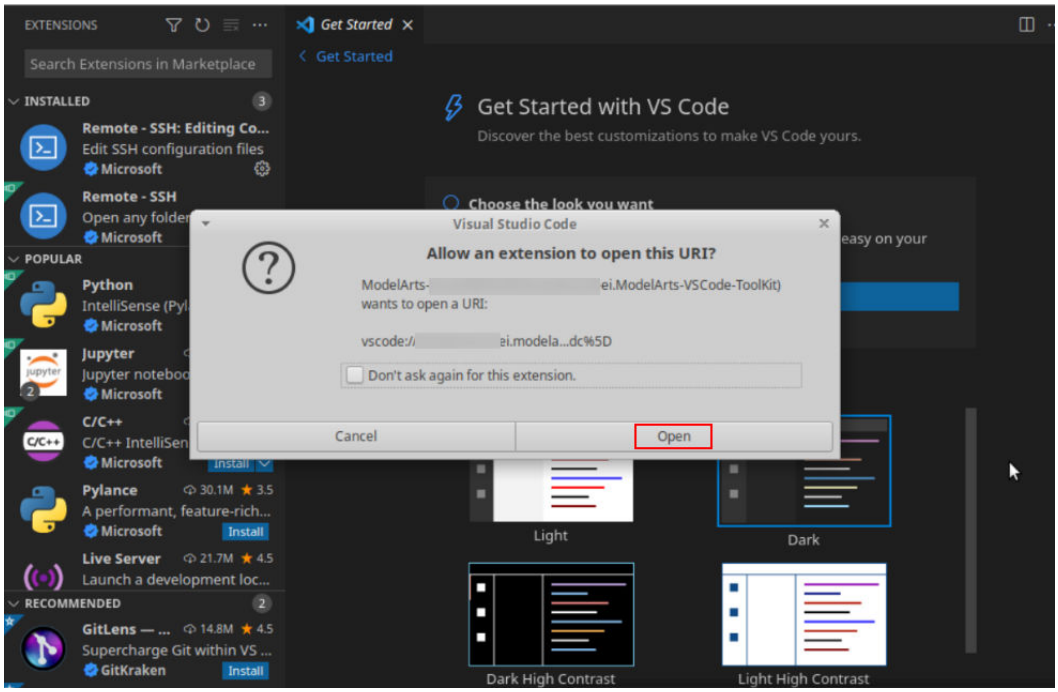




解决方法二

请使用非root用户安装VS Code后，回到ModelArts控制台界面再次单击界面上的“VS Code接入”按钮。





3.5.3 VS Code 连接开发环境失败时的排查方法

VS Code连接开发环境失败时，请参考以下步骤进行基础排查。

网络链路检查

1. 在ModelArts控制台查看Notebook实例状态是否正常，确保实例无问题。
2. 在VS Code Terminal里执行如下命令检测SSH命令是否可用；
   ```
   ssh -i <密钥相对路径> -p <端口> ma-user@<域名/ip>
   ```
   - SSH可用时跳过3继续远端排查。
   - SSH不可用，排查3。
3. 在VS Code Terminal里执行如下检查网络。如果网络异常，请执行命令检查端口。
   ```
   curl -kv telnet://<域名/ip>:<port>
   ```
   - 端口有问题，请联系技术支持。
   - 端口无问题请继续远端排查。

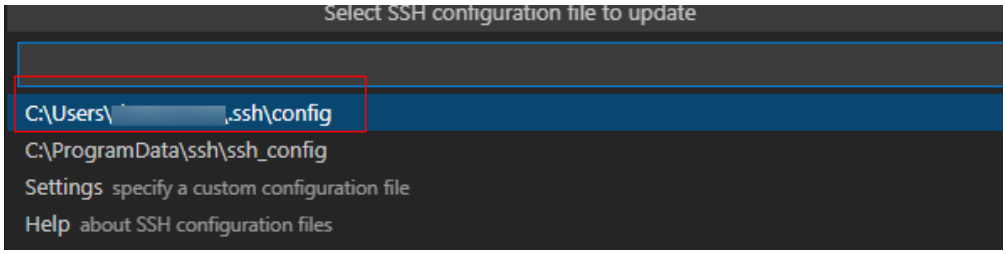
远端排查

1. 排查/home/ma-user目录权限是否为755/750，不是该权限，请执行如下命令设置权限。
```
chmod 755 /home/ma-user
```
2. 排查/home/ma-user/.ssh目录权限是否为755/750，不是该权限请修改。
3. 连接时如果报错密钥无权限，排查密钥是否为自己的密钥（可能使用了重名密钥），请更换密钥后重新连接实例。

本地排查

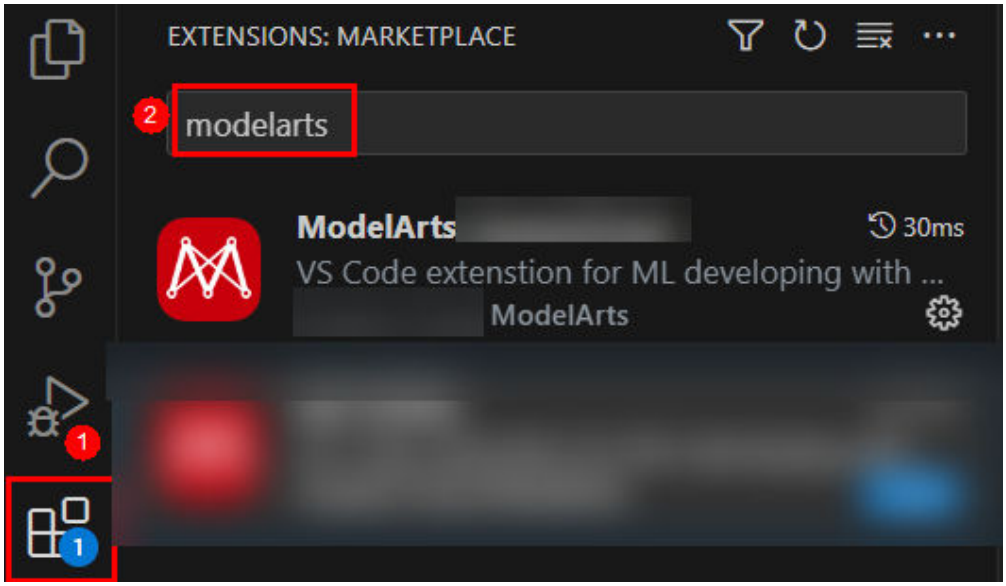
1. 检查配置是否正确。

   打开config文件进行检查：Host必须放在每组配置的第一行，作为每组配置的唯一ID。




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

   - 如果正确请按继续排查。
   - 如果不正确请按上面格式修改后继续排查。
2. 查看密钥文件的路径，建议放在C:\Users\{user}\.ssh下，并确保密钥文件无中文字符。
3. 排查插件包是否为最新版：在extensions中搜索，看是否需要升级。检查Remote-ssh三方插件是否兼容。

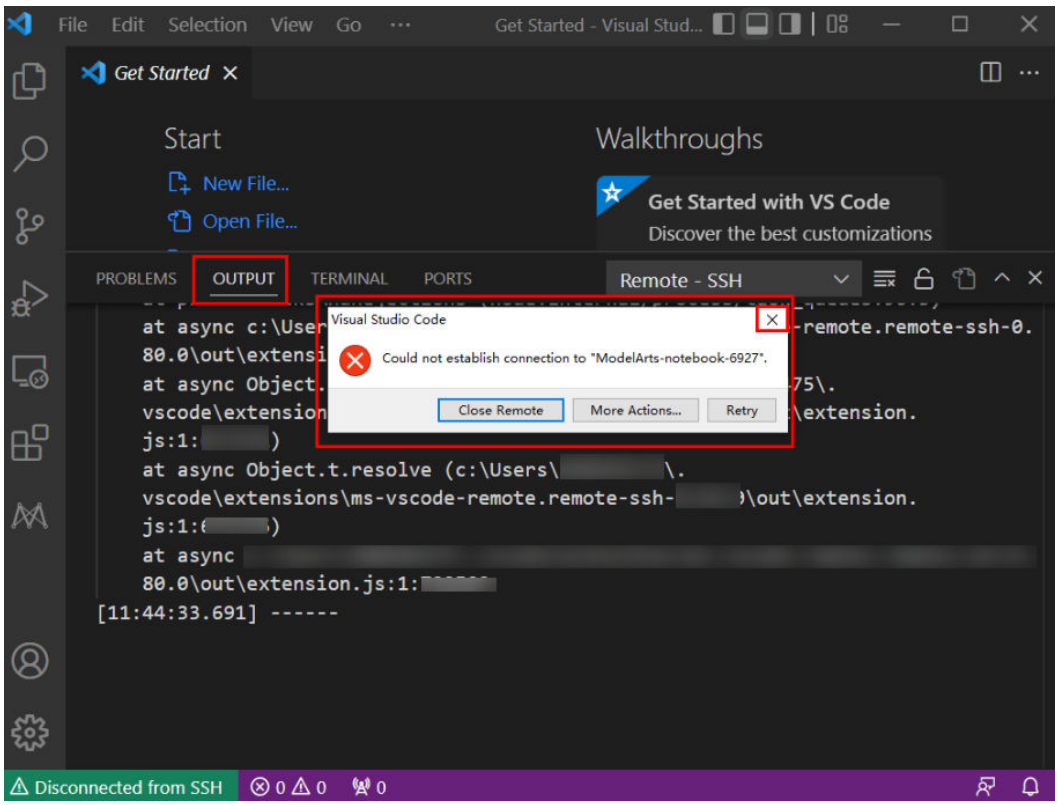


4. 检查本地Vscode是否为最新版，最新版可能有bug，建议使用推荐版本v1.82。

如果以上步骤排查均无问题仍未解决，请联系技术支持定位。

3.5.4 远程连接出现弹窗报错：Could not establish connection to xxx

问题现象



原因分析

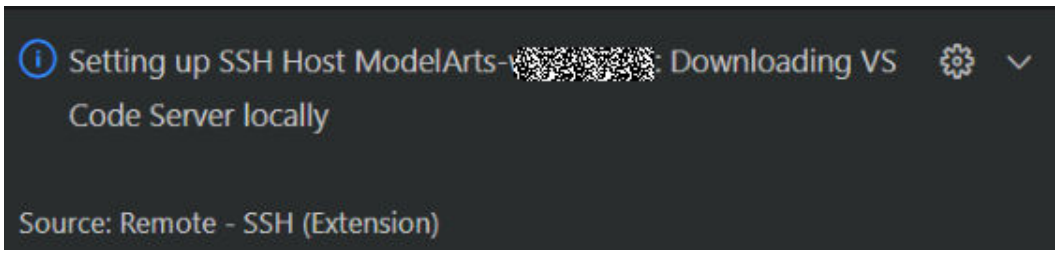
执行VS Code Remote SSH连接失败。

解决方法

单击弹窗右上角关闭弹窗，查看OUTPUT中的具体报错信息，并参考后续章节列举的几种常见报错解决问题。

3.5.5 连接远端开发环境时，一直处于"Setting up SSH Host xxx: Downloading VS Code Server locally"超过 10 分钟以上，如何解决?

问题现象

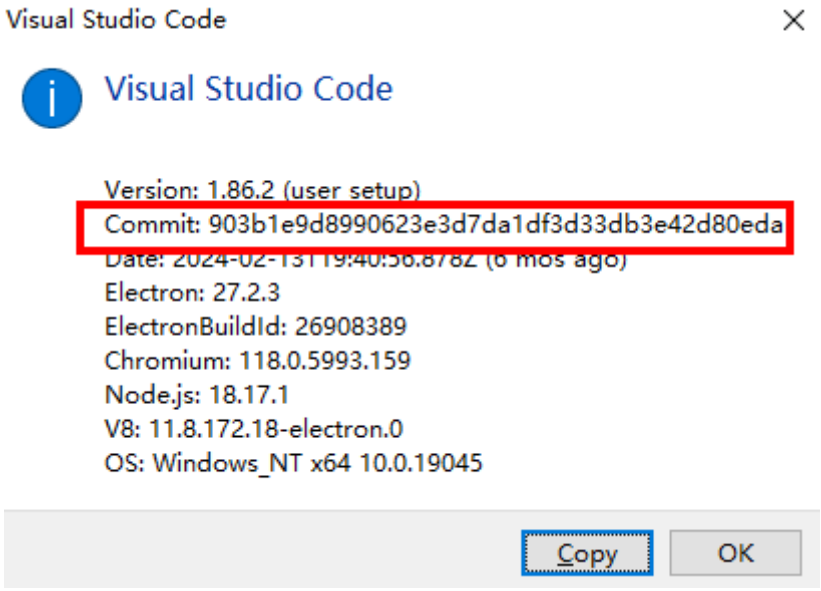


原因分析

当前本地网络原因，导致远程自动安装VS Code Server时间过长。

解决方法

1. 打开VS Code，选择“Help>About”，并记下“Commit”的ID码。



2. 确认创建Notebook实例使用的镜像的系统架构，可以在Notebook中打开Terminal，通过命令**uname -m**查看。
3. 下载对应版本的vscode-server，根据Commit码和Notebook实例镜像架构下载。

   说明

   如果下载报错“Not Found”，请下载别的版本VS Code重新在本地安装，目前推荐:Vscode-1.86.2。

   - 如果实例的架构是x86_64的，通过下面的链接，手动修改Commit码（Commit码替换时去掉尖括号），使用浏览器下载vscode-server-linux-x64.tar.gz文件。

     https://update.code.visualstudio.com/commit:<Commit码>/server-linux-x64/stable

- 如果实例的架构是aarch的，通过下面的链接，手动修改comment-id（commit-id替换时去掉尖括号），使用浏览器下载vscode-server-linux-arm64.tar.gz文件。下载完成后，将下载的vscode-server-linux-arm64.tar.gz文件重命名为“vscode-server-linux-x64.tar.gz”。

  ```
  https://update.code.visualstudio.com/commit:<提交的ID码>/server-linux-arm64/stable
  ```

  例如：commit-id是863d2581ecda6849923a2118d93a088b0745d9d6，os架构是x86_64，修改链接为：

  ```
  https://update.code.visualstudio.com/commit:863d2581ecda6849923a2118d93a088b0745d9d6/server-linux-x64/stable
  ```

4. 将下载的vscode-server-linux-x64.tar.gz，上传到ModelArts实例的“/home/ma-user/work”目录下。

   ```
   (PyTorch-1.4) [ma-user work]$ls vscode-server*
   vscode-server-linux-x64.tar.gz
   (PyTorch-1.4) [ma-user work]$pwd
   /home/ma-user/work
   (PyTorch-1.4) [ma-user work]$
   ```

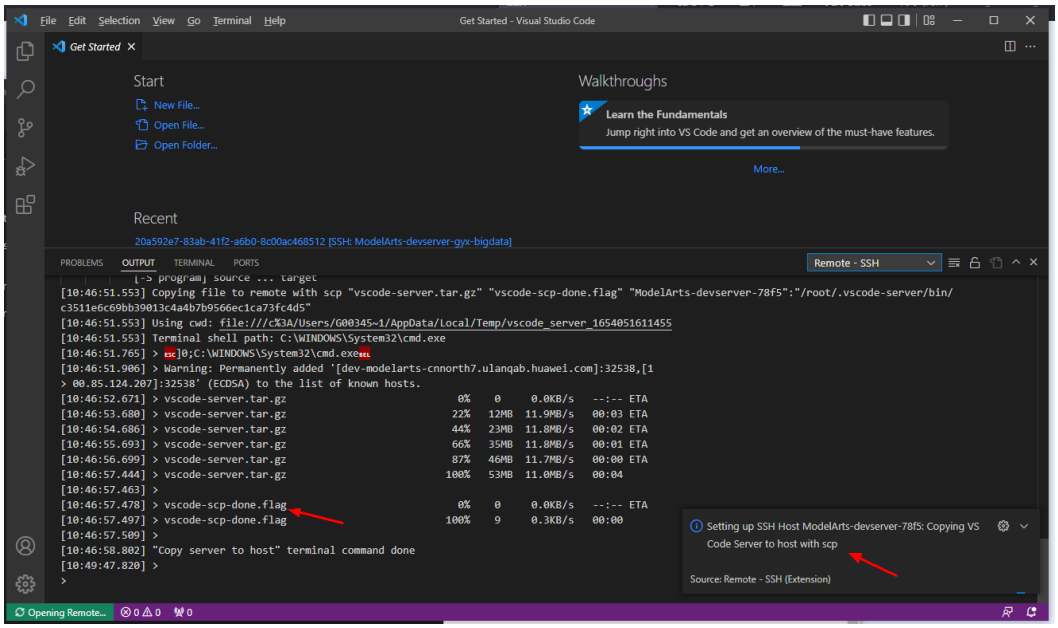
   执行下面命令，并指定commitId（注意：直接在Notebook的Terminal里执行，commit-id替换时去掉尖括号）

   ```
   commitId=<提交的ID码>
   mkdir -p /home/ma-user/.vscode-server/bin/$commitId
   tar -zxvf vscode-server-linux-x64.tar.gz -C /home/ma-user/.vscode-server/bin/$commitId  --strip=1
   chmod 750 -R /home/ma-user/.vscode-server/bin/$commitId
   ```

5. 关闭VS Code，重新从Notebook实例列表页面打开VS Code（注意：需要关闭本地vscode，否则可能会报多个安装进程正在运行中）。

3.5.6 连接远端开发环境时，一直处于"Setting up SSH Host xxx: Copying VS Code Server to host with scp"超过 10 分钟以上，如何解决?

问题现象



原因分析

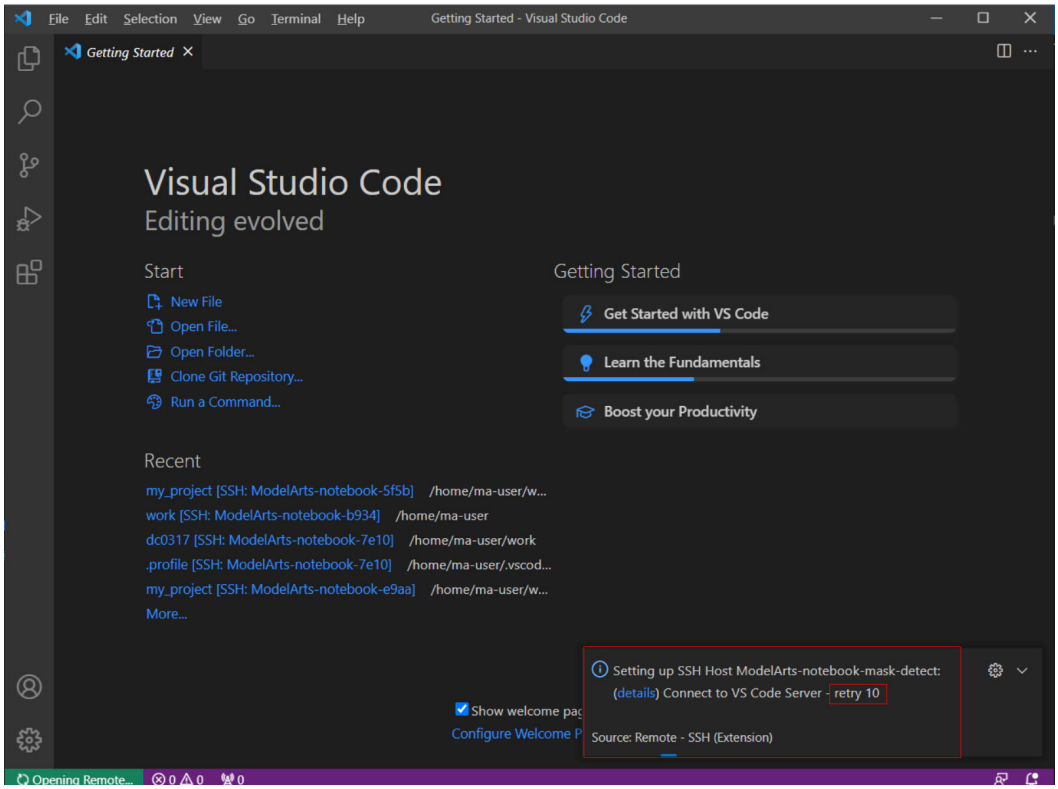
通过查看日志发现本地vscode-scp-done.flag显示成功上传，但远端未接收到。

解决方法

关闭VS Code所有窗口后，回到ModelArts控制台界面再次单击界面上的“VS Code接入”按钮。

3.5.7 远程连接处于 retry 状态如何解决?

问题现象



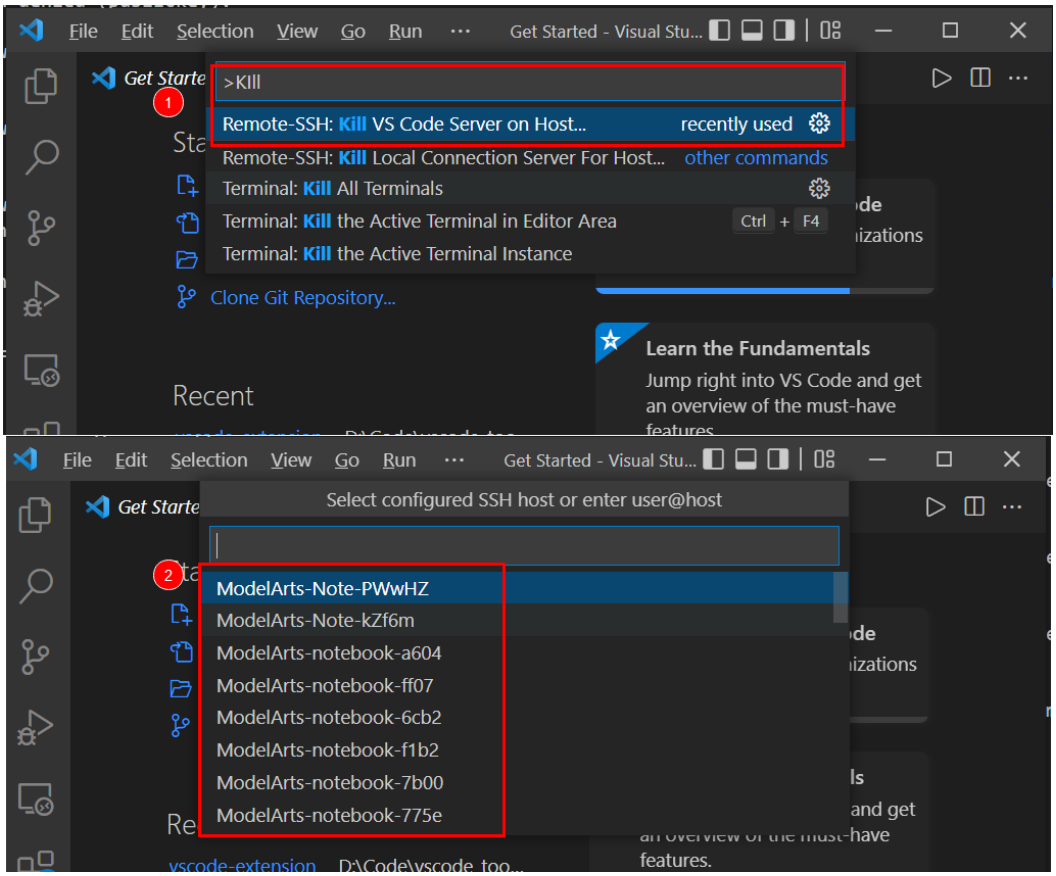
原因分析

之前下载VS Code server失败，有残留信息，导致本次无法下载。

解决方法

方法一（本地）：打开命令面板（Windows： Ctrl+Shift+P，macOS：Cmd+Shift+P），搜索“Kill VS Code Server on Host”，选择出问题的实例进行自动清除，然后重新进行连接。

图 3-8 清除异常的实例



方法二（远端）：在VS Code的Terminal中删除“/home/ma-user/.vscode-server/bin/”下正在使用的文件，然后重新进行连接。

```
ssh -tt -o StrictHostKeyChecking=no -i ${IdentityFile} ${User}@${HostName} -p ${Port}
rm -rf /home/ma-user/.vscode-server/bin/
```

参数说明：

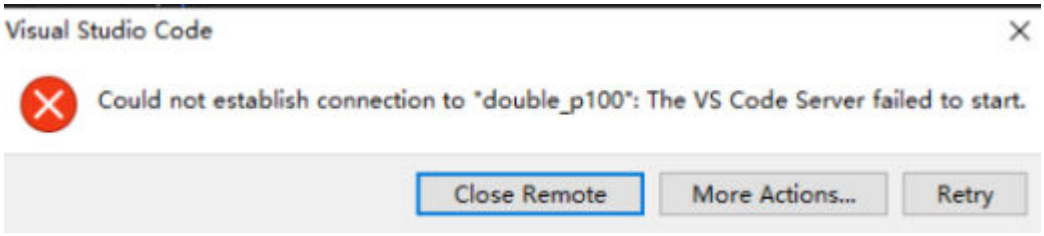
- IdentityFile：本地密钥路径
- User：用户名，例如：ma-user
- HostName：IP地址
- Port：端口号

说明

vscode-server相关问题也可以使用上述的解决方法。

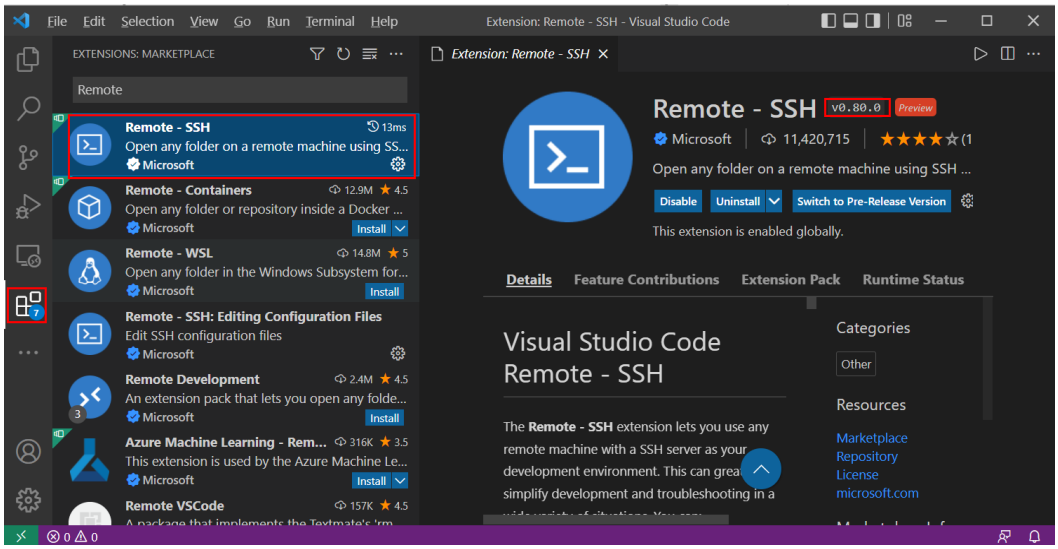
3.5.8 报错“The VS Code Server failed to start”如何解决?

问题现象



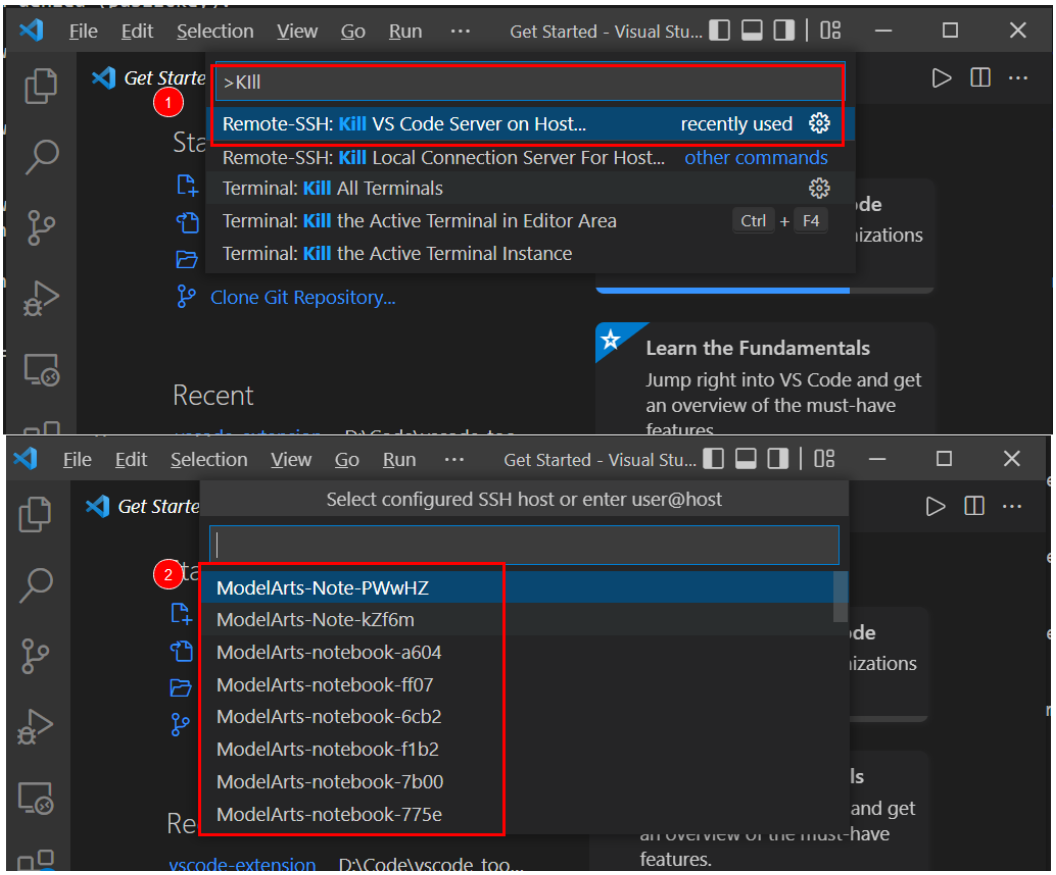
解决方法

步骤1 检查VS Code版本是否为1.78.2或更高版本，如果是，请查看Remote-SSH版本，如果低于v0.76.1，请升级Remote-SSH。



步骤2 打开命令面板（Windows： Ctrl+Shift+P，macOS：Cmd+Shift+P），搜索“Kill VS Code Server on Host”，选择出问题的实例进行自动清除，然后重新进行连接。

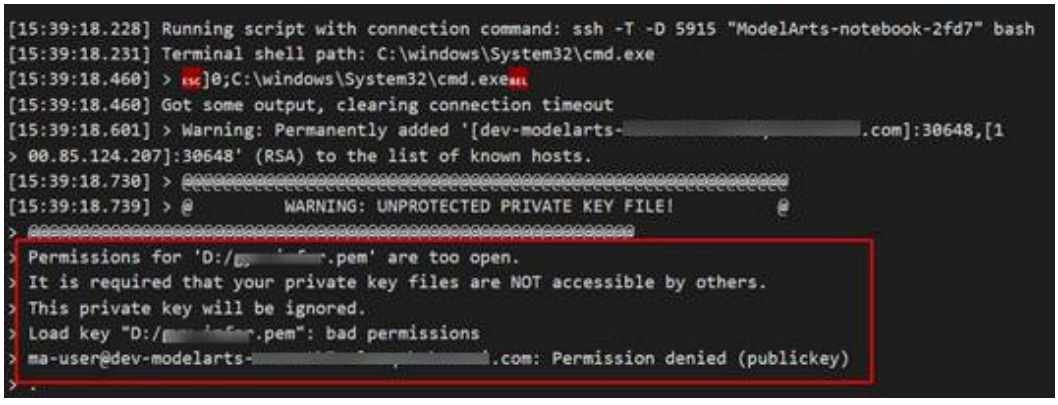
图 3-9 清除异常的实例



----结束

3.5.9 报错“Permissions for 'x:/xxx.pem' are too open”如何解决?

问题现象



原因分析

原因分析一：密钥文件未放在指定路径，详情请参考**安全限制**或**VS Code文档**。请参考解决方法一处理。

原因分析二：当操作系统为macOS/Linux时，可能是密钥文件或放置密钥的文件夹权限问题，请参考解决方法二处理。

解决方法

解决方法一：

请将密钥放在如下路径或其子路径下：

Windows：C:\Users\{{user}}

macOS/Linux： Users/{{user}}

解决方法二：

请**检查文件和文件夹权限**。

Local SSH file and folder permissions

macOS / Linux:

On your local machine, make sure the following permissions are set:

Folder / File	Permissions
`.ssh` in your user folder	`chmod 700 ~/.ssh`
`.ssh/config` in your user folder	`chmod 600 ~/.ssh/config`
`.ssh/id_rsa.pub` in your user folder	`chmod 600 ~/.ssh/id_rsa.pub`
Any other key file	`chmod 600 /path/to/key/file`

Windows:

The specific expected permissions can vary depending on the exact SSH implementation you are using. We recommend using the out of box Windows 10 OpenSSH Client.

In this case, make sure that all of the files in the `.ssh` folder for your remote user on the SSH host is owned by you and no other user has permissions to access it. See the Windows OpenSSH wiki for details.

For all other clients, consult your client's documentation for what the implementation expects.

3.5.10 报错“Bad owner or permissions on C:\Users\Administrator/.ssh/config”如何解决?

问题现象

VS Code连接开发环境时报错“Bad owner or permissions on C:\Users\Administrator/.ssh/config”。

原因分析

文件夹“.ssh”的权限不仅是Windows当前用户拥有，或者当前用户权限不足，故修改权限即可。

解决方案

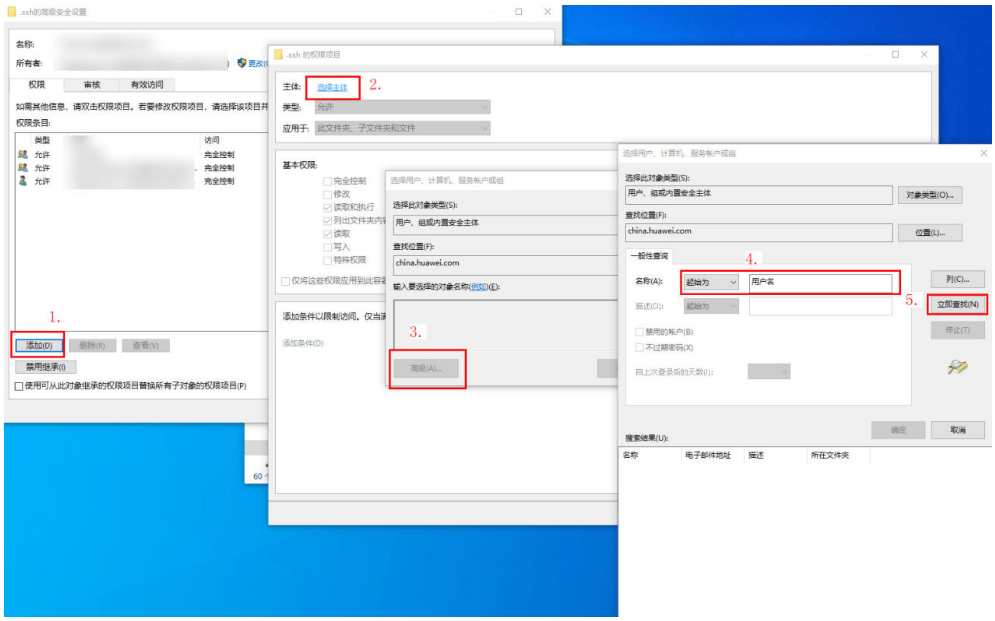
1. 找到.ssh文件夹。一般位于“C:\Users”，例如“C:\Users\xxx”。

   说明

   “C:\Users”目录下的文件名必须和Windows登录用户名完全一致。

2. 右键单击.ssh文件夹，选择“属性”。然后单击“安全”页签。
3. 单击“高级”，在弹出的高级安全设置界面单击“禁用继承”， 在弹出的“阻止继承”窗口单击“从此对象中删除所有继承的权限”。此时所有用户都将被删除。
4. 添加所有者：在同一窗口中，单击“添加”，在弹出的新窗口中，单击“主体”后面的“选择主体”，弹出“选择用户，计算机，服务账户或组”窗口，单击“高级”，输入用户名， 单击“立即查找”按钮，显示用户搜索结果列表。 选择您的用户账户，然后单击“确定”（大约四个窗口）以关闭所有窗口。

   图 3-10 添加所有者

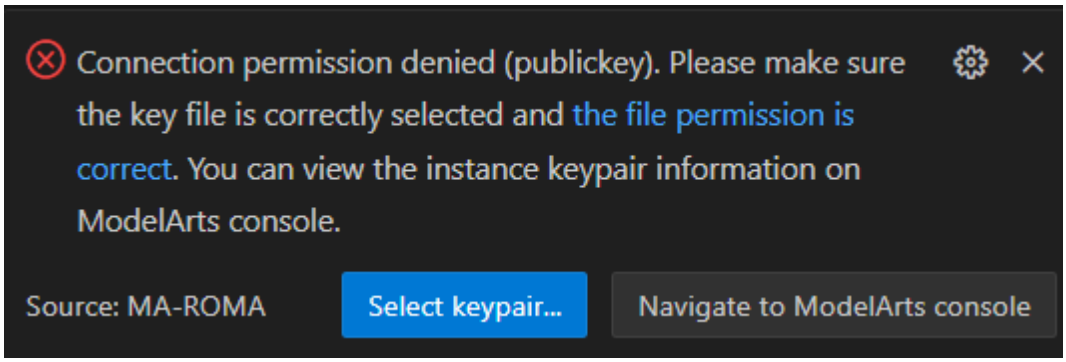
   

5. 完成所有操作后，再次关闭并打开VS Code并尝试连接到远程SSH主机。备注：此时密钥需放到.ssh文件夹中。

3.5.11 报错“Connection permission denied (publickey)”如何解决

问题现象

VS Code连接开发环境时报错“Connection permission denied (publickey). Please make sure the key file is correctly selected and the file permission is correct. You can view the instance keypair information on ModelArts console.”



原因分析

可能是密钥文件或放置密钥的文件夹权限问题，密钥不正确等，请按以下步骤排查。

解决方案

1. 排查/home/ma-user权限，建议将该目录权限设置为755或750，权限不能过于宽松，以保证用户隔离和安全。修改方法如下。
   ```
   chomd 755 /home/ma-user
   chomd 750 /home/ma-user
   ```
2. 排查密钥是否是和实例绑定的一致。
   a. 停止实例，进入实例详情页。
   b. 更新密钥：单击“认证”旁边的编辑按钮，然后单击“立即创建”创建并选择新密钥。
   c. 重新使用VS Code连接实例，选择新创建的密钥。

3.5.12 报错“ssh: connect to host xxx.pem port xxxxx: Connection refused”如何解决?

问题现象

原因分析

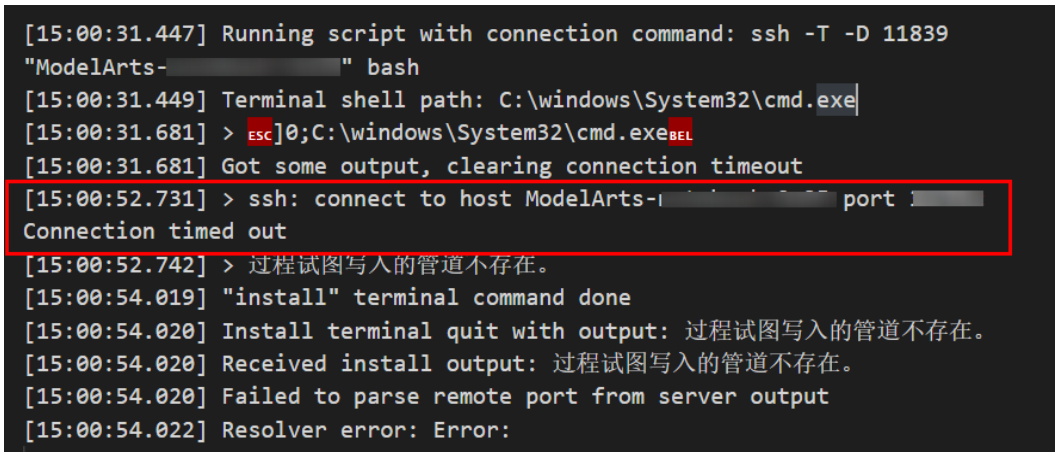
实例处于非运行状态。

解决方法

请前往ModelArts控制台查看实例是否处于运行状态，如果实例已停止，请执行启动操作，如果实例处于其他状态比如“错误”，请尝试先执行停止然后执行启动操作。待实例变为“运行中”后，再次执行远程连接。

3.5.13 报错"ssh: connect to host ModelArts-xxx port xxx: Connection timed out"如何解决?

问题现象



原因分析

原因分析一：实例配置的白名单IP与本地网络访问IP不符。

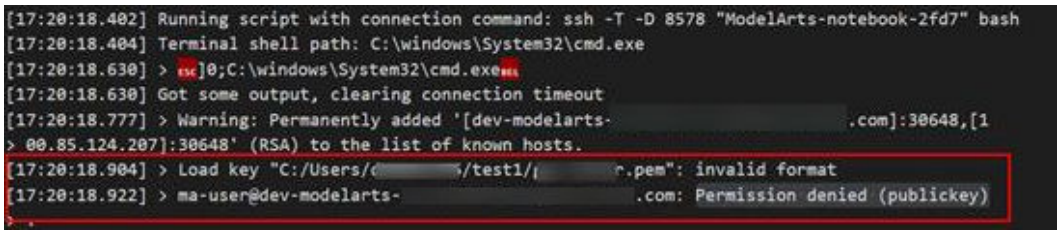
解决方法：请**修改白名单**为本地网络访问IP或者去掉白名单配置。

原因分析二：本地网络不通。

解决方法：检查本地网络以及网络限制。

3.5.14 报错“Load key "C:/Users/xx/test1/xxx.pem": invalid format”如何解决?

问题现象



原因分析

密钥文件内容不正确或格式不正确。

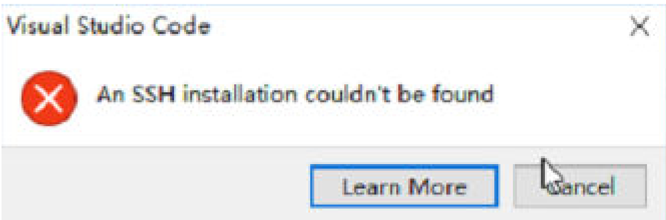
解决方法

请使用正确的密钥文件进行远程访问，如果本地没有正确的密钥文件或文件已损坏，可以尝试：

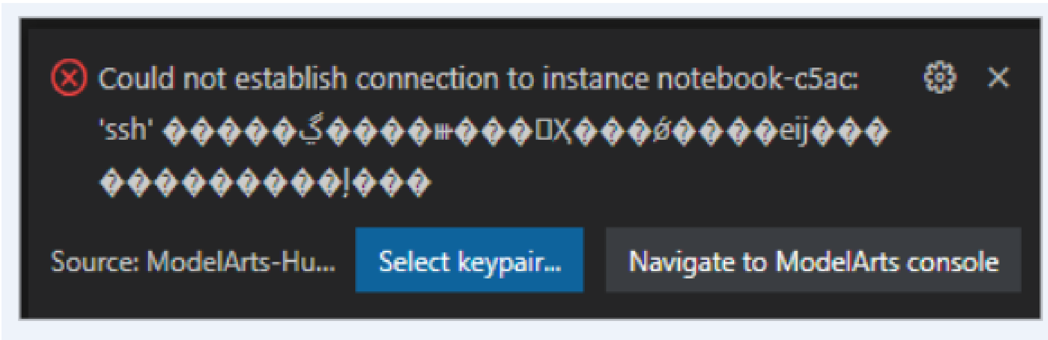
1. 登录控制台，搜索“数据加密服务 DEW”，选择“密钥对管理 > 账号密钥对”页签，查看并下载正确的密钥文件。
2. 如果密钥不支持下载且已无法找到之前下载的密钥，建议创建新的开发环境实例并创建新的密钥文件。

3.5.15 报错“An SSH installation couldn't be found”或者“Could not establish connection to instance xxx: 'ssh' ...”如何解决?

问题现象



或



VS Code连接Notebook一直提示选择证书，且提示信息除标题外，都是乱码。选择证书后，如上图所示仍然没有反应且无法进行连接。

原因分析

当前环境未装OpenSSH或者OpenSSH未安装在默认路径下，详情请参考**VS Code文档**。

解决方法

- 如果当前环境未安装OpenSSH，请**下载并安装OpenSSH**。

Installing a supported SSH client

OS	Instructions
Windows 10 1803+ / Server 2016/2019 1803+	Install the Windows OpenSSH Client.
Earlier Windows	Install Git for Windows.
macOS	Comes pre-installed.
Debian/Ubuntu	Run `sudo apt-get install openssh-client`
RHEL / Fedora / CentOS	Run `sudo yum install openssh-clients`

VS Code will look for the `ssh` command in the PATH. Failing that, on Windows it will attempt to find `ssh.exe` in the default Git for Windows install path. You can also specifically tell VS Code where to find the SSH client by adding the `remote.SSH.path` property to `settings.json`.

当通过“可选功能”未能成功安装时，请手动**下载OpenSSH安装包**，然后执行以下步骤：

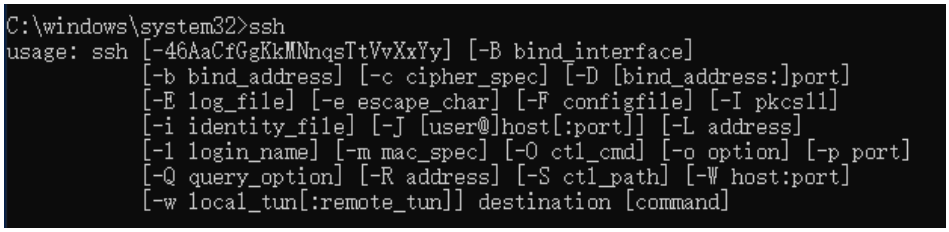
步骤1 下载zip包并解压放入“C:\Windows\System32”。

步骤2 以管理员身份打开CMD，在“C:\Windows\System32\OpenSSH-xx”目录下，执行以下命令：

```
powershell.exe -ExecutionPolicy Bypass -File install-sshd.ps1
```

步骤3 添加环境变量：将“C:\Program Files\OpenSSH-xx”（路径中包含ssh可执行exe文件）添加到环境系统变量中。

步骤4 重新打开CMD，并执行ssh，结果如下图即说明安装成功，如果还未装成功则执行**步骤5**和**步骤6**。



步骤5 OpenSSH默认端口为22端口，开启防火墙22端口号，在CMD执行以下命令：

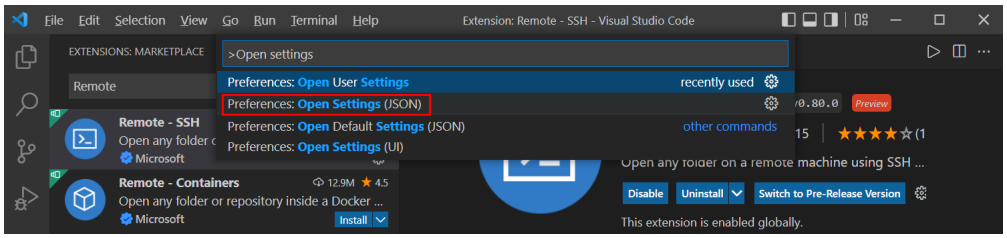
```
netsh advfirewall firewall add rule name=sshd dir=in action=allow protocol=TCP localport=22
```

步骤6 启动OpenSSH服务，在CMD执行以下命令：

```
Start-Service sshd
```

----结束

- 如果OpenSSH未安装在默认路径下，打开命令面板（Windows： Ctrl+Shift+P，macOS：Cmd+Shift+P），
  搜索“Open settings”。

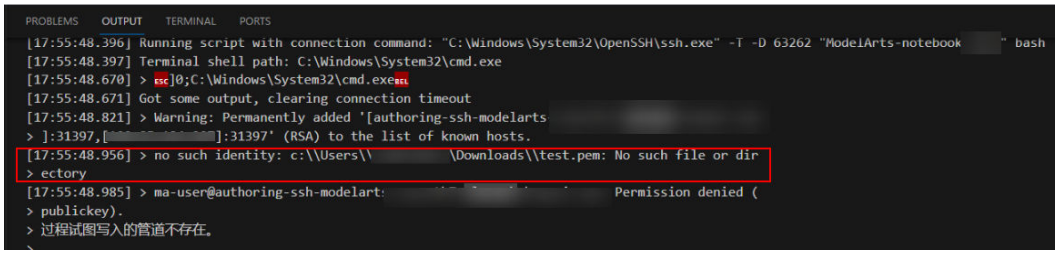


然后将remote.SSH.path属性添加到settings.json中，例如："remote.SSH.path": "*本地OpenSSH的安装路径*"

```
{
    "extensions.autoCheckUpdates": false,
    "extensions.autoUpdate": false,
    "remote.SSH.remotePlatform": {
        "ModelArts-notebook-          ": "linux"
    },
    "remote.SSH.path": "D:/OpenSSH-Win64/ssh.exe"
}
```

3.5.16 报错“no such identity: C:/Users/xx /test.pem: No such file or directory”如何解决?

问题现象



原因分析

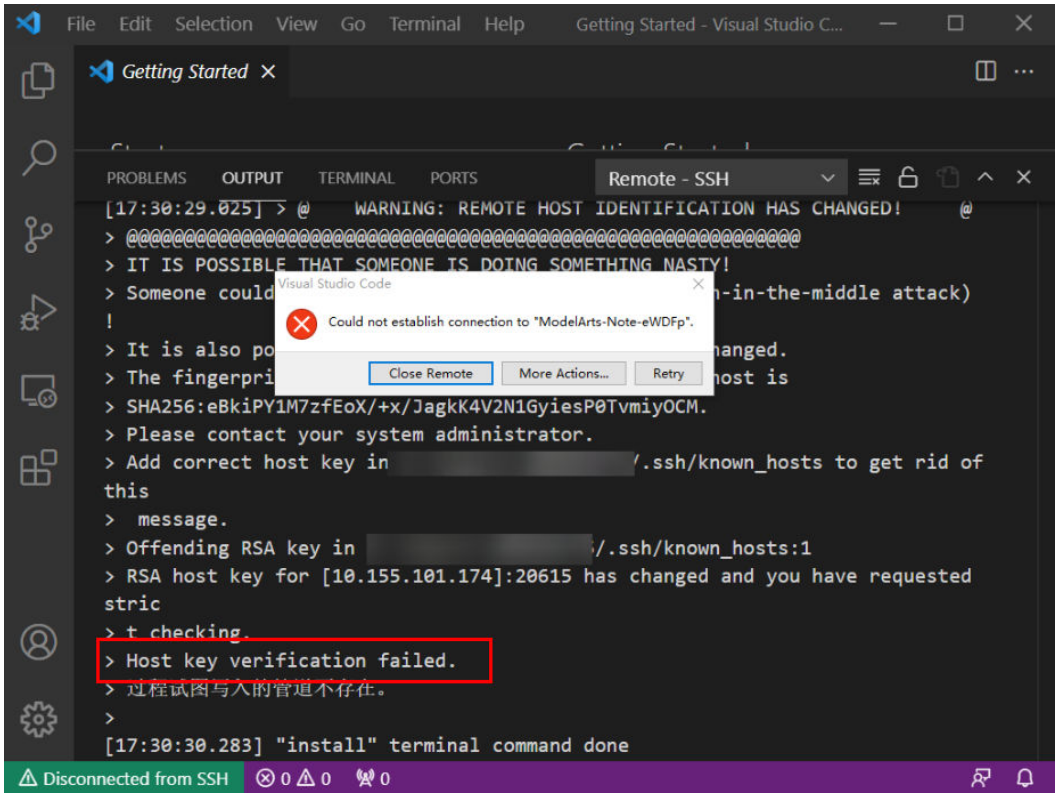
密钥文件不存在于该路径下，或者该路径下密钥文件名被修改。

解决方法

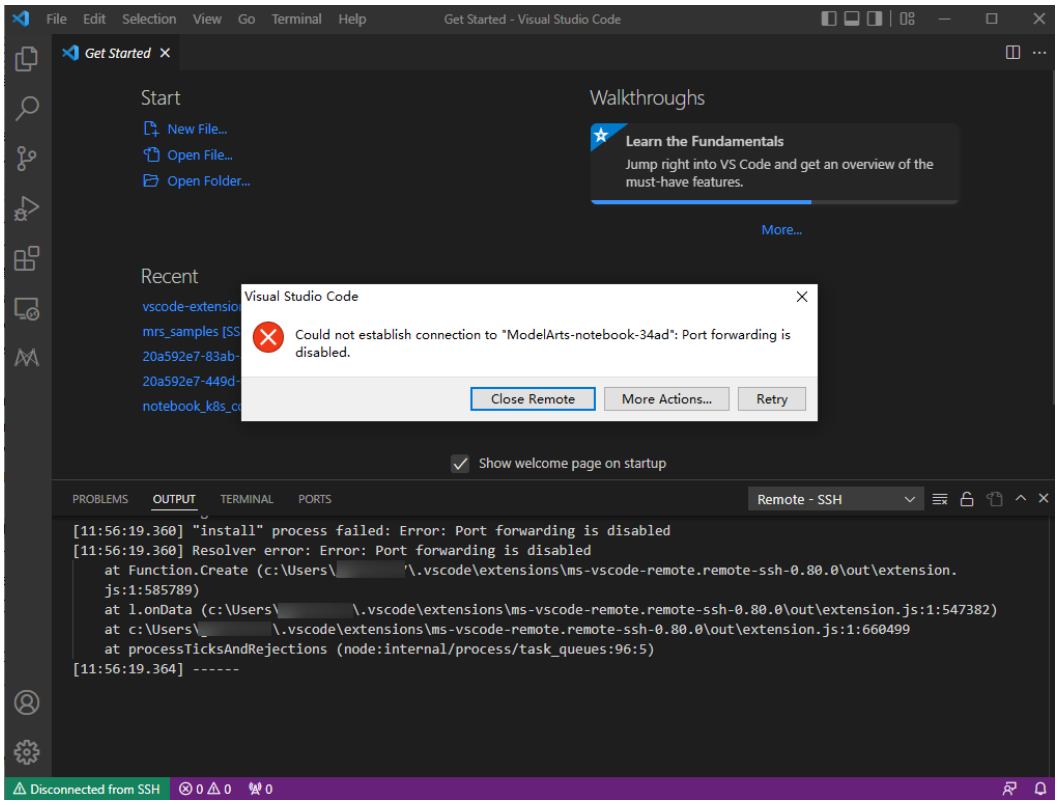
重新选择密钥路径。

3.5.17 报错“Host key verification failed.'或者'Port forwarding is disabled.”如何解决?

问题现象



或



原因分析

Notebook实例重新启动后，公钥发生变化，OpenSSH核对公钥发出警告。

解决方法

- 在VS Code中使用命令方式进行远程连接时，增加参数"-o StrictHostKeyChecking=no"

  ```
  ssh -tt -o StrictHostKeyChecking=no -i ${IdentityFile} ${User}@${HostName} -p ${Port}
  ```

  参数说明：

  - IdentityFile：本地密钥路径
  - User：用户名，例如：ma-user
  - HostName：IP地址
  - Port：端口号

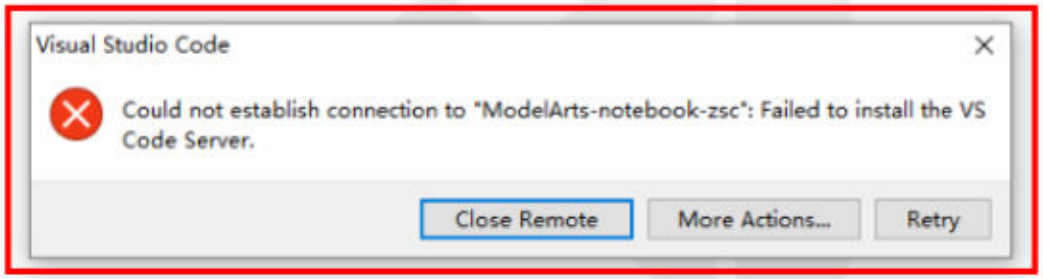
- 在VS Code中手工配置远程连接时，在本地的ssh config文件中增加配置参数“StrictHostKeyChecking no”和“UserKnownHostsFile=/dev/null”

  ```
  Host xxx
    HostName x.x.x.x #IP地址
    Port 22522
    User ma-user
    IdentityFile C:/Users/my.pem
    StrictHostKeyChecking no
    UserKnownHostsFile=/dev/null
    ForwardAgent yes
  ```

提示：增加参数后SSH登录时会忽略known_hosts文件，有安全风险。

3.5.18 报错“Failed to install the VS Code Server.”或“tar: Error is not recoverable: exiting now.”如何解决?

问题现象



或

```
[17:53:24.382] > vscode-scp-done.flag                          100%    9     0.2KB/s   00:00
[17:53:24.756] > Found flag and server on host
[17:53:24.765] > d3aeabcaa9c5%%2%%
> tar --version:
[17:53:24.789] > tar (GNU tar) 1.30
> Copyright (C) 2017 Free Software Foundation, Inc.
> License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>.
> This is free software: you are free to change and redistribute it.
> There is NO WARRANTY, to the extent permitted by law.
>
> Written by John Gilmore and Jay Fenlason.
[17:53:24.796] > tar: This does not look like a tar archive
>
> gzip: stdin: unexpected end of file
> tar: Child returned status 1
> tar: Error is not recoverable: exiting now
[17:53:24.804] >
> ERROR: tar exited with non-0 exit code: 0
> Already attempted local download, failing
> d3aeabcaa9c5: start
> exitCode==37==
```

原因分析

可能为/home/ma-user/work磁盘空间不足。

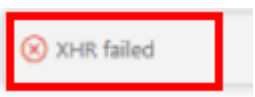
解决方法

删除/home/ma-user/work路径下无用文件。

3.5.19 VS Code 连接远端 Notebook 时报错“XHR failed”

问题现象

VS Code连接远端Notebook时报错“XHR failed”。

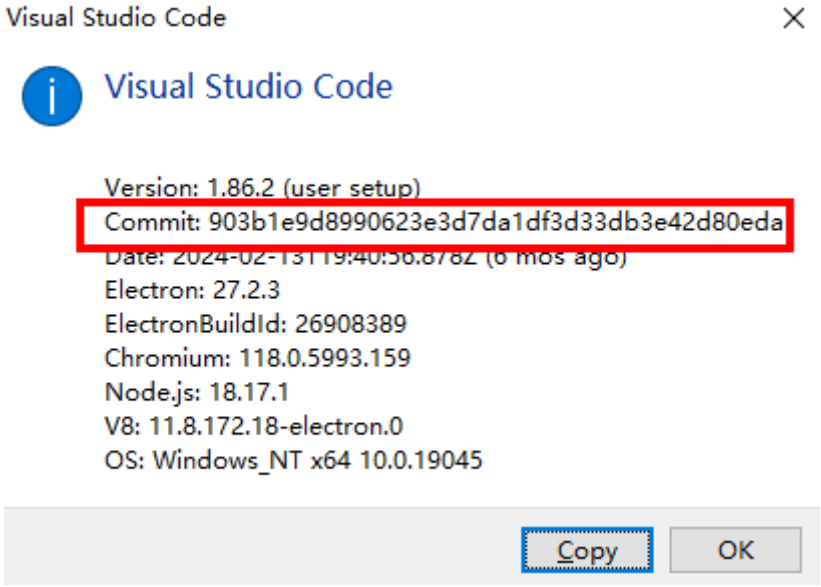


原因分析

可能是所在环境的网络有问题，无法自动下载VS Code Server，请手动安装。

解决方法

1. 打开VS Code，选择“Help>About”，并记下“Commit”的ID码。



2. 确认创建Notebook实例使用的镜像的系统架构，可以在Notebook中打开Terminal，通过命令**uname -m**查看。
3. 下载对应版本的vscode-server，根据Commit码和Notebook实例镜像架构下载。

   说明

   如果下载报错“Not Found”，请下载别的版本VS Code重新在本地安装，目前推荐:Vscode-1.86.2。

   - 如果实例的架构是x86_64的，通过下面的链接，手动修改Commit码（Commit码替换时去掉尖括号），使用浏览器下载vscode-server-linux-x64.tar.gz文件。

     ```
     https://update.code.visualstudio.com/commit:<Commit码>/server-linux-x64/stable
     ```

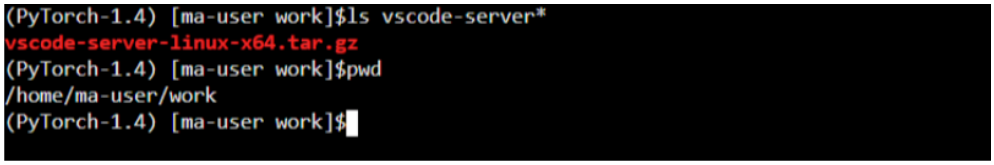
   - 如果实例的架构是aarch的，通过下面的链接，手动修改comment-id（commit-id替换时去掉尖括号），使用浏览器下载vscode-server-linux-arm64.tar.gz文件。下载完成后，将下载的vscode-server-linux-arm64.tar.gz文件重命名为“vscode-server-linux-x64.tar.gz”。

     ```
     https://update.code.visualstudio.com/commit:<提交的ID码>/server-linux-arm64/stable
     ```

   例如：commit-id是863d2581ecda6849923a2118d93a088b0745d9d6，os架构是x86_64，修改链接为：

   ```
   https://update.code.visualstudio.com/commit:863d2581ecda6849923a2118d93a088b0745d9d6/server-linux-x64/stable
   ```

4. 将下载的vscode-server-linux-x64.tar.gz，上传到ModelArts实例的“/home/ma-user/work”目录下。



执行下面命令，并指定commitId（注意：直接在Notebook的Terminal里执行，commit-id替换时去掉尖括号）

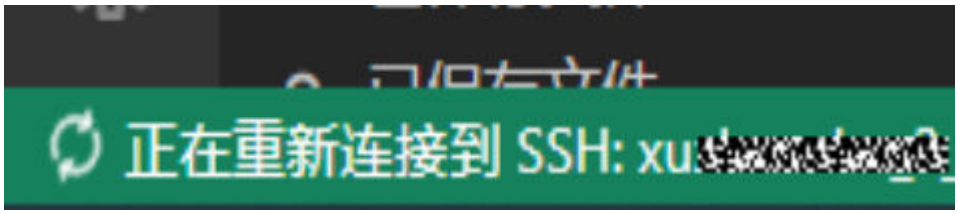
```
commitId=<提交的ID码>
mkdir -p /home/ma-user/.vscode-server/bin/$commitId
tar -zxvf vscode-server-linux-x64.tar.gz -C /home/ma-user/.vscode-server/bin/$commitId  --strip=1
chmod 750 -R /home/ma-user/.vscode-server/bin/$commitId
```

5. 关闭VS Code，重新从Notebook实例列表页面打开VS Code（注意：需要关闭本地vscode，否则可能会报多个安装进程正在运行中）。

3.5.20 VS Code 连接后长时间未操作，连接自动断开

问题现象

VS Code SSH连接后，长时间未操作，窗口未关闭，再次使用发现VS Code在重连环境，无弹窗报错。左下角显示如下图：



查看VS Code Remote-SSH日志发现，连接在大约2小时后断开了：

```
>
[21:32:39.136] Got some output, clearing connection timeout
[21:48:58.059] > 这会是正常的
[21:49:12.060] >
[22:40:58.740] >
>
[23:32:49.341] > Connection reset by 139.159.152.36 port 32528
>
```

原因分析

用户SSH交互操作停止后一段时间，防火墙对空闲连接进行了断开操作，SSH默认配置中不存在超时主动断连的动作，但是防火墙会关闭超时空闲连接（参考：http://bluebiu.com/blog/linux-ssh-session-alive.html），后台的实例运行是一直稳定的，重连即可再次连上。

解决方法

如果想保持长时间连接不断开，可以通过配置SSH定期发送通信消息，避免防火墙认为链路空闲而关闭。

- 客户端配置（用户可根据需要自行配置，不配置默认是不给服务端发心跳包），如图1，图2所示。

图 3-11 打开 VS Code ssh config 配置文件

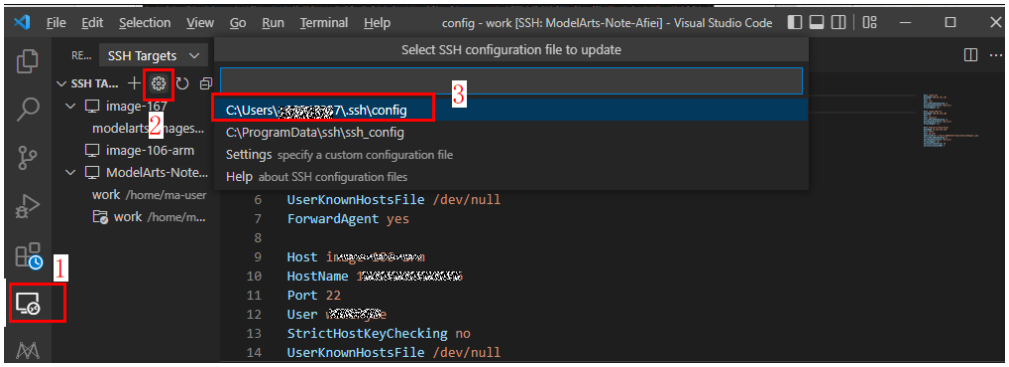
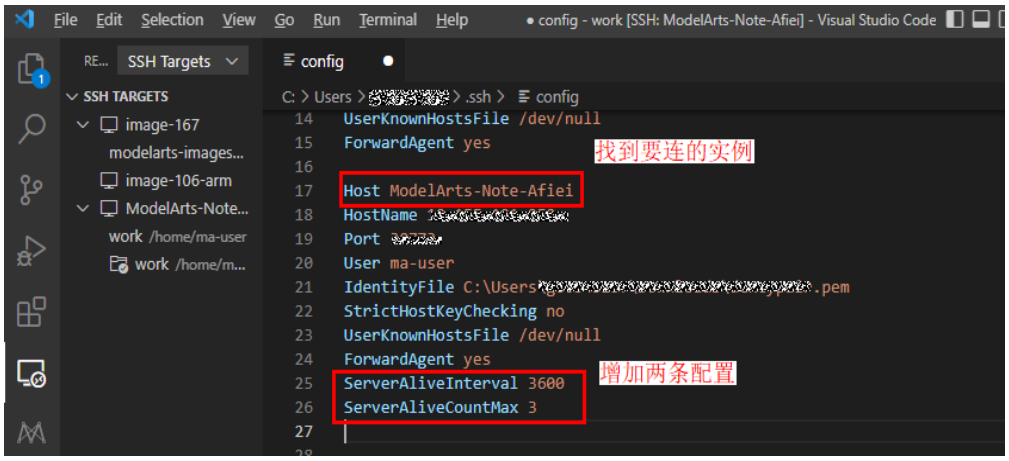


图 3-12 增加配置信息



配置信息示例如下：

```
Host ModelArts-xx
    ......
    ServerAliveInterval 3600  # 增加这个配置，单位是秒，每1h向服务端主动发个包
    ServerAliveCountMax 3  # 增加这个配置，3次发包均无响应会断开连接
```

比如防火墙配置是2小时空闲就关闭连接，那客户端配置ServerAliveInterval小于2小时（比如1小时），就可以避免防火墙将连接断开。

- 服务器端配置（Notebook当前已经配置，24h应该是长于防火墙的断连时间配置，该配置无需用户手工修改，写在这里仅是帮助理解ssh配置原理）配置文件路径：/home/ma-user/.ssh/etc/sshd_config

```
/modelarts/authoring(MindSpore) [ma-user work]$cat /home/ma-user/.ssh/etc/sshd_config |grep Client
ClientAliveInterval 1440m
ClientAliveCountMax 3
```

每24h向client端主动发个包，3次发包均无响应会断开连接

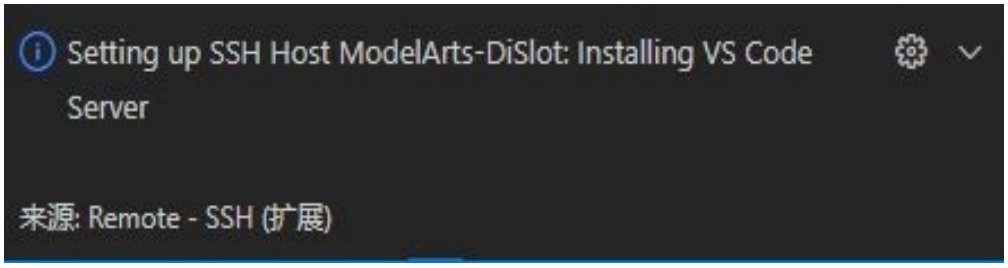
参考：https://unix.stackexchange.com/questions/3026/what-do-options-serveraliveinterval-and-clientaliveinterval-in-sshd-config-d

- 对于业务有影响的需要进行长链接保持的场景，尽量将日志写在单独的日志文件中，将脚本后台运行，例如：

```
nohup train.sh > output.log 2>&1 & tail -f output.log
```

3.5.21 VS Code 自动升级后，导致远程连接时间过长

问题现象



原因分析

由于VS Code自动升级，导致连接时需要重新下载新版vscode-server。

解决方法

禁止VS Code自动升级。单击左下角选择Settings项，搜索Update: Mode，将其设置为none。

图 3-13 打开 Settings

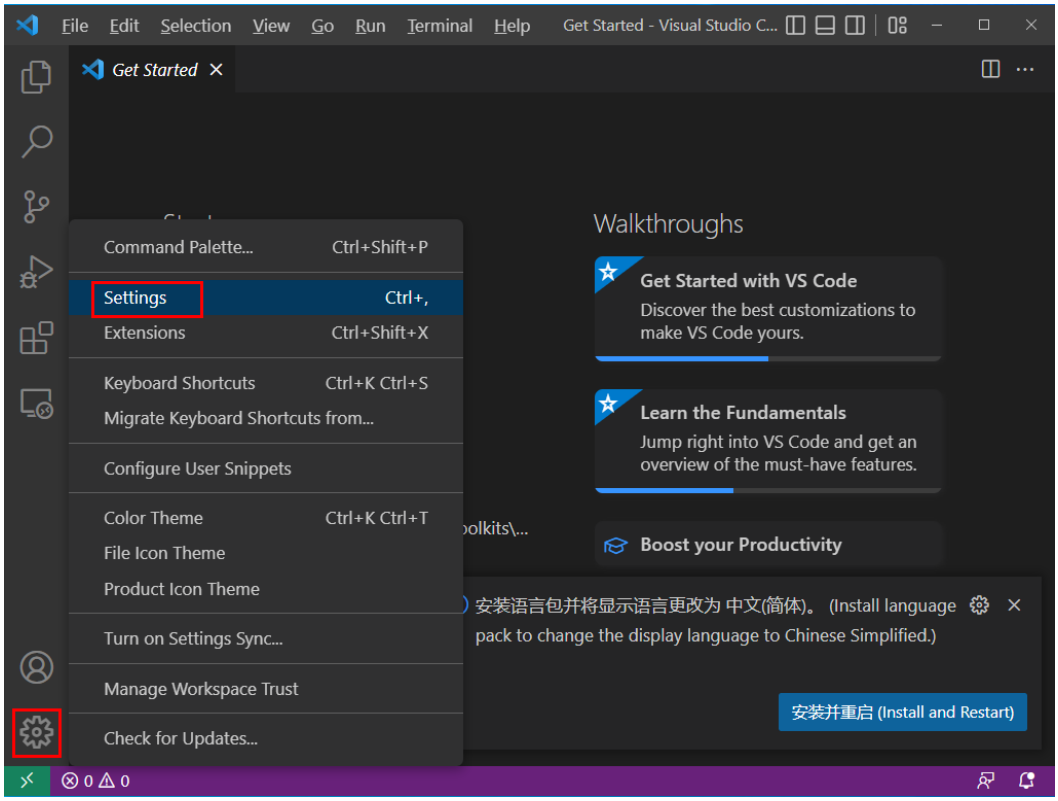
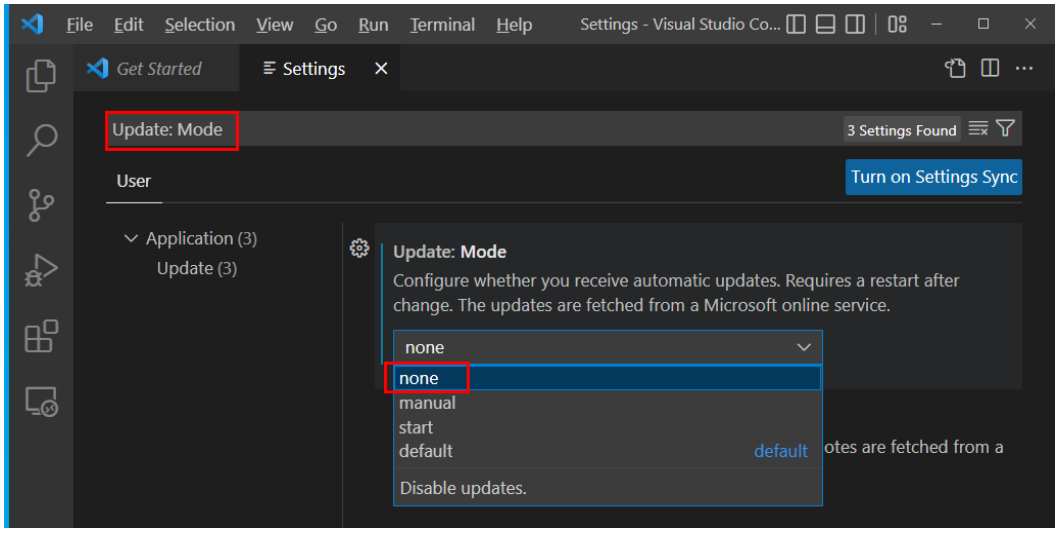
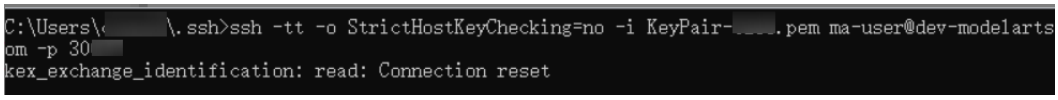


图 3-14 设置“Update: Mode”为“none”



3.5.22 使用 SSH 连接，报错“Connection reset”如何解决?

问题现象



原因分析

可能是用户网络限制原因。比如部分企业网络的SSH是默认屏蔽的。

解决方法

用户重新进行申请SSH权限。

3.5.23 使用 MobaXterm 工具 SSH 连接 Notebook 后，经常断开或卡顿，如何解决?

问题现象

MobaXterm成功连接到开发环境后，过一段时间会自动断开。

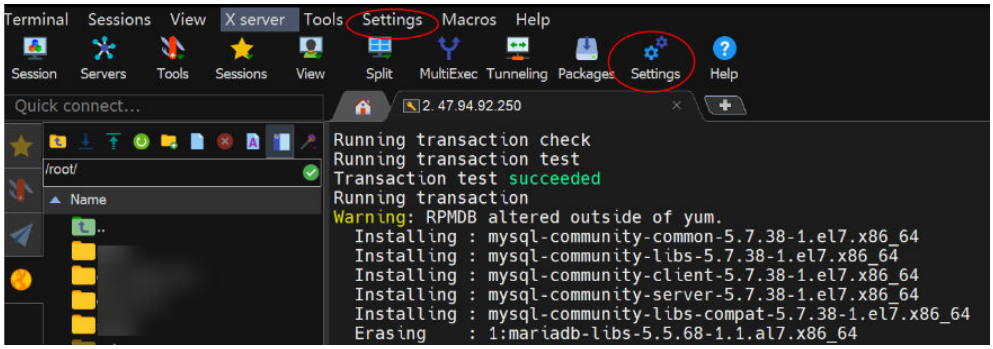
可能原因

配置MobaXterm工具时，没有勾选“SSH keepalive”或专业版MobaXterm工具的“Stop server after”时间设置太短。

解决方案

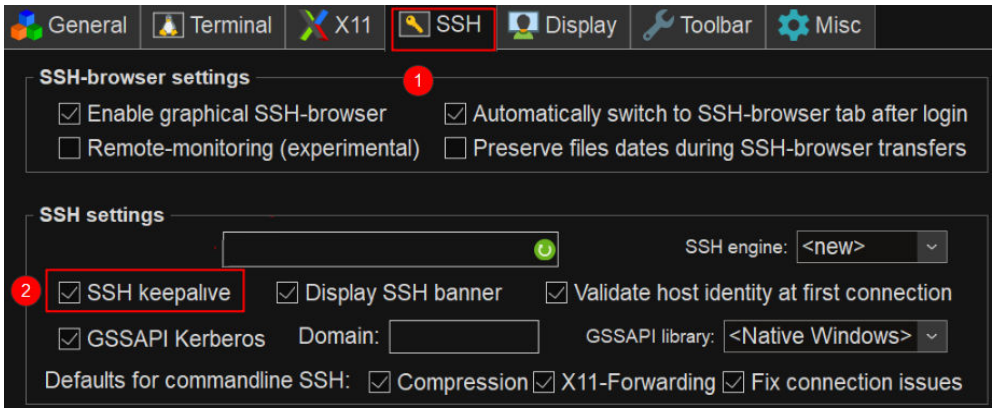
步骤1 打开MobaXterm，单击菜单栏“Settings”，如**图1 打开“Settings”**所示。

图 3-15 打开“Settings”



步骤2 在打开的“MobaXterm Configuration”配置页面，选择“SSH”选项卡，勾选“SSH keepalive”，如**图2 勾选“SSH keepalive”**所示。

图 3-16 勾选“SSH keepalive”

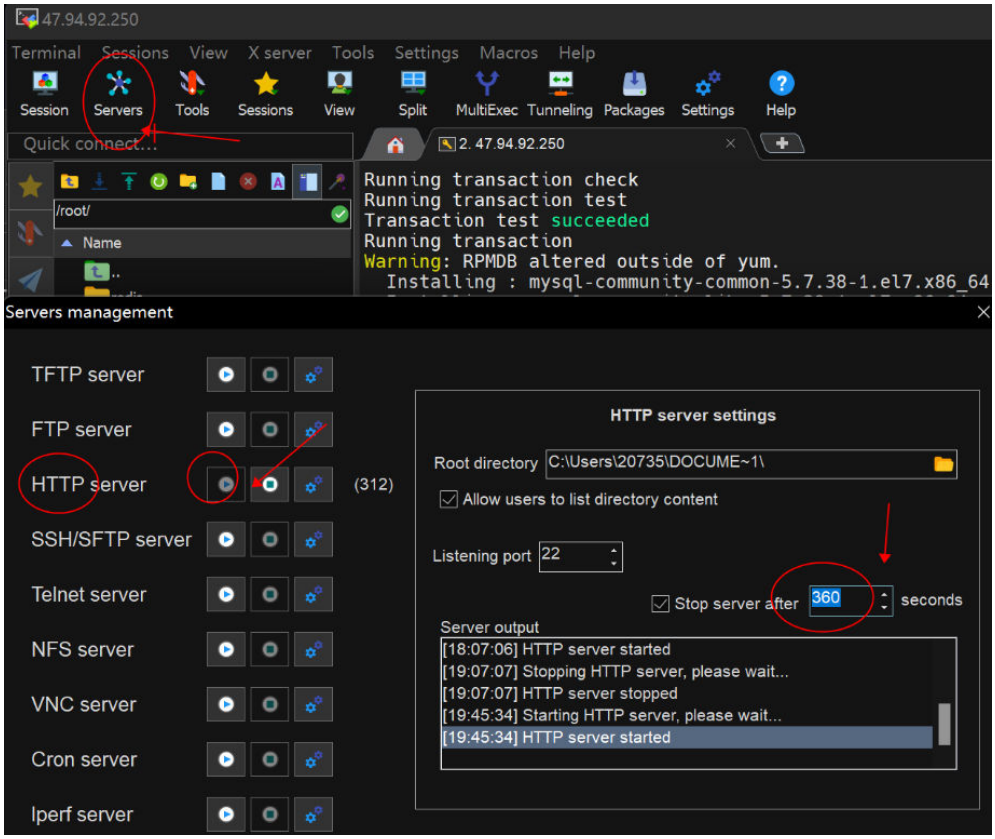


说明

如果使用的是专业版的MobaXterm工具，请执行**步骤3**。

步骤3 如果使用的是专业版的MobaXterm工具，请参考**图3 设置“Stop server after”**，此参数默认值为360s，将其设置为3600s或更大值。

图 3-17 设置“Stop server after”



----结束

3.5.24 VS Code 连接开发环境时报错 Missing GLIBC，Missing required dependencies

问题现象

VS Code连接开发环境时报错如下：

```
Warning: Missing GLIBC >= 2.28! from /lib/x86_64-linux-gnu/libc-2.27.so Error: Missing required dependencies. Please refer to our FAQ https://aka.ms/vscode-remote/faq/old-linux for additional information.
```

原因分析

该问题为用户使用VS Code 1.86版本软件导致的，需要用户使用较低版本的VS Code 。

解决方案

使用VS Code 1.85版本软件。下载链接：**https://code.visualstudio.com/updates/v1_85**。

3.5.25 使用 VSCode-huawei，报错：卸载了‘ms-vscode-remote.remot-sdh’，它被报告存在问题

问题现象

使用华为自研的VS Code软件时，报错“卸载了‘ms-vscode-remote.remot-sdh’，它被报告存在问题”。

原因分析

Remote - SSH只能在开源的VSCode软件中使用。

解决方案

推荐使用开源VS Code软件。

3.5.26 使用 VS Code 连接实例时，发现 VS Code 端的实例目录和云上目录不匹配

问题现象

用户使用VS Code连接实例时，发现VS Code端的实例目录和云上目录不匹配。

原因分析

实例连接错误，可能是配置文件写的不规范导致连接到别的实例。

解决方案

1. 检查用户.ssh配置文件（路径一般在“C:\Users\{User}\.ssh\config”下），检查每组配置文件是否规范：Host必须放在每组配置的第一行，作为每组配置的唯一ID。

   如下，第一组配置文件不规范将Host放到最后一行，用户要连的是下面这个Host ModelArts-Note-BmjiN实例，但SSH连到识别的是Host，错误地连到了Host ModelArts-Note-wZc6s这个实例。

```
HostName 10.155.119.26
Port 31251
User ma-user
IdentityFile ~\Downloads\history\202304-06\lewic-wly.pem
StrictHostKeyChecking no
UserKnownHostsFile /dev/null
ForwardAgent yes
Host ModelArts-Note-wZc6s

Host ModelArts-Note-BmjiN
HostName 10.155.119.26
Port 35338
User ma-user
IdentityFile ~\Downloads\history\202304-06\lewic-wly.pem
StrictHostKeyChecking no
UserKnownHostsFile /dev/null
ForwardAgent yes
```

2. 按ssh-config的标准写法更新配置，Host这里是每组配置的唯一标识，必填项且必须放在配置文件第一行。

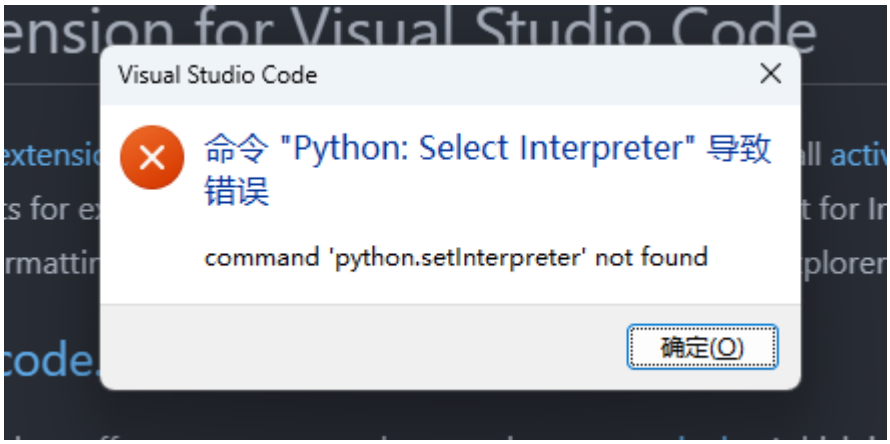
```
Host ModelArts-notebook-xxx
HostName authoring-ssh-modelarts-example.huawei.com
Port 31215
User ma-user
IdentityFile c:\Users\xxx\KeyPair-xxx.pem
StrictHostKeyChecking no
UserKnownHostsFile /dev/null
ForwardAgent yes
```

3.5.27 VSCode 远程连接时卡顿，或 Python 调试插件无法使用如何处理?

问题现象

VSCode远程连接Notebook时，单击“VS Code接入”跳转至连接界面时一直卡顿，或Python调试插件无法使用。

图 3-18 Python 调试插件错误



原因分析

该问题通常由VS Code安装了第三方中文插件引起。

解决方案

1. 卸载中文插件：如果安装了中文插件，建议先卸载。
2. 如果问题仍未解决，可以在VS Code官方社区查找相关解决方案或更新插件。

3.6 自定义镜像故障

3.6.1 Notebook 自定义镜像故障基础排查

当制作的自定义镜像使用出现故障时，请用户按照如下方法排查：

- 用户自定义镜像没有ma-user用户及ma-group用户组；
- 用户自定义镜像中/home/ma-user目录，属主和用户组不是ma-user和ma-group；
- 用户自定义镜像必须满足用户目录/home/ma-user权限为750，不能为其他权限；
- 用户自定义镜像使用远程SSH功能，OpenSSH版本要兼容或高于8.0；
- 用户制作的自定义镜像，在本地执行docker run启动，无法正常运行；
- 用户自行安装了Jupyterlab服务导致冲突的，需要用户本地使用Jupyterlab命令罗列出相关的静态文件路径，删除并且卸载镜像中的Jupyterlab服务；
- 用户自己业务占用了开发环境官方的8888、8889端口的，需要用户修改自己的进程端口号；
- 用户的镜像指定了PYTHONPATH、sys.path导致服务启动调用冲突的，需在实例启动后，再指定PYTHONPATH、sys.path；
- 用户使用了已开启sudo权限的专属池，使用自定义镜像时，sudo工具未安装或安装错误；
- 用户使用的cann、cuda环境有兼容性问题；
- 用户的docker镜像配置错误、网络或防火墙限制、镜像构建问题（文件权限、依赖缺失或构建命令错误）等原因导致的。

3.6.2 镜像保存时报错“there are processes in 'D' status, please check process status using 'ps -aux' and kill all the 'D' status processes”或“Buildimge,False,Error response from daemon，Cannot pause container xxx”如何解决?

问题现象

- 在Notebook里保存镜像时报错“there are processes in 'D' status, please check process status using 'ps -aux' and kill all the 'D' status processes”。
- 在Notebook里保存镜像时报错“Buildimge,False,Error response from daemon: Cannot pause container xxx”。

原因分析

执行镜像保存时，Notebook中存在状态为D的进程，会导致镜像保存失败。

解决方案

1. 在Terminal里执行**ps -aux**命令检查进程。

```
(PyTorch-1.8) [ma-user work]$ps -aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
ma-user        1  0.0  0.0   4532   392 ?        Ss   10:47   0:00 /modelarts/authoring/scrip
ma-user        8  0.0  0.0  22028  2196 ?        S    10:47   0:00 /bin/bash /modelarts/autho
ma-user      103  0.0  0.2 137000 76276 ?        SN   10:47   0:02 /modelarts/authoring/noteb
ma-user      115  0.0  0.0  13444   808 ?        S    10:47   0:00 /bin/bash /modelarts/autho
ma-user      116  0.0  0.0   7940   660 ?        S    10:47   0:00 tee /home/ma-user/log/note
ma-user      119  1.5  0.3 3800480 130936 ?      Sl   10:47   0:47 /modelarts/authoring/noteb
ma-user     3134  0.0  0.0  38536 18876 pts/0    SNs  10:58   0:00 /bin/bash -l
ma-user    11045  0.0  0.0   4388   392 pts/0    DN+  11:37   0:00 ./d_process
ma-user    11046  0.0  0.0   4388   392 pts/0    SN+  11:37   0:00 ./d_process
ma-user    11069  4.2  0.0  22148  2408 pts/1    SNs  11:37   0:00 /bin/bash -l
ma-user    11128  0.0  0.0   7936   656 ?        S    11:37   0:00 sleep 3
ma-user    11131  0.0  0.0  37796  1616 pts/1    RN+  11:37   0:00 ps -aux
(PyTorch-1.8) [ma-user work]$
```

2. 执行**kill -9 <pid>**命令将相关进程结束后，再次执行镜像保存即可。

3.6.3 镜像保存时报错“container size %dG is greater than threshold %dG”如何解决?

问题现象

在Notebook里保存镜像时报错“container size %dG is greater than threshold %dG”。

原因分析

Notebook容器当前的大小超过了阈值。

解决方案

需要减少容器大小。Notebook容器的大小分为两部分：镜像大小和容器中新安装文件的大小。因此有两种方法来解决该问题：

- 减少容器中新安装文件的大小
  a. 删除用户在Notebook新安装的内容，比如用户在Notebook中下载了很多文件，可以将这些文件删除。这种方法仅适用于除/home/ma-user/work和/cache目录外的其他目录，因为持久化存储的部分（home/ma-user/work目录的内容）不会保存在最终产生的容器镜像中、“/cache”目录下存储的是临时文件，不占用容器空间。
  b. 如果没有文件可以删除，或者不清楚哪些可以删除，那么可以使用相同的镜像重新创建一个Notebook，使用新建的Notebook时，注意减少软件包的安装或文件的下载等操作，也可以减少容器大小；
- 减少镜像文件的大小
  如果无法确认哪些包或文件可以不安装，那么可以选择一个较小的镜像来重建Notebook，然后在其中再安装需要的软件或文件。目前公共镜像中占用空间最小的是mindspore1.7.0-py3.7-ubuntu18.04。

3.6.4 保存镜像时报错“too many layers in your image”如何解决?

问题现象

保存镜像时报错“too many layers in your image”。

原因分析

用户创建Notebook时所选用的镜像是经过多次保存的自定义镜像或用户自行注册的镜像，基于该镜像所创建的Notebook已经无法再执行镜像保存的操作了。

解决方法

使用公共镜像或其他的自定义镜像来创建Notebook，完成镜像保存操作。

3.6.5 镜像保存时报错“The container size (xG) is greater than the threshold (25G)”如何解决?

问题现象

镜像保存时报错“The container size (30G) is greater than the threshold (25G)”，镜像创建失败。

原因分析

镜像保存本质是通过在资源集群节点上的agent中进行了docker commit，再配合一系列自动化操作来上传和更新管理数据等。每次Commit都会带来额外的一些开销，层数越多镜像越大，如果多次保存后就会有存储显示没那么大，但是镜像已经很大。镜像超大会导致加载的各种问题，所以这里做了限制。这种场景下，建议找到原始镜像重新构建环境进行保存。

解决方法

找到原始镜像重新构建环境。建议使用干净的基础镜像，最小化的安装运行依赖内容，并进行安装后的软件缓存清理，然后保存镜像。

3.6.6 镜像保存时报错“BuildImage,True,Commit successfully|PushImage,False,Task is running.”

问题现象

镜像保存时报错BuildImage,True,Commit successfully|PushImage,False,Task is running.

可能原因

镜像过大Push任务一直在运行，或实例节点有问题。

解决方法

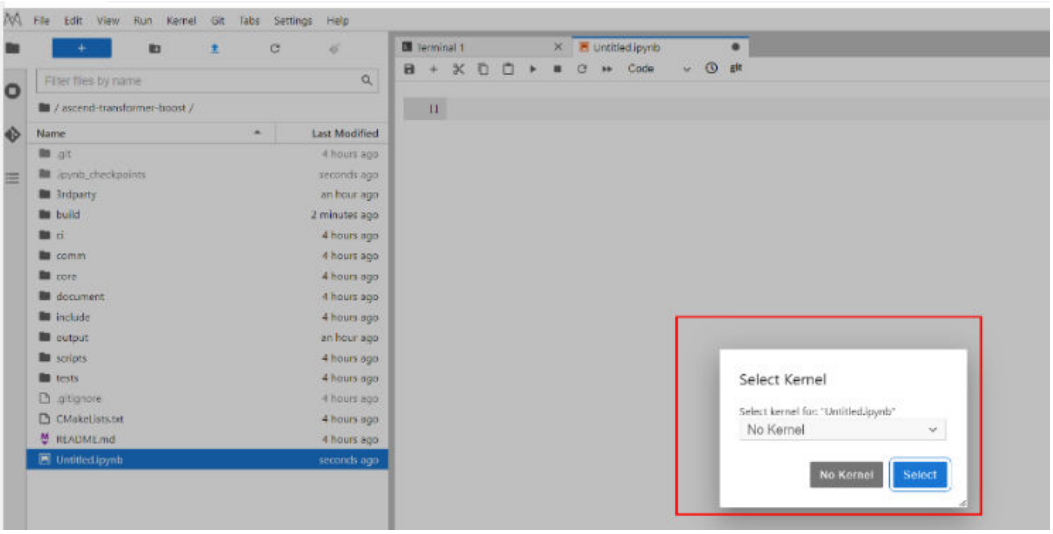
以对应租户的华为云账号登录SWR服务，查看镜像是否已经Push成功。

- 如果Push成功，请重新注册镜像。
- 如果未Push成功，联系SRE查看对应实例的节点是否有问题。

3.6.7 使用自定义镜像创建 Notebook 后打开没有 kernel

问题现象

使用自定义镜像创建实例启动后，打开JupyterLab>新建Notebook，选不到kernel。



原因分析

自定义镜像的python环境没有注册。

解决方案

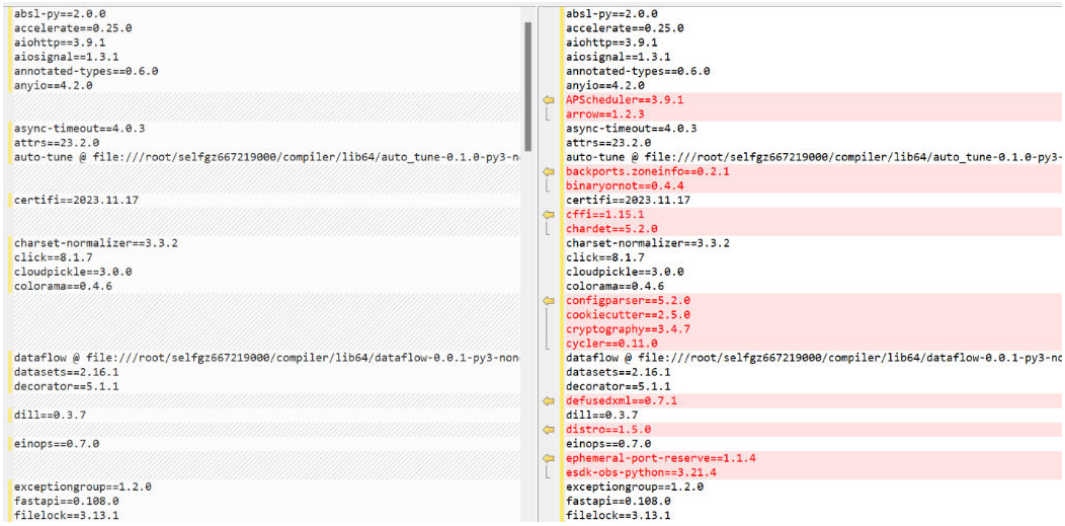
1. 在Terminal里执行命令排查实例存在几个Conda环境。
   ```
   conda env list
   ```
2. 执行如下命令分别切换到对应环境查看是否有ipykernel包。
   ```
   conda activate base # base替换为实际使用的python环境
   pip show ipykernel
   ```
3. 对应conda环境没有ipykernel，直接**在Notebook中添加自定义IPython Kernel**安装。

   ```
   [ma-user ~]$pip show ipykernel
   WARNING: Package(s) not found: ipykernel
   [ma-user ~]$
   ```

3.6.8 用户自定义镜像自建的 conda 环境会查到一些额外的包，影响用户程序，如何解决?

问题现象

用户的自定义镜像运行在Notebook里会查到一些额外的pip包。如下图所示，左侧为自定义镜像运行在本地环境，右侧为运行在Notebook里。



可能原因

Notebook自带moxing、modelart-sdk等功能，会将这些包嵌入到用户Conda环境。

解决方案

如果不需要使用moxing、sdk等功能，可以暂时删除modelarts.pth文件。

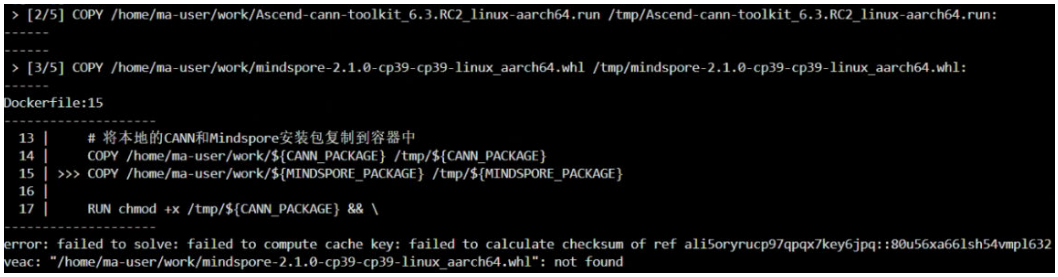
1. 执行如下命令在用户运行的Conda环境下查找modelarts.pth。
   ```
   # /home/ma-user/anaconda3指用户的python环境
   find /home/ma-user/anaconda3 -name modelarts.pth
   ```
2. 执行如下命令删除用户使用的python环境中的modelarts.pth文件。
   ```
   # /xxx/modelarts.pth 指用户通过第一步查出来的文件路径
   rm -rf /xxx/modelarts.pth
   ```

3.6.9 用户使用 ma-cli 制作自定义镜像失败，报错文件不存在（not found）

问题现象

用户使用ma-cli制作自定义镜像失败，报错文件目录不存在。

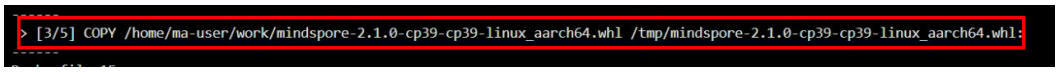
图 3-19 报错 xxx not found



原因分析

复制的文件需要放在Dockerfile同级文件夹或者子目录中，不能放在Dockerfile上层目录。

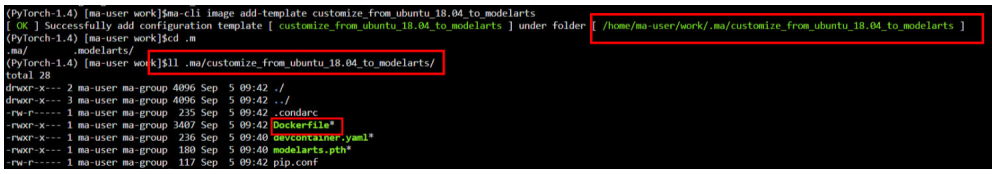
图 3-20 Dockerfile 复制文件路径错误



解决方案

1. 查看用户Dockerfile中的COPY命令中的文件的路径。将要复制的文件放到Dockerfile同级目录或子目录中，如图，Dockerfile在“./.ma/customize_from_ubuntu_18.04_to_modelarts/路径下”，需要将文件放到“/home/ma-user/work/.ma/customize_from_ubuntu_18.04_to_modelarts”下。

   图 3-21 查询 Dockerfile 的路径

2. Dockerfile命令修改为相对路径，举例如下：

   ```
   COPY ./mindspore-2.1.0-cp39-cp39-linux_aarch64.whl /tmp/mindspore-2.1.0-cp39-cp39 -
   linux_aarch64.whl
   ```

3.6.10 用户使用 torch 报错 Unexpected error from cudaGetDeviceCount

问题现象

在Notebook执行兼容gpu的脚本时报错不兼容，但是通过nvcc --version排查显示是兼容。

```
import torch
import sys
print('A', sys.version)
print('B', torch.__version__)
print('C', torch.cuda.is_available())
print('D', torch.backends.cudnn.enabled)
device = torch.device('cuda')
print('E', torch.cuda.get_device_properties(device))
print('F', torch.tensor([1.0, 2.0]).cuda())
```

报错如下

```
Traceback (most recent call last):
File "test.py", line 8, in <module>
print('E', torch.cuda.get_device_properties(device))
File "/opt/conda/lib/python3.7/site-packages/torch/cuda/__init__.py", line 356, in get_device_properties
_lazy_init() # will define _get_device_properties
File "/opt/conda/lib/python3.7/site-packages/torch/cuda/__init__.py", line 214, in _lazy_init
torch._C._cuda_init()
RuntimeError: Unexpected error from cudaGetDeviceCount(). Did you run some cuda functions before calling NumCudaDevices() that might have already set an error? Error 803: system has unsupported display driver / cuda driver combination</module>
```

解决方式

1. 先排查cuda和torch版本是否兼容。

   ```
   # cuda版本
   nvcc --version
   # nvidia-smi版本
   nvidia-smi

   # torch版本（要确定用户用的哪个conda下的python）
   python -c "import torch;print(torch.__version__)"
   ```

   通过pytorch官网可查兼容版本：**https://pytorch.org/get-started/previous-versions/**

2. 如果环境中装了多版本的cuda，可以排查LD_LIBRARY_PATH中的cuda优先级，需要手动调整下。

   举例：如果cuda只兼容cuda-9.1，查询到*LD_LIBRARY_PATH=/usr/local/cuda-11.8/lib64:/usr/local/cuda-9.1/lib64*

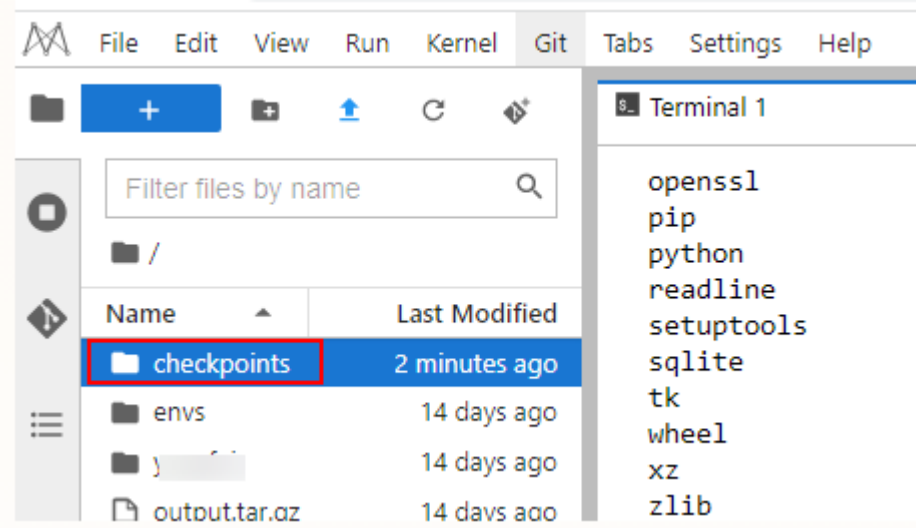
   需要手动调整优先级，执行命令*export LD_LIBRARY_PATH=/usr/local/cuda-9.1/lib64:$LD_LIBRARY_PATH*

3.7 其他故障

3.7.1 Notebook 中无法打开“checkpoints”文件夹

checkpoints是Notebook的关键字，如果用户创建文件夹命名为checkpoints，则在JupyterLab上无法打开、重命名和删除。此时可以在Terminal里使用命令行打开checkpoints，或者新建文件夹将checkpoints里的数据移动到新的文件夹下。

图 3-22 JupyterLab 浏览器左侧导航无法打开 checkpoints



操作步骤：

打开Terminal，用命令行进行操作。

方法一：执行**cd checkpoints命令**打开checkpoints文件夹。

方法二：新建一个文件夹，移动checkpoints文件夹的数据到新建的文件夹下。

1. 执行**mkdir *xxx***命令，新建一个文件夹，例如“xxx”（不要用checkpoints关键字命名）
2. 然后移动checkpoints文件夹的数据到新建的文件夹下，删除根目录下checkpoints文件夹即可。

```
mv checkpoints/* xxx
rm -r checkpoints
```

3.7.2 创建新版 Notebook 无法使用已购买的专属资源池，如何解决?

问题现象

已购买专属资源池，但创建Notebook时该资源池不可选择，无法创建Notebook。

提示当前专属资源池未初始化开发环境，请到专属资源池页面初始化开发环境。

原因分析

新购买的专属资源池，需要初始化环境才能用于创建Notebook。

解决方法

请到专属资源池页面初始化开发环境。

步骤1 进入“专属资源池”页面，单击目标资源池“操作”列的“更多 > 设置作业类型”。

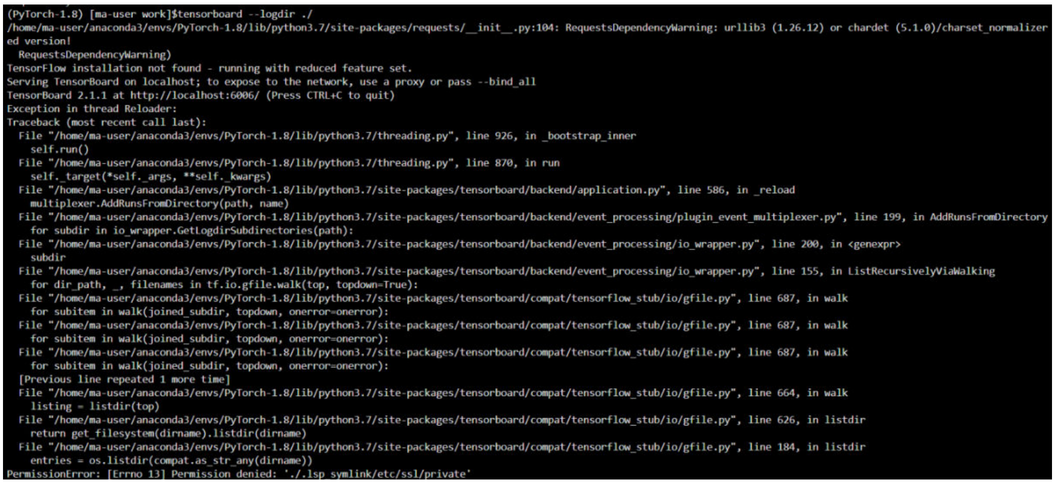
步骤2 在“设置作业类型”页面，勾选“开发环境”，单击“确定”。此时“开发环境”的状态为“环境初始化中”，等到状态为“已启用”，即可使用新购买的专属资源池。

----结束

3.7.3 在 Notebook 中使用 tensorboard 命令打开日志文件报错 Permission denied

问题现象

在Notebook的Terminal中执行**tensorboard --logdir ./**命令，报错[Errno 13] Permission denied……。



原因分析

当前目录下包含没有权限的文件。

解决方法

建议用户新建一个文件夹（例如：tb_logs），将tensorboard的日志文件（例如：tb.events）放到新建的文件夹下，然后执行tensorboard命令。示例命令如下：

```
mkdir -p ./tb_logs
mv tb.events ./tb_logs
tensorboard --logdir ./tb_logs
```

4 训练作业

4.1 OBS 操作相关故障

4.1.1 读取文件报错，如何正确读取文件

问题现象

- 创建训练作业如何读取“json”和“npy”文件。
- 训练作业如何使用cv2库读取文件。
- 如何在MXNet环境下使用torch包。
- 训练作业读取文件，出现如下报错：
  ```
  NotFoundError (see above for traceback): Unsucessful TensorSliceReader constructor: Failed to find any matching files for xxx://xxx
  ```

原因分析

在ModelArts中，用户的数据都是存放在OBS桶中，而训练作业运行在容器中，无法通过访问本地路径的方式访问OBS桶中的文件。

处理方法

读取文件报错，您可以使用Moxing将数据复制至容器中，再直接访问容器中的数据。请参见步骤**1**。

您也可以根据不同的文件类型，进行读取。请参见**读取“json”文件**、**读取“npy”文件**、**使用cv2库读取文件**和**在MXNet环境下使用torch包**。

1. 读取文件报错，您可以使用Moxing将数据复制至容器中，再直接访问容器中的数据。具体方式如下：
   ```
   import moxing as mox
   mox.file.make_dirs('/cache/data_url')
   mox.file.copy_parallel('obs://bucket-name/data_url', '/cache/data_url')
   ```
2. **读取**“json”**文件**，请您在代码中尝试如下方法：
   ```
   json.loads(mox.file.read(json_path, binary=True))
   ```
3. **使用“numpy.load”读取**“npy”**文件**，请您在代码中尝试如下方法：

- 使用MoXing API读取OBS中的文件
  ```
  np.load(mox.file.read(_SAMPLE_PATHS['rgb'], binary=True))
  ```
- 使用MoXing的file模块对OBS文件进行读写
  ```
  with mox.file.File(_SAMPLE_PATHS['rgb'], 'rb') as f:
  np.load(f)
  ```

4. **使用cv2库读取文件**，请您尝试如下方法：
   ```
   cv2.imdecode(np.fromstring(mox.file.read(img_path), np.uint8), 1)
   ```
5. **在MXNet环境下使用torch包**，请您尝试如下方法先进行导包：
   ```
   import os
   os.sysytem('pip install torch')
   import torch
   ```

4.1.2 TensorFlow-1.8 作业连接 OBS 时反复出现提示错误

问题现象

基于TensorFlow-1.8启动训练作业，并在代码中使用“tf.gfile”模块连接OBS，启动训练作业后会频繁打印如下日志信息：

```
Connection has been released. Continuing.
Found secret key
```

原因分析

这是TensorFlow-1.8中会出现的情况，该日志是Info级别的，并不是错误信息，可以通过设置环境变量来屏蔽INFO级别的日志信息。环境变量的设置一定要在import tensorflow或者import moxing之前。

处理方法

您需要通过在代码中设置环境变量“TF_CPP_MIN_LOG_LEVEL”来屏蔽INFO级别的日志信息。具体操作如下：

```
import os

os.environ['TF_CPP_MIN_LOG_LEVEL'] = '2'

import tensorflow as tf
import moxing.tensorflow as mox
```

“TF_CPP_MIN_LOG_LEVEL”与日志等级对应关系为：

```
import os
os.environ["TF_CPP_MIN_LOG_LEVEL"]='1'    # 默认的显示等级，显示所有信息
os.environ["TF_CPP_MIN_LOG_LEVEL"]='2'    # 只显示warning和Error
os.environ["TF_CPP_MIN_LOG_LEVEL"]='3'    # 只显示Error
```

4.1.3 TensorFlow 在 OBS 写入 TensorBoard 到达 5GB 时停止

问题现象

ModelArts训练作业出现如下报错：

```
Encountered Unknown Error EntityTooLarge
Your proposed upload exceeds the maximum allowed object size.:
If the signature check failed. This could be because of a time skew. Attempting to adjust the signer
```

原因分析

OBS限制单次上传文件大小为5GB，TensorFlow保存summary可能是本地缓存，在每次触发flush时将该summary文件覆盖OBS上的原文件。当超过5GB后，由于达到了OBS单次导入文件大小的上限，导致无法继续写入。

处理方法

如果在运行训练作业的过程中出现该问题，建议处理方法如下：

1. 推荐使用本地缓存的方式来解决，使用如下方法：
   ```
   import moxing.tensorflow as mox
   mox.cache()
   ```

4.1.4 保存模型时出现 Unable to connect to endpoint 错误

问题现象

训练作业保存模型时日志报错，具体信息如下：

InternalError (see above for traceback): : Unable to connect to endpoint

原因分析

OBS连接不稳定可能会出现报错，“Unable to connect to endpoint”。

处理方法

对于OBS连接不稳定的现象，通过增加代码来解决。您可以在代码最前面增加如下代码，让TensorFlow对ckpt和summary的读取和写入可以通过本地缓存的方式中转解决：

```
import moxing.tensorflow as mox

mox.cache()
```

4.1.5 OBS 复制过程中提示“BrokenPipeError: Broken pipe”

问题现象

训练作业在使用MoXing复制数据时，日志中出现报错“BrokenPipeError: [Errno xx] Broken pipe”。

原因分析

出现该问题的可能原因如下：

- 在大规模分布式作业上，每个节点都在复制同一个桶的文件，导致OBS桶限流。
- OBS Client连接数过多，进程/线程之间的轮询，导致一个OBS Client与服务端连接30S内无响应，超过超时时间，服务端断开了连接。

处理方法

1. 如果是限流问题，日志中还会出现如下报错，OBS相关的错误码解释请参见**OBS官方文档**，这种情况建议提工单。

```
[ModelArts Service Log]2021-01-21 11:35:42,178 - file_io.py[line:658] - ERROR:
    stat:503
    errorCode:None
    errorMessage:None
    reason:Service Unavailable
```

2. 如果是client数太多，尤其对于5G以上文件，OBS接口不支持直接调用，需要分多个线程分段复制，目前OBS侧服务端超时时间是30S，可以通过如下设置减少进程数。

```
# 设置进程数
os.environ['MOX_FILE_LARGE_FILE_TASK_NUM']=1
import moxing as mox

# 复制文件
mox.file.copy_parallel(src_url=your_src_dir, dst_url=your_target_dir, threads=0, is_processing=False)
```

说明

创建训练作业时，可通过环境变量“MOX_FILE_PARTIAL_MAXIMUM_SIZE”设置用户需要分段下载的大文件阈值（单位为Bytes），超过该阈值的文件将使用并发下载模式进行分段下载。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.1.6 日志提示“ValueError: Invalid endpoint: obs.xxxx.com”

问题现象

训练作业中使用Tensorboard直接写入到OBS路径，在日志中出现报错信息“ValueError: Invalid endpoint: obs. xxxx.com”。

原因分析

出现该问题的可能原因：

直接在OBS上写tensorboard文件，存在不稳定的风险。

处理方法

建议先将Tensorboard文件写到本地，然后再复制回OBS。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.1.7 日志提示“errorMessage:The specified key does not exist”

问题现象

在用moxing访问OBS路径时，出现如下错误：

```
ERROR:root:
stat:404
errorCode:NoSuchKey
errorMessage:The specified key does not exist.
```

原因分析

出现该问题的可能原因如下：

桶中的对象不存在，请检查OBS路径中的内容是否存在。具体错误码请参见**OBS官方文档**。

处理方法

1. 检查OBS路径及内容格式是否正常。
2. 必现的问题，使用本地Pycharm远程连接Notebook调试。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.2 云上迁移适配故障

4.2.1 无法导入模块

问题现象

ModelArts训练作业导入模块时日志报错：

```
Traceback (most recent call last):File "project_dir/main.py", line 1, in <module>from module_dir import module_file
ImportError: No module named module_dir
ImportError: No module named xxx
```

原因分析

- 训练作业导入模块时日志出现前两条报错信息，可能原因如下：

  代码如果在本地运行，需要将“project_dir”加入到PYTHONPATH或者将整个“project_dir”安装到“site-package”中才能运行。但是在ModelArts可以将“project_dir”加入到“sys.path”中解决该问题。

  使用**from module_dir import module_file**来导包，代码结构如下：

```
project_dir
|- main.py
|- module_dir
|  |- __init__.py
|  |- module_file.py
```

- 训练作业导入模块时日志出现“ImportError: No module named xxx”的报错，可以判断是环境中没有包含用户依赖的python包。

处理方法

- 训练作业导入模块时日志出现前两条报错信息，处理方法如下：
  a. 首先保证被导入的module中有“__init__.py”存在，创建“module_dir”的“__init__.py”，如**原因分析**中的结构所示。
  b. 由于无法知晓“project_dir”在容器中的位置，所以利用绝对路径，在“main.py”中将“project_dir”添加到“sys.path”中，再导入：

```
import os
import sys
# __file__为获取当前执行脚本main.py的绝对路径
# os.path.dirname(__file__)获取main.py的父目录，即project_dir的绝对路径
current_path = os.path.dirname(__file__)
sys.path.append(current_path)
# 在sys.path.append执行完毕之后再导入其他模块
from module_dir import module_file
```

- 训练作业导入模块时日志出现“ImportError: No module named xxx”的报错，请添加如下代码安装依赖包：

```
import os
os.system('pip install xxx')
```

4.2.2 训练作业日志中提示“No module named .*”

用户请按照以下思路进行逐步排查：

1. **检查依赖包是否存在**
2. **检查依赖包路径是否能被识别**
3. **检查训练作业使用的资源规格是否正确**
4. **建议与总结**

检查依赖包是否存在

如果依赖包不存在，您可以使用以下两种方式完成依赖包的安装。

- 方式一（推荐使用）：在**创建我的算法**时，需要在“代码目录”下放置相应的文件或安装包。

  请根据依赖包的类型，在代码目录下放置对应文件：

  - **依赖包为开源安装包时**

    在“代码目录”中创建一个命名为“pip-requirements.txt”的文件，并且在文件中写明依赖包的包名及其版本号，格式为“包名==版本号”。

    例如，“代码目录”对应的OBS路径下，包含模型文件，同时还存在“pip-requirements.txt”文件。“代码目录”的结构如下所示：

```
|---模型启动文件所在OBS文件夹
    |---model.py              #模型启动文件。
    |---pip-requirements.txt  #定义的配置文件，用于指定依赖包的包名及版本号。
```

    “pip-requirements.txt”文件内容如下所示：

```
alembic==0.8.6
bleach==1.4.3
click==6.6
```

- **依赖包为whl包时**

  如果训练后台不支持下载开源安装包或者使用用户编译的whl包时，由于系统无法自动下载并安装，因此需要在“代码目录”放置此whl包，同时创建一个命名为“pip-requirements.txt”的文件，并且在文件中指定此whl包的包名。依赖包必须为“.whl”格式的文件。

  例如，“代码目录”对应的OBS路径下，包含模型文件、whl包，同时还存在“pip-requirements.txt”文件。“代码目录”的结构如下所示：

```
|---模型启动文件所在OBS文件夹
    |---model.py              #模型启动文件。
    |---XXX.whl               #依赖包。依赖多个时，此处放置多个。
    |---pip-requirements.txt  #定义的配置文件，用于指定依赖包的包名。
```

  “pip-requirements.txt”文件内容如下所示：

```
numpy-1.15.4-cp36-cp36m-manylinux1_x86_64.whl
tensorflow-1.8.0-cp36-cp36m-manylinux1_x86_64.whl
```

- 方式二：可以在启动文件添加如下代码安装依赖包：

```
import os
os.system('pip install xxx')
```

方式一在训练作业启动前即可完成相关依赖包的下载与安装，而方式二是运行启动文件过程中进行依赖包的下载与安装。

检查依赖包路径是否能被识别

代码如果在本地运行，需要将“project_dir”加入到PYTHONPATH或者将整个“project_dir”安装到“site-package”中才能运行。但是在ModelArts可以将“project_dir”加入到“sys.path”中解决该问题。

使用**from module_dir import module_file**来导包，代码结构如下：

```
project_dir
|- main.py
|- module_dir
|  |- __init__.py
|  |- module_file.py
```

检查训练作业使用的资源规格是否正确

训练作业报错No module named npu_bridge.npu_init

```
from npu_bridge.npu_init import *
ImportError: No module named npu_bridge.npu_init
```

检查下训练作业使用的规格是否支持NPU，有可能是训练时使用了GPU规格，导致发生了NPU相关调用报错。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.2.3 如何安装第三方包，安装报错的处理方法

问题现象

- ModelArts**如何安装自定义库函数**，例如“apex”。
- ModelArts训练环境安装第三方包时出现如下报错：
  ```
  xxx.whl is not a supported wheel on this platform
  ```

原因分析

由于安装的文件名格式不支持，导致出现“xxx.whl is not a supported wheel on this platform”报错，具体解决方法请参见**2**。

处理方法

1. **安装第三方包**
   a. pip中存在的包，使用如下代码：
   ```
   import os
   os.system('pip install xxx')
   ```
   b. pip源中不存在的包，此处以“apex”为例，请您用如下方式将安装包上传到OBS桶中。 该样例已将安装包上传至“obs://cnnorth4-test/codes/mox_benchmarks/apex-master/”中，将在启动文件中添加以下代码进行安装。
   ```
   try:
       import apex
   except Exception:
       import os
       import moxing as mox
       mox.file.copy_parallel('obs://cnnorth4-test/codes/mox_benchmarks/apex-master/', '/cache/apex-master')
       os.system('pip --default-timeout=100 install -v --no-cache-dir --global-option="--cpp_ext" --global-option="--cuda_ext" /cache/apex-master')
   ```
2. **安装报错**
   “xxx.whl”文件无法安装，需要您按照如下步骤排查：
   a. 当出现“xxx.whl”文件无法安装，在启动文件中添加如下代码，查看当前pip命令支持的文件名和版本。
   ```
   import pip
   print(pip.pep425tags.get_supported())
   ```
   获取到支持的文件名和版本如下：
   ```
   [('cp36', 'cp36m', 'manylinux1_x86_64'), ('cp36', 'cp36m', 'linux_x86_64'), ('cp36', 'abi3', 'manylinux1_x86_64'), ('cp36', 'abi3', 'linux_x86_64'), ('cp36', 'none', 'manylinux1_x86_64'), ('cp36', 'none', 'linux_x86_64'), ('cp35', 'abi3', 'manylinux1_x86_64'), ('cp35', 'abi3', 'linux_x86_64'), ('cp34', 'abi3', 'manylinux1_x86_64'), ('cp34', 'abi3', 'linux_x86_64'), ('cp33', 'abi3', 'manylinux1_x86_64'), ('cp33', 'abi3', 'linux_x86_64'), ('cp32', 'abi3', 'manylinux1_x86_64'), ('cp32', 'abi3', 'linux_x86_64'), ('py3', 'none', 'manylinux1_x86_64'), ('py3', 'none', 'linux_x86_64'), ('cp36', 'none', 'any'), ('cp3', 'none', 'any'), ('py36', 'none', 'any'), ('py3', 'none', 'any'), ('py35', 'none', 'any'), ('py34', 'none', 'any'), ('py33', 'none', 'any'), ('py32', 'none', 'any'), ('py31', 'none', 'any'), ('py30', 'none', 'any')]
   ```
   b. 将“faiss_gpu-1.5.3-cp36-cp36m-manylinux2010_x86_64.whl”更改为“faiss_gpu-1.5.3-cp36-cp36m-manylinux1_x86_64.whl”，并安装，执行命令如下：
   ```
   import moxing as mox
   import os

   mox.file.copy('obs://wolfros-net/zp/AI/code/faiss_gpu-1.5.3-cp36-cp36m-manylinux2010_x86_64.whl','/cache/faiss_gpu-1.5.3-cp36-cp36m-manylinux1_x86_64.whl')
   os.system('pip install /cache/faiss_gpu-1.5.3-cp36-cp36m-manylinux1_x86_64.whl')
   ```

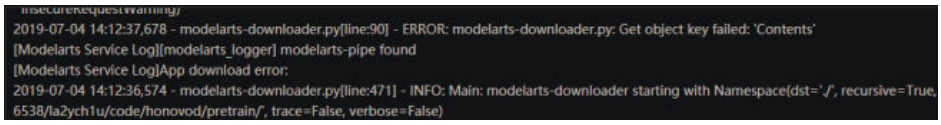
4.2.4 下载代码目录失败

问题现象

训练作业运行时下载失败，出现如下报错，请参见图4-1：

```
ERROR：modelarts-downloader.py: Get object key failed: 'Contents'
```

图 4-1 获取内容失败



原因分析

在创建训练作业时指定的代码目录不存在导致训练失败。

处理方法

请您根据报错原因排查创建训练作业时指定的代码目录，即OBS桶的路径是否正确。有两种方法判断是否存在。

- 使用当前账户登录OBS管理控制台，去查找对应的OBS桶、文件夹、文件是否存在。
- 通过接口判断路径是否存在。在代码中执行如下命令，检查路径是否存在。

```
import moxing as mox
mox.file.exists('obs://obs-test/ModelArts/examples/')
```

4.2.5 训练作业日志中提示“No such file or directory”

问题现象

训练作业运行失败，日志中提示“No such file or directory”。

例如：找不到训练输入的数据路径时，会提示“No such file or directory”。

例如：找不到训练启动文件时，也会提示“No such file or directory”。

原因分析

- 找不到训练输入数据路径，可能是报错的路径填写不正确。用户请按照以下思路进行逐步排查：
  a. **检查报错的路径是否为OBS路径**
  b. **检查报错的路径是否存在**
- 找不到启动文件，可能是训练作业启动命令的路径填写不正确，参考**使用自定义镜像创建训练作业时，检查启动文件路径**排查解决。
- 可能为多个进程或者worker读写同一个文件。如果使用了SFS，则考虑是否多个节点同时写同一个文件。分析代码中是否存在多进程写同一文件的情况。建议避免作业中存在多进程，多节点并发读写同一文件的情况。

检查报错的路径是否为 OBS 路径

使用ModelArts时，用户数据需要存放在自己OBS桶中，但是训练代码运行过程中不能使用OBS路径读取数据。

原因：

训练作业创建成功后，由于在运行容器直连OBS服务进行训练性能很差，系统会自动下载训练数据至运行容器的本地路径。所以，在训练代码中直接使用OBS路径会报错。例如训练代码的OBS路径为obs://bucket-A/training/，训练代码会被自动下载至${MA_JOB_DIR}/training/。

假设训练代码的OBS目录为obs://bucket-A/XXX/{training-project}/，“{training-project}”是存放训练代码的文件夹名称。训练时会自动下载OBS中{training-project}目录下的数据到训练容器的本地路径$MA_JOB_DIR/{training-project}/。

如果报错路径为训练数据路径，需要在以下两个地方完成适配，具体适配方法请参考自定义算法适配章节的**输入输出配置部分**：

1. 在创建算法时，您需要在输入路径配置中设置**代码路径参数**，默认为“data_url”。
2. 您需要在训练代码中添加超参，默认为“data_url”。使用“data_url”当做训练数据输入的本地路径。

检查报错的路径是否存在

由于用户本地开发的代码需要上传至ModelArts后台，训练代码中涉及到依赖文件的路径时，用户设置有误的场景较多。

推荐通用的解决方案：使用os接口得到依赖文件的绝对路径，避免报错。

示例：

```
|---project_root              #代码根目录
  |---BootfileDirectory       #启动文件所在的目录
    |---bootfile.py           #启动文件
  |---otherfileDirectory      #其他依赖文件所在的目录
    |---otherfile.py          #其他依赖文件
```

在启动文件中，建议用户参考以下方式获取依赖文件所在路径，即示例中的otherfile_path。

```
import os
current_path = os.path.dirname(os.path.realpath(__file__)) # BootfileDirectory, 启动文件所在的目录
project_root = os.path.dirname(current_path) # 工程的根目录，对应ModelArts训练控制台上设置的代码目录
otherfile_path = os.path.join(project_root, "otherfileDirectory", "otherfile.py")
```

使用自定义镜像创建训练作业时，检查启动文件路径

以OBS路径“obs://obs-bucket/training-test/demo-code”为例，训练代码会被自动下载至训练容器的“${MA_JOB_DIR}/demo-code”目录中，demo-code为OBS存放代码路径的最后一级目录，可以根据实际修改。

使用自定义镜像创建训练作业时，在代码目录下载完成后，镜像的启动命令会被自动执行。启动命令的填写规范如下：

- 如果训练启动脚本用的是py文件，例如train.py，运行命令可以写为python ${MA_JOB_DIR}/demo-code/train.py。

- 如果训练启动脚本用的是sh文件，例如main.sh，运行命令可以写为bash ${MA_JOB_DIR}/demo-code/main.sh。

其中demo-code为OBS存放代码路径的最后一级目录，可以根据实际修改。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE（VS Code）开发模型**。

4.2.6 训练过程中无法找到 so 文件

问题现象

ModelArts训练作业运行时，日志中遇到如下报错，导致训练失败：

```
libcudart.so.9.0 cannot open shared object file no such file or directory
```

原因分析

编译生成so文件的cuda版本与训练作业的cuda版本不一致。

处理方法

编译环境的cuda版本与训练环境不一致，训练作业运行就会报错。例如：使用cuda版本为10的开发环境tf-1.13中编译生成的so包，在cuda版本为9.0训练环境中tf-1.12训练会报该错。

编译环境和训练环境的cuda版本不一致时，可参考如下处理方法：

1. 在业务执行前加如下命令，检查是否能找到so文件。如果已经找到so文件，执行2；如果没有找到，执行3。
   ```
   import os;
   os.system(find /usr -name *libcudart.so*);
   ```
2. 设置环境变量LD_LIBRARY_PATH，设置完成后，重新下发作业即可。

   例如so文件的存放路径为：/use/local/cuda/lib64，LD_LIBRARY_PATH设置如下：
   ```
   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/cuda/lib64
   ```
3. 执行如下命令，查看训练环境的cuda版本，确认当前cuda版本是否支持so文件。
   ```
   os.system("cat /usr/local/cuda/version.txt")
   ```
   a. 支持。当前cuda版本无so文件，需外部导入so文件（自行在浏览器下载），再设置LD_LIBRARY_PATH，具体见2。
   b. 不支持。尝试更换引擎，重新下发作业。或者使用自定义镜像创建作业，可参考**使用自定义镜像创建作业**。

4.2.7 ModelArts 训练作业无法解析参数，日志报错

问题现象

ModelArts训练作业无法解析参数，遇到如下报错，导致无法正常运行：

```
error: unrecognized arguments: --data_url=xxx://xxx/xxx
error: unrecognized arguments: --init_method=tcp://job
absl.flags._exceptions.UnrecognizedFlagError:Unknown command line flag 'task_index'
```

原因分析

- 运行参数中未定义该参数。
- 在训练环境中，系统可能会传入在Python脚本里没有定义的其他参数名称，导致参数无法解析，日志报错。

处理方法

1. 参数定义中增加该参数的定义，代码示例如下：
   ```
   parser.add_argument('--init_method', default='tcp://xxx',help="init-method")
   ```
2. 通过使用解析方式**args, unparsed = parser.parse_known_args()**代替**args = parser.parse_args()**解决该问题。代码示例如下：
   ```
   import argparse
   parser = argparse.ArgumentParser()
   parser.add_argument('--data_url', type=str, default=None, help='obs path of dataset')
   args, unparsed = parser.parse_known_args()
   ```

4.2.8 训练输出路径被其他作业使用

问题现象

在创建训练作业时出现如下报错：操作失败！Other running job contain train_url: /bucket-20181114/code_hxm/

原因分析

根据报错信息判断，在创建训练作业时，同一个“训练输出路径”在被其他作业使用。

处理方法

一个“训练输出路径”只能被一个处于“运行中”、“排队中”或“初始化”状态的作业使用。

当出现此报错时，建议检查并重新填写训练作业的“训练输出路径”，以避免创建作业失败。

4.2.9 PyTorch1.0 引擎提示“RuntimeError: std:exception”

问题现象

在使用PyTorch1.0镜像时，必现如下报错：

```
“RuntimeError: std:exception”
```

原因分析

PyTorch1.0镜像中的libmkldnn软连接与原生torch的冲突，具体可参看**文档**。

处理方法

1. 按照issues中的说明，应该是环境中的库冲突了，因此在启动脚本最开始之前，添加如下代码。
```
import os
os.system("rm /home/work/anaconda3/lib/libmkldnn.so")
os.system("rm /home/work/anaconda3/lib/libmkldnn.so.0")
```
2. 必现的问题，使用本地Pycharm远程连接Notebook调试。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.2.10 MindSpore 日志提示“ retCode=0x91, [the model stream execute failed]”

问题现象

使用mindspore进行训练时，出现如下报错：
```
[ERROR] RUNTIME(3002)model execute error, retCode=0x91, [the model stream execute failed]
```

原因分析

出现该问题的可能原因如下：

数据读入的速度跟不上模型迭代的速度。

处理方法

1. 减少预处理shuffle操作。
```
dataset = dataset.shuffle(buffer_size=x)
```
2. 关闭数据预处理开关，可能会影响性能。
```
NPURunConfig(enable_data_pre_proc=false)
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.2.11 使用 moxing 适配 OBS 路径，pandas 读取文件报错

问题现象

使用moxing适配OBS路径，然后用较高版本的pandas读取OBS文件报出如下错误：

```
1.‘can't decode byte xxx in position xxx’
2.‘OSError:File isn't open for writing’
```

原因分析

出现该问题的可能原因如下：

moxing对高版本的pandas兼容性不够。

处理方法

1. 在适配OBS路径后，读取文件模式从‘r’改成‘rb’，然后将mox.file.File的'_write_check_passed'属性值改为‘True’，参考如下代码。

```
import pandas as pd
import moxing as mox

mox.file.shift('os', 'mox')  # 将os的open操作替换为mox.file.File适配OBS路径的操作

param = {'encoding': 'utf-8'}
path = 'xxx.csv'
with open(path, 'rb') as f:
    f._wirte_check_passed = True
    df = pd.read_csv(ff, **param)
```

2. 必现的问题，使用本地Pycharm远程连接Notebook调试。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.2.12 日志提示“Please upgrade numpy to >= xxx to use this pandas version”

问题现象

在安装其他包的时候，有依赖冲突，对numpy库有其他要求，但是发现numpy卸载不了。出现如下类似错误：

```
your numpy version is 1.14.5.Please upgrade numpy to >= 1.15.4 to use this pandas version
```

原因分析

出现该问题的可能原因如下：

conda和pip包混装，有一些包卸载不掉。

处理方法

参考如下代码，三步走。

1. 先卸载numpy中可以卸载的组件。
2. 删除你环境中site-packages路径下的numpy文件夹。

3. 重新进行安装需要的版本。

```
import os
os.system("pip uninstall -y numpy")
os.system('rm -rf /home/work/anaconda/lib/python3.6/site-packages/numpy/')
os.system("pip install numpy==1.15.4")
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.2.13 重装的包与镜像装 CUDA 版本不匹配

问题现象

在现有镜像基础上，重新装了引擎版本，或者编译了新的CUDA包，出现如下错误：

```
1.“RuntimeError: cuda runtime error (11) : invalid argument at /pytorch/aten/src/THC/
THCCachingHostAllocator.cpp:278”
2.“libcudart.so.9.0 cannot open shared object file no such file or directory”
3.“Make sure the device specification refers to a valid device, The requested device appeares to be a
GPU,but CUDA is not enabled”
```

原因分析

出现该问题的可能原因如下：

新安装的包与镜像中带的CUDA版本不匹配。

处理方法

必现的问题，使用本地Pycharm远程连接Notebook调试安装。

1. 先远程登录到所选的镜像，使用“nvcc -V”查看目前镜像自带的CUDA版本。
2. 重装torch等，需要注意选择与上一步版本相匹配的版本。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.2.14 创建训练作业提示错误码 ModelArts.2763

问题现象

创建训练作业时，提示ModelArts.2763：选择的支持实例无效，请检查请求中信息的合法性。

原因分析

用户选择的训练规格资源和算法不匹配。

例如：算法支持的是GPU规格，创建训练作业时选择了ASCEND规格的资源类型。

处理方法

1. 查看算法代码中设置的训练资源规格。
2. 检查创建训练作业时所选的资源规格是否正确，重新创建训练作业选择正确的资源规格。

4.2.15 训练作业日志中提示 “AttributeError: module '***' has no attribute '***'”

问题现象

训练日志中出现AttributeError: module '***' has no attribute '***'错误。如：AttributeError: module 'torch' has no attribute 'concat'。

原因分析

出现该问题的可能原因如下：

- 对应python包使用错误，该python包确实没有对应的变量或者方法
- 第三方pip源中的python包版本更新，导致在训练作业中安装的python包的版本可能也会发生变化。如训练作业之前无此问题，后面一直有此问题，则考虑是此原因。

处理方法

- 通过Notebook调试。
- 安装时指定版本。如：pip install xxx==1.x.x
- 第三方pip源可能随时更新，可通过制作自定义镜像，来避免该影响。可参见文档**模型训练中使用自定义镜像介绍**。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.2.16 系统容器异常退出

问题现象

在训练创建后出现“系统容器异常退出”的故障。

```
[ModelArts Service Log]2022-10-11 19:18:23,267 - file_io.py[1ine:748] - ERROR:
  stat:404
```

```
  errorCode:NoSuchKey
  errorMessage:The specifiedkey does not exist.
  reason:Not Found
  request-id:00000183C6C4010℃66D399E000COE3xx
  retry:0
[ModelArts Service Log]2022-10-11 19:18:23,267 - modelarts-downloader.py[line:90] - ERROR: modelarts-downloader. py: Download directory failed: [Errno
{'status': 404, ......}] file or directoryor bucket not found.
```

原因分析

出现该问题的可能原因如下：

1. OBS相关错误。
   a. OBS文件不存在。The specified key does not exist。
   b. 用户OBS权限不足。
   c. OBS限流。
   d. OBS其他问题。
2. 磁盘空间不足。

处理方法

1. 如果是OBS相关错误。
   a. OBS文件不存在。The specified key does not exist。
      参考**日志提示“errorMessage:The specified key does not exist”**章节处理。
   b. 用户OBS权限不足。
      参考 **5.5.1 日志提示“reason:Forbidden”**。
   c. OBS限流。
      参考**5.1.1 OBS复制过程中提示“BrokenPipeError: Broken pipe”**。
   d. OBS其他问题。
      请参考**OBS服务端错误码**或者采集request id后向OBS客服进行咨询。
2. 如果是空间不足。
   参考 **常见的磁盘空间不足的问题和解决办法**章节处理。

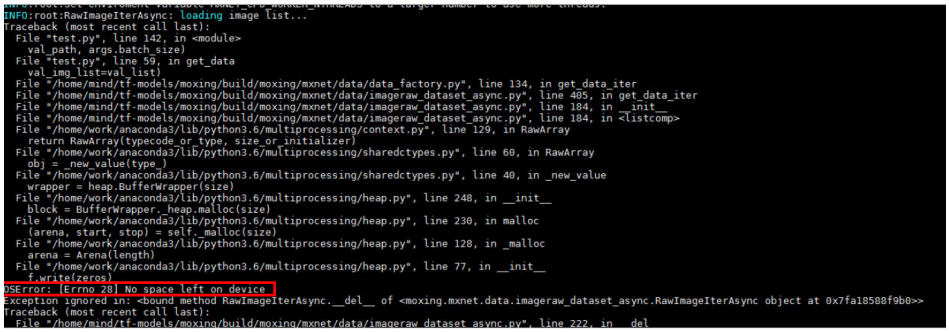
4.3 硬盘限制故障

4.3.1 下载或读取文件报错，提示超时、无剩余空间

问题现象

训练过程中复制数据/代码/模型时出现如下报错：

图 4-2 错误日志



原因分析

出现该问题的可能原因如下。

- 磁盘空间不足。
- 分布式作业时，有些节点的docker base size配置未生效，容器内“/”根目录空间未达到50GB，只有默认的10GB，导致作业训练失败。
- 实际存储空间足够，却依旧报错“No Space left on device”。

  同一目录下创建较多文件，为了加快文件检索速度，内核会创建一个索引表，短时间内创建较多文件时，会导致索引表达到上限，进而报错。

  说明

  触发条件和下面的因素有关：

  - 文件名越长，文件数量的上限越小
  - blocksize越小，文件数量的上限越小。（ blocksize，系统默认 4096B。总共有三种大小：1024B、2048B、4096B ）
  - 创建文件越快，越容易触发（机制大概是：有一个缓存，这块大小和上面的1和2有关，目录下文件数量比较大时会启动，使用方式是边用边释放 ）

处理方法

1. 可以参照**日志提示"write line error"**文档进行修复。
2. 如果是分布式作业有的节点有错误，有的节点正常，建议提工单请求隔离有问题的节点。
3. 如果是触发了欧拉操作系统的限制，有如下建议措施。
   - 分目录处理，减少单个目录文件量。
   - 减慢创建文件的速度。
   - 关闭ext4文件系统的dir_index属性，具体可参考：https://access.redhat.com/solutions/29894，（可能会影响文件检索性能 ）。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（ Pycharm或者VSCode ）联接云上环境调试请参考**使用本地IDE（ VS Code ）开发模型**。

4.3.2 复制数据至容器中空间不足

问题现象

ModelArts训练作业运行时，日志中遇到如下报错，导致数据无法复制至容器中。

```
OSError:[Errno 28] No space left on device
```

原因分析

数据下载至容器的位置空间不足。

处理方法

1. 请排查是否将数据下载至“/cache”目录下，GPU规格资源的每个节点会有一个“/cache”目录，空间大小为4TB。并确认该目录下并发创建的文件数量是否过大，占用过多存储空间会出现inode耗尽的情况，导致空间不足。
2. 请排查是否使用的是GPU资源。如果使用的是CPU规格的资源，“/cache”与代码目录共用10G，会造成内存不足，请更改为使用GPU资源。
3. 请在代码中添加环境变量来解决。
   ```
   import os
   os.system('export TMPDIR=/cache')
   ```

4.3.3 Tensorflow 多节点作业下载数据到/cache 显示 No space left

问题现象

创建训练作业，Tensorflow多节点作业下载数据到/cache显示：“No space left”。

原因分析

TensorFlow多节点任务会启动parameter server（简称ps）和worker两种角色，ps和worker会被调度到相同的机器上。由于训练数据对于ps没有用，因此在代码中ps相关的逻辑不需要下载训练数据。如果ps也下载数据到“/cache”，实际下载的数据会翻倍。例如只下载了2.5TB的数据，程序就显示空间不够而失败，因为/cache只有4TB的可用空间。

处理方法

在使用Tensorflow多节点作业下载数据时，正确的下载逻辑如下：

```
import argparse
parser = argparse.ArgumentParser()
parser.add_argument("--job_name", type=str, default="")
args = parser.parse_known_args()

if args[0].job_name != "ps":
    copy.....................
```

4.3.4 日志文件的大小达到限制

问题现象

ModelArts训练作业在运行过程中报错，提示日志文件的大小已达到限制：


```
modelarts-pope: log length overflow(max:1073741824; already: 107341771; new:90), process will continue running silently
```

原因分析

根据报错信息，可以判断是日志文件的大小已达到限制。出现该报错之后，日志不再增加，后台将继续运行。

处理方法

请您在启动文件中减少无用日志输出。

4.3.5 日志提示"write line error"

问题现象

在程序运行过程中，刷出大量错误日志“[ModelArts Service Log]modelarts-pipe: write line error”。并且问题是必现问题，每次运行到同一地方的时候，出现错误。

原因分析

出现该问题的可能原因如下：

- 程序运行过程中，产生了core文件，core文件占满了"/"根目录空间。
- 本地数据、文件保存将"/cache"目录3.5T空间用完了。

说明

云上训练磁盘空间一般指如下两个目录的磁盘空间：

1. “/”根目录，是docker中配置项“base size”，默认是10G，云上统一改为50G。
2. “/cache”目录满了，一般是3.5T存储空间满了。

处理方法

1. 如果在训练作业的工作目录下有core文件生成，可以在启动脚本最前面加上如下代码，来关闭core文件产生。
   ```
   import os
   os.system("ulimit -c 0")
   ```
2. 排查数据集大小，checkpoint保存文件大小，是否占满了磁盘空间。
3. 必现的问题，使用本地Pycharm远程连接Notebook调试。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

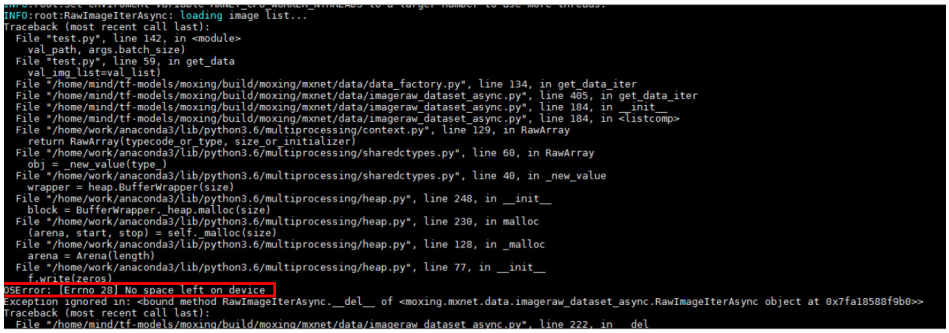
- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.3.6 日志提示“No space left on device”

问题现象

训练过程中复制数据/代码/模型时出现如下报错：

图 4-3 错误日志



原因分析

出现该问题的可能原因如下。

- 磁盘空间不足。
- 分布式作业时，有些节点的docker base size配置未生效，容器内“/”根目录空间未达到50G，只有默认的10GB，导致作业训练失败。
- 实际存储空间足够，却依旧报错“No Space left on device”。

  同一目录下创建较多文件，为了加快文件检索速度，内核会创建一个索引表，短时间内创建较多文件时，会导致索引表达到上限，进而报错。

  说明

  触发条件和下面的因素有关：

  - 文件名越长，文件数量的上限越小。
  - blocksize越小，文件数量的上限越小。（ blocksize，系统默认 4096B。总共有三种大小：1024B、2048B、4096B ）
  - 创建文件越快，越容易触发。

处理方法

1. 可以参照**日志提示"write line error"**文档进行修复。
2. 如果是分布式作业有的节点有错误，有的节点正常，建议提工单请求隔离有问题的节点。
3. 如果是触发了欧拉操作系统的限制，有如下建议措施。
   - 分目录处理，减少单个目录文件量。
   - 减慢创建文件的速度。
   - 关闭ext4文件系统的dir_index属性，具体可参考：https://access.redhat.com/solutions/29894，（ 可能会影响文件检索性能 ）。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.3.7 OOM 导致训练作业失败

问题现象

因为OOM导致的训练作业失败，会有如下几种现象。

1. 错误码返回137，如下图所示。
   ```
   Modelarts Service Log Trainina end with return code: 137
   Modelarts Service Log]handle outputs of training job
   ```
2. 日志中有报错，含有“killed”相关字段，例如：
   ```
   RuntimeError: DataLoader worker (pid 38077) is killed by signal: Killed.
   ```
3. 日志中有报错“RuntimeError: CUDA out of memory. ”，如下图所示：

   图 4-4 错误日志信息

   ```
   Traceback (most recent call last):
     File "memory_test.py", line 47, in <module>
       tmp_tensor = torch.empty(int(total_memory * 0.45), dtype=torch.int8, device='cuda')
   RuntimeError: CUDA out of memory. Tried to allocate 14.29 GiB (GPU 0; 14.29 GiB total capacity; 0 bytes
   already allocated; 14.29 GiB free; 0 bytes reserved in total by PyTorch)
   ```

4. Tensorflow引擎日志中出现“Dst tensor is not initialized”。

原因分析

按照之前支撑的经验，出现该问题的可能原因如下：

- 绝大部分都是确实是显存不够用。
- 还有较少数原因是节点故障，跑到特定节点必现OOM，其他节点正常。

处理方法

1. 如果是正常的OOM，就需要修改一些超参，释放一些不需要的tensor。
   a. 修改网络参数，比如batch_size、hide_layer、cell_nums等。
   b. 释放一些不需要的tensor，使用过的，如下：
      ```
      del tmp_tensor
      torch.cuda.empty_cache()
      ```
2. 必现的问题，使用本地Pycharm远程连接Notebook调试超参。
3. 如果还存在问题，可能需要提工单进行定位，甚至需要隔离节点修复。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

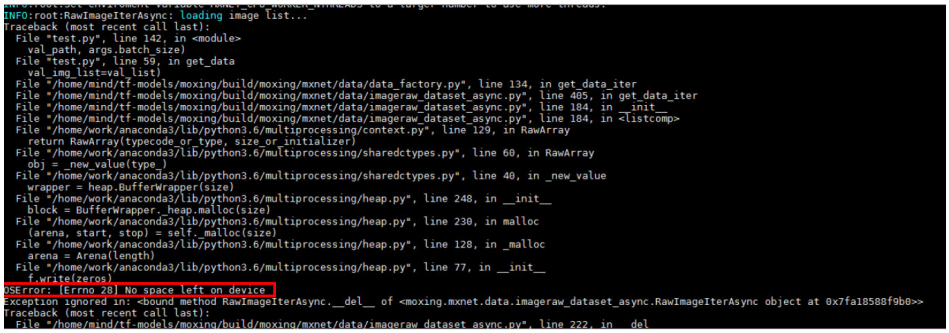
4.3.8 常见的磁盘空间不足的问题和解决办法

该章节用于统一整体所有的常见的磁盘空间不足的问题和解决办法。减少相关问题文档的重复内容。

问题现象

训练过程中复制数据/代码/模型时出现如下报错：

图 4-5 错误日志



原因分析

出现该问题的可能原因如下：

- 本地数据、文件保存将"/cache"目录空间用完。
- 数据处理过程中对数据进行解压，导致数据大小膨胀，将"/cache"目录空间用完。
- 数据未保存至/cache目录或者/home/ma-user/目录（/cache会软连接成/home/ma-user/），导致数据占满系统目录。系统目录仅支持系统功能基本运行，无法支持大数据存储。
- 部分训练任务会在训练过程中生成checkpoint文件，并进行更新。如更新过程中，未删除历史的checkpoint文件，会导致/cache目录逐步被用完。
- 实际存储空间足够，却依旧报错“No Space left on device”。可能是inode不足，或者是触发操作系统的文件索引缓存问题，导致操作系统无法创建文件，造成用户磁盘占满。

  说明

  触发条件和下面的因素有关：

  - 文件名越长，文件数量的上限越小。
  - blocksize越小，文件数量的上限越小。 blocksize系统默认为4096B，总共有三种大小：1024B、2048B、4096B。
  - 创建文件越快，越容易触发（机制大概是：有一个缓存，这块大小和上面的1和2有关，目录下文件数量比较大时会启动，使用方式是边用边释放）。
- 程序运行过程中，产生了core文件，core文件占满了"/"根目录空间。

处理方法

1. 排查数据集大小、数据集解压后的大小，checkpoint保存文件大小，是否占满了磁盘空间。
2. 如数据大小已超过/cache目录大小，则可以考虑通过SFS来额外挂载数据盘进行扩容。
3. 将数据和checkpoint保存在/cache目录或者/home/ma-user/目录。
4. 检查checkpoint相关逻辑，保证历史checkpoint不会不断积压，导致/cache目录用完。
5. 如文件大小小于/cache目录大小并且文件数量超过50w，则考虑为inode不足或者触发了操作系统的文件索引相关问题。需要：
   - 分目录处理，减少单个目录文件量。
   - 减慢创建文件的速度。如数据解压过程中，sleep 5s后再进行下一个数据的解压。
6. 如果训练作业的工作目录下有core文件生成，可以在启动脚本最前面加上如下代码，来关闭core文件产生。并推荐先在开发环境中进行代码调试。

```
import os
os.system("ulimit -c 0")
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.4 外网访问限制

4.4.1 日志提示“ Network is unreachable”

问题现象

在使用pytorch时，将torchvision.models中的pretrained置为了True，日志中出现如下报错：

```
‘OSError: [Errno 101] Network is unreachable’
```

原因分析

出现该问题的可能原因如下：

因为安全性问题，ModelArts内部训练机器不能访问外网。

处理方法

1. 将pretrained改成false，提前下载好预训练模型，加载下载好的预训练模型位置即可，可参考如下代码。

```
import torch
import torchvision.models as models
```

```
model1 = models.resnet34(pretrained=False, progress=True)
checkpoint = '/xxx/resnet34-333f7ec4.pth'
state_dict = torch.load(checkpoint)
model1.load_state_dict(state_dict)
```

2. 必现的问题，使用本地Pycharm远程连接Notebook调试。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.4.2 运行训练作业时提示 URL 连接超时

问题现象

训练作业在运行时提示URL连接超时，具体报错如下：

```
urllib.error.URLERROR:<urlopen error [Errno 110] Connection timed out>
```

原因分析

由于安全性问题在ModelArts上不能联网下载。

处理方法

如果在运行训练作业时提示连接超时，请您将需要联网下载的数据提前下载至本地，并上传至OBS中。

4.5 权限问题

4.5.1 训练作业访问 OBS 时，日志提示“stat:403 reason:Forbidden”

问题现象

训练作业访问OBS时，出现如下报错：

```
ERROR:root:Failed to call:
   func= <bound method ObsClient.getObjectMetadata of <moxing.framework.file.src.obs.client.ObsClient
object at 0x7fddb4ad06d0>>
   args=('bucket-cv-competition-bj4', 'fangjiemin/output/')
   kwargs={}
ERROR:root:
   stat:403
   errorCode:None
   errorMessage:None
   reason:Forbidden
   request-id:00000179D5ACCAC445CAA1A71019C9D0
   retry:0
```

原因分析

出现该问题的可能原因如下:

- OBS服务的权限出现问题，导致无法正常读取数据

处理方法

请检查OBS权限配置，如未解决问题可参考OBS文档的**已配置OBS权限，仍然无法访问OBS（403 AccessDenied）**。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。
- OBS服务相关报错可根据错误信息（包括errorCode、errorMessage等）判断具体错误原因。具体错误码请参考**OBS官方文档**:

4.5.2 日志提示"Permission denied"

问题现象

训练作业访问挂载的EFS，或者是执行.sh启动脚本时，出现如下错误:

- OSError: [Errno 13]Permission denied: '/xxx/xxxx'
- bash: /bin/ln: Permission denied
- 自定义镜像中，bash:/home/ma-user/.pip/pip.conf：Permission Denied
- 自定义镜像中，tee: /xxx/xxxx: Permission denied cp: cannot stat '': No such file or directory

原因分析

出现该问题的可能原因如下:

- [Errno 13]Permission denied: '/xxx/xxxx'
  - 上传数据时文件所属与文件权限未修改，导致训练作业以work用户组访问时没有权限了。
  - 在代码目录中的.sh复制到容器之后，需要添加“x”可执行权限。
- bash: /bin/ln: Permission denied

  因安全问题，不支持用户开通使用ln命令。
- bash:/home/ma-user/.pip/pip.conf：Permission Denied

  因从V1切换到V2时，ma-user的uid仍是1102未改变导致。
- tee: /xxx/xxxx: Permission denied cp: cannot stat '': No such file or directory

  可能原因是用户使用的启动脚本为旧版本的run_train.sh，脚本里面有某些环境变量在新版本下发的作业中并不存在这些环境变量导致。

- 可能原因是使用Python file接口并发读写同一文件。

处理方法

1. 对挂载盘的数据加权限，可以改为与训练容器内相同的用户组（1000），假如/nas盘是挂载路径，执行如下代码。

```
chown -R 1000：1000 /nas
或者
chmod 777 -R /nas
```

2. 如果是自定义镜像中拉取的.sh脚本没有执行权限，可以在自定义脚本启动前执行"chmod +x xxx.sh"添加可执行权限。
3. ModelArts控制台上创建训练作业自定义镜像入口，默认以1000 uid用户来启动v2容器镜像，将ma-user的uid从1102改为1000，改变方式如下（如果需要sudo权限，可取消sudoers行的注释）：

```
FROM {your-v1-custom-docker-image or other docker-image}

USER root

# prepare moxing_framework and seccomponent package
# and chmod/chown moxing_framework package to the right permission or owner (ma-user)

RUN groupadd ma-group -g 1000 && \
    useradd -d /home/ma-user -m -u 1000 -g 1000 -s /bin/bash ma-user && \
    chmod 770 /home/ma-user && \
    # usermod -a -G work ma-user && \
    # alien -i seccomponent-1.0.2-2.0.release.x86_64.rpm && \
    chmod 770 /root && \
    # or silver bullet of files permission
    # chmod -R 777 /root && \
    usermod -a -G root ma-user

# ENV LD_LIBRARY_PATH=/usr/local/seccomponent/lib:$LD_LIBRARY_PATH

# RUN echo "ma-user  ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers

# RUN pip install moxing_framework-2.0.0.rc2.4b57a67b-py2.py3-none-any.whl

USER ma-user
WORKDIR /home/ma-user
```

4. v1训练作业环境变量迁移v2说明：
   - v1的DLS_TASK_NUMBER环境变量，可以使用v2的MA_NUM_HOSTS环境变量替换，即选择的训练节点数。
   - v1的DLS_TASK_INDEX环境变量，当前可以使用v2的VC_TASK_INDEX环境变量替换，下一步使用MA_TASK_INDEX替换，建议使用demo script中的方式获取，以保证兼容性。
   - v1的BATCH_CUSTOM0_HOSTS环境变量，可以使用v2的${MA_VJ_NAME}-${MA_TASK_NAME}-0.${MA_VJ_NAME}:6666替换。
   - 一般而言，v1的BATCH_CUSTOM{N}_HOSTS环境变量，可以使用v2的${MA_VJ_NAME}-${MA_TASK_NAME}-{N}.${MA_VJ_NAME}:6666替换。
5. 分析代码中是否存在并发读写同一文件的逻辑，如有则进行修改。

   如用户使用多卡的作业，那么可能每张卡都会有同样的读写数据的代码，可参考如下代码修改。

```
import moxing as mox
from mindspore.communication import init, get_rank, get_group_size
init()
rank_id = get_rank()
# 仅让0号卡进行数据下载
if rank_id % 8 == 0:
    mox.file.copy_parallel('obs://bucket-name/dir1/dir2/', '/cache')
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.6 GPU 相关问题

4.6.1 日志提示"No CUDA-capable device is detected"

问题现象

在程序运行过程中，出现如下类似错误。

```
1.‘failed call to cuInit: CUDA_ERROR_NO_DEVICE:  no CUDA-capable device is detected’
2.‘No CUDA-capable device is detected although requirements are installed’
```

原因分析

出现该问题的可能原因如下：

- 用户/训练系统，将CUDA_VISIBLE_DEVICES传错了，检查CUDA_VISIBLE_DEVICES变量是否正常。
- 用户选择了1/2/4卡这些规格的作业，然后设置了CUDA_VISIBLE_DEVICES=‘1’这种类似固定的卡ID号，与实际选择的卡ID不匹配。

处理方法

1. 尽量代码里不要去修改CUDA_VISIBLE_DEVICES变量，用系统默认里面自带的。
2. 如果必须指定卡ID，需要注意1/2/4规格下，指定的卡ID与实际分配的卡ID不匹配的情况。
3. 如果上述方法还出现了错误，可以去notebook里面调试打印CUDA_VISIBLE_DEVICES变量，或者用以下代码测试，查看结果是否返回的是True。

```
import torch
torch.cuda.is_available()
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。

- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.6.2 日志提示“RuntimeError: connect() timed out”

问题现象

使用pytorch进行分布式训练时，日志中出现报错“RuntimeError: connect() timed out”。

原因分析

出现该问题的可能原因如下：

如果在此之前是有进行数据复制的，每个节点复制的速度不是同一个时间完成的，然后有的节点没有复制完，其他节点进行torch.distributed.init_process_group()导致超时。

处理方法

如果是多个节点复制不同步，并且没有barrier的话导致的超时，可以在复制数据之前，先进行torch.distributed.init_process_group()，然后再根据local_rank()==0去复制数据，之后再调用torch.distributed.barrier()等待所有rank完成复制。具体可参考如下代码：

```
import moxing as mox
import torch

torch.distributed.init_process_group()
if local_rank == 0:
    mox.file.copy_parallel(src,dst)

torch.distributed.barrier()
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.6.3 日志提示“cuda runtime error (10) : invalid device ordinal at xxx”

问题现象

训练作业失败，日志报出如下错误：

```
RuntimeError: cuda runtime error (10) : invalid device ordinal at xxx
```

图 4-6 错误日志

```
    main()
  File "train.py", line 278, in main
    torch.cuda.set_device(args.local_rank)
  File "/home/work/anaconda/lib/python3.6/site-packages/torch/cuda/__init__.py", line 300, in set_device
    torch._C._cuda_setDevice(device)
RuntimeError: cuda runtime error (10) : invalid device ordinal at /pytorch/torch/csrc/cuda/Module.cpp:37
```

原因分析

可以从以下角度排查：

- 请检查CUDA_VISIBLE_DEVICES设置的值是否与作业规格匹配。例如您选择4卡规格的作业，实际可用的卡ID为0、1、2、3，但是您在进行cuda相关的运算时，例如"tensor.to(device="cuda:7")"，将张量搬到了7号GPU卡上，超过了实际可用的ID号。
- 如果cuda相关运算设置的卡ID号在所选规格范围内，但是依旧出现了上述报错。可能是该资源节点中存在GPU卡损坏的情况，导致实际能检测到的卡少于所选规格。

处理方法

1. 建议直接根据系统分卡情况下传进去的CUDA_VISIBLE_DEVICES去设置，不用手动指定默认的。
2. 如果发现资源节点中存在GPU卡损坏，请联系技术支持处理。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.6.4 日志提示“RuntimeError: Cannot re-initialize CUDA in forked subprocess”

问题现象

在使用pytorch启动多进程的时候，出现如下报错：

```
RuntimeError: Cannot re-initialize CUDA in forked subprocess
```

原因分析

出现该问题的可能原因如下：

multiprocessing启动方式有误。

处理方法

可以参考**官方文档**，如下：

```
"""run.py:"""
#!/usr/bin/env python
import os
import torch
import torch.distributed as dist
import torch.multiprocessing as mp

def run(rank, size):
    """ Distributed function to be implemented later. """
    pass

def init_process(rank, size, fn, backend='gloo'):
```

```
    """ Initialize the distributed environment. """
    os.environ['MASTER_ADDR'] = '127.0.0.1'
    os.environ['MASTER_PORT'] = '29500'
    dist.init_process_group(backend, rank=rank, world_size=size)
    fn(rank, size)


if __name__ == "__main__":
    size = 2
    processes = []
    mp.set_start_method("spawn")
    for rank in range(size):
        p = mp.Process(target=init_process, args=(rank, size, run))
        p.start()
        processes.append(p)

    for p in processes:
        p.join()
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.6.5 训练作业找不到 GPU

问题现象

训练作业运行出现如下报错：

```
failed call to cuInit: CUDA_ERROR_NO_DEVICE: no CUDA-capable device is detected
```

原因分析

根据错误信息判断，报错原因为训练作业运行程序读取不到GPU。

处理方法

根据报错提示，请您排查代码，是否已添加以下配置，设置该程序可见的GPU：

```
os.environ['CUDA_VISIBLE_DEVICES'] = '0,1,2,3,4,5,6,7'
```

其中，0为服务器的GPU编号，可以为0，1，2，3等，表明对程序可见的GPU编号。如果未进行添加配置则该编号对应的GPU不可用。

4.7 业务代码问题

4.7.1 日志提示“pandas.errors.ParserError: Error tokenizing data. C error: Expected .* fields”

问题现象

使用pandas读取csv数据表时，日志报出如下错误导致训练作业失败：

```
pandas.errors.ParserError: Error tokenizing data. C error: Expected 4 field
```

原因分析

csv中文件的每一行的列数不相等。

处理方法

可以使用以下方法处理：

- 校验csv文件，将多出字段的行删除。
- 在代码中忽略错误行，参考如下：
  ```
  import pandas as pd
  pd.read_csv(filePath,error_bad_lines=False)
  ```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.2 日志提示“max_pool2d_with_indices_out_cuda_frame failed with error code 0”

问题现象

pytroch1.3镜像中，去升级了pytroch1.4的版本，导致之前在pytroch1.3跑通的代码报错如下：

```
“RuntimeError:max_pool2d_with_indices_out_cuda_frame failed with error code 0”
```

原因分析

出现该问题的可能原因如下：

pytorch1.4引擎与之前pytorch1.3版本兼容性问题。

处理方法

1. 在images之后添加contigous。
   ```
   images = images.cuda()
   pred = model(images.permute(0, 3, 1, 2).contigous())
   ```
2. 将版本回退至pytorch1.3。
3. 必现的问题，使用本地Pycharm远程连接Notebook调试。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。

- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.3 训练作业失败，返回错误码 139

问题现象

训练作业运行失败，返回错误码139，如下图所示：

```
[Modelarts Service Log]Training end with reeturn code: 139
INFO:root:Using MoXing-v1.17.2-c806a92f
INFO;root:Using OBS-Python-SDK-3.1.2
```

原因分析

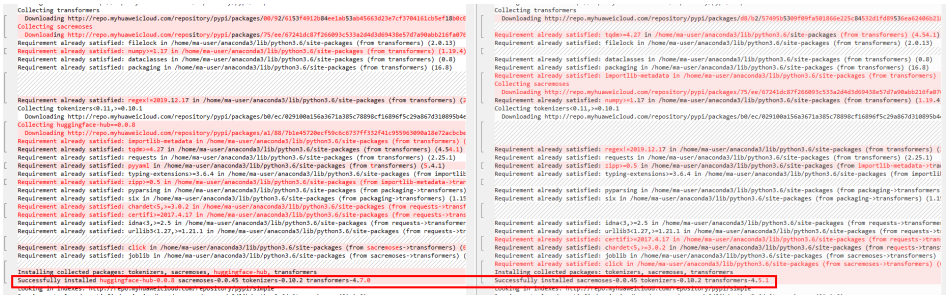
出现该问题的可能原因如下

- pip源中的pip包更新了，之前能跑通的代码，在包更新之后产生了不兼容的情况，例如transformers包，导致import的时候出现了错误。
- 用户代码问题，出现了内存越界、非法访问内存空间的情况。
- 未知系统问题导致，建议先尝试重建作业，重建后仍然失败，建议提工单定位。

处理方法

1. 如果存在之前能跑通，什么都没修改，过了一阵跑不通的情况，先去排查跑通和跑不通的日志是否存在pip源更新了依赖包，如下图，安装之前跑通的老版本即可。

   图 4-7 PIP 安装对比图

2. 推荐您使用本地Pycharm远程连接Notebook调试。
3. 如果上述情况都解决不了，请联系技术支持工程师。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.4 训练作业失败，如何使用开发环境调试训练代码？

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.5 日志提示“ '(slice(0, 13184, None), slice(None, None, None))' is an invalid key”

问题现象

训练过程中出现如下报错：

```
TypeError: '(slice(0, 13184, None), slice(None, None, None))' is an invalid key
```

原因分析

出现该问题的可能原因如下：

切分数据时，选择的数据不对。

处理方法

尝试如下代码：

```
X = dataset.iloc[:,:-1].values
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.6 日志报错“DataFrame.dtypes for data must be int, float or bool”

问题现象

训练过程中出现如下报错：

```
DataFrame.dtypes for data must be int, float or bool
```

原因分析

出现该问题的可能原因如下：

训练数据中出现了非int、float、bool类型数据。

处理方法

可参考如下代码，将错误列进行转换：

```
from sklearn import preprocessing
lbl = preprocessing.LabelEncoder()
train_x['acc_id1'] = lbl.fit_transform(train_x['acc_id1'].astype(str))
```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.7 日志提示“CUDNN_STATUS_NOT_SUPPORTED.”

问题现象

在pytorch训练时，出现如下报错：

```
RuntimeError: cuDNN error: CUDNN_STATUS_NOT_SUPPORTED. This error may appear if you passed in a non-contiguous input.
```

原因分析

出现该问题的可能原因如下：

数据输入不连续，cuDNN不支持的类型。

处理方法

1. 禁用cuDNN，在训练前加入如下代码。

   ```
   torch.backends.cudnn.enabled = False
   ```

2. 将输入数据转换成contiguous。

   ```
   images = images.cuda()
   images = images.permute(0, 3, 1, 2).contigous()
   ```

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.8 日志提示“Out of bounds nanosecond timestamp”

问题现象

在使用pandas.to_datetime转换时间时，出现如下报错：

```
pandas._libs.tslibs.np_datetime.OutOfBoundsDatetime: Out of bounds nanosecond timestamp: 1-01-02 13:20:00
```

原因分析

出现该问题的可能原因如下：

时间值越界，请参考**官方文档**。

处理方法

校验时间数据，pandas以纳秒表示时间戳。

- 最小时间：1677-09-22 00:12:43.145225
- 最大时间：2262-04-11 23:47:16.854775807，需注意上下界限。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.9 日志提示“Unexpected keyword argument passed to optimizer”

问题现象

在使用keras时，升级版本>=2.3.0之后，之前跑通的代码出现如下报错：

```
TypeError: Unexpected keyword argument passed to optimizer: learning_rate
```

原因分析

出现该问题的可能原因是“learning_rate”的参数名称写错了。keras官方文档中说明参数“lr”已重命名为“learning_rate”，在训练代码中必须写成“learning_rate”才能调用成功。keras官方文档请参见**https://github.com/keras-team/keras/releases/tag/2.3.0**。

处理方法

将训练代码里的参数名称“lr”改成“learning_rate”。

建议与总结

在创建训练作业前，推荐您先使用ModelArts开发环境调试训练代码，避免代码迁移过程中的错误。

- 直接使用线上notebook环境调试请参考**使用JupyterLab开发模型**。
- 配置本地IDE（Pycharm或者VSCode）联接云上环境调试请参考**使用本地IDE开发模型**。

4.7.10 日志提示“no socket interface found”

问题现象

在pytorch镜像运行分布式作业时，设置NCCL日志级别，代码如下：

```
import os
os.environ["NCCL_DEBUG"] = "INFO"
```

会出现如下错误：

```
job0879f61e-job-base-pda-2-0:712:71 2 [0] bootstrap.cc:37 NCCL WARN Bootstrap : no socket interface found
job0879f61e-job-base-pda-2-0:712:712 [0] NCCL INFO init.cC:128 -> 3
job0879f61e-job-base-pda-2-0:712:712 [0] NCCL INFO bootstrap.cc:76 -> 3
job0879f61e-job-base-pda-2-0:712:712 [0] NCCL INFO bootstrap.cc:245 -> 3    job0879f61e-job-base-pda-2-0:712:712 [0] NCCL INFO bootstrap.cc:266 -> 3
Traceback (most recent call last):
   File "train_net.py", line 1923, in <module>
      main_worker(args)
   File "train net.py", line 355, in main_ worker
      network = torch.nn.parallel.DistributedDataParallel(network, device_ids=device_ids, find_unused_parameters=True)
   File "/home/work/anaconda/lib/python3.6/site-packages/torch/nn/parallel/distributed.py", line 298, in init_self.broadcast bucket_size)
   File "/home/work/anaconda/lib/python3.6/site-packages/torch/nn/parallel/distributed.py", line 480, in _distributed broacIcast coalesced    dist. broadcast coalesced(seIf.process group, tensors, buffer size)
RuntimeError: NCCL error in: /pytorch/torch/lib/c10d/ProcessGroupNCCL.cpp:374, internal error
```

原因分析

可能原因如下：

- 原因1：未设置环境变量NCCL_IB_TC、NCCL_IB_GID_INDEX、NCCL_IB_TIMEOUT，因此会导致通信速度慢且不稳定，最后造成IB通信断连，偶发上述现象。
- 原因2：NCCL_SOCKET_IFNAME设置错误。当用户的NCCL版本低于2.14时，则需要手动设置NCCL_SOCKET_IFNAME环境变量。

处理方法

- 针对原因1，需要在代码中补充如下环境变量。

  ```
  import os
  os.environ["NCCL_IB_TC"] = "128"
  os.environ["NCCL_IB_GID_INDEX"] = "3"
  os.environ["NCCL_IB_TIMEOUT"] = "22"
  ```

- 针对原因2，需要在代码中设置环境变量NCCL_SOCKET_IFNAME。

  ```
  import os
  os.environ["NCCL_SOCKET_IFNAME"] = "eth0"
  ```

  说明

  只有当用户的NCCL版本低于2.14时，才需要进行以上设置。

4.7.11 日志提示“Runtimeerror: Dataloader worker (pid 46212) is killed by signal: Killed BP”

问题现象

训练作业日志运行出现如下报错：Runtimeerror: Dataloader worker (pid 46212) is killed by signal: Killed BP。

原因分析

由于batch size过大，导致Dataloader进程退出。

处理方法

请调小batch size的数值。

4.7.12 日志提示“AttributeError: 'NoneType' object has no attribute 'dtype'”

问题现象

代码在Notebook的keras镜像中可以正常运行，在训练模块使用tensorflow.keras训练报错时，出现如下报错：AttributeError: 'NoneType' object has no attribute 'dtype'。

原因分析

训练镜像的numpy版本与Notebook中不一致。

处理方法

在代码中打印出numpy的版本，查看是否为1.18.5版本，如果非该版本号则在代码开始处执行：

```
import os
os.system('pip install numpy==1.18.5')
```

如果依旧有报错情况，将以上代码修改为：

```
import os
os.system('pip install numpy==1.18.5')
os.system('pip install keras==2.6.0')
os.system('pip install tensorflow==2.6.0')
```

4.7.13 日志提示“No module name 'unidecode'”

问题现象

从mindspore开源gitee中master分支下载的tacotron2模型，修改配置文件后上传ModelArts准备训练，日志报错提示：No module name 'unidecode'。

原因分析

requirements.txt的Unidecode名字写错了，应该把U改成小写，所以导致训练作业的环境没有装上unidecode模块。

处理方法

将requirements.txt中的Unidecode改为unidecode。

建议与总结

您可以在训练代码里添加一行：

```
os.system('pip list')
```

然后运行训练作业，查看日志中是否有所需要的模块。

4.7.14 分布式 Tensorflow 无法使用“tf.variable”

问题现象

多机或多卡使用“tf.variable”会造成以下错误：

```
WARNING:tensorflow:Gradient is None for variable:v0/tower_0/UNET_v7/sub_pixel/Variable:0.Make sure this variable is used in loss computation.
```

原因分析

分布式Tensorflow不能使用“tf.variable”要使用“tf.get_variable”。

处理方法

请您将“启动文件”中的“tf.variable”替换为“tf.get_variable”。

4.7.15 MXNet 创建 kvstore 时程序被阻塞，无报错

问题现象

使用**kv_store = mxnet.kv.create('dist_async')**方式创建“kvstore”时程序被阻塞。如，执行如下代码，如果无法输出“end”，表明程序阻塞。

```
print('start')
kv_store = mxnet.kv.create('dist_async')
print('end')
```

原因分析

worker阻塞的原因可能是连不上server。

处理方法

将如下代码放在“启动文件”里“import mxnet”之前可以看到节点间相互通信状态，同时ps能够重新发送。

```
import os
os.environ['PS_VERBOSE'] = '2'
os.environ['PS_RESEND'] = '1'
```

其中，“os.environ['PS_VERBOSE'] = '2'”为打印所有的通信信息。
“os.environ['PS_RESEND'] = '1'”为在“PS_RESEND_TIMEOUT”毫秒后没有收到ACK消息，Van实例会重发消息。

4.7.16 日志出现 ECC 错误，导致训练作业失败

问题现象

训练作业日志运行出现如下报错：RuntimeError: CUDA error: uncorrectable ECC error encountered

原因分析

由于ECC错误，导致作业运行失败。

处理方法

当ECC错误且计数超过64时，系统会自动隔离故障节点，重启训练作业确认故障是否解决。如果未隔离的节点导致训练作业再次失败或卡死，请联系技术支持处理。

4.7.17 超过最大递归深度导致训练作业失败

问题现象

ModelArts训练作业报错：

```
RuntimeError: maximum recursion depth exceeded in __instancecheck__
```

原因分析

递归深度超过了Python默认的递归深度，导致训练失败。

处理方法

如果超过最大递归深度，建议您在启动文件中增大递归调用深度，具体操作如下：

```
import sys
sys.setrecursionlimit(1000000)
```

4.7.18 使用预置算法训练时，训练失败，报“bndbox”错误

问题现象

使用预置算法创建训练作业，训练失败，日志中出现如下报错。

```
KeyError: 'bndbox'
```

原因分析

用于训练的数据集中，使用了“非矩形框”标注。而预置使用算法不支持“非矩形框”标注的数据集。

处理方法

此问题有两种解决方法：

- 方法1：使用常用框架自行编码开发模型，支持“多边形”标注的数据集。
- 方法2：修改数据集，使用矩形标注。然后再启动训练作业。

4.7.19 训练作业进程异常退出

问题现象

训练作业运行失败，日志中出现如下类似报错：

```
[Modelarts Service Log]Training end with return code: 137
```

原因分析

日志显示训练进程的退出码为137。训练进程表示用户的代码启动后的进程，所以这里的退出码是用户的训练作业代码返回的。常见的错误码还包括247、139等。

- 退出码137或者247

  可能是内存溢出造成的。请减少数据量、减少batch_size，优化代码，合理聚合、复制数据。

  说明

  请注意，数据文件大小不等于内存占用大小，需仔细评估内存使用情况。

- 退出码139

  请排查安装包的版本，可能存在包冲突的问题。

排查办法

根据错误信息判断，报错原因来源于用户代码。

您可以通过以下两种方式排查：

- 线上环境调试代码（仅适用于非分布式代码）
  a. 在开发环境（notebook）申请相同规格的开发环境实例。
  b. 在notebook调试用户代码，并找出问题的代码段。
  c. 通过关键代码段 + 退出码尝试去搜索引擎寻找解决办法。，
- 通过训练日志排查问题
  a. 通过日志判断出问题的代码范围。
  b. 修改代码，在问题代码段添加打印，输出更详细的日志信息。
  c. 再次运行作业，判断出问题的代码段。

4.7.20 训练作业进程被 kill

问题现象

用户进程被Kill表示用户进程因外部因素被Kill或者中断，表现为日志中断。

原因分析

- CPU软锁

  在解压大量文件可能会出现此情况并造成节点重启。可以适当在解压大量文件时，加入sleep。比如每解压1w个文件，就停止1s。
- 存储限制

  根据规格情况合理使用数据盘，数据盘大小请参考**训练环境中不同规格资源大小**。
- CPU过载

  减少线程数。

排查办法

根据错误信息判断，报错原因来源于用户代码。

您可以通过以下两种方式排查：

- 线上环境调试代码（仅适用于非分布式代码）
  a. 在开发环境（notebook）申请相同规格的开发环境实例。
  b. 在notebook调试用户代码，并找出问题的代码段。
  c. 通过关键代码段 + 退出码尝试去搜索引擎寻找解决办法。，
- 通过训练日志排查问题
  a. 通过日志判断出问题的代码范围。
  b. 修改代码，在问题代码段添加打印，输出更详细的日志信息。
  c. 再次运行作业，判断出问题的代码段。

4.8 训练作业运行失败

4.8.1 训练作业运行失败排查指导

问题现象

训练作业的“状态”出现“运行失败”的现象。

原因分析及处理方法

- 查看训练作业的“日志”，出现报错“MoxFileNotExistsException(resp, 'file or directory or bucket not found.')”。
  - 原因：Moxing在进行文件复制时，未找到train_data_obs目录。
  - 处理建议：修改train_data_obs目录为正确地址，重新启动训练作业。

须知

另外在Moxing下载OBS对象过程中，不要删除相应OBS目录下的对象，否则Moxing在下载到被删除的对象时会下载失败。

- 查看训练作业的“日志”，出现报错“CUDA capability sm_80 is not compatible with the current PyTorch installation.The current PyTorch install supports CUDA capabilities sm_37 sm_50 sm_60 sm_70'”。
  - 原因：训练作业使用的镜像CUDA版本只支持sm_37、sm_50、sm_60和sm_70的加速卡，不支持sm_80。
  - 处理建议：使用自定义镜像创建训练作业，并安装高版本的cuda以及对应的PyTorch版本。
- 查看训练作业的“日志”，出现报错“ERROR:root:label_map.pbtxt cannot be found. It will take a long time to open every annotation files to generate a tmp label_map.pbtxt.”。
  - 如果使用的是AI Gallery订阅的算法，建议先检查数据的标签是否有问题。
  - 如果使用的是物体检测类算法，建议检查数据的label框是否为非矩形。

    说明

    物体检测类算法仅支持矩形label框。
- 查看训练作业的“日志”，出现报错“RuntimeError: The server socket has failed to listen on any local network address. The server socket has failed to bind to [::]:29500 (errno: 98 - Address already in use). The server socket has failed to bind to 0.0.0.0:29500 (errno: 98 - Address already in use).”。
  - 原因：训练作业的端口号有冲突。
  - 处理建议：更改代码中的端口号，重启训练作业。
- 查看训练作业的“日志”，出现报错“WARNING: root: Retry=7, Wait=0.4, Times tamp=1697620658.6282516”。
  - 原因：Moxing版本太低。
  - 处理建议：联系技术支持将Moxing版本升级至2.1.6及以上版本。

4.8.2 训练作业运行失败，出现 NCCL 报错

问题现象

训练作业的状态“运行失败”，查看训练作业的“日志”，存在NCCL的报错，例如“NCCL timeout”、“RuntimeError: NCCL communicator was aborted on rank 7”、“NCCL WARN Bootstrap : no socket interface found”或“NCCL INFO Call to connect returned Connection refused, retrying”。

原因分析

NCCL是一个提供GPU间通信原语的库，实现集合通信和点对点发送/接收原语。当训练作业出现NCCL的报错时，可以通过调整NCCL的环境变量尝试解决问题。

处理步骤

1. 进入状态“运行失败”的训练作业详情页，单击“日志”页签，查看NCCL报错。
   - 如果出现报错“NCCL timeout”或者“RuntimeError: NCCL communicator was aborted on rank 7”，则表示InfiniBand Verbs超时。单击右侧“重建”，重新创建训练作业，设置环境变量“NCCL_IB_TIMEOUT=22”，提交训练作业后等待作业完成。
   - 如果出现报错“NCCL WARN Bootstrap : no socket interface found”或“NCCL INFO Call to connect returned Connection refused, retrying”，则

表示NCCL无法找到通信网卡或者是无法正常访问IP地址。需要排查训练代码中是否有设置NCCL_SOCKET_IFNAME环境变量，该环境变量由系统自动注入，训练代码中无需设置。训练代码去除NCCL_SOCKET_IFNAME环境变量设置逻辑后，单击右侧“重建”，重新创建训练作业，提交训练作业后等待作业完成。

2. 等待训练作业是否变成“已完成”状态。
   - 是，故障处理完成。
   - 否，则联系技术支持排查节点状态。

建议与总结

- 环境变量NCCL_SOCKET_IFNAME用于指定通信的网卡名称。“NCCL_SOCKET_IFNAME=eth0”表示仅使用eth0网卡通信。该环境变量由系统自动注入，由于通信网卡名称不固定，因此训练代码不应默认设置该环境变量。
- 环境变量NCCL_IB_TIMEOUT用于控制InfiniBand Verbs超时。NCCL使用的默认值为18，取值范围是1~22。

4.8.3 自定义镜像训练作业失败定位思路

问题现象

使用自定义镜像训练作业时，训练失败。

定位思路

1. 确定镜像来源
   - 确认该自定义镜像的基础镜像是否来源于ModelArts提供的基础镜像，推荐用户使用ModelArts的基础镜像构建自定义镜像，具体请参见**使用ModelArts的基础镜像构建新的训练镜像**。
   - 如镜像来源于第三方，设法找到自定义镜像的制作者咨询，制作者一般对镜像如何使用更加了解。
2. 确定自定义镜像大小

   自定义镜像的大小推荐15GB以内，最大不要超过资源池的容器引擎空间大小的一半。镜像过大会直接影响训练作业的启动时间。

   ModelArts公共资源池的容器引擎空间为50G，专属资源池的容器引擎空间的默认为50G，支持在创建专属资源池时自定义容器引擎空间。
3. 确定错误类型
   - 提示找不到文件等错误，请参见**训练作业日志中提示“No such file or directory”**。
   - 提示找不到包等错误，请参见**训练作业日志中提示“No module named .*”**。
   - Ascend启动脚本和初始化脚本问题。

     确认相关脚本是否来源于官方文档并且是否严格按照官方文档使用。比如确认脚本名称是否正常、脚本路径是否正常。
   - 驱动版本与底层驱动不兼容

     当对自定义镜像的驱动进行升级时，请确定底层驱动是否兼容。当前支持哪种驱动版本，请从**基础镜像**中获取。

- 文件权限不足

  该问题可能为自定义镜像的用户与作业容器的用户不同导致的。请修改dockerfile文件：

  ```
  RUN if id -u ma-user > /dev/null 2>&1 ; \
  then echo 'MA 用户已存在' ; \
  else echo 'MA 用户不存在' && \
  groupadd ma-group -g 1000 && \
  useradd -d /home/ma-user -m -u 1000 -g 1000 -s /bin/bash ma-user ; fi && \
  chmod 770 /home/ma-user && \
  chmod 770 /root && \
  usermod -a -G root ma-user
  ```

- 其他现象，可以在已有的**训练故障案例**查找。

建议与总结

用户使用自定义镜像训练作业时，建议按照**训练作业自定义镜像规范**制作镜像。文档中同时提供了端到端的示例供用户参考。

4.8.4 使用自定义镜像创建的训练作业一直处于运行中

问题现象

使用自定义镜像创建训练作业，训练作业的“状态”一直处于“运行中”。

原因分析及处理办法

日志打印如下内容，表示自定义镜像的CPU架构与资源池节点的CPU架构不一致。

```
standard_init_linux.go:215: exec user process caused "exec format error"
libcontainer: container start initialization failed: standard_init_linux.go:215: exec user process caused "exec format error"
```

常见场景为使用自定义镜像创建作业时选择的资源类型和规格错误。例如，自定义镜像是ARM CPU架构，应选用NPU规格的资源，却使用了X86 CPU/X86 GPU规格的资源。

4.8.5 使用自定义镜像创建训练作业找不到启动文件

问题现象

使用自定义镜像创建训练作业，出现如下报错，提示找不到运行的主文件：no such file or directory。

原因分析

根据报错提示可以判断是运行命令的启动文件目录不正确导致运行失败。

处理方法

需要排查执行命令的启动文件目录是否正确，具体操作如下：

在ModelArts管理控制台，使用训练的自定义镜像创建训练作业时，“创建方式”选择“自定义算法”，“启动方式”选择“自定义”。

例如，当训练代码启动脚本在OBS路径为“obs://bucket-name/app/code/train.py”，创建作业时配置代码目录为“/bucket-name/app/code/”。则代码目录配

置完成后，执行如下命令，那么“run_train.sh”将选中的“code”文件夹下载到训练容器的“/home/ma-user/modelarts/user-job-dir”目录中。

```
bash /home/ma-user/modelarts/user-job-dir/run_train.sh  #训练自定义镜像-预置命令场景
```

运行命令就可以设置为：

```
bash /home/ma-user/modelarts/user-job-dir/run_train.sh python /home/ma-user/modelarts/user-job-dir/code/train.py {python_file_parameter}  #训练自定义镜像-预置命令场景
```

4.8.6 训练作业的监控内存指标持续升高直至作业失败

问题现象

训练作业的“状态”为“运行失败”。

原因分析

训练作业的监控内存指标持续升高导致最后训练作业失败。

处理步骤

1. 查询训练作业的日志和监控信息，是否存在明确的OOM报错信息。
   - 是，训练作业的日志里存在OOM报错，执行**2**。
   - 否，训练作业的日志里没有OOM报错，但是存在监控指标异常，执行**3**。
2. 排查训练代码是否存在不断占用资源的代码，使得资源未被合理使用。
   - 是，优化代码，等待作业运行正常。
   - 否，提高训练作业使用的资源规格或者联系技术支持。
3. 重启训练作业，使用CloudShell登录训练容器监控内存指标，确认是否有突发性的内存增加现象。
   - 是，排查内存突发增加的时间点附近的训练作业日志，优化对应的代码逻辑，减少内存申请。
   - 否，提高训练作业使用的资源规格或者联系技术支持。

4.9 专属资源池创建训练作业

4.9.1 创建训练作业界面无云存储名称和挂载路径排查思路

问题现象

创建训练作业界面没有云存储名称和挂载路径这两个选项。

原因分析

用户的专属资源池没有进行网络打通，或者用户没有创建过SFS。

处理方法

在专属资源池列表中，单击资源池“ID/名称”，进入详情页。单击右上角“配置NAS VPC”，检查是否开启了NAS VPC。详情页面的“NAS VPC名称”和“NAS 子网ID”如果为空则证明没有开启，单击右上角配置NAS VPC即可。

如果单击开启后报错，可能是由于对应的VPC已经创建了对等连接，删除对等连接即可。

4.9.2 创建训练作业时出现“实例挂卷失败”的事件

问题现象

训练作业的状态一直在“创建中”，查看训练作业的“事件”，有异常信息“实例挂卷失败”，详情为“Unable to mount volumes for pod xxx ... list of unmounted volumes=[nfs-x]”。

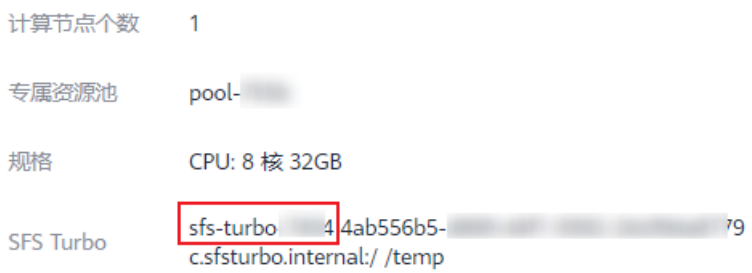
原因分析

用户账号下的SFS Turbo所在的VPC网络需要与专属资源池所在的网络打通，运行于该专属资源池的训练作业才能正常挂载SFS。因此，当训练作业挂载SFS失败时，可能是网络不通导致的。

处理步骤

1. 进入训练作业详情页，在左侧获取SFS Turbo的名称。

   图 4-8 获取 SFS Turbo 的名称

2. 登录弹性文件服务SFS控制台，在SFS Turbo列表找到训练作业挂载的SFS Turbo，单击名称进入详情页。获取VPC信息、安全组信息和endpoint信息。
   - VPC信息：SFS Turbo详情页的“虚拟私有云”。
   - 安全组信息：SFS Turbo详情页的“安全组”。
   - endpoint信息：SFS Turbo详情页的“共享路径”，去除“:/”即为sfs-turbo-endpoint。例如共享路径为“4ab556b5-d689-44f1-9302-24c09daxxxxc.sfsturbo.internal:/”，则sfs-turbo-endpoint为“4ab556b5-d689-44f1-9302-24c09daxxxxc.sfsturbo.internal”。
3. 查看SFS Turbo的VPC网段是否满足如下2个条件。

   条件一：SFS Turbo网段不能与192.168.20.0/24重叠，否则会和专属资源池的网段发生冲突，因为专属资源池的默认网段为192.168.20.0/24。专属资源池实际使用的网段可以在资源池的详情页面查看“网络”获取。

条件二：SFS Turbo网段不能与172网段重叠，否则会和容器网络发生冲突，因为容器网络使用的是172网段。

- 如果不满足条件，则修改SFS Turbo的VPC网段，推荐网段为10.X.X.X。具体操作请参见**修改虚拟私有云网段**。
- 如果满足条件，则继续下一步。

4. 查看SFS Turbo的VPC网段的安全组是否被限制了。

   在所选专属资源池中新建一个未挂载的SFS Turbo的训练作业，当训练作业处于“运行中”时，通过Cloud Shell功能登录训练作业worker-0实例，使用**curl {sfs-turbo-endpoint}:{port}**命令检查port是否正常打开，SFS Turbo所需要入方向的端口号为111、445、2049、2051、2052、20048，具体请参见**创建文件系统**的“安全组”参数。Cloud Shell功能的操作指导请参见**使用CloudShell登录训练容器**。

   - 是，则修改安全组的配置，具体操作请参见**修改安全组规则**。
   - 否，则继续下一步。

5. 确认SFS Turbo是否存在异常。

   新建一个和SFS Turbo在同一个网段的ECS，用ECS去挂载SFS Turbo，如果挂载失败，则表示SFS Turbo异常。

   a. 是，联系SFS服务的技术支持处理。
   b. 否，联系ModelArts的技术支持处理。

4.10 训练作业性能问题

4.10.1 训练作业性能降低

问题现象

使用ModelArts平台训练算法训练耗时增加。

原因分析

可能存在如下原因：

1. 平台上的代码经过修改优化、训练参数有过变更。
2. 训练的GPU硬件工作出现异常。

处理方法

1. 请您对作业代码进行排查分析，确认是否对训练代码和参数进行过修改。
2. 检查资源分配情况（cpu/mem/gpu/snt9/infiniband）是否符合预期。
3. 通过CloudShell登录到Linux工作页面，检查GPU工作情况：
   - 通过输入“nvidia-smi”命令，查看GPU工作是否异常。
   - 通过输入“nvidia-smi -q -d TEMPERATURE”命令， 查看TEMP参数是否存在异常， 如果温度过高，会导致训练性能下降。

5 推理部署

5.1 模型管理

5.1.1 创建模型失败，如何定位和处理问题?

问题定位和处理

创建模型失败有两种场景：创建模型时直接报错或者是调用API报错和创建模型任务下发成功，但最终模型创建失败。

1. 创建模型时直接报错或者是调用API报错。一般都是输入参数不合法导致的。您可以根据提示信息进行排查修改即可。
2. 创建模型任务下发成功，但最终模型创建失败。需要从以下几个方面进行排查：
   - 在模型详情页面，查看“事件”页签中的事件信息。根据事件信息分析模型失败原因，进行处理。
   - 如果模型状态为“构建失败”，可以在模型详情页面，查看“事件”页签中的“查看构建日志”。构建日志中有对应的构建镜像失败的详细原因，根据构建失败的原因进行排查处理。

图 5-1 查看构建日志

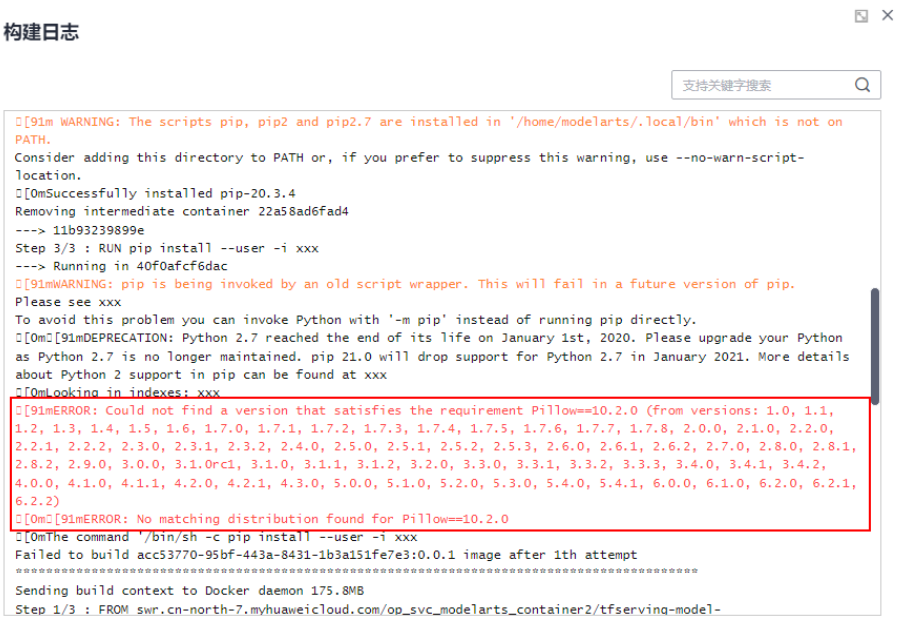
常见问题

1. 模型文件目录下不能出现dockerfile文件；
   “查看构建日志”中显示“Not only a Dockerfile in your OBS path, please make sure, The dockerfile list”，表示dockerfile文件目录有问题，模型文件目录下不能出现dockerfile文件，需要去掉模型文件目录下存在dockerfile文件。

   图 5-2 构建日志：dockerfile 文件目录有问题

2. pip软件包版本不匹配，需要修改为日志中打印的存在的版本。

   图 5-3 pip 版本不匹配



3. 构建日志中出现报错：“exec /usr/bin/sh: exec format error”。

   这种报错一般是因为所用镜像系统引擎和构建镜像的系统引擎不一致引起的，例如使用的是x86的镜像却标记的是arm的系统架构。

   可以通过查看模型详情看到配置的系统运行架构。

5.1.2 导入模型提示该账号受限或者没有操作权限

问题现象

在导入AI应用时，提示用户账号受限。

原因分析

提示用户账号受限，常见原因有如下几种：

1. 导入模型账号欠费导致被冻结；
2. 导入模型账号没有对应工作空间的权限；
3. 导入模型账号为子账号，主账号没有给子账号赋予模型相关权限。

须知

权限说明请参见：**策略及授权项说明**；

处理方法

1. 确认是账号欠费冻结，补交对应欠费，等待账号解冻即可；
2. 如果是导入模型没有对应的工作权限，可以参考**创建自定义策略**对相应账号赋予导入模型相关权限。

5.1.3 用户创建模型时构建镜像或导入文件失败

问题现象

- 用户创建模型时，构建镜像失败，失败日志中提示下载obs文件失败（Get object size from OBS failed！）。

  图 5-4 下载 obs 文件失败

  ```
  #================================ download_obs_file ================================
  Successfully to download file cnnorth7-infer-model1/2d3d3591-1aaa-49e8-af22-483d0a6b31bc/Dockerfile from OBS
  Successfully to download file cnnorth7-infer-model1/2d3d3591-1aaa-49e8-af22-483d0a6b31bc/model/config.json from
  OBS
  Get object size from OBS failed! errorCode: None, errorMsg: None
  Download directory from OBS failed! Exception: (None, None)
  Download file from OBS failed! Exception: ('Get object size from OBS failed! ', OBSException(None, None))
  Download directory from OBS failed! Exception: ('Download file from OBS failed! ', Exception('Get object size
  from OBS failed! ', OBSException(None, None)))
  Retry for the 1 th times
  ```

- 用户创建模型时，事件提示：复制模型文件失败，请检查OBS权限是否正常（Failed to copy model file due to obs exception. Please Check your obs access right.）或用户%s没有OBS的obs:object:PutObjectAcl权限（User %s does not have obs:object:PutObjectAcl permission.）。

图 5-5 复制模型文件失败

原因分析

由于ModelArts的使用权限依赖OBS服务的授权，需要为用户授予OBS的系统权限。子用户的IAM权限是由其主用户设置的，如果主用户没有赋予OBS的putObjectAcl权限即会导致创建模型构建失败。

处理方法

说明

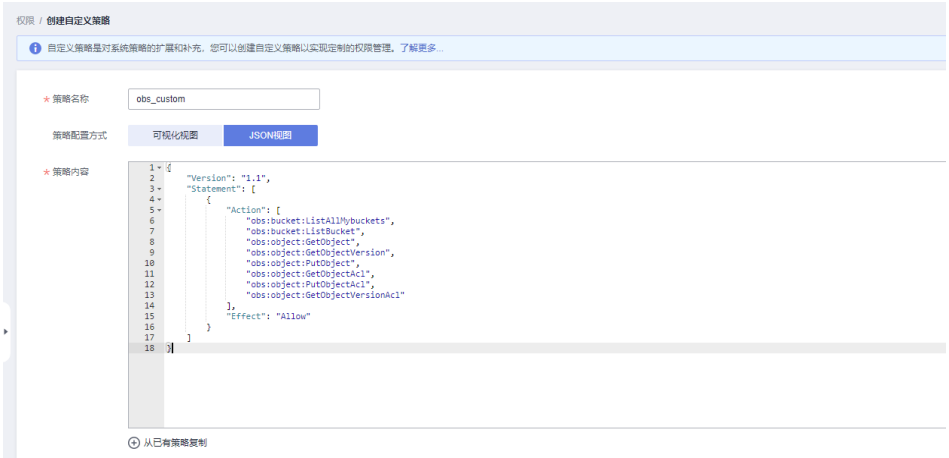
了解ModelArts依赖的OBS权限自定义策略，请参见**ModelArts依赖的OBS权限自定义策略样例**。

在统一身份认证服务为用户增加自定义策略权限。详细操作请参见**创建自定义策略**。

1. 登录“统一身份认证服务”控制台，左侧菜单选择“权限管理 > 权限”，单击右上角“创建自定义策略”，创建自定义策略权限。

图 5-6 统一身份认证服务添加权限

图 5-7 创建自定义策略



权限内容如下：

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Action": [
                "obs:bucket:ListAllMybuckets",
                "obs:bucket:ListBucket",
                "obs:object:GetObject",
                "obs:object:GetObjectVersion",
                "obs:object:PutObject",
                "obs:object:GetObjectAcl",
                "obs:object:PutObjectAcl",
                "obs:object:GetObjectVersionAcl"
            ],
            "Effect": "Allow"
        }
    ]
}
```

2. 在子用户所属用户组中添加该自定义策略权限。

   在用户组页面，单击子用户所属用户组的名称，进入用户组详情页。

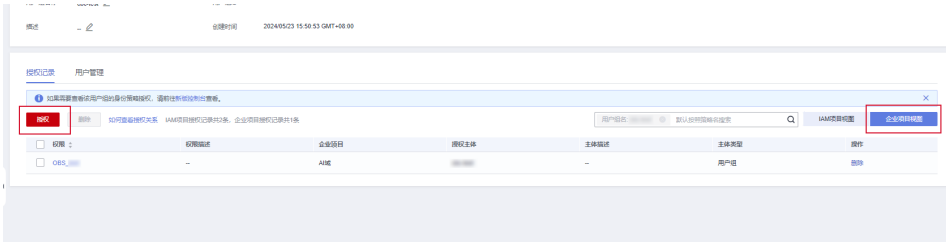
图 5-8 进入用户组详情

在授权记录页签下，单击“授权”，选择您刚才创建的自定义策略及授权方案。

由于OBS服务是全局级服务，无法指定区域项目进行授权，如果需要根据项目进行权限管理，请在选择授权方案选择“指定企业项目资源”。

成功授权后，您可在“企业项目视图”中，看到权限及对应的授权范围。

图 5-9 子用户添加权限



5.1.4 创建模型时，OBS 文件目录对应镜像里面的目录结构是什么样的?

问题现象

创建模型时，元模型来源指定的OBS目录下存放了自定义的文件和文件夹，都会复制到镜像中去。复制进去的路径是什么，怎么读取对应的文件或者文件夹里面的内容?

原因分析

通过OBS导入模型时，ModelArts会将指定的OBS目录下的所有文件和文件夹复制到镜像中的指定路径下，镜像内路径可以通过self.model_path获取。

处理方法

获取镜像内的路径方法见**模型推理代码编写说明**。

5.1.5 通过 OBS 导入模型时，如何编写打印日志代码才能在 ModelArts 日志查询界面看到日志

问题现象

用户通过OBS导入模型时，选择使用基础镜像，用户自己编写了部分推理代码实现自己的推理逻辑，出现故障后希望通过故障日志排查定位故障原因，但是通过logger打印日志无法在“在线服务”的日志中查看到部分内容。

原因分析

推理服务的日志如果需要显示出来，需要代码中将日志打印到Console控制台。当前推理基础镜像使用的python的log模块，采用的是默认的日志级别Warning，即当前只有Warning级别的日志可以默认查询出来。如果想要指定INFO等级的日志能够查询出来，需要在代码中指定logger的输出日志等级为INFO级别。

处理方法

在推理代码所在的py文件中，指定日志输出到Console的默认级别为INFO级别，确保将对应级别的日志打印出来。参考代码如下：

```
import log
# 创建一个logger
logger = log.getLogger(__name__)
# 测试日志输出
logger.info("This is an info message")
```

5.1.6 通过 OBS 创建模型时，构建日志中提示 pip 下载包失败

问题现象

通过OBS创建模型构建失败，查看构建日志，提示pip下载包失败。如下载numpy 1.16版本失败。

原因分析

一般下载包失败时，可能有如下几个原因：

1. pip源中不存在该包，当前默认pip源为pypi.org中的包，请在pypi.org中查看是否有对应版本的包并查看包安装限制。
2. 下载的包与对应基础镜像架构不匹配，如arm系统下载了x86的包，python2版本的pip下载了python3的包。具体基础镜像运行环境请参见**推理基础镜像列表**。
3. 安装pip包有先后依赖关系。

处理方法

1. 到pypi.org上查询依赖的待安装包是否存在，如果不存在则建议使用whl包进行安装（将待安装的whl包放到模型所在的OBS目录下）。
2. 查看待安装包的安装限制和前置依赖等，排查是否满足相关要求。
3. 如果包有依赖关系，请参考**导入模型时，模型配置文件中的安装包依赖参数如何编写？**章节配置包的先后依赖关系。

5.1.7 通过自定义镜像创建模型失败

问题现象

通过用户自定义镜像创建模型失败。

原因分析

可能原因如下：

- 导入模型使用的镜像地址不合法或实际镜像不存在
- 用户给ModelArts的委托中没有SWR相关操作权限
- 用户为子账号，没有主账号SWR的权限
- 使用的是非自己账号的镜像
- 使用的镜像为公开镜像

处理方法

1. 到SWR检查下对应的镜像是否存在，对应镜像的镜像地址是否和实际地址一致，大小写，拼写等是否一致。
2. 检查用户给ModelArts的委托中是否有SWR的权限，可以在权限管理中查看对应用户的授权内容，查看授权详情。如果没有对应权限，需要到统一身份认证服务给对应委托中加上对应权限。

图 5-10 权限管理

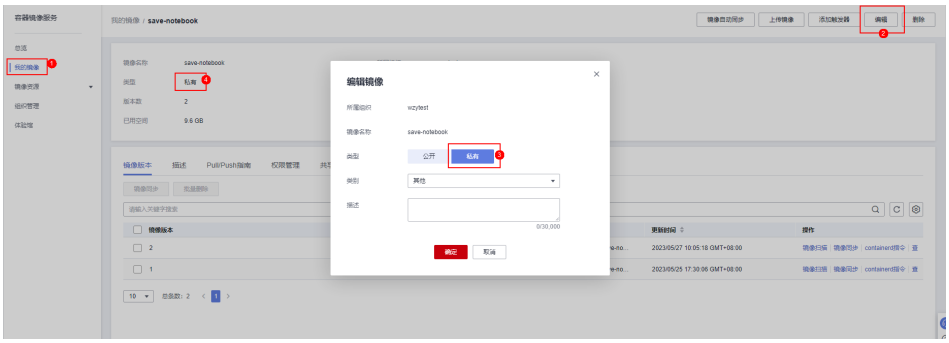
图 5-11 查看权限详情和去 IAM 修改委托权限

图 5-12 给委托添加授权

3. 将镜像设置成私有镜像

   登录容器镜像服务（SWR），左侧导航栏选择“我的镜像”，查看镜像详情，单击右上角“编辑”按钮，把镜像类型修改为“私有”。

图 5-13 修改镜像类型为私有



5.1.8 导入模型后部署服务，提示磁盘不足

问题现象

用户在导入模型后，部署服务时，提示磁盘空间不足：“No space left on device”。

原因分析

ModelArts部署使用的是容器化部署，容器运行时有空间大小限制，当用户的模型文件或者其他自定义文件，系统文件超过Docker size大小时，会提示镜像内空间不足。

处理方法

公共资源池容器Docker size的大小最大支持50G，专属资源池Docker size的大小最大支持50G。

如果使用的是OBS导入或者训练导入，则包含基础镜像、模型文件、代码、数据文件和下载安装软件包的大小总和。

如果使用的是自定义镜像导入，则包含解压后镜像和镜像下载文件的大小总和。

5.1.9 创建模型成功后，部署服务报错，如何排查代码问题

问题现象

创建模型成功后，部署服务失败，如何定位是模型代码编写有问题。

原因分析

用户自定义镜像或者通过基础镜像导入的模型时，用户自己编写了很多自定义的业务逻辑，这些逻辑有问题将会导致服务部署或者预测失败，需要能够排查出哪里有问题。

处理方法

1. 服务部署失败后，进入服务详情界面，查看服务部署日志，明确服务部署失败原因（用户代码输出需要使用标准输入输出函数，否则输出的内容不会呈现到前端页面日志）。根据日志中提示的报错信息找到对应的代码进行定位。

5.1.10 自定义镜像导入配置运行时依赖无效

问题现象

通过API接口选择自定义镜像导入创建模型，配置了运行时依赖，没有正常安装pip依赖包。

原因分析

自定义镜像导入不支持配置运行时依赖，系统不会自动安装所需要的pip依赖包。

处理方法

重新构建镜像。

在构建镜像的dockerfile文件中安装pip依赖包，例如安装Flask依赖包。

```
# 配置华为云的源，安装 python、python3-pip 和 Flask
RUN cp -a /etc/apt/sources.list /etc/apt/sources.list.bak && \
sed -i "s@http://.*security.ubuntu.com@http://repo.huaweicloud.comxxx@g" /etc/apt/sources.list && \
sed -i "s@http://.*archive.ubuntu.com@http://repo.huaweicloud.comxxx@g" /etc/apt/sources.list && \
```

```
apt-get update && \
apt-get install -y python3 python3-pip && \
pip3 install  --trusted-host https://repo.huaweicloud.comxxx -i https://repo.huaweicloud.comxxx/repository/pypi/simple  Flask
```

5.1.11 通过 API 接口查询模型详情，model_name 返回值出现乱码

问题现象

通过API接口查询模型详情，model_name返回值出现乱码。例如model_name为query_vec_recall_model，但是api接口返回结果是query_vec_recall_model_b。

```
[2022/08/12 00:03:25 GMT+0800][INFO]Execute user name is xxx. user id is 04ef6da71400125321f15c01f1d1xxxx, job id is 6ABxxx
[2022/08/12 00:03:25 GMT+0800][INFO]Request url is https://modelarts.xxx.xxx.com/v1/88exxxta/models?model_name=query_vec_recall_model
[2022/08/12 00:03:25 GMT+0800][INFO]Request query param is nul
[2022/08/12 00:03:25 GMT+0800][INFO]Request method is GET
[2022/08/12 00:03:25 GMT+0800][INFO]Request header is {REST_API_MARK=REST API MARK, User-Agent=Dayu}
[2022/08/12 00:03:26 GMT+0800][INFO]Response body: {"count":3"total_count":0"models":[{"model id":"ca12cbdb-e7eb-4084-9ea3-36c0bd6axxxx","model name":"query_vec_recall_model_b","model_version":"0.0.1","model_type":"TensorFlow".....
```

原因分析

当模型名称包含下划线时，下划线涉及转义处理。

处理方法

需要在请求中增加exact_match参数，且参数值设置为true，确保model_name返回值正确。

5.1.12 导入模型提示模型或镜像大小超过限制

问题现象

在导入模型时，提示模型或镜像大小超过限制。

原因分析

如果使用的是OBS导入或者训练导入，则是基础镜像、模型文件、代码、数据文件和下载安装软件包的大小总和超过了限制。

如果使用的是自定义镜像导入，则是解压后镜像和镜像下载文件的大小总和超过了限制。

处理方法

精简模型或镜像后，重新导入。

5.1.13 导入模型提示单个模型文件超过 5G 限制

问题现象

在导入模型时，提示单个模型文件大小超过5G限制。

原因分析

在不使用动态加载的情况下，系统对单个模型文件的限制大小为5G，超过时无法进行导入。

处理方法

- 精简模型文件后，重新导入。
- 使用动态加载功能进行导入。

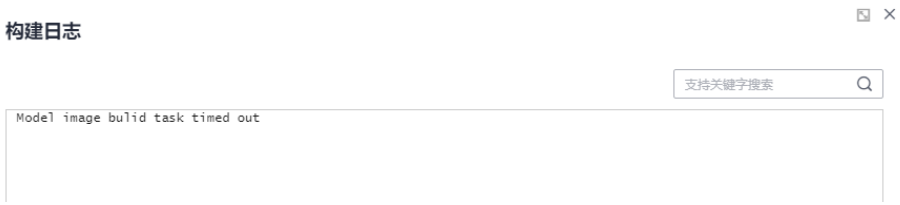
图 5-14 使用动态加载

5.1.14 创建模型失败，提示模型镜像构建任务超时，没有构建日志

问题现象

创建模型失败，构建日志提示超时“Model image build task timed out”，没有详细构建日志。

图 5-15 模型镜像构建任务超时



原因分析

imagePacker构建镜像有超时时间限制，默认值为30min（各区域可能存在差异）。当模型镜像构建时间太长，构建日志最后未能完成构建任务，构建超时中断，即会出现“Model image build task timed out”提示，不显示详细的构建日志。

处理方法

- 预先准备需要编译下载的依赖包，减少依赖包下载和编译的时间。可通过线下wheel包方式安装运行环境依赖。线下wheel包安装，需确保wheel包与模型文件放在同一目录。
- 优化模型代码，提高构建模型镜像的编译效率。

5.2 服务部署

5.2.1 自定义镜像模型部署为在线服务时出现异常

问题现象

在部署在线服务时，部署失败。进入在线服务详情页面，“事件”页签，提示“failed to pull image, retry later”，同时在“日志”页签中，无任何信息。

解决方法

出现此问题现象，通常是因为您部署的模型过大导致的。解决方法如下：

- 精简模型，重新导入模型和部署上线。
- 购买专属资源池，在部署上线为在线服务时，使用专属资源池进行部署。

5.2.2 部署的在线服务状态为告警

问题现象

在部署在线服务时，状态显示为“告警”。

解决方法

使用状态为告警的服务进行预测，可能存在预测失败的风险，请从以下4个角度进行排查，并重新部署。

1. 后台预测请求过多。

   如果您使用API接口进行预测，请检查是否预测请求过多。大量的预测请求会导致部署的在线服务进入告警状态。

2. 业务内存不正常。

   请检查推理代码是否存在内存溢出或者内存泄漏的问题。

3. 模型运行异常。

   请检查您的模型是否能正常运行。例如模型依赖的资源是否故障，需要排查推理日志。

4. 实例pod数量异常 。

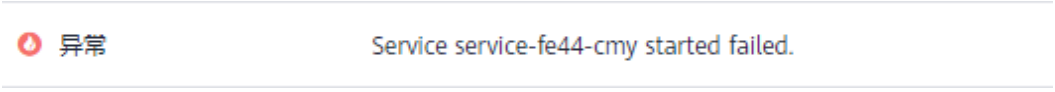
   如果您曾经找过运维人员删除过异常的实例pod，事件中可能会出现告警“服务异常，不正常的实例数为XXX”。在出现这种告警后，服务会自动拉起新的正常实例，从而恢复到正常运行状态。请您耐心等待。

5.2.3 服务启动失败

问题现象

当服务事件中出现如下事件时，表示容器启动失败。

图 5-16 服务启动失败



原因分析

服务启动失败的原因比较多样，可能有如下几种情况：

- **AI应用本身问题，无法启动**
- **镜像中配置的端口错误**
- **健康检查配置有问题**
- **模型推理代码customize_service.py编写有问题**
- **镜像拉取失败**
- **资源不足，服务调度失败**

模型本身问题，无法启动

如果创建模型使用的镜像本身有问题，需要在创建模型之前，参考**从0-1制作自定义镜像并创建AI应用**，确保镜像可以正常启动，并可以在本地curl通返回预期内容。

镜像中配置的端口错误

模型可以正常启动，但是因为镜像中启用的端口非8080，或者镜像启用的端口与创建模型时配置的端口不一致，导致部署服务时register-agent无法与模型通信，超过一定时间后（最长20分钟）认为模型启动失败。

需要检查两个地方：自定义镜像中的代码开放的端口和创建模型界面上配置的端口。确认两处端口保持一致。模型创建界面如果不填端口信息，则ModelArts会默认监听8080端口，即镜像代码中启用的端口必须是8080。

图 5-17 自定义镜像中的代码开放的端口

```
# host must be "0.0.0.0", port must be 8080
if __name__ == '__main__':
    app.run(host="0.0.0.0", port=8080)
```

图 5-18 创建模型界面上配置的端口

健康检查配置有问题

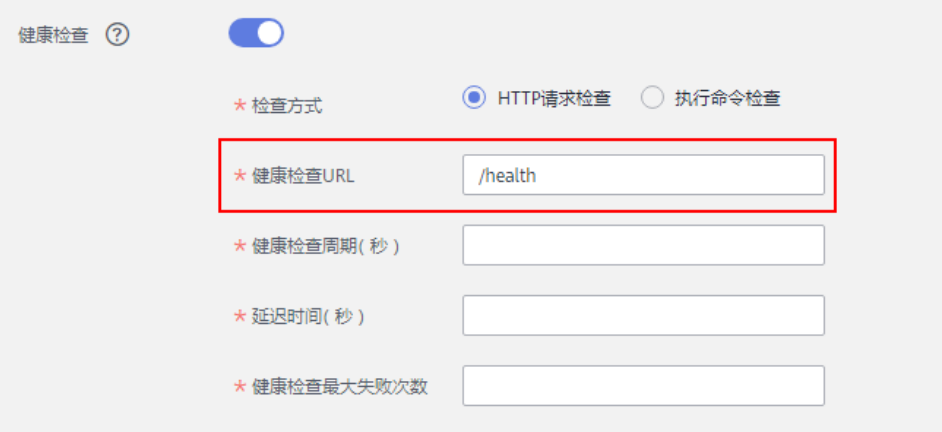
镜像如果配置了健康检查，服务启动失败，从以下两个方面进行排查：

- 健康检查端口是否可以正常工作

  自定义镜像中配置了健康检查，需要在测试镜像时，同步测试健康检查接口是否可以正常工作，具体参考**从0-1制作自定义镜像并创建AI应用**中的本地验证镜像方法。
- 创建模型界面上配置的健康检查地址与实际配置的是否一致

  如果使用的是ModelArts提供的基础镜像创建模型，健康检查URL默认必须为/health。

图 5-19 设置健康检查 URL



模型推理代码 customize_service.py 编写有问题

如果模型推理代码customize_service.py编写有误，可以通过查看服务运行日志，定位具体原因进行修复。

拉取镜像失败

服务启动失败，提示拉取镜像失败，请参考**服务部署、启动、升级和修改时，拉取镜像失败如何处理?**

资源不足，服务调度失败

服务启动失败，提示资源不足，服务调度失败，请参考**服务部署、启动、升级和修改时，资源不足如何处理?**

内存不足

服务启动失败，提示内存不足，请参考**内存不足如何处理?**

5.2.4 服务部署、启动、升级和修改时，拉取镜像失败如何处理?

问题现象

服务部署、启动、升级和修改时，拉取镜像失败。

原因分析

节点磁盘不足，镜像大小过大。

解决方法

1. 首先考虑优化镜像，减小节点磁盘的占用。
2. 优化镜像无法解决问题，请联系系统管理员处理。

5.2.5 服务部署、启动、升级和修改时，镜像不断重启如何处理?

问题现象

服务部署、启动、升级和修改时，镜像不断重启。

原因分析

容器镜像代码错误

解决方法

根据容器日志进行排查，修复代码，重新创建模型，部署服务。

5.2.6 服务部署、启动、升级和修改时，容器健康检查失败如何处理?

问题现象

服务部署、启动、升级和修改时，容器健康检查失败。

原因分析

容器提供的健康检查接口调用失败。容器健康检查接口调用失败，原因可能有两种：

- 镜像健康检查配置问题
- 模型健康检查配置问题

解决方法

根据容器日志进行排查，查看健康检查接口失败的具体原因。

- 镜像健康检查配置问题，需修复代码后重新制作镜像创建模型后部署服务。了解镜像健康接口配置请参考**模型配置文件编写说明**中health参数说明。
- 模型健康检查配置问题，需重新创建模型或者创建模型新版本，配置正确的健康检查，使用新的模型或版本重新部署服务。了解模型健康检查请参考**制作模型镜像并导入**中的“健康检查”参数说明。

5.2.7 服务部署、启动、升级和修改时，资源不足如何处理?

问题现象

启动服务失败，报错：资源不足，服务调度失败。（Schedule failed due to insufficient resources. Retry later.或ModelArts.3976：No resources are available for the selected specification.）

图 5-20 资源不足，服务调度失败

原因分析

- 实例配置的规格过大，CPU或者内存剩余资源不足；（"insufficient CPU" / "insufficient memory"）
- 模型需要的磁盘空间大，磁盘空间不足；（"x node(s) had taint {node.kubernetes.io/disk-pressure: }" / "No space"）

解决方法

在遇到资源不足的情况时，ModelArts会进行三次重试，在服务重试期间，如果有资源释放出来，则服务可以正常部署成功。

如果三次重试后依然没有足够的资源，则本次服务部署失败。参考以下方式解决：

- 如果是在公共资源池部署服务，可等待其他用户释放资源后，再进行服务部署。
- 如果是在专属资源池部署服务，在满足模型需求的前提下，尝试选用更小的容器规格或自定义规格，进行服务部署；
- 如果当前资源池的资源确实不够，也可以考虑将资源池扩容后再进行服务部署。公共资源池扩容，请联系系统管理员。专属资源池扩容，可参考**扩缩容资源池**。
- 如果磁盘空间不够，可以尝试重试，使实例调度到其他节点。如果单实例仍磁盘空间不足，请联系系统管理员，更换合适的规格。

说明

如果是大模型导入的模型部署服务，请确保专属资源池磁盘空间大于1T（1000GB）。

5.2.8 模型使用 CV2 包部署在线服务报错

问题现象

使用CV2包部署在线服务报错。

原因分析

使用OBS导入元模型，会用到服务侧的标准镜像，标准镜像里面没有CV2依赖的so的内容。所以ModelArts不支持从对象存储服务（OBS）导入CV2模型包。

处理方法

需要您把CV2包制作为自定义镜像，上传至容器镜像服务（SWR），选择从容器镜像中导入元模型，部署在线服务。如何制作自定义镜像请参考**从0-1制作自定义镜像并创建AI应用**。

5.2.9 服务状态一直处于“部署中”

问题现象

服务状态一直处于“部署中”，查看模型日志未发现服务有明显错误。

原因分析

一般情况都是模型的端口配置有问题。建议您首先检查创建模型的端口是否正确。

处理方法

模型的端口没有配置，如您在自定义镜像配置文件中修改了端口号，需要在部署模型时，配置对应的端口号，使新的模型重新部署服务。

如何修改默认端口号，请参考**使用自定义镜像创建在线服务，如何修改默认端口**。

5.2.10 服务启动后，状态断断续续处于“告警中”

问题现象

预测流量不大但频繁出现以下报错

- Backend service internal error. Backend service read timed out
- Send the request from gateway to the service failed due to connection refused, please confirm your service is connectable
- Send the request from gateway to the service failed due to connection timeout, please confirm your service is able to process the new request

原因分析

该报错是因为发送预测请求后，服务出现停止后又启动的情况。

处理方法

需要您检查服务使用的镜像，确定服务停止的原因，修复问题。重新创建模型部署服务。

5.2.11 服务部署失败，报错 No Module named XXX

问题现象

服务部署失败，报错：No Module named XXX

原因分析

No Module named XXX，表示模型中没有导入对应依赖模块。

处理方法

依赖模块没有导入，需要您在模型推理代码中导入缺失依赖模块。

例如您的模型是Pytorch框架，部署为在线服务时出现告警：ModuleNotFoundError: No module named ‘model_service.tfserving_model_service’，则需要您在推理代码customize_service.py里使用from model_service.pytorch_model_service import PTServingBaseService。示例代码：

```
import log
from model_service.pytorch_model_service import PTServingBaseService
```

5.2.12 批量服务输入/输出 obs 目录不存在或者权限不足

问题现象

1. 输入输出目录不存在，报如下错误
   ```
   "error_code": "ModelArts.3551",
   "error_msg": "OBS path xxxx does not exist."
   ```
2. 当访问目录权限不足时，报如下错误
   ```
   "error_code": "ModelArts.3567",
   "error_msg": "OBS error occurs because Access Denied."
   ```

原因分析

ModelArts.3551：数据输入或者输出的obs目录不存在

ModelArts.3567：使用的数据输入或者输出obs目录存在，但是当前账号无权限访问

处理方法

ModelArts.3551：到obs检查输入数据目录是否存在，如果不存在，请按照实际需要创建obs目录；如果检查发现目录存在，但依然报同样的错，可以提工单申请技术支持

ModelArts.3567：用户只能访问自己账号下的obs目录，ModelArts在读取其他用户obs下的数据时，需要用户委托权限，没有创建委托，就没有权限使用其他用户obs中的数据。

登录ModelArts控制台，管理控制台，在左侧导航栏中选择“权限管理”，单击“查看权限”，检查是否配置了obs的委托权限。

图 5-21 查看权限

如果检查后已经存在委托，但是仍然无法访问，可以提工单寻求技术支持。

5.2.13 部署在线服务出现报错 No CUDA runtime is found

问题现象

部署在线服务出现报错No CUDA runtime is found，using CUDA_HOME='/usr/local/cuda'。

原因分析

从日志报错信息No CUDA runtime is found分析，是cuda runtime没有找到。

处理方法

建议您按以下步骤排查处理：

1. 确认部署在线服务时是否选择了GPU规格。
2. 在customize_service.py中添加一行代码os.system('nvcc -V)查看该镜像的cuda版本（ customize_service.py编写指导请见**模型推理代码编写说明** ）。
3. 确认该cuda版本与您安装的mmcv版本是否匹配。

说明

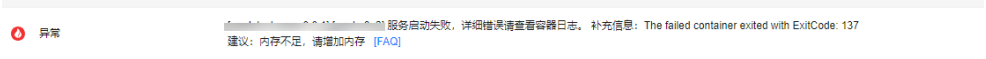
部署时是否需要使用GPU，取决于的模型需要用到CPU还是GPU，以及推理脚本如何编写。

5.2.14 内存不足如何处理?

问题现象

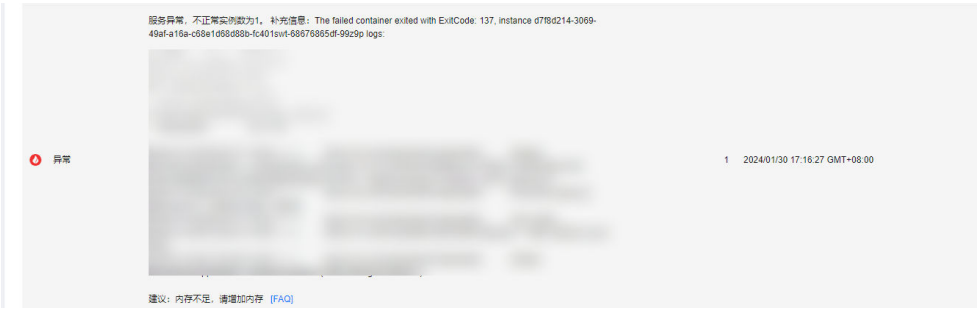
- 在部署或升级在线服务时，如果部署或升级失败，并且在事件中出现如下类似提示。

图 5-22 内存不足提示样例 1



- 运行中服务出现告警时，在事件中出现建议：内存不足，请增加内存。

图 5-23 内存不足提示样例 2



原因分析

- 部署或升级时出现该提示，可能原因是选择的计算节点规格内存太小，无法满足应用部署，请增大内存规格。
- 运行中服务告警中出现该提示，可能代码有问题导致内存溢出或者业务使用量太大导致内存需求增多。

处理方法

- 在部署或升级在线服务时，选择更大内存规格的计算节点。

图 5-24 选择计算节点规格

- 运行中服务出现告警时，需要分析是您的代码是否出现漏洞导致内存溢出、是否因为业务使用量太大需要更多的内存。如果因业务原因需要更多内存，请升级在线服务选择更大内存规格的计算节点。

5.3 服务预测

5.3.1 服务预测失败

问题现象

在线服务部署完成且服务已经处于“运行中”的状态，向服务发起推理请求，预测失败。

原因分析及处理方法

服务预测需要经过客户端、外部网络、APIG、Dispatch、模型服务多个环节。每个环节出现都会导致服务预测失败。

图 5-25 推理服务流程图

1. 出现APIG.XXXX类型的报错，表示请求在APIG（API网关）出现问题而被拦截。
   常见问题请参见**服务预测失败，报错APIG.XXXX**。
   其他被APIG（API网关）拦截的场景：
   - **Method Not Allowed**
   - **请求超时返回Timeout**
2. 出现ModelArts.XXXX类型的报错，表示请求在Dispatcher出现问题而被拦截。
   常见报错：
   - **在线服务预测报错ModelArts.4302**
   - **在线服务预测报错ModelArts.4206**
   - **在线服务预测报错ModelArts.4503**
3. 当使用推理的镜像并且出现MR.XXXX类型的错误时，表示已进入模型服务，一般是模型推理代码编写有问题。
   请根据构建日志报错信息，定位服务预测失败原因，修改模型推理代码后，重新导入模型进行预测。
   经典案例：**在线服务预测报错MR.0105**
4. 出现其他情况，优先检查客户端和外部网络是否有问题。
5. 以上方法均未解决问题，请联系系统管理员。

5.3.2 服务预测失败，报错 APIG.XXXX

请求在APIG（API网关）出现问题被拦截，报错APIG.XXXX。

常见报错：

- **APIG.0101 预测地址错误**
- **APIG.0201 请求体内容过大**
- **APIG.0301 鉴权失败**
- **APIG.1009 AppKey和AppSecret不匹配**

查看更多的APIG（API网关）错误码含义及处理方案可参考API错误码。

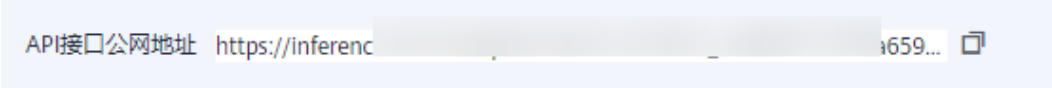
APIG.0101 预测地址错误

当预测的地址有问题时，APIG（API网关）将拦截请求，报错“APIG.0101”：“The API does not exist or has not been published in the environment”，请到在线服务详情界面，“调用指南”页签中获取正确的API接口地址。

说明

如果您在配置文件url中有定义路径，需要在API调用body体中调用路径后拼接自定义路径，例如：您定义url为“/predictions/poetry”，那么在API调用时路径为“{API接口地址}/predictions/poetry”。

图 5-26 获取 API 接口地址



APIG.0201 请求体内容过大

请求体内容过大时，APIG（API网关）会拦截请求，报错“APIG.0201”：“Request entity too large”。请减少预测请求内容后重试。

当使用API调用地址预测时，请求体的大小限制是12MB，超过12MB时，请求会被拦截。

使用ModelArts console的预测页签进行的预测，由于console的网络链路的不同，要求请求体的大小不超过8MB。

图 5-27 请求报错 APIG.0201

APIG.0301 鉴权失败

通过API进行服务预测，或者使用Token进行APP认证，需要获取正确的Token鉴权，当Token不合法时，APIG（API网关）拦截请求，报错“APIG.0301”：“Incorrect IAM authentication information： decrypt token fail”。请获取正确的token填入X-Auth-Token，进行预测。如何获取Token请参考**获取IAM用户Token**。

APIG.1009 AppKey 和 AppSecret 不匹配

当服务预测使用的AppKey和AppSecret不匹配时，报错“APIG.1009”：“AppKey or AppSecret is invalid”。

查询AppKey和AppSecret，使用APP认证访问在线服务，请参考**访问在线服务（APP认证）**。

5.3.3 在线服务预测报错 ModelArts.4206

问题现象

在线服务部署完成且服务已经处于“运行中”的状态，向服务发起推理请求，报错“ModelArts.4206”。

原因分析

ModelArts.4206表示该API的请求流量超过了设定值。为了保证服务的平稳运行，ModelArts对单个API的推理请求流量做了限制，同时为了保证推理服务可以稳定运行在合理区间，ModelArts将限流值设定在一个较高区间。

处理办法

降低API的流量，如果确有超高并发的需求，请提工单处理。

5.3.4 在线服务预测报错 ModelArts.4302

问题现象

在线服务部署完成且服务已经处于“运行中”的状态后，向运行的服务发起推理请求，报错ModelArts.4302。

原因分析及处理方法

服务预测报错ModelArts.4302有多种场景，以下主要介绍两种场景：

1. "error_msg": "Gateway forwarding error. Failed to invoke backend service due to connection refused. "

   出现该报错有两种情况：

   - 流量超过了模型的处理能力。可以考虑降低流量或者增加模型实例数量。
   - 镜像自身有问题。需要单独运行镜像确保镜像本身能正确提供服务。

2. "error_msg"：“Due to self protection, the backend service is disconnected, please wait moment."

   出现该错误，是因为模型报错太多。当模型报错太多时，会触发dispatcher的熔断机制，导致预测失败。建议您检查模型返回结果，处理模型报错问题，可尝试通过调整请求参数、降低请求流量等方式，提高模型调用的成功率。

5.3.5 在线服务预测报错 ModelArts.4503

问题现象

在线服务部署完成且服务已经处于“运行中”的状态后，向运行的服务发起推理请求，报错ModelArts.4503。

原因分析及处理方法

服务预测报错ModelArts.4503有多种场景，常见场景如下：

1. 通信出错

   请求报错：{"error_code":"ModelArts.4503","error_msg":"Failed to respond due to backend service not found or failed to respond"}

   基于高性能考虑，ModelArts会复用同模型服务的连接。根据tcp协议，连接的断开可以由该连接的client端发起，也可以由server端发起。断开连接需要经过四次握手，所以可能会存在作为服务端的模型服务侧发起断开连接，但是该连接正在被作为客户端的ModelArts使用，从而导致通信出错，返回此错误信息。

   如果您使用的是自定义镜像导入的模型，请增大自定义镜像中所使用的web server的keep-alive的参数值，尽量避免由服务端发起关闭连接。如您使用的**Gunicorn**来作为web server，可以通过**Gunicorn**命令的--keep-alive参数来设置该值。其他方式导入的模型，服务内部已做处理。

2. 协议错误

   请求报错：{"error_code":"ModelArts.4503", "error_msg":"Failed to find backend service because SSL error in the backend service, please check the service is https"}

   部署在线服务使用的模型是从容器镜像中导入时，容器调用接口协议填写错误，会导致此错误信息。

出于安全考虑，ModelArts提供的推理请求都是https请求，从容器镜像中选择导入模型时，ModelArts允许使用的镜像提供https或http服务，但必须在“容器调用接口”中明确指定该镜像使用的是https或http服务。如下图所示：

图 5-28 容器调用接口

如果您在“容器调用接口”中选择的结果跟您镜像实际提供的结果不匹配，例如您在这里选择的是https，但镜像里面实际提供的是http，就会遇到上述错误。反之，如果您选择的是http，但镜像里面实际提供的是https，也会遇到类似错误。

您可以创建一个新的模型版本，选择正确的协议（http或者https），重新部署在线服务或更新已有在线服务。

3. 请求预测时间过长

   报错：{"error_code": "ModelArts.4503", "error_msg": "Backend service respond timeout, please confirm your service is able to process the request without timeout. "}及报错：{"error_code": "ModelArts.4503", "error_msg": "Failed to find backend service because response timed out, please confirm your service is able to process the request without timeout. "}

   因APIG（API网关）限制，平台每次请求预测的时间不超过40秒。数据从平台发送到服务，服务预测推理，再将结果返回的时间不超过限制，可以成功返回预测结果。当服务预测的时间过长或者频繁预测导致服务接收不过来请求，即会出现该报错。

   可以通过以下方式解决问题：

   - 服务预测请求内容过大时，会因数据处理慢导致请求超时，优化预测代码，缩短预测时间。
   - 推理速度与模型复杂度强相关，优化模型，缩短预测时间。
   - 扩容实例数或者选择性能更好的“计算节点规格”，例如使用GPU资源代替CPU资源，提升服务处理能力。

4. 服务出错

   报错：{"error_code": "ModelArts.4503","error_msg": "Backend service respond timeout, please confirm your service is able to process the request without timeout. "}

   服务日志输出：

   ```
   [2022-10-24 11:37:31 +0000] [897] [INFO] Booting worker with pid: 897
   [2022-10-24 11:41:47 +0000] [1997] [INFO] Booting worker with pid: 1997
   [2022-10-24 11:41:22 +0000] [1897] [INFO] Booting worker with pid: 1897
   [2022-10-24 11:37:54 +0000] [997] [INFO] Booting worker with pid: 997
   ```

   服务异常进程反复重启导致预测请求无法发送到服务实例。

   可以通过以下方式解决问题：

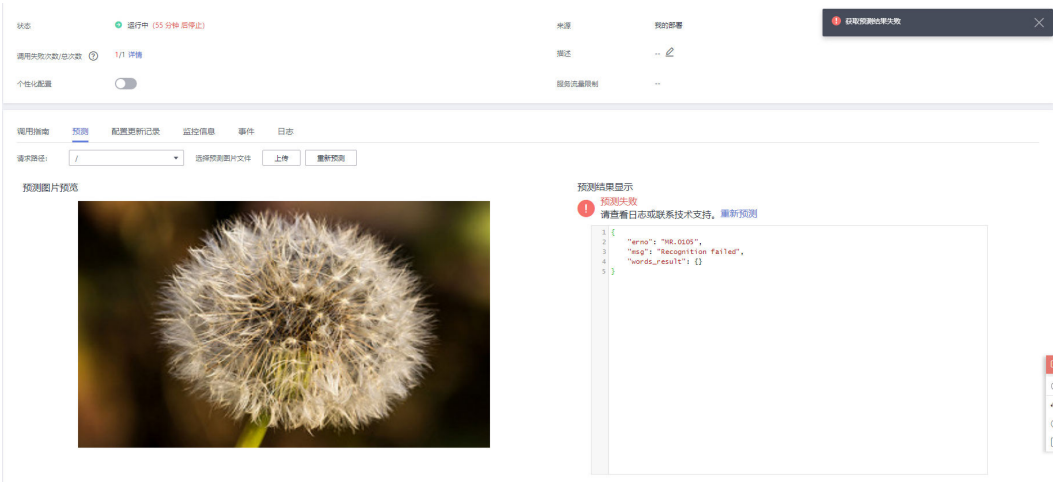
- 缩小预测请求数量看是否问题还复现，如果不复现是因为负载过大导致服务进程退出，需要扩容实例数量或者提升规格。
- 推理代码本身存在错误，请排查推理代码解决。

5.3.6 在线服务预测报错 MR.0105

问题现象

部署为在线服务，服务处于运行中状态，预测时报错：{ "erno": "MR.0105", "msg": "Recognition failed","words_result": {}}。

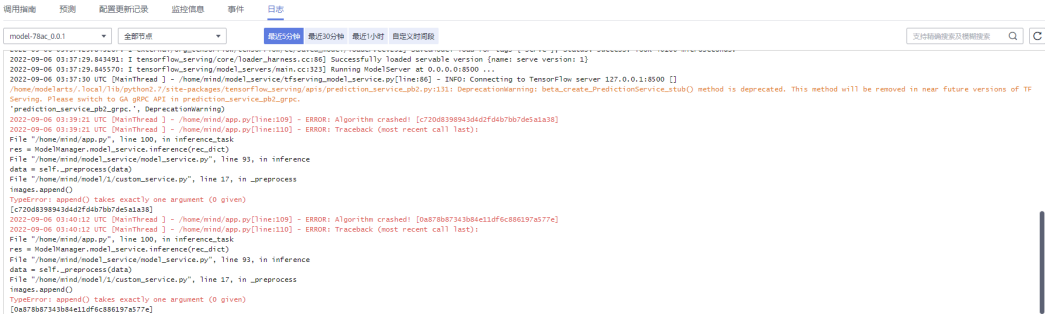
图 5-29 预测报错



原因分析

请在“在线服务”详情页面的日志页签中查看对应的报错日志，分析报错原因。

图 5-30 报错日志



从上图报错日志判断，预测失败是模型推理代码编写有问题。

解决方法

根据日志报错提示，append方法中缺少必填参数，修改模型推理代码文件“customize_service.py”中的代码，给append方法中传入合理的参数。

如需了解更多模型推理代码编写说明，请参考**模型推理代码编写说明**。

5.3.7 在线服务预测报错 ModelArts.2803

问题现象

服务预测报错：Method Not Allowed

原因分析

服务预测默认注册的API需要使用POST方法调用。如您使用了GET方法，APIG（API网关）将会拦截请求。

处理方法

使用POST方法调用。

5.3.8 请求超时返回 Timeout

问题现象

服务预测请求超时，报错{"error_code": "ModelArts.4205","error_msg":"Connection time out."}

原因分析

请求超时，大概率是APIG（API网关）拦截问题。需排查APIG（API网关）和模型。

处理方法

1. 优先排查APIG（API网关）是否是通的，可以在本地使用curl命令排查，命令行：**curl -kv {预测地址}**。如返回Timeout则需排查本地防火墙，代理和网络配置。
2. 检查模型是否启动成功或者模型处理单个消息的时长。因APIG（API网关）的限制，模型单次预测的时间不能超过40S，超过后系统会默认返回Timeout错误。

5.3.9 自定义镜像导入模型部署上线调用 API 报错

部署上线调用API报错，排查项如下：

1. 确认配置文件模型的接口定义中有没有POST方法。
2. 确认配置文件里url是否有定义路径。例如：“/predictions/poetry”（默认为“/”）。
3. 确认API调用中body体中的调用路径是否拼接自定义路径。如：“{API接口地址}/predictions/poetry”。

5.3.10 在线服务预测报错 DL.0105

问题现象

在线服务预测报错DL.0105，报错日志：“TypeError:‘float’object is not subscriptable”。

原因分析

根据报错日志分析，是因为一个float数据被当做对象下标访问了。

处理方法

将模型推理代码中的x[0][i]修改为x[i]，重新部署服务进行预测。

6 MoXing

6.1 使用 MoXing 复制数据报错

问题现象

1. 调用**moxing.file.copy_parallel()**将文件从开发环境的OBS桶中复制到其他OBS桶里，但是桶内没有出现目标文件。
2. 使用MoXing复制数据不成功，出现报错。如：
   - ModelArts开发环境使用MoXing复制OBS数据报错：keyError: 'request-id'
   - ModelArts使用MoXing复制报错：No files to copy
   - socket.gaierror: [Errno -2] Name or service not known
   - ERROR:root:Failed to call:

     func=<bound method ObsClient.getObject of <obs.client.ObsClient object at 0x7fd705939710>>

     args=('bucket', 'data/TFRecord/HY_all_inside/no_adjust_light_3/09_06_6x128x128_0000000212.tfrecord')
3. 使用MoXing复制数据报错，提示超时。如：
   - 报错：TimeoutError: [Errno 110] Connection timed out
   - WARNING:root:Retry=9,Wait=0.1, Timestamp = 1567152567.5327423

原因分析

当使用MoXing复制数据不成功，可能原因如下：

- 源文件不存在。
- OBS路径不正确或者是两个OBS路径不在同一个区域。
- 训练作业空间不足。

处理方法

按照报错提示，需要排查以下几个问题：

1. 检查**moxing.file.copy_parallel()**的第一个参数中是否有文件，否则会出现报错：No files to copy

- 文件存在，请执行2。
- 文件不存在，请忽略该报错继续执行后续操作。

2. 检查复制的OBS的路径是否与开发环境或训练作业在同一个区域。

   进入ModelArts管理控制台，查看其所在区域。然后再进入OBS管理控制台，查看您使用的OBS桶所在的区域。查看是否在同一区域。

   - 是，请执行3。
   - 否，请在ModelArts同一区域的OBS中新建桶和文件夹，并将所需的数据上传至此OBS桶中。

3. 检查OBS的路径是否正确，是否写为了“obs://xxx”。可使用如下方式判断OBS路径是否存在。

   mox.file.exists('obs://bucket_name/sub_dir_0/sub_dir_1')

   - 路径存在，请执行4。
   - 路径不存在，请在更换为一个可用的OBS路径。

4. 检查使用的资源是否为CPU，CPU的“/cache”与代码目录共用10G，可能是空间不足导致，可在代码中使用如下命令查看磁盘大小。

   os.system('df -hT')

   - 磁盘空间满足，请执行5。
   - 磁盘空间不足，请您使用GPU资源。

5. 如果是在Notebook使用MoXing复制数据不成功，可以在Terminal界面中使用**df -hT**命令查看空间大小，排查是否因空间不足导致，可在创建Notebook时使用EVS挂载。

如果代码写作正确，仍然无法解决该问题，请提交工单，由专业工程师为您分析并解决问题。

6.2 如何关闭 Mox 的 warmup

问题现象

训练作业mox的Tensorflow版本在运行的时候，会先执行“50steps” 4次，然后才会开始正式运行。

warmup即先用一个小的学习率训练几个epoch（warmup），由于网络的参数是随机初始化的，如果一开始就采用较大的学习率会出现数值不稳定的问题，这是使用warmup的原因。等到训练过程基本稳定之后就可以使用原先设定的初始学习率进行训练。

原因分析

Tensorflow分布式有多种执行模式，mox会通过4次执行50 step记录执行时间，选择执行时间最少的模型。

处理方法

创建训练作业时，在“运行参数”中增加参数“variable_update=parameter_server”来关闭Mox的warmup。

6.3 Pytorch Mox 日志反复输出

问题现象

ModelArts训练作业算法来源选用常用框架的Pytorch引擎，在训练作业运行时Pytorch Mox日志会每个epoch都打印Mox版本，具体日志如下：

```
INFO:root:Using MoXing-v1.13.0-de803ac9
INFO:root:Using OBS-Python-SDK-3.1.2
INFO:root:Using MoXing-v1.13.0-de803ac9
INFO:root:Using OBS-Python-SDK-3.1.2
```

原因分析

Pytorch通过spawn模式创建了多个进程，每个进程会调用多进程方式使用Mox下载数据。此时子进程会不断销毁重建，Mox也就会不断的被导入，导致打印很多Mox的版本信息。

处理方法

为避免训练作业Pytorch Mox日志反复输出的问题，需要您在“启动文件”中添加如下代码，当“MOX_SILENT_MODE = “1””时，可在日志中屏蔽mox的版本信息：

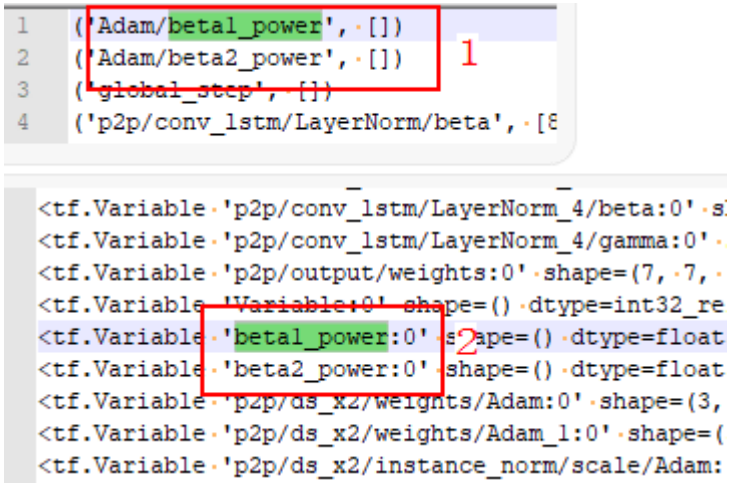
```
import os
os.environ["MOX_SILENT_MODE"] = "1"
```

6.4 moxing.tensorflow 是否包含整个 TensorFlow，如何对生成的 checkpoint 进行本地 Fine Tune?

问题现象

使用MoXing训练模型，“global_step”放在Adam名称范围下，而非MoXing代码中没有Adam名称范围，如图6-1所示。其中1为使用MoXing代码，2代表非MoXing代码。

图 6-1 代码示例



处理方法

Fine Tune就是用别人训练好的模型，加上自己的数据，来训练新的模型。相当于使用别人的模型的前几层，来提取浅层特征，然后在最后再落入自己的分类中。

由于一般新训练模型准确率都会从很低的值开始慢慢上升，但是Fine Tune能够在比较少的迭代次数之后得到一个比较好的效果。Fine Tune的好处在于不用完全重新训练模型，从而提高效率，在数据量不是很大的情况下，Fine Tune会是一个比较好的选择。

moxing.tensorflow包含所有的接口，对TensorFlow做了优化，里面的实际接口还是TensorFlow的原生接口。

当非MoXing代码中没有Adam名称范围时，需要修改非MoXing代码，在其中增加如下内容：

```
with tf.variable_scope("Adam"):
```

在增加代码时不建议使用自定义“global_step”，推荐使用**tf.train.get_or_create_global_step()**。

6.5 训练作业使用 MoXing 复制数据较慢，重复打印日志

问题现象

- ModelArts训练作业使用MoXing复制数据较慢。
- 重复打印日志“INFO:root:Listing OBS”。

原因分析

1. 复制数据慢的可能原因如下：
   - 直接从OBS上读数据会造成读数据变成训练的瓶颈，导致迭代缓慢。
   - 由于环境或网络问题，读OBS时遇到读取数据失败情况，从而导致整个作业失败。
2. 重复打印日志，该日志表示正在读取远端存在的文件，当文件列表读取完成以后，开始下载数据。如果文件比较多，那么该过程会消耗较长时间。

处理方法

在创建训练作业时，数据可以保存到OBS上。不建议使用TensorFlow、MXNet、PyTorch的OBS接口直接从OBS上读取数据。

- 如果文件较小，可以将OBS上的数据保存成“.tar”包。训练开始时从OBS上下载到“/cache”目录，解压以后使用。
- 如果文件较大，可以保存成多个“.tar”包，在入口脚本中调用多进程进行并行解压数据。不建议把散文件保存到OBS上，这样会导致下载数据很慢。
- 在训练作业中，使用如下代码进行“.tar”包解压：

```
import moxing as mox
import os
mox.file.copy_parallel("obs://donotdel-modelarts-test/AI/data/PyTorch-1.0.1/tiny-imagenet-200.tar", '/cache/tiny-imagenet-200.tar')
os.system('cd /cache; tar -xvf tiny-imagenet-200.tar > /dev/null 2>&1')
```


6.6 MoXing 如何访问文件夹并使用 get_size 读取文件夹大小?

问题现象

- 使用MoXing无法访问文件夹。
- 使用MoXing的“get_size”读取文件夹大小，显示为0。

原因分析

使用MoXing访问文件夹，需添加参数：“recursive=True”，默认为False。

处理方法

获取一个OBS文件夹的大小：

```
import moxing as mox
mox.file.get_size('obs://bucket_name/sub_dir_0/sub_dir_1', recursive=True)
```

获取一个OBS文件的大小：

```
import moxing as mox
mox.file.get_size('obs://bucket_name/obs_file.txt')
```

7 API/SDK

7.1 安装 ModelArts SDK 报错“ERROR: Could not install packages due to an OSError”

问题现象

安装ModelArts SDK报错，完整报错信息“ERROR: Could not install packages due to an OSError: [WinError 2] The system cannot find the file specified: 'c:\python39\Scripts\ephemeral-port-reserve.exe' -> 'c:\python39\Scripts\ephemeral-port-reserve.exe.deleteme ”。

原因分析

用户使用权限问题导致。

处理方法

用户电脑切换到管理员角色，键盘快捷键（Windows+R模式）并输入cmd，进入黑色窗口，执行如下命令：

```
python -m pip install --upgrade pip
```

7.2 ModelArts SDK 下载文件目标路径设置为文件名，部署服务时报错

问题现象

ModelArts SDK在OBS下载文件时，目标路径设置为文件名，在本地IDE运行不报错，部署为在线服务时报错。

代码如下：

```
session.obs.download_file（obs_path, local_path）
```

报错信息如下：


```
2022-07-06 16:22:36 CST [ThreadPoolEx] - /home/work/predict/model/customize_service.py[line:184] -
WARNING: 4 try: IsADirectoryError(21, 'Is a directory'). update products failed!
```

原因分析

用户代码中设置的目标路径（local_path）有误。

处理方法

需要将local_path路径设置为文件夹且后缀必须以“/”结尾。

7.3 调用 API 创建训练作业，训练作业异常

问题现象

调用API接口创建训练作业（专属资源池为CPU规格），训练作业状态由“创建中”转变为“异常”，训练作业详情界面“规格信息”为“--”。

原因分析

调用接口传入了CPU规格的专属资源池不支持的参数。

处理步骤

检查API请求的请求体中是否有“flavor_id”参数，CPU规格的专属资源池不支持使用“flavor_id”参数。

7.4 用户执行 huaweicloud.com 相关 API 超时

问题现象

用户在Notebook里通过request请求接口时超时：GET pangu-xxx.cn-southwest-2.myhuaweicloud.com。

原因分析

在Notebook中访问公网需要通过代理，访问huawei.com不通过公网代理，huaweicloud.com域名在no_proxy/NO_PROXY中包含，就访问不了。

解决方式

执行以下命令查看在no_proxy/NO_PROXY中是否包含huaweicloud.com域名。

```
env | grep -i no_proxy
```

如果包含，请重新设置，或者直接去掉相关环境变量。

方式一: 重新设置

```
export no_proxy=xxx
export NO_PROXY=xxx
```

方式二: 删掉相关环境变量


```
unset no_proxy
unset NO_PROXY
```


8 资源池

8.1 创建资源池失败

资源配额限制

在使用专属资源池时（如资源扩缩容、创建VPC、创建VPC-子网、打通VPC），如果提示相关资源配额受限，请**提交工单**处理。

创建失败/变更失败

1. 登录ModelArts管理控制台，在左侧导航栏中选择“AI专属资源池 > 弹性集群 Cluster”，进入“弹性集群 Cluster”页面。
2. 您可以通过单击“购买AI专属集群”右侧的“操作记录”，查看当前处于失败状态的资源池信息。

图 8-1 创建失败资源池信息

3. 鼠标悬停在“状态”列的上，即可看到该操作失败的具体原因。

说明

失败的记录默认按照操作的申请时间排序，最多显示500条并保留3天。

8.2 Standard 资源池节点故障定位

节点故障定位

对于Standard资源池，ModelArts平台在识别到节点故障后，通过给K8S节点增加污点的方式（taint）将节点隔离避免新作业调度到该节点而受到影响，并且使本次作业不受污点影响。当前可识别的故障类型如下，可通过隔离码及对应检测方法定位故障。

表 8-1 隔离码

隔离码	分类	子类	异常中文描述	检测方法
A05 0101	GPU	显存	GPU ECC错误。	通过nvidia-smi -a查询到存在Pending Page Blacklist为Yes的记录，或多比特Register File大于0。对于Ampere架构的GPU，存在以下场景： • 存在不可纠正的SRAM错误。 • 存在Remapping Failure记录。 • dmsg中存在Xid 95事件。 （参考**NVIDIA GPU Memory Error Management**） Ampere架构GPU显存错误分级： • L1: 可纠正ECC错误（单比特ECC错误），不影响业务。观测方式：nvidia-smi -a中查询到Volatile Correctable记录。 • L2: 不可纠正ECC错误（多比特ECC错误），当次业务受损，重启进程可恢复。观测方式：nvidia-smi -a中查询到Volatile Uncorrectable记录。 • L3: 错误未被抑制，可能影响后续业务，需要重置卡或重启节点。观测方式：Xid事件中包含95事件。(Remapped的Pending记录只作为提示，当业务空闲时进行卡重置触发重映射即可) • L4: 需要换卡，SRAM Uncorrectable>4或者Remapped Failed。
A05 0102	GPU	其他	nvidia-smi返回信息中包含ERR。	通过nvidia-smi -a查询到ERR!，通常为硬件问题，如电源风扇等问题。
A05 0103	GPU	其他	nvidia-smi执行错误，超时或者不存在。	执行nvidia-smi退出码非0。

隔离码	分类	子类	异常中文描述	检测方法
A05 0104	GPU	显存	ECC错误到达64次。	通过nvidia-smi -a查询到Retired Pages中，Single Bit和Double Bit之和大于64。
A05 0148	GPU	其他	infoROM告警。	执行nvidia-smi的返回信息中包含“infoROM is corrupted”告警。
A05 0109	GPU	其他	GPU其他错误。	检测到的其他GPU错误，通常为硬件问题，请联系技术人员支持。
A05 0147	IB	链路	IB网卡异常。	ibstat查看网卡非Active状态。
A05 0121	NPU	其他	npu dcmi接口检测到driver异常。	NPU驱动环境异常。
A05 0122	NPU	其他	npu dcmi device异常。	NPU设备异常，昇腾dcmi接口中返回设备存在重要或紧急告警。
A05 0123	NPU	链路	npu dcmi net异常。	NPU网络链接异常。
A05 0129	NPU	其他	NPU其他错误。	检测到的其他NPU错误，通常为不可自纠正的异常，请联系技术人员支持。
A05 0149	NPU	链路	hccn tool网口闪断检查。	NPU网络不稳定，存在闪断情况。通过“hccn_tool-i ${device_id} -link_stat -g”查看24小时内闪断5次以上。
A05 0951	NPU	显存	NPU ECC次数达到维修阈值。	NPU的HBM Double Bit Isolated Pages Count值大于等于64。
A05 0146	Runtime	其他	ntp异常。	ntpd或者chronyd服务异常。
A05 0202	Runtime	其他	节点NotReady。	节点不可达，k8sNode存在以下污点之一： ● node.kubernetes.io/unreachable ● node.kubernetes.io/not-ready
A05 0203	Runtime	掉卡	AI正常卡数和实际容量不匹配。	检测到存在GPU或NPU掉卡情况。
A05 0206	Runtime	其他	Kubelet硬盘只读。	“/mnt/paas/kubernetes/kubelet”目录为只读状态。
A05 0801	节点管理	节点运维	资源预留。	节点被标记为备机，并具有备机污点。
A05 0802	节点管理	节点运维	未知错误。	节点被标记为具有未知故障污点。

隔离码	分类	子类	异常中文描述	检测方法
A200001	节点管理	驱动升级	GPU升级。	节点正在执行GPU驱动升级。
A200002	节点管理	驱动升级	NPU升级。	节点正在执行NPU驱动升级。
A200008	节点管理	节点准入	准入检测。	节点正在进行节点准入检测，包括基本的节点配置检查和简单的业务验证。
A050933	节点管理	容错Failover	当节点具有该污点时，会将节点上容错（Failover）业务迁移走。	当节点标记该污点时，会将节点上容错（Failover）业务迁移走。
A050931	训练toolkit	预检容器	训练预检容器检测到GPU错误。	训练预检容器检测到GPU错误。
A050932	训练toolkit	预检容器	训练预检容器检测IB错误。	训练预检容器检测IB错误。